SYBASE iAnywhere

Application Developer Guide for M-Business Anywhere™

Published: January 2008

Part number: DC00304-01-0670-02

Copyright and trademarks

iAnywhere Solutions, Inc. is a subsidiary of Sybase, Inc.

This document is not a commitment on the part of iAnywhere to do or refrain from any activity, and iAnywhere may change the content of this document at its sole discretion without notice. Except as otherwise provided in a written agreement between you and iAnywhere, this document is provided “as is”, and iAnywhere assumes no liability for its use or any inaccuracies it may contain.

Contents

About this guide

Focus of this guide

This guide provides the detailed implementation information that developers working with M-Business Anywhere™ need to smoothly implement an efficient M-Business channel to serve a target audience. It provides guidance on creation of handheld-friendly channel content with basic HTML. It does not cover details programming in JavaScript or C/PODS.

You should read this guide if you are creating, administering, or trouble-shooting an M-Business channel.

For your convenience, different versions of this document are provided for Enterprise and consumer service users. This is the Enterprise version. Both versions are available in HTML and PDF formats.

You can access both formats of the Enterprise version from the M-Business Anywhere product manuals page: (http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere/).

Conventions

The following table lists the formatting conventions used throughout this guide.

Table 1. Formatting conventions

Item	Treatment	Example
Name of publication	Italic	*Administrator Guide for M-Business Server*
Items on which user is to take an action	Bold	Click the **Reset** button.
Multi-level menu selections	Bold with "»" separator	Choose **Start»Settings» Control Panel**.
Text you type	Bold, fixed width font	Type **`Admin`** in this field.
Text displayed in a file or on the screen	Fixed width font	The screen reads: `Backup Complete`
Keyboard key	Angle brackets	<Enter>
File names and paths	Italic	*.../conf/sync.conf*
Literals in code synopsis	Bold	void **PODSaddRef(**PODSObject* *podsobj***);**
Variables in code synopsis	Italic	void **PODSaddRef(**PODSObject* *podsobj***);**
Variables in text	Angle brackets plus italic	http://<*servername*>:<*port*>

The M-Business Anywhere documentation set

In addition to this document, there are several other iAnywhere Solutions publications available that you may find useful in setting up and using M-Business Server.

Note
Unless otherwise noted, all of these publications are available from: http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere/
In order for links between different PDF files to work correctly, you must open the files directly from the web site, or download them from the web site into the same local directory.

- *Developer Quick Start Guide for M-Business Anywhere*
- *Release Notes for M-Business Anywhere*
- *User Guide for M-Business Anywhere Client*
- *M-Business Anywhere, an Introduction*
- *Application Developer Guide for M-Business Anywhere*
- *API Reference for M-Business Anywhere*
- *Ensuring Mobile Security from the Device to the Datacenter*, available from http://www.ianywhere.com/whitepapers/ensuring_security.html

Related publications

UltraLite for M-Business Anywhere

For more information about using UltraLite® for M-Business Anywhere, please refer to the following iAnywhere Solutions™ publications:

- *UltraLite for M-Business Anywhere Quick Start*, available from http://www.ianywhere.com/developer/product_manuals/sqlanywhere/1001/en/html/ulagen10/ag-preparing-evb-development.html
- *Exploring the CustDB Samples for UltraLite* http://www.ianywhere.com/developer/product_manuals/sqlanywhere/1001/en/html/ulfoen10/fo-fo-custdb.html
- *UltraLite M-Business Anywhere Programming, version 10*, available from http://www.ianywhere.com/developer/product_manuals/sqlanywhere/1000/en/pdf/ulagen10.pdf

Adaptive Server Anywhere

For more information about Adaptive Server® Anywhere, please refer to the following iAnywhere Solutions publications.

- *SQL Anywhere 10* documentation, available from http://www.ianywhere.com/developer/product_manuals/sqlanywhere/1001/en/html/index.html
- *Setting Up Adaptive Server Anywhere as a Cluster Database Service*, available from http://www.ianywhere.com/developer/technotes/asa_cluster_db_service.html

Web servers

In addition to the iAnywhere documentation, you may want to refer to the following if you need comprehensive information on configuration settings for Apache and Tomcat servers:

- *Apache webserver* documentation, available from http://httpd.apache.org/docs/mod/mod_log_config.html
- *Tomcat 4 Servlet/JSP Container* documentation, available from http://jakarta.apache.org/tomcat/tomcat-4.1-doc/index.html

If the documentation has moved, go to the Apache Jakarta project URL below and search on tomcat 4 servlet documentation to find its new location: http://jakarta.apache.org/

Recommended references

In addition to the related publications from iAnywhere Solutions, you may want to consult references on the standards and third party software that are incorporated in the M-Business Anywhere architecture.

Note
This guide documents in detail only those features of JavaScript, DOM, and CSS that differ from the published standards.

If you already have a favorite reference on any of these topics, there is no need to seek out another. However, if you need additional reference material and do not already have it at hand, the following list of sources is a good place to start.

Cascading Style Sheets (CSS)

W3C Cascading Style Sheets home page.

HTML: http://www.w3.org/Style/CSS/

A comprehensive reference on CSS maintained by the W3C.

Document Object Model (DOM)

W3C Document Object Model (DOM) Level 1 Specification (Second Edition)

PDF: http://www.w3.org/TR/2000/WD-DOM-Level-1-20000929/DOM.pdf plain text: http://www.w3.org/TR/2000/WD-DOM-Level-1-20000929/DOM.txt

A basic reference on the DOM spec.

DOM Tutorial

HTML: http://www.w3schools.com/dom/default.asp

A good introduction to DOM maintained by W3Schools.

Dynamic HTML (DHTML)

Introduction to Dynamic HTML.

HTML: http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/dhtml.asp

The Microsoft Developer Network's tutorial/reference on DHTML.

HTML

Dynamic HTML, the Definitive Reference

Author: Danny Goodman Publisher: O'Reilly, August 1998

Amazon.com online description: http://www.amazon.com/exec/obidos/tg/feature/-/6779/103-8587514-0449445

A good source of examples, with tutorials, organized more for learning than for reference; content is targeted at desktop browsers.

Java

Designing Enterprise Applications with the J2EE™ Platform, Second Edition

HTML: http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/DEA2eTOC.html

Describes designing Web applications with J2EE, which is a common platform you can use/adapt to producing M-Business Anywhere applications.

JavaScript

Netscape DevEdge JavaScript Central

HTML: http://devedge-temp.mozilla.org/central/javascript/index_en.html

A basic reference on JavaScript; free online. If site has moved and does not redirect to the new URL, go to http://www.mozilla.org/ and search on "JavaScript" there; the permanent home is slated to be somewhere on the mozilla.org site.

JavaScript Bible, 5th Edition

Authors: Danny Goodman and Michael Morrison

A more current popular basic reference on JavaScript, containing numerous coding examples.

Extended HTML (XHTML)

XHTML™ 1.0: The Extensible HyperText Markup Language

HTML: http://www.w3.org/TR/xhtml1/

A comprehensive reference on XHTML maintained by the W3C.

XHTML Tutorial

HTML: http://www.w3schools.com/xhtml/default.asp

A good introduction to XHTML maintained by W3Schools.

Contacting iAnywhere Solutions

Technical support

If you need assistance using iAnywhere software, in North America, please contact iAnywhere Technical Support by calling 1-800-8SYBASE (800-879-2273) and then selecting option 3. You can call Monday through Friday (except major US holidays) between 9:00 a.m. and 9:00 p.m. Eastern time. Services will be provided in accordance with your support agreement.

Outside of North America, for your local support number and hours, please see: http://www.sybase.com/contactus/support

Registering as a Named Contact

Calling the 800-number during business hours should always work to get you technical support — a Customer Number is created for you as soon as your purchase is completed. You will find it faster and easier to get technical support, by phone or online, if you have registered as a Named Contact.

When you purchase an iAnywhere product, a *Sybase Technical Support Contact Form* will automatically be emailed to you within 7-10 days. If your company should need to add another Named Contact, or change the one initially registered, call the Technical Support 800-number and request a *Sybase Technical Support Contact Change Form*.

The *Sybase Technical Support Contact Form* will contain your Customer Number, with spaces for you to provide an email address and other identifying information for the Named Contact for your product. Fill in the requested information and fax the form back to the phone number indicated.

When your fax is received, an email will be sent to you, providing your Technical Contact ID number. You can then use this number to speed up the process when you call for technical support, and to access technical support online.

Using the Sybase Online Support Services

A major benefit of using the Sybase Online Support Services is 24x7 availability. Online support also allows you to look up and review past and current support issues.

When you register as a Named Contact, the email sent to you with your Technical Contact ID number also contains instructions for registering and using the Sybase Online Support Services. Follow these instructions to register as a first-time user, or to update your account with information for the new product you have purchased.

If you have any trouble registering for the Sybase Online Support Services, you can of course call iAnywhere Technical Support for assistance!

Application development — customizing iAnywhere software

If you need help with customizing iAnywhere software to better serve your enterprise, please contact iAnywhere Solutions Professional Services at contact_us@ianywhere.com.

Product information

If you need information about other iAnywhere products for your enterprise, please contact iAnywhere Workforce Sales at contact_us@ianywhere.com.

Feedback on documentation

If you have questions or suggestions about this document or other iAnywhere technical publications, please contact iAnywhere Technical Publications at iasdoc@ianywhere.com.

We would like to receive your opinions, suggestions, and feedback on this documentation. Although we do not reply to individual emails, we read all suggestions with interest and attempt to incorporate them in future releases.

Part I. Before You Begin

CHAPTER 1

Introduction to channel development

Contents

About channel development

This document assumes that you are already a competent web developer who is comfortable with desktop web development. That is, you are familiar with the issues involved in developing content for a web site that is designed to be viewed in Microsoft Internet Explorer or Netscape Navigator, running on a desktop PC with a high resolution color monitor and a live Internet connection.

The goal of this introductory chapter is to orient you to the different design and management issues that you face in developing an M-Business channel.

The rest of the chapters in this document detail the solutions that are available to you in meeting these design challenges.

If you are really in a hurry to get your channel up and running, you should at least skim the section in this chapter titled "What is an M-Business channel?" on page 5.

What is an M-Business channel?

In its most basic form, an M-Business channel is just another web site.

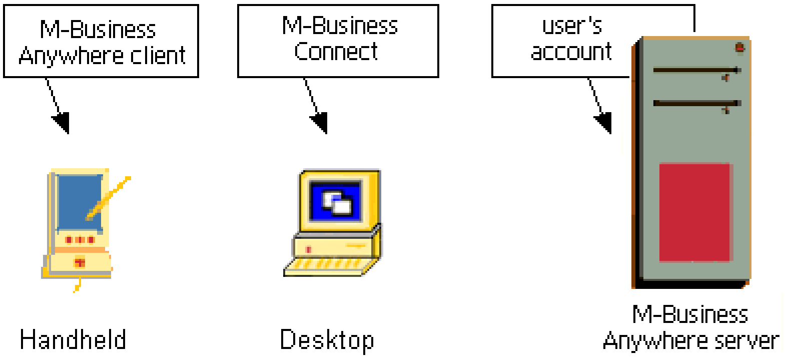
There is more to it than that, of course. An M-Business channel page does not contain all the bells and whistles you might see on, say, a web page in Internet Explorer 5.x. But if you know how to create a web page, you can make yourself an M-Business channel.

As a simple introduction, look at what happens when you subscribe to (and subsequently download) an M-Business channel. Some elements have been simplified for the sake of brevity, but you will get a basic understanding of the process.

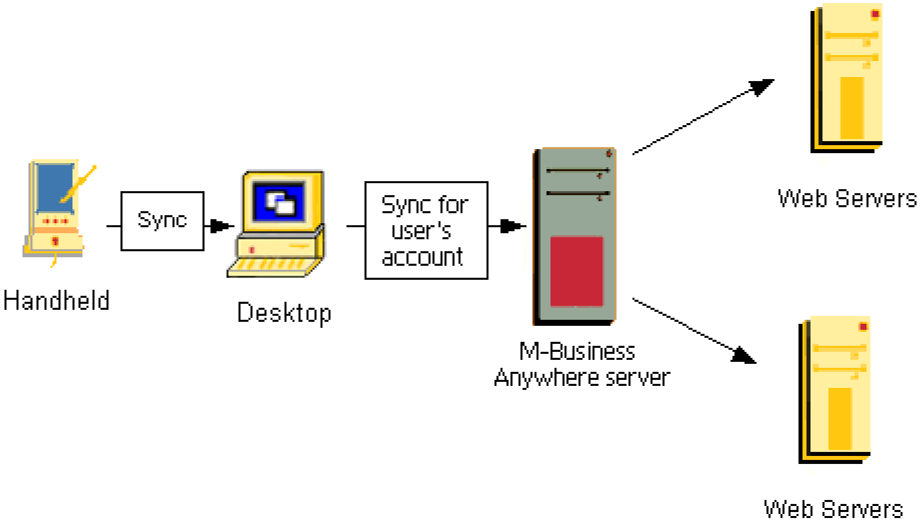
The first thing you need to do is open up and configure an account on the M-Business Sync Server. This account contains certain relevant information: your user name and password, what channels you want to subscribe to, and so on.

After that, you install the software necessary to view M-Business channel. This includes:

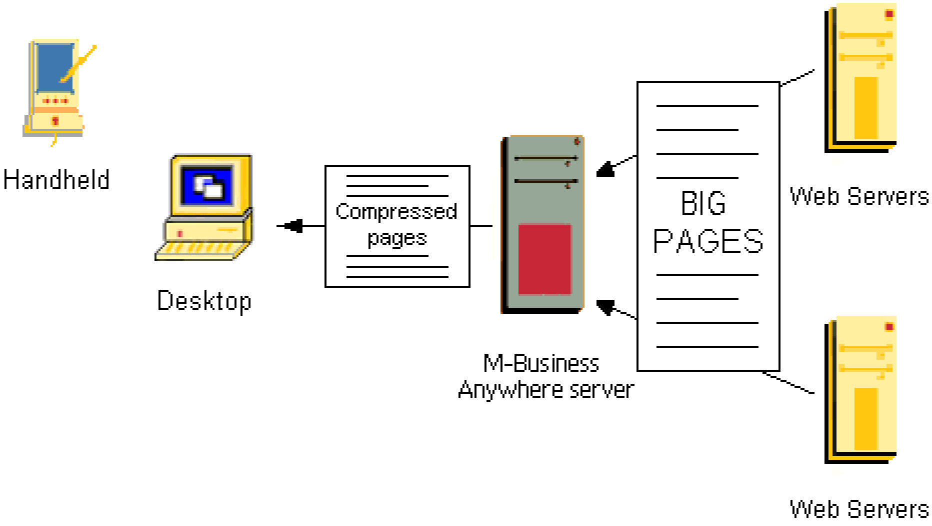
- **M-Business Client**. This is a web browser (and a mini web server) that is installed on your Palm OS, Window Mobile Pocket PC, Windows XP, Windows Mobile 5, or Microsoft Smartphone device. Some information about your account is also stored here.
- **M-Business Connect**. This is installed on your desktop computer. It is the software that enables you to connect to your M-Business Sync Server whenever you synchronize your device.



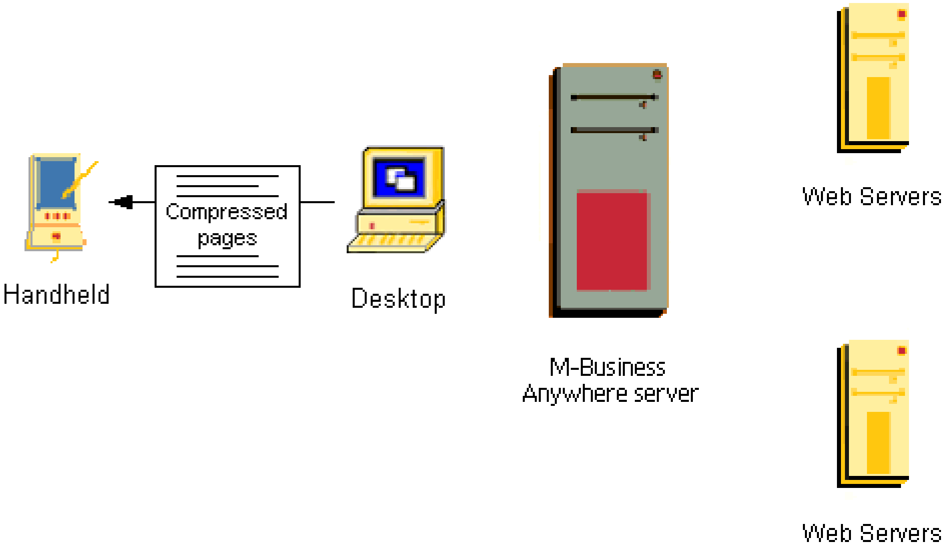
When you synchronize your mobile device, Mobile Link will take over and connect to your M-Business Sync Server. Your M-Business Sync Server, after looking up what channels you are subscribed to, will download those pages from sites on the Internet. In many cases, these sites are distinct areas that contain pages optimized specifically for M-Business channels.



Your M-Business Sync Server downloads all these pages and performs some pre-processing on them. This includes shrinking images too large for the mobile device's screen, discarding pieces that cannot be used by M-Business Client (such as Java applets) and compressing the rest of the HTML.



Once that is done, these pages are uploaded to your mobile device, and M-Business Connect disconnects. All the pages from a channel are uploaded at once, so that you can view the entire channel without having to re-connect to the Internet.



At this point, you can take your mobile device with you and browse the channel at your leisure. A mini web server runs on the device and displays pages whenever you click a link, and the M-Business Client browser reads and formats the pages just like any other HTML browser.

Unlike most other browsers, M-Business Client allows you to submit forms while you are offline, and continue browsing (or submitting more forms), without having to wait for a response from a server.

M-Business Anywhere development vs. desktop web development

This section describes the important design and management issues that you will encounter in M-Business channel development that you probably have not had to deal with in desktop web development.

Topics in this section:

- "Hardware differences" on page 8
- " M-Business Anywhere limitations" on page 8
- "Offline differences" on page 9

Hardware differences

The hardware differences in mobile devices, as compared with current desktop PCs, pose a number of web design challenges beyond what you are used to dealing with in desktop web development.

- **Smaller screens**: The usable area is only 150 by 150 pixels on a typical Palm OS device, and not a whole lot larger (177 by 208 pixels) on a Nokia 3650 smart phone or a typical Pocket PC device (215 by 270 pixels). A desktop PC screen is typically in the neighborhood of 1024 by 768 pixels. Large graphics and wide tables are out. See "Designing pages for the channel web site" on page 58 and "Designing graphics for channel pages" on page 69.
- **Big differences in platforms**: The hardware differences between various mobile devices are much greater than the differences you are used to dealing with between IE vs. Netscape, or PC vs. Mac. This makes it much harder to use a lowest common denominator approach, in which a single set of web pages with a fixed standard for graphics will look OK on all platforms. See "Customizing content for users and devices" on page 159.
- **Limited memory**: Some users will turn off graphics to get more content onto their devices and users may remove your channel if it takes up too much memory on a device. See "Designing the channel web site" on page 51.

M-Business Anywhere limitations

In order to optimize use of the limited resources on mobile devices, M-Business Client and your M-Business Sync Server place additional limitations on the web content that can be served through your channel.

- **Limited subset of HTML** — Frames, style sheets, image maps, and a quite a few other features you may be used to using are not supported. See Appendix "HTML 4 support in M-Business Client 5.x and later" on page 295.
- **No fancy plug-ins** — Plug-ins, such as Macromedia Flash, are not currently supported.

- **Image size limits** — If a channel page contains a graphic that is larger than 150x150 pixels, your M-Business Sync Server may scale it down before downloading it. For ways to get around image size limits, see "Breaking maximum image size limits" on page 75.
- **Channel size limit** — You must specify a Maximum Channel Size in your channel setup — your M-Business Sync Server stops downloading pages once that size is reached. See "Size limit issues" on page 52.
- **Cookies stored on server** — With an M-Business channel the cookies are stored on the server so that they do not have to be uploaded at the start of every synchronization. This means that you have to use different methods to inspect cookies when you are troubleshooting your channel. See "Not sure cookie is being set properly in your channel" on page 46 and "Tips for debugging cookies" on page 46.
- **JavaScript compiled on server** — Popular browsers running on the desktop compile JavaScript code as it is executed, but your M-Business Sync Server compiles your JavaScript before downloading channel pages to a user's device. Thus you cannot do things that depend on JavaScript being compiled within the browser, such as having JavaScript commands create additional JavaScript.

Offline differences

The majority of mobile device users are currently operating in an offline mode, without a wireless connection. These users periodically doc and synchronize their mobile devices with a desktop PC that has an Internet connection, then they work offline for extended periods of time. Even those users who do have a wireless connection generally tend to connect only periodically to synchronize, and then disconnect and work offline.

- **Offline browsing** — Relatively few mobile device users currently have a wireless connection, and many of those who do just connect via modem, synchronize their devices, and then disconnect and work offline.

  This means that you must design a fairly tight chunk of pages to download with the home page. You determine which pages are downloaded from your site when a user synchronizes your channel by specifying a setting for *Link Depth* — the number of links from your home page within which your pages are to be downloaded.

  You can also specify whether pages within that link depth, but outside of your web site, are to be downloaded in a synchronization. Also, it is very important to avoid links to outside pages that are not optimized for mobile devices.

  If you don't plan carefully, important pages may not be downloaded — either because they are beyond the Link Depth, or because the total size of pages exceeds the maximum size set for your channel. See "Size limit issues" on page 52.
- **Offline form submissions** — You must ensure that any forms used in your channel web site do not require the user to wait for a response from the server after the form is submitted. Forms submitted offline are queued for submission to the server in the next synchronization. If it is absolutely necessary, you can prevent an offline user from submitting the same form again before synchronizing and getting back the server's response page.

In addition, you will probably want to set up special handling of form submissions to ensure that the user understands what has happened to the submitted form and can easily find the response page after synchronizing. See “Managing channel form submissions” on page 173.

Date and time information in M-Business channels

M-Business Anywhere derives all date and time information from external sources:

- The operating system where M-Business Server is installed provides date and time information for all user synchronizations and for all transactions updating the M-Business Anywhere database.
- Web servers supplying content for M-Business channels provide date and time information for requested channel pages that M-Business Server uses to manage caching of those pages.
- Applications that supply database information to web pages that are part of M-Business channels control any date and time information that is included in channel pages.
- Applications running in M-Business Client, on mobile devices to which M-Business Server downloads M-Business channel content, control date and time information when users enter or update application data on the mobile device — for a variety of reasons, the application should store the date/time at which the information is uploaded to the application server, in place of any date/time information available from the device OS.

Recommendations

If precise date and time information is critical in your M-Business application, iAnywhere recommends the following approach to minimize potential problems.

- **Daylight saving time** — Run all components that support M-Business Anywhere so that they are not subject to daylight saving time; run them on a constant clock, such as Greenwich Mean Time (GMT) or Coordinated Universal Time (UTC).

  You can use one of these standards for date and time values in all components that supply M-Business channel content to M-Business Server, then let the application code that runs in M-Business Client on mobile devices be responsible for conversions to users' local time zones, including adjustments for daylight saving time.
- **Device OS time** — For a variety of reasons, it is best to rely on date/time information provided by the device OS as little as possible.

  When users update information on a mobile device, the date/time information that really counts is the time at which the updated information is synchronized with the backend database. If date/time information for when the user updates information on the device is stored at all, it should be treated as secondary to the date/time that the information is uploaded to the application server in a synchronization.
- **M-Business Server time zone** — The choice of which time zone M-Business Server uses is unfortunately often based on where the company's headquarters are located at the time the server is first created. This causes no problems, as long as the company does not have operations in other time zones.

  For larger companies that has operations in multiple time zones, and for any smaller company that may grow into multiple time zones or move its headquarters to a different time zone, the best approach is to establish a company standard early on that the server runs under UTC and all clients are responsible for translating from UTC to local time.

Channel FAQ

This FAQ is divided into the following sections:

1. Basic questions

1.1 What does an M-Business channel do, exactly?

Basically, when a user synchronizes his or her iAnywhere Solutions account, we go out and grab web pages off of the sites they subscribe to, and send a compressed version of these pages to the user's mobile device where the user can browse them at his or her leisure. You can think of an M-Business channel as a combination of a web spider, mobile device browser, and a translation tool. For a detailed description of how a channel works, see "What is an M-Business channel?" on page 5.

1.2 If I want to create an M-Business channel, does that mean I have to start using M-Business Server to serve my web pages?

No. You continue using the same web server that you have been using.

1.3 So what work do I need to do to create an M-Business channel?

Your job is to basically create a lite version of your web site in HTML that looks good on a small screen, optimized for at least two device types (for example, Palm and Windows Mobile Pocket PC), and serve it from your web server. Your M-Business Sync Server will take care of downloading the pages, compressing them, and displaying them for whatever mobile device the user has. See "Creating your first channel" on page 27, and "Design considerations for channel content" on page 49.

1.4 Does an M-Business channel work with ASP? What about ColdFusion? Vignette? PHP?

An M-Business channel will work with any publishing system you have, as long as it is outputting HTML pages.

1.5 I want to test my M-Business channel, but I do not have a Palm OS or Microsoft OS device. What should I do?

There is a nice emulator available at Palm's web site, although you have to jump through a few legal hoops to get access to their ROMs. And Microsoft now has robust emulators for Windows Mobile 5 and Microsoft

Smartphone platforms. Emulators have many advantages — up to a point. The biggest drawback is that you can never be 100% certain that the emulator is behaving exactly like a physical device at all times.

Our advice would be to convince a friend to lend you their physical device for a week or two. It will give you a better sense of how people will really use your M-Business channel. Or just go buy a mobile device of your own — they are fairly inexpensive these days. (And if you are lucky, you might be able to expense it.)

1.6 How do I design a page that looks nice on a small screen?

Just follow a few simple guidelines about which HTML tags to avoid and how to optimize graphics for different platforms. See "Designing pages for the channel web site" on page 58 and "Designing graphics for channel pages" on page 69.

1.7 How do I preview my channel before I take it live?

You can just create a custom channel using the URL of the channel you are developing. See "Creating your first channel" on page 27.

1.8 How large can my channel be?

You can specify your maximum channel size to be anything you would like, to a reasonable limit. Most channels have a 100KB maximum size by default. While some channels choose to have a maximum size as large as 500KB, we recommend trying to stay closer to the 100-250KB range. See "Size limit issues" on page 52.

1.9 Do you support (insert HTML tag here)?

For a list of the specific HTML tags and JavaScript elements that are available for you to use in your channel pages, see Appendix "HTML 4 support in M-Business Client 5.x and later" on page 295.

1.10 Can I set up permanent channel content on users' devices — content that never expires?

If you have a channel that you expect will have unchanging content over an extended period of time, you can define that channel as Sync Only. This eliminates most of the communication with the web server that is required to determine if the web pages on the device need to be updated.

With connected users, this decreases the time it takes to synchronize the device. With wireless users, it decreases the time it takes to display the page every time the user accesses it. Wireless users are more likely to notice the difference, because M-Business Client does not have to go online to check for an update before displaying Sync Only channel content. See "Understanding channel settings" [*M-Business Anywhere Administrator Guide*].

1.11 Are backslash characters ("\") OK in URLs in channel web pages?

No. Some browsers, such as Internet Explorer, process automatically convert backslash characters in URLs to forward slashes ("/"). M-Business Client does not.

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

1.12 Can I use mailto: and phoneto: in my channels on devices that support an email or phone capability?

Yes. M-Business Client supports use of `mailto:` and `phoneto:` with the `<a>` tag's `href` attribute. See "HTML 4.01 tags and attributes" on page 297.

2. Advanced topics (custom content, managing server load, PODS)

2.1 How can I serve custom content for different devices?

M-Business Client sends up several HTTP headers that include information such as screen size, device OS, and bit depth. The information in these headers is Base64-encoded, so you need to grab our HTTP headers, decode their values, and then use that information to serve up custom content.

Uh... I have no idea what you just said.

Not to worry. Check out "Using M-Business Client HTTP request headers to customize content" on page 161.

2.2 How do I minimize the load on my web server?

Perhaps the best way to minimize your channel's impact on your server is to set it up for caching. The longer your pages and page elements are set to cache, the less often our server will hit yours with requests for fresh content. If you have not already, please see "Caching to improve channel performance" on page 81.

Another way to minimize load on your server is to check your channel pages for links to pages that are not handheld friendly (for example, your main web site). These leaks cause our server to crawl pages that are not optimized for display on mobile devices and to download large images, flash animations, java applets, and other elements that can result in greater load on your server and longer synchronization times for your subscribers. Such leaks may also cause your channel to exceed your maximum channel size.

2.3 What is PODS?

PODS is a custom server. It allows you to create some logic on the device so you can create more of an interactive session on the mobile device without waiting for the user to synchronize.

2.4 Can I write my own PODS for my site?

Yes, but most likely you do not need to. PODS are platform-specific modules compiled from C code. Almost all of the public API is accessible from JavaScript. Since JavaScript is cross-platform, just about the only reason to write C code is to get maximum speed when you really need it, for example when processing very large data sets. For more information about determining when to develop in C vs. JavaScript, see "Choosing a language" [*M-Business Anywhere API Reference*].

2.5 Can I write plug-ins for the M-Business Client browser?

Not at this time. We do not currently have a plug-in system for the M-Business Client browser beyond the PODS infrastructure.

3. M-Business Client wireless

3.1 Do you support M-Business Client wirelessly on the Palm VII?

No. The Palm VII's wireless modem does not use normal TCP/IP. They use a different, semi-proprietary paging network instead (PQA, Web-clipping), which M-Business Client does not support. You can, however, still use the Palm VII in normal cradle-synchronization mode.

3.2 Who is connecting to M-Business Client wirelessly?

Primarily, users who have wireless modems for their mobile devices or connect through a wi-fi network or Bluetooth connection. People with those connections can browse an M-Business channel or the open Internet using wirelessly. If you have a phone with an IR port, it is also possible to use the IR port to connect your mobile device and your phone to use your phone as a makeshift modem.

3.3 What is the breakdown between online and offline users?

The large majority of M-Business Anywhere users are offline users. That is, they are using PDAs without a modem attached. (Or they turn on their modem, synchronize their channels, and then disconnect their modem and read the cached pages.) While you should design primarily for this group, you should still keep M-Business Anywhere online wireless users in mind, as they become a more substantial portion of our user base as time goes on.

Even if you are developing a specific application, for which you expect all your users to have wireless connections, remember that those connections will at times be unavailable for a variety of reasons. So be sure that your all-wireless application behaves nicely when users find themselves offline. The key issue to deal with in order to make your wireless application work smoothly offline is form submissions. For more information on handling offline form submissions, see “Managing channel form submissions” on page 173.

3.4 What is the difference between viewing a page in offline mode and viewing a page wirelessly?

When you synchronize your M-Business channels, whether through your desktop or through your wireless modem, your M-Business Sync Server will still go out and grab all the pages it can, up to its channel size and link depth limit.

The difference comes when you request a page that is *not* available on the device — generally, this is the result of a form submission, but it could also be a link that is past the channel's current link depth. If you are in offline mode, you either get a Your information will be submitted next time you sync, or a This page is not yet available on the device message. If you are in online mode, your M-Business Sync Server will go out on the web and attempt to retrieve the appropriate page in real time.

Generally, you will find that wireless channels have a link depth of 0 or 1, primarily consisting of a search page where users can search for the map / phone number / CD / restaurant they are interested in.

3.5 I am designing a channel that is primarily intended for offline users. Is there anything I should consider for online mode?

Keep in mind that the `submitNoResponse()` function will return the results of a form action in online mode, so you want any resulting page to be somewhat user-friendly. See “Managing channel form submissions” on page 173. Other than that, your channel should work the same.

3.6 So, what strategy should I employ for designing wireless channels?

Most people who intentionally set out to create a wireless-only channel prefer to create one that is separate from their offline channel, or at least has wireless and offline functionality clearly demarcated.

For online wireless users, you can be freer with your interactivity. You can have the entire contents of your site available, with a search function so users can find the products they are interested in. And the entire channel does not need to be downloaded at once.

For offline users, you generally need to upload all of your site's content at once. This involves incorporating more personalization and customized content so you can guess ahead of time what content they will be interested in and placing it on the device.

As a general rule, offline M-Business channels work well if you are viewing them on an online device. But not vice versa.

4. Tracking subscribers

4.1How can I tell how many users are subscribed to my channel?

As long as your channel is a group channel, as opposed to a personal channel, all the members of the group for which the channel was defined are subscribed to the channel. All you have to do is check how many users belong to the group:

- Log into the M-Business Server Administrator Console.
- Click the **Groups** tab.
- Locate the row for your channel's group — the number of users is listed in the **Users** column,

For more detailed instructions, plus a discussion of differences between group and public channels, see “About channels” [*M-Business Anywhere Administrator Guide*].

5. The state of JavaScript

5.1 What is the state of JavaScript on mobile devices?

M-Business Client, version 5.0 and later, supports a major subset of the JavaScript standard — not quite everything you can do in Microsoft's Internet Explorer, but all of the most important features. See “JavaScript” on page 320.

5.2 Can I use JavaScript to do things like putting M-Business Client in offline mode and disabling toolbar buttons?

Yes, as long as your users have version 5.0 or later. To force M-Business Client to work offline, use the following JavaScript:

```
avantgo.disconnect();
```

The following JavaScript shows how to disable individual toolbar buttons:

```
toolbar.buttonForName("back").visible = false;
toolbar.buttonForName("forward").visible = false;
toolbar.buttonForName("reload").visible = false;
```

For more information on JavaScript support in M-Business Client, version 5.0 and later, see “JavaScript” on page 320 and Appendix “M-Business JavaScript engine reference” on page 331.

6. Security

6.1 Do you support SSL?

Yes. SSL was implemented in M-Business Client version 3.3. All current users of M-Business Client can connect securely to web sites.

6.2 Really? How does that work?

For details on how a user sets this up, see "Enabling a secure connection to a server" [*M-Business Anywhere Client User Guide*].

6.3 You say that M-Business Client uses a Certicom Elliptical Encryption Algorithm. I do not think my web server supports that.

No problem — that is just on the device itself. When M-Business Sync Server talk to your web server, we use the normal RSA encryption that your server has been dealing with for years.

6.4 What do I do if I want pages on my M-Business channel to be secure?

You would create them the same way you would on your web site. Keep in mind, however, that only users who have manually turned on Connect Securely in their M-Business Client preferences will be able to view secure pages.

6.5 How can I tell if a user is or is not connecting securely?

You can look at their HTTP headers for the presence or lack of an `X-AvantGo-SecureSync` header. See "Using M-Business Client HTTP request headers to customize content" on page 161 for more information on how to do this.

6.6 I am creating a channel and I want to restrict it to only certain users. How can I do this?

Make sure that you create a group channel, not a public channel. Group channels are automatically restricted to members of the group for which they are created.

First of all, create a managed group for your channel subscribers. See "Creating a group" [*M-Business Anywhere Administrator Guide*].

Then add to that group the users you want to have subscribed to your channel. See "Managing users for a group" [*M-Business Anywhere Administrator Guide*].

As long as you do not make the channel available to other groups, it will be restricted to the members of the group for which it was defined.

6.7 Can I password protect my channel content in M-Business Client on users' devices?

Yes, but the client password protection must be enabled globally on the M-Business Server where your channel is based. For instructions on implementing client password protection, see "Understanding channel settings" [*M-Business Anywhere Administrator Guide*].

Note
You can also password protect individual channels. This would be a good approach if there are only one or two channels which you want to be secure on your users' devices. To password protect an individual channel, you would use cookies. See "Using cookies to support personal channels" on page 196.

7. Other

7.1 I have a question that is not listed here!

First, check out the rest of the M-Business Anywhere documentation set, referenced in "The M-Business Anywhere documentation set" on page x.

If you still cannot find what you are looking for, please contact iAnywhere at 800-8SYBASE (800-879-2273); services will be provided in accordance with your support agreement.

If you need help with customizing iAnywhere software to better serve your enterprise, please contact iAnywhere Solutions Professional Services at contact_us@ianywhere.com.

The M-Business Anywhere channel creation checklist

You can use the M-Business Anywhere channel creation checklist to help you create your M-Business channel.

- **If you don't have a mobile device that can run M-Business Client, talk to your manager about expensing one.**

  You're really not going to get a good sense of how M-Business Client works without using a mobile device yourself. And while it's possible to use an emulator as your sole testing platform, we don't really recommend it.

- **Create a directory or sub-domain somewhere on your web server to put your channel content.**

  For security, the web server should be located inside your firewall.

- **Place a few test pages where your channel is going to be, and create a custom channel to test/view your channel.**

  This is just to get you started in previewing your channel on a mobile device. There's no better way to figure out how a page is going to look than to try it yourself.

  For more information, see “Creating your first channel” on page 27.

- **Once you can view your test pages, start creating a separate template for your M-Business channel pages within your site's publishing system.**

  Different sites produce pages in different ways, but most likely, this will involve your creating a special M-Business channel template within Vignette, ColdFusion, or ASP.

- **Play around with the formatting until you're happy with the results.**

  For more information, see “Designing pages for the channel web site” on page 58.

  - **Place <meta name="HandheldFriendly" content="true"> tags on the top of every page.**

  - **Avoid tables that go over 150 pixels wide for Palm and Microsoft Smartphone, or 213 for Windows Mobile Pocket PC.**

    Users really dislike having to scroll horizontally within an M-Business channel. This most often happens if you accidentally represent a table using pixel width values instead of percentage width values.

  - **There's no need to place your entire content in a 150-pixel table.**

    This is a common mistake that I often see — sites publish all their news stories inside a 150-pixel table. There's no reason to do this — just use normal text. M-Business Client will word-wrap your text for the appropriate screen size. (Plus, if you put all your text into a 150-pixel table, it will look rather odd on a Windows Mobile Pocket PC, or larger screen device.)

  - **Don't create your entire content with <font size="-2">.**

This is also a common mistake — people feel that if they're creating something for a small screen, they should be using a font that fits many characters within 150 pixels. This isn't necessary, because the default font is already quite small for these mobile devices.

Because the Palm OS doesn't support small fonts, using a `<font size=-2>` tag will look perfectly normal when you're testing your page on Palm. But Microsoft OS devices, as well as the newer Palm OS°5 devices, support many more fonts, including `<font size=-2>`, meaning that your font will be smaller than you probably intended.

- **Adjust your logo so it looks good on mobile devices.**

  Usually this involves converting your image to a 216-color Web-safe palette. For more information, see "Designing graphics for channel pages" on page 69.

- **Preview your page on all device types that your company supports.**

  For Microsoft OS devices, you can also preview your page in Internet Explorer, sizing its window to the approximate size of the target device. Ask yourself: Do you have any text that looks strangely shifted to the left? Expecting to see a page break after a logo? Those formatting glitches will show up on Microsoft OS devices as well.

- **Handle form submissions elegantly.**

  We'll break this down into sub-steps as well.

  - **Do not place results in the Forms Manager. Instead, use cookies to link to search results on the front page.**

    Remember the first rule about using the Forms Manager: Don't use the Forms Manager. Nobody knows about it. Even when users do know about it, most of them don't like using it. Use cookies to remember the user's last few search results and place links to them directly on the front page. For more information, see "Submitting forms the right way" on page 179.

  - **Replace any Submit buttons with SubmitNoResponse buttons.**

    This gives you the opportunity to customize the dialogue box that appears, and you don't have to clutter the Forms Manager with unneeded forms. For more information, see "Using submitNoResponse() with the Submit button" on page 179.

- **Set up your channel for caching.**

  Your servers will thank you for it later, and the synchronization time of your channel will greatly improve. See "Caching to improve channel performance" on page 81. Specifically...

  - **Set up static stories to cache for at least 1 hour.**

    Personally, I recommend 2 hours or more.

  - **Set up images to cache for 24 hours or more.**

    This will probably be your biggest savings, as images are usually the largest chunks of data you're transmitting. Plus, you can safely cache them for longer periods of time.

- ♦ **Set up personalized pages to be Cache-Control: Private .**

  For more information, see “Caching guidelines” on page 181.

♦ **Test! Test! Test!**

Testing your channel before it goes live is always a good thing. DON'T just preview your channel in a web browser and assume it will work. Some web browsers, especially Internet Explorer, are more lenient about invalid HTML than M-Business Client is. You need to test your channel on M-Business Client.

Part II. Channel Basics

CHAPTER 2

Creating and testing a channel

Contents

About creating and testing channels

This chapter provides the information you will need to create, test, debug, and modify an M-Business channel. The rest of this guide deals with creating the content for your channel, making your channel available to the public, and subscribing readers.

All of the testing and debugging information is concentrated in one place in this chapter for two reasons:

- You will want to begin testing your channel immediately after you have set it up.

  After you have completed "Creating your first channel" on page 27, you can go right on into "Testing your M-Business channel" on page 37, and then take a look at "Channel troubleshooting and debugging tips" on page 42.

- You will want to test — and debug — your channel continuously as you develop and refine its content.

  Placing all the testing and debugging information in one place makes it easy for you to find when you need it. And you will probably want to have this information handy every time you add to or change your channel's content, set up caching, or customize content for different users or devices.

Even if you are really in a hurry to get your channel up and running, you should take your time with two sections in this chapter:

- "Creating your first channel" on page 27, and
- "Testing your M-Business channel" on page 37

Following the instructions in these topics will get your channel set up properly in a convenient test environment. You will be able to begin immediately to create and test content.

You will want to make use of "Channel troubleshooting and debugging tips" on page 42 if you encounter any problems in your initial channel testing. Otherwise, you can just skim this section now and return to it later, whenever the need arises.

Creating your first channel

There are up to four steps that you have to perform in order to create a functioning channel:

- “Making web pages available” on page 27
- “Setting up a test user” on page 27
- “Setting up a test group” on page 28
- “Specifying channel settings” on page 28

Additional topics in this section:

- “Viewing your channel” on page 35
- “Changing channel settings later” on page 35

Making web pages available

If you already have a web site that you want to use for your channel, you can use those pages in the instructions that follow. If you do not yet have a web site ready, just assume that you want to create a channel for your hypothetical newspaper, the Hello World Gazette, and build a test web site to turn into a channel. You would start off by building a simple HTML page, something like this:

```
<html>
<head>
<title>Hello World Gazette</title>
</head>

<body>
<center>
<p>Welcome to the Hello World Gazette.</p>
<p><b>Hello, world!</b></p>
</center>

</body>
</html>
```

Since it is just ordinary HTML, you can view the page in your favorite browser. It is not a particularly fascinating site, but it will do for a start.

Tip
As a general rule, it is a good idea to put all your M-Business channel pages in a separate directory.

Setting up a test user

In order to test your channel you will need to have a test user defined on M-Business Server. If you already have a user set up that you plan to use, you can skip this section.

♦ To set up a test user

1. Log in to the M-Business Server Administrator Console as an administrator.
2. In the Navigation Panel, choose **Users»Create User**.
3. Fill in the information on the Create User page.

   Note that only the **User Name**, **Password**, and **Confirm Password** fields are required.
4. Click the **Save** button at the bottom of the form.

For more information on setting up users, see “Managing users” [*M-Business Anywhere Administrator Guide*].

Setting up a test group

In order to test your channel you will need to have a test group defined on M-Business Server. If you already have a group set up that you plan to use, you can skip this section.

♦ To set up a test group

1. Log in to the M-Business Server Administrator Console as an administrator.
2. In the Navigation Panel, choose **Groups»Create Group**.
3. Fill in the information on the Create Group page.

   Note that only the **Name** field is required. A value for the **Type** field will always be supplied, with the default being **Managed** (an administrator or group administrator must assign users and users cannot remove themselves).
4. Click the **Create** button at the bottom of the form.

   The Edit Group page appears for the group you just created.
5. In the Navigation Panel, choose **Groups»Edit Group»Users»Add by List**.

   The Listing Users page appears.
6. Locate your test user in the list, then click the **Add** link at the right end of the row.
7. In the Navigation Panel, choose **Groups»Edit Group»Users»List** to verify that the user was added.

For more information on setting up groups, see “Managing groups” [*M-Business Anywhere Administrator Guide*].

Specifying channel settings

Your next step is to tell your M-Business Sync Server about your new channel.

♦ To specify channel settings

1. Log in to the M-Business Server Administrator Console.
2. In the Navigation Panel, choose **Groups»List Groups**.
3. Locate your test group in the list and click its link under **Group Name**.

   The Edit Group page appears.
4. In the Navigation Panel, choose **Groups»Edit Group»Channels»Create**.

   The Create a Channel for Group form appears.

> **Note**
> You can also create public channels, which are available to all users. If private channels are allowed, individual users may create channels that are only available to them. Group channels are the most common type of M-Business channel.

The rest of this section describes the settings on this form.

> **Note**
> Although you are probably setting up a group channel, notes are included below about differences in the settings available when you set up a public channel, or when a user sets up a personal channel.

Description of channel settings

Title

Asks for the name of your channel. Enter the name of your channel, or put in "Hello World" if you are using our example for your first channel.

> **Note**
> The Title you specify cannot match the title of any existing channel.

> **Note**
> Keep your channel title short, simple, and descriptive. It should convey the main idea of what your channel is about to potential subscribers.

Location

The URL of the home page of your channel Website. If you are using our example for your first channel, you would enter `http://www.hwgazette.com/avantgo/index.html` here.

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (/). The backslash character (\) in place of a forward slash will cause an error. When you are setting up directories in a Windows file system, where backslashes separate levels, be sure to change these characters to forward slashes if you copy parts of the file system path into a URL.

Category (public channels only)

The Category setting is available only for public channels. The admin user creates public channel categories and subcategories. A list of all categories and subcategories is available in the Category property's drop-down menu. This setting allows M-Business Server to organize public channels by the categories that the admin user creates.

For a test channel, you can leave this as **None (Miscellaneous).** If you wish to create a special category, such as test, click the **Create Category** link on the left.

For more information on categories, see "Organizing public channels in categories" [*M-Business Anywhere Administrator Guide*].

Description (public channels only)

The Description setting only is available for public channels. The admin user types a short channel description in this field. Users browsing for a public channel see this description and it helps them to understand what kind of information is contained in the public channel.

Channel Size Limit

The maximum size in kilobytes (KB) that the entire channel can consume on a device. M-Business Server compresses web pages for efficiency: The size limit reflects the compressed size, not the original size.

Some channels are very large and consume a large portion of the available memory on a mobile device. By entering a size limit, you prevent the channel from growing beyond that limit.

If a channel's pages total up more than the size you specify here, some pages will not be downloaded in a synchronization. 100KB should be a large enough size limit for getting started with your first channel.

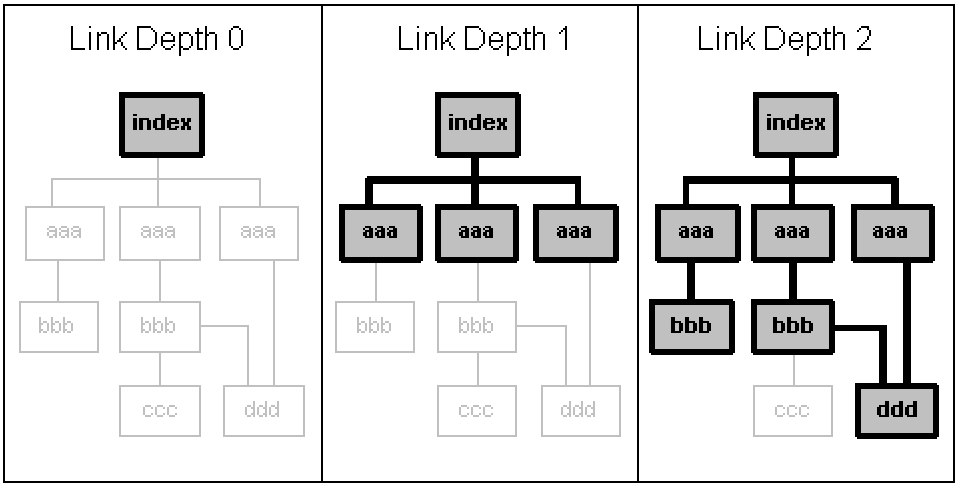
For more information on how the channel size limit is calculated and the trade-offs you face when you set this limit, see "Size limit issues" on page 52 and "Your channel's Channel Size Limit" on page 54.

For information on troubleshooting "This page is not available on the device" messages, which are common when your channel exceeds its maximum size limit, see "Channel troubleshooting and debugging tips" on page 42.

Note
The Channel Size Limit setting works together with the Link Depth setting. If Channel Size Limit is reached first, it supersedes Link Depth. If Link Depth is reached first, it supersedes Channel Size Limit.

Link Depth

Specifies the number of levels of links to follow from the channel's main page. If you set this to 0, for example, M-Business Sync Server returns only the page for the main location of the channel. If you set this to 1, it follows the links on the main page and downloads all of those pages as well. The greater the link depth, the more pages your channel downloads, which consumes more memory. It is rarely a good idea to set the link depth to greater than 1 or 2.



Set the Link Depth to the level beyond the channel's home page that should be downloaded in a synchronization.

Caution
Any pages in your site that are beyond the link level you set will not be downloaded.

Note
It will simplify your work and ensure that you are only dealing with one problem at a time if you leave Link Depth set at the default value of 0 until you have successfully tested the download of your home page in a synchronization.

Note
It is possible to get around your Link Depth setting if you have to. See "Downloading a page beyond channel's Link Depth" on page 56.

Note
The Channel Size Limit setting works together with the Link Depth setting. If Channel Size Limit is reached first, it supersedes Link Depth. If Link Depth is reached first, it supersedes Channel Size Limit.

Include Images

If you are creating your first channel using our Hello World Gazette, even though you do not have any graphics yet, you are creating this channel with the M-Business channel limitations in mind, and will be making graphics that are nice and small. So select the **Include Images** checkbox.

> **Note**
> Even if you select **Include Images** here, mobile device users still have the option of not displaying them.

For information on creating images for your channel site, see "Designing graphics for channel pages" on page 69.

Color Depth (group and personal channels only)

Sets the allowed color depth for a channel's images. The default selection is **Automatic**, which downloads color images to the device with the highest color depth that the device can support. Select a different setting to cap the color depth at a lower level and prevent images from exceeding channel capacity on devices that support greater color depth.

The following Color Depth options are available:

- **Black & White** (1-bit)
- **Greyscale** (2-bit)
- **16 Colors** (4-bit)
- **256 Colors** (8-bit)
- **High Color** (16-bit)

Lowering the number of bits allowed reduces synchronization time and saves memory on your device.

Follow Offsite Links

If your **Link Depth** is set to 1 or higher, indicates whether your channel should include pages from a server other than the one on which the main page of your channel resides. An offsite link is:

- any URL outside of the original HTTP site's domain
- a different machine name, even if the machine is in the same domain
- any link that is a re-direct

Sync Only

Check the **Sync Only** box if you want the channel content never to expire on the device. You would only do this with a channel that you expect have unchanging content over an extended period of time. Checking Sync Only for such a channel eliminates most of the communication with the web server that is required to determine if the web pages on the device need to be updated.

With connected users, this decreases the time it takes to synchronize the device. With wireless users, it decreases the time it takes to display the page every time the user accesses it — wireless users are more likely to notice the difference.

Allow Binary Distribution

Check **Allow Binary Distribution** only if the channel is a binary file, for example, an *.exe* or *.dll* file. Because most binary files are applications, which do not require frequent updating, also set the refresh rate to **Only once**. It is also useful to check the **Hide from User** option.

Hide From Users (group and personal channels only)

Check **Hide From Users** to hide the channel subscription from a user. A user will not see this channel listed either in the M-Business Server Personal Mode interface or in M-Business Client on the device. Hiding a channel is useful when synchronizing binary files.

Web Fetch Timeout

The number of seconds that M-Business Server should wait for the web server hosting the channel's content to respond when a user is synchronizing. If that web server does not respond within this period of time, the channel's content will not be updated in the current synchronization. The default is 90 seconds.

Start Sync URL (group channels only)

The **Start Sync URL** field is available only for group channels. Enter a URL in this field only if the channel is delivering a custom application that requires some type of backend processing to be performed *before* the channel’s web content is downloaded. For more information on using Start Sync and End Sync URLs, see “Using Start Sync and End Sync URLs to perform processing before and after channel synchronization” on page 237.

Note
The `SendSyncStartAndEndEvents` in the *sync.conf* file must be enabled in order for the specified Start Sync URL to be delivered to your custom application. See “Sync event settings” [*M-Business Anywhere Administrator Guide*].

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

End Sync URL (group channels only)

The **End Sync URL** field is available only for group channels. Enter a URL in this field only if the channel is delivering a custom application that requires some type of backend processing to be performed *after* the channel’s web content is downloaded. For more information on using Start Sync and End Sync URLs, see “Using Start Sync and End Sync URLs to perform processing before and after channel synchronization” on page 237.

Note

The `SendSyncStartAndEndEvents` in the *sync.conf* file must be enabled in order for the specified Start Sync URL to be delivered to your custom application. See "Sync event settings" [*M-Business Anywhere Administrator Guide*].

Note

URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

Refresh (group and personal channels only)

Check the **Refresh** setting to specify the frequency of on-device channel updates.

Remember the following when considering Refresh options.

- Refresh options do not affect online or connected browsing. The content from a site visited while browsing wirelessly is as current as the site itself. This is true no matter how or where a user is browsing.
- Refresh setting options only affect the synchronization of a device on a per user, per channel basis.
- Refresh setting options are set by the admin user or group administrator for group channels and by the admin user or the user for personal channels.

Table 1. Refresh options

Refresh option	Behavior	Reasons users might prefer this option
Only once	Synchronize one time, but never again.	Synchronizing a binary program file, reference guide, or template.
Every sync	Update sets of pages that change more than once a day by enabling a synchronization of the channel during every device synchronization.	Checking on the latest press release, stock price, or news articles.
Only once every <x> hours	Update once every <x> hours daily, or **only on weekdays.** The **Only once every <x> hours** option synchronizes and then disappears until the next synchronization time period starts. After a synchronization, this method does not synchronize again until the specified hourly period has passed.	Keeping track of stock quotes that change regularly.

Refresh option	Behavior	Reasons users might prefer this option
Daily at <time>	Update at specific times during the day. The daily refresh option synchronizes and then disappears until the next daily time period starts. After synchronizing using this method, the channel will not synchronize again until the next day and time.	Viewing a daily newsletter posted at a particular time.
Days of the week	Checking only those days of the week on which the **Daily at <time>** synchronization should occur restricts synchronizations to those days.	Viewing pages that change once a week or even less frequently, such as particular websites or reference materials.

Settings that users can change

If you set up a public channel instead of a group channel, users set the Refresh properties after the subscribe to the channel. Otherwise, no channel settings can be changed by users.

Saving channel settings

Make sure all the settings are what you want them to be, then click the **Create** button. You are returned to the Channel List for Group page for the group to which your test channel now belongs. The channel you just created now appears in the Group Channels list.

Viewing your channel

Synchronize your mobile device, start up M-Business Client on the mobile device, and start browsing your channel — the Hello World Gazette, if you are using our example for your first channel.

If you are unable to view your channel on the mobile device, see “Channel troubleshooting and debugging tips” on page 42.

For more detailed information about testing your channel, see “Testing your M-Business channel” on page 37.

Changing channel settings later

Any time you need to change any of the settings you have specified for a channel, follow the steps below to make changes.

♦ To change group channel settings

1. Log in to the M-Business Server Administrator Console.
2. In the Navigation Panel, choose **Groups»List Groups**.
3. Locate your channel's group in the list and click its link under **Group Name**.

   The Edit Group page appears.
4. In the Navigation Panel, choose **Groups»Edit Group»Channels»List**.
5. Locate your channel in the list and click its link under **Channel Title**.

   This displays the Edit Channel for Group form, on which you can change any of your channel settings.

♦ To change public channels

1. Log in to the M-Business Server Administrator Console.
2. In the Navigation Panel, choose **Public Channels»List Channels**.
3. Locate your channel in the list and click its link in the Title column to display the Edit a Public Channel form

Testing your M-Business channel

So you have got your channel set up, and you are adding content. This also means you are probably testing it all the time. Here are some tips for making your testing go a little more smoothly.

Topics in this section:

- “Synchronizing just one channel” on page 37
- “Testing on emulators” on page 39
- “Testing on different devices” on page 41

Synchronizing just one channel

You are probably a little tired of waiting for all of your M-Business channel to synchronize up every time you want to update the channel you are testing. One solution is to remove all of your channels except the test channel, but if the mobile device you are testing with also happens to be the mobile device you use personally, it is not the best solution. After all, it would be nice to keep your personal M-Business channels without having to remove and re-install them constantly. Luckily, there is another solution:

♦ To synchronize just one channel

1. Get yourself a second account.

   Log in to the M-Business Server Administrator Console and choose **Users»Create User**.

2. Keep one of your accounts (if your name is Todd, you might call it toddpersonal — or maybe you've already named it todd) as your personal account.

   Keep all the channels you normally use in your personal account.

3. Make the second one (maybe todddev if your name is Todd) into your development account.

   Add your test channel to (and remove anything else in) the development account.

   The next time you synchronize your mobile device, it will add the second account on your device.

4. When you are ready to start testing your channel, configure the server that contains your personal account.

   For detailed instructions, see the *User Guide for M-Business Anywhere Client* chapter for your test platform:

   - “Configuring a server connection on Palm OS” [*M-Business Anywhere Client User Guide*]
   - “Configuring a server connection on Windows Mobile Pocket PC ” [*M-Business Anywhere Client User Guide*]
   - “Configuring a server connection on Windows XP” [*M-Business Anywhere Client User Guide*]

- “Configuring a server connection on Windows Mobile 5 or 6, or Microsoft Smartphone” [*M-Business Anywhere Client User Guide*]
- “Configuring a server connection on Symbian OS” [*M-Business Anywhere Client User Guide*]

5. Clear the **Enabled** setting.

   This tells your M-Business Sync Server not to synchronize any of the channels on this account.

6. Do not touch the **Refresh All Content** setting.

   This is usually handled automatically by your M-Business Sync Server.

7. Double-check and make sure the server with your development account is enabled.

8. Click **Done** to save your settings.

9. Synchronize your mobile device.

   If everything is set up correctly, you will only synchronize the account that contains your development channel. The synchronizing process should be a whole lot speedier than before.

   Even though the server with your personal account is disabled, you will still be able to view your personal channels — just switch between accounts.

   For detailed instructions on switching between accounts, see the *User Guide for M-Business Anywhere Client* chapter for your test platform:

   - “Using M-Business Client on Palm OS” [*M-Business Anywhere Client User Guide*]
   - “Using M-Business Client on Windows Mobile Pocket PC” [*M-Business Anywhere Client User Guide*]
   - “Using M-Business Client on Windows XP” [*M-Business Anywhere Client User Guide*]
   - “Using M-Business Client on Windows Mobile 5 or 6, or Microsoft Smartphone” [*M-Business Anywhere Client User Guide*]
   - “Using M-Business Client on Symbian OS” [*M-Business Anywhere Client User Guide*]

   > **Note**
   > Remember that while your personal account is disabled, the channels on that account will not update throughout the day as you synchronize.

10. When you want to start synchronizing your personal channels again, re-enable your personal account.

    For detailed instructions, see the *User Guide for M-Business Anywhere Client* chapter for your test platform:

    - “Configuring a server connection on Palm OS” [*M-Business Anywhere Client User Guide*]
    - “Configuring a server connection on Windows Mobile Pocket PC ” [*M-Business Anywhere Client User Guide*]

- “Configuring a server connection on Windows XP” [*M-Business Anywhere Client User Guide*]
- “Configuring a server connection on Windows Mobile 5 or 6, or Microsoft Smartphone” [*M-Business Anywhere Client User Guide*]
- “Configuring a server connection on Symbian OS” [*M-Business Anywhere Client User Guide*]

Synchronize your mobile device one last time to update your personal channels, and you are done!

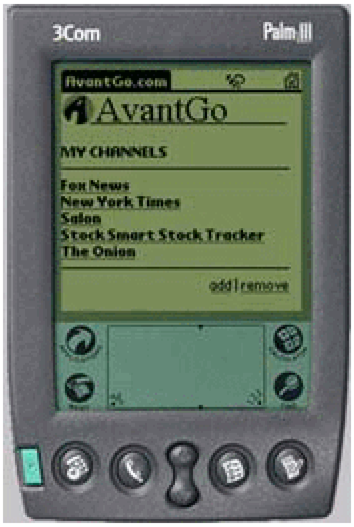
Testing on emulators

So you are previewing your channel on a high-end Palm device. It looks nice, but how does it look on a Palm III? Or a Palm IIIx? Or a Palm VII? Or a Palm IIIe? Or a Windows Mobile Pocket PC, Windows Mobile°5, Windows°XP, or a Microsoft Smartphone?

Palm Computing has an excellent emulator available to the public from:

http://www.palmos.com/dev/tools/emulator/

The emulator is available for free, although in order to run it, you need ROM images of the various mobile devices. These are also available for free, but for legal reasons, you need to sign a license agreement in order to gain access to them.



Microsoft also has an emulator for Windows Mobile 5 and Microsoft Smartphone devices, available to the public from:

http://www.microsoft.com/

From the Microsoft home page, search for "standalone device emulator" to locate the download pages. (It's basically one emulator, with a separate image you must also download for Windows Mobile 5 and Microsoft Smartphone.)

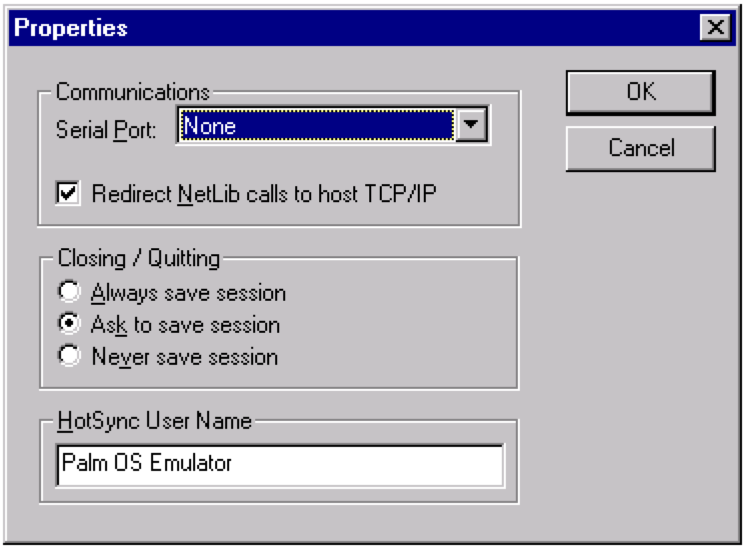
Installing M-Business Client on the Palm emulator

I will not go through the entire process of installing and setting up the emulator. There is pretty good documentation available at Palm's site. But here are a few steps you need to perform in order to run M-Business Client on the emulator.

♦ To install M-Business Client on the Palm emulator

1. Start up the emulator.
2. Right click the emulator to display the shortcut menu.
3. Choose **Install Application»Database»Other»Install**.
4. Select *avantgo.prc*, *libsmal.prc*, and *MobileLink.prc*.

   If you have recently installed M-Business Client on your Palm OS device, chances are they are available somewhere on your hard drive, most likely in your *Palm\(username)\backup\* directory.
5. Right click the emulator to display the shortcut menu.
6. Choose **Settings»Properties** to display the Properties dialog.



Serial Port: None

☑ Redirect NetLib calls to host TCP/IP

○ Always save session
● Ask to save session
○ Never save session

HotSync User Name: Palm OS Emulator

7. Select the checkbox that says **Redirect NetLib calls to host TCI/IP**.
8. Click **OK** to save your settings.

Configuring M-Business Client in an emulator

Now you need to set up M-Business Client in the emulator, just like you would on a physical device.

♦ To configure M-Business Client in the emulator

1. In the emulator itself, run M-Business Client.
2. Go to the Server Preferences screen.
3. Choose **Server»New Server** from the Server Preferences menu.

   Your server address should be the name of the server where M-Business Server is installed (for example, mbiz.mycompany.com), port **80**, and use the **User name** and **Password** for whatever M-Business account you would like to synchronize with.

4. Click **Done**.

   This will set up M-Business Client on your emulator to synchronize with one of your current M-Business Anywhere accounts.

Synchronizing your M-Business Anywhere account in an emulator

To synchronize your M-Business Anywhere accounts, go to the menu, and Choose **Channels»Modem Sync**. If everything is working correctly, your M-Business Anywhere account will synchronize and you will be able to use M-Business Client as if it were on a real Palm OS device.

Testing on different devices

Whether you use emulators or physical devices, it is extremely important that you test your channel on as many different devices as you can arrange to use. At a minimum, you should test on all the specific devices that you expect your users to have — the ones that are officially authorized and supported.

For information on optimizing channel pages for different devices, see “Customizing content for users and devices” on page 159.

Channel troubleshooting and debugging tips

This section lists some of the most common problems that are encountered in trying to get a channel set up and operating properly.

This page is not available... message on your test device

If you get this message on the device after you synchronize, it means that your M-Business Sync Server, for some reason, has not downloaded the page for viewing on the mobile device. There are four common causes for this message:

- The page was not available on the web server.
- The Link Depth is set too low to include the page.
- Your content exceeded the Channel Size Limit.
- Follow Offsite Links brought in content that caused the Channel Size Limit to be exceeded.

♦ To troubleshoot and debug a page not available message

1. Check your channel's current settings.

    - Go to the edit form for the channel. For a group channel or a personal channel, this would be the **Edit Channel** form; for a public channel, the **Edit Public Channel** form.
    - Note the settings currently in effect for **Channel Size Limit**, **Link Depth**, **Include Images**, and **Follow Offsite Links**.

2. Check the sync log.

    The sync log will tell you immediately:

    - if your channel exceeds your Channel Size Limit
    - if the page was simply not found on the web server
    - if the device is out of memory

    If it is any of the above, correct the problem and synchronize again. Otherwise, continue below.

3. Check the size of your channel.

    - Go into the file system on the web server where your channel is hosted.
    - Locate the web pages that are within your Link Depth setting and total up the bytes for the HTML and image files.

    This will give you an estimate of your channel's size. Because of compression and stripping out of extraneous HTML, your channel's actual, downloaded size will be less.

Fix: If the downloaded size of your channel is well below the Channel Size Limit setting, most likely the page or pages that are not available on the device were too large.

- Break these pages into two or more smaller pages.
- Synchronize again and see if this clears up the problem.

4. **Next steps:**

- If the estimated size of your channel is *not* larger than your Channel Size Limit:

  (1) **Check Link Depth** — is the page not available because it is on your channel site but beyond the specified Link Depth? For more information, see the "Link Depth" topic in "Description of channel settings" on page 29.

  (2) **Check Follow Offsite Links** — is the page not available because it is outside your channel site and you have Follow Offsite Links set to No? For more information, see the "Follow Off-Site Links" topic in "Description of channel settings" on page 29.

  (3) **Check server speed** — if the estimated of your channel is *not* larger than your Channel Size Limit and the page(s) are not missing because of your channel's Link Depth or Follow Offsite Links settings, then the problem is most likely caused by a slow web server on your channel site. Your M-Business Sync Server times out if its page request is not processed fairly quickly; for more information, see "Server timeouts encountered" on page 47.

- If the estimated of your channel is larger than your Channel Size Limit, the first thing you need to do is figure out *why*. Continue with the following steps in this section.

5. If **Follow Offsite Links** is set to **Yes**, look for leaks to other sites.

Leaks are links to pages that you don't really want downloaded with your channel. Leaks probably account for two-thirds of the channel size problems we see. The fixes below may apply if your channel has Follow Off-Site Links set to Yes.

Fix 1: If you really don't want to download pages from other sites with your channel pages, then *every* off-site is a leak!

- Set **Follow Off-Site Links** to **No**.
- Synchronize again and see if that clears up the problem.

Fix 2: If you *do* need certain off-site pages to be available with your channel pages:

- Carefully scan all your channel pages looking for off-site links that you do *not* want downloaded.
- Remove them from your site entirely, or move them to pages that are outside the Link Depth you have set.
- Synchronize again and see if that clears up the problem.

6. Look for leaks within your site.

Leaks within your site are links to pages that you don't really want downloaded with your channel, even though they are part of your site. Again, leaks probably account for probably two-thirds of the channel size problems we see. The fixes below may apply if your channel has Follow Off-Site Links set to No, *or* if it is set to Yes and you have already eliminated links to other sites.

Fix: Work your way through the links within your channel site and eliminate links to unwanted pages:

- Make a list of any pages within your site that are not available after a synchronization. (At the same time that you are looking for leaks within your site, you can determine whether any of the missing pages are beyond your channel's Link Depth setting.)
- Start with links from your home page.
- Then proceed to links from pages that are linked from your home page, working your way down the links until you have reached your channel's Link Depth. For a detailed description of Link Depth, see the "Link Depth" topic in "Description of channel settings" on page 29.
- Check off any of the missing pages that you come to while you are following links — you are confirming that these pages are within your channel's Link Depth.
- Look for links to pages that you do not want downloaded with your channel.
- Remove them from your site entirely, or move them to pages that are outside the Link Depth you have set.
- Synchronize again. and see if that clears up the problem.

7. If any missing pages are beyond your channel's Link Depth setting, bring them within it.

Your M-Business Sync Server will never download any page on your channel site that is more links away from your home page than the number specified for your channel's Link Depth. For a detailed discussion of link depth, see the "Link Depth" topic in "Description of channel settings" on page 29.

If you just checked for leaks within your site, you have already determined whether any of the missing pages are beyond your Link Depth setting. If you skipped that step:

- Make a list of any pages within your site that are not available after a synchronization.
- Start from your home page and count the number of links it takes to reach each missing page (by the shortest route).
- Note any pages for which this link count is more than your Link Depth setting.

If any of the missing pages are beyond your channel's Link Depth, there are two possible solutions.

Fix 1: If increasing your channel's Link Depth would add a lot of pages that you don't want to download in a synchronization:

- Add at least one link to each missing page from a page that is close enough to the home page — that is, you can get to it with a count of links equal to the channel Link Depth minus one.
- If it really doesn't make sense to link a missing page from any page other than one that is already at your channel's Link Depth, you can add an invisible link to it from your home page. See "Downloading a page beyond channel's Link Depth" on page 56.

Fix 2: If increasing your channel's Link Depth would not add a lot of pages that you don't want to download in a synchronization:

- Go to the Channel Properties screen for the channel.
- Increase the **Link Depth** setting to a value that will include the missing pages.
- Synchronize again and see if this clears up the problem.

8. Make your **Channel Size Limit** fit your channel content.

   If none of the troubleshooting steps above has eliminated your missing page(s) problem, your channel content is probably just too large for the Channel Size Limit you have set.

   Fix 1: If your current Channel Size Limit setting is really small:

   - Go to the Edit... Channel form for the channel.
   - Increase the **Channel Size Limit** setting to a value that will be large enough to hold the missing pages.

     For more information on setting Channel Size Limit, see the "Channel Size Limit" topic in "Description of channel settings" on page 29.
   - Synchronize again and see if this clears up the problem.

   Fix 2: If your current Channel Size Limit setting is as large as you are willing to make it:

   - Redesign your site, eliminating some pages or putting them beyond the channel's Link Depth.
   - Consider setting **Follow Off-Site Links** and **Include Images** to **No**.
   - Synchronize again and see if this clears up the problem.

This page is not available... message on another device

It is entirely possible that even after you have solved all your problems with missing pages on your initial test device, a co-worker with a different device encounters this message. What is going on?

♦ To troubleshoot and debug a page not available message on another device

1. Make sure you both have the same **Channel Size Limit** settings for your channels.

   It is possible you increased yours to 250KB one day for testing purposes and forgot to change it back.
2. Make sure all of your other channel settings are the same.

   Maybe you have **Follow Offsite Links** turned off and your co-worker has it turned on, so he is detecting a leak to another web site that you are not seeing. If in doubt, unsubscribe and re-subscribe to your channel to get the default settings that your customers would normally see.
3. The other thing you should check is that you have similar devices.

Remember, graphic size will vary significantly from, say, a Palm III to a Windows Mobile Pocket PC device. If you have a site with a lot of graphics and you are pushing the limit on your 4-gray mobile device, you will probably go over the limit on a color device.

Insufficient device memory message

If you receive an Insufficient device memory message, it indicates that the M-Business XML conduit is attempting to download more data than can be accommodated by the device memory. Redesign your application to download a smaller dataset.

Not sure cookie is being set properly in your channel

Because you are setting cookies using a normal cookie standard, you should be able to test all your cookie functionality in a normal desktop browser.

Try viewing your channel in a desktop browser such as Netscape or IE, but alter your preferences so that your browser prompts you before adding a new cookie. In Netscape or IE, remove all your old cookies set by your channel, and then try viewing your page again.

You can use a program such as tcpTrace, from PocketSoap (http://www.pocketsoap.com), or Cookie Pal, from Kookaburra Software (http://www.kburra.com), to tell you the values of your cookies. Or, if you are using Netscape 6, you can use the **View Stored Cookies** button in the Cookies section of your preferences to view your current cookie status.

In addition, you can create a script to print out all HTTP headers that are being sent to your site and view that page through M-Business Client. That should display any cookie information that M-Business Client is sending to your site for your account.

Tips for debugging cookies

To obtain all the information necessary to debug the full range of cookie problems that you might encounter, iAnywhere Solutions recommends the tcpTrace program, from PocketSoap (http://www.pocketsoap.com). Using its logging capabilities, you can view all the details of all the cookies that are passed between M-Business Server and the web server hosting your channel.

Need to remove all cookie information from your M-Business Anywhere account

At some point in testing, you will probably want to remove all cookie information from your M-Business Anywhere account in order to see what happens when a particular cookie is added for the first time.

♦ To remove all cookie information from your M-Business Anywhere account

1. Log in to the M-Business Server Administration Console as your test user.
2. From the Navigation Panel, choose **Settings»Clear Web Caches**.

3. Click the **Clear Cookies** button.

Tips for debugging on M-Business Server

If you need to see exactly how your channel is being processed on M-Business Server, you can specify that the maximum level of detail is recorded in M-Business Server log files. See "Enabling debug level logging" [*M-Business Anywhere Administrator Guide*].

Server timeouts encountered

There are two things that will make your M-Business Sync Server timeout:

- It does not receive an entire page 90 seconds after first requesting it.
- 45 seconds elapse between receiving packets from your web server.

Fix: Anything you can do to increase the performance of your channel site's web server will help:

- Make sure you have set up caching for your channel.
- See "Caching to improve channel performance" on page 81. If lots of users are accessing your channel site through your M-Business Sync Server, performance can be greatly improved by setting up caching.

  Put your channel site on a faster server.

  If server timeouts are still a problem, about the only thing you can do is move your channel site to a faster server, or you may be able to negotiate enhanced service from your current hosting service.

Not sure forms are being sent properly

If you set `isHidden` to `false` while you are testing your channel, you can preview your form submissions in the Forms Manager to make sure they are being sent properly.

Only last offline submission of same form works

If you want offline users to be able to submit the same form again and again, without synchronizing, you must set the `isMulti` flag of your `submitNoResponse()` button to `true`. If it is set to `false`, then only the most recent form submission is sent and the others are dropped.

If you check and find that the `isMulti` flag of your `submitNoResponse()` button to `true`, then the multiple submissions are probably interfering with each other because data that is stored in a single cookie is being changed by each submission. You may have to change the way the information in the cookie is stored. See "Avoiding collisions in multiple offline submissions" on page 189.

Caching prevents you from counting user synchronizations

Recording hits on your channel site is just not possible in the same way that it is for pages viewed in a desktop browser.

Need to troubleshoot a JavaScript problem

If you believe you have a JavaScript problem, you can use JavaScript to turn on the display of JavaScript errors right before the section where you believe the problem is occurring, then turn off the display of JavaScript errors right after that point. For more information, see "Displaying and hiding JavaScript errors" on page 332.

CHAPTER 3

Design considerations for channel content

Contents

Overview of channel design

When you are designing a web site for an M-Business channel, you have to make design decisions at two levels:

- **Site level** — how many links away from the home page within your site will be downloaded in the channel; whether or not links to pages outside your site will be included in the channel; the maximum size within which your channel must fit. These issues are covered in "Designing the channel web site" on page 51.
- **Page level** — the nitty-gritty details of each page; the specific HTML used; details of graphics to that are included; which mobile devices you want to optimize your pages for and which pages you want to optimize. These issues are covered in "Designing pages for the channel web site" on page 58.

  At the page level, the design decisions you need to make about graphics for your channel warrant their own section in this chapter — "Designing graphics for channel pages" on page 69.

Even if you are really in a hurry to get your channel up and running, you should spend some time going over the section on "Designing the channel web site" on page 51 before you start designing your channel pages. You can skim the rest of this chapter, returning to sections that deal with specific issues as you encounter them.

Designing the channel web site

Before you deal with individual pages, it's a good idea to work out the site design within which the individual pages need to fit. Remember that, compared with desktop web development, channel development forces you to work within fairly tight constraints. This section describes the design considerations that you will have to deal with, and the trade-offs that you might choose to make.

Choosing content for the channel web site

The important thing to remember when deciding what information to store on the mobile device is that it should be essential. In general, this means information that a user will access regularly or will urgently require in certain situations. Of course, different people have different needs, so you certainly want to keep your audience in mind as well. It is also advisable to let users customize their content when possible. User preferences can be handled with cookies in the same way that this is done in desktop web design.

Organizing pages for the channel web site

When you put information into a group of hypertext pages, you have to strike a balance between the depth of the page hierarchy and the length of the individual pages. Given the smaller screens of mobile devices, you should avoid including more than one topic onto the same page. This is primarily because it can be difficult to find your place in a lengthy section of text using the scroll bar.

If you are creating a news site, and have several stories that are several handheld-screens long, it is a better user experience to place each story in its own page rather than placing them all together.

Considering mode: online or offline

When considering whether to break up long pages, you should also consider whether your users will operate primarily in wireless mode or offline synchronization mode and set the channel link depth accordingly. In wireless mode, the user has a constant connection to the Internet through a wireless connection, just like on the desktop. If the user clicks a link for a page that has not been downloaded to the device, it will be downloaded at that time. In offline synchronization mode, the mobile device has no modem connection to the network. If the user clicks a link for a page that has not been downloaded to the device, they see a message saying that the page is not available.

You should design your pages to work well in offline mode, because one of the key advantages of M-Business Client is that it does not require a constant network connection. To do so, simply adjust the link depth for your channel to ensure that all necessary pages are downloaded to the device during a synchronization. A link depth of *n* means download the index page and any page I can get to in *n* links from that page. Thus, if your channel consists of a master index that links to ten content pages, set the link depth to one to download all of these pages during a synchronization. If one or more of these content pages in turn link to several other necessary content pages, set the link depth to two.

Note
If your channel site contains any forms, the major issue you will face in setting up your channel to work equally well online and offline is managing the forms that your readers submit. You can first design and test all the other aspects of your site, including creating pages with forms, then work out the management of form submissions before you make your channel available to readers. When you are ready, see "Managing channel form submissions" on page 173.

Size limit issues

The most important issues you are likely to face in designing your channel site involve size limitations. There are basically three types of size limitations for an M-Business channel:

- **M-Business Sync Server limits** — restrictions on the maximum size of any individual graphic. See "M-Business Sync Server limits" on page 52.
- **Your channel's Maximum Channel Size limit** — the upper limit you set for your M-Business Sync Server to download from your channel in a synchronization. See "Your channel's Channel Size Limit" on page 54.
- **User limitations** — users' resistance to subscribing to a channel that is too large for limited mobile device memory and 2MB account limit. See "User limitations" on page 55.

Basically, you could get around your M-Business Sync Server's limitations by simply increasing the Maximum Channel Size setting until all your pages fit. But your users will be very unhappy if it takes too long to synchronize your channel.

M-Business Sync Server limits

Take a quick look at a typical page and a typical image and see what happens to their files sizes when they get compressed and sent up to the device.

A page's original size

Suppose a typical page has 15KB of HTML. This is actually a fairly large page, taking up several screens worth. You might accidentally have a few things included that are not supported on the device we are viewing with. (For instance, somebody with a Palm OS device can safely ignore all of the `<font face="verdana" size="2">` tags.) This is the page file's *original size*.

A page's compressed size

Your M-Business Sync Server will take this page, strip out anything it does not need, and then compress the remaining text. Depending on the page, how well it compresses, and how much we strip out, the final size will probably end up somewhere between 3KB and 8KB. Assume it is 5KB for our example. This is the file's *compressed size*. This is the page size that is totaled up, along with any images, until the Maximum

Channel Size is reached. A page's compressed size will vary with the type of device onto which it is downloaded.

A page's display size

This compressed page file is sent to the device, where it waits for somebody to view it. When a user asks to see the page on the screen, M-Business Client then uncompresses it. This will be approximately equal to its original size, minus any unsupported HTML we stripped out along the way. This is the file's *display size*. With our example page, this will probably end up being about 14KB. (Unless it is 2KB of HTML, and 13KB of unsupported client-side `imagemap` code.)

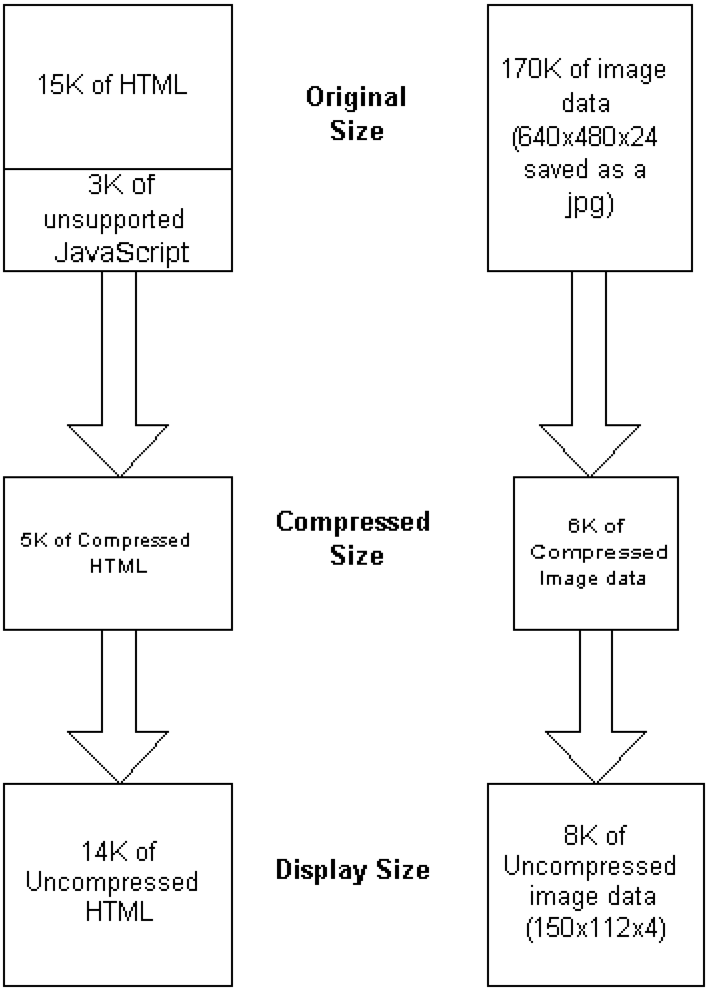
Image sizes

A similar process happens with images. With images, however, there are two important differences.

The first difference is that the compressed size (and the display size) will vary much more between different devices, because we will convert the image to the appropriate bit depth. For example, look at a 150x150 color JPEG that is 16KB in its original size. If I synchronize it to a 216-color Palm III, it will end up about 15KB. If I synchronize it to a 16-gray Palm V, it will end up about 12KB. And if I synchronize it to a 4-gray Palm III, it will end up about 6KB.

Image size limits

Your M-Business Sync Server may scale down your images to the maximum size that the mobile device can display. See “How M-Business Client processes images” on page 69.


15K of HTML
3K of unsupported JavaScript
Original Size
170K of image data (640x480x24 saved as a jpg)
5K of Compressed HTML
Compressed Size
6K of Compressed Image data
14K of Uncompressed HTML
Display Size
8K of Uncompressed image data (150x112x4)

Your channel's Channel Size Limit

Your channel's compressed size cannot go over the Channel Size Limit in your channel settings. This case does not give you any sort of error message, except that you will occasionally receive This page is not available on the device messages on pages that you would normally expect to have on the device.

For information on troubleshooting This page is not available on the device messages, see “Channel troubleshooting and debugging tips” on page 42.

For information on increasing your channel's Channel Size Limit, see the "Channel Size Limit" topic in “Description of channel settings” on page 29.

User limitations

Since your channel's Maximum Channel Size setting is under your control, why not just specify our channel's default maximum size to be 500KB, or whatever size ensures that all your channel's pages will always fit, and not have to worry about the Maximum Channel Size issue ever again?

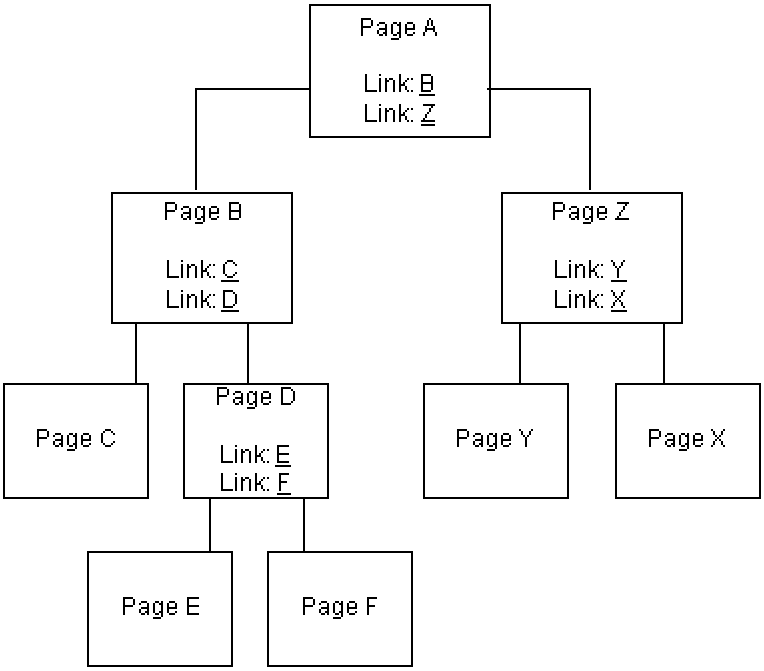
Your users will be very unhappy if it takes too long to synchronize your channel.

Pages are returned in unpredictable order

Pages from your channel site are returned to your M-Business Sync Server in a generally unpredictable order. As a consequence, you *must* ensure that all the pages that you want downloaded in your channel fit within your Maximum Channel Size setting.

Why? If your channel content within the set Link Depth exceeds your Maximum Channel Size setting, you cannot predict which pages or graphics will fail to download. And the exact combination of pages and graphics that fails to download can change every time you synchronize, because you never know what pages your M-Business Sync Server will receive first before your channel reaches its maximum size.

For example, in a site like the following...



Your M-Business Sync Server will grab page A, and then make requests for page B and Z. After that, it all depends on what comes back first. If page Z comes back first, we will request pages Y and X before requesting C and D. But if B comes back first, we will request pages C and D next. (And heck, we may end up grabbing pages E and F before we even get page Z back.). Pretty much the only thing you can guarantee is that you will receive a page before any of its children (that is, you will receive page B before C, D, E, and F, assuming no other pages link to them.)

Avoiding grabbing pages multiple times

Your M-Business Sync Server is smart enough to know whether it has grabbed a page already or not. It does this by keeping track of every URL it is grabbed during the current synchronization. If you request a page that has already been downloaded, your M-Business Sync Server will simply point to the downloaded page, rather than getting the page all over again.

Control case of channel URLs

Channel URLs are case sensitive, even if your web server is not. That is, suppose you have a page called *articles.html* — sometimes you link to it as *articles.html*, other times as *Articles.html*. If you are on a web server running in Windows 2000 or NT, these two URLs are treated as the same thing, and your web site will look and act perfectly normal. But as far as your M-Business Sync Server is concerned, the *articles.html* page and the *Articles.html* page are two totally different pages, so it will request that page twice every time a user synchronizes!

If you are running your web site on a Unix-based server, page names are case sensitive. This means you probably will not run into this problem, because if you had a page called *articles.html*, linking to it as *Articles.html* would get you a 404-not-found error.

Make all URLs pointing to same page identical

Along the same lines, it is also a bad idea to point to the same page with different URLs. Say, for example, you were including a `link_from` parameter in the URL that pointed to the referring page. You are only using this parameter for page-tracking purposes; it does not affect content, meaning that these three URLs point to the exact same page:

```
http://www.mywebsite.com/get_story.asp?storyID=50&link_from=TOC
```

```
http://www.mywebsite.com/get_story.asp?storyID=50&link_from=story51
```

```
http://www.mywebsite.com/get_story.asp?storyID=50&link_from=story49
```

This would be a bad idea for an M-Business channel because it would treat each of these URLs as different pages, meaning it would request that story 3 times during the same synchronization. It also would not provide you with much useful information if you were trying to do this for page-tracking reasons, since it would grab every link and you would not know where the user really requested the story from.

Downloading a page beyond channel's Link Depth

Suppose you have one or two pages that are beyond your channel's Link Depth. You investigate and find that redesigning your site so that those one or two pages would lie within the Link Depth would not make sense. So you increase your Link Depth to include the pages — with the result that your channel's download size increases drastically, because there are dozens of other pages that also get included by the increased Link Depth. Downloading all those pages that you don't want, just to get the one or two that you need, would *not* be a good solution! What else can you do?

Note
You can include a page that is beyond your channel's Link Depth by placing a hidden link to it on your home page. For example, you might use a transparent.GIF so that the user does not actually see the link. This gives the page an effective link depth of one.

Preventing download of unwanted pages with Follow Offsite Links

If you need to include in your channel one or more pages from a web site that is outside your channel web site, you have to be careful. You have no control over what additional links appear on those pages, and so you risk having completely irrelevant content included when users synchronize your channel. At a minimum, this will unnecessarily increase your channel's synchronization time. In the worst case, this could even cause some of your channel's pages not to download, if the irrelevant pages are picked up first and cause your channel's Channel Size Limit to be exceeded.

Note
You can prevent unwanted offsite pages from being included in your channel by placing all of the offsite links on pages within your channel web site which are at Link Depth minus one. This ensures that a link a page outside your web site will not return any additional pages that are linked from the page that you want, because they will lie beyond your Link Dept setting. For a full explanation of how Link Depth works, see the "Link Depth" topic in “Description of channel settings” on page 29.

Channel caching

If you expect to have any significant volume of users subscribing to your channel, you will definitely want to implement caching. For more information on enabling caching for your channel, see “Caching to improve channel performance” on page 81.

Designing pages for the channel web site

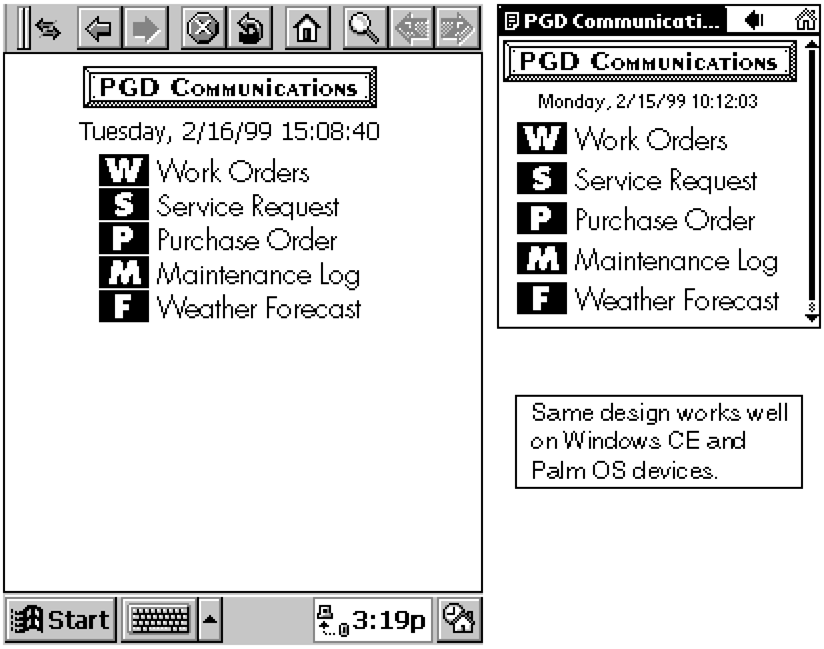
This section summarizes the guidelines you need to follow to create effective pages within the screen size and font constraints of mobile devices, using only the available subset of HTML.

Page layout

Designing a quality layout for mobile devices requires you to channel some of your creativity into planning for economy. The key is to decide what information is essential and then present that information in manner that is clear, concise, and easy to navigate.

To produce effective results, here are a few rules to which you should try to adhere:

- Maximize screen use while avoiding clutter.
- Minimize page length.
- Well-designed mobile web pages



PGD Communications

Tuesday, 2/16/99 15:08:40

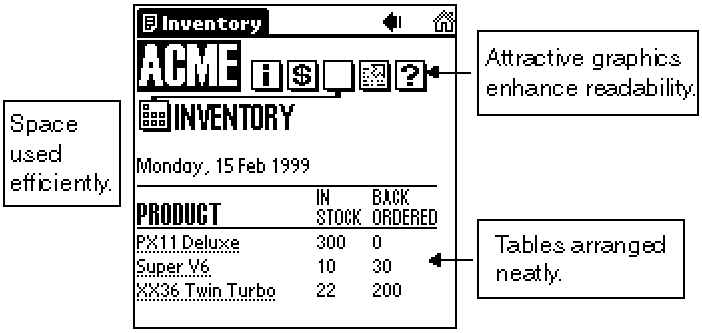
W Work Orders
S Service Request
P Purchase Order
M Maintenance Log
F Weather Forecast

PGD Communications

Monday, 2/15/99 10:12:03

W Work Orders
S Service Request
P Purchase Order
M Maintenance Log
F Weather Forecast

Same design works well on Windows CE and Palm OS devices.



INVENTORY

Monday, 15 Feb 1999

PRODUCT	IN STOCK	BACK ORDERED
PX11 Deluxe	300	0
Super V6	10	30
XX36 Twin Turbo	22	200

Space used efficiently.

Attractive graphics enhance readability.

Tables arranged neatly.

To see these concepts in action, look at a couple of well-designed pages.

Other considerations — landscape mode and minimum font size

Keep in mind when designing your channel pages that users are now able to do two things that potentially can have a major impact on your page layout.

- **Landscape mode:** On newer devices that support it, users can switch the page view between portrait and landscape. Web pages are typically designed for viewing in a vertical, portrait mode. M-Business Client currently supports landscape mode on Sony Clie PEG-UX40, Sony Clie PEG-UX50, and Palm Tungsten T3.
- **Minimum font size:** On all devices using M-Business Client Version 5.7 or later, users can set a minimum font size that may be larger than what your page specifies. This can affect your page layout.
- **Full screen**: On all devices using M-Business Client Version 5.7 or later, users can expand the page view area to fill the full screen. This user control is the least likely impact your page layout, but keep it in mind as you design.

To minimize the negative impact of these two user controls on your page layouts:

- **Control full screen setting programmatically**: If the full screen setting is critical on a certain page in your channel, when that page loads you can read and store the current setting, switch to the setting you need, then restore the previous setting when the page unloads. For details, see “fullScreen” [*M-Business Anywhere API Reference*].
- **Size table widths dynamically**: Make sure your tables size dynamically so that they will expand to the page margins when in landscape mode, without forcing users to scroll horizontally in portrait mode.
- **Center full-width objects**: If you have an important header or graphic that fits exactly in the width of the page in the narrower view of the page, make sure it is centered, so that it changes as little as possible if white space is added for a widened page view.
- **Use small fonts sparingly**: Make sure that, wherever small fonts are used, if users increase their size, your dynamically resized layout will be acceptable.

- **Test on a sampling of user devices**: Determine the range of devices your users have, decide which ones will be officially supported, and then test your pages to see how they look on different devices with different user settings.

Using the HandheldFriendly META tag

Include a `<META NAME="HandheldFriendly" content="True">` tag in the head element of every page. This tells your M-Business Sync Server that you have optimized your page for being viewed on a mobile device. Without it, tables, JavaScript and certain image tags will be dropped when the page is downloaded.

Language and character set support

The tables below detail the languages and character sets that M-Business Client version 6.2 and later support.

> **Note**
> This section details the languages and character sets that M-Business Client supports. In order for HTML pages to display correctly, it is also necessary for the fonts required by supported languages and character sets to be present on the device where M-Business Client is installed.

Localized versions of M-Business Client

Platform	English	French	Italian	German	Spanish	Japanese	Traditional Chinese	Simplified Chinese
Palm OS	Yes	Yes	Yes	Yes	Yes	No	No	No
Windows Mobile family*	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Windows XP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Microsoft Smartphone	Yes	Yes	Yes	Yes	Yes	No	No	No
Symbian OS	Yes	Yes	Yes	Yes	Yes	No	No	No

* Windows Mobile Pocket°PC, Windows Mobile°5, and Windows Mobile°6.

Character sets for HTML rendering

Platform	Windows-1252	ISO-8859-1	ISO-8859-2	ISO-8859-3	ISO-8859-4	ISO-8859-5	UTF-8	EUC-JP	Shift-JIS
Palm OS	No	Yes	No	No	No	No	Yes	No	No

Platform	Windows-1252	ISO-8859-1	ISO-8859-2	ISO-8859-3	ISO-8859-4	ISO-8859-5	UTF-8	EUC-JP	Shift-JIS
Windows Mobile family*	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes
Windows XP	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes
Microsoft Smartphone	Yes	Yes	Yes	No	No	Yes	No	No	No
Symbian OS	Yes	Yes	Yes	No	No	Yes	No	No	No

* Windows Mobile Pocket°PC, Windows Mobile°5, and Windows Mobile°6.

In addition to character sets in the table above, on Microsoft OS platforms only:

- Greek is now supported, using either ISO-8895-7 or Windows-1253.
- Cyrillic is now supported, using Windows 1251.

Character sets for M-Business XML conduit

Platform	Windows-1252	ISO-8859-1	ISO-8859-2	ISO-8859-3	ISO-8859-4	ISO-8859-5	UTF-8	EUC-JP	Shift-JIS
Palm OS	No	Yes	No	No	No	No	Yes	No	No
Windows Mobile family*	No	Yes	No	No	No	No	Yes	No	No
Windows XP	No	Yes	No	No	No	No	Yes	No	No
Microsoft Smartphone	No	Yes	No	No	No	No	Yes	No	No
Symbian OS	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

* Windows Mobile Pocket°PC, Windows Mobile°5, and Windows Mobile°6.

Note
M-Business XML conduit is not supported on Symbian OS.

Languages supported by character sets

The languages supported by the different character sets in the tables above are listed below. For additional information on Asian languages, see "Working with double-byte character sets (DBCS)" on page 62.

- **EUC-JP** Covers Japanese.

- **ISO-8859-1 (Latin°1)** Covers most western European languages: Albanian, Basque, Catalan, Danish, Dutch, English, Faroese, Finnish, French, German, Icelandic, Irish, Italian, Norwegian, Portuguese, Rhaeto-Romanic, Scottish, Spanish, Swedish. Also covers Afrikaans and Swahili.
- **ISO-8859-2 (Latin°2)** Covers most central and eastern European languages: Czech, Croatian, Hungarian, Polish, Romanian, Slovak, Slovenian, Sorbian.
- **ISO-8859-3 (Latin°3)** Covers Esperanto, Maltese, and Turkish.
- **ISO-8859-4 (Latin°4)** Covers Estonian, Greenlandic, Lappish, Latvian, Lettish, and Lithuanian.
- **ISO-8859-5 (Cyrillic)** Covers Bulgarian, Byelorussian, Macedonian, Russian, Serbian, and pre-1990 Ukrainian.
- **ISO-8859-7 (Greek)** Covers modern Greek.
- **Shift-JIS** Covers Japanese.
- **UTF-8 (Unicode)** Covers any language with Unicode support. iAnywhere recommends that you use UTF-8 if you can. For more information on the Unicode standard, see http://unicode.org/.
- **Windows-1251** Character set for languages Cyrillic, used by Microsoft operating systems.
- **Windows-1252** A superset of ISO-8859-1 that supports the same "Latin°1" languages. This character set is also known as Codepage 1252 or Windows Latin°1.
- **Windows-1253** Character set for modern Greek, used by Microsoft operating systems.

Working with double-byte character sets (DBCS)

Supported platforms

M-Business Anywhere Version 6.7 only supports DBCS on the following platforms:

- **M-Business Server** - all platforms
- **M-Business Client** - Microsoft OS devices only

Supported double-byte character sets

M-Business Anywhere Version 6.7 supports the double-byte character sets listed below.

Note
If at all possible, iAnywhere strongly recommends using UTF-8. This is the most global, most robust, most tested double-byte character set for use with M-Business Anywhere. All of the national character sets that are supported have known problems; for example, Shift_JIS causes some JavaScript commands to fail when certain JavaScript sequences are interpreted as Japanese characters instead of JavaScript instructions. If you plan character usage carefully with UTF-8, you may even be able to avoid the need to translate large portions of your content between different Asian languages.

- **UTF-8** - Unicode - recommended by iAnywhere
- **Shift_JIS** - Japanese
- **EUC-JP** - Japanese
- **Windows 932** - Microsoft standard Japanese
- **GB-2312** - simplified Chinese
- **EUC-CN** - simplified Chinese
- **Windows 936** - simplified Chinese
- **Big5** - traditional Chinese
- **EUC-TW** - traditional Chinese
- **Windows 950** - traditional Chinese

Specifying a character set with the Charset META tag

If your channel pages use a particular character set, such as a double-byte Asian set, or an Eastern European variety, you must ensure that your pages display correctly in M-Business Client by explicitly specifying the character set. You can do this in the `<META>` tag of each page:

```
<META http-equiv="Content-Type" content="text/html; charset=
charset_id
"... >
```

...where *charset_id* is the appropriate character set identifier, selected from the list above, under "Supported double-byte character sets" on page 62.

Specifying a character set with a Content-Type tag in the HTTP header

Instead of placing a `Charset` META tag on each page, you can set up your web server to include a `charset` attribute in the HTTP header it sends with your channel pages to M-Business Client. The details are similar in form to the HTML META tag.

The `Content-Type` entity-header field indicates the media type of the entity-body sent to the recipient, along with optional attributes like character set:

```
Content-type: <media_type>; charset=charset_id
```

...where *charset_id* is the appropriate character set identifier, selected from the list above, under "Supported double-byte character sets" on page 62.

The following example specifies HTML pages using the UTF-8 character set:

```
Content-type: text/html; charset=utf-8
```

Note
You can specify a global character set in your web server's HTTP headers, then override it with the `Charset` META tag on individual pages. The individual page's `Charset` META tag always takes precedence.

Avoiding problems with double-byte characters in XML data

There are three things you can to avoid problems with DBCS in XML data:

- Include a `deviceEncoding=utf-8` attribute in each element definition in your XSD file. See "Supporting double-byte character set (DBCS) data" on page 277.
- In your XML files, avoid use of DBCS outside of the XML data itself and the related labels. In particular, do *not* use DBCS in element names.
- In the HTML pages that refer to the XML data, specify UTF-8 as the character set in a `Charset` META tag (see "Specifying a character set with the Charset META tag" on page 63) or in an HTTP header (see "Specifying a character set with a Content-Type tag in the HTTP header" on page 63.

In addition, if you are using ISO-8859-1 encoded XML data, do *not* include the `deviceEncoding` attribute in the XSD file. Instead use ISO-8859-1 encoding in the HTML that refers to the data. The HTML encoding must be specified with META tags or with HTTP headers. HTML without encoding information will be processed by default as ISO-8859-1.

Available HTML in M-Business Client 5.x and later

M-Business Client 5.x supports a large subset of HTML 4, which adds the following to the HTML support of M-Business Client 4.x:

- Inline Cascading Style Sheets (CSS)
- Major expansion of JavaScript, called M-Business JavaScript engine

For more complete details on the HTML tags supported by M-Business Client 5.x, see Appendix "HTML 4 support in M-Business Client 5.x and later" on page 295.

Using the TITLE tag

On Palm OS devices, M-Business Client displays the title in the title bar of the application. M-Business Client truncates the title to the first 15 characters. Try to keep HTML page titles as short as possible. At least try to make the first 15 characters unique.

On Windows Mobile Pocket PC devices, M-Business Client does not display the page's title at all. The title bar contains the name of the application, "M-Business Client."

On both Palm OS and Windows Mobile Pocket PC devices, users can see the complete page titles if they go to the Channel Manager page and select your channel.

Using tables judiciously

Big tables are a nightmare for users trying to view them on the small screens of mobile device screen. But if you are careful, you can use tables to good effect in some situations.

Advantages of tables

Tables in HTML have several advantages that you are probably used to taking advantage of in desktop web design:

- Columns can resize automatically so the table always fills different screen widths.
- Cell borders and background colors can enhance readability.
- With M-Business Client 5.x and later you have the option of using JavaScript to do powerful things, like sort a table in order by a column when the user clicks the column header.

Disadvantages of tables

Tables also have distinct *dis*advantages on mobile devices:

- You can only squeeze in a small number of columns before the table width causes horizontal scrolling on smaller screens.
- Making columns narrow to prevent horizontal scrolling will decrease readability of text in cells, as a paragraph is stacked into one or two words per line.
- Page size is increased vs. the same content displayed without a table.
- Page rendering is slowed.

Guidelines for using tables

Whenever you are considering using a table, ask yourself these questions:

- **Can the content be better presented another way?**

  Consider alternative ways of presenting the same content. For example, you may have a table of names addresses and phone numbers. It's laid out with one row per person. This information would be better presented on a small screen in one column of regular text (with a `<BR>` tag at the end of each line) in which name, address, and phone number are stacked for each person, looking more like a mailing label.

- **Would a simulated table work just as well for the content?**

  If the content looks best in tabular form, would it work just as well to render it with a fixed width font, with columns of data aligned by filling the space in between with the appropriate number of spaces? The `<PRE>` tag does this quite nicely, as long as you provide the right number of spaces to maintain column alignment.

- **If you do use a table, try to:**

  - **Eliminate horizontal scrolling** on smaller screens. Be sure you test it on a small Palm OS screen and keep adjusting the table tagging as necessary until any horizontal scrolling is eliminated.

  - **Let columns size dynamically**, so that the table width can expand and contract to fit the available screen width.

  - **Keep it lean and simple** — for example, set parameters at the table level and avoid using *both* `CELLPADDING` and `CELLSPACING`, avoid nesting tables.

Using forms

Use of HTML forms is encouraged. The M-Business Sync Server is designed to handle all form submissions and requests for users from within your channel. This makes for ease-of use by your users. Never design forms with the expectation that the user will manage them using the Forms Manager. The Forms Manager is an advanced option not designed to for handling user-interaction with your content.

Mobile devices often have limited storage space, so it is crucial that you limit your content to what users actually want to see. One way to make sure you have done this is to use forms to ask users what they want to see. For example, financial sites use forms to have users set up watch lists of stocks they want to track. Weather sites use forms to have users enter the city and country they want the forecast for.

M-Business Client supports all of the form elements except the file input element. In addition, it supports the following:

- The scribble element (`INPUT type=scribble`) is supported on Palm OS. Users click this element to open a small window that allows them to scribble inside. It is useful for capturing signatures. For example:

  ```
  <INPUT TYPE="scribble" NAME="name" SIZE="width, height"
  ALT="Alternate Text">
  ```

- The push button element (`INPUT type=button`) introduced in HTML 4.0.

- The scripting hooks defined in the 4.0 specification: onChange, onClick, onFocus, onBlur, and so on.

A huge difference with your channel vs. desktop web design is that you can expect most forms to be submitted when the user is offline. This means that the user won't be able to get back the response page from the form submission until the next synchronization. Besides creating the forms to be used in your channel, you need to ensure that your offline form submissions are handled smoothly. For more information on how to do this, see "Managing channel form submissions" on page 173.

Designing pages for multiple devices

While a majority of your users may use the same type of device, there are probably a significant number of users out there who use a different type of device and you should design for multiple platforms. Windows Mobile Pocket PC devices generally have a larger screen than Palm OS devices. A few good rules to keep in mind if you want the same page to look good on both Palm OS and Windows Mobile Pocket PC devices:

- If you are trying to center graphics or text, use the <CENTER> tag. Do not pad on white space to your graphics so they look centered on a 150-pixel wide screen. While it might look centered on a Palm, it will look strangely shifted to the left on a Windows Mobile Pocket PC device.
- Similarly, do not assume that just because you have an image that is 150 pixels wide, it will take up the full width of the screen.
- Be careful about creating graphics that include text. Windows Mobile Pocket PC devices tend to have denser screens than Palm OS devices. The denser the screen is, the smaller the text will appear in a graphic.
- Test your site on both a Palm OS and Windows Mobile Pocket PC device.

Your site may have certain pages (like your home page) or graphics (like your logo) that you really want to look absolutely as good as possible on different mobile devices. With a little extra work, it is possible to do this. For instructions, see "Using M-Business Client HTTP request headers to customize content" on page 161 and "Using server URL macros to customize content" on page 166.

Personalizing pages for different users

You can use cookies to improve personalize the content delivered to each of your channel subscribers. Cookies work with M-Business Client and the M-Business Sync Server the same way they work with anybody else browsing the Web. The only exception being that cookies are sent through the M-Business Sync Server instead of being stored on the mobile device itself. This difference should not affect the way you set and receive cookies, except that you cannot access them using JavaScript. We suggest you use cookies for:

- Online ordering
- Site personalization
- Targeted marketing

- User IDs

For an in-depth look at cookies and how to use them in a sample M-Business channel, please see “Using cookies to support personal channels” on page 196 and “Using personal channels to bypass the Forms Manager” on page 185.

Using Form and JavaScript elements

We won't go into any detail about the Form or JavaScript elements in this chapter. For more information on those topics, see “Customizing content for users and devices” on page 159 and “Managing channel form submissions” on page 173.

Designing graphics for channel pages

How M-Business Client processes images

The original Palm OS mobile device had a screen size of 160x160, but a usable area of 150x150. The typical Windows Mobile Pocket PC device is a bit larger, typically 240x320 or so. Some current platforms, such as Microsoft Smartphone devices, have screens that are similar in size to the original Palm device. Other current devices, such as Tablet PCs running Windows°XP, have screens that are almost as large as a desktop PC. M-Business Client follows the rules below in processing each image for a particular device type.

- **Pixel packing** — M-Business Client compresses the image to save both download time and device memory.

> **Note**
> On Palm devices, this may actually result in GIF images that are larger than the original. You can use this formula to calculate an image's actual size on a Palm OS device:
> width-in-pixels × height-in-pixels × pixels-per-byte

- **Resizing** — M-Business Client *may* resize the image, making it smaller.
  - **Pages without `HandheldFriendly` META tag** — M-Business Client will resize any image that is larger than the maximum screen size for the device type. Avoid creating images that are larger than 150x150, if you want to support the older Palm OS devices; although M-Business Client will resize them, there is no point in serving a really large graphics file only to have it shrunk down later.
  - **`HandheldFriendly` pages** (with `HandheldFriendly` META tag on page) — M-Business Client will not resize the image to fit the screen size. In most cases, you should still avoid creating images that are larger than 150x150, if you want to support the older Palm OS devices.
  - **Palm database (PDB) limit** — on Palm OS devices, M-Business Client will resize any image that is larger than the maximum PDB limit — 32 KB on Palm°OS°3.2 and earlier, 64 KB on later Palm OSes.

To get around any of the resizing constraints, see “Breaking maximum image size limits” on page 75.

Photorealistic images and line art

For **photo-realistic images**, you do not need to do anything special with them besides re-size them. M-Business Client does a pretty good job of dithering the image for the appropriate device. You might want to consider sharpening the image using something like Photoshop's Sharpen filter, which tends to emphasize the lines between contrasting areas. (But do not expect miracles: There is only so much we can do with a 4-color grayscale display.)

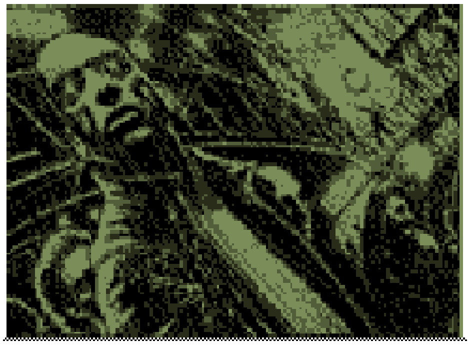
For **line-art**, such as logos and maps, the situation is more complicated. Dithering does not look as good with line art as it does with photographic images. And while there are many devices (like the Palm III) that support only 2-bit, 4-gray images, there are others (like the Palm V) that support 4-bit 16-gray images, while others (like Windows Mobile Pocket PC machines and the Palm IIIc) use 8-bit color.

There are two strategies one can use when approaching line-art images:

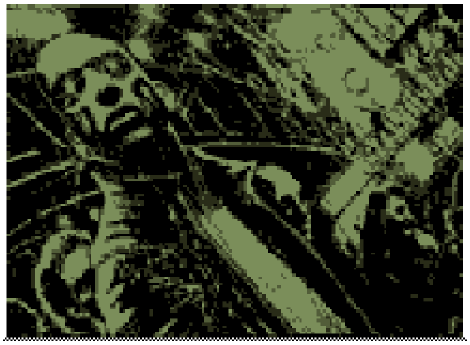
Strategy 1: Design for the lowest common denominator and optimize your images for 4-grayscale displays.

The best way to optimize line-art for 4-color gray is to convert the image to grayscale, use the **Image»Adjust»Posterize** command in *Photoshop*, and select 4 levels of gray. This will adjust every color in the image to the closest non-dithering gray. Should you choose this strategy, keep in mind you will probably have to re-design your images at some point in the future as color and 16-gray devices become more prevalent.

For example, this is how a line-art drawing looks when viewed on a 4-gray Palm III device. The version on the left is the original image without any modifications, the version on the right is the original image posterized to 4 levels of gray. The version on the right looks a little sharper than the image on the left, although some subtle detail is missing.



Line-art, not posterized

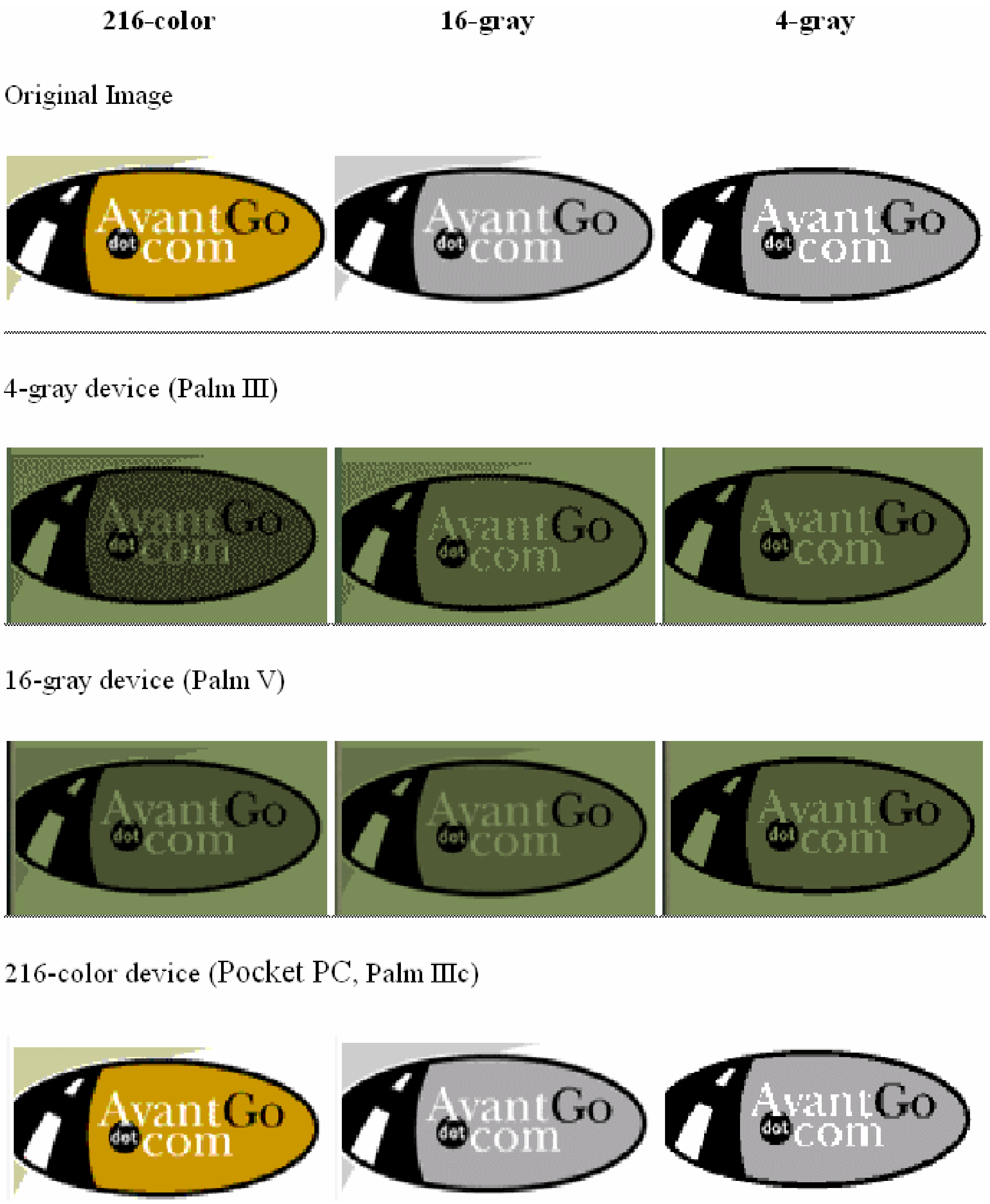


Line-art, posterized

Images are copyright 1999 by Jason Felix. Not to be redistributed without permission.

Strategy 2. Design for the best quality on the more advanced devices. The best way to optimize line-art for color is to convert it to the 216-color web palette, as that is the palette that the M-Business Client browser uses. Most graphics programs give you the option to save your image as a 216-color Web-safe palette image.

As a quick demonstration, this is how our logo looks as a 216-color Web-safe image, a 16-color posterized grayscale image, and a 4-color posterized grayscale image. Not surprisingly, the 4-gray image looks best on the 4-gray device, the 16-gray image on the 16-gray device, and the 216-color image on the 216-color device.



216-color 16-gray 4-gray

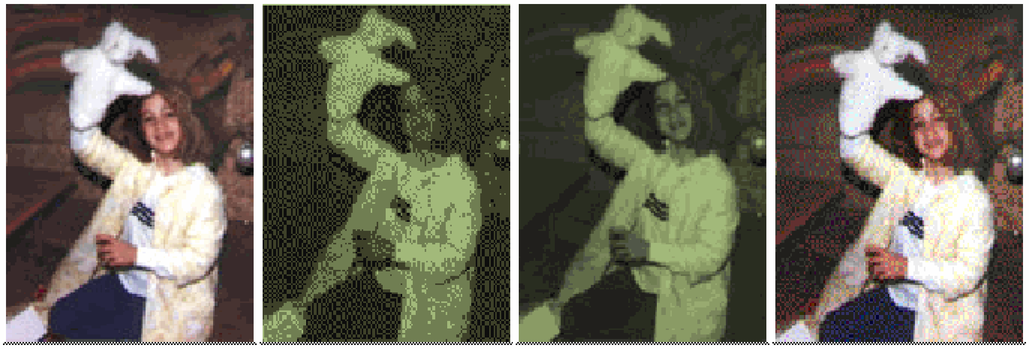
Original Image

4-gray device (Palm III)

16-gray device (Palm V)

216-color device (Pocket PC, Palm IIIc)

The figure below shows how a typical photo looks on the various devices.



Original Image

Image on a 4-gray device (Palm III)

Image on a 16-gray device (Palm V)

Image on a 216-color device (Pocket PC, Palm IIIc)

It is also possible to deliver different content depending on the screen size and color depth of the device, which can provide the best quality image on all platforms. For more information, see “Using server URL macros to customize content” on page 166 and “Using M-Business Client HTTP request headers to customize content” on page 161.

Of course, the real key to increasing the quality of your channel is to test it thoroughly. See “Testing your M-Business channel” on page 37.

Image size

When choosing images for use on the mobile web pages, it is important to remember the size of the display. As stated previously, mobile devices come with a variety of screen sizes. Once the screen is painted with the title bar and scroll bar, you are left with an even smaller usable area. The following table gives some examples of screen widths for various supported devices and how much of the screen is available for displaying HTML pages:

Table 1. Screen sizes

Device	Screen size	Usable area
Largest HPC Pro devices	800 X 600	740 X 585
Largest HPC devices	640 X 240	580 X 226
Typical palm-sized PC	320 X 240	260 X 225
Palm Computing device	160 X 160	150 X 150

If you attempt to display an image that is larger than 150 pixels in either dimension, your M-Business Sync Server will scale it to fit inside the viewing area. Scaling reduces the quality of many images, so you might consider avoiding images wider than the smallest supported viewing area (which is on the Palm Computing devices).

Remember, M-Business Client does not support the `height` and width `attributes` of the `IMG` element, so you cannot rely on the browser to resize an image to your specifications.

Image detail

It is good practice to avoid large, detailed images because the scaling operation may obscure critical information. If there is a convenient way to convey the same information without using an image, you are probably better off dispensing with the image altogether. If you must use detailed images, however, you should tailor them especially for mobile devices in order to achieve a predictable result.

Image colors

M-Business Client displays images in color on devices that have color displays. On grayscale or monochrome devices, M-Business Client converts the color images to grayscale. The original color scheme of an image plays an important role in the way M-Business Client renders it on grayscale and monochrome devices. The process we employ to convert color images to black and white ones works best on images that have a high contrast ratio between colors and that have crisp edges in the details of the picture.

Guidelines for color images on color devices

M-Business Client supports color for both Windows Mobile Pocket PC and certain Palm OS devices, using the palette supported by the device. If your image includes a color that is not part of the device palette, the color will be dithered.

Tips for color images:

- Just as important as the quality of the color images is the size of the images. Keep images small.
- Save photo-realistic images as JPEGs (they tend to compress the best).
- Make sure the images are the appropriate dimensions.
- Other than resizing the image, it is not necessary to alter photo-realistic images in any other way. M-Business Client does a good job of dithering the image to the 216-color palette.
- Convert line-art, such as maps and logos to the Web-safe 216-color palette and save it as a GIF. Line art looks best when not dithered. For images that are not dithered, GIFs compress most clearly.

Guidelines for color images on grayscale devices

Not all mobile devices support color images, of course. Many of the older Palm devices support only 4-color grayscale. While you cannot expect amazing image quality with four grays, images still come out looking fairly decent. Some people prefer to cater their images to this lowest common denominator. If so, here are some tips for creating images for grayscale devices:

- Just like with color devices, you only need to resize photo-realistic images and save them as JPEGs. Let your M-Business Sync Server dither the image down to the appropriate color depth.
- With some photos, however, it helps to use Photoshop's Unsharp Mask, which sharpens the image and emphasizes boundaries between two different color areas.
- For line-art images, such as maps and logos, use Photoshop's Posterize command to reduce the image down to 4 non-dithering grays.
- Before using the Posterize command with line-art images, it is usually a good idea to also use Auto Levels to make sure your image's levels are distributed evenly.

For examples of various images displayed on various devices, see “Designing graphics for channel pages” on page 69.

Designing images

The easiest and most predictable way of deploying images for mobile devices is to make your own custom images (both JPEG and GIF are supported) that take into account the reduced screen size of mobile devices. In general, it is good practice to design your images with clean lines and simple uncomplicated elements tend to appear ragged. It is better to use an image that is deliberately simple rather than try using a complex image that looks confusing. Keep in mind what is appropriate for the medium. If you want to look at high-resolution images, save them for the desktop and reserve your mobile web pages for less graphical information.

ALT tags

Many users may choose not to load images, so it is extremely important to place meaningful alternate text tags in each of your embedded images. When creating ALT tags, keep in mind that you are trying to convey the message of the picture, not describe it. Therefore, putting Yellow button as an alt tag for a yellow button is not nearly as effective as using an * instead.

Image maps

M-Business Client does not currently support image maps. You can simulate them, however, by splitting your image into sections and assigning a different anchor to each one. This way, you can have horizontal navigation bars that act just like their image map equivalents. Just remember to keep your images small and simple. Also, keep in mind that the image should give the user some indication of its function. Remember that you do not have the benefit of the cursor turning into a hand when it is over a link.

Breaking maximum image size limits

Note
If your pages include the `HandheldFriendly` META tag, your images will not be resized to fit within the maximum screen size for the platform. However, on Palm OS devices, independent of screen size, images will be resized if it is necessary to fit within the Palm database (PDB) limit — see "How M-Business Client processes images" on page 69. On those devices, you could use this technique to get around the PDB size limit.

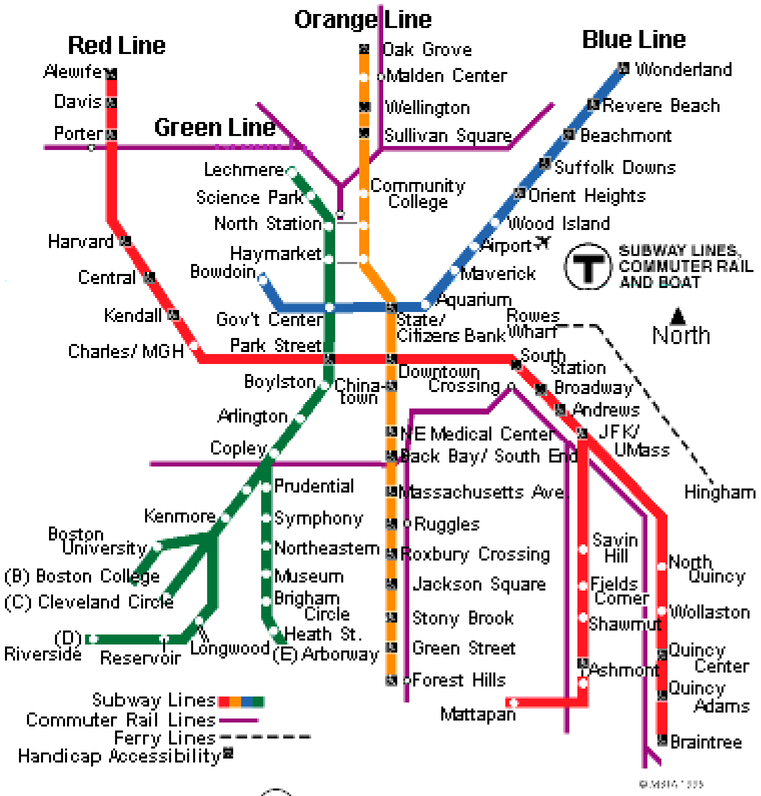
Before you continue, we should point out that the technique below has two bad side effects:

- You are going to force many of your readers to scroll horizontally, which they generally dislike, and
- If you go overboard with this technique, it is probably going to cause memory problems.

So I would recommend doing this only for your own personal custom channels or if you are absolutely sure your users really want this. In other words, do not use this technique to create a super-huge banner ad.

The original oversized image

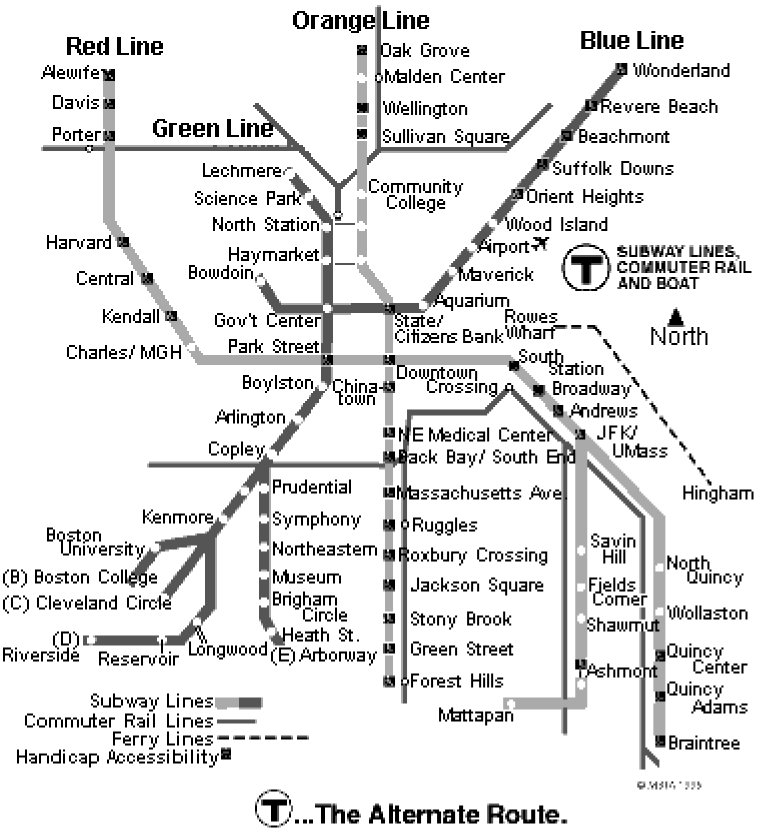
Suppose that you wanted to take this map of the T (the Boston subway system) and put it on your mobile device. For this example, we will assume that you want to stay within the 150-pixel screen size limit to support older Palm OS devices.



T...The Alternate Route.

Image Copyright 1999 by the Massachusetts Bay Transit Authority. Not to be reproduced without express written consent from the MBTA.

The first thing you would do, of course, is convert it to greyscale and posterize it down to a 4-gray image so that it will not dither on a 4-gray device like a Palm III.

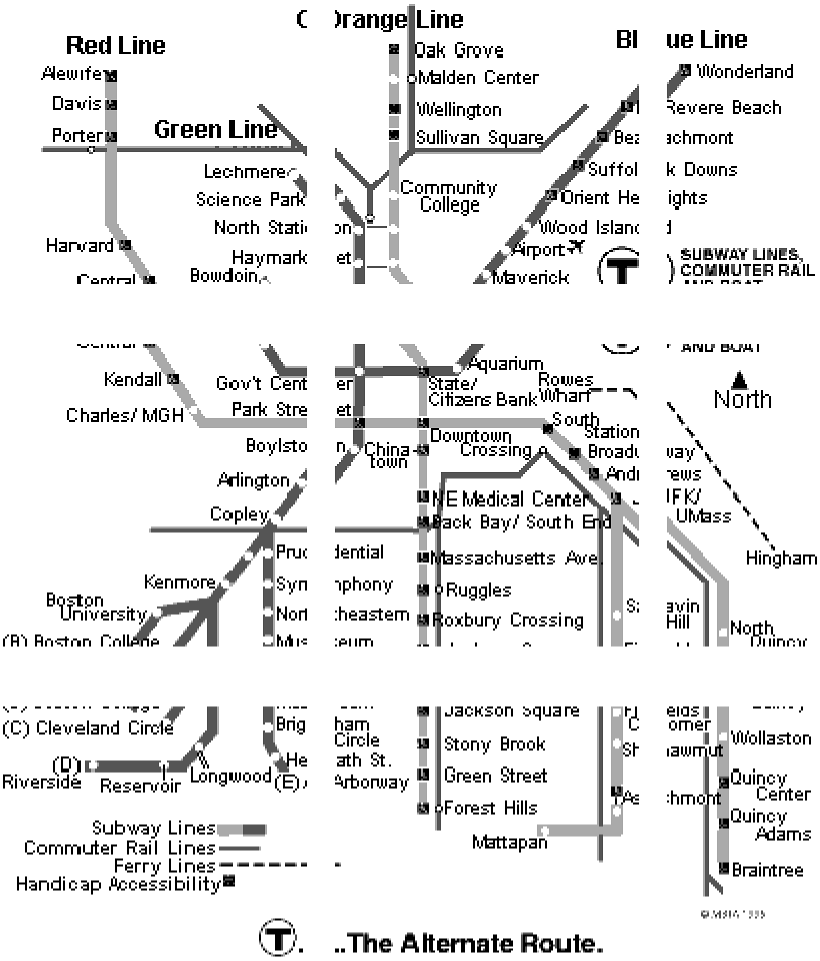


Right now, the image is 381x426 pixels — that is way over our 150-pixel in either dimension limit, so you are going to have to shrink it down (and if you do not, your M-Business Sync Server will do it for you). This means you will end up with a 134x150 graphic that looks something like this.

This clearly is not good. None of the station names in the image are readable.

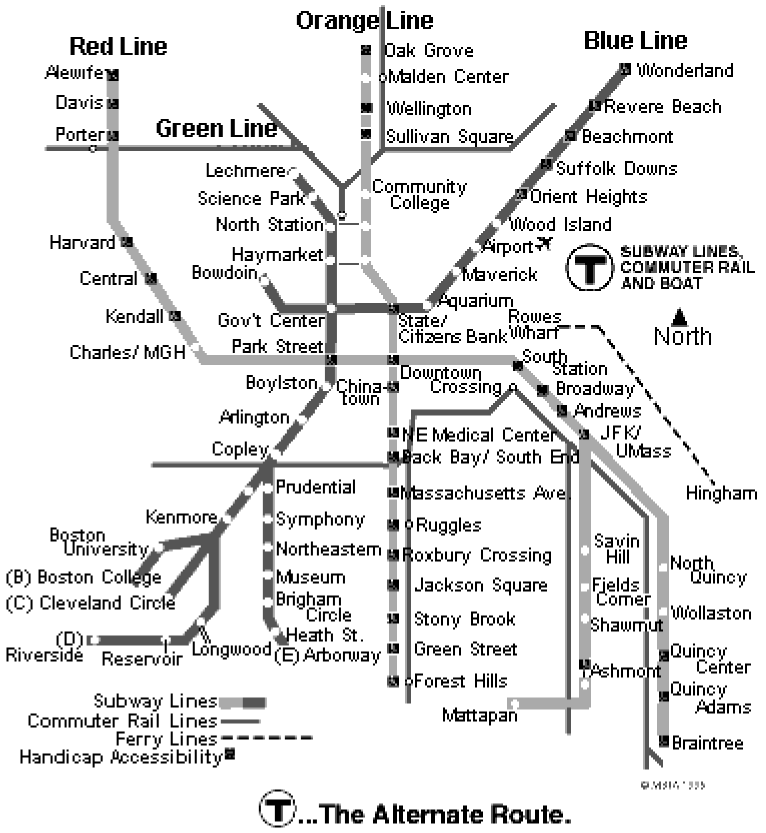
Breaking up the oversized image

While it is true that your M-Business Sync Server will not accept any image larger than 150 pixels, that does not mean you cannot give readers the illusion that it is a whole lot larger. First, with your favorite graphics-editing program, break this image up into several chunks that are no larger than 150 pixels in any side.



Re-assembling the image

Now all you have to do is create a table with `border=0`, `cellpadding=0`, and `cellspacing=0`. Then place the different chunks of your image in that table.



When you look at the image in M-Business Client, it will look like one large image. Now I will be able to look at my Palm and navigate the Boston subway flawlessly! (Except for the fact that I live in San Francisco, this would be really convenient.)

Easy ways to break up images

This looks really tedious to do by hand! Is there an easy way to break up images?

Both Adobe's *ImageReady* and Macromedia's *Fireworks* have Image Slice tools. These will easily let you split up an image into several parts, which you can then save into a table. Both programs will create the HTML for the table, as well as create several image files (one for each of the image chunks) and place them in a folder of your choosing.

If you are generating images dynamically, there is probably a way to automate this process. Perl and PHP will let you alter JPEG files if you have the GD graphics library. And if you are writing Java servlets, the `drawImage()` and `clipRect()` functions should prove quite useful.

Synchronizing only forms or M-Business XML conduit

If you want to do so, with M-Business Client 5.0 and later you can create a form that causes only the channel's forms or M-Business XML conduit to be synchronized. The HTML below shows how to do this.

```
<form type=post action=agreserved://limitedchannelsync>
<input type="any" name="channel_id" value="channel_database_id">
<input type=button name="Submit" value="Sync"
onClick="avantgo.beginSync();">
</input>
```

CHAPTER 4

Caching to improve channel performance

Contents

Overview of caching

If you anticipate anything more than a handful of users subscribing to your channel, you should definitely consider setting up caching for your channel's pages.

Caching can significantly reduce the time it takes your readers to download your channel when they synchronize. The higher the volume of channel traffic — number of subscribers times size of channel — the greater the reduction in download time that caching can potentially provide.

If you are not sure that caching is worth the trouble, be sure to read "Introduction to caching" on page 83. Substitute your own projections for the numbers of downloads you expect on your channel. Unless you expect a very small number of subscribers, you will probably conclude that caching *is* worth the trouble!

Note
If you do decide to implement caching, *wait* until you are finished testing your channel to set it up.

Introduction to caching

The most significant change you can make to your page, performance-wise, is setting it up for caching. And this is what is going to keep your readers around. People can be impatient, and even if you have the most beautiful channel in the world, your readers are going to ditch it if they have to spend several minutes waiting for it to finish updating. That is valuable time they could be spending elsewhere.

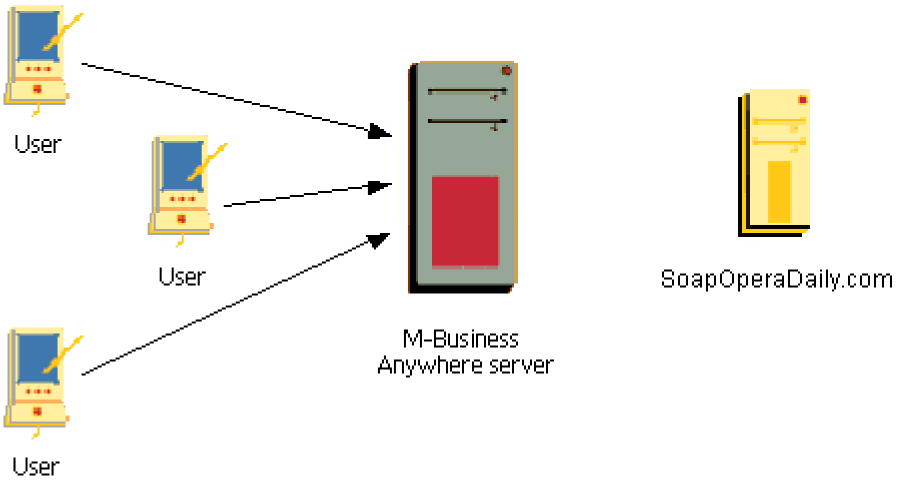
Channels often neglect caching. Partly because it is a little esoteric, and partly because it involves a little excursion into the world of servers. But in the long run, it *is* worth it.

How caching works — the concept

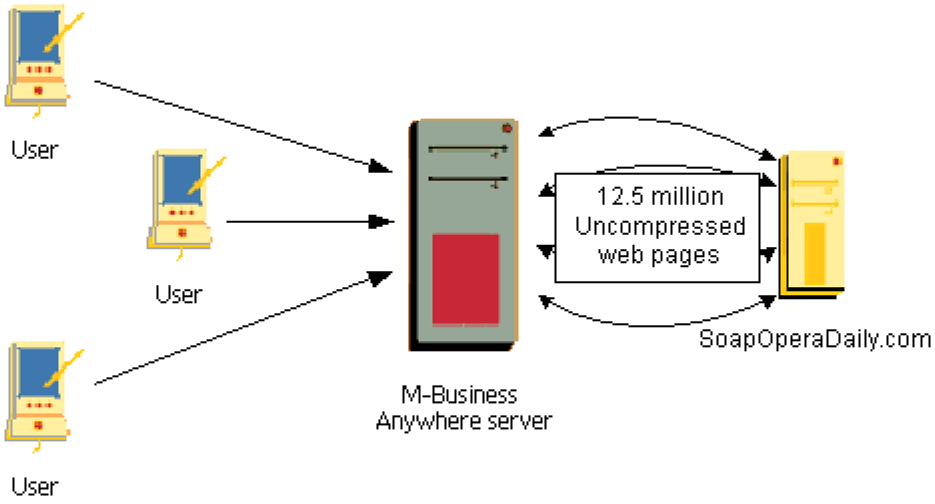
Assume you are working on a Soap Opera Daily channel that consists of 25 individual pages, each of which are updated every day at midnight. It is a very popular channel, and you have also got a subscriber base of half a million people. Your customers, wanting to get the latest Soap Opera news every day, are updating their channels daily like clockwork.

Here is what happens if you do not have your channel set up for caching:

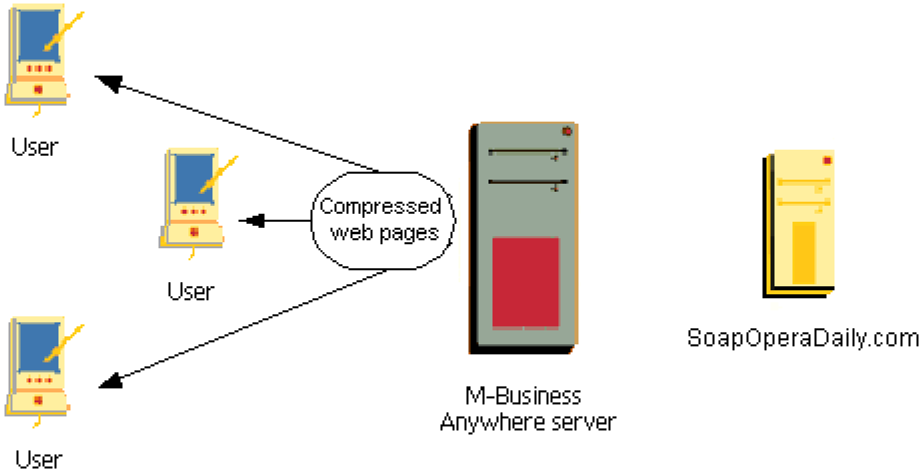
Step 1: Every day, half a million people ask your M-Business Sync Server for updates to your channel.



Step 2: Your M-Business Sync Server hits your home page, grabbing 25 individual pages half a million times a day, meaning your server is dishing out 12.5 million pages a day just to your M-Business Sync Server.


User
User
User
12.5 million Uncompressed web pages
M-Business Anywhere server
SoapOperaDaily.com

Step 3: The M-Business Sync Server compresses these files and uploads them to the users' Palm OS or Microsoft OS device.

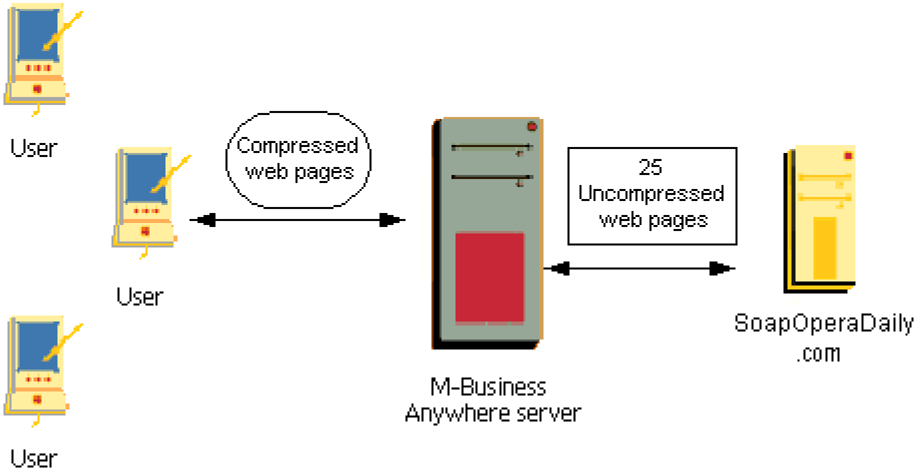


This is a pretty slow process. Your server is probably struggling under the weight of dishing out 12.5 million pages a day just for your M-Business channel. Worse, your readers are waiting a long time for their channels to synchronize because your M-Business Sync Server first has to get the pages from your overworked server, compress them for better viewing on M-Business Client, and then upload them to their mobile device.

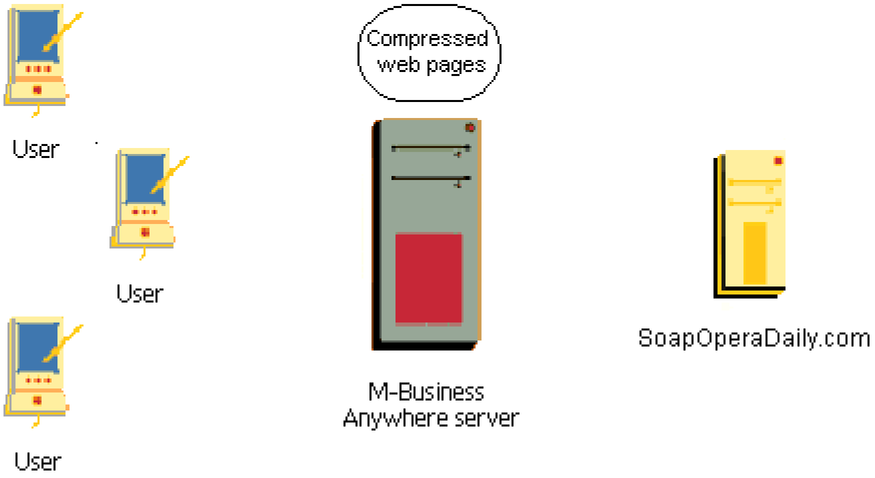
This is what happens if you do set up your page for caching:

Step 1: You specify that all the pages on your channel are good for, say, 24 hours.

Step 2: Each day, the first time a user asks to update the channel, your M-Business Sync Server hits your home page, grabbing 25 individual pages. It compresses them, and uploads them to the user's M-Business Client.



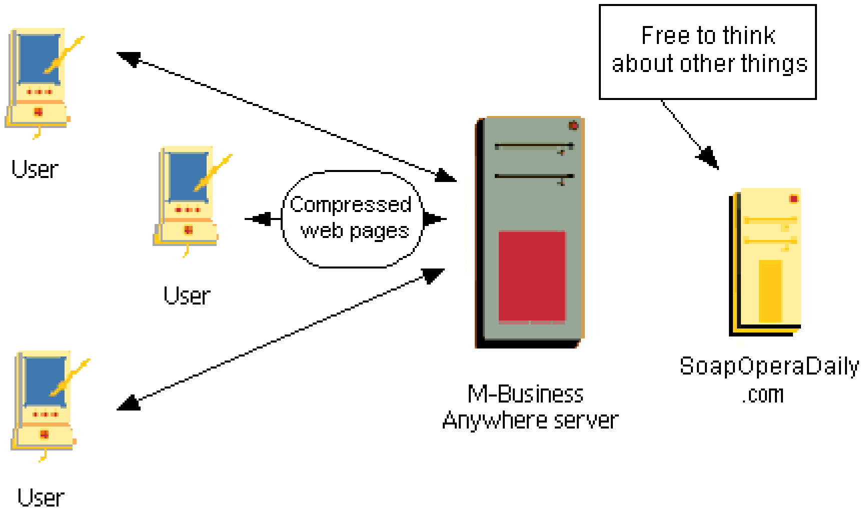
Step 3: The M-Business Sync Server holds on to those pages (already compressed) for future use.



Step 4: The remaining 499,999 times a user asks to update the channel, the M-Business Sync Server simply sends off the pre-compressed, pre-loaded channel pages to users' Palm OS or Microsoft OS devices.

Note

This is an oversimplification. The M-Business Sync Server caches different versions of the compressed page depending on the M-Business Client device type (Palm OS vs. Microsoft OS). So it is not actually a savings of 500,000 to 1. It is more like a savings of 500,000 to 60. Which is still pretty good.



Your server is happy because its only working to push out 25 pages a day instead of 12.5 million. Your users are happy because they can get pages directly from the M-Business Sync Server instead of having to wait for that server to retrieve them from your web server and then compress them.

How caching works — technical details

The M-Business Sync Server works very much like any other proxy server. If you are not familiar with proxy servers, that is OK. Read on.

HTTP headers

When a web server sends a page, whether to an ordinary browser, or your M-Business Sync Server, it first sends a group of HTTP headers that look something like this:

```
HTTP/1.1 200 OK
Date: Fri, 21 Jan 2000 17:28:02 GMT
Server: Apache/1.3.9 (Unix)
Cache-Control: max-age=18000
Last-Modified: Fri, 21 Jan 2000 17:27:09 GMT
ETag: "0-3cb-38888fe5"
Accept-Ranges: bytes
Content-Length: 1971
Connection: close
Content-Type: text/html
```

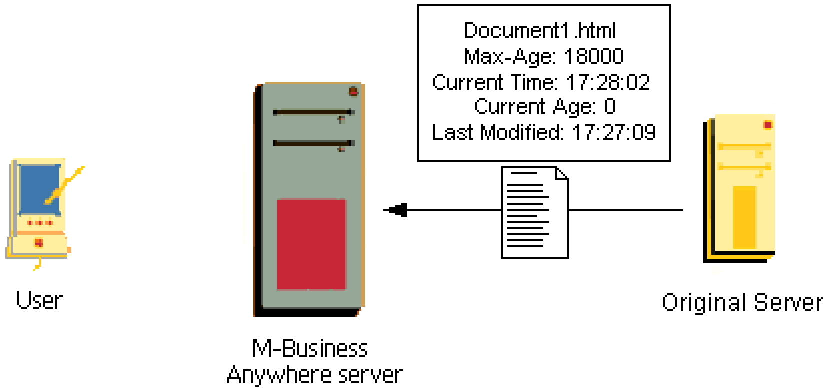
The Cache-Control header and the max-age directive

Most of this is of little importance right now (although some of it is interesting), except for that one line that reads `Cache-Control: max-age=18000`. The M-Business Sync Server uses this line to determine how old that web page is allowed to be before we have to search for a fresh version. The time listed there is in seconds, so these headers are telling the M-Business Sync Server that in 18000/3600 = 5 hours, we should consider the associated web page out of date.

Note
As you will see, there are other ways to control caching, such as setting up an `Expires:` header, but we prefer using `Cache-Control` instead.

Suppose that the M-Business Sync Server is grabbing a page from your channel with the above headers. It would notice the `Cache-Control` header, and keep the document on its hard drive. It would also note the current time and the document's `Last-Modified` date.

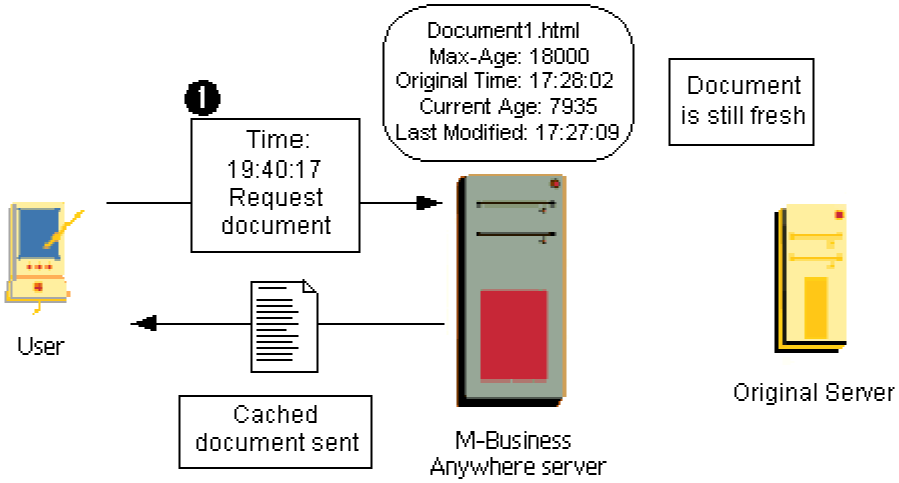
Note
Technically, we add on a few seconds depending on how long it took us to retrieve the document, but this is close enough for the purposes of instruction.



Calculating a page's age

For the purposes of determining when a web page is born, we look at the time we first access a document; not the document's `Last-Modified` date. This means that if you create a page at 10:00 with a `max-age` of 1 hour, but nobody requests it until 10:04, it will expire at 11:04 instead of 11:00. If you want to be incredibly strict about when your web page expires, use an `Expires` header instead of a `max-age` value. But this can often be problematic — see "Using Expires headers" on page 94.

Suppose that later, a different user requests that same document from your channel. Your M-Business Sync Server knows it has that page locally on its hard drive. The first thing it will do is calculate the page's age. The local time is 19:40:17 GMT, and your page was born on 17:28:02, so the page's age is 2:12:15, or 7935 seconds old.



Determining if the page is still fresh

The M-Business Sync Server compares this value to the `max-age` value. Since 7935 is less than 18000, the page is considered fresh and will be uploaded to the client without having to contact your server over the Internet.

Note
Actually, we do a second level of caching here. The client on the mobile device also keeps track of the document's age and its `max-age`. If the version of the document on the mobile device is still fresh, we do not even bother requesting the page in the first place. If we get a new version of the page, but a checksum reveals that the two pages are exactly alike, we do not upload the same document again. This saves the user a lot of time, as uploading data to the mobile device can be a slow process.
This cache stored on the client is known as the *client cache*. The cache on your M-Business Sync Server, which is used by many different clients, is called the *shared cache*.

Suppose a few more hours go by, and once again, somebody requests this page from your channel. The local time is 22:36:44. Now, that page is 5:08:42, or 18522 seconds old. 18522 is greater then 18000, so the page is considered stale.

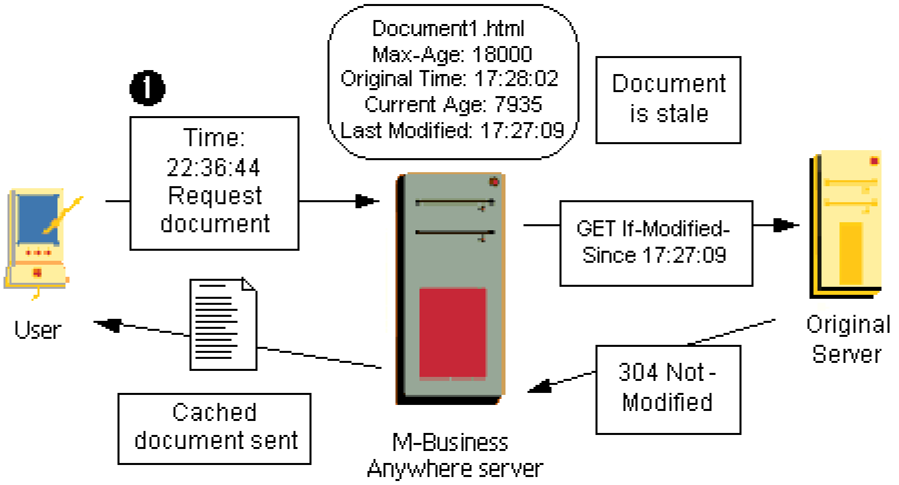
Determining if a newer page is available

At this point, we have to go out on the Internet and see if there is a newer page available on your server. We do something rather clever where we ask if the page has been modified since the `Last-Modified` date we have for our local copy. This is done using the `GET If-Modified-Since` command, for those of you familiar with HTTP 1.1.

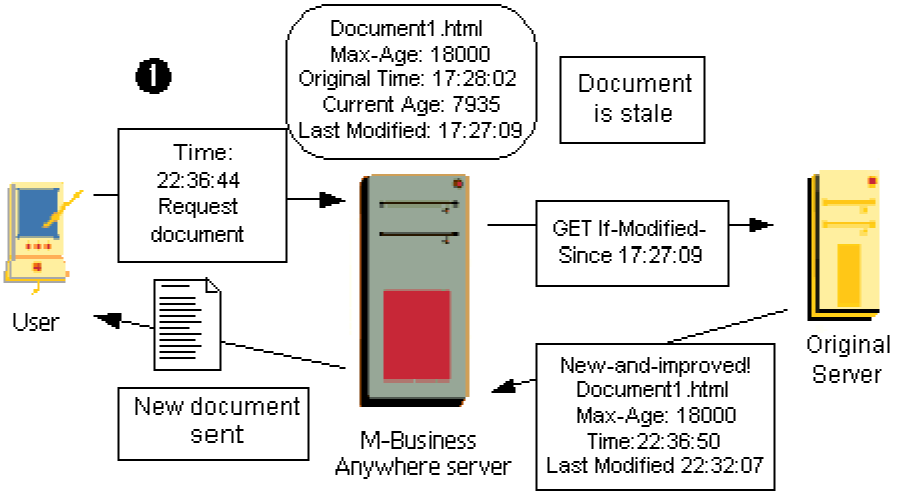
If the page has not been modified, the server on the Internet will tell us that the page has not changed. This comes in the form of a `304 Not-Modified` message. If this is the case, then even though your M-Business

Sync Server's copy of the page is stale, it is still the most recent, and we serve it anyway. By doing so, we avoid loading the entire page over again from your web server. All we need to receive is the little 304 message telling us that nothing has changed.

Note
Not all web servers add a `Last-Modified` date by default. If your web server does not, you should configure it so that it does.



If the page on the Internet *has* been updated, then we go out and grab the newer, fresher version of the page, note the time, and then use this newer page in our cache.



What about those desktop browser caches?

Is this the same thing as the local hard-drive cache that Netscape or IE uses?

No. The cache that you have on your hard drive is similar to the client cache that is stored on the mobile device. But they both function differently than the shared cache that is used on your M-Business Sync Server.

Seeing the HTTP headers for a page

How do I see those HTTP headers for *my* web page?

The easiest way is to Telnet to your web server using port 80. So, you could type `telnet www.mywebsite.com:80` from the **Start»Run** menu in Windows, or use your own favorite `telnet` client.

Then enter:

`GET /somedirectory/somepage.html HTTP/1.1` (Enter)

`Host: www.mywebsite.com` (then hit Enter twice)

You should get the HTTP headers for that page, followed by the HTML, and then the server will disconnect. It is probably worth saving the session to a log file or having a big buffer, so you can see the headers at the beginning of the file.

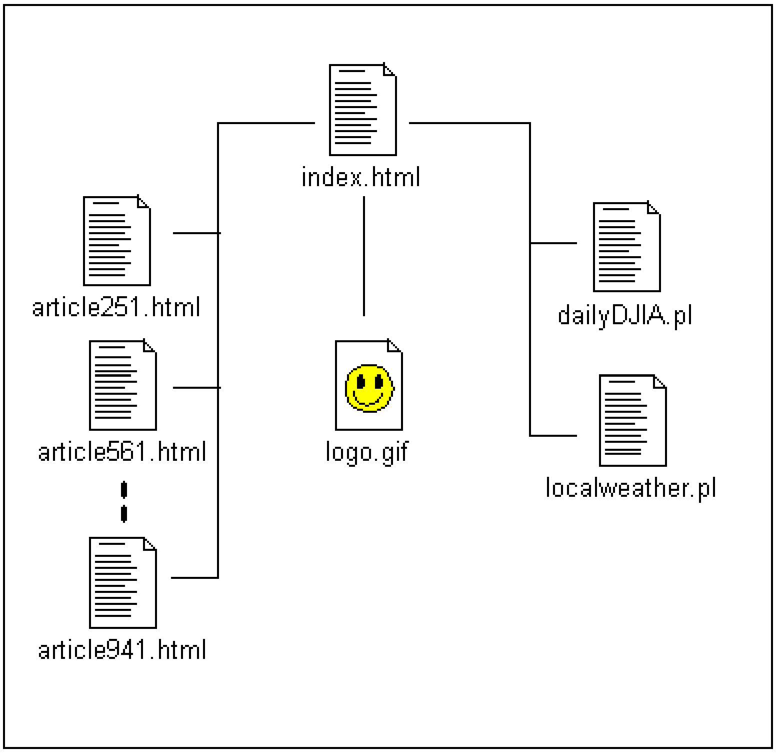
Caching strategies

This section examines some general caching strategies, before we get into implementing caching.

When you set up a web site for caching, all you are doing is specifying how long you want each page to exist on the proxy server. The more time you think will go by before a page is altered, the longer you can keep it cached.

Cache a page for less time than the period in which it changes

Consider a hypothetical news site. It has a front page (which we will call *index.html*) that includes a logo, and links to several articles. The front page changes daily, occasionally adding new articles and removing old ones. It also includes a financial page that displays the Dow Jones Average (updated every 20 minutes), and a personalized page that gives the weather report for a reader's zip code.



How would you want to cache this? Consider each section:

The logo is not going to change day-to-day. Or week-to-week, for that matter. You can safely give it a cache time of at least a week, and you could probably make it more, if you really wanted.

Your articles are not going to change, either. Sure, you will be putting up new ones every day, and taking down old stories after a while, but while they are up on your channel, they will still be the same old articles. So you can give them a cache time of 1 week as well.

The front page changes every day, so you do not want it to cache for more than 24 hours. And if there is late-breaking news, you will want this page to update at least a few times a day. A caching time of 1 hour is good.

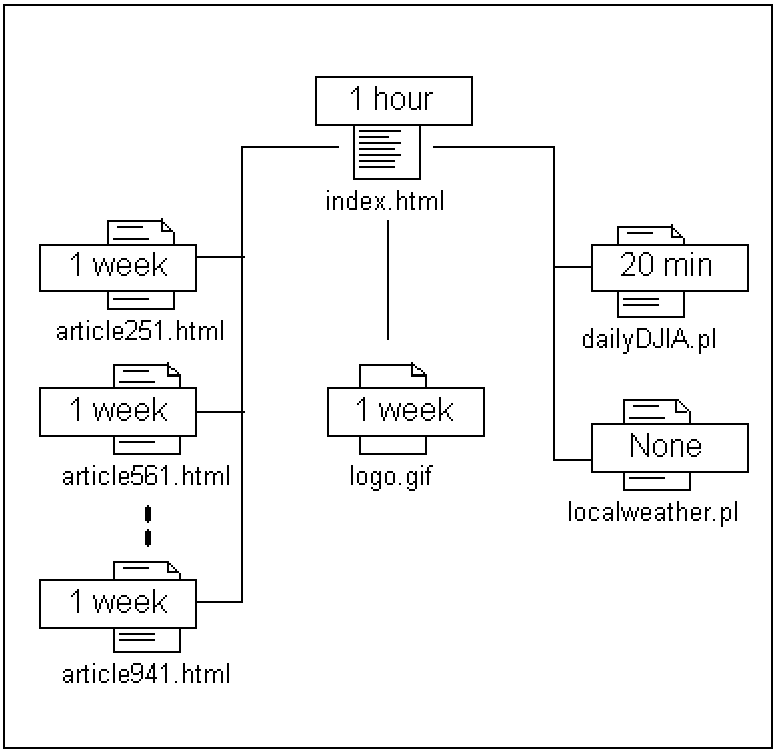
Note
You will get dramatic performance improvements by increasing the caching age to 1 hour. After that, the improvements are not quite as dramatic.

The Dow Jones Average page would have a caching time of 20 minutes. If you felt like being especially clever, you could write a script to increase its caching time to several hours once the market closes.

Do not cache personalized pages

Finally, the personalized weather page would not be cached at all — at least, not on your M-Business Sync Server. This might seem strange, given that your weather reports probably do not change more than once a day. But if you are creating personalized content, the page is going to change from person to person, so the cached version on the M-Business Sync Server is essentially useless. After all, you do not want to be accidentally giving somebody in Minneapolis the weather report for Phoenix.

You do want the personalized weather page to be cached on users' mobile devices. For more information, see “Caching personalized content on user's device” on page 99.



Now that you understand caching strategies, the next section looks at how to implement caching on an M-Business channel.

Implementing caching

There are two ways to implement caching. Both of these involve adding content to your HTTP headers.

Using Cache-Control headers

The first way, and the one we generally recommend, is to add the `Cache-Control` header mentioned above. The general format is:

```
Cache-Control: max-age=num_seconds
```

where *num_seconds* is the amount of time in seconds you want the page to last before it expires.

Here is a handy table for those of you who do not have a calculator readily available.

Table 1. Number of seconds in different time periods

Time period	Number of seconds
1 hour	3600
12 hours	43,200
1 day	86,400
1 week	604,800
1 month	2,592,000

Using Expires headers

Alternately, if you do not want to use `Cache-Control` headers, you can set an `Expires` header instead. An `Expires` header is useful if you need to be really specific as to when your page expires. The general format for an `Expires` header is:

```
Expires: Tue, 25 Jan 2007 10:30:00 GMT
```

The date and time must be presented in the HTTP date format, and it must be in Greenwich Mean Time.

Why we recommend against using Expires headers

Although the `Expires` header looks like it might be better at first glance, there are several reasons we recommend you use the `Cache-Control` headers instead:

- You have to be careful with the `Expires` header that all your pages do not expire at the same time.

If all your content expires at once, anybody synchronizing at that time will suddenly hit your server instead of the M-Business Sync Server's cache and you will experience a dramatic spike in traffic on your server. This spike generally lasts about a minute, and we do some work to help minimize the spike, but you will still get it.

- At least with Apache, it is a little easier to set specific `Cache-Control` headers for specific pages.
- There are some nice customizations you can add on to the `Cache-Control` headers (which we will detail below).

Using META HTTP-Equiv tags

An alternative to the `Cache-Control` HTTP header is to use `META HTTP-Equiv` tags. These tags produce the same result as the equivalent HTTP header when used within your channel pages and are processed by the M-Business Sync Server and displayed in M-Business Client. We strongly recommend that you use HTTP headers instead of `META HTTP-Equiv` tags if at all possible. If you are unable to use HTTP caching headers because your server does not support them or you do not have the necessary administrative privileges to enable them, you can use `META HTTP-Equiv` tags as an alternative method to specify caching on your pages.

Caution
Proxy servers generally ignore `META HTTP-Equiv` tags, and some browsers also do not support these tags. If you serve the same content to both your M-Business Sync Server and other web clients such as desktop browsers, by using `META HTTP-Equiv` tags you may not get the same benefits of caching for these other clients that you would if you used HTTP headers.

`META HTTP-Equiv` tags have the following form:

```
<META HTTP-EQUIV="name" CONTENT="content">
```

Where *`name`* is the name of the HTTP header, and *`content`* is the parameter string that you would use with the HTTP header.

For example, the following `Cache-Control` header:

```
Cache-Control: max-age=604800
```

Would translate to the following `META HTTP-Equiv` tag:

```
<META HTTP-EQUIV="Cache-Control" CONTENT="max-age=604800">
```

And the following `Expires` header:

```
Expires: Tue, 25 Jan 2000 10:30:00 GMT
```

Would translate to the following `META HTTP-Equiv` tag:

```
<META HTTP-EQUIV="expires" CONTENT="Wed, 26 Feb 1997 08:21:57 GMT">
```

Note
Like all `META` tags, `META HTTP-Equiv` tags must be placed in the `HEAD` section of the page.

What about META Expires tags?

I've used `<META Expires=...>` tags before. Can I somehow use these for your M-Business Sync Server?

No. The `META Expires` tags are used only for certain browser caches, like IE and Netscape.

Adding HTTP headers to your channel pages

So how do you go about actually adding these headers to your documents? Unfortunately, there is no single answer because it depends on how your pages are being created and how your documents are being served. While I can not really get into full detail on every case, I will cover the most common case, which is using static pages on an Apache web server.

If you are using a server other than Apache, for implementation information specific to your web server, be sure to check out Mark Nottingham's caching tutorial at:

http://www.mnot.net/cache_docs/#IMP-SERVER

If you are a Webmaster, with full access to all the Apache files, the easiest way to control caching is to edit your *httpd.conf file* (located in your conf directory). Within that file, use a combination of `<Location>` directives and `Header` directives to add or remove headers. The location directive allows you to specify the URL of the directory or file you are interested in, while the `Header` directive specifies what to do with the headers themselves.

Special note for Apache webmasters

The header functionality only works if you have enabled the header mod. Please refer to your local documentation for more information on how to do this.

For example, the `<Location>` directive below would add the line `Cache-Control: max-age=604800` to every image that comes out of `www.mysite.com/AGChannel/images/`:

```
<Location /AGChannel/images/> Header append
Cache-Control "max-age=604800" </Location>
```

Likewise, the `<Location>` directive below would add `Cache-Control: max-age=43200` to every web page that is located in `www.mysite.com/AGChannel/articles` :

```
<Location /AGChannel/articles/> Header append
Cache-Control "max-age=43200" </Location>
```

What if there is one web page in the `/AGChannel/articles/` directory that should have a shorter cache time? If you follow the previous directive with the `<Location>` directive below, it won't work:

```
<Location /AGChannel/articles/hourlynews.html> Header append
Cache-Control "max-age=3600" </Location>
```

The `<Location>` directive above produces the `Cache-Control` header below:

```
Cache-Control: max-age=43200, max-age=3600
```

This is pretty confusing and error-prone, and it is hard to predict which age the proxy server will actually obey. Instead, try something like this:

```
<Location /AGChannel/articles/> Header append
Cache-Control "max-age=43200" </Location>
```

```
<Location /AGChannel/articles/hourlynews.html>
Header unset Cache-Control
Header append Cache-Control "max-age=3600" </Location>
```

The `Header unset Cache-Control` directive will remove any and all `Cache-Control` headers, so that you have a clean slate on which to add your own.

Non-Apache servers

What if I do not have Apache as my web server using static web pages? Then what?

The answer depends on what you are using. Both Netscape and IIS have sleek configuration controls that allow you to specify header information. For more information on your specific web server, again, we recommend that you check out Mark Nottingham's caching tutorial at:

http://www.mnot.net/cache_docs/#IMP-SERVER

If you are creating a page with a Perl script, you can print out the `Cache-Control` header with a print statement. ASP allows you to do something similar with a `Response` command. Consult your local web guru. If you are the local web guru, consult your local O'Reilly book.

Caching personalized content on user's device

Back to your hypothetical news page — you have a page, *localweather.pl*, that changes from person to person depending on their zip code. What is the best way to cache this?

You really do not want this document cached on the M-Business Sync Server. If you keep a cached version of *localweather.pl* that was created for a reader in Texas, and then repost that same page to somebody in Alaska, you could cause all sorts of weather-related havoc (not to mention fashion faux pas).

But remember, there is a second cache that exists on the user's mobile device. If your user in Texas synchronizes twelve times a day, you do not want to keep fetching new versions of the same weather report (especially if the weather report is only updated once a day.) Somehow, you have to find a way of telling the user's client cache to hold on to the web page, while telling your M-Business Sync Server's shared cache to discard the page.

Cache-Control's private directive

You tell the user's client cache to hold on to the web page, while telling your M-Business Sync Server's shared cache to discard the page, by adding a `private` directive in addition to the `max-age` directive. In other words, you want your HTTP header to look something like

```
Cache-Control: max-age=10800, private
```

If you were creating the daily weather report using a Perl script, your Perl script might contain the lines:

```
print "Content-type: text/html\n";

print "Cache-Control: max-age=10800, private\n";
```

The `max-age` value is still used by the client cache. But your M-Business Sync Server will ignore it entirely.

Other Cache-Control directives

There are a few other `Cache-Control` directives besides `private`. Here are a few of them:

no-store : If you mark a page as no-store, it means it will not be stored anywhere — not on the M-Business Sync Server's shared cache; not on the client cache. To be honest, the only time you would want to use the no-store directive is if you were using the non-caching, 1x1 transparent GIF trick described below in the Frequently Asked Questions.

no-cache : At first glance, this seems like the obvious header to include when you do not want something to cache. But it is not. What no-cache actually does is force the server to check with the origin server that the document is still valid (it is somewhat like giving it a `max-age` of 0), but it might not refresh the content for each different user. So if you have personalized content that you do not want to be cached on our server, do not use `no-cache`. Use `private` instead.

Content that cannot be cached

There are a couple of cases in which documents will not be cached, no matter what your cache-control header says. Specifically:

- Secure documents. While we currently do not support secure, encrypted documents, rest assured that when we do support them in the future, they will not be cached.
- Authenticated documents — pages that use the 401 Authentication Required HTTP code.

Cookies do not block caching

Note, however, that we *do* cache pages with cookies in them. This is because many people like to include cookies on every page, whether they are actually used or not. If we did not cache pages with cookies, we would barely cache anything. So if you have personalized content, do not assume the presence of a cookie will prevent it from being cached. Use a `private` directive.

Configuring caching parameters on M-Business Server

In addition to all the other steps you can take to optimize caching at each stage as your channel pages pass from your web server to M-Business Server to M-Business Client on your users' devices, you can specify the amount of resources that M-Business Server devotes to caching.

The caching parameters that can be controlled through M-Business Server are specified in the *sync.conf* file. For more information, see “Settings in the sync.conf file” [*M-Business Anywhere Administrator Guide*]. Look for the following section headings within the *sync.conf* file:

- “Server-side cache settings” [*M-Business Anywhere Administrator Guide*]
- “Memory- and directory-based cache settings” [*M-Business Anywhere Administrator Guide*]
- “On-device cache settings” [*M-Business Anywhere Administrator Guide*]

More tips for better caching

Here are a few more tips for getting your pages to cache better:

- Include valid `Date` headers. We occasionally encounter servers whose `Date` headers are not valid. This tends to make caching difficult.
- Include `Last-Modified` headers. Not all servers do this by default, although most have the ability to do so. We take advantage of this header, so please include it.
- If the content of your page has not changed, do not touch the file. Occasionally, people have programs that automatically re-write a page that has not changed, only updating the time stamp in a comment somewhere. Unfortunately, this updates the `Last-Modified` date and forces the page to be retrieved again on both your M-Business Sync Server and M-Business Client on the mobile devices. Also, we do a checksum on the HTML before determining if it needs to be uploaded to the client. Changing a timestamp in the comment can affect that check as well.
- Give each unique page a unique URL. If you are generating 20 stories each day, do not give them generic file names like *story01.html* and *story02.html*. Instead, give them unique names like *14mar00story01.html* and *14mar00story02.html* — this will make it easier for you to cache each page, because you do not have to worry about overlapping URLs.
- On a more personal level, for the channels in your M-Business Anywhere account, update the **Refresh** setting so it refreshes, say, every 2 hours, instead of every synchronization. This only affects you personally. All it means is that if you are synchronizing your mobile device twice in a row within a short time period, your M-Business Sync Server will not bother updating your channel. Again, this does not affect the cache or improve performance for other users in any way. But if you happen to be synchronizing very frequently, this will improve synchronization time on your end. Incidentally, do not do this for channels you are currently testing. In those situations, you *do* want to update your channel every synchronization.

Caching FAQ

Q: This seems like an awful lot of work. Is it really worth it?

A: Yes! And not just for your M-Business channel. There are many ISPs out there that take advantage of caching as well. Once you have set up your channel for caching, go and set up the rest of your site for caching. Your web site will suddenly seem to download a whole lot faster. So it is definitely worth the time.

Q: I set up my front page's logo to expire in several months. Now they are telling us we have a new logo. How do I change it?

A: Once a file has been cached, there is not a whole lot you can do to change it. What you can do, however, is create a new file for your logo and have your front page point to version 2 of the logo instead. Once your front page expires (which should be significantly less than 6 months), the new version that points to the new logo will be used.

For this reason, we recommend that if you are just testing your channel, *you should keep it all non-cached before it goes live.*

Q: Hey, I set up my channel for caching, but I am still getting thousands of hits a day! What is wrong?

A: Nothing is wrong. Remember that we have several sync servers, each with its own cache, and we cache different versions for different devices. You might reasonably expect a single page to be hit 50 to 60 times per caching period before it is finally cached in every form on every sync server. This means that if you have a channel with 20 pages, your server could still get hit 1000 times by your M-Business Sync Server for every cache period.

Q: It seems like that would defeat the purpose of caching?

A: No. Remember, that is pretty much an upper limit to how often your page will get hit. That is, if your subscriber numbers doubled, your server would still only get hit about 1000 times by your M-Business Sync Server. If you want to reduce your page hits even more, increase your caching time.

Q: How can I tell I have set caching up properly?

A: The easiest way is check your HTTP headers. Telnet to your web site using port 80 and retrieve the page using a `GET` command. For example, if your channel is `www.mysite.com/AGChannel/index.html`, you would telnet to `www.mysite.com:80`, then type:

`GET /AGChannel/index.html HTTP/1.1` (Enter) `Host: www.mysite.com` (Enter)(Enter)

Look at the headers that get returned. If you see the `Cache-Control` headers, then caching is set up properly.

Those of you who work in Unix-based systems might also have the `GET` command at your disposal. If you type something like the command below, you will receive your channel's index page, along with any HTTP headers at the beginning.

```
GET -e "http://www.mysite.com/AGChannel/index.html"
```

Using a command like the one below will only display your HTTP headers, without any HTML content:

```
GET -ed "http://www.mysite.com/AGChannel/index.html"
```

Another way to test caching is to try synchronizing your channel several times in a row. You should notice your synchronization gets faster after synchronizing a few times. (In order to test this, you should only be synchronizing the channel you are testing. The presence of other channels will hide your results. Also, for testing purposes, you do want the **Refresh** setting for this channel in your M-Business Anywhere account to be set to **Every sync**.)

A third way is to check out the Cacheability Engine at:

http://www.mnot.net/cacheability/

Follow either of the two links in the "As a public web Engine" section. These will bring you to web sites that will report on any URL you would like for basic cacheability. Keep in mind their results are geared for general proxy servers, not the M-Business Sync Server, but they should be a fairly accurate guideline.

Finally, if you load your page in Netscape and choose **View»Page Info** you can sometimes tell if your page is set up for caching because a date will appear in the **Expires** field down below. Note, though, that this is not always accurate. Sometimes you have to unload Netscape's Memory and Disc Cache, and then hit **Reload**. So do not trust this one too much.

Q: Where can I find more information on caching?

A: There is an excellent tutorial at:

http://www.mnot.net/cache_docs/

Also, for the latest news and juicy gossip about the caching community, be sure to check out:

http://www.web-caching.com/

Q: I like to analyze traffic on my web server, because then I can see how often my pages are being accessed. If I start caching documents on the M-Business Sync Server, I will not get the hits on my web server. Is there anything I can do?

A: Well, in all honesty, this is a pretty inaccurate way of figuring out your web traffic. Remember that the M-Business Sync Server updates all pages in all channels every time you synchronize. This means whether a user a) Reads your entire channel, b) Skims the front page, or c) Ignores your channel entirely, every page in your channel is going to get hit once the next time he synchronizes his mobile device.

If you absolutely must keep track of hits on your site, one trick many people do is to include a transparent 1x1 *.gif* file that does not cache (using, say, a `no-store` or `private` directive) on their front page. (You do not want to include a non-caching GIF on *every* page, because every page gets hit whenever a user synchronizes. So it is just redundant information.) Or just keep your very first page non-cacheable.

Q: It seems like this would just move the bottleneck from my web server to my M-Business Sync Server?

A: No. Remember that people have to hit the M-Business Sync Server whether your page is cached or not. The only difference is that if your web page is not cached, our server has to go out and retrieve it, compress it, and then upload it. Whereas if your web page is cached, our server only needs to upload the pre-compressed version. So caching your pages helps us, too.

CHAPTER 5

Deploying channels to users

Contents

Overview

This chapter covers the basics of subscribing users to your channel as quickly and easily as possible.

- There are a variety of ways to set up users on M-Business Server and provide them with M-Business Client software. See "Adding users and providing them with the M-Business Client software" on page 107.
- If your channel is to be delivered to all members of certain groups within your organization, the best way to do this is to manage the whole process through the M-Business Server administrative interface. See "Subscribing users via the M-Business Server Administrator Console" on page 108.
- If your channel is open to anyone in your organization and subscribing to it is optional, you can create a .mal file that users can access from your organization's intranet to subscribe to the channel. See "Configuring users' server connections via a .mal file" on page 109.
- If becoming an M-Business Anywhere user is optional, you can let users do it themselves. See "Letting users do it themselves — self-registration" on page 111.

Adding users and providing them with the M-Business Client software

Although it is not absolutely necessary for a user to have a profile defined on M-Business Server in order to be subscribed to your channel, this is the most common approach. There are several different ways to create user profiles on M-Business Server:

- If users in your organization are already defined in an NT domain, and you have NT domain integration enabled on your M-Business Server, all of the integrated NT domain users are automatically set up as M-Business Anywhere users.
- If users in your organization are already defined in a database other than an NT domain, you can probably import them into M-Business Anywhere by exporting them from the existing database in a CSV file.
- Manually creating a user profiles is always an option, although this is only practical for small numbers of users; this is a good way to do the minimum setup for initial channel testing.
- Allowing users to self-register is also an option, although this is only practical if user interaction with M-Business Anywhere is optional.

Users must have the M-Business Client software installed before they can receive your channel content. There are also a number of different ways to provide users with M-Business Client software:

- Users who self-register from the M-Business Server login page will download the M-Business Client software as part of the process. Note that self-registration is not available if NT domain integration is implemented.
- Any prospective user can download the M-Business Client software from the M-Business Server login page.
- You can email users zipped up executable installer that they can extract locally and run to install the M-Business Client software.

All of the options for adding new users to M-Business Anywhere and providing them with M-Business Client are detailed in “Managing users” [*M-Business Anywhere Administrator Guide*]. NT domain integration is described in “External user authentication” [*M-Business Anywhere Administrator Guide*].

Subscribing users via the M-Business Server Administrator Console

If your channel is to be delivered to all members of one or more groups within your organization, you can manage the whole process through the M-Business Server administrative user interface. This is the most common way of ensuring that the right users are subscribed to the right channels in an enterprise setting.

- First, define a group in the M-Business Server UI for the set of users who will need to be subscribed to your channel. You can probably base the group on a department, workgroup, team, or other organizational unit.
- Next, assign users to the group.
- Finally, set up your channel as a group channel for the group.

Detailed instructions for all of the above tasks are provided "Managing groups" [*M-Business Anywhere Administrator Guide*].

Configuring users' server connections via a .mal file

A *.mal* file configures a server connection for a user who already has the M-Business Client software installed. On Windows workstations where M-Business Client is already installed, the *.mal* file extension is automatically associated with M-Business Connect. Thus you can make the *.mal* file available on an intranet web page, or email it to users directly. When M-Business Connect executes the *.mal* file, the specified server is configured for the user and will synchronize the next time the user synchronizes.

A *.mal* file is a text file with a *.mal* extension. The file begins with an `XML` tag:

```
<?xml version="1.0"?>
```

The rest of the *.mal* file consists of opening and closing `MALServer` tags:

```
<MALServer
...
...>
</MALServer>
```

The opening `MALServer` tag contains the parameters that configure the server. These parameters are detailed in the table below.

Table 1. .mal file parameters

Parameter	Description
`hostname`	Hostname for server.
`port`	Sync port number server is using.
`username`	M-Business Anywhere user name, as defined in the M-Business Server admin database.
`password`	Password for above user, as defined in the M-Business Server admin database.
`passwordIsHashed`	Is the user password hashed? Valid values: `TRUE`, `FALSE`
`passwordIsEncoded`	Is the user password encoded? Valid values: `TRUE`, `FALSE`
`nonce`	Is it a non-CE device? Valid values: `TRUE`, `FALSE`
`disabled`	Is the user account to be disabled? Valid values: `TRUE`, `FALSE`
`friendlyName`	User-friendly server name to display in M-Business Client.

Parameter	Description
userUrl	Server URL.
description	Status to display, such as, "The server is running."
confirmationcaption	Confirmation message to display on success, such as "Connected to <*server_name*>."
serverUri	Server URI.
sendDeviceInfo	Send device info. to server in a synchronization? Valid values: TRUE, FALSE
hashPassword	Hash user password in a synchronization? Valid values: TRUE, FALSE
requestPassword	Require user password to synchronize? Valid values: TRUE, FALSE
ConnectSecurely	Require user to connect securely in a synchronization? Valid values: TRUE, FALSE
AllowSecureConnection	Allow user to connect securely in a synchronization? Valid values: TRUE, FALSE
ConnectSecureOnly	Allow only secure connection? Valid values: TRUE, FALSE
DeviceSecureOverride	Let user override secure connection requirement? Valid values: TRUE - FALSE
confirmation	Confirmation message to display on successful completion.

The full source code for a *.mal* file is listed in "Using a .mal file to subscribe users to a channel" on page 418.

Letting users do it themselves — self-registration

If user interaction with M-Business Anywhere is optional — users can decide if they want to participate at all, and if so, which channels they will subscribe to, you can set it up so that the users register and subscribe themselves. Users can even skip the registration step and just subscribe to your channel, if it is not a group channel.

- First, enable self-registration in the M-Business Server admin UI. See "Managing users" [*M-Business Anywhere Administrator Guide*].
- Next set your channel up as a public channel. See "Managing channels" [*M-Business Anywhere Administrator Guide*].
- Finally, send your potential users an email message telling them the URL of the M-Business Server login page and giving minimal instructions on how to register and subscribe.

Part III. Application Basics

CHAPTER 6

Introduction to application development

Contents

Special design considerations for enterprise applications

The term enterprise application refers to an application that offers all the functionality of a full-featured, dedicated, client-server or distributed processing application.

Before desktop web browsers were adapted for this purpose, such an application would have been presented through a compiled user interface that had to be installed separately on each user's PC. Making these applications Web-based, for a connected desktop environment, required developers to come up with new solutions to:

- Simulate the UI features that can be programmed in compiled code. On M-Business Client, you just have a more limited subset of HTML 4.01 to work with; see “Using DHTML to create a robust user interface” on page 117.
- Generate HTML pages dynamically from data stored in server databases. With M-Business Client, the data has to be stored locally; see “Dynamically generating pages from on-device data” on page 118.
- Maintain the application state between different web pages, forcing the user to complete specific processes in a specific sequence. On M-Business Client, cookies are not available to do this, but several other methods are; see “Maintaining the application state” on page 124.

Designing enterprise applications that can work on disconnected mobile devices requires you to solve these same problems in different ways.

Note
If you are using Java, or contemplating using it, on the server side to produce your M-Business Anywhere application, see Sun's J2EE guidelines, referenced in “Java” on page xiii.

Using DHTML to create a robust user interface

Note
A detailed sample of code illustrating use of DHTML on the M-Business Anywhere platform is available online — see "Dynamic HTML - ExecDash" on page 385.

In both desktop and mobile Web-based enterprise applications, dynamic HTML (DHTML) is used to simulate the user interface features that would otherwise be programmed in compiled code, including:

- Expanding/collapsing menu trees
- Pop-up help text
- Toolbars and navigation bars
- Consistency in text formatting

The big differences

- **No layers** — when a hidden element appears, such as pop-up text or an expanded menu tree, anything downstream on the page moves to make room for it.
- **CSS is in-line only** (inside HTML elements) — internal style sheets (inside the `<head>` tag) and external style sheets (*.css* files) are not currently supported in M-Business Client.
- **No onmouseover/onmouseout** — most mobile devices use a stylus in place of a mouse, so JavaScript events typically must be triggered by `onmousedown` / `onmouseup` or `onclick`, which register either mouse or stylus actions.

A detailed reference on HTML 4 features that are supported by M-Business Anywhere appears in Appendix "HTML 4 support in M-Business Client 5.x and later" on page 295.

Dynamically generating pages from on-device data

Comparables in Model View Controller design vs. M-Business Anywhere application design

- The Model comes from on-device data: either M-Business XML datastore or UltraLite on-device database.
- The Controller is programmed using JavaScript and is primarily engaged using the EventHandlers.
- The View is generated from a combination of pre-generated pages with DOM manipulations to display the appropriate data.
- The Controller and Views are abstractions of the functionality of the web application, but are not directly portable — M-Business Anywhere does not support JavaBeans or Java classes. For this reason, applications for the device often require some modification of the current desktop application in order to perform as desired.

Basic approach in M-Business Anywhere applications

For each generated page to be on the device, write scripts to generate JavaScript and HTML for those pages. Have the same script also copy back-end data that will be accessed into the on-device database. Then make the JavaScript refer to the on-device copy of the data.

For lists of 10 items or less, display as a series of `<a href...>` elements. For lists of more than 10 items, use the List Viewer. See "Using the List Viewer" on page 222.

To do a lot of sorting and searching using the M-Business XML conduit and the M-Business XML POD, you should stay under 1000 records. To do a lot of sorting and searching on more than 1000 records, you should use Adaptive Server Anywhere on the server and UltraLite on the device; have your script generate SQL to run against the UltraLite on-device database.

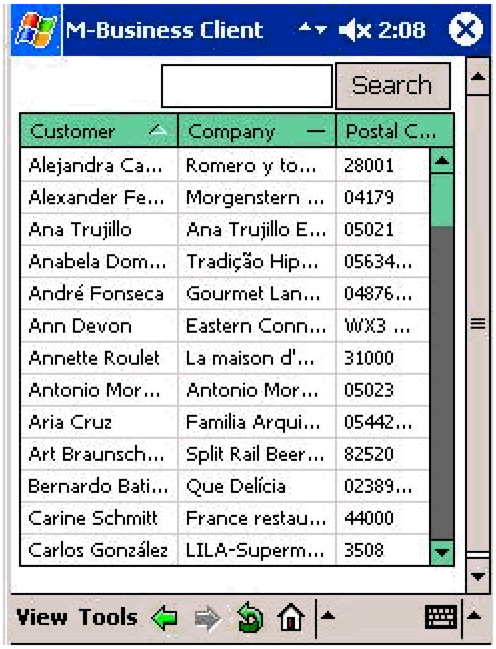
Be sure to make your high-level navigation drop-down lists are dynamic to reflect the dynamic incoming data. You can use the List Viewer if you first populate an array in the M-Business XML datastore using JavaScript. Then you can have the List Viewer display the values from the array. See "Using the List Viewer" on page 222.

Using the List Viewer with M-Business XML datastore

You can use dynamically generated pages in your M-Business channel in much the same way you use them in the desktop Internet world. The big difference is that, in order for any data that needs to be available to a user without synchronizing again, server data must be copied into on-device data at the same time as the generated page is downloaded to the device. The ASP or JSP page on the web server dynamically generates an HTML page with JavaScript. When the page is opened on the device, the JavaScript loads data from the on-device data.

Account summary data example

The figure below shows a page displaying account summary data. This is actually the master page from the Master-Detail Page example. The text that follows focuses on how this page in the Master-Detail Page example uses the List Viewer to dynamically generate and display the data from an on-device M-Business XML datastore. For additional information on this M-Business Anywhere sample code, see "Master-Detail page" on page 386.



Customer	Company	Postal C...
Alejandra Ca...	Romero y to...	28001
Alexander Fe...	Morgenstern ...	04179
Ana Trujillo	Ana Trujillo E...	05021
Anabela Dom...	Tradição Hip...	05634...
André Fonseca	Gourmet Lan...	04876...
Ann Devon	Eastern Conn...	WX3 ...
Annette Roulet	La maison d'...	31000
Antonio Mor...	Antonio Mor...	05023
Aria Cruz	Familia Arqui...	05442...
Art Braunsch...	Split Rail Beer...	82520
Bernardo Bati...	Que Delícia	02389...
Carine Schmitt	France restau...	44000
Carlos González	LILA-Superm...	3508

Getting supporting data onto the device

In the sample, we use a static XML file and schema. In your application, the application server could generate appropriate XML within the application upon request. Static XML can be used for purposes of development, or you could have a separate batch system which asynchronously updates the XML file. This last approach is good for long running queries).

Embedding the List Viewer in the page

To embed an instance of the List Viewer in HTML, you use the `object` tag with the `type` set to `application/avantgo-mime-list`. In the "Master page" on page 419 of the Master-Detail Page example, the following markup embeds the List Viewer and sets values for parameters that control the layout and formatting of the table of data it will display:

```
<object id="accountTable"
    type="application/avantgo-mime-list"
    width="220" height="240"
    style="vertical-align: baseline; padding-left:
    3px; border-color: black; border-width: 1;
```

```
        background-color: white; color: black;
        font-size: 9px; font-family: verdana;">
    <param name="numRows" value="10"></param>
    <param name="rowMargin" value="2"></param>
    <param name="colMargin" value="4"></param>
    <param name="headerBgColor" value="#66cc99"></param>
    <param name="headerBdColor" value="#000000"></param>
    <param name="cellBgColor" value="#f9f9f9"></param>
    <param name="cellBdColor" value="#cecece"></param>
    <param name="cellSelectBgColor" value="#cecece"></param>
    <param name="cellSelectBdColor" value="#f9f9f9"></param>
    <param name="scrollbarBdColor" value="#000000"></param>
    <param name="scrollbarBgColor" value="#666666"></param>
    <param name="scrollbarButtonColor" value="#66cc99"></param>
    <param name="scrollbarArrowColor" value="#000000"></param>
    <param name="scrollbarBubbleColor" value="#66cc99"></param>
    </object>
```

Connecting the List Viewer to the on-device data

Global variables to store datastore information are defined in the master page immediately after the `script` tag:

```
// on-device data object name...
var dbmgr;
// account datastore name...
var accountDb;
// array for account sort order options...
var accountSortOrderArray = new Array(3)
// array for account column internal labels...
var accountCols = new Array(3);
```

Within the `body` tag of the master page, the `initPage()` function is set to execute immediately when the page is loaded:

```
<body onload="initPage()" onUnload="exitPage()">
```

The `initPage()` function stores the "Customers" datastore name in a variable and opens the on-device datastore:

```
var dbname = "Customers";

if (undefined == dbmgr) {
    dbmgr = CreateObject('avantgo.db');
}
if (null == dbmgr) {
    alert('Error: unable to create avantgo.db');
    return false;
}

accountDb = dbmgr.open(dbname, "r");
if (null == accountDb) {
    alert("failed to open db");
    return;
}
```

The `initPage()` function then sets up the List Viewer object:

```
list = document.getElementById("accountTable");
list.setDbSet(accountDb);
```

And lastly the `initPage()` function sets the attributes the List Viewer object will need to display the data:

```
list.setColumnCount(3);

list.setColumnTitle(0, "Customer");
list.setColumnWidth(0, 60);
list.setColumnDSName(0, "ContactName");
list.setColumnHeaderImage(0, 1);

list.setColumnTitle(1, "Company");
list.setColumnWidth(1, 60);
list.setColumnDSName(1, "CompanyName");
list.setColumnHeaderImage(1, 1);

list.setColumnTitle(2, "Postal Code");
list.setColumnWidth(2, 40);
list.setColumnDSName(2, "PostalCode");
```

Note
Labels in `setColumnDSName()` functions must match the data object labels in the XML data file downloaded to the device.

In summary, the List Viewer object is a convenient tool to provide user access to large datasets. Other approaches involving the dynamic generation of a table are more resource-intensive and cause more user perceivable transitions as the page is regenerated for searches and sorting. This functionality is currently supported by the M-Business XML datastore and is not available for UltraLite development.

Filtering the displayed data on Contact Name

Doing searches on large datasets can be time consuming. The recommended amount of data for a given device platform is dependent on the CPU, memory access speed, OS read/write efficiency, and available memory. In general, the newer the device, the more data that can be supported while continuing to supply an acceptable user experience.

If a dataset is really large, it can be a major convenience to the user to allow the display to be filtered to show only those entries that match what the user enters. The List Viewer does not directly support filtering of the dataset it is working with, so that must be handled in the page code.

Immediately after the body tag, the master page defines a form that allows the user to enter text that will be used to filter the displayed data on Contact Name. For example, if the user enters smith, the display will be restricted to Contact Names beginning with smith — Smith, Smithe, Smithson, Smithy, etc.

The filter input form at the top of the page calls a `setFilter()` function when the user taps the Search button:

```
<form>
<table width="220" cellspacing="0" cellpadding="0" border="0">
<tr>
    <td align="right">
        <input style = "background-color: #ffffcc; font-family: verdana;
        font-size: 9px; padding-top: 2px; padding-bottom:1px;"
        type="text" name="filter" id="filterInput" value="" size="8">
    <input type="button" value="Search" onclick="setFilter()">
    </td>
```

```
</tr>
</table>
</form>
```

After defining its variables, the `setFilter()` function stores the List Viewer object (named `accountTable` in this example) in a variable named `list` :

```
list = document.getElementById("accountTable");
```

...then stores the text entered by the user (given the `id` of `filterInput` in the form) in a variable named `filterInput`:

```
filterInput = document.getElementById("filterInput");
```

If the user has not entered any text to filter on, the filter is set to null:

```
if ((null == filterInput.value) || (0 ==filterInput.value.length)) {
    search = accountDb.setFilter(null);
}
```

Otherwise the `setFilter()` function then creates the search string:

```
filter = "ContactName ~ " + filterInput.value;
```

...moves the pointer for the Customers datastore (`accountDb`) to the beginning of the file:

```
accountDb.moveTo(0);
```

...performs a search:

```
search = accountDb.createSearch(filter);
```

...and sets the filter to the search results:

```
accountDb.setFilter(search);
```

Finally, the `setFilter()` function has the List Viewer refresh its dataset to reflect the filtering:

```
list.refreshDbSet();
```

...and has the List Viewer redraw the display:

```
list.refreshScreen();
```

Sorting the display order on different columns

Although a dataset may have a key field which logically orders the data — Customer ID for the Customers dataset in this example — it may at times be convenient for the user to see the data ordered by a different field. The List Viewer supports this directly.

Note
Depending on a range of factors, sorting can easily begin to be unacceptably slow for your users. Be sure to test any sorting feature you implement with the largest datasets users may encounter, on the slowest hardware configuration a user is likely to have.

Maintaining the application state

As a user moves from one web page to another, you need to keep track of what the user has done on the previous page and make data available for what the user is able to do on the next page — maintaining the application state. In desktop web applications, this is usually accomplished by storing values in cookies.

In the M-Business Anywhere environment, cookies are stored on M-Business Server, which uses them in communicating with your channel's web server. These cookies are not available on the device and so cannot be used to maintain the application state. In place of cookies, several other methods are available for this purpose:

- "Using the PODSPrefs object" on page 124 to store values in the `PODSPrefs` object in device memory, where it is automatically cleared when the session terminates.
- "Using on-device data to maintain the application state" on page 127 to store values in on-device data, which your code must initialize at the start of each session.

Using the PODSPrefs object

To store a value in the `PODSPrefs` object, use this JavaScript command in the page from which your application is passing control to another page:

```
avantgo.preferences.setStringValueForKey(key, valueStr)
```

Where:

- *preferences* is the `PODSPrefs` object
- *key* is the preference key value
- *valueStr* is the value to which the preference is to be set

To retrieve a value from the `PODSPrefs` object, use this JavaScript command in the page to which your application is passing control from another page:

```
avantgo.preferences.getStringValueForKey(key)
```

Where:

- *preferences* is the `PODSPrefs` object
- *key* is the preference key value

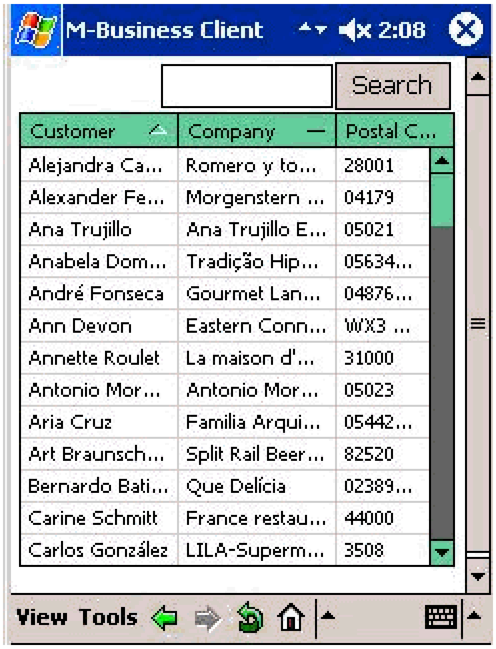
For more information, see "getStringValueForKey( )" [*M-Business Anywhere API Reference*] and "setStringValueForKey( )" [*M-Business Anywhere API Reference*]. Accessing the `PODSPrefs` object is documented in "preferences" [*M-Business Anywhere API Reference*].

Master-detail page example

An example that has widespread application in business data is the master-detail page relationship. A master page provides summary level information on the customer, account, product, or whatever category of information is represented. From this master page, the user can drill down and see the detail level of information for a selected entry on the master page. In desktop applications, the detail page may be displayed in a special area at the bottom of the master page; in mobile applications, the detail page typically has to be a completely separate page, and therefore you have to maintain the application state as the user switches between the master and detail pages.

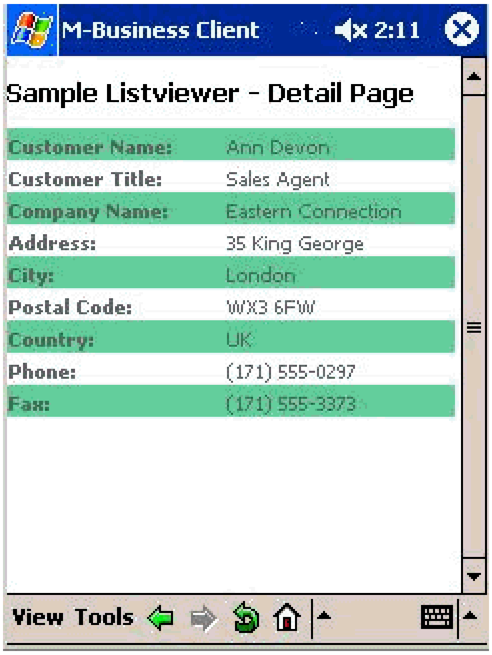
The text that follows focuses on how the Master-Detail Page example uses the `PODSPrefs` object to maintain the application state. For additional information on this M-Business Anywhere sample code, see "Master-Detail page" on page 386.

An accounts master page, also commonly referred to as the list view, displays just enough information in each account's row to allow the user to identify the account. The master page would look something like the figure below.



Customer	Company	Postal C...
Alejandra Ca...	Romero y to...	28001
Alexander Fe...	Morgenstern ...	04179
Ana Trujillo	Ana Trujillo E...	05021
Anabela Dom...	Tradição Hip...	05634...
André Fonseca	Gourmet Lan...	04876...
Ann Devon	Eastern Conn...	WX3 ...
Annette Roulet	La maison d'...	31000
Antonio Mor...	Antonio Mor...	05023
Aria Cruz	Familia Arqui...	05442...
Art Braunsch...	Split Rail Beer...	82520
Bernardo Bati...	Que Delícia	02389...
Carine Schmitt	France restau...	44000
Carlos González	LILA-Superm...	3508

When the user taps a name in the **Customer** column, a detail page, also commonly referred to as the detail view, appears for that account. The detail page might look something like the figure below.



Sample Listviewer - Detail Page

Customer Name:	Ann Devon
Customer Title:	Sales Agent
Company Name:	Eastern Connection
Address:	35 King George
City:	London
Postal Code:	WX3 6FW
Country:	UK
Phone:	(171) 555-0297
Fax:	(171) 555-3373

Storing application state information in the PODSPrefs object

Within the `body` tag of the master page, the `initPage()` function is set to execute immediately when the page is loaded:

```
<body onload="initPage()"...>
```

The `initPage()` function stores the Customers datastore name in a variable:

```
var dbname = "Customers";
```

An `onclick` event handler calls a `gotoDetailPage()` function when it determines the user has tapped a customer row for which the detail page will display more detailed information:

```
gotoDetailPage("Customers", accountDb.CustomerID, "CustomerDetail.html");
```

The `gotoDetailPage()` function takes the parameters that were passed and stores the datastore name and the ID of the customer row that was tapped, then opens the detail page:

```
function gotoDetailPage(dbname, id, pageurl)
{
    avantgo.preferences.setStringValueForKey("detailPage_"+dbname+"Id", id);
    window.location = pageurl;
}
```

Retrieving application state information in the detail page

The detail page defines an `onload` function that executes as soon as the page is loaded:

```
<body onload="onloadHandler()"...>
```

The `onloadHandler()` function accesses the specific data for the customer row that the user tapped and binds it to the detail page template:

```
function onloadHandler() {
    // Retrieve dbname and CustomerID.
    var dbname = avantgo.preferences.getStringValueForKey("ListPage_dbname");
    var id = avantgo.preferences.
        getStringValueForKey("detailPage_"+dbname+"Id");

    dbset = dbMgr.open(dbname, r");
    if(dbset == null) {
        alert("Could not open the '" + dbname + "' database.");
        return;
    }


    // Locate the customer record that matches CustomerID.
    var search = dbset.createSearch("CustomerID ==" + id);
    dbset.setFilter(search);
    dbset.moveFirst();


    // Bind customer data to detail page template.
    document.getElementById("customer_name").firstChild.data =
         dbset.ContactName;
    document.getElementById("customer_title").firstChild.data =
         dbset.ContactTitle;
    document.getElementById("customer_company").firstChild.data =
         dbset.CompanyName;
    document.getElementById("customer_address").firstChild.data =
         dbset.AddressLine1;
    document.getElementById("customer_city").firstChild.data =
         dbset.City;
    document.getElementById("customer_zip").firstChild.data =
         dbset.PostalCode;
    document.getElementById("customer_country").firstChild.data =
         dbset.Country;
    document.getElementById("customer_phone").firstChild.data =
         dbset.Phone;
    document.getElementById("customer_fax").firstChild.data =
         dbset.Fax;
    }
```

Note

The labels above following `.firstChild.data = dbset.` match the data object labels in the XML file downloaded to the mobile device.

Using on-device data to maintain the application state

You can use either the M-Business XML datastore or the UltraLite on-device database to store values that your application pages can use to maintain the application state.

You may want to use the M-Business XML datastore for this, even if you are using UltraLite for application's primary data needs, because the setup is simpler — the M-Business XML POD can create a schema on the fly.

Note
When you use on-device data to maintain the application state, be sure your application code initializes the data at the start of every session.

M-Business XML datastore is more suitable if the data is not synchronized to a back-end datastore, since there is an additional step of deploying a schema. In most cases, the M-Business XML datastore is a simpler method to implement session state maintenance.

For more information on using on-device data, refer to the chapters for the on-device data solution you wish to use:

- For **UltraLite**, see "Using UltraLite for M-Business Anywhere for on-device data" on page 247.
- For **M-Business XML POD**, see "Using M-Business XML conduit and API" on page 261, and "Building applications with M-Business XML conduit" on page 273.

Mobilizing existing desktop web content

To preview the major design feature of existing desktop Web-based applications that you will have to change: everything that works on a request/response model will have to be reworked, in order for your mobile users to be able to continue working while they are offline.

Checklist of things to look for

If you have an existing Web-based application that has been developed for a typical desktop PC that is always connected to the Internet, you can be fairly certain that it will not work nicely as an M-Business channel without some major changes.

One approach to finding out what needs to be reworked is to create a channel for the application as it exists, then synchronize and see what does not work. You can use the checklist below as a guide to specific things to look for.

- Login is problematic.

  You always have to synchronize twice — once to log in, and then once more to get updated pages. You can solve this by increasing the timeout value for cookies coming from mobile users. On a connected desktop, 20 minutes without user activity might be a reasonable time to expire a cookie. On a mobile device that synchronizes periodically, the expiration time for cookies should be set to just a little more than the normal time between synchronizations. If the same web server will be receiving requests from both desktop and mobile users, use the HTTP headers to distinguish between these groups and set cookies accordingly.

- Logout links will exit.

  If you have a logout link within the application, your users will simply never get the application. As M-Business Server spiders through the links within your channel's Link Depth, encountering a logout link causes a logout. So you must remove all logout links.

- Pages linked to via JavaScript will not be downloaded.

  Locate any links implemented via JavaScript and replace them with regular HTML `<a href=...>` links.

- Pop-up windows implemented via JavaScript will not be available.

  You will have to re-do these as `<div>` units that can be hidden and displayed by JavaScript.

- Links that should open in a new window open in same window.

  If this is a problem, add a prominent link to return to the previous page.

- Synchronizing is painfully slow, some pages are not downloaded, or size of downloaded content is much larger than expected.

Take a close look at your channel web site design, in terms of your Link Depth and Follow Offsite Links settings. You may need to redesign your site so that you can set a Link Depth value that does get all the important pages but does not bring in unnecessary pages. If you have Follow Offsite Links set to Yes, you can design your site so that your links offsite pages return only the specific pages that you link. See "Preventing download of unwanted pages with Follow Offsite Links" on page 57.

- Some channel pages are not downloaded.

  There are a variety of possible causes for individual pages not being available. See "Channel troubleshooting and debugging tips" on page 42.

- Application state is indeterminate — application logic locks up.

  If your application depends on values stored in cookies to determine what the user can do next, these values are not available on device. To maintain an application state, you will have to store values locally. See "Maintaining the application state" on page 124.

> **Note**
> Record updates must be submitted as forms with session state information. Forms by nature are contextless. We have found that users will typically submit the prior values as well as the current values in order to understand the state of the device at the time of last synchronization as well as the proposed values. An application will then compare the prior values with the current values from the back-end and, if they are the same, the conclusion is that the back-end has not been modified and the application can safely apply the modifications from the device. Otherwise, there is the potential for a conflict.

- Data that should be available is not, unless user synchronizes again.

  Any data that needs to be available without requiring the user to synchronize again must be stored on the device. See the chapter for the on-device data solution you are using:

  - "Using UltraLite for M-Business Anywhere for on-device data" on page 247
  - "Using M-Business XML conduit and API" on page 261

- External and internal style sheets do not work.

  External and internal style sheets are not supported. In most cases, M-Business Client will do an acceptable job of page rendering on the mobile device without style sheets. Where CSS is critical, as in use with JavaScript to hide and display elements, you will have to convert any CSS external or internal style sheets into in-line markup.

- Users must scroll horizontally to see full content.

  Content designed for desktop browsers frequently fixes page width at somewhere around 500 pixels, in order to control some aspect of the layout. On a mobile device. you have to let M-Business Client render the page to fit the device screen and find other solutions to layout issues that the fixed width solved on the desktop.

- Some pages look too cluttered on the mobile device.

With a much smaller screen, come of the content that seemed important enough to include in the desktop version simply isn't worth including in the mobile version. Look for things you can leave out because they aren't really that critical.

- Some pages are just too big on the mobile device.

  With a much smaller screen, vertical scrolling can become a problem on a mobile device when it was not a problem on the desktop. You can either add frequent links to the top of the page, or split the page into multiple, shorter pages.

- Some pages look OK on one mobile platform, but not on another.

  If allowing M-Business Client to render pages still leaves some pages looking bad on some platforms, consider optimizing the content of those pages for different platforms.

- Selecting from long list or setting date/time is awkward when form is on small screen.

  If the smaller screen size makes selecting an item from a long list awkward, implement the List Viewer widget — see “Using the List Viewer” on page 222. If your calendar pop-up to help specify a date doesn't work well in the small screen, implement the Date/Time Picker widget — see “Using the Date/Time Picker” on page 214.

Using on-device data in place of cookies

In interactions with your channel's web server, cookies work just like cookies in the desktop Internet world. Users' cookies are passed back and forth between M-Business Server and your channel's web server, and those cookies are used by the web server the same way they would be in the desktop Internet world.

With M-Business Anywhere, however, the cookies are not stored locally where the browser is running (on the device), but in M-Business Server's database. This means that your application cannot use values stored in cookies to determine what to do next.

M-Business Anywhere provides two on-device database options that you can use in your applications:

- **UltraLite for M-Business Anywhere**, which allows your application to update the back-end database from on-device user changes. For more information, see “Using UltraLite for M-Business Anywhere for on-device data” on page 247.

- **M-Business XML POD**, which uses less on-device resources than UltraLite, but does not allow your application to update the back-end database from on-device user changes. For more information, see “Using M-Business XML conduit and API” on page 261and “Building applications with M-Business XML conduit” on page 273.

Using the List Viewer to dynamically generate pages from on-device data

If you are using M-Business XML POD for your on-device data, you can use the List Viewer to automate large portions of several dynamic features that you would otherwise have to manually code, such as displaying a formatted table of data and sorting the display on a particular column. For more information, see “Using the List Viewer with M-Business XML datastore” on page 118.

Using the M-Business SOAP API

If you want to customize the M-Business Administrator console, or automate any of its functions, the M-Business SOAP API supports web services that allow developers to programmatically perform any task that can be performed through the Administrator Console UI.

The SOAP API also allows customers or independent software vendors to re-brand or customize the Administrator Console, or to completely replace it, making it possible to OEM or embed M-Business Anywhere with a completely custom interface.

Note
You cannot use the M-Business SOAP API within M-Business Client. The SOAP functionality can only be used in desktop applications that interact directly with M-Business Server.

For more information on the M-Business SOAP API, see “M-Business SOAP API reference” [*M-Business Anywhere API Reference*].

CHAPTER 7

Basic application creation

Contents

Introduction to M-Business JavaScript engine

M-Business JavaScript engine supports most of the standard capabilities of JavaScript. JavaScript engine also provides direct access to most of the features of M-Business client extension API (PODS). For details on the JavaScript engine feature set, see Appendix "M-Business JavaScript engine reference" on page 331.

Many good references on standard JavaScript are available; see "Related publications" on page xi for specifics.

This chapter explains how to access the features of JavaScript engine that are unique to the M-Business Client environment. These include use of extensions to the JavaScript `Form` object when submitting forms, accessing standard PODS functions, and extending JavaScript's capabilities by accessing custom PODS functions.

Submitting forms

M-Business Client supports form submissions on HTML pages. In desktop browsers, a submitted form is sent immediately to an HTTP server. M-Business Client also sends a submitted form immediately to an HTTP server, if the mobile device is online. If the device is offline, M-Business Client queues the form for submission the next time the device is synchronized to the M-Business Sync Server.

In M-Business Client (and desktop browsers) the `submit()` method of the `Form` object can be used to submit a form programmatically. If the device is online, this method submits the form and displays a new page. If the device is offline, this method queues up the form submission for submission the next time the device is synchronized.

M-Business JavaScript engine also supports the following extensions to the JavaScript methods of the `Form` object for form submission: `submitForm( )`, `submitNoResponse( )`, and `submitWithResponse( )`. Use these methods if the form attributes are decided at run time. For example, you might want to set the `agsumbitSize` attribute to a smaller value when online than when offline. Otherwise, you should embed the properties in the `Form` tag in your HTML.

For more information:

- For a list of the special M-Business Anywhere properties, see “Form extensions” on page 340.
- For information on the M-Business Anywhere-specific submit methods for the `Form` object, see “M-Business JavaScript engine methods” on page 343.

Using DHTML

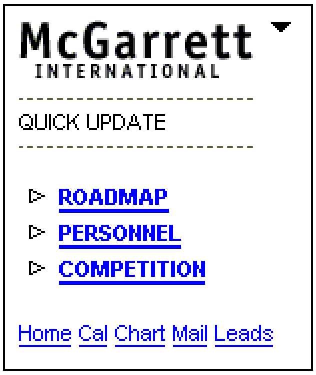
The dynamic HTML (DHTML) that M-Business JavaScript engine supports makes it possible to create robust, desktop-style applications entirely in JavaScript. DHTML makes it possible for a web page, in response to user interaction, to dynamically change the information displayed without any support from a server.

This section describes a sample application that illustrates some of the features that you can implement with DHTML. The source code for all the examples in this section is available online. See "Dynamic HTML - ExecDash" on page 385.

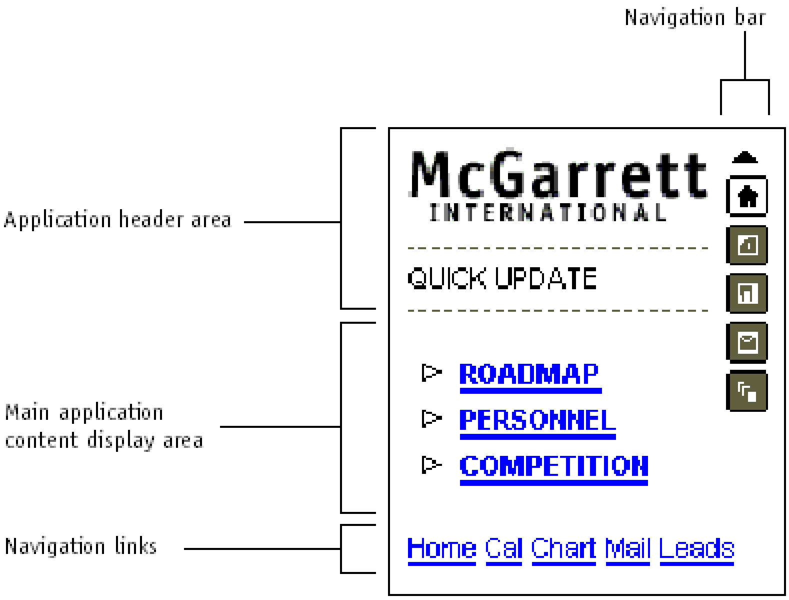
Home page: expanding and collapsing hierarchical text

A home page is the first page users see when they run your application. The home page in this example, *index.html*, presents some current information and provides access to the other pages in the application. For a full listing of the DHTML source code for this page, see "Home page sample: expanding and collapsing hierarchical text" on page 351.

The home page, as it displays initially, is shown in the following figure.



When the user taps the down arrow icon in the upper right corner, a navigation bar appears.



The figure above shows the different parts of the home page.

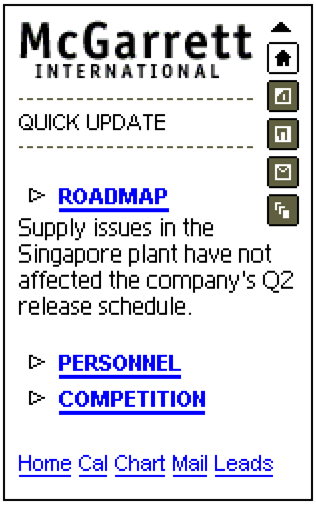
- **Application header area** displays a consistent header for all pages of the application. McGarrett International is the hypothetical company's name. The lower portion displays a brief descriptive title for the current page, QUICK UPDATE for the home page.
- **Main application content display area** is where the content for each page is displayed. This area expands vertically as needed.
- **Navigation links** at the bottom of the page are always available, and serves as an alternative to the icons in the drop-down navigation bar.
- **Navigation bar** is a graphical navigation aid, displayed only when the user taps the down arrow in the upper right corner of the page.

The page is laid out in a one-row, two-column table. The navigation bar occupies the right-hand cell in the row. Everything else is in the left-hand cell.

Note
This page layout is used for all pages in this sample application. Only the descriptive title and the content display area change on different pages. The parts that do not change are described only in this section for the home page.

The main content area of the home page lists some headings for which additional text is available. When the user taps a heading, the additional text is displayed below it. Tapping the same heading again causes the additional text to disappear. This allows the mobile user to display only the desired details.

The following figure shows the home page with the additional text displayed for the ROADMAP heading.



Supply issues in the Singapore plant have not affected the company's Q2 release schedule.

Here is the DHTML for the first heading, ROADMAP, with key portions that implement the hiding and displaying of the additional text shown in bold:

```
<!-- Quick Update 1 -->
<IMG height=11 src="pointer1.gif" width=16 border=0>
<b>
<A onclick="return ExpandDisp('item1Text');" href ="#">
ROADMAP
</A>
</b>

<div ID="item1Text" style="display:none;">
Supply issues in the Singapore plant have not
affected the company's Q2 release schedule.
</div>
```

The additional text bracketed by `<div>` tags is initially hidden by the `style="display:none;"` setting. The name given to the additional text by the `ID="item1Text"` attribute setting allows the `onclick` event handler for the main ROADMAP heading to target the additional text when it calls the `ExpandDisp()` function: `onclick="return ExpandDisp('item1Text');"`.

The `ExpandDisp()` function is located at the top of the page code, in the `<script>` section:

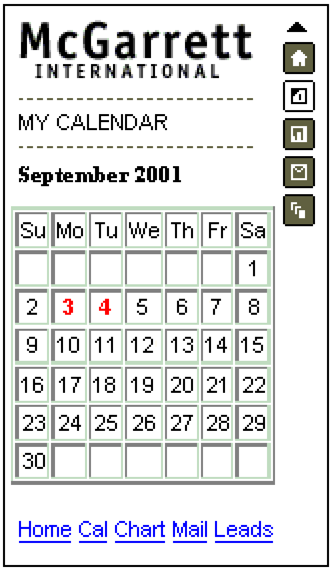
```
function ExpandDisp(id)
{
    var element = document.getElementById(id);
    if (element.style.display == "none")
        element.style.display = "block";
    else if (element.style.display == "block")
        element.style.display = "none";
    }
```

The `item1Text` identifier is passed to the `ExpandDisp()` method as the `id` parameter. The method uses the DOM `document.getElementById()` method to get a reference to the `DIV` containing the text based on this ID. Then the state of the `DIV` element's display property is toggled.

Calendar page: hiding and displaying page elements

Many applications involve some sort of calendar information. The calendar page in this sample application receives event information from a server application and allows the user to view the schedule details for any particular day by tapping that day in the calendar. For a full listing of the DHTML source code for this page, see "Calendar page sample: hiding and displaying page elements" on page 355.

The calendar page, as it displays initially, is shown in the figure below.



McGarrett
INTERNATIONAL

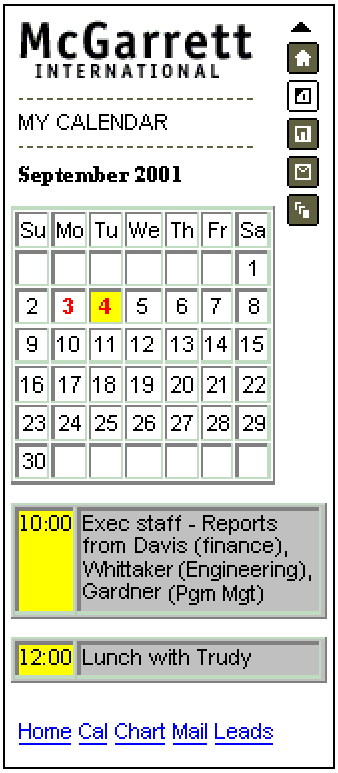
MY CALENDAR

September 2001

Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

Days displayed in red have schedule information available. When the user taps a red-number day, any schedule information for that day is displayed. When the user taps a different day, the schedule information for the previous day disappears at the same time any schedule information for the new day is displayed.

The following figure shows the calendar page with schedule information displayed for the fourth.



McGarrett
INTERNATIONAL

MY CALENDAR

September 2001

Su	Mo	Tu	We	Th	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

10:00	Exec staff - Reports from Davis (finance), Whittaker (Engineering), Gardner (Pgm Mgt)

12:00	Lunch with Trudy

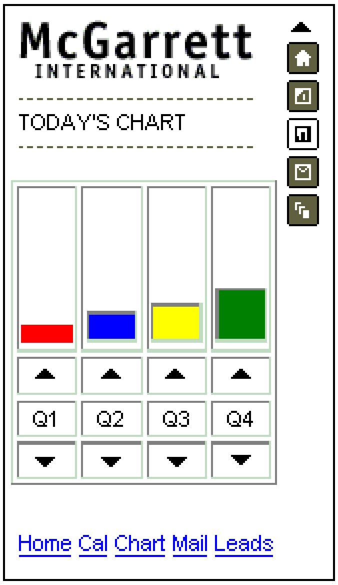
Home Cal Chart Mail Leads

The display of schedule information that is available for a day that the user taps in this sample calendar page is handled in the same way as the expanding and collapsing of the additional text for a header in the home page example, in "Home page: expanding and collapsing hierarchical text" on page 138. Any page element can be hidden and displayed this way, under the control of your DHTML, in response to user actions.

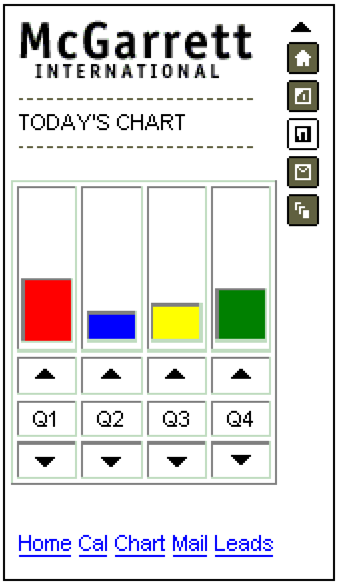
Chart page: dynamically changing graphic information

Many applications involve some sort of graphic display of information. The chart page in this sample application presents a bar chart with data for four quarters and allows the user to adjust any quarter's value up or down by tapping an up or down arrow for that quarter. For a full listing of the DHTML source code for this page, see "Chart page sample: dynamically changing graphic information" on page 361.

The chart page, as it displays initially, is shown in the following figure.



Each quarter has an up arrow and a down arrow which the user can tap to increase or decrease, respectively, the value displayed for that quarter. The chart page, as it might look after some user interaction, is shown in the figure below.



The DHTML code first assigns a variable for each quarter an index value, then initializes the color and revenue values for the quarters in a JavaScript array — this is like a mini-database inside the web page:

```
//
// Globals
//
var intQ1 = 0;
```

```
    var intQ2 = 1;
    var intQ3 = 2;
    var intQ4 = 3;

    //
    // [bar chart color, initial revenue in millions]
    //
    var arrQ = [["red", 7], ["blue", 12], ["yellow", 15], ["green", 21]];
```

Each quarter's up arrow icon is set up with an onclick event handler to call the `BumpUp()` function, passing the index for that quarter. The code for quarter 1 is shown below:

```
<td align="center" onclick="BumpUp(intQ1)">
<img src="open.gif" border="0" width="16" height="11">
</td>
```

Similarly, each quarter's down arrow icon is set up with an `onclick` event handler to call the `BumpDown()` function, passing the index for that quarter.

The `BumpUp()` function accepts the quarter index passed as the local variable `intQ_IN`.

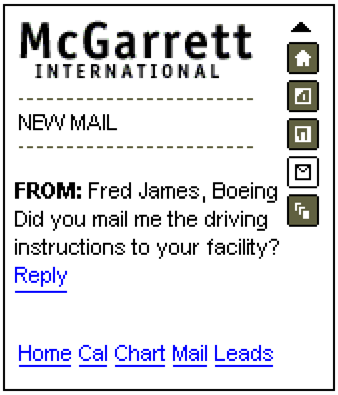
```
function BumpUp(intQ_IN)
{

    if (arguments.length != 1)
        throw new Error("function BumpUp() incorrectly called");

    var obj = document.getElementById("Q" + (intQ_IN + 1) +  "_Bar");

    if (arrQ[intQ_IN][intQ_Rev] < 53)
    {
        arrQ[intQ_IN][intQ_Rev] += 6;
        obj.style.height = arrQ[intQ_IN][intQ_Rev] + "px";
        obj.style.width = "20px";
        obj.style.backgroundColor = arrQ[intQ_IN][intQ_Color];
    }

    return;

} // BumpUp
```

The `BumpDown()` function works much the same way, but it decrements the height of the bar for the quarter.

Mail page: a form submission

Many applications involve some sort of form submission. The mail page in this sample application submits a form. The page displays messages and allows the user to reply by tapping a Reply link. The Reply link displays the form. For a full listing of the DHTML source code for this page, see "Mail page sample: a form submission" on page 367.

The mail page, as it displays initially, is shown in the following figure.

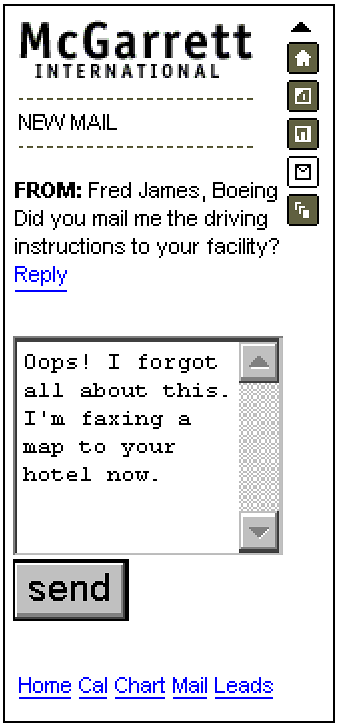


NEW MAIL

FROM: Fred James, Boeing
Did you mail me the driving instructions to your facility?
Reply

This sample page displays short messages with a Reply link for the user to use to send a reply. When the user taps the Reply link, a scrolling text box appears for the user to enter the text of a reply. The figure below shows the mail page after the user has tapped the Reply link and entered text for a reply message.

The user taps the **Send** button when the reply text is complete, and the page submits the form to the server, which handles forwarding the message to the recipient.



NEW MAIL

FROM: Fred James, Boeing
Did you mail me the driving instructions to your facility?
Reply

Oops! I forgot all about this. I'm faxing a map to your hotel now.

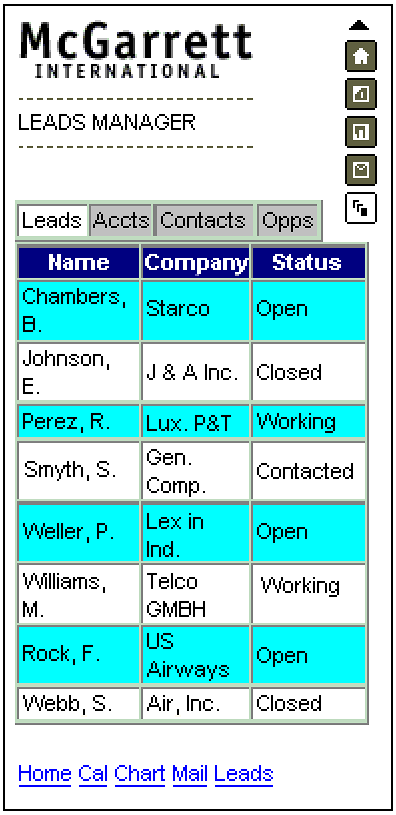
The DHTML code for the mail page uses the same approach as the "Home page: expanding and collapsing hierarchical text" on page 138 and the "Calendar page: hiding and displaying page elements" on page 141 to hide and display the form containing the scrolling text box.

The **Send** button submits the form to the server, which would handle the message from there.

Leads page: dynamically creating and sorting a table

If your application is data intensive, and you want the user to be able to sort the data after it is received, then using the full power of DHTML to create, update, and rearrange HTML elements on a page can be superior to dynamically generating a page on the server that becomes a static page on the user's mobile device. For a full listing of the DHTML source code for this page, see "Leads page sample: dynamically creating and sorting a table" on page 371.

The leads page displays four tabs: Leads, Accts, Contacts, and Opps. Initially, the page displays the Leads tab, as shown in the figure below.

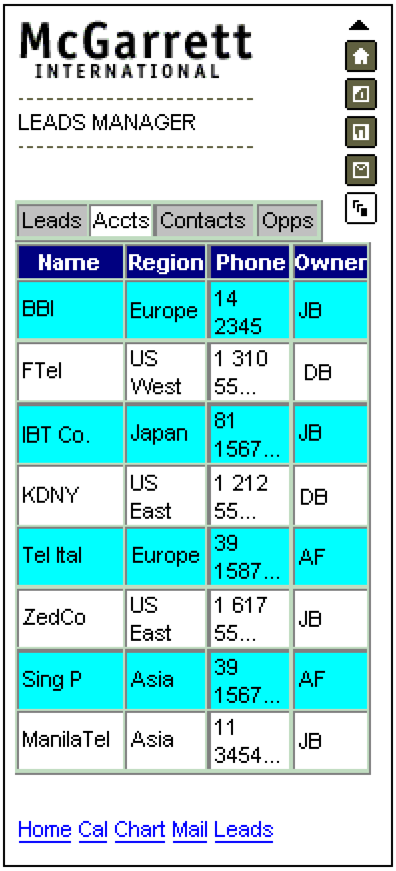


McGarrett INTERNATIONAL

LEADS MANAGER

Name	Company	Status
Chambers, B.	Starco	Open
Johnson, E.	J & A Inc.	Closed
Perez, R.	Lux. P&T	Working
Smyth, S.	Gen. Comp.	Contacted
Weller, P.	Lex in Ind.	Open
Williams, M.	Telco GMBH	Working
Rock, F.	US Airways	Open
Webb, S.	Air, Inc.	Closed

Tapping on a tab displays the table for the type of information identified by the tab label. Tapping the Accts tab displays the table of accounts information, as shown in the following figure.

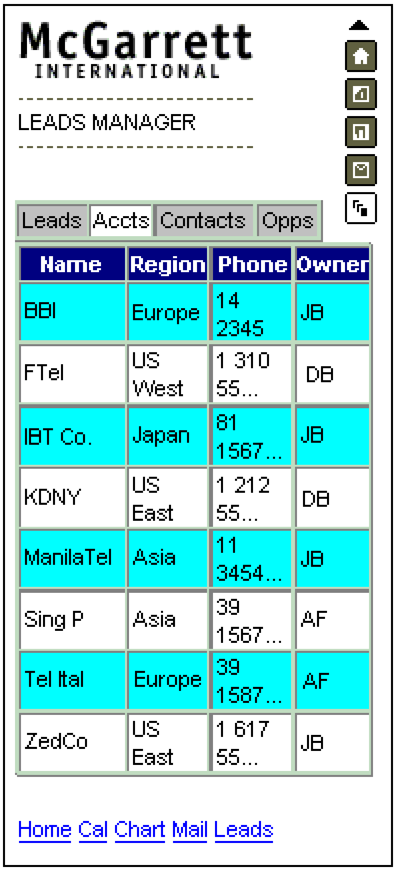


McGarrett INTERNATIONAL

LEADS MANAGER

Name	Region	Phone	Owner
BBI	Europe	14 2345	JB
FTel	US West	1 310 55...	DB
IBT Co.	Japan	81 1567...	JB
KDNY	US East	1 212 55...	DB
Tel Ital	Europe	39 1587...	AF
ZedCo	US East	1 617 55...	JB
Sing P	Asia	39 1567...	AF
ManilaTel	Asia	11 3454...	JB

Tapping on a column heading in a table sorts the table by that column's data. For example, tapping the Name column heading in the Accts table sorts that table on the Name column, as shown in the following figure.



McGarrett INTERNATIONAL

LEADS MANAGER

Name	Region	Phone	Owner
BBI	Europe	14 2345	JB
FTel	US West	1 310 55...	DB
IBT Co.	Japan	81 1567...	JB
KDNY	US East	1 212 55...	DB
ManilaTel	Asia	11 3454...	JB
Sing P	Asia	39 1567...	AF
Tel Ital	Europe	39 1587...	AF
ZedCo	US East	1 617 55...	JB

All of the tables are created dynamically. The HTML for the `<div>` where the tables will be displayed is initially just a placeholder. The `id` tells the DHTML code where to do its work:

```
<div id="TabTablesDiv">
<!---------- START TABS CONTENT ---------->
</div>
```

All of the data arrives in JavaScript arrays within the page. The code below assigns an index value to the Leads tab and initializes an array to hold the Leads tab data:

```
var intLM_LeadsTab = 1;
var arrLM_Leads =[
    ["Name", "Company", "Status"],
    ["Chambers, B.", "Starco", "Open"],
    ["Johnson, E.", "J & A Inc.", "Closed"],
    ["Perez, R.", "Lux. P&T", "Working"],
    ["Smyth, S.", "Gen. Comp.", "Contacted"],
    ["Weller, P", "Lexin Ind.", "Open"],
    ["Williams, M.", "Telco GBMH", "Working"],
    ["Rock, F.", "US Airways", "Open"],
    ["Webb, S.", "Air, Inc.", "Closed"]
    ];
```

All of the tables are dynamically generated from the array data. The code below creates the basic table structure and calls a function to create each table's rows and cells and then load the table's data:

```
objMasterTabsDIV = document.getElementById("TabTablesDiv");

for (i = 1; i <= 4; i++)
{
    //
    // DIV
    //
    objDIV = document.createElement("DIV");
    objMasterTabsDIV.appendChild(objDIV)
    objDIV.id="MainT" + i + "Div";
    objDIV.style.display = (i == 1) ? "block" : "none";


    //
    // DIV => TABLE
    //
    objTABLE = document.createElement("TABLE");
    objDIV.appendChild(objTABLE);
    objTABLE.id = "Table" + i;
    objTABLE.border = "1";
    objTABLE.cellSpacing = "0";
    objTABLE.width = "134px";
    objTABLE.style.fontSize = "8px";


    //
    // DIV => TABLE => TBODY
    //
    objTBODY = document.createElement("TBODY");
    objTABLE.appendChild(objTBODY);
    //
    // DIV => TABLE => TBODY
    //     => BR
    //
    objBR = document.createElement("BR");
    objDIV.appendChild(objBR);
} // for


//
// Dynamically load the data into the Lead
// Manager tables. (Adds the TR and TD
// elements to the above tables.)
//
LoadLeadManagerTable("Table1", arrLM_Leads);
LoadLeadManagerTable("Table2", arrLM_Accounts);
LoadLeadManagerTable("Table3", arrLM_Contacts);
LoadLeadManagerTable("Table4", arrLM_Opps);

return;
```

The code below is the first part of the `LoadLeadManagerTable()` function. This part deletes any existing table rows:

```
function LoadLeadManagerTable(strTableName, arrTableData)
{
    var objTable;
    var eleTR;
    var intTableRows;
    var intTableCols;
    var i;

    if (arguments.length != 2)
        throw new Error("function LoadLeadManagerTable()
        incorrectly called");

    fChangeColor = 0;

    objTable = document.getElementById(strTableName);

    //
    // Delete the existing rows in the table.
    //
    intTableRows = objTable.rows.length;

    for (i = 0; i < intTableRows; i++)
    objTable.deleteRow(0);


    //
    // Add the records to the table.
    //
    fChangeColor = 0;

    //
    // How many columns does this table have?
    //
    if (arrTableData.length > 0)
    intTableCols = arrTableData[0].length
```

The code below is the second part of the `LoadLeadManagerTable()` function. This part creates a table's rows and cells and loads a table's data from the associated array:

```
//
// Load the data into the table.
//
for (i = 0; i < arrTableData.length; i++)
{
    //
    // Insert a new row for each record: adds a TR element.
    //
    eleTR = objTable.insertRow(i);

    if (i == 0) {
        eleTR.style.color = "white";
        eleTR.align = "center";
        eleTR.valign = "top";
        }
    else {
        eleTR.style.color = "black";
        }

    //
    // Add cells to the row: add TD elements.
    //
    for (j = 0; j < intTableCols; j++)
    {

        cell = eleTR.insertCell(j);
        cell.border = 1;

        if (i == 0) {
```

```
                  cell.onclick = SortLeadManagerTable;
                  cell.style.backgroundColor = "navy";
                  }
              else {
                  cell.style.backgroundColor = (fChangeColor == 1)
              ? "aqua" : "white";
                  }
              cell.appendChild(document.createTextNode(arrTableData[i][j]));
          } // for

          //
          // Create the alternating row colors.
          //
          if (fChangeColor == 1)
              fChangeColor = 0;
          else
              fChangeColor = 1;
      } // for add rows

      return;

  } // LoadLeadManagerTable
```

When the user taps a column heading, a sort routine is called to sort the array data for the table in order by that column. DHTML code then reloads the existing table with the sorted array values, again using the `LoadLeadManagerTable()` function, without having to regenerate the table. The only thing that changes is the values in the table.

Creating simulated multi-page forms

Sometimes you may want to break a single form into several logical pages, which compose a single form submission. There are two basic approaches to doing this in M-Business Anywhere:

- Using DHTML to selectively hide and display information for the different logical pages.
- Use DOM operations on the Submission Manager to build multiple pages, and a single form, in which the application operates on a single form submission.

These approaches are described in more detail in "Serializing forms: breaking one form into several pages" on page 195. A pair of code samples, each illustrating one of the above approaches, is available online. See "Serializing form submissions with M-Business Client" on page 386.

Using PODS functions from JavaScript engine

PODS functions can greatly extend the capabilities available in JavaScript engine alone. For example, you cannot use JavaScript engine to dynamically generate an HTML page in M-Business Client; the `document.write()` method, which performs this function in standard JavaScript, is not implemented. However, you can use the PODS `ADOMObject` from JavaScript engine to achieve the same result. See "ADOMDOMImplementation object" [*M-Business Anywhere API Reference*].

PODS functions are accessed from JavaScript engine in two different ways, depending on the type of PODS function being accessed:

- **Standard PODS objects** that PODS automatically exposes to JavaScript are accessed by simply prefixing the PODS attribute or method name with the corresponding JavaScript object name. See "M-Business JavaScript engine proprietary JavaScript objects" on page 335. It is not necessary to look up this prefix every time you want to use a standard PODS function; for all attributes and methods that are exposed to JavaScript, the JavaScript object name is included in the JavaScript synopsis for the attribute or method in the reference material in the *API Reference for M-Business Anywhere* — beginning with the chapter, "PODS API mechanics" [*M-Business Anywhere API Reference*].
- **Custom PODS objects** require you to create a JavaScript object corresponding to your custom PODS object, using the JavaScript `avantgo.createObject()` method. You then can access your POD's functions by simply prefixing the attribute or method name with the name of the JavaScript object you created for your POD. The following section explains the use of `avantgo.createObject()` to access custom PODS objects.

Note
There are several things you must do in your custom POD's code, before it can be accessed from JavaScript. See "Exporting a PODS object to M-Business JavaScript engine" [*M-Business Anywhere API Reference*]. The instructions in the rest of this section assume that you have completed these tasks.

Using CreateObject( ) to expose a POD to JavaScript engine

The JavaScript `avantgo.createObject()` method returns a PODS object exported by a POD currently loaded in M-Business Client. This function queries all loaded PODS modules, passes the name to each, and asks each if it can provide an object in return. `avantgo.createObject()` either returns a PODS object, or if no POD can provide an object with the given name, it returns `NULL`. The synopsis is:

```
CreateObject(name)
```

The *name* argument is a string containing the name of the object or type of object to return.

Most PODs interpret a name passed to `avantgo.createObject()` as a class name (in other words, the name of a type of object) and create a new object instance for each call to `avantgo.createObject()`. For example, Acme Network Company might create a POD used for network communication that can provide objects of the class `"Acme.Socket"`. When this POD is present on a mobile device, a JavaScript author could call `CreateObject("Acme.Socket")` to create a new socket object used for network communication. Each successive call to `CreateObject("Acme.Socket")` would create an entirely new socket.

Other PODs may interpret a name passed to `CreateObject()` as the name of a single object that the POD can return to JavaScript. For example, SuperBank Inc. might create a POD that stores checking information in an electronic checkbook on a handheld device. If there is only one checkbook per device, then the call `CreateObject("SuperBank.Checkbook")` could return it. Successive calls to `CreateObject("SuperBank.Checkbook")` would return the same, unique `Checkbook` object.

The method name, `CreateObject()`, reflects the fact that many PODs create a new object in response to each call to this method. If you are writing JavaScript code that calls a POD and you did not write the POD, you should consult the POD's author or documentation to find out whether the POD creates a new object on each call to `CreateObject()`.

The example below shows that JavaScript code can access a PODS object's properties (for example, `checkbook.balance`) or methods (for example, `checkbook.update()`), just as it can access properties or methods of any other object. In the example, when JavaScript calls `checkbook.update()` or accesses the `checkbook.balance` property, M-Business Client calls the POD, which implements the method and property using native code.

```
<SCRIPT>
checkbook = CreateObject("SuperBank.Checkbook");
</SCRIPT>
...
<INPUT type="text" name="mytext">
<INPUT type="button" name="update" onClick="checkbook.update()">
<INPUT type="button" name="balance"
    onClick="document.myform.mytext.value = checkbook.balance">
```

How M-Business Client uses getMethod( )

> **Note**
> If you do not know the names of methods within a PODS object, no method currently is provided to query the object for a list of its methods.

When JavaScript code invokes a method on a PODS object, M-Business Client calls the object's `getMethod()` method, which is one of the methods in the `PODSObject` interface. The `getMethod()` method has the following prototype:

```
PODSMethod getMethod(PODSString name, PODSString* type)
```

`PODSMethod` is defined as follows in *pods.h*:

```
typedef void (*PODSMethod)(PODSObject* self,...);
```

M-Business Client passes the method's name as the first argument to `getMethod()`. If the object has no method with the given name, `getMethod()` returns `NULL` and a JavaScript error results. Otherwise, `getMethod()` returns a non-null `PODSMethod()`, which is a pointer to a function that implements the method.

As detailed in "Objects, interfaces, and methods" [*M-Business Anywhere API Reference*], as with any PODS method, this function's first argument is a pointer to the object whose method is being called; the function may take additional method arguments as well. The getMethod( ) method also returns a string that defines the types of the method's parameters and return value. For information on this type string, see "Type strings returned by getMethod( )" [*M-Business Anywhere API Reference*].

As an example, suppose that JavaScript code is calling a method compute of a PODS object:

```
var p = CreateObject("Acme.Computer");
var s = p.compute("hello", "world");
```

M-Business Client calls the PODS object's getMethod( ) method, passing the string "compute" as the name argument. Suppose that the object implements the method using the following C function:

```
PODSInt32 compute(PODSObject* *self, PODSString s, PODSString t)
{
    return (PODSInt32) (strlen(s) + strlen(t));
}
```

Then the getMethod( ) method returns a pointer to the compute function. The getMethod( ) method also returns the type string "ss_i", which indicates that the method takes two string arguments and returns an integer. For a complete description of type strings returned by getMethod( ), see "Type strings returned by getMethod( )" [*M-Business Anywhere API Reference*].

When JavaScript engine calls this function, it passes a pointer to the object p as the first parameter (self), then passes the strings "hello" and "world" as the second and third parameters.

Getting and setting properties

JavaScript allows objects to have named properties. When JavaScript code gets a property of a PODS object, M-Business Client calls the object's getMethod( ) method and passes a special method name consisting of the prefix "get_", plus the property name. For example, if p is a PODS object and JavaScript calls

```
var x = p.color;
```

then M-Business Client calls p's getMethod( ) method and asks for a method named "get_color". Any method returned as a property getter must take no parameters, other than the PODS object, and must return a value. For example, the get_color method could have type "_s" or "_I".

Similarly, when JavaScript code sets a property of a PODS object, M-Business Client asks the object for a method whose name is "set_", plus the property name. Any method returned as a property setter must take a single parameter and must return no value. For example, a set_color method could have type "s" or "I". If you execute the JavaScript code

```
p.color = "blue";
```

and the set_color method has type "s", then M-Business Client passes the string "blue" as the method's first parameter. If the set_color method has type I, then M-Business Client reports an error because the string "blue" cannot be converted to an integer.

The *pods.h* file contains the macros PODS_GETTER_PREFIX and PODS_SETTER_PREFIX, which expand to the strings "get_" and "set_", respectively. In your code you should use these macros rather than the hard-coded strings "get_" and "set_". The *pods.h* file also contains the macros PODS_GETTER and PODS_SETTER that expand a property name to the name of its getter or setter method.

Indexed properties

A JavaScript object may be an array, accessed using array index notation ([...]), for example, p[4]. PODS supports arrays by providing get_element and set_element methods, accessed through PODS_ARRAY_GETTER and PODS_ARRAY_SETTER macros. When a JavaScript object is referenced with an array index subscript, M-Business Client calls the object's getMethod() method and passes the special method name, "get_element".

A PODS object may also use arrays as arguments or return values of methods. This requires an implementation of the PODSArray data type. M-Business Client provides an implementation called PODSStdArray, which satisfies this need, but it is not as efficient as an implementation as a POD could supply to itself with knowledge of the data that the array would contain.

For example, suppose that p is a PODSObject and JavaScript code executes:

```
var x = p[7];
```

M-Business Client calls p's getMethod() method and passes the method name "get_element". The getMethod() method might return the type string "i_s" and a pointer to the C function:

```
PODSString getElement(PODSObject* o, PODSInt32 index)
{
    MyObject* m = (MyObject*) o;
    return m->strings[index];
}
```

Then M-Business Client would call the function, passing the integer 7 as the value of the index parameter.

Similarly, when M-Business Client sets an indexed property of a PODS object, M-Business Client calls the object's getMethod() method and passes the special method name "set_element". The getMethod() method should return a pointer to a method that takes two parameters: an integer index and a value. The second parameter may be of any PODS type.

Garbage collection and memory management in M-Business Client

In order to maximize performance on the device, M-Business Client performs major memory clean-up — "garbage collection" functions — only when the user closes M-Business Client.

The amount of memory that becomes unavailable in each hour that a user browses typical pages on the device was found to be minimal — users could browse for days without running into memory problems. On the other hand, the performance degradation experienced by users if garbage collection is performed more frequently was found to be significant enough that users objected: whether garbage collections was performed each time the user moves to a new page, or at a specified time interval like every 5 minutes, users found the slow-down objectionable. Thus the design decision was made early on that M-Business Client would perform major memory clean-up functions only when the user closes the application at the end of a session.

It is possible to create pages that will use up all available memory before the user is ready to close M-Business Client. To avoid running out of memory, developers writing JavaScript or C code to be executed from HTML pages displayed in M-Business Client need to use good programming practices. You must ensure that any allocated memory is freed up as soon as it is no longer needed:

- In JavaScript loops, objects must be properly allocated and variables must be set to NULL when no longer needed, or these items will be missed by JavaScript's garbage collection routines.
- In C code, constructors and destructors must be coordinated, strings must be registered, and any files opened must be closed or memory is not freed.
- In JavaScript or C, programmatically submitting forms from the same page will eventually run out of memory — you must set a maximum number that will be allowed.

For details on JavaScript programming practices that keep applications from running out of memory on M-Business Client, see “Resource allocation limitations” on page 346.

For information on C programming practices that keep applications from running out of memory on M-Business Client, see “Managing memory” [*M-Business Anywhere API Reference*].

Tip
If a user never closes M-Business Client on a device, eventually all available memory will be used up. To avoid this potential problem, you should design applications in such a way that users are encouraged to close M-Business Client at the end of an application session.

CHAPTER 8

Customizing content for users and devices

Contents

Overview of content customization

There are many reasons why you might want to customize the content of your channel pages — that is, have the same URL display a page that is different in its details when it is downloaded *by* different individuals or *to* different devices.

In customizing content for individuals, you typically let the user make some selections which you store as *preferences* in a *cookie*. For example, if you have weather reports available for a number of cities, your users would choose a city and then all the weather reports downloaded as part of your channel would be for that city.

Customizing web pages for user preferences is common on the Internet today. If you have done much desktop web development, you are probably familiar with using cookies to do this.

Customizing web pages for different devices is important if you want your channel to look as good as it can on the different devices on which it may be viewed. The differences in the way that different mobile devices display text and graphics are major, far greater than what you may be used to dealing with getting your pages to look good on both Microsoft Internet Explorer and Netscape Navigator, running under both Windows and Mac OS.

Note
The M-Business Sync Server automatically detects the type of device that is synchronizing and optimizes graphics as best it can for that device. You only need to customize content for different devices when you want to achieve the best possible page display on different devices.

Customizing content involves a fair amount of additional work, but you may only need to do it for a few key pages, like your home page, or for the graphic heading at the top of each page.

Using M-Business Client HTTP request headers to customize content

There are a lot of different mobile devices out there, all running different operating systems, with different screen sizes and different color depths. And for the most part, your M-Business Sync Server does a pretty good job of handling all of these devices pretty elegantly. It dithers your 16-bit color JPEG into a 4-gray image, and we know that if your table width is 100%, it means something totally different on a Palm OS device compared to a Windows Mobile Pocket PC device.

But sometimes you want finer control. The best example of this would be with logos. Sure, a color logo looks great on a high-end color Palm or a Windows Mobile Pocket PC device, but it looks dithered on a black-and-white Palm. And while a posterized 4-color grayscale logo looks nice on a black-and-white Palm, when you look at it on a color device, it looks like a 4-color grayscale logo. What you would really like is the ability to serve up different images based on the device that is viewing your channel.

That is where the M-Business Client headers come in. As you already know, when a browser accesses your web server, it sends across several headers in the HTTP page request. These are informational headers telling your server a little about the browser; what type of browser it is, what host it thinks it is connecting to, the language the client prefers, and so on.

Specifically, there are headers that M-Business Client currently sends with every HTTP request.

Available HTTP request headers

Table 1. HTTP request headers sent by M-Business Client

Header	Explanation	Base 64?[1]	Typical Values
`User-Agent`	Standard `User-Agent` string sent with any browser request.	No	`Mozilla/4.0 (compatible; AvantGo 6.0; FreeBSD)`
`X-AvantGo-ColorDepth`	Bit depth of mobile device accessing your page.	Yes	`2 4 8`
`X-AvantGo-DeviceOS`	Operating system of mobile device accessing your page.	Yes	`PALM_OS WINCE_OS WINCE_OS_SmartPhone WIN32_OS`

`X-AvantGo-DeviceOSVersion`	Version of operating system on mobile device accessing your page.	Yes	`4.2`
			`5.01`

Header	Explanation	Base 64?[1]	Typical Values
X-AvantGo-DeviceID	Serial number for device, available only on Palm and Pocket PC 2002 or later.	Yes	See manufacturer's documentation.
X-AvantGo-OnlineRequest	Is user browsing with live connection (as opposed to normal HotSync.)? Not present if user not online.	No	TRUE
X-AvantGo-Screensize	Size, in pixels, of screen of mobile device accessing your page.	Yes	150x150 240x320
X-AvantGo-SecureSync	Is user browsing with secure connection? Not present if connection not secure.	No	TRUE
X-AvantGo-UserID	Name of M-Business account accessing your page.	Yes	tkwebtest todddev
X-AvantGo-Version	Version of M-Business Client accessing your page. Not present if that version is 3.1.	No	5.7 6.0 6.2

[1] Is header base 64 encoded?

X-AvantGo-DeviceOS values returned by different platforms

- **PALM_OS** Returned by all Palm OS devices.
- **WINCE_OS** Returned by Windows Mobile Pocket PC and Windows Mobile 5 devices.
- **WINCE_OS_SmartPhone** Returned by Microsoft Smartphone devices running Windows Mobile 5.
- **WIN32_OS** Returned by Windows XP devices.

Tip
To determine whether the `WINCE_OS` value returned for `X-AvantGo-DeviceOS` is for Windows Mobile Pocket PC device or a Windows Mobile 5 device, check the value of X-AvantGo-DeviceOSVersion. It is 5.0 or higher for Windows Mobile 5; 4.x or lower for Windows Mobile Pocket PC.

Retrieving M-Business Client HTTP request headers

Note
Information for select HTTP request headers is available to the M-Business Sync Server through server URL macros, without the need to retrieve or decode it. For more information, see "Using server URL macros to customize content" on page 166.

And how exactly do I retrieve HTTP request headers?

That depends on what you are using to create these dynamic pages in the first place. A server running a cgi-bin script, for example, will pass along these values as environment variables with a slightly different name: the whole header is capitalized, hyphens are replaced with underscores, and `HTTP_` is prepended to the header. For example, the M-Business Client request header `X-AvantGo-DeviceOS` is sent across as the environment variable `HTTP_X_AVANTGO_DEVICEOS`.

If you are using Perl with the CGI library installed, you can use a `HTTP('HTTP_X_AVANTGO_DEVICEOS')` function to retrieve the value of the `X-AvantGo-DeviceOS` header. Actually, the `HTTP()` function is smart enough to understand that `HTTP('X-AvantGo-DeviceOS')` is the same thing.

Ask your local Webmaster if you need more help decoding HTTP headers for your particular method of serving up web pages.

Decoding M-Business Client HTTP request headers

Note
Information for select HTTP request headers is available to the M-Business Sync Server through server URL macros, without the need to retrieve or decode it. For more information, see "Using server URL macros to customize content" on page 166.

OK, I retrieved the value of my M-Business Client HTTP headers, and it is all gobbledygook! What is wrong?

This is where it gets a little tricky. Most of the values of M-Business Client headers are Base64-encoded before they are sent across. So you might find that your `X-AvantGo-Screensize` header has a value equal to something like `MTUweDE1MA==`. You will need to decode these headers before you can actually deal with them in a reasonable manner.

Decoding a Base64 Encoded string is generally pretty easy, albeit a little obscure. It is a little different depending on what you are using to serve up pages:

- If you are using Perl, include a use `MIME::Base64` line at the beginning of your script to use Perl's built-in Base64 library. Then you can use the `decode_base64()` function to decode whatever string you would like.
- If you are using PHP, you can use the `base64_decode()` function.

Please note that iAnywhere Solutions cannot provide support for the various third-party libraries. You are better off contacting these third parties directly.

For example, here is a Perl script that prints a custom front page based on the user's mobile device. It analyzes the bit depth and chooses a logo based on that value. Then it analyzes the device's operating system and prints out some text based on that value.

```
#!D:\Perl\bin\perl.exe use CGI qw/:standard/; use MIME::Base64;

# devicecontext.pl # # Todd's sample code for
changing images and text # based on M-Business Client headers # # Note that I
am using Perl on a Apache Server on an
# NT machine, hence the weird first line.

#
Just print out the beginning... print "Content-type: text/html\n";
print "Cache-Control: max-age=7200\n"; print <<END_of_Start;
<HTML> <HEAD> <TITLE>Greetings</TITLE> <META
Name="HandheldFriendly" content="True">
</HEAD>

<BODY> END_of_Start

# And now process the headers. # # We'll only
look for the M-Business Client # headers we're interested in
(COLORDEPTH and DEVICEOS) # # They need to be base64-decoded before we can do
#
anything useful to them so I'm using the decode_base64 # function that's
included in Perl's MIME:Base64 library.

$colordepth = decode_base64(http('HTTP_X_AVANTGO_COLORDEPTH'));
$deviceos = decode_base64(http('HTTP_X_AVANTGO_DEVICEOS')); print
"<center><b>Greetings!</b><p>And welcome
to...<\p>\n";

 if ($colordepth >
2) { # If the colordepth is greater than 2, print the color logo print
"<img src=\"/colorlogo.gif\"
alt=\"MyCompany.com\">\n ";, } else { # Otherwise, we'll go
with the black and white logo print "<img src=\"/bwlogo.gif\"
alt=\"MyCompany.com\">\n"; }

if ($deviceos eq 'PALM_OS') { print "<p>The web site
that puts PalmOS users first!\n"; } elsif ($deviceos eq 'WINCE_OS') {
print "<p>The web site that puts Pocket PC users first!\n"; }
else { print "<p>The web site that puts non-Palm and Pocket PC users
first!\n"; } print
"</center></body></HTML>\n";

# # Copyright (c) 2000, iAnywhere Solutions, Inc., # all rights
reserved. # # IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT # THE
SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR # IMPLIED, INCLUDING BUT NOT
LIMITED TO THE IMPLIED # WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR # PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE # LIABLE
FOR
ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT # OF USING, MODIFYING OR
DISTRIBUTING THIS SOFTWARE OR # ITS DERIVATIVES.
```

Here is the same page in PHP.

```
<?php
header("Cache-Control: max-age=7200"); ?> <HTML>
```

```
<HEAD> <TITLE>Greetings</TITLE> <META
Name="HandheldFriendly" Content="TRUE">
</HEAD>

<BODY> <center>
<b>Greetings!</b> <p>And welcome
to...</p>

<?php
$colordepth=base64_decode(getenv("HTTP_X_AVANTGO_COLORDEPTH"));
$deviceos = base64_decode(getenv("HTTP_X_AVANTGO_DEVICEOS")); if
($colordepth > 2) { print( "<img src=\"/colorlogo.gif\"
alt=\"MyCompany.com\">\n"); } else { print( "<img
src=\"\bwlogo.gif\" alt=\"MyCompany.com\">\n");
}

 if ($deviceos == 'PALM_OS') { print
("<p>The web site that puts PalmOS users first!\n"); } elseif
($deviceos == 'WINCE_OS') { print ("<p>The web site that puts Pocket
PC users first!\n"); } else { print ("<p>The web site that puts
Pocket PC and non- Palm users first!\n"); } ?>

 </center> </body> </html> <?php # #
Copyright (c) 2000, iAnywhere Solutions, Inc., # all rights reserved. # #
IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT # THE SUITABILITY OF
THE
SOFTWARE, EITHER EXPRESS OR # IMPLIED, INCLUDING BUT NOT LIMITED TO THE
IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR # PURPOSE, OR
NON-INFRINGEMENT. IANYWHERE SHALL NOT BE # LIABLE FOR ANY DAMAGES SUFFERED BY
LICENSEE AS A RESULT # OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR #
ITS DERIVATIVES. ?>
```

Why you should not label these pages as Cache-Control: private

Now what about caching? Why not label these pages as `Cache-Control: private`?

The M-Business Sync Server already caches different versions of your web page depending on the screen size, operating system, and bit depth of the mobile device that is loading your page. So if you are only customizing content based on these settings, you can go ahead and cache them normally; there is no need to make them private.

Using server URL macros to customize content

URL macros on your M-Business Server allow you to set up a single channel that delivers different content to different users. The content differences can be based on the user name, the user's device, or even the processor on the user's device. Server URL macros automatically make available some of the information from M-Business Client HTTP request headers. For information on the full range of HTTP request header information available, see "Using M-Business Client HTTP request headers to customize content" on page 161.

Note
Server URL macros automatically make available to the M-Business Sync Server some of the information from M-Business Client HTTP request headers. For information on the full range of HTTP request header information available, see "Using M-Business Client HTTP request headers to customize content" on page 161.

Available server URL macros

The URL macros that are supported by the M-Business Sync Server are detailed in the table below.

Table 2. M-Business Sync Server URL macros

Macro	Description	Typical Values
`AG_DEVICEOS`	The Operating System of the device sending the synchronization request. Useful to optimize content for different device parameters.	`PALM_OS WINCE_OS WINCE_OS_SmartPhone WIN32_OS`
`AG_DEVICEPROCESSOR`	The processor of the device sending the synchronization request. Useful when downloading binaries through your channel to Pocket PC devices.	(Only applies when `AG_DEVICEOS` is `WINCE_OS` or `WINCE_OS_SmartPhone`; blank for all other OS values) `ARM MIPS SH3 SH4`
`AG_CLIENTVERSION`	The version of M-Business Client on the device sending the synchronization request. Useful when different versions of M-Business Client require different binaries.	`5.7 6.0 6.2`
`AG_USER`	The User ID for the device sending the synchronization request. Useful whenever content needs to be user-specific.	The User ID set up on the M-Business Sync Server.

AG_DEVICEOS values returned by different platforms

- **PALM_OS** Returned by all Palm OS devices.
- **WINCE_OS** Returned by Windows Mobile Pocket PC and Windows Mobile 5 devices.
- **WINCE_OS_SmartPhone** Returned by Microsoft Smartphone devices running Windows Mobile 5.
- **WIN32_OS** Returned by Windows XP devices.

Inserting server URL macros into your channel URLs

You use URL macros by inserting them into some portion of the channel URL (**Location** field on the **New Channel** or **Channel Properties** page) when you define or modify a channel. M-Business Client HTTP request headers supply M-Business Server with the specific values with which to replace the URL macro in the channel when M-Business Sync Server receives an M-Business Client synchronization request.

Note
You must ensure that content exists for all the specific URLs that will result when users synchronize and URL macros in the channel URL are replaced by specific values.

Note
If you are able to generate content dynamically, it is more efficient to have a single process with a single URL which generates pages custom pages based on information in the M-Business Client HTTP request headers. See "Using M-Business Client HTTP request headers to customize content" on page 161 and "Using cookies to support personal channels" on page 196.

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error. When you are setting up directories in a Windows file system, where backslashes separate levels, be sure to change these characters to forward slashes if you copy parts of the file system path into a URL.

AG_USER examples

Below are some simplified examples of mapping the `AG_USER` server URL macro to web server directories and files. In these examples, the it is assumed that the web server's domain is `http://myserver.com`, that this domain maps to the file system path *D:\webserver\contentroot*, and that you have three user IDs: `aasmith`, `bjwong`, and `ccgomez`.

- Different files for different users, in same directory:

  Channel URL - `http://myserver.com/myapp/AG_USER.asp`

Files for different users - *D:\webserver\contentroot\myapp\aasmith.htm D:\webserver\contentroot \myapp\bjwong.htm D:\webserver\contentroot\myapp\ccgomez.htm*

- Different directories for different users, with same file name:

  Channel URL - `http://myserver.com/AG_USER/myapp.htm`

  Files for different users -

  D:\webserver\contentroot\aasmith\myapp.htm

  D:\webserver\contentroot\bjwong\myapp.htm

  D:\webserver\contentroot\ccgomez\myapp.htm

- User ID passed as parameter to web application:

  Channel URL - `http://myserver.com/myapp.asp?id=AG_USER`

  Files for different users - *D:\webserver\contentroot\myapp.asp* (The same file, *myapp.asp*, generates different content when each user ID replaces `AG_USER` in the channel URL.)

AG_DEVICEOS examples

Below are some simplified examples of mapping the `AG_DEVICEOS` URL macro to web server directories and files. In these examples, the it is assumed that the web server's domain is `http://myserver.com`, that this domain maps to the file system path *D:\webserver\contentroot*, and that you have two types of device OS to support: Windows Mobile Pocket PC and Palm. The value of `AG_DEVICEOS` for Windows Mobile Pocket PC is `WINCE_OS`, and that for Palm is `PALM_OS`.

- Different files optimized for different device OSes, in same directory:

  Channel URL - `http://myserver.com/myapp/AG_DEVICEOS.htm`

  Files for different users -

  D:\webserver\contentroot\myapp\WINCE_OS.htm

  D:\webserver\contentroot\myapp\PALM_OS.htm

- Different directories for different device OSes, with same file name:

  Channel URL - `http://myserver.com/AG_DEVICEOS/myapp.htm`

  Files for different users -

  D:\webserver\contentroot\WINCE_OS\myapp.htm

  D:\webserver\contentroot\PALM_OS\myapp.htm

- Different directories for different device OSes, with different file names — as necessary for executables for your application.

  Use the different directories... with same file name approach, then within the two files, place links to the files that need to have different names on the two device OSes:

  Channel URL - `http://myserver.com/AG_DEVICEOS/myappprograms.htm`

  Files for different device OSes -

  D:\webserver\contentroot\WINCE_OS\myappprograms.htm

  D:\webserver\contentroot\PALM_OS\myappprograms.htm

  Content of ...*\WINCE_OS\myappprograms.htm*:

  ```
  <html> <a href="mimelist.dll></a> <a
  href="dbpod_quad.dll></a> </html>
  ```

  Content of ...*\PALM_OS\myappprograms.htm*:

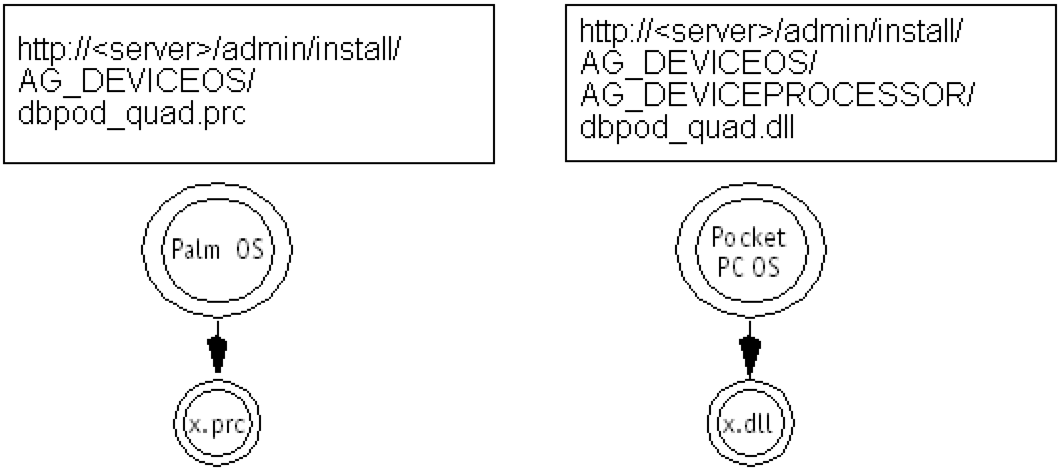
  ```
  <html> <a href="mimelist.prc></a> <a
  href="dbpod_quad.prc></a> </html>
  ```

  For an example of the above concept in context, see the “Master-Detail page example” on page 419. Different binaries are referenced for Windows Mobile Pocket PC vs. Palm at the end of the listing for “Detail page” on page 432.

Setting up binary files for delivery to M-Business Client

M-Business Anywhere includes several important platform-specific extensions to M-Business Client. These extensions are known as PODS and they provide functions like on-device database support, a List Viewer, and a Date/Time Picker. In order to use these extensions, you must set up a channel that can deliver binary files to M-Business Client, and that channel must automatically provide the correct platform-specific version to each user's device.

The following directory structure and URL syntax is for creating a channel that delivers the correct binary executable for the database POD via a single channel to different types of mobile devices.



When you set up this directory structure, M-Business Server uses the `AG_DEVICEOS` and/or the `AG_DEVICEPROCESSOR` variables in the channel pathname to direct (parse) the user synchronization to the appropriate binary. These variables are implemented via server URL macros. For information on additional variables that are available, see "Using server URL macros to customize content" on page 166.

The following table displays patterns for creating a file structure that supports the URL syntax for a set of binary files. Note that M-Business Client would send these URLs containing server URL macros to M-Business Sync Server, which would replace the URL macros with literal values based on the mobile device type. For more information on how this substitution works, see "Using server URL macros to customize content" on page 166.

Table 3. Pattern for syntax of binary distribution

Directory	AG_DEVICEOS	AG_DEVICEPROCESSOR	File name	Binary type

http://filelocation	`PALM_OS`
	`WINCE_OS`
	`WINCE_OS_Smart Phone WIN32_OS`
		`ARM`
		`MIPS`
		`SH3`
		`X86`
			<filename>	*.dll, .prc, .pdb, .cab, .exe, .<any other binary type>*

For example, synchronizing a Pocket PC device with a MIPS processor to a *.dll* would result in a path like this:

```
http://filelocation/WINCE_OS/MIPS/filename.dll
```

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

Note
The platform variables described above should handle most of your platform-specific binary downloads. If you should need to provide different binaries, or no binary at all, for different versions of M-Business Client that are running on the same hardware, you can use the `AG_CLIENTVERSION` variable to distinguish client versions. The `AG_CLIENTVERSION` variable will have a value in the form `<M>.<P>`, where `<M>` is the major version number and `<P>` is the release point number. For the current M-Business Client, this would be `6.7`; for the previous release, `6.5`.

Note
If you created a binary downloads channel in M-Business Anywhere version 5.7 or earlier, implemented through M-Business Server, and simply migrated the database when you upgraded to version 6.7, the old URL for the channel was imported and will no longer work. You will need to manually change the URL for the channel.

Channel setup for binary distribution

If your web server only delivers to a single type of device, or if you want to deliver a specific binary file to a set of specific devices, you can omit the device operating system and device processor variables. Instead, assign a channel to a group of users for a binary type.

Configuring the MIME type with Netscape SuiteSpot servers

The M-Business Admin Server is pre-configured to vend the appropriate MIME types for binary files. If you are using the binary files from the built-in location within M-Business Server, no additional web-server configuration is needed. However, if you are serving the binary files from another web server, you may need to configure that web server to handle the MIME types correctly. Follow the instructions below if you need to do this.

The web server hosting the binaries must be configured to give binaries a MIME type of `application/octet-stream`. Consult your web server documentation for help with changing MIME types.

A sample procedure for enabling a web server is provided below.

♦ To enable a web server in an enterprise where Netscape Suitespot servers are installed

1. Log in to the Netscape Administration server as the admin user.
2. Click the button next to the Netscape Enterprise Server. This is the web server part of Netscape Suitespot.
3. Go to the **Server Preferences** tab, then select **MIME Types**.
4. Look to see if you already have an application/octet-stream content-type, if you do, edit the one already present; if not, add one (category: type).
5. Specify the file extension you are adding in the **File Suffix** file. You can add multiple suffixes, so, for example, to enable DLL and CAB files, you would make the File Suffix equal **dll,cab**.
6. Click **Save**.

When you synchronize the mobile device, the server places *.exe* files in the Start menu; other files are placed in the ***avantgoclient/pods*** directory.

CHAPTER 9

Managing channel form submissions

Contents

Overview of channel form submissions

If your channel includes forms that your readers can submit, you will definitely want to implement the suggestions covered in this chapter. Why? Because at least some of your readers some of the time will be browsing your channel in offline mode. More likely, *most* of your readers *most* of the time will be browsing offline. So when they click the Submit button, they cannot receive a response from the server until the next time that they synchronize.

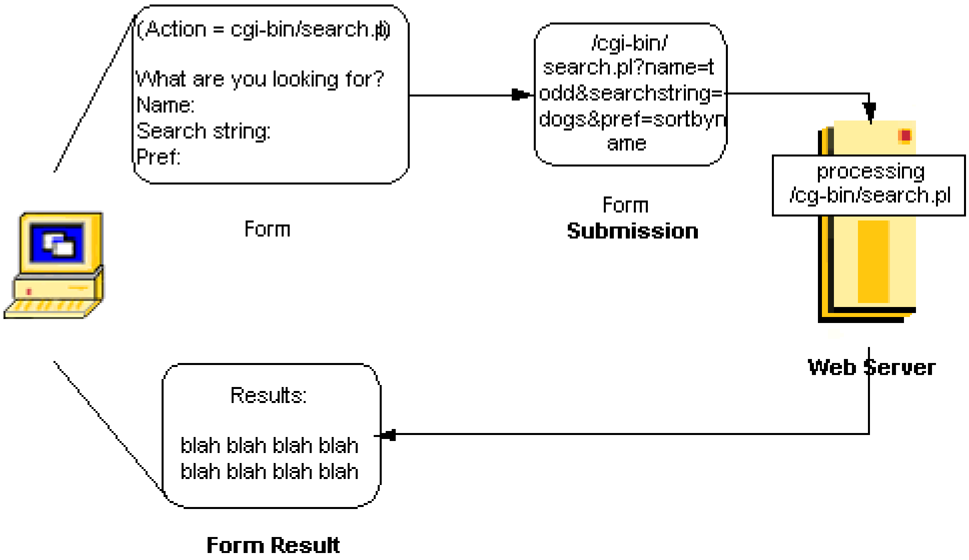
If you do not provide any additional instructions on what should happen after a form is submitted offline, M-Business Client provides a minimal means of handling the form submission — the user receives a message saying that the form will be submitted in the next synchronization and that the response page will be available in something called the Forms Manager.

If you follow a few simple guidelines presented in this chapter, you can give your readers a much more elegant experience, and you have complete control over the messages displayed and how the response pages are accessed after the next synchronization.

If you set up offline form submissions to work just the way you want, M-Business Client will automatically handle any online form submissions the right way — by returning the response page immediately, just like a desktop web browser would.

Forms in the online world

You should already have a sense of how forms work in the normal online world of desktop web browsers. You create a form and associate it with an action — often a cgi-bin script of some sort. A user enters data into the form through text fields, checkboxes, and so on. Using M-Business Client's capabilities, you also can create a form with fields pre-populated with a user's email address and zip code, and, if you feel like being clever, you can enter data for the user through hidden fields. Regardless of what elements are in the form, the data is passed on as parameters to the action associated with that form. The action will perform some sort of processing on the data received, and then outputs its results in the form of an HTML document.



This all works fine in the online world, like with your desktop web browser at home. You are connected to the web server when you first get the form, you are connected when you complete the form and transmit it to the action, and you are still there when the action is complete and the web server has an HTML document to throw back at you.

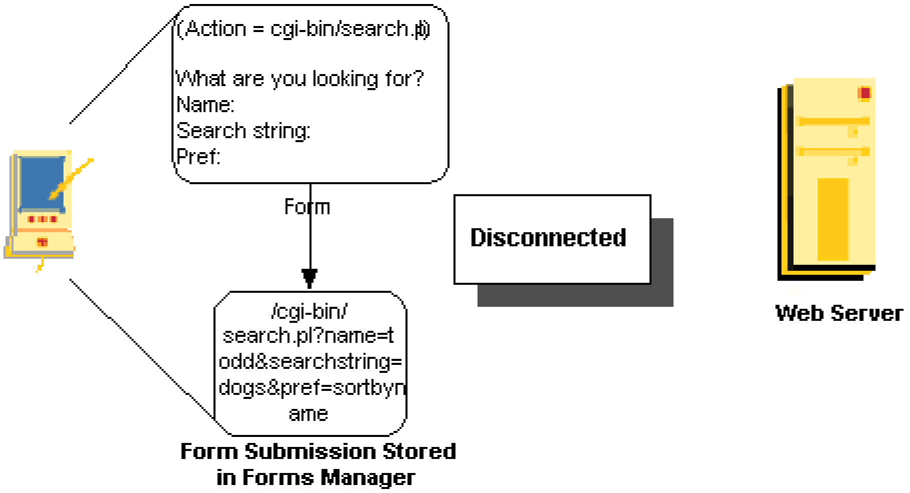
Indeed, this is exactly how M-Business Client works when you are using a mobile device that is in online mode (usually via a modem). You can access forms, send your input, and view results all at once. However, if you are like the majority of mobile device owners without a modem, forms work a little differently.

Forms in the offline world

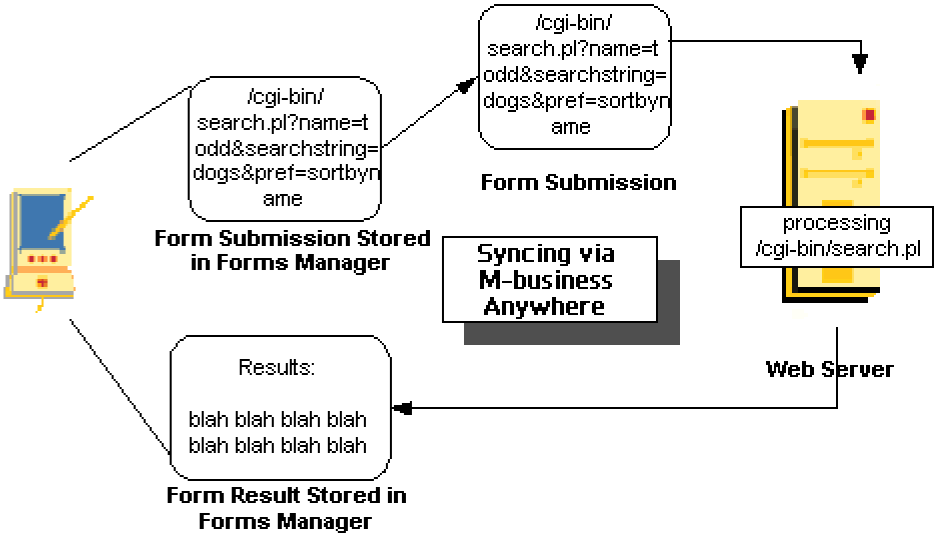
The majority of mobile devices do not come with modems, so dealing with forms and user input is a little more tricky. Your users are viewing your forms offline. They can enter all their data into the forms using text fields, checkboxes, radio buttons, and all the usual form objects.

What the Forms Manager does

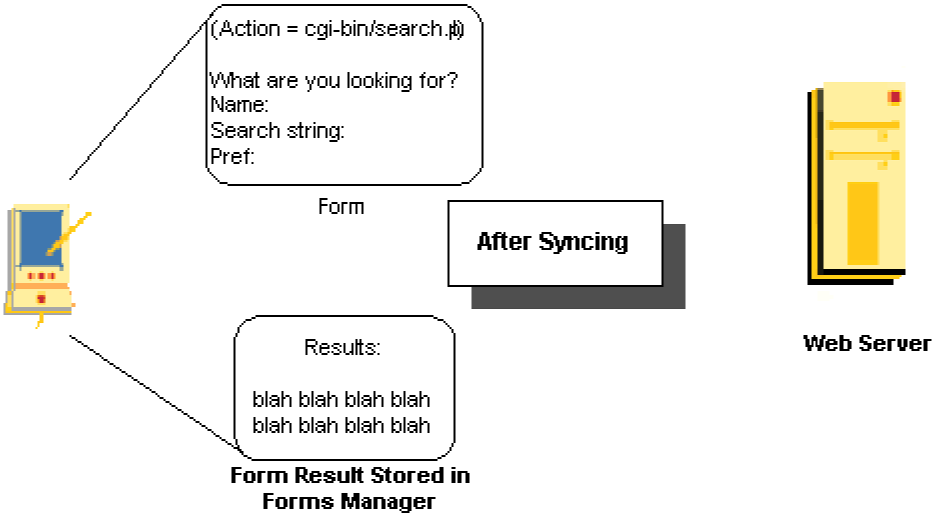
When users submit a form, it waits around on the device in a repository called the Forms Manager. This is where your form is kept until your mobile device is synchronized again. At that point, the data is sent to the cgi-bin script (or other action) associated with the form. It is processed, then the resulting HTML page is sent back to the mobile device.


(Action = cgi-bin/search.pl)
What are you looking for?
Name:
Search string:
Pref:
Form
Disconnected
Web Server
/cgi-bin/search.pl?name=todd&searchstring=dogs&pref=sortbyname
Form Submission Stored in Forms Manager

There is just the one remaining question of where to place the results. It would probably be a bad idea to replace your channel's front page with the form results. So again we place the result into the Forms Manager. This is the area on your mobile device where we store both the filled-out forms waiting to be processed, as well as the returned results for the user to view.


/cgi-bin/search.pl?name=todd&searchstring=dogs&pref=sortbyname
Form Submission Stored in Forms Manager
/cgi-bin/search.pl?name=todd&searchstring=dogs&pref=sortbyname
Form Submission
Syncing via M-business Anywhere
processing /cgi-bin/search.pl
Web Server
Results:
blah blah blah blah
blah blah blah blah
Form Result Stored in Forms Manager

Within M-Business Client, if you select the menu, and then choose **Channels»Forms Manager**, you will find yourself looking at any forms you currently have queued for processing, along with any earlier forms you have submitted and their results.



So when a user submits a form, it is stored on the mobile device, processed during the next synchronization, and the result is kept in the Forms Manager for the user to read at his leisure. In theory.

Why you shouldn't use the Forms Manager for end users

In theory, you can create forms the exact same way as you did before, for online submission. In practice, it turns out that most people do not know or remember to look in the Forms Manager for their results. The Forms Manager is useful for corporate users who like keeping a record of their past queries and, more importantly, have received some training about the Forms Manager. If you are designing for a consumer M-Business channel, your readers are going to expect to find the results somewhere in your channel.

What would be really nice is a link on your front page that says something like, To see the results of your last request, click here! This is, in fact, possible. In order to fully understand the process, you are going to need to know a little about personalizing an M-Business Client. We will revisit this issue later, in "Using personal channels to bypass the Forms Manager" on page 185.

Submitting forms the right way

If you want your channel form submissions to work nicely, whether the user is online or offline, all you have to do is be sure that the form submissions work well in offline mode. This section describes how to design the forms themselves. The next section, "Using personal channels to bypass the Forms Manager" on page 185, describes how to manage the response pages that are returned.

Using submitNoResponse() with the Submit button

Assume you have set up your Movie Review Channel the way we have outlined in "Cookie personalization example: a Movie Review Channel" on page 196 and described more thoroughly in "Implementation details for this example" on page 200. Everything should be working well, but it is not quite perfect. There are a couple of improvements you can make to your channel.

For one thing, even though you have essentially obviated any need for the user to go to the Forms Manager, users' form submissions are still sent to the Forms Manager, and the response (the debug information we receive from the `setprefs.pl` script) is still sent into your Forms Manager. This will not ruin anything, of course, but all the debug information is not a pretty sight.

While we are at it, the Your submission has been recorded and will be sent during the next Synchronization message could probably be personalized. Also, it is a little weird that as soon as a user finishes sending his submission, he is sent right back to the form he just filled out, with the values that existed *before* he entered his data. A user who is not familiar with the inner workings of your channel might think that his results were not processed.

Luckily, you can fix all of this with just one line.

Look at the `Submit` section of the form produced by the `editprefs.pl` script. Right now, it is just a standard Submit button. But M-Business Client allows a few customizations that let you tweak the submission options to work better with your channel.

For instance, within the `editprefsform.pl` script, we create the submit button with the following HTML:

```
<input type="submit" name="submit" value="Submit">
```

Instead, you can create it with this little bit of JavaScript:

```
<input type="button" name="submit" value="Submit"
    onClick="document.forms[0].submitNoResponse(
        'Your updated movie choices will be here next time you sync',
        true, false);
    back()"
>
```

So what does this all mean? The rest of this section explains the JavaScript above.

The first line just creates a standard button. It is the `onClick` part that looks a little unfamiliar. We are calling a `submitNoResponse()` function that is specifically for M-Business Client.

The `submitNoResponse()` function uses the format:

```
submitNoResponse(confirmMessage, isHidden, isMulti)
```

The table below describes the arguments.

The `submitNoResponse()` function tells M-Business Client that you want to submit the results of this form, but to not keep the results in the Forms Manager. Note, however, that the results are still looked at for useful commands like `Set-Cookie` in the HTTP header. It is just not kept around after being viewed. Since the result of the `setprefs.pl` script is just some debug information, you want to use `submitNoResponse()` to hide those results.

Table 1. Arguments for submitNoResponse()

Argument	Description
`confirmMessage`	A string that includes the message you want to display to your user instead of the usual Your submission has been recorded... box.
`isHidden`	This boolean value determines whether you want the form to even be visible in the Forms Manager before submission. If you want users to have the ability to go into their Forms Manager and alter their data, then make this false. If you want the forms to be hidden immediately after submission (either for neatness sake, or because you have confidential information entered into a form that you do not want a casual observer to see), then make this true.
`isMulti`	This Boolean value determines whether you want to allow multiple form submissions. If this is set to false and the user enters data into a form more than once, only the most recent version will get sent to the server. For example, if you have a user who changes his mind a lot and enters his preferences 87 different times before synchronizing, you do not want to process 87 different forms. You only want to process the most recent form, so you would set this to false. If you were making a stock ticker page, for example, and could reasonably expect a user to enter 5 different forms to indicate 5 different stocks to follow, you would set this to true. **Note:** If you set `isMulti` to `true`, do not assume the forms will be processed in the same order they were entered into the mobile device. All forms in the Forms Manager are sent to be processed at once.

Note
The `submitNoResponse()` function is specific to the M-Business Client browser. If you try to run this page with Internet Explorer or Netscape, it will generate JavaScript errors. If you occasionally want to test your channel pages in IE first, start out with a normal `Submit` button. Then when you are ready to test it out on your mobile device, replace the `Submit` button with a `submitNoResponse()` button.

Finally, the `back()` command at the end is a piece of JavaScript that takes the user back to the previous page when he is all done editing his preferences. You probably do not want to keep the user around at this form once he is finished — the form will revert back to its pre-filled out stage after the user is done with it, and it might trick him into thinking his data got thrown out.

There are some other interesting tricks you can pull with forms and JavaScript.

- For M-Business Client 5.x and higher JavaScript, see "JavaScript" on page 320.

Caching guidelines

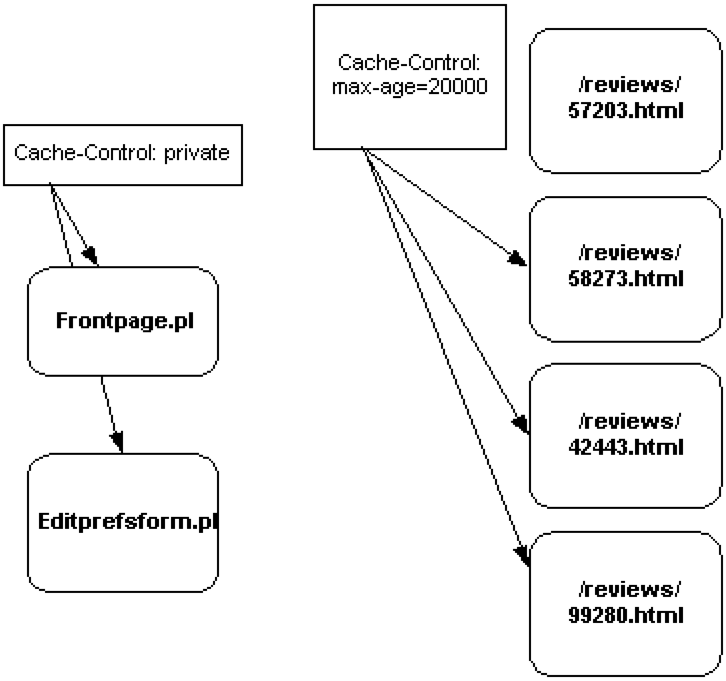
I am sure that, like any good channel developer, you are constantly thinking about the best way to cache your content. If you were hiking in the woods and you suddenly walked between a grizzly bear and her two cubs, the first thought that should go through your mind is, Will this affect the caching of my M-Business channel?

Cache any static pages in a personalized channel as usual

At first glance, it might seem like a personalized page is uncacheable. After all, there is no way of predicting when a user will change his preferences. And that much is true. But there are still some steps you can take to improve the cacheability of a channel. For instance, in the Movie Review Channel, while the front page may change all the time, the movie reviews it points to are all static and can be set to cache for long periods of time.

Cache personalized pages on the mobile device

Second, you can add a `Cache-Control: private` header onto the result of your `frontpage.pl` and `editprefsform.pl` scripts. This means that the pages are cached on the mobile device (but nowhere else). The M-Business Sync Server will always execute `frontpage.pl` every time a user synchronizes (because there is no `max-age`), but it will take a look at the resulting HTML page and compare it to what exists on the user's device. If they are exactly the same, it will not bother uploading the page onto the mobile device again, thereby saving a good chunk of time.



For more information about caching, be sure to check out “Caching to improve channel performance” on page 81.

Accommodating online users

To review, there are two ways you can view an M-Business channel.

- **Offline mode** is the way a vast majority of users currently view M-Business Client. This means that they synchronize their channels, either through their desktop computer or through a modem, disconnect their mobile device from the Internet (usually by disconnecting the device from its cradle or by hanging up on their modem) and then take the mobile device with them to browse pages at their leisure.
- **Online mode** means that users have a live connection to the Internet, usually through a modem attached to their mobile device or through the cool PDA/phone hybrids. It is true that even users with modems tend to browse M-Business channel pages in offline mode, as keeping a modem turned on often drains batteries and costs money. But you will find that as modems become more common (and plans become cheaper) the number of people viewing your page in online mode will be a fairly significant portion of your readers.

As it turns out, M-Business Client works slightly differently for people who are online. When an online user submits a form, the `submitNoResponse()` button acts just like a normal Submit button. It will send the form and immediately retrieve and display the response. Because M-Business Client jumps to the new response page, it never has a chance to process the `back()` command that comes after the `submitNotResponse()` function.

With our current script, the user will be presented with the debug information we have been trying so hard to hide with the `submitNoResponse()` button. Technically speaking, the `submitNoResponse()` button does work as intended, in that it does not store this result in the Forms Manager. In typical computer fashion, it is doing what you tell it to do, not what you want it to do.

This means that, in order to accommodate your online viewers, you have to make sure the response page you get back from the script is still something you would be comfortable sharing with your viewers.

Designing a response page for online users

Here are a few suggestions for what you can display instead of the debug information.

1. Make your response page a simple link back to your front page.

   Thank you for your input! Please go back to the `<a href="/cgi-bin/frontpage.pl">Front Page<a>` to read more reviews. (And do not forget to reload the page to see your new movies.)

2. You can generate the same HTML that would have been generated by the `frontpage.pl` script, so the user is presented with his new front page as soon as he presses the Submit button.

3. If you feel like bring really clever, you can include a JavaScript redirect...

   ```
   BODY onload="window.location='/cgi-bin/movies2/frontpage.pl'">
   ```

   ...to automatically bring the user back to the front page. (Note that this command will only work if you have the `HandheldFriendly META` tag in your response page and the user is in online mode — do not expect redirects like this to work in offline mode.)

Personally, I like the first option the best. Although it is not as clever as the other two, I feel it is the least disorienting.

Your new and improved channel is up and ready to go. Below are cross references to the source code for the new and improved Movie Review Channel, version 2 — or should we say the sequel?

- `frontpage.pl` — see "frontpage.pl, version supporting online users" on page 395.
- `editprefsform.pl` — see "editprefsform.pl, version supporting online users" on page 398.
- `setprefs.pl` — see "setprefs.pl, version supporting online users" on page 400.

And, just to be complete, here is a different way of creating the Movie Review Channel, by simply storing all of the user's preferences in a cookie instead of in a separate database.

- `frontpage.pl` — see "frontpage.pl, no-database version" on page 401.
- `editprefsform.pl` — "editprefsform.pl, no-database version" on page 404.
- `setprefs.pl` — "setprefs.pl, no-database version" on page 406.

Well that was a lot to take in at once, but the information included here will allow you to create all sorts of interesting channels. With this knowledge under your belt, find out how to best create form responses using an additional personalization method.

Using personal channels to bypass the Forms Manager

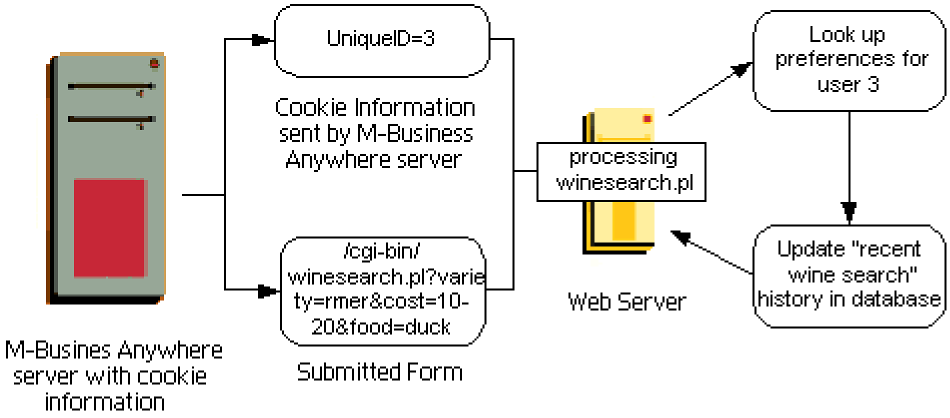
If you have been following everything closely up to now, you probably have a good guess as to how to return form submissions without having to resort to the Forms Manager — you make it just another personalized channel.

A hypothetical Wine Finder Channel

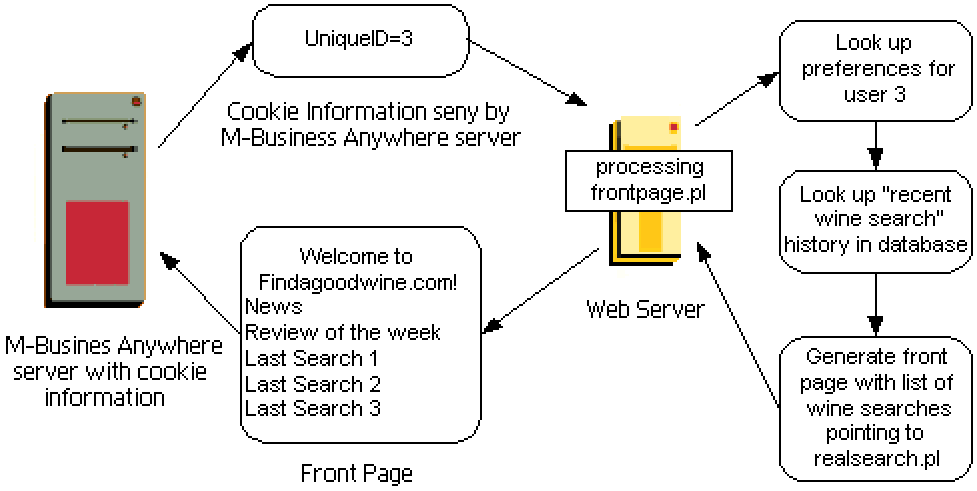
Assume you are creating a hypothetical channel — findagoodwine.com. — and you want to return form submissions through a personal channel. This web site contains the ever-so-useful Help me pick out a wine! form. A user enters the price range, preferred category, region, and food being served (if any), and your handy web site prints out a list of recommended wines.

You are going to create a personalized channel, but instead of keeping track of your user's preferences, you will keep track of his last three requests. The general flow of the channel would work something like this:

1. You set up a unique findagoodwine.com cookie for each user. This cookie in turn matches a user up with a unique ID stored on a local database where you would keep track of the three most recent wine searches they have requested along with their corresponding results.
2. When a user requests a search, the `winesearch.pl` script does not actually conduct a search. Instead, it updates the user's search history in its database.

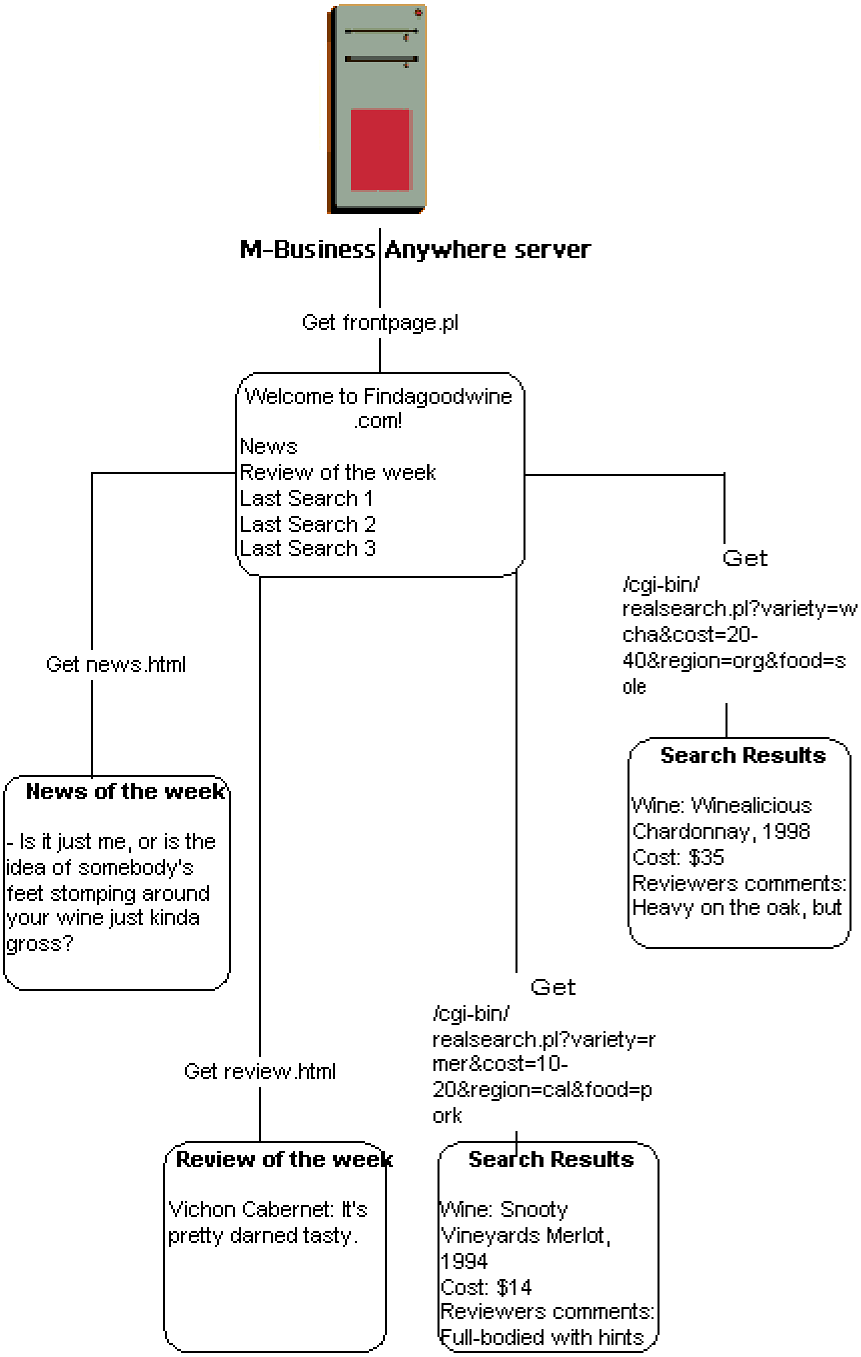

UniqueID=3
Cookie Information sent by M-Business Anywhere server
/cgi-bin/winesearch.pl?variety=rmer&cost=10-20&food=duck
Submitted Form
M-Busines Anywhere server with cookie information
processing winesearch.pl
Web Server
Look up preferences for user 3
Update "recent wine search" history in database

3. When you generate the user's front page, you include links on the front page to his three most recent searches, as indicated in the database. The user can follow those links, and the link will take him to a page that displays the actual results of those wine searches.


UniqueID=3
Cookie Information seny by M-Business Anywhere server
processing frontpage.pl
Web Server
Look up preferences for user 3
Look up "recent wine search" history in database
Generate front page with list of wine searches pointing to realsearch.pl
Welcome to Findagoodwine.com!
News
Review of the week
Last Search 1
Last Search 2
Last Search 3
Front Page
M-Busines Anywhere server with cookie information

4. If no such cookie exists (or there is no history of any previous searches stored on the database), we simply generate our usual front page without the list.

5. When the user synchronizes the channel, the M-Business Sync Server will follow the links it finds on the front page, including any and all searches you may have specified.


M-Business Anywhere server
Get frontpage.pl
Welcome to Findagoodwine .com!
News
Review of the week
Last Search 1
Last Search 2
Last Search 3
Get news.html
News of the week
- Is it just me, or is the idea of somebody's feet stomping around your wine just kinda gross?
Get review.html
Review of the week
Vichon Cabernet: It's pretty darned tasty.
Get /cgi-bin/realsearch.pl?variety=rmer&cost=10-20&region=cal&food=pork
Search Results
Wine: Snooty Vineyards Merlot, 1994
Cost: $14
Reviewers comments: Full-bodied with hints
Get /cgi-bin/realsearch.pl?variety=wcha&cost=20-40&region=org&food=sole
Search Results
Wine: Winealicious Chardonnay, 1998
Cost: $35
Reviewers comments: Heavy on the oak, but

6. Before you publish your channel to users, you set up caching.

It could really load your web server if you have to regenerate a user's last three searches every time he synchronizes. Is there anything you can do? Yes. Once again, this can be fixed through the magic of caching.

- First, make sure your form responses include a line such as `Cache-Control: private,max-age=604800`, since these are old search results and they probably will not change.
- Second, make sure your form submissions use `METHOD=GET` instead of `METHOD=POST` — the M-Business Sync Server looks at the URL to determine if a page should be cached or not.

  If you use `METHOD=GET`, the URL will uniquely identify the search and we will not have to go out and retrieve it all over again. When the M-Business Sync Server notices it is retrieving a URL that already exists in the client cache, it will not bother getting an entirely new (but exactly the same) search on your server.

Note
Note that if you *do* have a search that you want to update every time the user synchronizes (maybe the user is searching for available items on your auction site), just do not include the `max-age` part of the `Cache-Control` header.

Note
An alternate way is to store the results of past searches as local files on your server and have links on your front page point to these files. This is useful if you want more direct control of the resulting files, or you do not feel like generating URLs that exactly duplicate the long search strings, or you just do not trust our caching mechanism.

Source code for the Wine Finder Channel

By now, you probably understand the personalization process enough that you could go and replicate the `findagoodwine.com` channel all on your own. But just in case, here is the source code I used to generate the channel.

- `frontpage.pl` generates a front page. See "frontpage.pl" on page 408 in the sample code appendix.
- `searchform.html` is the plain HTML form used to submit a search. See "frontpage.pl" on page 408 in the sample code appendix.
- `winesearch.pl` stores wine search information in a database. See "frontpage.pl" on page 408 in the sample code appendix.
- `realsearch.pl` is the real search mechanism — lacking real wine data, we just generated random results.

You should now have everything you need to create some really interesting channels.

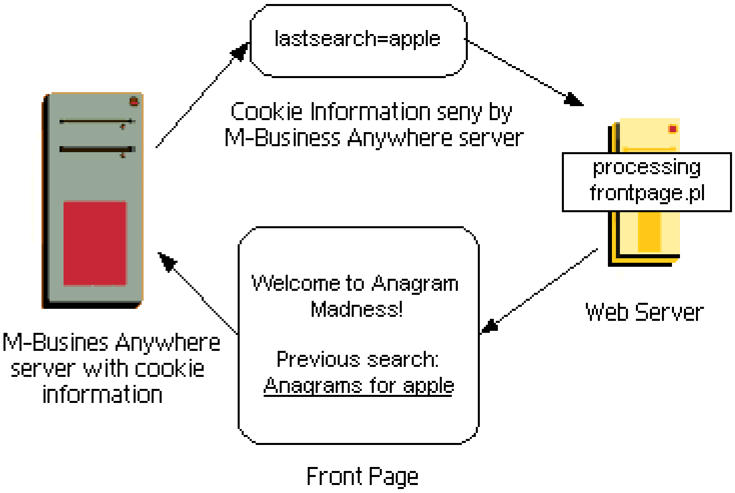
Avoiding collisions in multiple offline submissions

When the same form is submitted more than once while the user is offline, cookies in the successive submissions can interfere with each other when the M-Business Sync Server processes them. There are several ways to avoid this problem.

The problem — how offline submissions can collide

Can you see any reason why the following channel setup will not work?

Suppose you have a channel that lets users look up anagrams for any text string they enter. Being a good channel developer, you do not place these results in the Forms Manager. Instead, you create a cookie with a `name="lastsearch"` and the value equal to the string that the user is requesting anagrams for. You use this cookie to create a link on the front page that links to their anagram results.

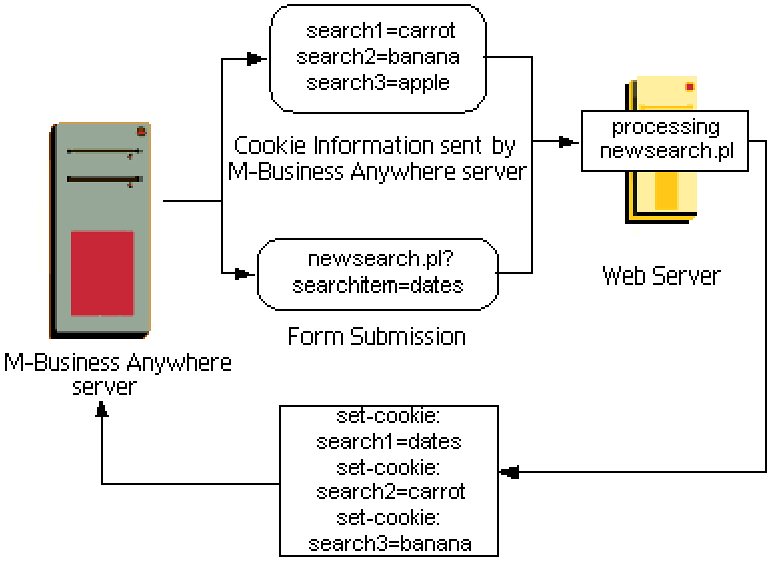


So far so good.

A while later, you start getting requests from users to keep track of more than one search. Suppose somebody wants to enter several anagram searches between synchronizes, or wants to keep track of a previous anagram search. It would be great if you could keep track of the last three searches. It seems like a good idea, so you go with the following strategy:

- You keep three cookies: `search1`, `search2`, and `search3`.
- Every time you get an anagram search request, you take the value of `search2` and put it into `search3`, and take the value of search1 and put it into `search2`.
- Then you take the new search and put it into `search1`.

  In this way, you always have the three most recent searches.



But something weird happens. Suppose your last three searches consist of Carrot, Banana, and Apple. You then enter three searches for Dates, Eggplant, and Fig and then synchronize. You would expect that your last three searches would then consist of Fig, Eggplant, and Dates. But no — instead, you get Eggplant, Carrot, and Banana. What happened? Why does it seem like two of your three form submissions got dropped?

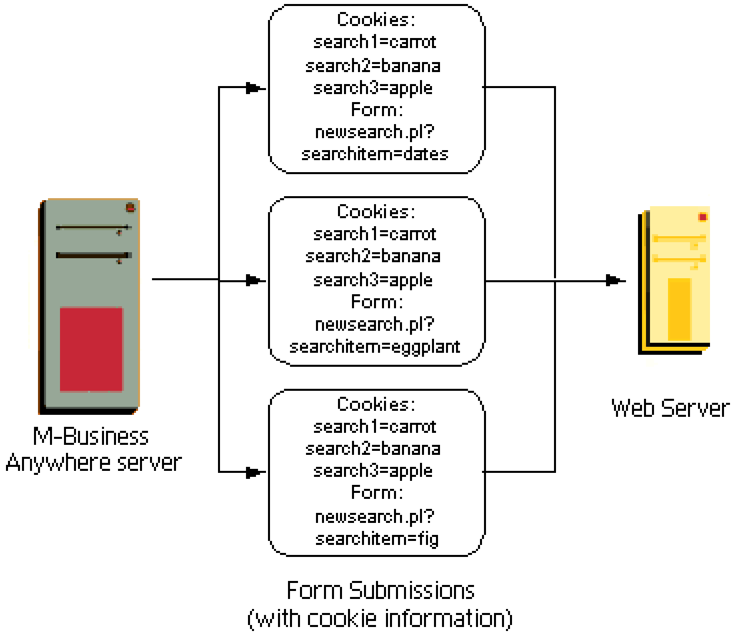
If the `isMulti` flag of your `submitNoResponse()` button is set to `true`. If it is set to `false`, then only the most recent form submission will be sent and the other search requests will be dropped. In this case, assume that you remembered to set `isMulti` to `true`.

So what else could be wrong? Read on!

The answer is that all three forms are submitted at once when you synchronize. Take a close look at what happens in our anagram channel example.

A form submission searching for anagrams for Dates is sent off to your server with cookie `search1=carrot`, `search2=banana`, and `search3=apple`.

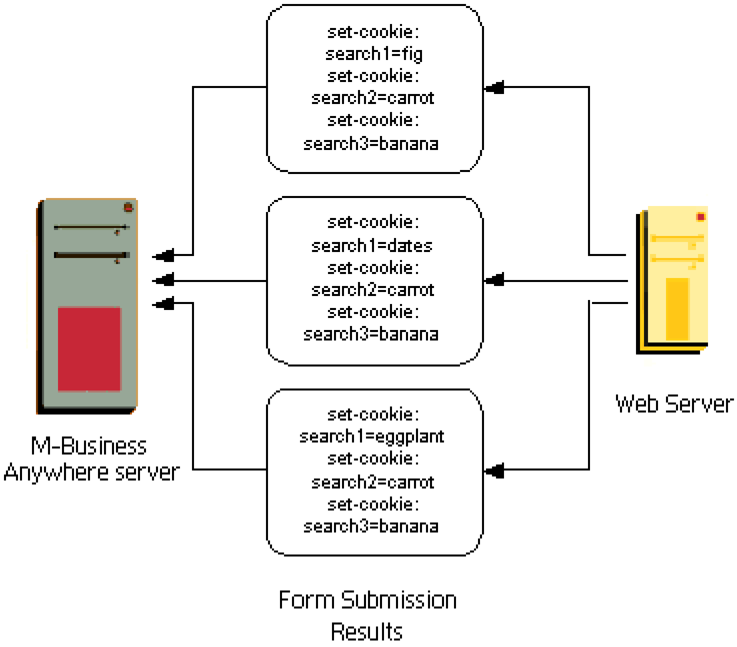
At the same time, form submissions for Eggplant and Fig are also sent off with cookie `search1=carrot`, `search2=banana`, and `search3=apple`.



Suppose the fig submission is finished first. (There are not all that many anagrams for fig, after all.) Your server sends back a response page that sets cookie `search1=fig`, `search2=carrot`, and `search3=banana`.

The Dates submission finishes next. It also sends back a response page that sets cookie `search1=dates`, `search2=carrot`, and `search3=banana`. Remember, the form action has no idea that fig was entered anywhere into the cookie list. It only knows about the cookies that existed at the time the form was submitted. When the M-Business Sync Server receives this, it overwrites the values of the currently existing cookies, essentially sending the `fig` value into oblivion.

Finally, the Eggplant submission finishes last. It sends back a response page that sets cookie `search1=eggplant`, `search2=carrot`, and `search3=banana`. This the last response page that sets these cookies, so it is only these changes that stick.



After all the form responses are received, M-Business Sync Server then goes about retrieving your front page for your channel. This channel looks at the user's cookies and includes a list to their three previous searches: Eggplant, Carrot, and Banana.

Solutions

So how do you get around this problem? Fortunately, there are several solutions:

- "Use a database, store only user ID in cookie" on page 192
- "Use multiple cookies" on page 193
- "Use multiple cookies with relative expiration dates" on page 194

Use a database, store only user ID in cookie

One solution is to use the same strategy that we used in "Customizing content for users and devices" on page 159 — that is, store only the user's ID in a cookie and keep track of his three last searches in a database. Any database worth its salt will make sure that one request is finished altering a row before the next one is allowed to start.

Caution

Even here, there is room for error. Suppose your program does something like this:

Step 1: `SELECT search1, search2, search3 FROM searchtable WHERE userid= $cookie_userid;`

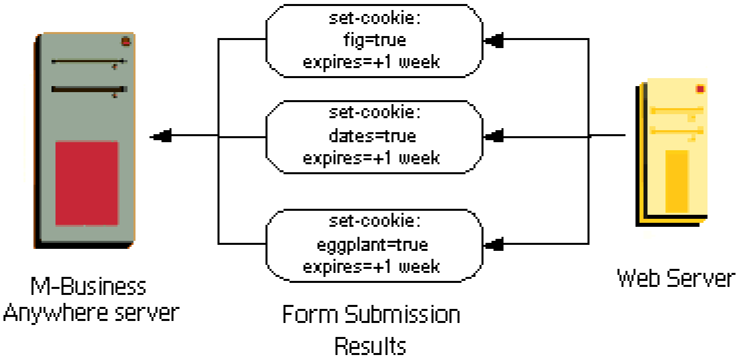
Step 2: (code to set `$newsearch3=search2`, `$newsearch2=search1`, `$newsearch1= $new_search_string`)

Step 3: `UPDATE searchtable SET search1=$newsearch1, search2=$newsearch2, search3=$newsearch3 WHERE userid=$cookie_userid;`

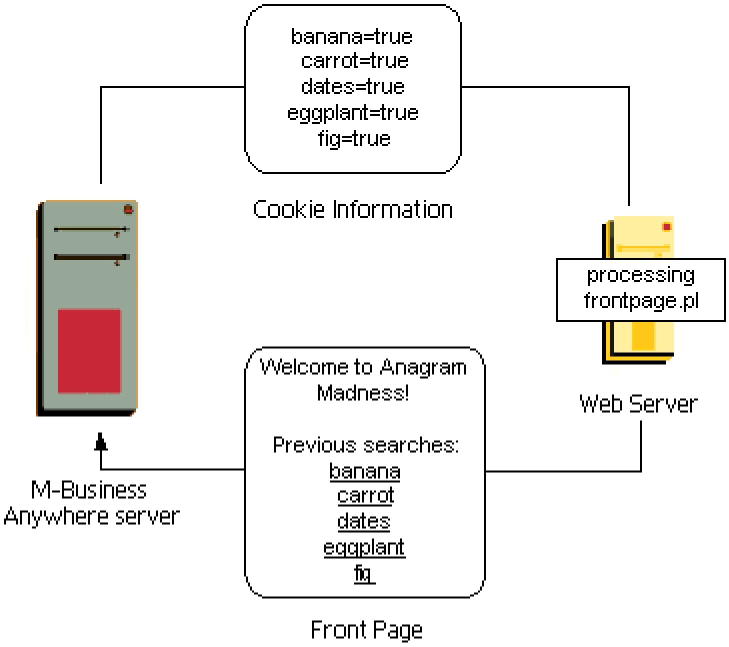
You can run into problems if a concurrently running program executes an `UPDATE` while you are busy running step 2. You will probably want to insert a `LOCK` statement before step 1, and an `UNLOCK` statement after step 3, to prevent other programs from reading that row until you are finished altering it.

Use multiple cookies

Another solution is to make every cookie have its own name. That is, rather than having cookies named `search1=date`, `search2=carrot` etc., create cookies that are named `eggplant=true`, `fig=true`, `date=true`, and so on. That way, you can set multiple cookies and you do not have to worry about any conflicts.



At this point, you would probably set the expiration date for each cookie to 1 week into the future, and create an all searches in the last week section instead of a last three searches section.



Use multiple cookies with relative expiration dates

You could also get a reasonably accurate last three searches section if you really wanted by creating cookies with relative expiration dates and then have your front page script pick the three cookies with the latest expiration dates and delete all the others. But an All searches in the last week section is probably sufficient.

Serializing forms: breaking one form into several pages

Sometimes it is desirable to break a single form into several logical pages, which compose a single form submission. There are two basic approaches to doing this in M-Business Anywhere:

- **Serialize from a single page** — Put all of your form fields on a single page, then use DHTML's CSS `display` attribute to selectively hide and display information for the different logical pages, so that only one "page" appears at a time. All "pages" are processed in a single form submission, preserving the idea of one form/one page.
- **Serialize using Submission Manager** — Use DOM operations on the Submission Manager to build multiple pages, and a single form, in which the application operates on a single form submission. As a page is loaded, the value is concatenated to a submission whose name is stored in a persistent datastore to keep track of all the associated form values. The persistent datastore can be the `avantgo.preferences` object, the Forms Manager, or the on-device M-Business XML datastore. This example uses the `avantgo.preferences` object.

A pair of code samples, each illustrating one of the above approaches, is available online. See “Serializing form submissions with M-Business Client” on page 386.

Using cookies to support personal channels

You probably know the basic concept of a cookie, which is that it is a personalized chunk of information stored in a user's browser that your site, and only your site, is allowed to read and alter. They are often used to keep track of a user's preferences so you can adjust your site to their personality. But remember, you should never store highly sensitive information, such as a user's credit card number, within a cookie.

For example, `my.yahoo.com` knows to display the results of the most recent Red Sox game whenever I visit. (And to display a Better luck next year message at the end of the season.) It does this by reading the cookie on my desktop. This cookie could either contain a) A list of my baseball team preferences, or more likely, b) A unique UserID that is used in coordination with a database on Yahoo's server to keep track of all my preferences.

Cookies through M-Business Client and M-Business Sync Server work in much the same way. The only difference is that cookies are not stored on the mobile device. They are stored on M-Business Sync Server and are associated with the user's M-Business Anywhere account. In practice, this should not affect the way you send or receive cookies, except that you cannot use the JavaScript `document.cookie` variable.

Setting cookies in your HTTP headers

This means that you set cookies by including a `Set-Cookie` line somewhere in your HTTP headers, and you receive a cookie by examining any `Cookie` header sent to your server. Actually, if you are running a CGI script, the server will pass the cookie information along to the script in the form of an `HTTP_COOKIE` environment variable. Depending on how you are serving your pages, there is probably a short-cut for doing this.

If you are using Perl, there is a `CGI::Cookie` module available from:

http://www.perl.org/

Using Perl's `CGI::Cookie` module, you could retrieve cookies with the `fetch CGI::Cookie` command, create them with `new CGI::Cookie` and set them using a `print "Set-Cookie: $mycookie"` command.

If you are using Active Server Pages, you would use the `Request.Cookies()` and `Response.Cookies()` functions to retrieve and set cookies. There is a very good chance you are using these already. Ask your local web guru for more information.

Cookie personalization example: a Movie Review Channel

So with our knowledge of cookies, we will tackle a typical example of a personalized channel. We will also need to draw on some information about forms, which are covered in detail in “Managing channel form submissions” on page 173.

Suppose you were creating The Movie Review Channel, a (fictional) channel that provides new DVD, video, and theater reviews daily. It tends to be a lot of information and the reviews can be pretty long-winded. So it would be nice if you could pre-screen them and only present the ones the user is interested in. Maybe your reader is into action flicks. Or comedies. And maybe the idea of seeing a family-oriented kiddie film makes him break out in a rash.

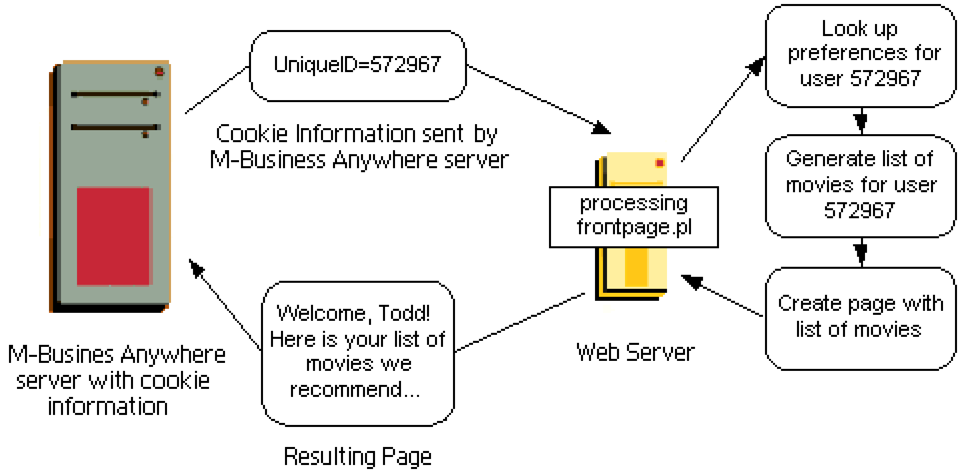
What you would like is the ability for any user to customize his movie preferences through a form submission. He could go to a form, select the movie genres he prefers, and the next time he synchronizes, he would be presented with a list of movies that fit into one of those genres.

Design concept: user ID in cookie, the rest in a database

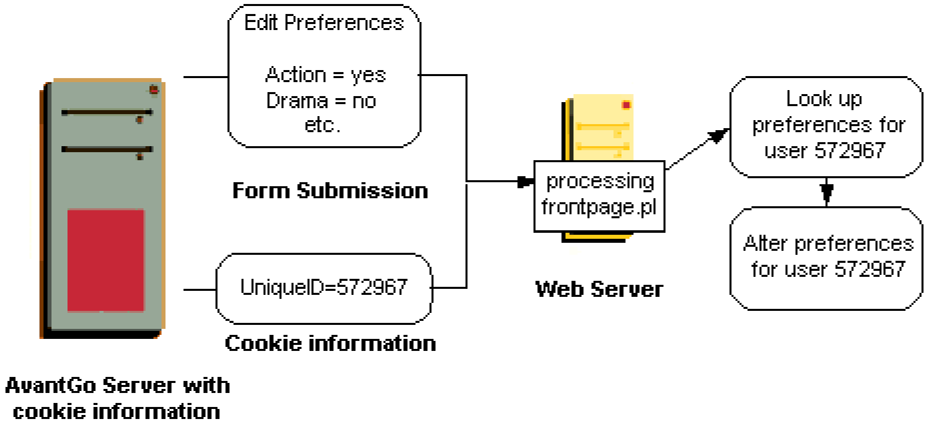
Here is the high level, birds-eye view of the process involved in creating this channel:

You create a cookie for the user that contains a the user's unique ID. You keep a database on your server that contains a list of every user, along with his genre preferences.

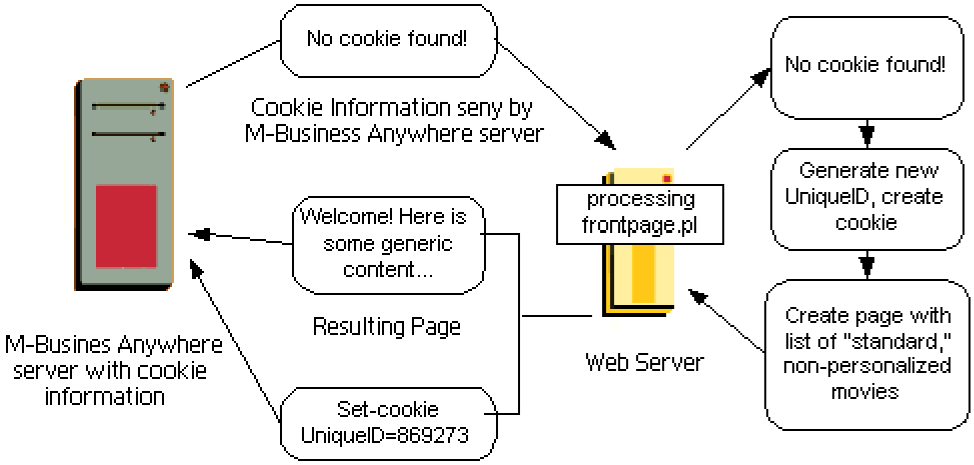
When the M-Business Sync Server accesses The Movie Review Channel's front page, you would look up the cookie for the user (retrieved from the M-Business Sync Server) to obtain his ID. You can use that ID to look up a list of his preferences and generate a custom list of movie reviews for him to read.


UniqueID=572967
Cookie Information sent by M-Business Anywhere server
Look up preferences for user 572967
Generate list of movies for user 572967
Create page with list of movies
processing frontpage.pl
Web Server
Welcome, Todd! Here is your list of movies we recommend...
Resulting Page
M-Busines Anywhere server with cookie information

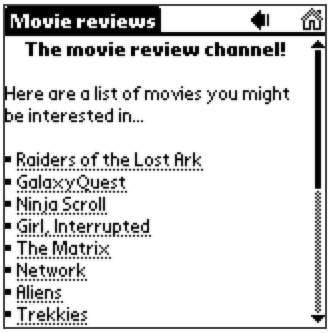
What if he wants to change his preferences? You can provide him with a form that allows him to check or uncheck his movie preferences. This form, once submitted, will alter his preferences stored on your database.



What if this is his first time visiting our channel? We can also determine that by a cookie; specifically, the lack of one. In that case, it is often a good idea to present your user with some default content, along with a prominent message telling him to customize his preferences.


No cookie found!
Cookie Information seny by M-Business Anywhere server
No cookie found!
Generate new UniqueID, create cookie
Welcome! Here is some generic content...
Resulting Page
processing frontpage.pl
Web Server
Create page with list of "standard," non-personalized movies
M-Busines Anywhere server with cookie information
Set-cookie UniqueID=869273

Here are a couple of pages of the Movie Review Channel in action. The figure below shows the default list of movies that appears when user first visits the site, before a cookie has been set up.

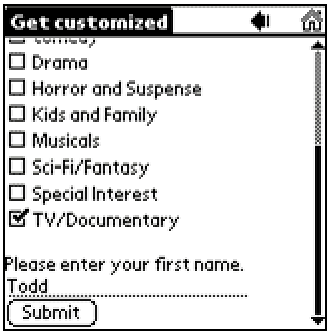


The movie review channel!

Here are a list of movies you might be interested in...

- Raiders of the Lost Ark
- GalaxyQuest
- Ninja Scroll
- Girl, Interrupted
- The Matrix
- Network
- Aliens
- Trekkies

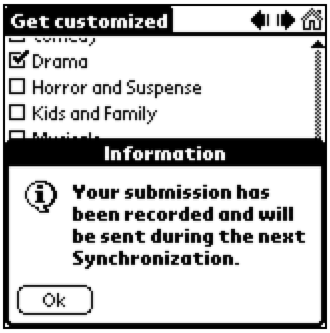
The figure below shows a form on which a user can select custom categories that the user is interested in.



□ Drama
□ Horror and Suspense
□ Kids and Family
□ Musicals
□ Sci-Fi/Fantasy
□ Special Interest
☑ TV/Documentary

Please enter your first name.
Todd

The following figure shows a standard form submission dialogue box, presented when user taps **Submit**. We will improve on this later, in “Managing channel form submissions” on page 173.

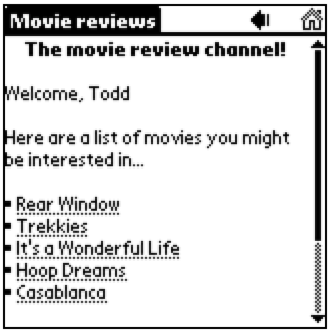


☑ Drama
□ Horror and Suspense
□ Kids and Family

Information

Your submission has been recorded and will be sent during the next Synchronization.

The figure below shows the customized Movie Channel home page, generated by using ID in cookie to look up user preferences in a database.



Movie reviews

The movie review channel!

Welcome, Todd

Here are a list of movies you might be interested in...

- Rear Window
- Trekkies
- It's a Wonderful Life
- Hoop Dreams
- Casablanca

Keeping page requests in sync with preferences

Now, hold on a sec. Assume I change my preferences and then synchronize this channel. At the moment I synchronize, I will be performing two activities at once. I am going to be submitting the form to change my preferences, but at the same time, I am also going to be grabbing the front page of my channel. How do I know the front page is going to be based on my new preferences instead of my old ones?

Good question. When a form is submitted through the M-Business Sync Server, we make sure that all the forms have finished submitting information and all the actions called by those forms have run to completion *before* we start requesting any pages from that channel.

OK, so you know the basic strategy for approaching personalized channels. The next section takes a closer look at implementing the channel.

Implementation details for this example

If you are comfortable using cookies and the strategy outlined above, you can safely skip the rest of this chapter and move on to the issue of managing the forms your channel users submit — see "Managing channel form submissions" on page 173. If you really want the nitty-gritty details of how to implement a personalized channel, read on.

But before we do, a couple of quick disclaimers. First, the Legal Notice of Liability:

The information in this document is distributed on an As Is basis, without warranty. While every precaution has been taken in the preparation of the tutorial, neither the author nor AvantGo shall have any liability to any person or entity with respect to loss or damage caused or alleged to be caused directly or indirectly by the instructions contained herein.

Note
The code examples you are about to see should be used as general guidelines only. While the channel works, it probably is not as efficient or as bullet-proof as something you could write yourself. So feel free to look at what I have done here and follow the same processes for your channel, but do not reuse my code. I am sure you can do better starting from scratch.

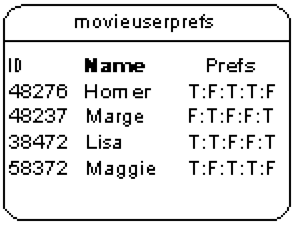
All code was created with Perl 5.005 with the CGI and `CGI::Cookie` modules (although I do not really use them to their full extent). I also made use of Perl's DBM capabilities, which gives me access to a very simple database. I am running these scripts on an Apache server under Windows 2000, although I am not using `mod_perl`, or any higher level tools like Mason, PHP, or ASP.

Perl can sometimes be a confusing language for those not familiar with it, but I tried to avoid some of the crazier idiomatic tricks. Hopefully, if you have some programming experience, you will be able to follow along.

Setting up the databases

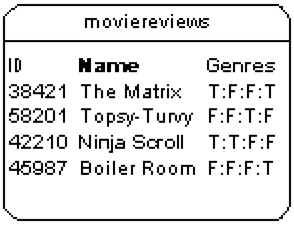
I have created two databases here (they are both simple DBM files). One is a table of users. It contains a Unique ID number, the user's first name, followed by a list of 11 true or false values depending on whether the user likes that corresponding genre or not. Note that simple database management programs only allow one field per record, so my first name and 11 preferences are actually crammed into one large text string.

The other DBM file is a table of movies. It contains a Unique ID number, the name of the movie, followed by a list of 11 true or false values depending on whether the movie belongs in that genre or not. I created the table this way so that a movie can belong in more than one genre (For instance GalaxyQuest can be considered both Sci-Fi and Comedy.)



movieuserprefs

ID	**Name**	Prefs
48276	Homer	T:F:T:T:F
48237	Marge	F:T:F:F:T
38472	Lisa	T:T:F:F:T
58372	Maggie	T:F:T:T:F



moviereviews

ID	**Name**	Genres
38421	The Matrix	T:F:F:T
58201	Topsy-Turvy	F:F:T:F
42210	Ninja Scroll	T:T:F:F
45987	Boiler Room	F:F:F:T

Note
In my current implementation described above, I have to be very careful in ensuring that all the lists of 11-true-or-false values are always in the correct order. Plus, it makes it difficult to add or remove genres later. There are better ways of implementing this through a relational database such as MySQL. But this will do as an example.

Listing 1: frontpage.pl

This section analyzes the key parts of the sample source code for generating the front page. For a full listing of this source code, see “frontpage.pl, basic version” on page 389.

First, we load some Perl modules, and print out a standard HTTP header (along with a `Cache-Control:private` header — see the next section about why we do this).

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie; print header(-'Cache-Control'=>'private');
```

Next, we retrieve any cookies associated with this page.

```
%cookies = fetch CGI::Cookie;
```

If we find a cookie marked `movierevieWID`, then we open our database, look for a user with that ID number, grab his first name into the variable `propername` and his movie preferences into an array of 11 true-or-false values called `prefs`. (Again, remember that all my values are stored in the database as one large string with different fields separated by `~:~` characters. If I were using a more sophisticated database, I could go with something more elegant.)

```
if ($cookies{'moviereviewID'} ) {

# If a cookie exists, we get its value
# (which is the user's ID number)

$myID = $cookies{'moviereviewID'} -> value;

dbmopen(%prefs,"movieuserprefs",0644)
    || die "Can't open movieuserprefs! $!";

# We look up the user's preferences based on
# his ID within the database.
#
# The record itself is a big string of values
# separated by ~:~ characters. The "split"
# function will break up that string and place
# the resulting values into the variable
# "propername" and the array "prefs"

($propername, @prefs) = split(/~:~/, $prefs{$myID});

dbmclose(%prefs);

} else {
```

If such a cookie does not exist, we load up some default values and note that we are just loading default values.

```
# We make prefs an array of default values if
    # the user's cookie does not exist.

    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);

    # This is a flag that indicates we're using
    # default values.

    $default = true;
}
```

We print out some standard text for the beginning of the web page.

```
print <<END_of_Start;
<HTML> <HEAD>
<TITLE>Movie reviews</title>
<META NAME="HandheldFriendly" content="True">
</HEAD>
<BODY>
<Center><h2>The movie review channel!</h2></center>

END_of_Start
```

```
print "<p>Welcome, $propername </p>" unless $default;
print "<p>Here are a list of movies you might be
interested in...</p>\n <ul>\n";
```

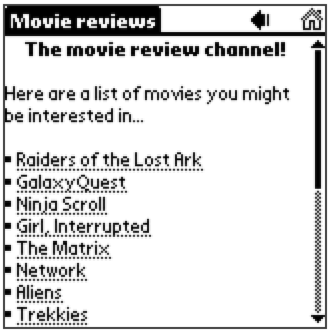
This is where the work really happens. We go through every movie and check the genres it fits into. If we find it belongs to one of the genres that our user has selected on his list, then we print a link to the movie with the URL of `reviews/<movieIDnumber>.html`.

```
dbmopen(%movies,"moviereviews",0644)
    || die "Can't open moviereviews! $!";
# We iterate over each movie ID within the moviereviews
# database. foreach $filmnumber (keys (%movies)) {
# Once again, breaking up a string into the variable
# "filmname" and the "genres" array by using a split
# command.

($filmname, @genres) = split(/~:~/, $movies{$filmnumber});
$match = 0;
for ($i =0; $i < 11; $i++)
    { if ($genres[$i] && $prefs[$i])
    { $match = 1; # We've found a match! last;
                  # Break out of the loop.
                  # no need to keep looking.
    }
    }

# If this film has a match, include a link to it.
If ($match) {
    print "<li><a href=\"reviews/
           ${filmnumber}.html\">
           $filmname</a>\n";
    }
} dbmclose(%movies);

print "</UL>\n<p>Tune in next time for more movie reviews!<p>\n";
```

We include a link to the personalization page, and some encouragement for users who have not set up their cookies yet.

```
If ($default) { print "<center>Want personalized content? Of
        course you do!\n";
    print "<br><a href=\"/cgi-bin/movies/editprefsform.pl\">
        Edit your preferences</a></center>\n"; }
else {
    print "<a href=\"/cgi-bin/movies/editprefsform.pl\">
        Edit preferences</a>\n"; } print "</body>\n</html>\n";
```

This will present a front page that looks a little something like this:



Movie reviews

The movie review channel!

Here are a list of movies you might be interested in...

- Raiders of the Lost Ark
- GalaxyQuest
- Ninja Scroll
- Girl, Interrupted
- The Matrix
- Network
- Aliens
- Trekkies

Listing 2: editprefsform.pl

For a full listing of this source code, see "editprefsform.pl, basic version" on page 391.

Suppose our user wants to set or alter his preferences. All we really need to do is offer a form consisting of several checkboxes (one for each genre) plus a text field for his name. This could be done with a standard HTML page. But we want to make sure the initial state of the form represents the user's current choices. (In other words, if our user has already asked to receive comedy and drama reviews, the comedy and drama fields should already be checked.) So once again, we are creating a script to produce this form.

We start with the standard headers.

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;

%cookies = fetch CGI::Cookie;
```

If a cookie for this person already exists, we use that information to grab his name and preferences from our database as before.

```
If ($cookies{'moviereviewID'} ) {
    $myID = $cookies{'moviereviewID'} -> value;
    dbmopen(%prefs,"movieuserprefs",0644)
        || die "Can't open movieprefs! $!";
    ($propername, @prefs) = split(/~:~/,
    $prefs{$myID});
} else {
```

If no cookie is available, we could give him an entirely blank form, or present him with a form where the default categories are already checked. This is a matter of taste, really. I chose to go with the default values already checked.

```
@prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0); $default = 1; }
```

Print out some HTML:

```
print header(-'Cache-Control'=>'private');

print <<END_of_FormTop; <HTML> <HEAD>
    <TITLE>Get customized</TITLE>
    <META NAME="HandheldFriendly" content="True">
    </HEAD>
```

```
<BODY> <p>Please indicate what movies you
    would like to receive reviews for:</p>
    <form name="choosemovies"
    action="/cgi-bin/movies/setprefs.pl"> <p>

END_of_FormTop
```

Now, we print out a checkbox for each genre category. If we find that, according to our user's preferences, he is already subscribed to that category, we mark it checked by default.

```
@genrenames = ( "Action/Adventure", "Art House/Indie",
    "Classics", "Comedy", "Drama", "Horror and Suspense",
    "Kids and Family", "Musicals", "Sci-fi/Fantasy",
    "Special Interest", "TV/Documentary" );

for ($i = 0; $i<11; $i++) {
    print "<input type=\"checkbox\" name=\"$i\" value=\"1\"";
    if ($prefs[$i]) {
        print " checked";
        }
    print ">\n$genrenames[$i]<Br>\n"; }
```

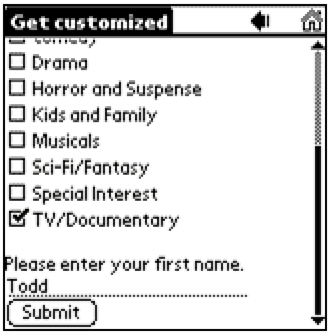
Do something similar with the name field:

```
print "<p>\nPlease enter your first name.\n";
print "<input type=\"text\" name=\"firstname\" size=\"15\"
    maxlength=\"15\" value=\"$propername\">\n </p>\n";
```

And we are done, except for a little clean up work. Note the **Submit** button right now looks like every other submit button you have seen before. We are actually going to change this in the next section, so keep that in mind.

```
Print <<END_of_Page;

<input type="submit" name="submit" value="Submit">

</form> </BODY> </HTML>

END_of_Page
```

This will give us a checklist that looks a little something like this:



☐ Drama
☐ Horror and Suspense
☐ Kids and Family
☐ Musicals
☐ Sci-Fi/Fantasy
☐ Special Interest
☑ TV/Documentary

Please enter your first name.
Todd

Listing 3: setprefs.pl

For a full listing of this source code, see "setprefs.pl, basic version" on page 393.

All we need now is a script to process the results of the form we generated above. In other words, we need to set a cookie for the user, if one is not set already, and edit his preferences in our database.

We start off again with the usual headers (plus an `srand` call to start up our random number generator).

```
#!D:\Perl\bin\perl.exe use CGI qw/:standard/; use CGI::Cookie; srand;
```

Then we grab the parameters returned by the form. Note that all of our genre preference parameters within the previous form are named 1, 2, 3, and so on. So it is relatively easy to grab their values from a `for` loop.

We put the user's name into a variable called username and his preferences into an array of 11 true-or-false values. That last line there replaces any `~:~` strings with `~;~` in case somebody happened to include our record-separating string within his name. (Although that would be a really strange name!)

```
for ($i=0; $i<11; $i++) { $prefs[$i] = param($i) || "0"; }
    $username = param("firstname"); $username =~ s/~:~/~;~/g;
```

If the user's cookie indicates he has an ID already, we use it. Otherwise, we generate one using the getrandomID function — this is a function created below that generates a random ID for us.

```
%cookies = fetch CGI::Cookie;
if ($cookies{'moviereviewID'} )
{
    $myID = $cookies{'moviereviewID'}->value;
}
else {
    $myID = getrandomID();
}
```

Then we use Perl's CGI::cookie function to create a cookie. I have set the expire time to 10 years from now, since we really do not want our cookies to expire between synchronizations. (If I wanted to be thorough, I could also define my own domain and path values for the cookie.)

```
$tkcookie = new CGI::Cookie(-name=>'moviereviewID',
    -value=>$myID, -expires=>'+10y');
```

We write the user's preferences into the database, joining them together as a `~:~` -separated list of values.

```
dbmopen(%prefs, "movieuserprefs", 0644) ||
    die "cannot open DBM $!"; $prefstring = $username. "~:~".
    join ("~:~", @prefs); $prefs{$myID} = $prefstring; dbmclose(%prefs);
```

Finally, we print out our HTML document. Note the set-cookie line among the HTTP headers. This document will be placed in the Forms Manager during the next synchronization. It is really just a bunch of debug information, and I do not really want our users to see it. (After the next synchronization, our users will be looking at the new `frontpage.pl`.) In the next section, we will find out how to hide this from the Forms Manager.

```
print "Content-type: text/html\n";
print "Set-Cookie: $tkcookie\n";
print "Cache-Control: private\n";
print <<END_of_Response;

<HTML> <HEAD> <TITLE>Cookie has been set</TITLE> </HEAD>
<BODY> Your cookie has been set. <p>

END_of_Response

print "This is debug information you'll never see. \n";
print "Your prefs string is $prefstring <p> \n";
print "And your ID is $myID \n </body></HTML>";
```

Last but not least is our `getrandomID()` function, which takes the current time (in seconds) and appends a 7-digit random number to it.

```
sub getrandomID { return (time(). sprintf("%07d",(int(rand(5000000))))); }
```

Caution
Do not use this approach with `getrandom()` if you are using a version of Perl before 5.004. Perl versions before 5.004 used `time()` to seed the random number generator.

Expiring cookies differently for desktop vs. channel viewers

It is fairly common for web sites on the open Internet to expire cookies in a fairly short time, like 15 or 20 minutes. This may be done for reasons of security. If there has been no user interaction for 15-20 minutes, the user may have walked away from his or her computer with a browser window left open, logged into your site. To prevent another person from walking up to the computer and having full access to the user's account, you make cookies fairly quickly.

If your channel is drawing its pages from the same source as a web site on the open Internet that expires cookies quickly, those rapidly expiring cookies could cause problems for your offline channel viewers. You would want to give your channel viewers considerably longer between interactions before you expire their cookies.

In order to set cookies to expire differently for desktop vs. channel viewers, you simply send different `Set-cookie` headers to your M-Business Sync Server vs. desktop browsers. You can identify requests from your M-Business Sync Server by examining the `User-Agent` header in the HTTP request. The `User-Agent` header sent by an M-Business Sync Server will look something like:

```
User-Agent: Mozilla/3.0 (compatible; AvantGo 3.2)
```

or

```
User-Agent: Mozilla/3.0 (compatible; AvantGo 5.2; FreeBSD)
```

When the `User-Agent` header looks like this, you set the cookie to expire with a longer time, tailored to channel viewers. Otherwise you keep the cookie expiring with the shorter time, which you have set up for desktop viewers.

Putting it all together

When I create my channel, I will use `frontpage.pl` as my front page. Or, more likely, I will create a server-side-include that makes *index.html* point to *customfrontpage.pl*. This will generate a custom front page, as well as a list of links to static movie reviews. It will generate a custom form for people who wish to alter their personalized content, and all these pages are downloaded to the user's mobile device.

Storing information in cookie vs. database

Hey, why did you store all of the user's movie-preference information in a local database instead of within the cookie itself?

No overwhelming reason. I just felt like, in the long run, if I were to make dramatic changes to the inner workings of my channel, it would be a whole lot easier to make sweeping changes to a database than to deal with a lot of obsolete cookies. The drawback is that I am making my server do more work in maintaining a local database. If I am not careful, my server might be too busy doing database work to serve pages efficiently.

Part IV. Building Full-featured Applications

CHAPTER 10

Tools to add special features to a channel

Contents

Overview of special features

This chapter describes some special features that are available for you to use in your channel. Instructions provided show how to use the M-Business Date/Time Picker and List Viewer to streamline user inputs to forms in your channel pages. Information is also provided for interfacing with a Symbols Technologies scanner, and for capturing signatures on a mobile device.

M-Business version-specific on-device code files

If you are using M-Business Client version 5.7 or earlier, you should use the PODS versions that came with that version of the client. This applies to the files listed below.

Table 1. M-Business version-specific on-device code files

Functionality	File name on Microsoft OSes	File name on Palm OS
On-device XML data	*dbpod_quad.dll*	*dbpod_quad.prc*
Date/Time Picker	*mimedtpicker.dll*	*mimedtpicker.prc*
List Viewer	*mimelist.dll*	*mimelist.prc*

The versions of these files that ship with M-Business Server version 6.5 do not work with M-Business Client version 6.2 and earlier. Beginning with M-Business Server version 6.5, the version numbers of the shipping PODS are synchronized with the M-Business Server version number. If you use any of these PODS in your applications, note the following compatibility requirements and make sure that none of your users with M-Business Client version 6.2 and earlier inadvertently get PODS from M-Business Server version 6.5, unless you first upgrade them to M-Business Client version 6.5.

Table 2. M-Business Server version and shipping PODS version

M-Business Client version	Required M-Business PODS version
6.2 or earlier	2.0.x
6.5	2.1.x

M-Business Server version 6.5 bundles the 2.1.x version of the PODS. If your M-Business solution is deployed exclusively on the 6.5 version of M-Business Client, simply proceed as directed in "Upgrading and installing M-Business Server" [*M-Business Anywhere Administrator Guide*].

If your solution is deployed on both 6.5 and earlier versions of M-Business Client, use the `AG_CLIENTVERSION` macro to vend the correct PODS version to each M-Business Client version. If you support some devices running Palm OS and other devices running Microsoft OSes, use the `AG_DEVICEOS` macro in your logic to vend the correct PODS version for the platform. See "Using server URL macros to customize content" on page 166 for details regarding URL macros.

If your solution uses custom PODS, there is a compatibility issue only if the PODS make use of the `getCharsetEncoding()` or `rebuildSelectionList()` methods. PODS that utilize either of these methods must be recompiled with new header files in order to work on M-Business Client version 6.5. The new header files are supplied in the *pods.zip* file, which you can download from http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere — look for the "PODS API resources..." link.

Table 3. Compatibility of custom PODS and M-Business Client versions

M-Business Client version	Custom POD without affected methods	Custom POD with affected methods
6.2 or earlier	Use existing POD	Use existing POD
6.5	Use existing POD	Recompile POD with new header files

Using the Date/Time Picker

The M-Business Date/Time Picker is a MIME player that allows the user to select a date and/or a time. A Date/Time Picker instance is initially displayed as a read-only text field. The user taps on the text field to expand the Date/Time Picker in-line; the user taps on the text field again to return the Date/Time Picker to its original state.

Note
The Date/Time Picker currently is only available for use with the M-Business XML POD, which receives data from the M-Business XML conduit. If you use the UltraLite on-device database, you will have to provide comparable functionality.

Caution
If you use the Date/Time Picker, you must make sure that the supporting code file on the device is the correct one for the version of M-Business Client on the device. For more information, see "M-Business version-specific on-device code files" on page 212.

Note
The Date/Time Picker is not currently supported on Symbian OS.

The rest of this section covers setting up the object tag to embed the Date/Time Picker in an HTML page and testing your Date/Time Picker application on an emulator or on a physical device. For additional information on using the List Viewer, see the following sources:

- For an overview of the online sample code that illustrates the use of the Date/Time Picker in context, see "Date/Time Picker" on page 385.
- For instructions on embedding an instance of the Date/Time Picker in HTML, see "Specifying Date/Time Picker object tag parameters" on page 214.
- For reference material you will need to write JavaScript or C code to use the Date/Time Picker, see "M-Business Date/Time Picker API" [*M-Business Anywhere API Reference*]

Specifying Date/Time Picker object tag parameters

To embed an instance of the Date/Time Picker in HTML, use the `object` tag with `type` set to `application/avantgo-mime-datetimepicker`:

```
<object type="application/avantgo-mime-datetimepicker"
    width=pixelsWide height=pixelsHigh> </object>
```

Where:

- *`pixelsWide`* is the width of the Date/Time Picker area in pixels.

- *pixelsHigh* is the height of the Date/Time Picker area in pixels.

For a list of parameters you can specify within the object tag, see the table below. For reference material you will need to write JavaScript or C code to use the Date/Time Picker, see "M-Business Date/Time Picker API" [*M-Business Anywhere API Reference*].

Table 4. Date/Time Picker object tag parameters

Specification	Description
abbreviatedMonthLabels	Shorter, abbreviated labels to be displayed for the months of the year. Default value: 'Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec'
abbreviatedWeekdayLabels	Shorter, abbreviated labels to be displayed for the days of the week. Default value: 'Sun, Mon, Tue, Wed, Thu, Fri, Sat'
datePickerArrowWidth	The width, in pixels, of the arrow. **Note:** This is different from the length of the arrow.
datePickerAvgoRange	Whether the server-side date range should be enforced: true or false. When the default value, true, is selected, the user can only select a date between the years 1901 and 2037. **Note:** This option does not apply to the PalmOS version. PalmOS always has a date range set between the years 1904 and 2031.
datePickerBdColor	Border color for the Date Picker. Default value: Navy (000080).
datePickerCellBgColor	Background color for the Date Picker. Default value: gray (#cccccc).
datePickerCellFgColor	Foreground color for the Date Picker. Default value: black.
datePickerCellSelectBgColor	Background color for the selected date in the Date Picker. Default value: black.
datePickerCellSelectFgColor	Foreground color for the selected date in the Date Picker. Default value: white.
datePickerFontFamily	Font family for the Date Picker. Default value: sans-serif.

Specification	**Description**
`datePickerFontSize`	Font size for the Date Picker. Default value: `9`.
`datePickerHeaderBgColor`	Background color for the weekday names area in the Date Picker. Default value: `black`.
`datePickerHeaderFgColor`	Foreground color for the weekday names area in the Date Picker. Default value: `white`.
`datePickerMonthBgColor`	Background color for the month selector in the Date Picker. Default value: `gray (#cccccc)`.
`datePickerMonthFgColor`	Foreground color for the month selector in the Date Picker. Default value: `navy (#000080)`.
`datePickerStartDayOfWeek`	Start day of the week. Default value: `0` (Sunday). Possible values are: `0` (Sunday) `1` (Monday) `2` (Tuesday) `3` (Wednesday) `4` (Thursday) `5` (Friday) `6` (Saturday)
`datePickerVisibility`	Visibility of the Date Picker. Default value: `true`.
`datePickerYearBgColor`	Background color for the year selector in the Date Picker. Default value: `gray (#cccccc)`.
`datePickerYearFgColor`	Foreground color for the year selector in the Date Picker. Default value: `navy (#000080)`.
`dateTimeValue`	Specified date/time value: an integer value representing the number of seconds since Jan. 1, 1970 00:00:00.
`format`	Format specification for the read-only text field. See "Field format specifications" on page 226, for a list of specification values and their associated descriptions.

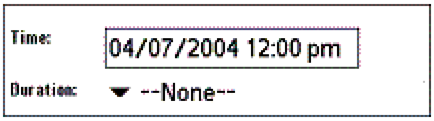
Specification	**Description**
`fullMonthLabels`	The full labels to be displayed for the months of the year. Default value: `'January, February, March, April, May, June, July, August, September, October, November, December'`
`fullWeekdayLabels`	The full labels to be displayed for the days of the week. Default value: `'Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday'`
`setCurrent`	Initialize the Date/Time Picker with the current date/time: true or false. Default value: `false`.
`setCurrentTime`	Initialize the Date/Time Picker with the current time only: true or false. Default value: `false`.
`shortWeekdayLabels`	The very shortest labels to be displayed for the days of the week. Default value: `'S,M,T,W,T,F,S'`
`timePicker24hrFormat`	Use 24-hour format to display the Time Picker (instead of am/pm), `true` or `false`. Default value: `false`. See "Field format specifications" on page 226, for a list of specification values and their associated descriptions.
`timePickerAMDesignator`	Designator string for a.m. times.
`timePickerBdColor`	Border color for the Time Picker. Default value: `navy (#000080)`.
`timePickerBgColor`	Background color for the Time Picker. Default value: `white`.
`timePickerFgColor`	Foreground color for the Time Picker. Default value: `black`.
`timePickerFontFamily`	Font family for the Time Picker. Default value: `sans-serif`.
`timePickerFontSize`	Font size for the Time Picker. Default value: `9`.
`timePickerPMDesignator`	Designator string for p.m. times.
`timePickerSelectBgColor`	Background color for the selected component in the Time Picker. Default value: `black`.

Specification	Description
`timePickerSelectFgColor`	Foreground color for the selected component in the Time Picker. Default value: `white`.
`timePickerVisibility`	Visibility of the Time Picker, `true` or `false`. Default value: `true`.

Using the Palm OS emulator with the Date/Time Picker

The following figures illustrate how to use the Date/Time Picker on a Palm OS emulator.

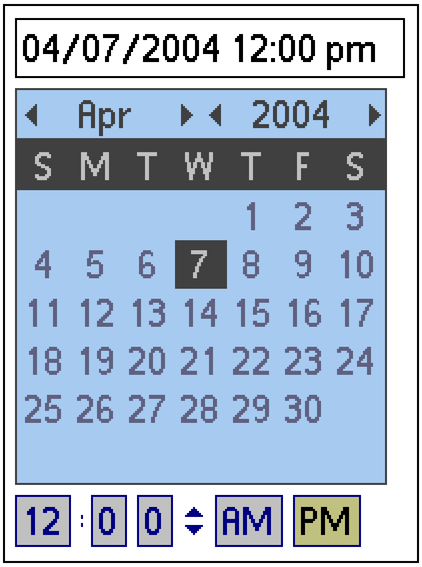
- Date/Time Picker initially displays a blank text field.



Time: 04/07/2004 12:00 pm

Duration: --None--

- The Date Picker initially displays the current date, and the Time Picker displays the default time value of 12:00 p.m.



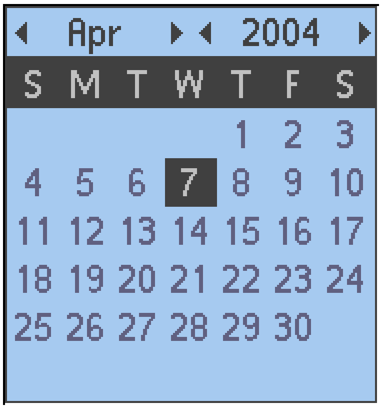
04/07/2004 12:00 pm

Apr 2004

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

12 : 0 0 AM PM

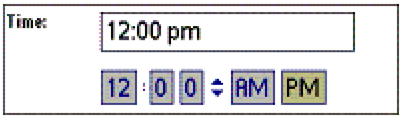
- The Date Picker alone looks like this when the HTML writer hides the Time Picker from view:



Apr 2004

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

- The Time Picker alone looks like this when the HTML writer hides the Date Picker from view:

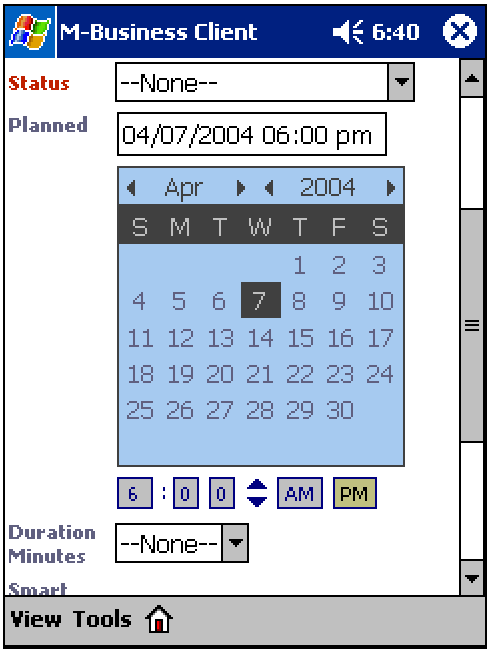


Time: 12:00 pm

Using a Microsoft OS device with the Date/Time Picker

The following figures illustrate how to use the Date/Time Picker on a Microsoft OS device.

- Date/Time Picker initially displays a blank text field.



Status --None--

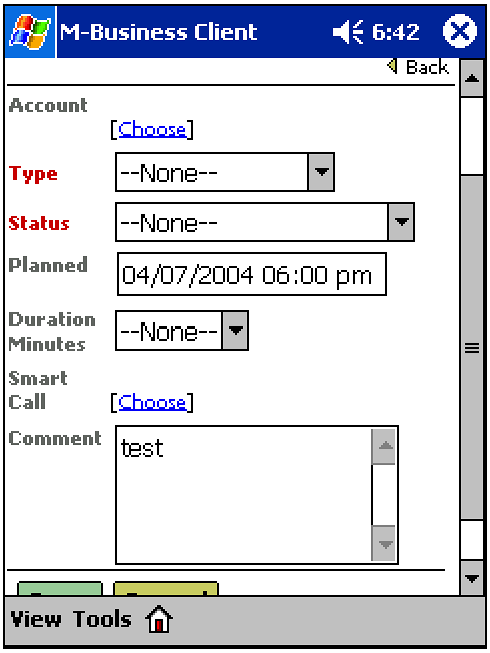
Planned 04/07/2004 06:00 pm

Apr 2004

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Duration Minutes --None--

- The Date Picker initially displays the current date, and the Time Picker displays the default time value of 00:00 a.m.



Account [Choose]

Type --None--

Status --None--

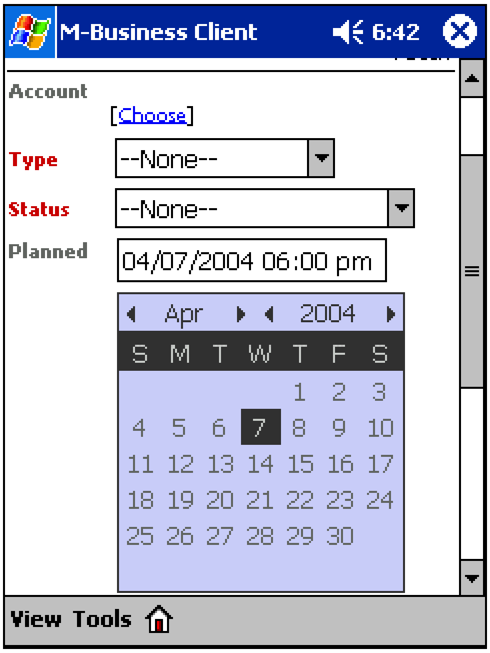
Planned 04/07/2004 06:00 pm

Duration Minutes --None--

Smart Call [Choose]

Comment test

- The Date Picker alone looks like this when the HTML writer hides the Time Picker from view:



Account [Choose]

Type --None--

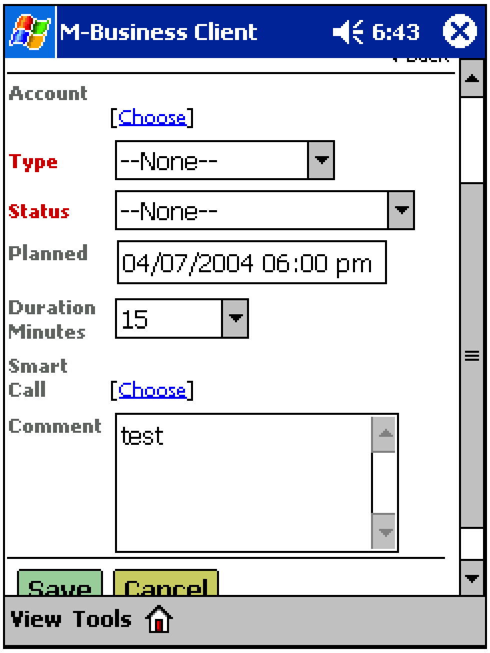
Status --None--

Planned 04/07/2004 06:00 pm

Apr 2004

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

- The Time Picker alone looks like this when the HTML writer hides the Date Picker from view:



Account [Choose]

Type --None--

Status --None--

Planned 04/07/2004 06:00 pm

Duration Minutes 15

Smart Call [Choose]

Comment test

Using the List Viewer

The M-Business List Viewer is a MIME player that allows the user to sort and filter the displayed contents of an M-Business XML datastore `agdbset` or of an `ExtendedDBSet`.

Note
The List Viewer currently is only available for use with the M-Business XML POD, which receives data from the M-Business XML conduit. If you use the UltraLite on-device database, you will have to provide comparable functionality.

Caution
If you use the List Viewer, you must make sure that the supporting code file on the device is the correct one for the version of M-Business Client on the device. For more information, see "M-Business version-specific on-device code files" on page 212.

Note
The List Viewer is not currently supported on Symbian OS.

The rest of this section covers setting up the object tag to embed the List Viewer in an HTML page and testing your List Viewer application on an emulator or on a physical device. For additional information on using the List Viewer, see the following sources:

- For an overview of the online sample code using the List Viewer, see "Using the List Viewer with M-Business XML datastore" on page 118.
- For instructions on embedding an instance of the Date/Time Picker in HTML, see "Specifying List Viewer object tag parameters" on page 222.
- For reference material you will need to write JavaScript or C code to use the Date/Time Picker, see "M-Business List Viewer API" [*M-Business Anywhere API Reference*]

Specifying List Viewer object tag parameters

To embed an instance of the List Viewer in HTML, use the `object` tag with the `type` set to `application/avantgo-mime-list` :

```
<object id=objName type="application/avantgo-mime-list"
    width=pixelsWide height=pixelsHigh
    style=inlineStyleSpec>
```

Where:

- *`objName`* is the label you will use to refer to the list object.
- *`pixelsWide`* is the width of the list area in pixels.

- *pixelsHigh* is the height of the list area in pixels.
- *inlineStyleSpec* is an inline style specification to control the formatting of the list display.

The "Master-Detail page example" on page 419 illustrates the use of the List Viewer.

For a list of parameters you can specify within the object tag, see the table below. For reference material you will need to write JavaScript or C code to use the Date/Time Picker, see "M-Business Date/Time Picker API" [*M-Business Anywhere API Reference*].

For a list of parameters you can specify within the object tag, see the table below. After the object tag for the List Viewer is specified, you can use the "M-Business List Viewer API" [*M-Business Anywhere API Reference*] on the object.

Specific format specifications for the read-only text field are listed in "Field format specifications" on page 226.

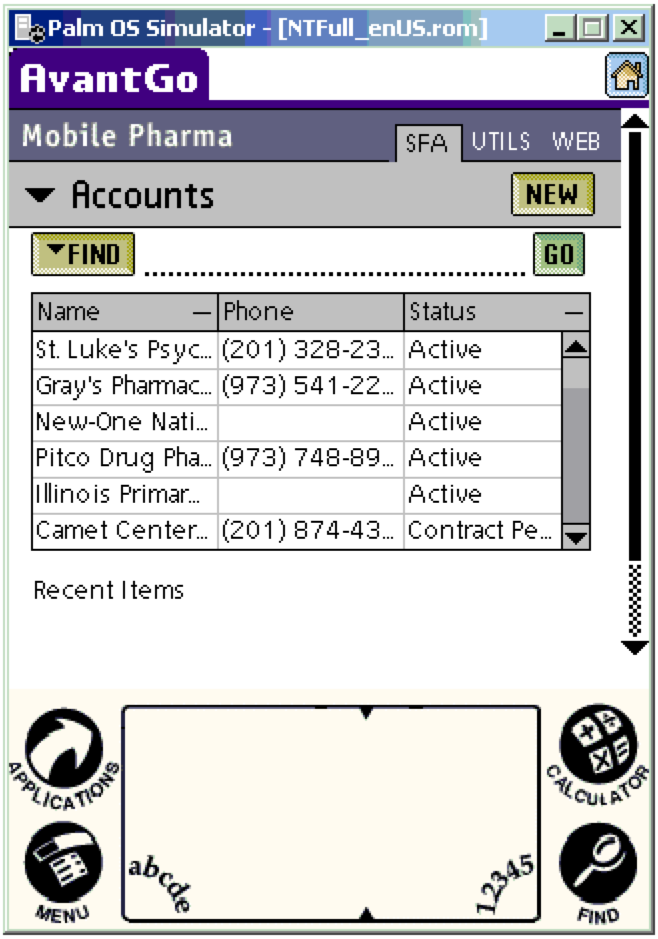
Table 5. List Viewer object tag parameters

Specification	Description
cellBgColor	The background color for the selected cell.
cellSelectBgColor	The background color for the selected cell.
cellBdColor	The border color for the cell.
cellSelectBdColor	The border color for the selected cell.
colAlignment_n	Horizontal placement of the contents for cells in column n. This attribute can also be set by calling setColumnAlignment() from JavaScript. Possible values for alignment are: 0 (left alignment) 1 (right alignment) 2 (center alignment)
colDSName_n	Data source column name for column n. This attribute can also be set by calling setColumnDSName() from JavaScript.

Specification	**Description**
`colFormat_n`	Format specifications for column n. This attribute works for columns of integer and datetime types only. This attribute can also be set by calling `setColumnFormat()` from JavaScript. See "Field format specifications" on page 226, for a list of specification values and their associated descriptions.
`colMargin`	The margin between each column in the list.
`colTitle_n`	Title for column n. This attribute can also be set by calling `setColumnTitle()` from JavaScript.
`colWidth_n`	Suggested width for column n. This attribute can also be set by calling `setColumnWidth()` from JavaScript.
`dbname`	Name of the M-Business XML datastore.
`headerBgColor`	The background color for the header.
`headerBdColor`	The border color for the header.
`numCols`	Number of columns in the list. This attribute can also be set by calling `setColumnCount()` from JavaScript.
`numRows`	Suggested number of rows in the list.
`realTimeScroll`	Scroll list data in real time? `true`/`false`; default is `true`.
`rowMargin`	The margin between each row in the list.
`scrollbarArrowColor`	The color for the scroll bar arrows.
`scrollbarBubbleColor`	The color for the scroll bar bubble.
`scrollbarBdColor`	The border color for the scroll bar.
`scrollbarBgColor`	The background color for the scroll bar.
`scrollbarButtonColor`	The background color for the scroll bar up/down buttons.

Using the Palm OS emulator with the List Viewer

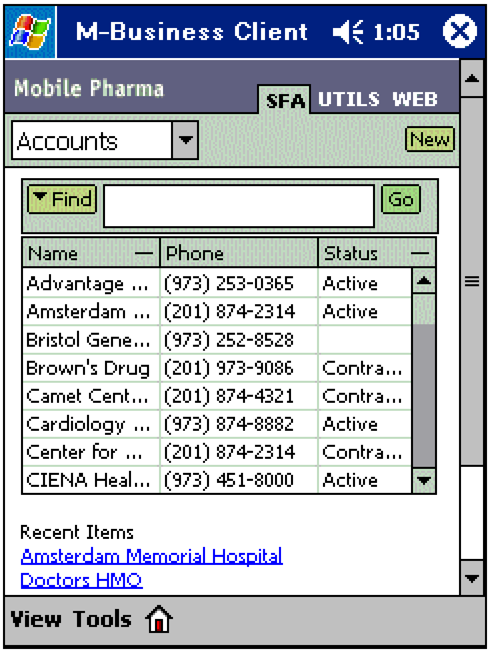
The following figure shows how the List Viewer displays database contents on a Palm OS Emulator. The list of contents results in a scrollable region on the Palm OS device.



Name	Phone	Status
St. Luke's Psyc...	(201) 328-23...	Active
Gray's Pharmac...	(973) 541-22...	Active
New-One Nati...		Active
Pitco Drug Pha...	(973) 748-89...	Active
Illinois Primar...		Active
Camet Center...	(201) 874-43...	Contract Pe...

Using a Microsoft OS device with the List Viewer

The following figures show how the List Viewer displays database contents on a Microsoft OS device. The two figures represent the full scrollable region as it would appear on a Microsoft OS device.



Name	Phone	Status
Advantage ...	(973) 253-0365	Active
Amsterdam ...	(201) 874-2314	Active
Bristol Gene...	(973) 252-8528	
Brown's Drug	(201) 973-9086	Contra...
Camet Cent...	(201) 874-4321	Contra...
Cardiology ...	(973) 874-8882	Active
Center for ...	(201) 874-2314	Contra...
CIENA Heal...	(973) 451-8000	Active

Recent Items
Amsterdam Memorial Hospital
Doctors HMO

Using a Microsoft Smartphone device with the List Viewer

On Microsoft Smartphone devices, you can set up the List Viewer so that users can navigate listed data using keys on the device. By setting an `onKeyPress` handler for the list, you can make the device keys function as follows:

- Arrow keys navigate into and out of the list.
- Up and Down Arrow keys and the numbers 2 and 8 on the keypad scroll up and down the list.
- Numbers 3 and 9 on the keypad page up and down the list.
- Pressing 5, or the center button of the Dpad, selects the highlighted row and opens the "Detail page" on page 432.
- Pressing 0 sorts the mimelist.
- Pressing 0 repeatedly cycles through ascending and descending sorts on different columns.

For sample code that implements these features with the List Viewer on Microsoft Smartphone, see "Master page for Microsoft Smartphone" on page 424.

Field format specifications

The following table lists the field formatting specifications for the List Viewer's read-only text fields. This information applies to both the Date/Time Picker and the List Viewer.

Specification	Description
%A	Replaced by the national representation of the full weekday name.
%a	Replaced by the national representation of the abbreviated weekday name, where the abbreviation is the first three characters.
%B	Replaced by the national representation of the full month name.
%b	Replaced by the national representation of the abbreviated month name, where the abbreviation is the first three characters.
%C	Replaced by (year / 100) as decimal number; single digits are preceded by a zero.
%c	Replaced by the national representation of time and date (this format is similar with that produced by asctime (3)).
%D	Equivalent to ''%m/%d/%y ''
%d	Replaced by the day of the month as a decimal number (01 - 31).
%E* %O*	POSIX locale extensions. The sequences %Ec %EC %Ex %EX %Ey %EY %Od %Oe %OH %OI %Om %OM %OS %Ou %OU %OV %Ow %OW %Oy are supposed to provide alternate representations. Additionally, %Ef is implemented to represent the short month name / day order of the date, %EF represents the long month name / day order, and %OB represents alternative month names (used standalone, without the day mentioned).
%e	Replaced by the day of month as a decimal number (1-31); single digits are preceded by a blank.
%G	Replaced by the year as a decimal number with century. This year is the one that contains the greater part of the week (with Monday as the first day of the week).
%g	Replaced by the same year as in ''%G '' but as a decimal number without century (00-99).
%H	Replaced by the hour (24-hour clock) as a decimal number (00-23).
%h	Same as date/time format as used in %b.
%I	Replaced by the hour (12-hour clock) as a decimal number (01-12).
%k	Replaced by the hour (24-hour clock) as a decimal number (0-23); single digits are preceded by a blank.
%L	Replaced by the minute as a decimal number (0-59, no leading 0).
%l	Replaced by the hour (12-hour clock) as a decimal number (1-12); single digits are preceded by a blank.
%M	Replaced by the minute as a decimal number (00-59).
%m	Replaced by the month as a decimal number (01-12).

Specification	Description
%N	Replaced by the month as a decimal number (1-12).
%n	Replaced by a newline.
%O*	Same as %E *.
%P	Replaced by the national representation of first character of either "ante meridian" or * "post meridian" as appropriate.
%p	Replaced by the national representation of either "ante meridian" or * "post meridian" as appropriate.
%R	Equivalent to ’’%H:%M ’’.
%r	Equivalent to ’’%I:%M:%S %p ’’.
%S	Replaced by the second as a decimal number (00-60).
%T	Equivalent to ’’%H:%M:%S ’’.
%t	Replaced by a tab.
%v	Equivalent to ’’%e-%b-%Y ’’.
%X	Replaced by the national time representation.
%x	Replaced by the national date representation.
%Y	Replaced by the year with the century as a decimal number.
%y	Replaced by the year without the century as a decimal number (00-99).
%z	Replaced by the year as a decimal number (1-10).
%+	Replaced by the national representation of the date and time (the format is similar to that produced by date(1)).
%%	Replaced by "%".

List Viewer sort sequence

The List Viewer supports programmatic sorting of on-device data and allows users to sort on any column by tapping or clicking the column header. For sample code, see “Master-Detail page” on page 386. Regarding the "collating sequence" in which text is sorted, you should be aware of the following:

- **Default order** List Viewer is not automatically sorted in the initial display. If you want the data to be sorted on a particular column, you must do that programmatically. Users may sort on a particular column at any time be tapping or clicking the column header.

- **Default sort order** Unless it is changed programmatically, or the user initiates sorts, the default sort order is descending.
- **Case sensitivity** List Viewer sorting is based strictly on the sequence of character codes in the character set that you specify. This means that it is completely case sensitive. In ascending order. "JONES" comes before "Jones", which comes before "jones".
- **Data in non-Latin alphabets** The List Viewer has been tested with Windows 1252 and ISO 8859-1 character sets. Any alphabetic-based, left-to-right language represented in UTF-8 should sort as expected.
- **Right-to-left alphabets** Data represented in character sets for languages that read right-to-left are *not* sorted correctly.

List Viewer examples

The following sample code contains examples of how to use the APIs documented in this section.

```
<html>
<head>
<meta name='handheldfriendly' content='true'>
<title>account</title>

<script>

var dbmgr;
var accountDb;
var extendedAccountDb;
var accountSortOrderArray = new Array(6);
var accountCols = new Array(6);

function initPage()
{
    var list;

    // Open database
    if (undefined == dbmgr) {
        dbmgr = CreateObject('avantgo.db');
    }

    if (null == dbmgr) {
        alert('error: unable to create avantgo.db');
        return;
    }

    accountDb = dbmgr.open("AvGoAccount", "r");
    if (null == accountDb) {
        alert("failed to open db");
        return;
    }

    // Create extended dbset for accountDb

    extendedAccountDb = CreateObject("ExtendedDBSet");
    if (null == extendedAccountDb) {
        alert('error: unable to create ExtendedDBSet');
        return;
    }
    extendedAccountDb.setDBSet(accountDb);
```

```
        // Set attributes, and create event handler for accountTable

        list = document.getElementById("accountTable");
        list.setDbSet(extendedAccountDb);

        list.onclick = onClickHandler;

        // Initialize vars

        accountCols[0] = "name";
        accountCols[1] = "revenue";
        accountCols[2] = "boolean";
        accountCols[3] = "created";
        accountCols[4] = "int16";
        accountCols[5] = "uint16";

        accountSortOrderArray[0] = 0;
        accountSortOrderArray[1] = 0;
        accountSortOrderArray[2] = 0;
        accountSortOrderArray[3] = 0;
        accountSortOrderArray[4] = 0;
        accountSortOrderArray[5] = 0;

        // Set attributes

        list.setColumnCount(6);

        list.setColumnTitle(0, "Account Name");
        list.setColumnWidth(0, 40);
        list.setColumnDSName(0, "name");

        list.setColumnTitle(1, "Revenue");
        list.setColumnWidth(1, 70);
        list.setColumnDSName(1, "revenue");
        list.setColumnFormat(1, "$#,###");
        list.setColumnAlignment(1, 1);  // align_right

    list.setColumnTitle(2, "Flag");
        list.setColumnWidth(2, 40);
        list.setColumnDSName(2, "boolean");
        list.setColumnAlignment(2, 2);  // align_center

        list.setColumnTitle(3, "Created");
        list.setColumnWidth(3, 100);
        list.setColumnDSName(3, "created");
        list.setColumnFormat(3, "%c");

        list.setColumnTitle(4, "Int16");
        list.setColumnWidth(4, 60);
        list.setColumnDSName(4, "int16");

        list.setColumnTitle(5, "UInt16");
        list.setColumnWidth(5, 60);
        list.setColumnDSName(5, "uint16");
    }

    function exitPage()
    {
        extendedAccountDb.close();
    }

    function onClickHandler(e)
    {
        var list;
```

```
var component;
var i;
var n;
var data;
var colId;

list = document.getElementById("accountTable");

// Figure out where the click happened
// Passing (x, y) in document coordinate system

component = list.getComponent(e.clientX, e.clientY)

// Possible component values:
//  TARGET_NONE = 0,
//  TARGET_PREV_PAGE_BTN = 1,
//  TARGET_NEXT_PAGE_BTN = 2,
//  TARGET_HEADER = 3,
//  TARGET_ITEM = 4,
//  TARGET_SCROLL_UP_BTN = 5,
//  TARGET_SCROLL_DOWN_BTN = 6,
//  TARGET_SCROLLBAR_BUBBLE = 7,

if (4 == component) {

    n = list.getSelectionDbSetIndex();
    extendedAccountDb.moveTo(n);
    data = extendedAccountDb.getStringField("name");

    alert("row selected - " + data);

    // Don't call default handler after this since we are done.
    return false;
}
else if (3 == component) {

    var hasSelection = false;
    var selectionDbIndex = -1;
    var selectionId = null;

    // save current selection
    n = list.getSelectionIndex();
    if (-1 != n) {
        hasSelection = true;
        selectionDbIndex = list.getSelectionDbSetIndex();
        extendedAccountDb.moveTo(selectionDbIndex);
        selectionId = extendedAccountDb.getStringField("name");
    }

    // Figure out which column the user clicked on
    colId = list.getColumnId(e.clientX, e.clientY);

    //  SORT_NONE = 0,
    //  SORT_ASCENDING = 1,
    //  SORT_DESCENDING = 2,

    for (i = 0; i < 6; i++) {
        if (colId == i) {
            if (1 == accountSortOrderArray[i]) {
                accountSortOrderArray[i] = 2;
            }
            else {
                accountSortOrderArray[i] = 1;
            }
```

```
                }
                else {
                    accountSortOrderArray[i] = 0;
                }
            }

            // Set sort order for the db set
            extendedAccountDb.setSort(accountCols[colId],
                accountSortOrderArray[colId] == 1);

            // Set sort order for the column
            list.setColumnSortOrder(colId, accountSortOrderArray[colId] == 1);

            // Refresh mimelist to use the modified dbset
            list.refreshDbSet();

            // Set mimelist selection

            if (hasSelection) {
                var filter = "name = " + selectionId;
                var search = extendedAccountDb.createSearch(filter);
                extendedAccountDb.moveTo(0);
                if (extendedAccountDb.find(search)) {
                    var index = extendedAccountDb.index;
                    list.setSelectionDbSetIndex(index);
                }
            }

            // Force a redraw of the mimelist
            list.refreshScreen();

            // Don't call default handler after this since we are done.
            return false;
        }
        else {
            // Call default handler since we haven't handled this event
            return true;
        }
    }

    function setFilter()
    {
        var list;
        var filterInput;
        var filter;
        var search;

        list = document.getElementById("accountTable");
        filterInput = document.getElementById("filterInput");

        if ((null == filterInput.value) > (0 == filterInput.value.length)) {
            search = extendedAccountDb.setFilter(null);
        }
        else {
            // Set filter for the db set
            filter = "lastName ~ " + filterInput.value;
            extendedAccountDb.moveTo(0);
            search = extendedAccountDb.createSearch(filter);
            extendedAccountDb.setFilter(search);
        }

        // Refresh mimelist to use the modified dbset
        list.refreshDbSet();
        // Force a redraw of the mimelist
        list.refreshScreen();
    }

    </script>
    </head>

    <body onload="initPage()" onUnload="exitPage()">


    <form>
    <table width="350" cellspacing="0" cellpadding="0" border="0">
    <tr>
        <td align="right">
            <input style="background-color:#ffffcc;font-family:verdana;
                font-size:9px;padding-top:2px;padding-bottom:1px;"
                type="text" name="filter" id="filterInput" value=""
                size="8">

            <input type="button" value="Set Filter" onclick="setFilter()">
        </td>
    </tr>
    </table>
    </form>

    <form>

    <object id="accountTable" type="application/avantgo-mime-list"
        width="250" height="100"
        style="vertical-align:baseline;padding-left:3px;
        border-color:black;border-width:1;background-color:white;
        color:black;font-size:9px;font-family:verdana;">

    <param name="numRows" value="5"></param>
    <param name="rowMargin" value="2"></param>
    <param name="colMargin" value="4"></param>
    <param name="headerBgColor" value="#66cc99"></param>
    <param name="headerBdColor" value="#000000"></param>

    <param name="cellBgColor" value="#f9f9f9"></param>
    <param name="cellBdColor" value="#cecece"></param>
    <param name="cellSelectBgColor" value="#cecece"></param>
    <param name="cellSelectBdColor" value="#f9f9f9"></param>
    <param name="scrollbarBdColor" value="#000000"></param>

    <param name="scrollbarBgColor" value="#666666"></param>
    <param name="scrollbarButtonColor" value="#66cc99"></param>
    <param name="scrollbarArrowColor" value="#000000"></param>
    <param name="scrollbarBubbleColor" value="#66cc99"></param>
    </object>

    </form>

    </body>
    </html>
```

Using a Symbol scanner

M-Business Anywhere provides you with a `PODSSymbolScanner` interface that implements a Symbol Technologies API for both Palm OS and Microsoft OS platforms.

This custom object is an M-Business Anywhere-specific extension to JavaScript and is a non-standard JavaScript element, imported from the corresponding PODS object.

The `PODSSymbolScanner` object provides the API for working with a Symbol Technologies bar-code scanner device. Your POD might allow users to update information in a POD-specific database by scanning in that new information. The `PODSSymbolScanner` object provides a way to do this. For API details, see "PODSSymbolScanner calls" [*M-Business Anywhere API Reference*].

Note
You must perform a binary download of the `scanner.dll` file into the `/pods` directory for Microsoft OS devices.

You access the `PODSSymbolScanner` object by using the `PODSAvantGo` method `createObject( )` — for more information, see "createObject( )" [*M-Business Anywhere API Reference*]. Because the `PODSAvantGo` object creates the `PODSSymbolScanner` object, you do not have to free the memory associated with it.

The `PODSSymbolScanner` object's API is essentially a wrapper for the Scan Manager API provided by Symbol Technologies. Refer to the Symbol Technologies documentation for complete descriptions of how to use the Scan Manager.

Capturing signatures with an in-line scribble widget

M-Business Anywhere provides you with an in-line scribble widget that allows signature capture on the device at the scribble label level. There is no large dialog that pops up, so the signer of the document is clearly signing the agreed-upon document.

You can create an in-line scribble widget directly through HTML or by using the `DEFAULTCHECKED` attribute in JavaScript.

HTML

If you are creating an in-line scribble widget directly through HTML, you will be happy to know that you can actually go ahead and use the word `INLINE` in your `INPUT` tag. This is because our HTML parser will convert that `INLINE` attribute into the `DEFAULTCHECKED` attribute for you. For example:

```
<input type="scribble" width=200 height=80 name="jot1" inline>
```

is the same as

```
<input type="scribble" width=200 height=80 name="jot2" defaultchecked>
```

JavaScript

If you want to change a regular scribble widget into an inline scribble widget through JavaScript, you have to use the `DEFAULTCHECKED` attribute. For example:

```
if (myForm.jot1.defaultChecked)
    myForm.jot1.defaultChecked = false;
else
    myForm.jot1.defaultChecked = true;
```

Windows Mobile Pocket PC — full screen mode and shortcut menu

On Windows Mobile Pocket PC devices, M-Business Client supports the tap-and-hold shortcut menu. To avoid missing the first few pixels of the signature capture, you must disable this shortcut menu support, by setting the `PODSPref` object's `DisableRightClickPopup` key to `true` on the page where the signature is to be captured. For information on setting `PODSPref` object keys, see “PODSPrefs object” [*M-Business Anywhere API Reference*].

Tip

The shortcut menu is the only way for a user to get out of full screen mode on a Windows Mobile Pocket PC device. Therefore, when the user puts the device in full screen mode, M-Business Client automatically sets `DisableRightClickPopup` to `false` and this setting takes precedence over your setting for the scribble widget.

To ensure that users are able to capture every pixel of their signatures on Windows Mobile Pocket PC devices, add a note at the top of the page recommending that they exit full screen mode before signing.

If you are supporting multiple device types and want to have such a note appear only on Mobile Pocket PC devices, see "Customizing content for users and devices" on page 159.

Using Start Sync and End Sync URLs to perform processing before and after channel synchronization

Beginning with M-Business Anywhere Version 6.0, developers have the ability to set up processing that will be performed at either of two different points:

- Just before the channel synchronizes — the Start Sync URL.
- Immediately after the channel synchronizes, before any forms submitted in the synchronization are processed — the End Sync URL.

Note
Start Sync and End Sync URLs can be implemented only on group channels.

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

You can set up any type of back-end processing that you need to be performed in conjunction with a channel synchronization in order to achieve better control of your application. For example, a Start Sync URL could reserve memory or initialize variables for the synchronization. An End Sync URL could clean up memory, end processes, or ensure that submitted forms are processed in the same order in which the user entered them.

Controlling processing sequence of submitted forms

If a user who is working offline submits multiple forms, after synchronizing these forms may end up being forwarded from M-Business Server to the server hosting your channel in a sequence that is very different from the order in which the user filled them in and clicked the Submit button. This could cause serious problems for many applications where, for example, any of the following are:

- User could change the same record twice.
- User could first add a record, then delete it before synchronizing.
- Servicing customers is involved — in general, customers should be serviced in the order in which they ask for service.

You can set up a back-end process, launched by the Start Sync URL, that intercepts the form submissions and sequences them by date/time stamp before they are forwarded to the server that will process them.

Professional Services assistance may be needed

The back-end processes that you set up to work with Start Sync and End Sync URLs must mesh with internals of the channel synchronization process that are beyond the scope of this guide to document. Because of this complexity, you may want to consult with iAnywhere Professional Services before attempting an implementation. For contact information, see "Application development — customizing iAnywhere software" on page xv.

Customizing and rebranding M-Business Client

Parts of M-Business Client that can be customized or rebranded

The following M-Business Client features can be modified:

- **Home page** — you can customize the initial page displayed when M-Business Client starts up, by either modifying the default page or by displaying an entirely different page.
- **Menus** — given the menu items that are available on a particular M-Business Client platform, you can selectively enable and disable menu items, change the labels displayed, and remove menu items so that they do not even appear.
- **Startup icon** — you can replace the default M-Business Client startup icon on user devices with another image, if you write a custom launcher application.
- **Startup error message** — if you write a custom launcher application, you can also customize the message that appears when M-Business Client starts up and is unable to display the home page.

Using a different home page

You can point M-Business Client to an entirely different home page, that is customized to your exact needs. The home page is specified in the M-Business Anywhere database.

To change the home page specification, first, determine whether you can use the same home page file for all the device platforms you support:

- If you can use the same home page file for all the device platforms that you support, go to the next section, "Setting up a single home page" on page 239.
- If you want to have a separate, optimized home page for each supported platform, go to "Setting up separate home pages for each platform" on page 240.

Setting up a single home page

Follow the procedure below if you can use the same home page file for all the device platforms that you support.

♦ To set up a single home page

1. Determine the location in your web server file system corresponding to the URL that you want your custom home page to have.

   You may place this page anywhere that is convenient. You may also name this page anything you like, for example, *index.html*.

2. Create the custom home page in that location, or copy it from elsewhere.

3. Continue with "Pointing M-Business Server to your custom home page(s)" on page 240.

Setting up separate home pages for each platform

Follow the procedure below if you want to have a separate, optimized home page for each supported platform.

♦ To set up separate home pages for each platform

1. First create a directory branch on a web server with subdirectories for each device platform that you support.

> **Note**
> You must name the directories for the platforms exactly as specified in "Using server URL macros to customize content" on page 166. For example, if you place the directory structure directly below the web server's content root, you would have the following directories if you support Palm OS, Windows Mobile Pocket PC, and Windows XP devices:
> *<Web_server_content_root>*
> *PALM_OS* (for Palm OS devices)
> *WIN32_OS* (for Windows XP devices)
> *WINCE_OS* (for Windows Mobile Pocket PC devices)

2. Next, create a custom M-Business Client home page for each device platform that you support and place it in the appropriate directory.

> **Note**
> You may name the home page files anything you like, but all home page files must have the same name, for example, *index.html*.

3. Continue with the next section, "Pointing M-Business Server to your custom home page(s)" on page 240.

Pointing M-Business Server to your custom home page(s)

Follow the procedure below to change the M-Business Client home page specification in the M-Business Server database.

♦ To point M-Business Server to your custom home page(s)

1. Run the SQL Anywhere SQL Query Tool on the machine where M-Business Server is installed:

   - Windows:***<M-Business_Home>/ASA/win32/dbisqlc.exe***
   - Unix:***<M-Business_Home>/ASA/bin/dbisqlc***

Refer to the *ASA Database Administration Guide*, referenced in "Related publications" on page xi, for instructions on starting this utility and connecting to the M-Business Server database, located at *<M-Business_Home>/ASA/data/agdb.db* on the server.

You will need to log in to the database as:

- User ID: **`agdb`**
- Password: **`password`**

  If you have changed the default password for the `agdb` user, enter the new password in place of `password`.

2. In the command window, enter one of the following SQL commands, depending on whether you have set up a single home page, or separate home pages for each platform.

> **Note**
> URLs entered in any M-Business Anywhere component must contain only forward slashes (/). The backslash character (\) in place of a forward slash will cause an error.

- **Single home page for all platforms** :

```
UPDATE AGRESERVEDSUBS SET url =
    'http://www.<name_of_this_server>/<path>/index.html'
    WHERE ID = 1
```

  If you have used a file name other than *index.html* for the home page, use that name in the above command.

- **Separate home pages for each platform** :

```
UPDATE AGRESERVEDSUBS SET url =
    'http://www.<name_of_this_server>/<path>/AG_DEVICEOS/index.html'
    WHERE ID = 1
```

  If you have used a file name other than *index.html* for the home page(s), use that name in the above command.

> **Note**
> When M-Business Server retrieves the page for the last URL above, it substitutes a value for `AG_DEVICEOS` based on the platform of the requesting client device. This pulls the correct version of the file from the specific directory that you set up in "Setting up separate home pages for each platform" on page 240 — *PALM_OS*, *WIN32_OS*, or *WINCE_OS*.

Your M-Business Server will now download your custom home page(s) to M-Business Client on user devices when they synchronize.

Customizing M-Business Client menus

You can use the menu customization functions in the M-Business client extension API to customize the M-Business Client menus.

Using JavaScript to customize menus

From JavaScript, `menu` is the name of the JavaScript object. You do not need to instantiate the `menu` object before accessing it from JavaScript.

Customizing the main menu

At the level of the M-Business Client main menu, you can control remove a menu so that it does not even appear. On platforms where a user action is required to display the M-Business Client main menu, you can customize the menu softkey label.

JavaScript commands that access the `menu` object to customize the main menu take the following form:

```
menu.menu_function
```

Valid values for *menu_function* are given in the following table.

Values for menu_function to use with the M-Business Client main menu

menu_function	Data type	Description
`label`	string	The menu softkey label displayed on platforms where a user action is required to display the M-Business Client main menu.
`isAvail()`	boolean	False for a menu that has been removed; true otherwise.
`remove()`	void	Removes the specified menu from the M-Business Client main menu. The menu is automatically restored in the user's next M-Business Client session, but cannot be restored in the current session.

Customizing individual menu items

JavaScript commands that access the `menu` object to customize individual menu itens take the following form:

```
menu.menu_item.menu_item_function
```

Valid values for *menu_item* and *menu_item_function* are given in the following two tables.

Values for menu_item_function to use with individual menu items

menu_function	Data type	Description
`enabled`	boolean	Returns true if *menu_item* is enabled; false otherwise. Setting to true enables *menu_item*; setting to false disables it. For some menu items that must be kept in synch with the M-Business Client state, your setting will be ignored the next time the user accesses the menu that contains *menu_item*.
`label`	string	The label displayed for *menu_item*.
`isAvail()`	boolean	False if *menu_item* is unavailable on the current M-Business Client platform, or has been removed; true otherwise.
`remove()`	void	Removes *menu_item* from display in its M-Business Client menu. Removed menu items are automatically restored in the user's next M-Business Client session, but cannot be restored in the current session.

Values for menu_item

menu_item	Default menu label
`about`	About
`addBookmark`	Add Bookmark
`back`	Back
`bookmarkManager`	Bookmark Manager
`cacheManager`	Cache Manager
`channelManager`	Channel Manager
`copy`	Copy
`cut`	Cut
`exit`	Exit
`find`	Find
`findNext`	Find Next
`findPrevious`	Find Previous
`formManager`	Forms Manager
`forward`	Forward

menu_item	Default menu label
`fullScreen`	Full Screen
`goHome`	Home
`help`	Help
`openPage`	Open Page
`pageOption`	Page Options
`paste`	Paste
`reloadPage`	Reload Page
`selectAll`	Select All
`serverOption`	Server Options
`syncAll`	Sync All
`workOffline`	Work Offline
`workOnline`	Work Online

Note
Some menu items are not available across all M-Business Client platforms. For example, the Home menu item is not available on Windows XP devices. An attempt to access a missing menu item simply results in a no-op and does not cause any exceptions. Use the `isAvail()` method can use to test for the availability of a given menu item.

Note
The valid labels for *menu_item* in the table above are based on their respective labels on the in M-Business Client on the Windows Mobile 5 and higher platform family. Some menu items have different labels on other M-Business Client platforms. For example, the Page Options menu item is called Preferences, or just Options, on some platforms.

If you attempt to access a menu item that is not available, it results in a no-op that does not cause any exceptions. You can use `isAvail()` to determine the availability of a menu item.

Examples

Disable the Forms Manager menu item if it is enabled:

```
if(menu.formManager.enabled)
    menu.formManager.enabled = false;
```

Remove the Forms Manager menu item completely:

```
menu.formManager.remove();
```

Change the label of the Sync All menu item:

```
menu.syncAll.label = "Update all channels";
```

Customizing menus using C and PODS

If you have already decided to develop your application in C using PODS, you can call all the menu customiztion functions from C as well as JavaScript. For more information, see “PODSMenu object” [*M-Business Anywhere API Reference*] and “PODSMenuItem object” [*M-Business Anywhere API Reference*].

If you need some guidance on when to use JavaScript versus C, see “JavaScript versus PODS” [*M-Business Anywhere API Reference*].

Writing a custom launcher

If you write a custom launcher application for M-Business Client, you can have M-Business Client use a custom icon and display a custom message if the home page is unavailable. It is necessary to write a launcher application specific for the device operating system. You can write such a launcher application using any language you choose that can produce a binary executable for the target platform. iAnywhere Professional Services also offers this service.

Each supported platform must have a separate version of the launcher application that is compiled specifically for that platform. For information on getting the correct version of your custom launcher downloaded to different user devices, see “Setting up binary files for delivery to M-Business Client” on page 170.

M-Business Client executable

Your custom launcher's primary task is to start the M-Business Client executable on the device. The name of this program is platform-specific:

- Palm OS devices: ***avantgo.prc***
- Microsoft OS devices: ***avantgo.exe***

Strings to be defined

You will need to incorporate into your custom launcher the following key text strings:

- Label displayed for icon on the device.
- Message text for error that occurs when the M-Business Client home page is not present, and presumably as not been synchronized to the device.
- Message text for error that occurs when the M-Business Client executable is not found, and presumably has not been installed.

Icons for Microsoft OS devices

Follow the guidelines below to use custom icons with your custom launcher on Microsoft OS devices:

- Design two icons: 16 x 16 pixels and 32 x 32 pixels.
- Render each icon in 4-bit color and 8-bit color; optionally, render each in 24-bit color as well.
- Save these icons in the *.ico* file format, as generated by Microsoft Developer Studio.

Icons for Palm OS

Follow the guidelines below to use custom icons with your custom launcher on Palm OS:

- Design two icons: 9 x 5 pixels and 32 x 32 pixels.

  The larger icon, with32 x 32 pixels, will have a visible area of only 22 x 22 pixels in the default view. The smaller icon, 9 x 15 pixels, is used in the "icon view".
- Design high-density versions for the above two icon sizes — double both dimensions to make the versions for hi-density devices.
- Render each icon in 1-bit black-and-white and 8-bit color; optionally, render each in 2- and 4-bit grayscale as well.
- Save these icons in the *.bmp* file format.

  On most Palm devices, the background of the launcher icon is usually white, but this is not necessarily the case. If you want areas of your bitmap to be transparent, select a color to be treated as transparent.

CHAPTER 11

Using UltraLite for M-Business Anywhere for on-device data

Contents

Summary of trade-offs, UltraLite vs. M-Business XML datastore

The decision on whether to use UltraLite or M-Business XML datastore depends on the needs of the developer.

UltraLite is a robust database solution with a high performance synchronization engine that will scale for thousands of users. UltraLite has features, like indexing, which allow searching and improved responsiveness with much larger datasets, rich dynamic SQL support, transaction processing to comply with the existing enterprise rules, and other features expected of a full-featured database. MobiLink also offers robust, flexible bi-directional synchronization with enterprise data stores.

The M-Business XML datastore is simple to develop with but provides only one-way synchronization of data. Its main advantage is that it does not require a pre-loaded schema on the device.

The table below summarized the trade-offs between UltraLite and M-Business XML datastore.

For more information on UltraLite, see the*UltraLite M-Business Anywhere Programming* and UltraLite tutorials, referenced in "Related publications" on page xi. The sample that follows is focused on recently introduced features that improve the deployment and user experience of using these technologies.

For more information on the M-Business XML datastore, see "Using M-Business XML conduit and API" on page 261, and "Building applications with M-Business XML conduit" on page 273.

Table 1. Comparison of UltraLite and M-Business XML datastore

Feature	UltraLite on-device database	M-Business XML datastore
Basic functionality	Full-featured RDBMS: indexes, referential integrity, transactions (rollbacks and commits)	Minimal datastore: simple to develop with but provides only one-way synchronization of data and does not scale well for large data volumes
Flexibility	Very flexible; can be configured in many different ways	Limited flexibility
Ideal back-end datastore	Enterprise database (Oracle, MS SQL, DB2, ASE, ASA, etc.)	Back-end datastore that can emit XML upon receiving an HTTP request
Interface to back-end database	Standard DB interface (ODBC); highly efficient	Custom XML data stream; inefficient
Scalability	Scales easily for thousands of users	Limited scalability
Simple for those familiar with...	SQL and relational databases	XML and ADO

Feature	UltraLite on-device database	M-Business XML datastore
Simplicity of use and provisioning	More complicated — pre-defined schema required for synchronization with MobiLink	Relatively simple — no schema required
Synchronization	Two way; extremely efficient	One way; requires HTTP forms and conversion of back-end information into a specific XML format for upstream data; inefficient

UltraLite POD for SQL Anywhere 10

An UltraLite POD (ULPOD — compiled, platform-specific code) is available for use with M-Business Anywhere version 6.2 and the UltraLite in SQL Anywhere version 10. ULPOD does not work with earlier versions of UltraLite.

ULPOD greatly increases the former limit of 65 thousand records to approximately 16 million. And this version of UltraLite stores the schema information within the database file and provides better support for more character sets. For more information, see *UltraLite M-Business Anywhere Programming*, referenced in “Related publications” on page xi.

Sample application using UltraLite for M-Business Anywhere

Note
This sample code is available online. See "One-button synchronization of UltraLite and M-Business Anywhere using MobiLink Redirector" on page 386.

Note
For a detailed explanation of One Button Synchronization, see http://www.ianywhere.com/developer/product_manuals/sqlanywhere/0902/en/html/ulagen9/00000037.htm.
For MobiLink Redirector, see http://www.ianywhere.com/developer/product_manuals/sqlanywhere/0902/en/html/dbmlen9/00000113.htm.

This sample application uses the MobiLink Redirector and one-button synchronization features of M-Business Client, version 5.5 and higher, along with MobiLink in Adaptive Server Anywhere. For basic information on using the UltraLite API for M-Business Anywhere, see *UltraLite M-Business Anywhere Programming*, referenced in "Related publications" on page xi. That document includes a tutorial and complete details on the UltraLite for M-Business Anywhere interface.

This sample application demonstrates how to develop and deploy a basic web application using M-Business Client and M-Business Server to deploy the user interface, along with MobiLink data synchronization and UltraLite remote database to synchronize with a consolidated Adaptive Server Anywhere back-end database.

Note
It is highly recommended that you complete the *UltraLite for M-Business Anywhere Tutorial*, referenced in "Related publications" on page xi, before working with this sample code. That tutorial goes into much greater detail on the implementation of UltraLite and M-Business Anywhere. This sample code is more advanced and focuses on the features introduced in Adaptive Server Anywhere (ASA) 9.02

System requirements

- **M-Business Anywhere Application Edition**, version 5.5 or higher, running on a supported platform.
- **Adaptive Server Anywhere**, version 9.02 and above. This is available in a bundled developer package. In your Adaptive Server Anywhere installation, you need to install the samples in order for this sample to operate.
- **Web server** capable of serving HTML web pages. The IIS web server on Windows 2000 is suitable.

Note
If you are serving this POD from IIS, you must ensure that *.dll* extensions are retrieved as `application/octet-stream`.

- **Microsoft OS device or Tablet PC** supported by M-Business Client, version 5.3 and above.

Download sample files and configure the web server

♦ To download sample files and configure the web server

1. Download sample files.

   See "One-button synchronization of UltraLite and M-Business Anywhere using MobiLink Redirector" on page 386 for the sample code URL.

2. Unzip files to a local work directory on your development or test box that has IIS server running.

3. Create a virtual directory to point to the work directory just created.

   We'll assume here that the virtual directory you are using is *exercise_app*.

4. Set up IIS MIME types (*.dll* and *.prc* maps to `application/octet-stream`) to allow binary download.

   For details, see "Configuring M-Business Server" [*M-Business Anywhere Administrator Guide*].

Configure MobiLink, back-end database, and M-Business Server to synchronize

After unpacking the sample files and configuring the web server, the sequence of operations to configure MobiLink, the Adaptive Server Anywhere consolidated database, and M-Business Server to synchronize is as follows (all page numbers refer to the *UltraLite M-Business Anywhere Programming*, referenced in "UltraLite for M-Business Anywhere" on page xi):

- **Task 1 - Configure ASA consolidated database, MobiLink synchronization server, and M-Business Server (page 206)** and add a Publication that the MobiLink synchronization engine will use to synchronize the remote database.

- **Task 2 - Create the ODBC Data Source (page 207)** that the remote database will synchronize through to the consolidated database.

- **Task 3 - Start the ASA Consolidated Database (page 207)** process.

- **Task 4 - Start the MobiLink process and generate the MobiLink scripts (page 207)**.

- **Task 5 - Create the UltraLite schema file and copy to operating system-specific folder for binary download (page 208)** — this is required on the device to create the UltraLite database.

- **Task 6 - Configure M-Business Server to redirect MobiLink synchronization (page 208)** requests to the appropriate MobiLink server.

- **Task 7 - Configure one-button synchronization through HTTP of M-Business Client and MobiLink (page 209)** to initiate a MobiLink synchronization as part of the M-Business Anywhere synchronization. The order of synchronization is:

a. MobiLink synchronization

b. Web content

c. Binary files

d. XML datastore retrieval

- ♦ **Task 8 - Create an M-Business Server group for your channel (page 210)**, to define the group to which your channel content will be delivered.
- ♦ **Task 9 - Create an M-Business channel for the group created above (page 210)**, to define the channel to the group to which its content will be delivered.
- ♦ **Task 10 - Add your M-Business user account to the group created above (page 211)**, to define yourself as the first member of the group to which your channel content will be delivered.

Task 1 - Configure ASA consolidated database, MobiLink synchronization server, and M-Business Server

♦ To configure ASA consolidated database, MobiLink synchronization server, and M-Business Server

- Create the ASA consolidated database and add a Publication.

Within the install directory, locate the ***reload.sql*** script:

```
CREATE TABLE "DBA"."exeuser"
(
    "exeuserid"     uniqueidentifier NOT NULL DEFAULT newid(*),
    "fname"         varchar(100) NULL,
    "address"       varchar(200) NULL,
    "weight"        integer NULL,
    "height"        varchar(10) NULL,
    "bodyfat"       integer NULL,
    "datecreated"   date NULL,
    PRIMARY KEY ("exeuserid"),

)
go
CREATE TABLE "DBA"."bikeride"
(
    "bikerid"       uniqueidentifier NOT NULL DEFAULT newid(*),
    "ridedate"      date NULL,
    "fname"         varchar(100) NULL,
    "miles"         integer NULL,
    "hrate"         integer NULL,
    "ridelength"    integer NULL,
    "loop"          varchar(100) NULL,
    "notes"         varchar(1000) NULL,
PRIMARY KEY ("bikerid"),

)
go
CREATE PUBLICATION "DBA"."exeuser"
(
    TABLE "DBA"."bikeride"( "bikerid", "ridedate", "fname",
        "miles", "hrate", "ridelength", "loop", "notes" ),
    TABLE "DBA"."exeuser"( "exeuserid", "fname", "address",
        "weight", "height", "bodyfat", "datecreated" )
);
go
commit;
```

The key functions create two tables (`exeuser` and `bikeride`) within the `exercisedb` consolidated database, and a publication named `exeuser`. The publication is used to synchronize this database using MobiLink.

Task 2 - Create the ODBC Data Source

This is a command line with parameters and is found in the *createDSN.cmd* file. Executing this file first runs the *setEnv.bat* file (errors in this file will interfere with execution of the command line), and then executes the following command line:

```
%ASANY9%\dbdsn -w "exesample"
         "uid=DBA;pwd=SQL;dbf=exercisedb.db;eng=exercisedb"
```

Notice that the parameters include the data source name `exesample`, the default user name and password, and the database name. For more information on these parameters, see the *SQL Anywhere Studio* documentation, referenced in "Related publications" on page xi.

Task 3 - Start the ASA Consolidated Database

Do this by executing the *start_database.bat* file. This batch file is primarily built around this single command line:

```
dbeng9 exercisedb.db
```

This command starts up the consolidated database and you should see an icon appear in your lower right hand corner indicating the database is running. There are many parameters that can be used in conjunction with the *dbeng9* executable. For more information, see the *SQL Anywhere Studio* documentation, referenced in "Related publications" on page xi.

Task 4 - Start the MobiLink process and generate the MobiLink scripts

We will utilize command line parameters to configure the synchronization. This is done by executing *ml.bat*. The operable line in this batch file is:

```
dbmlsrv9 -c "uid=dba;pwd=sql;dsn=exesample"
-x http(port=82) -o mlserver.mls -v+
-dl -za -zu+
```

Where the following parameters configure MobiLink:

♦ The first set of parameters following `-c` (`"uid=dba;pwd=sql;dsn=exesample"`) represent the ODBC connection to the consolidated database that will be used.

- The `-x` parameter specifies the network protocol that MobiLink should listen to. TCP/IP is the default and with this command we instruct MobiLink to use HTTP listening to port 82. The default is port 80, but by default, the M-Business Sync Server binds to this port.
- The `mlserver.mls` file is the MobiLink logging file to use.
- The command line parameters `-v+ -dl -za -zu+` create the MobiLink scripts for each column of the consolidated database for the data source name `exesample`. This is very convenient for rapidly mobilizing a database to the device.

Task 5 - Create the UltraLite schema file and copy to operating system-specific folder for binary download

The UltraLite schema file is necessary for MobiLink synchronization, as well as creating the UltraLite on-device database. This is created by executing the *createpub.bat* file. The key command within this batch file is:

```
%ASANY9%\ulinit
-c "dsn=exesample;uid=dba;pwd=sql"
-f ulexe.usm -n exeuser
```

- This command generates the *ulexe.usm* file, which later needs to be retrieved to the device. By default, this script creates *ulexe.usm* in the same folder in which the script is executed. For purposes of simplicity in this example, copy this file to the *web/WINCE_OS* and *web/WIN32_OS* locations.
- Binary download of the file is done by creating a device OS-specific *ulpod.htm* file for each device operating system and fetching an anchor `href` for *AG_DEVICEOS/ulpod.htm*. One could imagine copying this to a folder based on the M-Business user name and then using the `AG_USERNAME` macro capability to fetch the UltraLite schema specific to each user.

Task 6 - Configure M-Business Server to redirect MobiLink synchronization

This process is detailed in the *MobiLink Administration Guide*, in the "M-Business Anywhere Redirector" topic. In brief, there are four steps to enabling MobiLink redirection within M-Business Server:

a. Configure the *sync.conf* file within M-Business Anywhere to load the `iaredirect` module and the redirect location. It is also advisable to change the `LogLevel` to `debug`. This will output information regarding the MobiLink synchronization.

b. Copy the *mod_iaredirect.dll* binary into the bin location.

c. Configure and copy the redirector.config file into the *<M-Business_Home>\conf* directory with appropriate port information for M-Business Anywhere and MobiLink.

d. Confirm the operation of the redirector by using a browser to navigate to http://localhost/iaredirect/ml, which is the redirect location defined by default. You will receive an error in the *<M-Business_Home>/logs/sync_error.log* as follows:

```
[Thu Oct 28 13:43:59 2004] [error] [client 10.22.61.76] (2)No
         such file or directory: mod_iaredirect: ml-client-id
```

```
header NOT provided or could not be allocated, aborting
request..
```

Task 7 - Configure one-button synchronization through HTTP of M-Business Client and MobiLink

When synchronizing with Microsoft OS platforms, the sample retrieves the *ulpod.cab* file. This file extracts itself upon retrieval through M-Business Anywhere, and copies *ulconnect.exe*, *ulconnect.usm*, and *ulpod9.dll* into the */Program Files/AvantGo/PODS* directory, and *ul9rt.dll* to the Microsoft OS device's */Windows* directory.

After extracting this package, you need to set the synchronization parameters. This is done by executing the following JavaScript as part of the application:

```
conn.syncParms.STREAM_TYPE_HTTP;
conn.syncParms.setUserName(document.dbsync.syncusrname.value );
conn.syncParms.setVersion(document.dbsync.syncver.value );
conn.syncParms.setStreamParms(document.dbsync.syncparm.value );
conn.syncParms.setMBAServer('<your_MBA_server_URL>',
        '<your_redirector_port>','/iaredirect/ml');
conn.saveSyncParms();
```

Where *<your_MBA_server_URL>* is the URL or IP address of your M-Business Server, and *<your_redirector_port>* is the redirector port it is using.

In this particular application, the synchronization parameters are set within the *sync_page.htm* file. For ease of use, it is convenient to set these parameters in the `OnLoad` event of the first page and use your application server to pre-populate the synchronization parameters with appropriate values. One can imagine setting these parameters using the `X-AvantGo-User` header information and using this user name within M-Business Server as the UltraLite user name.

After executing this JavaScript, MobiLink is configured to synchronize at the same time as an M-Business Client synchronization. Thereafter, at the beginning of a typical M-Business Anywhere synchronization (cradle or wireless), *ulconnect.exe* is executed and synchronizes MobiLink with the parameters saved, along with M-Business Anywhere content through HTTP.

Once MobiLink is configured to listen to HTTP, the redirector is properly configured, and the web page is setting the synchronization parameters, UltraLite will synchronize at the same time as M-Business Client.

Note that independent synchronization of the UltraLite on-device database is also available. There is a `SkipMBASync` setting that needs to be set to `True` in order for this to occur. For this and other advanced settings, see *UltraLite M-Business Anywhere Programming*, referenced in "Related publications" on page xi.

Task 8 - Create an M-Business Server group for your channel

The channel identifies the web content (HTML and JavaScript) that will be retrieved to your device in order to run the application.

The web pages contain the interface to UltraLite for M-Business Anywhere (embedded and hidden, or user configurable). M-Business Server retrieves the *ulconnect.cab* file, which contains the UltraLite for M-

Business Anywhere binary, and the *ulconnect.exe* executable, which initiates one button synchronization, along with the UltraLite on-device database schema file.

For further details, see "Managing channels" [*M-Business Anywhere Administrator Guide*].

Task 9 - Create an M-Business channel for the group created above

Use the following settings:

- Channel Title: **Exercise App**
- Channel URL: **http://your_Webserver/exercise_app/main.htm**

> **Note**
> URLs entered in any M-Business Anywhere component must contain only forward slashes (/). The backslash character (\) in place of a forward slash will cause an error. When you are setting up directories in a Windows file system, where backslashes separate levels, be sure to change these characters to forward slashes if you copy parts of the file system path into a URL.

- Channel Size: **100** kb
- Channel Link Depth: **2**
- Allow Binary Distribution: true (**checked**)
- Use defaults for all other properties.

Task 10 - Add your M-Business user account to the group created above

For more information on any of the ASA topics, please review the *SQL Anywhere Studio* documentation, referenced in "Related publications" on page xi.

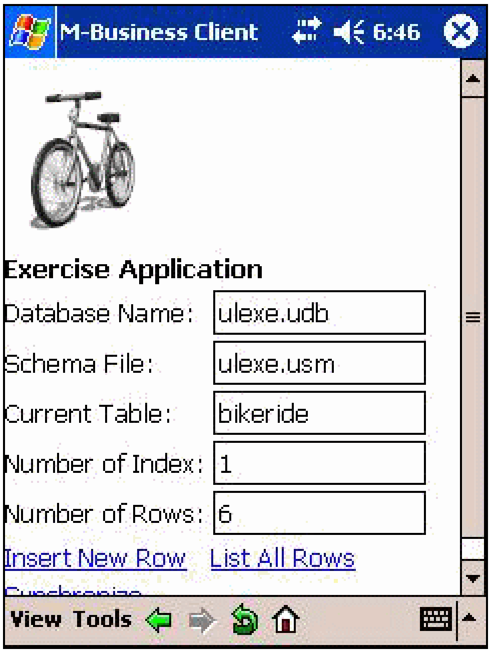
We strongly recommend completing the tutorial included in the *UltraLite M-Business Anywhere Programming*, referenced in "Related publications" on page xi, before running this sample.

Using the sample

♦ To use the sample

1. Synchronize the sample channel to your device.
   - To make sure you can run the sample, confirm that you have the UltraLite Pod version 9.0.2.2751 or above — on the device, from M-Business Client, choose **View»About M-Business**.
   - Synchronize your device.
2. On the M-Business Client home page, tap the **Exercise App** channel link.

- On first opening the channel, you will receive a message that the database was not found and is being created.
- A confirmation dialog appears asking, Do you have network connection right now? By default, synchronization will occur with the localhost computer. If you do not have an active connection, tap **Cancel** — otherwise, you will receive an error. If you do have an active connection and you tap **OK**, a synchronization will be attempted.
- Review the **Exercise Application** start page (*main_page.htm*). This start page contains information on the UltraLite database — the database file name, schema file name, and information about records in the database:



Exercise Application

Database Name: ulexe.udb

Schema File: ulexe.usm

Current Table: bikeride

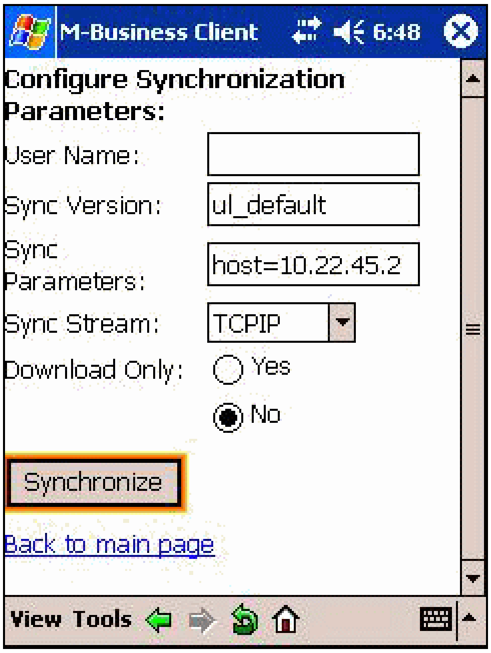
Number of Index: 1

Number of Rows: 6

- Review the **Configure Synchronization Parameters** page (*sync_page.htm*) by tapping the **Synchronize** link on the start page. The rotating activity indicator indicates that the UltraLite database is being synchronized.

Note
You can synchronize either at the same time as M-Business Client or asynchronously.

Fields are described below the page image:



Configure Synchronization Parameters:

User Name:

Sync Version: ul_default

Sync Parameters: host=10.22.45.2

Sync Stream: TCPIP

Download Only: ○ Yes ● No

Synchronize

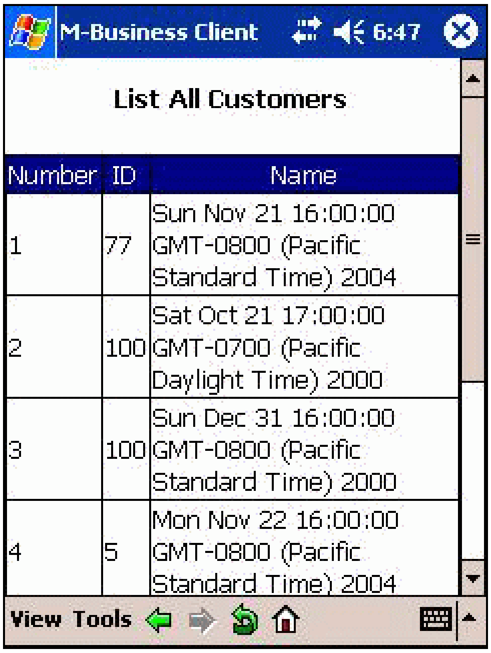
Back to main page

Sync Version corresponds to the value in the MobiLink Connection Scripts. This defines the method of synchronization.

Sync Parameters include the IP Address of the MobiLink server if it is running remotely. Leave blank for MobiLink servers running on the localhost.

Sync Stream is the method in which the device will communicate with the MobiLink server. Acceptable protocols are TCPIP, HTTP, HTTPS, and HOTSYNC. For this sample, we have hard coded the use of HTTP.

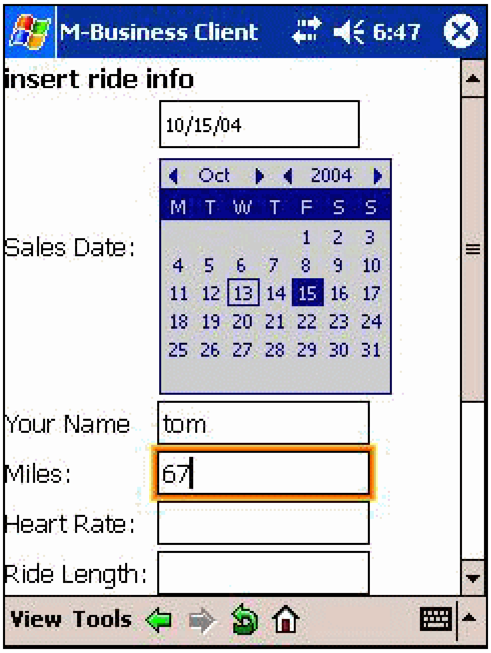
- Review the **List All Customers** page (*list_page.htm*) by tapping the **List All Rows** link on the start page. This page lists the contents of the UltraLite database:



List All Customers

Number	ID	Name
1	77	Sun Nov 21 16:00:00 GMT-0800 (Pacific Standard Time) 2004
2	100	Sat Oct 21 17:00:00 GMT-0700 (Pacific Daylight Time) 2000
3	100	Sun Dec 31 16:00:00 GMT-0800 (Pacific Standard Time) 2000
4	5	Mon Nov 22 16:00:00 GMT-0800 (Pacific Standard Time) 2004

- ♦ Review the **insert ride info** page (*insert_page.htm*) by tapping the **Insert New Row** link on the start page — try inserting a new record:



insert ride info

Sales Date: 10/15/04

Oct					2004	
M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Your Name tom

Miles: 67

Heart Rate:

Ride Length:

Doing a code walkthrough

Make sure you have downloaded the full source code — see "Download sample files and configure the web server" on page 251.

For complete details on the code syntax used within the samples, see the *SQL Anywhere Studio* documentation, referenced in "Related publications" on page xi. That document contains a detailed discussion of the Adaptive Server Anywhere (ASA) database creation, MobiLink script creation, ASA and MobiLink administration, and configuration parameters, along with tutorials and other information to utilize ASA as your consolidated database and MobiLink as your data synchronization.

For further information on UltraLite, please see the *UltraLite M-Business Anywhere Programming* and UltraLite tutorials, referenced in "Related publications" on page xi.

For detailed information on the UltraLite for M-Business Anywhere server, review the UltraLite for *UltraLite M-Business Anywhere Programming*, referenced in "Related publications" on page xi. The tutorial in particular has a more detailed walkthrough of the configuration and interaction of UltraLite with M-Business Client.

For general information on developing web pages and delivering them to mobile devices through M-Business Server, see the rest of this guide.

CHAPTER 12

Using M-Business XML conduit and API

Contents

Overview of M-Business XML conduit

M-Business Anywhere Application Edition provides access to data stored in legacy database systems by supporting synchronization of XML data sources to an on-device XML API. Support for XML provides seamless integration with industry-leading databases, including Oracle, Microsoft SQL Server and Sybase. M-Business XML API provides a uniform, cross-platform method to access data with JavaScript.

With the M-Business XML conduit and the M-Business XML API, you can send data to mobile devices, present that data in M-Business Client, and report changes to the originating database server using the M-Business Anywhere standard form submission mechanism. The M-Business XML conduit and M-Business XML API rely on XML-based standards, including XML-formatted DBMS responses and W3C-approved XML 1.0-compliant schema documents for describing databases.

Caution
If you use the M-Business XML conduit, you must make sure that the supporting code file on the device is the correct one for the version of M-Business Client on the device. For more information, see "M-Business version-specific on-device code files" on page 212.

Note
M-Business XML conduit is not currently supported on Symbian OS.

This chapter provides an overview of the M-Business XML conduit and the M-Business XML API. It is intended to be used in combination with "M-Business XML API reference" [*M-Business Anywhere API Reference*].

For a detailed example complete with sample code, see "Building applications with M-Business XML conduit" on page 273.

Resources

The following resources are available from the M-Business Anywhere Product Manuals page, referenced in "Related publications" on page xi:

- *DatabasePods.zip* - includes debugging tools, header files, *.prc*, and *.dll* files.
- *DB_Sample.zip* - includes the sample code referenced in "Building applications with M-Business XML conduit" on page 273.

Introduction to M-Business XML conduit

The M-Business XML conduit delivers data from a back-end database server (DBMS) to the device. The DBMS must be configured to accept queries over HTTP GETs or POSTs, and must respond to those queries in a particular XML format. All the application developer has to do is configure the DBMS and M-Business Server appropriately, and M-Business Server will be ready to deliver data to the device. For example, a developer might have application data residing in an instance of Microsoft SQL Server. By installing and configuring the Microsoft SQLXML add-on, the application data is ready to be synchronized by M-Business Server.

The M-Business XML conduit obtains XML-formatted data from any source that is able to deliver data over HTTP. More specifically, if you are able to set up a DBMS, a Java servlet, or a web server that can serve properly formatted XML when handed a URL, you have everything you need in order to work with the M-Business XML conduit.

How a database synchronization works

While the M-Business XML conduit moves data from the back-end database to the mobile device, it does not move data in the opposite direction. To report changes in on-device databases, use the M-Business Server standard form submission mechanism. For information on form submissions, see "PODS submission-related objects" [*M-Business Anywhere API Reference*].

M-Business Server will explicitly undo changes made to an on-device database at every synchronization; thus, you may freely modify the on-device data, report the changes in one or more form submissions, and then rely on the server conduit to conform the device's data with the server's data at the next synchronization.

Defining the database channel

XML data alone is not enough to describe a database. For each XML file you create, you also must create an XML Schema Definition (XSD) document, just as a table in a database requires a schema to describe the fields within the table. M-Business Client supports the W3C standard XML Schema, which provides element-centric or attribute-centric formatting for schemas.

Each M-Business XML conduit channel describes a single database table. You must create an XSD file and an M-Business Server database channel for each table you want to synchronize.

To be valid, the XSD file must meet the following guidelines:

- The `name` attribute must be unique. For example, `'uint32Field'` could be `'myBirthDate'`, or `'stringField'` could be `'addressLine1'`.
- The `'theKeyField'` can be any column in the table.
- You are required to have a primary key field in the table. It can have any name, just modify the line, `<xsd:field xpath="@theKeyField"/>`, to match your column name.

The sample XML and XSD documents below illustrate the format and supported data types you should use when creating the XML and XSD files that will describe your database channel.

Sample XML document

```
<xsd:schema xmlns:msch="urn:schemas-microsoft-com:mapping-schema"
    xmlns:dt="urn:schemas-microsoft-com:datatypes"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<?xml version="1.0" encoding="utf-8"?>
<Root>
    <Contacts ID="1" First="Halle" Last="Berry"
        Phone="510-555-2100"
        Email="hberry@localhost" Web="hberry" />
    <Contacts ID="2" First="Sean" Last="Connery"
        Phone="510-555-2101"
        Email="sconnery@localhost" Web="sconnery" />
    <Contacts ID="3" First="Abe" Last="Vigoda"
        Phone="44(0) 2077764235"
        Email="avigoda@localhost" Web="avigoda" />
    <Contacts ID="4" First="A" Last="Godzilla"
        Phone="800-000-0000"
        Email="godzilla@localhost" Web="godzilla" />
</Root>
```

Sample XSD document

```
<xsd:schema xmlns:msch="urn:schemas-microsoft-com:mapping-schema"
    xmlns:dt="urn:schemas-microsoft-com:datatypes"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element
    name="Contacts"
    type="Contacts_type"
    msch:relation="Contacts"/>
<xsd:complexType name="Contacts_type">
    <xsd:attribute name="ID" type="xsd:string"/>
    <xsd:attribute name="First" type="xsd:string"/>
    <xsd:attribute name="Last" type="xsd:string"/>
    <xsd:attribute name="Phone" type="xsd:string"/>
    <xsd:attribute name="Email" type="xsd:string"/>
    <xsd:attribute name="Web" type="xsd:string"/>
    xsd:key name="PrimaryKey">
        <xsd:selector xpath="."/>
        <xsd:field xpath="@ID"/>
    </xsd:key>
    </xsd:complexType>
    </xsd:schema>
```

The above sample XSD document works with the sample data in the Sample XSD document. For a complete list of the XSD data types that are supported, see "Creating the XSD file" on page 276.

Setting null data options

Beginning with version 5.3, the M-Business XML conduit supports null values for data elements. In earlier versions, numeric data elements could have a value of zero and string data elements could have a value of spaces, but they could not be null.

Note
You can specify that null values are allowed on not at the column level. If a column is specified as required, it cannot have a null value. In attribute-based schemas, null values are *allowed* by default; in element-based schemas, null values are *dis*allowed by default.

Caution
If you are supporting users with M-Business Client version 5.2, be sure that you set up your database fields to disable support for null values; M-Business Client version 5.2 will not display nulls correctly.

Allowing null data values

If you want to allow null values in a column:

- **Attribute-based schema** — allows nulls by default; the default is `use="optional"`, so the column below is nullable.

  ```
  <xsd:attribute name="nullableColumn" type="xsd:string"/>
  ```

  You make the data for this column null by omitting any specific value for it from the XML.

- **Element-based schema** — disallows nulls by default; the default is `nillable="false"`, so you must add `nillable="true"`.

  ```
  <xsd:element name="nullableColumn" type="xsd:string" nillable="true"/>
  ```

  You make the data for this column null by using `xsi:nil="true"`.

  ```
  <nullableColumn xsi:nil="true"></nullableColumn>
  ```

Disallowing null data values

If you want to disallow null values in a column:

- **Attribute-based schema** — allows nulls by default; add `use="required"` to the attribute.

  ```
  <xsd:attribute name="oldStyle" type="xsd:string" use="required" />
  ```

- **Element-based schema** — disallows nulls by default; the default is `nillable="false"`, so the column below is not nullable.

  ```
  <xsd:element name="oldStyle" type="xsd:string"/>
  ```

Creating the database channel

After the database channel has been defined it can be deployed to users through M-Business Server's Web-based Administration console. The Administration console binds back-end data retrieved from the XML data source to a named database on the mobile device, and associates that data with the appropriate schema that describes the database.

The process for creating a database channel is similar to the process for creating any other type of group channel for M-Business Server.

In order to create a database channel you (or an M-Business Server System Administrator) must:

Set up a group, or use an existing group of M-Business Anywhere users who will synchronize your database channel — database channels can only be defined for groups. See “Managing groups” [*M-Business Anywhere Administrator Guide*].

♦ Set up a URL for the data. For example, *http://intranet.somewhere.com/catalog.xml*

> **Note**
> URLs entered in any M-Business Anywhere component must contain only forward slashes (/). The backslash character (\) in place of a forward slash will cause an error. When you are setting up directories in a Windows file system, where backslashes separate levels, be sure to change these characters to forward slashes if you copy parts of the file system path into a URL.

♦ Set up a URL for the data's XML Schema Definition For example, *http://intranet.somewhere.com/catalog.xsd*.

♦ Established a name for the database containing the data.

> **Note**
> Use ALL CAPS to ensure maximum compatibility with future devices and releases of M-Business Anywhere.

♦ Create one or more HTML pages to present the data.

For example, *http://intranet.somewhere.com/catalog.html*.

For more information about creating database channels, see “Creating custom database channels” [*M-Business Anywhere Administrator Guide*].

Performing incremental synchronizations

To reduce the time required to synchronize your database channel, you can specify that only incremental changes in the data be downloaded in a synchronization.

> **Caution**
> When on-device data is being updated incrementally, if you change that database schema you will need to ensure that all users perform a Refresh All synchronization to ensure that they get the correct dataset.

When you (or an M-Business Server system administrator) are creating the database channel, select the checkbox for **Fetch Only Modifications - Incremental Sync**.

> **Note**
> When the checkbox for Fetch Only Modifications - Incremental Sync is selected, the **Delete Type** and **Delete URL** fields are available. You use these fields to specify how the incremental synchronization is to work.

> **Note**
> URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error.

Delete Type field

Use the Delete Type field to specify what type of data you will provide in the Delete URL:

- **Implicit** — the Delete URL will provide a list of records that are to remain in the database; any records that do *not* appear in this list are to be deleted.
- **Explicit** — the Delete URL will provide a list of records that are to be deleted from the database; any records that *do* appear in this list are to be deleted.

Delete URL field

Use the Delete URL field to specify the URL that will provide the list of records that determines which records are to be deleted. The Delete URL must return a list of records formatted similarly to the examples below. All of the following assume that you have sent down an entire database. For this example, we will use a very simple one:

```
<root>
<testCase date="20001215T12:15:36" string="1"/>
<testCase date="19041215T12:15:36" string="2"/>
<testCase date="19691231T23:59:59" string="3"/>
<testCase date="19700101T12:00:00" string="4"/>
</root>
```

Now the next time you synchronize you will get an `X-AvantGo-LastSyncTimestamp` header from M-Business Server that will tell you the last time a user synchronized the device. You can *add* two records to the device that have been added since then by supplying:

```
<root>
<testCase date="20031215T12:15:36" string="5"/>
<testCase date="20351215T12:15:36" string="6"/>
</root>
```

If you selected Explicit for Delete Type and you wanted to delete records 3 and 4, your Delete URL should return:

```
<root>
<testCase string="3"/>
```

```
<testCase string="4"/>
</root>
```

If you selected Implicit for Delete Type and you wanted to delete records 3 and 4, your Delete URL should return:

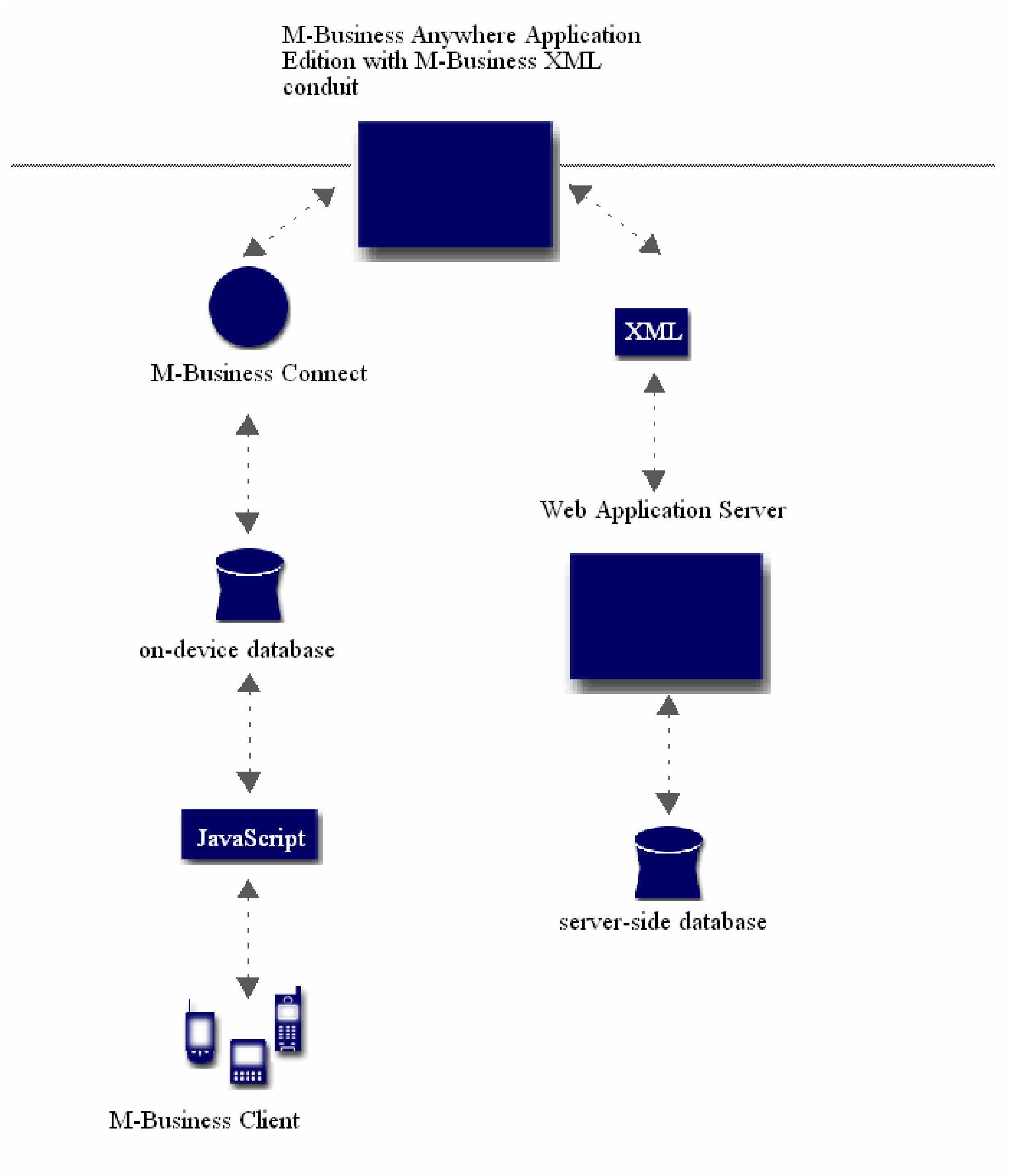
```
<root>
<testCase string="1"/>
<testCase string="2"/>
<testCase string="5"/>
<testCase string="6"/>
</root>
```

And the resulting dataset should be:

```
<root>
<testCase date="20001215T12:15:36" string="1"/>
<testCase date="19041215T12:15:36" string="2"/>
<testCase date="20031215T12:15:36" string="5"/>
<testCase date="20351215T12:15:36" string="6"/>
</root>
```

About M-Business XML API

If you purchased M-Business Anywhere Application Edition with M-Business XML conduit, you have access to the M-Business XML API, which enables you to present server-side database information on M-Business Client. The M-Business XML API works in conjunction with the M-Business XML conduit to send data to mobile devices and to report changes in the data back to the originating database server through M-Business Client's standard form submission mechanism. The M-Business XML API relies on XML-based standards, including XML-formatted DBMS response and W3C-approved XML, Schema 1.0-compliant schema documents for describing databases.



The client-side PODS component presents an on-device database API that allows you to manipulate the data that the server conduit delivers to the mobile device through a JavaScript or PODS interface. This API is derived from the Microsoft ActiveX Data Objects (ADO) API, and thus should be familiar to many application developers. For example, you could write a web page containing JavaScript that iterates a database containing products from a merchandise catalog. The script would dynamically generate a page on the device that lists the available products. This makes it much easier for you to write an application that maintains a separation between the application's data (residing in the device database), the application's presentation layer (HTML), and the application's business logic (JavaScript). See “M-Business XML API reference” [*M-Business Anywhere API Reference*].

CHAPTER 13

Building applications with M-Business XML conduit

Contents

Introduction to building applications

With the Application Edition of M-Business Anywhere, you can quickly develop dynamic, data-centric web applications. One of the most efficient ways of developing web applications is to keep application data separate from the presentation of the data. Separating presentation from data when you are developing web applications for connected desktop browsers, like Microsoft's Internet Explorer, allows you to present the application data in a variety of formats, for different form factors, and even multiple languages. When you develop web applications in this way for use on mobile devices — where a network connection is not always available — you can use web technologies to quickly create dynamic, data driven applications.

This chapter provides a step-by-step example for creating an application using the M-Business XML conduit and M-Business XML API. In this example, the data resides in a SQL Server 2000 database; the XML and web pages are generated using ASP.NET. You also could use MySQL, Apache and Perl, a J2EE server and Oracle or any setup that allows you to generate web pages and XML from a data source. The instructions included in this section assume that you already have installed M-Business Anywhere Application Edition.

Designing an application

The example is focused on a simple company employee directory. The data follows the following schema: last name, first name, phone number, ID.

This sample application will provide the following functionality:

- View a list of all employees
- View and update an employee's information
- Add a new employee

Although you could create this simple application using offline web pages, in doing so, you would encounter a few limitations. First, in order to display all of the application data, you would need to create three basic pages:

- A list page that displays all the employees names
- An update and detail page for each employee
- An add new user page

Another thing to consider is that because the server generates the update/detail pages, you will synchronize 102 web pages to the device if you have 100 employee records in the directory. In addition, because the update/detail pages are generated by the server, none of the changes made to the update/detail page are be reflected in the list page until the changes have been synchronized with the server.

Although the same application created using DHTML and the M-Business XML conduit contains the same basic HTML pages listed above, you will implement them differently.

When using the M-Business XML conduit you will generate three types of files from the web application server:

- HTML pages that will access and format the data on the device.
- XSD pages used by the M-Business XML conduit to process the XML file that contains the application data. This is the schema definition of the XML data.
- XML pages that contain the data that is synchronized down to the device by the M-Business XML conduit and M-Business Server. There is a one-to-one relationship between XSD schema pages and XML pages.

Building an application using a database channel

The first step to creating the application is to set up the data and the schema files that correspond to the data. In this section, we will create the application and the XSD and XML format and function.

Creating the data source

Again, the database will have a very simple schema. The schema will contain:

- A unique identifier
- A Last name
- A First name
- A Phone number

The unique identifier will serve as the primary key for the database queries. Below is an example of this schema from SQL Server 2000.

SQL Server 2000 database schema

```
CREATE TABLE [dbo].[Employees] (
[id] [int] IDENTITY (1, 1) NOT NULL,
[lastname] [varchar] (50) COLLATE
    SQL_Latin1_General_CP1_CI_AS NOT NULL,
[firstname] [varchar] (50) COLLATE
    SQL_Latin1_General_CP1_CI_AS NULL,
[phonenumber] [varchar] (50) COLLATE
    SQL_Latin1_General_CP1_CI_AS NULL
) ON [PRIMARY]
```

Creating the XSD file

When you have a simple, non-changing schema, as in this example, you can create the schema once and vend it to the M-Business XML conduit from a static file. When you are creating a more complex application, you might generate the XSD file dynamically from the back-end data source. The sample code below shows the XSD file for this application. The table below that lists the data types that are supported in XSD files.

Sample XSD file

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:dt="urn:schemas-microsoft-com:datatypes"
    xmlns:msch="urn:schemas-microsoft-com:mapping-schema">
        <xsd:element name="Employees" msch:relation="Employees"
        type="Employees_type" />
        <xsd:complexType name="Employees_type">
            <xsd:attribute name="id" type="xsd:int" />
            <xsd:attribute name="lastname" type="xsd:string" />
            <xsd:attribute name="firstname" type="xsd:string" />
            <xsd:attribute name="phonenumber" type="xsd:string" />
                <xsd:key name="PrimaryKey">
                    <xsd:selector xpath="." />
                    <xsd:field xpath="@id" />
                </xsd:key>
        </xsd:complexType>
</xsd:schema>
```

Note
You can find the sample code shown in the figure above in the *employees.xsd* file, which is available in *DB_sample.zip* file on the M-Business Anywhere Product Manuals page, referenced in "Related publications" on page xi.

Table 1. Supported data types in XSD files

Data type	XSD type specification
64-bit unsigned integer	`type = "xsd:double"`
32-bit signed integer	`type = "xsd:int"`
32-bit unsigned integer	`type = "xsd:unsignedInt"`
16-bit signed integer	`type = "xsd:short"`
16-bit unsigned integer	`type = "xsd:unsignedShort"`
character string (32kB maximum size)	`type = "xsd:string"`
boolean value (valid values: `"true"`, `"false"`)	`type = "xsd:boolean"`
date/time value (supports dates from January 1, 1902, to January 1, 2038 — format is: `YYYY:MM:DD:T:hh:mm:ss`)	`type = "xsd:datetime"`

Supporting double-byte character set (DBCS) data

The sample XSD in "Creating the XSD file" on page 276 does not specify a character set. This works fine for English and most western European languages. If your data uses a double-byte character set, or a special character set for Eastern European languages, you must include a character set specification in the definition of each XSD data element. You do this by including a `deviceEncoding` attribute in the element definition. In our example in "Creating the XSD file" on page 276, this would change the `Employees` element definition as follows — added text is bolded:

```
<xsd:element name="Employees"
    msch:relation="Employees"
    type="Employees_type"
    deviceEncoding="UTF-8"/>
```

Note
M-Business XML conduit's DBCS support is currently limited to the UTF-8 character set. For more general information about handling XML data that involves DBCS double-byte character sets, see "Avoiding problems with double-byte characters in XML data" on page 64.

Note
When you add the appropriate `deviceEncoding` attribute to each element definition, double-byte character data is fully supported. However, you should avoid using double-byte characters in element names, or elsewhere outside the data itself, if you plan to use the M-Business XML API. Some functions and methods may fail when double-byte character labels are encountered outside of the data.

Generating the XML file

There are many ways you could dynamically generate an XML file from a database. The following example was created using C# code running in IIS, connecting to a local SQL Server 2000 database called `agdb`, using a user called `web`, and retrieving data from a table called `Employees`.

C# sample code to generate XML file from back-end data

```
SqlConnection myConnection =
    new SqlConnection("Initial Catalog=agdb;" +
    "Data Source=localhost;"
    + "user id=web; password=web" );
         SqlCommand myCommand =
            new SqlCommand(
                "SELECT id, lastname, firstname,
                phonenumber " + "FROM Employees",
                myConnection); myConnection.Open();

                // read the data
                SqlDataReader dr = myCommand.ExecuteReader();

                // write out the header for the xml response
                Response.Output.WriteLine("<?xml version=\"1.0\"
                    " + "encoding=\"utf-8\" ?>");
                Response.Output.WriteLine("<root>");
                // write out xml for data from database
                while (dr.Read())
                {
                    Response.Output.WriteLine(
                        "<Employees id=\"" +
                            dr.GetInt32(0).ToString() + "\"
                            lastname=\"" + dr.GetString(1) + "\"
                            firstname=\"" + dr.GetString(2) + "\"
                            phonenumber=\"" + dr.GetString(3) + "\" />");
                }
                Response.Output.WriteLine("</root>");

                // close the connection to the database
                myConnection.Close();
```

Note
You can find the sample code shown in the figure above in the *list.aspx.cs* file, which is available in *DB_sample.zip* file on the M-Business Anywhere Product Manuals page, referenced in "Related publications" on page xi.

Creating a group for the database channel

Database channels are group channels, so you must first define a group — or select an existing group — for which you will create the database channel.

For instructions on listing existing groups and their members, see "Listing groups" [*M-Business Anywhere Administrator Guide*].

For instructions on creating a group, see "Creating a group" [*M-Business Anywhere Administrator Guide*].

Adding users to the group

If you have created a new group, you will need to add all the users who are to receive your database channel. If you are using an existing group, you may need to add users.

For instructions on adding users to a group, see "Adding users to a group" [*M-Business Anywhere Administrator Guide*].

Note
For testing purposes, you need only a single user in the group. You can add the rest of the users later, after you finish testing.

Enabling the download of the M-Business XML POD (database POD) to the group

In order for a database channel to work properly, each user's mobile device must first have a platform-specific database POD downloaded. You do this by creating a group channel, for the same group that is to receive the database channel.

For general instructions on creating a group channel, see "Creating a group channel" [*M-Business Anywhere Administrator Guide*]. Use the information below to fill in the group channel creation form as indicated for each platform.

Note
If you are supporting both Microsoft OS devices and Palm OS devices, you will need to create two separate group channels.

Note
URLs entered in any M-Business Anywhere component must contain only forward slashes (`/`). The backslash character (`\`) in place of a forward slash will cause an error. When you are setting up directories in a Windows file system, where backslashes separate levels, be sure to change these characters to forward slashes if you copy parts of the file system path into a URL.

For Microsoft OS devices

- **Title:** dbpod
- **Location:** http://localhost:8091/admin/install/AG_DEVICEOS/AG_DEVICEPROCESSOR/dbpod_quad.dll

  The hostname and port number (localhost:8091 above) should be the same as the `UIPort` and `UIHost` variables in your M-Business Server's *defaults* file. For more information, see "Settings in the defaults file" [*M-Business Anywhere Administrator Guide*].
- **Channel Size Limit:** 400 (KB)
- **Include Images:** UNchecked
- **Color Depth:** Automatic
- **Allow Binary Distribution:** CHECKED
- **Hide From Users:** CHECKED
- Leave the default settings for the other entries, especially, **Refresh:** Every sync

For Palm OS devices

- **Title:** dbpod_palm
- **Location:** http://localhost:8091/admin/install/AG_DEVICEOS/dbpod_quad.prc

  The hostname and port number (localhost:8091 above) should be the same as the `UIPort` and `UIHost` variables in your M-Business Server's *defaults* file. For more information, see "Settings in the defaults file" [*M-Business Anywhere Administrator Guide*].
- All other entries: same as for Microsoft OS devices above

The next time that any member of your database group synchronizes, the M-Business XML POD will be downloaded to their device. To verify the database POD is present on a device, see "Verifying download of the M-Business XML POD (database POD) to a device or emulator" on page 281.

Creating a sample database channel

You must create a database channel to configure M-Business XML conduit to distribute the database content to users.

For general instructions, see “Setting up the sample database channel” [*M-Business Anywhere Administrator Guide*]. Use the information below to fill in the database channel creation form for the sample database channel — if you are setting up your own database channel, enter values for Data URL and Schema URL that reflects the locations and names of your files:

- **Name:** avgoEmpDir
- **Data URL:** http://localhost/EmpDir/employees.xsd

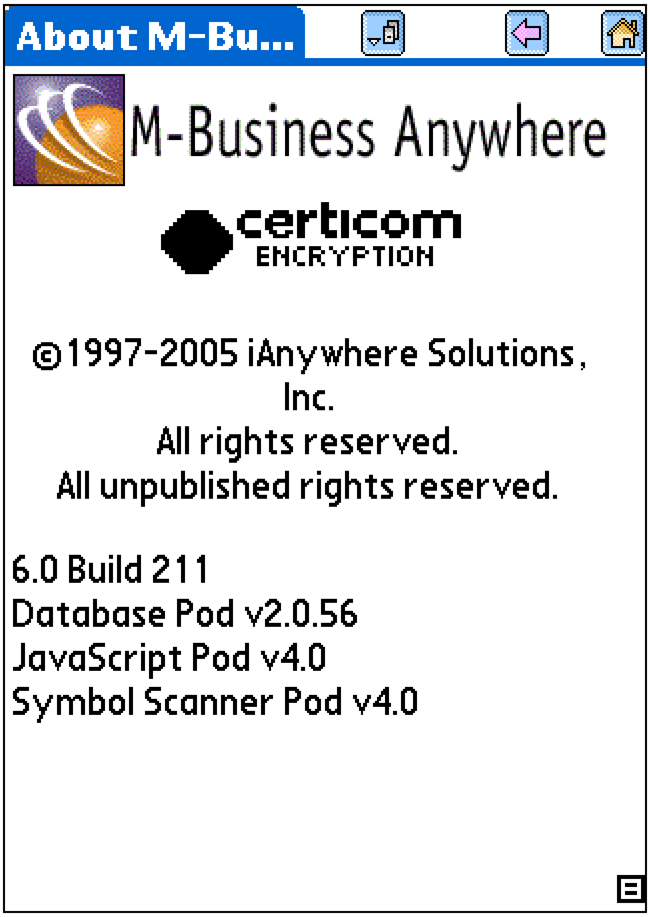
> **Note**
> URLs entered in any M-Business Anywhere component must contain only forward slashes (/). The backslash character (\) in place of a forward slash will cause an error.

- **Schema URL:** http://localhost/EmpDir/list.aspx.cs
- **Format:** Attribute-based
- Leave the default settings for the other entries.

The steps you have completed thus far have set up everything so that each time a member of the database channel group synchronizes their mobile device, they will get the data retrieved from the database. However, in order for users to display, edit and update that data you will need to create web pages that synchronizes down to the device through another group channel, which will access the data using JavaScript that accesses the downloaded M-Business XML POD. See “Displaying the data in HTML” on page 283.

Verifying download of the M-Business XML POD (database POD) to a device or emulator

Before you proceed with testing, make sure that the M-Business XML POD is installed on the device (or emulator) on which you will develop your application. With M-Business Client running on the device (or emulator), choose **Options»About M-Business**. If the M-Business XML POD is installed, it will be listed on this screen, as shown in the figure below, listed as "Database Pod".



©1997-2005 iAnywhere Solutions, Inc.
All rights reserved.
All unpublished rights reserved.

6.0 Build 211
Database Pod v2.0.56
JavaScript Pod v4.0
Symbol Scanner Pod v4.0

Displaying the data in HTML

As you may recall from "Designing an application" on page 275, generating static web pages would require 102 on-device pages to display the data for a 100-person employee directory. Because the data for the employee list now resides in an on-device database, you only need to create three pages:

- A list page
- A detail template page
- A new employee form

Creating the list page

The HTML for creating the list page requires the following simple elements:

- A heading that reads Employee Directory
- An `onLoad` function in the body tag called `buildEmployeeList`.
- A `span` element with an id of `listSpan`.

> **Note**
> You can find the HTML for the sample list page shown in "Verifying download of the M-Business XML POD (database POD) to a device or emulator" on page 281 in the file *index.html*, which is available in *DB_sample.zip* file on the M-Business Anywhere Product Manuals page, referenced in "Related publications" on page xi.

The `onLoad` function `buildEmployeeList` builds an ordered list that contains the data from the avgoEmpDir database.

♦ To create the list page

1. Initialize the M-Business XML POD.
2. Open the avgoEmpDir database with the read flag set.
3. Check for error.

   Code for steps 1 - 3:

   ```
   dbMgr = CreateObject("avantgo.db");
   dbSet = dbMgr.open("avgoEmpDir", "r");
       if(!dbSet) {
            //return an error
   }
   ```

4. Find the span element using the built-in `document` object in JavaScript.
5. Create an ordered list <ol> element and make it a child of the span element.

Code for steps 4 and 5:

```
span = document.getElementById("listSpan");
list = document.createElement("ol");
span.appendChild(list);
```

6. Loop until the end of the data set is reached.
7. Create a list element and make it a child of the list
8. Create an anchor element.
9. Set the id of the anchor tag to the id of the record, since it is the primary key to retrieve database record. This passes the id into the onclick event.
10. Set the onclick event to the JavaScript function switchToDetail.
11. Create a string comprised of the first name, last name and phone number, then make the string a child of the anchor element.
12. Make the anchor element a child of the list element.
13. Move the data set pointer to the next record.

Code for steps 6 - 13:

```
while(!dbSet.ateof()) {
    var linktext;
    listelement = document.createElement("li");
    list.appendChild(listelement);

    anchor = document.createElement("a");
    anchor.id = dbSet.id;
    anchor.onclick = switchToDetail;

    var description = dbSet.firstname + " "
        + dbSet.lastname + " "
        + dbSet.phonenumber

    linktext = document.createTextNode(description);

    anchor.appendChild(linktext);
    listelement.appendChild(anchor);

    dbSet.moveNext();
}
```

The `switchToDetail` function shown above, creates a list where each list element is comprised of the data from a row in the avgoEmpDir data set. The row data is represented as a link. When clicked, each link calls the `switchToDetail` JavaScript function, to be discussed in more detail later. First, however, it is important to understand how information is passed from one page to another in offline mobile applications.

The switchToDetail function

```
function switchToDetail() {
    // the id of the anchor has the record id
    PropertiesDBSetValue("sessiondb", "currentid", this.id);

    // send the broswer to the detail page
    window.location = "/EmpDir/update_employee.html";
}
```

You can use the M-Business XML POD to save the state between pages. In the Employee Directory example, you can save the key/value pair that tells the detail page which employee information to display. The `switchToDetail` function shown in the figure above, shows an example of saving the value of the anchor's ID (which was set to the record id of the employee) in an on-device database called *sessiondb* and associating it with the key `currentid.` After determining which employee's information you want to display, you call `window.location` to switch M-Business Client to that detail page.

The code for the `PropertiesDBSetValue` is included with the code for this example.

Creating the detail template page

When processing data through a web server while connected to the Internet you can use one of two options:

- Create a link to the detail page and pass data in the URL.

  This is how the static web page solution works:

  The links to the detail page are in this form:

  /EmpDir/detailpage.html?id=1

  The detail page then reads the record id from the URL string, retrieves the rest of the employee's information from the database, then creates the page and sends it down to the device.

- Use a cookie to keep state between pages.

  When connected to the Internet, you can use the `switchToDetail` function to set a cookie, then send the browser to a detail page. The detail page would then read the cookie and look up the employee information from the cookie.

Note
You can see an example of this in the sample code files *static_list.aspx*, *static_update.aspx* in the *DB_Sample.zip* file.

Creating the new employee form

The detail page that is created from the code in the previous example essentially is a form whose field values reflect the current values. If you were working with data of a more complex nature than the simple employee directory example used here, you may want to have separate pages for view and edit functions.

Below is the form for this simple employee directory example.

Example code for new employee form

```
<body onLoad="buildEmployeeList()">
    <h3>Employee Directory</h3>
    <!-- this span will give us a placeholder to dynamically
        create the employee list -->
```

```
      <span id="listSpan"></span> \
      <hr>
      <a href="./newemp.html">Add new employee</a>
      <!-- put a hidden link to the update page so it gets downloaded -->
      <a href="./update_employee.html"></a>
  </body>
```

The JavaScript `onLoad` event in the body tag calls the `populateForm()` function, which loads the correct information from the database into the form fields. The `onClick` function calls the `doForm()` function, which saves the data in the database and prepares it for sending to the server.

There is no `action` attribute in the `form` element. Because the above example assumes an offline status, all of the form processing is done in JavaScript by the page itself. Although it is possible to write a POD to handle the form processing and call the POD from the action URL, it is outside the scope of this example. See "Writing C Code for PODS" [*M-Business Anywhere API Reference*].

♦ To create the new employee form

1. Read the value of the stored database key and use that value to look up the rest of the row values.

   Code for step 1:

   ```
   lastName = PropertiesDBGetValue("sessiondb", currentid");
   if (!lastName) {
       //handle error here
       return;
    }
   ```

2. Initialize the M-Business XML POD.

3. Open the avgoEmpDir database.

4. Reset the record pointer to the beginning.

   Code for steps 2 - 4:

   ```
   dbMgr = CreateObject("avantgo.db");
   dbSet = dbMgr.open("avgoEmpDir", "r");
   dbSet.moveFirst();
   ```

5. Search the database for the record you want.

6. If the record isn't found, display an error, then go back to the list page.

   Code for steps 5 and 6:

   ```
   var search = dbSet.createSearch("id = " + empId);
       if(!dbSet.find(search)) {
           alert("Could find record " + empId);
           window.location = "index.html";
           return;
       }
   ```

7. Set the value of the form elements using the DOM.

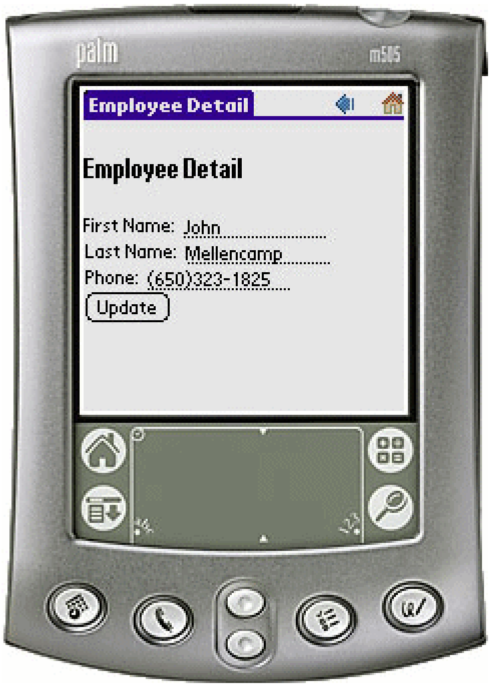
8. Close the data set.

   Code for steps 7 and 8:

```
document.updateForm.firstName.value = dbSet.firstname;
document.updateForm.lastName.value = dbSet.lastname;
document.updateForm.phoneNumber.value = dbSet.phonenumber;

dbSet.close();
```

Sending updates to the back-end database

Now that the page has loaded and the form elements have been populated, the page displayed should look something like the page shown in the figure below.



Employee Detail

Employee Detail

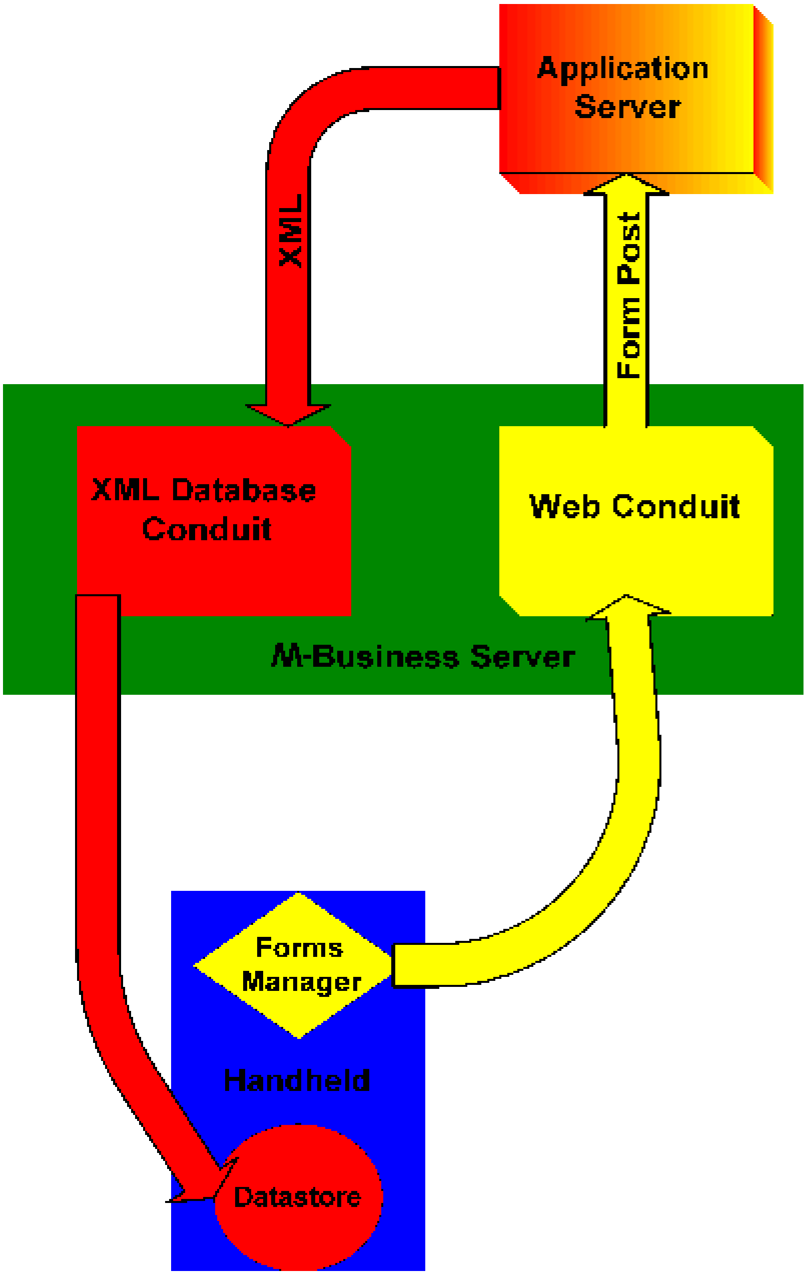
First Name: John

Last Name: Mellencamp

Phone: (650)323-1825

Because the M-Business XML conduit is a one-way synchronization, meaning that it does not send the changed data back to the database, you will need to use form posts for updating or adding new records to the back-end database. As a result, the form data needs to be saved in two places:

- In the on-device database as an update.
- In the Forms Manager stored as a form post.



The flow diagram in the figure above, illustrates the process of moving data in M-Business Anywhere.

First, the Forms Manager sends form posts to the M-Business web conduit, then the web conduit submits form posts to the application server. Next, the data from the application is sent to the M-Business XML conduit via an XML form post. Finally, the M-Business XML conduit synchronizes down the data to the on-device database.

Since the form posts are processed by the web conduit before the data is retrieved from the application server by the M-Business XML conduit, a new or updated record is processed correctly.

Writing data to on-device database

Saving the data from the form to the on-device database is very simple. Use the same key value pair to look up the correct record, change the values, commit, and close.

Writing data to on-device database

```
dbMgr = CreateObject("avantgo.db");
    dbSet = dbMgr.open("avgoEmpDir", "w");
    var search = dbSet.createSearch("id = " + empId);
        if(!dbSet.find(search)) {
            alert("Could find record " + empId);
            window.location = "index.html";
            dbSet.close();
            return;
        }

        dbSet.lastname = document.updateForm.lastName.value;
        dbSet.firstname = document.updateForm.firstName.value;
        dbSet.phonenumber = document.updateForm.phoneNumber.value;

        dbSet.commit();
        dbSet.close();
```

Saving data in Forms Manager as a form post

The data is written to the Forms Manager using the `avantgo.submissionManager` object. See “PODSSubmissionMgr object” [*M-Business Anywhere API Reference*].

♦ To save data in Forms Manager as a form post

1. Access the `submissionManager`.
2. Look through the forms manager for a previous submission.
3. Test the form title to is if it is the correct form.
4. If there is a previous form for this record, update the submission and return.

   Code for steps 1 - 4:

```
var subMgr = avantgo.submissionManager;

// First look to see if we have edited this
// user's info before
var cnt = subMgr.length;
while(cnt--) {
    submission = subMgr[cnt];

    // Handle the deleted submissions
    if(submission == null)
        continue;
    // this is not the one we are looking for
    if((submission.title != document.updateForm.empId.value))
        continue;

    element =
        subMgr.submissionElementForName("lastName");
    element.value =
        document.updateForm.lastName.value;
    element =
        subMgr.submissionElementForName("firstName");
    element.value =
        document.updateForm.lastName.value;
    element =
        subMgr.submissionElementForName("phoneNumber");
    element.value =
        document.updateForm.lastName.value;

    return;
}
```

5. Create a new form submission.

6. Set the form element name/value pair into the submission.

7. Set the submission attributes, hiding the form so the user cannot edit it from the Forms Manager (they can simply edit the form again), then trash the response.

8. Set the title of the submission as the ID of the database record so that you can uniquely identify the submission, preventing multiple submissions for an individual record.

9. Return to the previous page.

   Code for steps 5 - 9:

```
subm = subMgr.createSubmission(0, new Date(), 0, 0, 0, 0, 0, 0, 0);

    subm.appendSubmissionElement(
    subm.createSubmissionElement("empId",
document.updateForm.empId.value));

    subm.appendSubmissionElement(
    subm.createSubmissionElement("lastName",
        document.updateForm.lastName.value));

    subm.appendSubmissionElement(
    subm.createSubmissionElement("firstName",
        document.updateForm.firstName.value));

    subm.appendSubmissionElement(
    subm.createSubmissionElement("phoneNumber",
        document.updateForm.phoneNumber.value));

    subm.status = "UNSUBMITTED_SUBMIT_STATUS";
    subm.isHidden = true;
    subm.trashResponse = true;
    subm.formIndex = 0;
    subm.actionMethod = "P";
    subm.title = document.updateForm.empId.value;
    subm.sourceURL = "/EmpDir/update_employee.html";
    subm.actionURL = "/EmpDir/update_emp.aspx";
    subMgr.saveSubmission(subm);
```

```
        back();
```

For more information on using the M-Business XML API, see “M-Business XML API reference” [*M-Business Anywhere API Reference*].

Part V. Appendices

Appendix A

HTML 4 support in M-Business Client 5.x and later

Contents

Overview of HTML 4 support

M-Business Client 5.x and later supports a major portion, but not all, of HTML 4. As you are developing web pages for devices with M-Business Client 5.x and later, consult this reference to be sure that the cool features of dynamic HTML you are planning to use *are* supported.

This appendix only *lists* features of the HTML 4 specification, indicating which are supported by M-Business Client 5.x and later. For information on how to use any of these features, see one of your favorite online references, such as:

- World Wide Web Consortium, at http://www.w3.org/
- W3 Schools, at http://www.w3schools.com/
- Web Monkey, at http://webmonkey.com/

HTML 4.01 tags and attributes

This section lists the HTML 4.01 tags and attributes from the W3C spec, indicating those that are supported by M-Business Client. This table includes CSS attributes.

Supported and non-supported features are indicated as follows:

- **Bold** text (e.g., **<a>** tag): HTML 4.01 feature supported by M-Business Client
- `Plain` text (e.g., `Anchor` object): HTML 4.01 feature *not* supported by M-Business Client

Note
Where M-Business Client does not support a tag from the HTML 4.01 spec, "HTML 4.01 tags and attributes" on page 297does not list the individual attributes and event handlers for that tag.

Note
Some features of HTML 4 related to fonts, which are listed as supported by M-Business Client in "HTML 4.01 tags and attributes" on page 297, are not supported by certain hardware platforms. In general, the Palm°OS platform supports a more limited set of font options.

Note
M-Business Client support for Cascading Style Sheets (CSS 1.0) is limited to `style` attribute properties. External and inline style sheets are not supported in M-Business Client 5.x and later. For a list of `style` attribute properties supported, see "CSS level 1 style attribute properties" on page 310.

Note
M-Business Client support for the `<a>` tag's `href` attribute has been expanded to include `phoneto:` — `phoneto:` can be used in place of `http:` or `mailto:`, as in this example:

```
<a°href="phoneto:19252364802">Phone me!</a>
```

When the user taps or clicks a `phoneto:` link, M-Business Client launches the appropriate dialer application with the phone number filled in. The user then taps or clicks a Dial or Call button or icon to initiate the call. If the device does not have a dialer application, an error message appears.

Table 1. Supported HTML 4.01 tags and attributes

HTML tag	Attributes or attribute settings	Event handlers
`<!-->`	N/A	N/A
`<!doctype>`	N/A	N/A

HTML tag	Attributes or attribute settings	Event handlers
<a>	accesskey **charset** coords **href*** **hreflang** **name** shape target type **rel** **rev** **tabindex** * see Note on phoneto: before this table plus "Core and language attributes" on page 308	**onblur** **onfocus** plus "Mouse and key event attributes" on page 309
<abbr>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<acronym>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<address>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<applet>	Tag deprecated. Use <object>.	
<area>	**accesskey** **alt** **coords** **href** **nohref** **shape** **tabindex** **target** plus "Core and language attributes" on page 308	**onblur** onfocus plus "Mouse and key event attributes" on page 309
<b>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<base>	**href** target	N/A
<basefont>	Tag deprecated. Use "CSS level 1 style attribute properties" on page 310	

HTML tag	Attributes or attribute settings	Event handlers
<bdo>	—	—
<big>	“Core and language attributes” on page 308	“Mouse and key event attributes” on page 309
<blockquote>	**cite** plus “Core and language attributes” on page 308	“Mouse and key event attributes” on page 309
<body>	alink* background* **bgcolor*** link* **text*** vlink* * deprecated plus “Core and language attributes” on page 308	**onload** **onunload** plus “Mouse and key event attributes” on page 309
<br>	class **clear** **id** **style** **title**	N/A
<button>	accesskey **disabled** **name** **tabindex** **type** **value** plus “Core and language attributes” on page 308	onblur onfocus plus “Mouse and key event attributes” on page 309
<caption>	**align*** * deprecated plus “Core and language attributes” on page 308	“Mouse and key event attributes” on page 309
<center> Tag deprecated. Use “CSS level 1 style attribute properties” on page 310.	“Core and language attributes” on page 308	“Mouse and key event attributes” on page 309
<cite>	“Core and language attributes” on page 308	“Mouse and key event attributes” on page 309

HTML tag	Attributes or attribute settings	Event handlers
<code>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<col>	—	—
<colgroup>	—	—
<dd>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<del>	cite datetime plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<dfn>	—	—
<dir> Tag deprecated. Use <ul>.	**compact** plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<div>	**align*** * except align=justify plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<dl>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<dt>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<em>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<fieldset>	—	—
<font> Tag deprecated. Use "CSS level 1 style attribute properties" on page 310.	**color** **face** size plus "Core and language attributes" on page 308	N/A

HTML tag	Attributes or attribute settings	Event handlers
<form>	**accept** **accept-charset** **action** **enctype** **method** **name** plus "Core and language attributes" on page 308	**onreset** **onsubmit** plus "Mouse and key event attributes" on page 309
<frame>	—	—
<frameset>	—	—
<h#> (<h1>...<h6>)	**align** plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<head>	dir **lang** **profile**	N/A
<hr>	**align*** **noshade*** **size*** width* * deprecated, use "CSS level 1 style attribute properties" on page 310 plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<html>	version dir **lang**	N/A
<i>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<iframe>	—	—

HTML tag	Attributes or attribute settings	Event handlers
<img> **Note:** Image transparency is not supported.	**align*** **alt** **border*** **height**** **hspace** **ismap** longdesc name src **usemap** **vspace** **width**** * deprecated ** height and width supported only to scale image *downward*; and image will never be larger than device screen plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<input>	**accept** accesskey **align** **alt** **checked** **disabled** ismap **maxlength** **name** **size** **src** **readonly** **tabindex** **type*** **usemap** **value** * except type=file) plus "Core and language attributes" on page 308	**onblur** **onchange** **onfocus** **onselect** plus "Mouse and key event attributes" on page 309
<ins>	cite datetime plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<isindex>	Tag deprecated. Use <input>.	
<kbd>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309

HTML tag	Attributes or attribute settings	Event handlers
<label>	—	—
<legend>	—	—
<li>	**compact*** start* **type*** **value*** * deprecated plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<link>	—	—
<listing>	Tag obsolete. Use <pre>.	
<map>	**name** plus "Core and language attributes" on page 308	onfocus onblur "Mouse and key event attributes" on page 309
<menu> Tag deprecated. Use <ul>.	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<meta>	**content** **http-equiv** **name** scheme	dir **lang**
<noframes>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<noscript>	—	—

HTML tag	Attributes or attribute settings	Event handlers
<object>	**align** **archive** **border** classid **codebase** **codetype** **data** declare **height** hspace **name** **standby** **tabindex** **type** **usemap** vspace **width** plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<ol>	**compact*** **start*** **type*** * deprecated plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<optgroup>	—	—
<option>	disabled **label** **selected** **value** "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<p>	**align** plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<param>	**id** **name** **type** **value** **valuetype**	N/A
<plaintext>	Tag obsolete. Use <pre>.	

HTML tag	Attributes or attribute settings	Event handlers
<pre>	**width*** * deprecated plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<q>	—	—
<s> Tag deprecated. Use <del>.	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<samp>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<script>	**charset** **defer** **language*** **src** **type** * deprecated	N/A
<select>	**disabled** **multiple** **name** **size** **tabindex** plus "Core and language attributes" on page 308	**onblur** **onchange** **onfocus** plus "Mouse and key event attributes" on page 309
<small>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<span>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<strike> Tag deprecated. Use <del>.	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<strong>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<style>	—	—
<sub>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309

HTML tag	Attributes or attribute settings	Event handlers
<sup>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<table>	**align** **bgcolor** **border** **cellpadding** **cellspacing** frame **rules** summary **width** plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<tbody>	**align*** char charoff **valign** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<td>	abbr **align*** axis **bgcolor** char charoff **colspan** **height** headers **nowrap** **rowspan** scope **valign** **width** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309

HTML tag	Attributes or attribute settings	Event handlers
<textarea>	accesskey **cols** **disabled** **name** **readonly** **rows** **tabindex** plus "Core and language attributes" on page 308	**onblur** **onchange** **onfocus** **onselect** plus "Mouse and key event attributes" on page 309
<tfoot>	**align*** char charoff **valign** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<th>	abbr **align*** axis **bgcolor** char charoff **colspan** **height** headers **nowrap** **rowspan** scope **valign** **width** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<thead>	**align*** char charoff **valign** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309

HTML tag	Attributes or attribute settings	Event handlers
<title>	dir **lang**	N/A
<tr>	**align**** **bgcolor** char charoff **valign** * except align=justify and align=char plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<tt>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<u> Tag deprecated. Use "CSS level 1 style attribute properties" on page 310.	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<ul>	**compact*** **type*** * deprectated plus "Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<var>	"Core and language attributes" on page 308	"Mouse and key event attributes" on page 309
<xmp>	Tag obsolete. Use <pre>.	

Core and language attributes

In "HTML 4.01 tags and attributes" on page 297, the label core and language attributes refers to the following HTML 4.01 attributes:

- class
- dir
- **id**
- **lang**

- **style**
- **title**

Mouse and key event attributes

In “HTML 4.01 tags and attributes” on page 297, the label mouse and key event attributes refers to the following HTML 4.01 attributes:

- **onclick**
- ondblclick
- **onkeydown** - except with form controls
- **onkeypress** - except with form controls
- **onkeyup** - except with form controls
- **onmousedown** - except with form controls
- **onmousemove** - except with form controls
- **onmouseout** - except with form controls
- onmouseover
- **onmouseup** - except with form controls

CSS level 1 style attribute properties

Note
M-Business Client support for Cascading Style Sheets (CSS 1.0) is limited to `style` attribute properties. External and inline style sheets are not supported in this release.

This section details the CSS level 1 `style` attribute properties that are supported and not supported in M-Business Client.

Supported and non-supported properties are indicated as follows.

- **`Bold`** text (e.g., **`'background'`**): W3C CSS level 1 feature supported by M-Business Client
- `Plain` text (e.g., `'background-attachment'`): W3C CSS level 1 feature *not* supported by M-Business Client

Table 2. Supported properties of style attribute

CSS property group	Properties

Box properties	**`'border'`**
	`'border-bottom'`
	`'border-bottom-width'`
	`'border-color'`
	`'border-left'`
	`'border-left-width'`
	`'border-right'`
	`'border-right-width'`
	`'border-style'`
	`'border-top'`
	`'border-top-width'`
	`'border-width'`
	`'clear'`
	`'float'`
	`'height'`
	`'margin'`
	`'margin-bottom'`
	`'margin-left'`
	`'margin-right'`
	`'margin-top'`*
	`'padding'`
	`'padding-bottom'`
	`'padding-left'`
	`'padding-right'`
	`'padding-top'`
	`'width'`
	
	* values less than 1.0 must supply the leading zero (for example, 0.75, not .75)

CSS property group	Properties
Classification properties	**'display'*** **'list-style'** 'list-style-image' **'list-style-position'** **'list-style-type'** **'white-space'**** * does not work with `tbody`, `thead`, or `tfoot` ** no `'nowrap'`
Color and background properties	**'background'** 'background-attachment' **'background-color'** 'background-image' 'background-position' 'background-repeat' **'color'**
Font properties	**'font'** **'font-family'** **'font-size'** **'font-style'** **'font-variant'*** **'font-weight'** * no `small-caps`
Text properties	'letter-spacing' **'line-height'** **'text-align'** **'text-decoration'** 'text-indent' 'text-transform' **'vertical-align'** 'word-spacing'

DOM level 1

The table below details the DOM level 1 interface methods and attributes that are supported in M-Business Client. For documentation of the full M-Business DOM 1.0 implementation, see "ADOMDOMImplementation object" [*M-Business Anywhere API Reference*].

For each interface listed, supported and non-supported features are indicated as follows.

- **`Bold`** text (e.g., **`Attr`** interface): W3C DOM level 1 feature supported by M-Business Client
- `Plain` text (e.g., `CDATASection` interface): W3C DOM level 1 feature *not* supported by M-Business Client

> **Note**
> Where M-Business JavaScript engine does not support an interface from the DOM level 1 standard, the table below does not list the individual attributes and methods from that interface.

> **Note**
> iAnywhere Solutions has selectively implemented a small number of DOM level 2 features, plus a few features that are not included in DOM level 2. All these extensions to DOM level 1 are documented in "M-Business extensions to W3C DOM level 1" [*M-Business Anywhere API Reference*].

The JavaScript synopsis for each of the supported DOM level 1 features is detailed in "The W3C DOM spec and corresponding M-Business DOM calls" [*M-Business Anywhere API Reference*]

Table 3. Supported features from DOM level 1 spec

W3C DOM interface°	Attributes	Methods
`Attr`	**`name`** **`specified`** **`value`**	N/A
`CDATASection`	—	—

`CharacterData`	**`data`**
	`length`
		`appendData()`
		`deleteData()`
		`insertData`
		`replaceData`
		`substringData`

`Comment`	—	—

W3C DOM interface°	Attributes	Methods

Document	doctype
	documentElement
	implementation
		createAttribute()
		createElement()
		createCDATASection()
		createComment()
		createDocumentFragment()
		createEntityReference()
		createProcessingInstruction()
		createTextNode()
		getElementsByTagName()

DocumentFragment	—	—
DOMImplementation	N/A	**hasFeature()**
DocumentType	—	—

Element	**tagName**	**getAttribute()**
		getAttributeNode()
		getElementsByTagName()
		normalize()
		removeAttribute()
		removeAttributeNode()
		setAttribute()
		setAttributeNode()

Entity	—	—
EntityReference	—	—

HTMLAnchorElement	accesskey
	charset
	coords
	href
	hreflang
	name
	rel
	rev
	shape
	tabindex
	target
	type
		blur()
		focus()

HTMLAppletElement	—	—
HTMLAreaElement	accesskey **alt** **coords** **href** **noHref** **shape** **tabindex** **target**	N/A
HTMLBaseElement	—	—

W3C DOM interface°	Attributes	Methods
HTMLBaseFontElement	—	—
HTMLBlockquoteElement	Interface deprecated in favor of DOM level 2 HTMLQuoteElement	
HTMLBodyElement	**aLink** **background** **bgColor** **link** **text** **vLink**	N/A
HTMLBRElement	**clear**	N/A
HTMLButtonElement	accesskey **disabled** **form** **name** **tabindex** **type** **value**	N/A

HTMLCollection	**length**	**item()**
		namedItem()

HTMLDirectoryElement	**compact**	N/A
HTMLDivElement	**align**	N/A
HTMLDListElement	**compact**	N/A

HTMLDocument	**anchors**
	applets
	body
	close()
	domain
	forms
	images
	links
	open()
	referrer
	title
	URL
	write()
	writeln()
		getElementById()
		getElementsByName()

HTMLElement	**className** **dir** **id** **lang** **title**	N/A
HTMLFieldSetElement	—	—

W3C DOM interface°	Attributes	Methods
HTMLFontElement	**color** **face** **size**	N/A

HTMLFormElement	**acceptCharset**
	action
	elements
	enctype
	length
	method
	name
	target
		reset()
		submit()

HTMLFrameElement	—	—
HTMLFramesetElement	—	—
HTMLHeadElement	**profile**	N/A
HTMLHeadingElement	**align**	N/A
HTMLHRElement	**align** **noShade** **size** **width**	N/A
HTMLHtmlElement	**version**	N/A
HTMLIFrameElement	—	—
HTMLImageElement	**align** **alt** **border** **height** **hspace** **isMap** **longDesc** **lowSrc** **name** **src** **useMap** **vspace** **width**	N/A

W3C DOM interface°	Attributes	Methods

HTMLInputElement	**accept**
	accesskey
	align
	alt
	checked
	defaultChecked
	defaultValue
	disabled
	form
	maxLength
	name
	readOnly
	size
	src
	tabindex
	type
	useMap
	value
		blur()
		click()
		focus()
		select()

HTMLIsIndexElement	–	—
HTMLLabelElement	— accesskey **form** **htmlFor**	—
HTMLLegendElement	—	—
HTMLLIElement	**type** **value**	N/A
HTMLLinkElement	–	—
HTMLMapElement	**areas** **name**	N/A
HTMLMenuElement	**compact**	N/A
HTMLMetaElement	**content** **httpEquiv** **name** **scheme**	N/A
HTMLModElement	**cite** **dateTime**	N/A

W3C DOM interface°	Attributes	Methods
HTMLObjectElement	**align** **archive** **border** **code** **codeBase** **codeType** **data** **declare** **form** **height** **hspace** **name** **standby** **tabindex** **type** **useMap** **vspace** **width**	N/A
HTMLOListElement	**compact** **start** **type**	N/A
HTMLOptGroupElement interface (page 143)	—	—
HTMLOptionElement	**defaultSelected** **disabled** **form** **index** **label** **selected** **text** **value**	N/A
HTMLParagraphElement	**align**	N/A
HTMLParamElement	**name** **type** **value** **valueType**	N/A
HTMLPreElement	**width**	N/A
HTMLQuoteElement	**cite**	N/A
HTMLScriptElement	**charset** **defer** **event** **htmlFor** **src** **text** **type**	N/A

W3C DOM interface°	Attributes	Methods

HTMLSelectElement	disabled
	form
	length
	multiple
	name
	options
	selectedIndex
	size
	tabindex
	type
	value
		add()
		blur()
		focus()
		remove()

HTMLStyleElement	disabled media type	N/A
HTMLTableCaptionElement	align	N/A
HTMLTableCellElement	abbr align axis bgColor cellIndex ch chOff colSpan headers height noWrap rowSpan scope vAlign width	N/A
HTMLTableColElement	align ch chOff span vAlign width	N/A

HTMLTableElement	align
	bgColor
	border
	caption
	cellPadding
	cellSpacing
	frame
	rows
	rules
	summary
	tBodies
	tFoot
	tHead
	width
		createCaption()
		createTFoot()
		createTHead()
		deleteCaption()
		deleteRow()
		deleteTFoot()
		deleteTHead()
		insertRow()

W3C DOM interface°	Attributes	Methods
HTMLTableRowElement	align bgColor cells ch chOff deleteCell rowIndex sectionRowIndex vAlign	insertCell()

HTMLTableSectionElement	align
	ch
	chOff
	rows
	vAlign
		deleteRow()
		insertRow()

HTMLTextAreaElement	accesskey
	cols
	defaultValue
	disabled
	form
	name
	readOnly
	rows
	tabindex
	type
	value
		blur()
		focus()
		select()

HTMLTitleElement	text	N/A
HTMLUListElement	compact type	N/A
NamedNodeMap	—	—

Node	attributes
	childNodes
	firstChild
	lastChild
	nextSibling
	nodeName
	nodeType
	nodeValue
	ownerDocument
	parentNode
	previousSibling
		appendChild()
		cloneNode()
		getAttributes()
		hasChildNodes()
		insertBefore()
		removeChild()
		replaceChild()

NodeList	length	item()
Notation	—	—
ProcessingInstruction	—	—
Text	N/A	splitText()

JavaScript

The table below details the features from JavaScript standards that are supported and not supported in the current release of M-Business Client. For each object listed, supported and non-supported features are indicated as follows.

- **Bold** text (e.g., **Button** object): standard JavaScript feature supported by M-Business JavaScript engine
- Plain text (e.g., Anchor object): standard JavaScript feature *not* supported by M-Business JavaScript engine

Note
Where M-Business JavaScript engine does not support a standard JavaScript object, the table below does not list the individual properties, methods, and event handlers from that object.

The Standard column in the table below indicates the specific standard in which the M-Business Sync Server supported JavaScript object is defined:

- DOM 0 is client-side JavaScript specified in the standard called DOM Level 0 API.
- ECMA is core JavaScript specified in ECMA-262, 3rd Edition.
- W3C DOM is document object model features specified in the W3C DOM 1.0+ standards.

Table 4. Supported features from standard JavaScript

Object	Standard	Properties	Methods	Event Handlers
Anchor	DOM 0	—	—	—
Applet	DOM 0	—	—	—
Area	DOM 0	—	—	—
Arguments	ECMA	**callee** **length**	N/A	N/A

Object	Standard	Properties	Methods	Event Handlers
Array	ECMA	length prototype	concat join pop push reverse shift slice sort splice toLocaleString toString unshift	N/A
Attr	DOM 0	name specified value	N/A	N/A
Boolean	ECMA	N/A	toString valueOf	N/A
Button	DOM 0	form name type value	blur click focus	onblur onclick ondblclick onfocus onkeydown onkeypress onkeyup onmousedown onmouseup
Checkbox	DOM 0	checked defaultChecked form name type value	blur click focus	onblur onclick ondblclick onfocus onkeydown onkeypress onkeyup onmousedown onmouseup

Object	Standard	Properties	Methods	Event Handlers
Date	ECMA	N/A **Note:** in the Date methods to the right, [UTC] indicates that two methods exist, one for local time (for example, getDate) and one for UTC time (for example, getUTCDate).	**get[UTC]Date** **get[UTC]Day** **get[UTC]FullYear** **get[UTC]Hours** **get[UTC]Milliseconds** **get[UTC]Minutes** **get[UTC]Month** **get[UTC]Seconds** **getTime** **getTimeZoneOffset** **getYear** **set[UTC]Date** **set[UTC]FullYear** **set[UTC]Hours** **set[UTC]Milliseconds** **set[UTC]Minutes** **set[UTC]Month** **set[UTC]Seconds** **setTime** **setYear** toDateString **toGMTString** toLocaleDateString **toLocaleString** toLocaleTimeString **toString** toTimeString **toUTCString** **valueOf** **parse** **UTC**	N/A
document See “Example: setting forms property attributes ” on page 328 and “Example: working around innerText and document.write” on page 329	DOM 0	alinkColor **anchors** **applets** **bgColor** classes cookie **domain** embeds fgColor **formName** **forms** ids **images** lastModified layers linkColor **links** plugins referrer **title** **URL** vlinkColor	captureEvents close contextual getSelection handleEvent open releaseEvents routeEvent write writeln	N/A

Object	Standard	Properties	Methods	Event Handlers
event	W3C DOM	**altKey** **clientX** **clientY** **ctrlKey** **data** height **keyVal** layerX layerY modifiers pageX pageY **screenX** **screenY** **shiftKey** **target** **type** which width	N/A	N/A
Error	ECMA	**message** **name**	**toString**	N/A
EvalError	ECMA	—	—	—
FileUpload	DOM 0	—	—	—
Form	DOM 0	**action** **elements** encoding **length** **method** **name** target	**reset** **submit**	**onreset** **onsubmit**
Frame	DOM 0	—	—	—
Frameset	DOM 0	—	—	—
Function	ECMA	—	—	—

Object	Standard	Properties	Methods	Event Handlers
Global	ECMA	Infinity NaN undefined	decodeURI decodeURIComponent encodeURI encodeURIComponent escape eval isFinite isNaN parseFloat parseInt unescape	N/A
Hidden	DOM 0	form name type value	N/A	N/A
History	DOM 0	current length next previous	back forward go	N/A
Image	DOM 0	border complete height hspace lowsrc name src vspace width	N/A	N/A
java	DOM 0	—	—	—
JavaArray	DOM 0	—	—	—
JavaClass	DOM 0	—	—	—
JavaObject	DOM 0	—	—	—
JavaPackage	DOM 0	—	—	—
Layer	DOM 0	—	—	—
Link	DOM 0	—	—	—

Object	Standard	Properties	Methods	Event Handlers
Location	DOM 0	**hash** **host** **hostname** **href** **pathname** **port** **protocol** **search**	**reload** **replace**	N/A
Math	ECMA	**E** **LN10** **LN2** **LOG2E** **LOG10E** **PI** **SQRT1_2** **SQRT2**	**abs** **acos** **asin** **atan** **atan2** **ceil** **cos** **exp** **floor** **log** **max** **min** **pow** **random** **round** **sin** **sqrt** **tan**	N/A
MimeType	DOM 0	—	—	—
navigator	DOM 0	**appCodeName** **appName** **appVersion** **language** mimeTypes **platform** plugins userAgent	**javaEnabled** plugins.refresh preference savePreferences taintEnabled	N/A
netscape	DOM 0	—	—	—
Number	ECMA	**MAX_VALUE** **MIN_VALUE** **NaN** **NEGATIVE_INFINITY** **POSITIVE_INFINITY**	toExponential toFixed toLocaleString toPrecision **toString**	N/A
Object	ECMA	**constructor**	**hasOwnProperty** **isPrototypeOf** **propertyIsEnumerable** toLocaleString **toString** **valueOf**	N/A

Object	Standard	Properties	Methods	Event Handlers
Option	DOM 0	**defaultSelected** **form** **index** **label** **length** **selected** **text** **value**	N/A	N/A
Packages	DOM 0	—	—	—
Password	DOM 0	**defaultValue** **form** **name** **type** **value**	**blur** **focus** select	**onblur** **onfocus**
Plugin	DOM 0	—	—	—
Radio	DOM 0	**checked** **defaultChecked** **form** **name** **type** **value**	**blur** **click** **focus**	**onblur** **onclick** **onfocus**
RangeError	ECMA	**message** **name**	N/A	N/A
ReferenceError	ECMA	**message** **name**	N/A	N/A
RegExp	ECMA	**global** **ignoreCase** **lastIndex** multiline **source**	**exec** **test** **toString**	N/A
Reset	DOM 0	**form** **name** **type** **value**	**blur** **click** **focus**	**onblur** **onclick** **onfocus**
screen	DOM 0	**availHeight** **availLeft** **availTop** **availWidth** **colorDepth** **height** **pixelDepth** **width**	N/A	N/A

Object	Stand-ard	Properties	Methods	Event Handlers
Select	DOM 0	**form** **length** **name** **options** **selectedIndex** **type**	**blur** **focus**	**onblur** **onchange** **onfocus**
String	EC-MA	**length**	**charAt** **charCodeAt** **concat** **fromCharCode** **indexOf** **lastIndexOf** localeCompare **match** **replace** **search** **slice** **split** **substring** **substr** toLocaleLowerCase toLocaleUpperCase **toLowerCase** **toString** **toUpperCase** **valueOf**	N/A
Style	DOM 0	—	—	—
Submit	DOM 0	**form** **name** **type** **value**	**blur** **click** **focus**	**onblur** **onclick** **onfocus**
sun	DOM 0	—	—	—
SyntaxError	EC-MA	**message** **name**	N/A	N/A
Text	DOM 0	**defaultValue** **form** **name** **type** **value**	**blur** **focus** select	**onblur** **onchange** **onfocus** onselect
TextArea	DOM 0	**defaultValue** **form** **name** **type** **value**	**blur** **focus** select	**onblur** **onchange** **onfocus** **onkeydown** **onkeypress** **onkeyup** onselect

Object	Standard	Properties	Methods	Event Handlers
TypeError	ECMA	**message** **name**	N/A	N/A
URIError	ECMA	—	—	—
window	DOM 0	closed crypto defaultStatus **document** frames **history** innerHeight innerWidth length **location** locationbar menubar name offscreenBuffering opener outerHeight outerWidth pageXOffset pageYOffset parent personalbar screenX screenY scrollbars **self** status statusbar toolbar **top** **window**	**alert** atob **back** blur btoa captureEvents **clearInterval** **clearTimeout** close **confirm** crypto.random crypto.signText disableExternalCapture enableExternalCapture find focus **forward** handleEvent **home** moveBy moveTo open print **prompt** releaseEvents resizeBy resizeTo routeEvent scroll scrollBy scrollTo setHotKeys **setInterval** setResizable **setTimeout** setZOptions stop	onblur ondragdrop onerror onfocus **onload** onmove onresize **onunload**

Example: setting forms property attributes

Example of setting forms property attributes via JavaScript:

```
document.forms[0].setAttribute('agsubmitLinkDepth', 3);
document.forms[0].setAttribute('agsubmitHidden', true);
document.forms[0].setAttribute('agsubmitDiscardResponse', true);
```

Example of setting forms property attributes via HTML:

```
<form action=http://search.yahoo.com/bin/search
agsubmitLinkDepth=3 agsubmitDiscardResponse=true
agsubmitHidden=true>
```

For a more complex example of form submission via HTML, see "Form submission via HTML - complex example" on page 388.

Example: working around innerText and document.write

M-Business JavaScript engine does not support `innerText` or `document.write`.

innerText

Working around the absence of `innerText` is fairly straightforward. For example, instead of:

```
Cell.innerText = i;
```

You can use:

```
Cell.appendChild(document.createTextNode(i));
```

document.write

For `document.write`, the workaround depends on the complexity of the inserted content.

- You can sometimes use the `visibility` attribute to hide then reveal content.
- Another option is to use a Clone Node.
- Another option that works for more complex structures is to use `createObject`, as in the following example:

```
var oTable = document.createElement("TABLE");
var oTBody0 = document.createElement("TBODY");
oTable.appendChild(oTBody0);
var oRow, oCell;
var i;

for( i=0; i < 5; i++) {
     oRow = document.createElement("TR");
     oTBody0.appendChild(oRow);
     oCell = document.createElement("TD");
     oCell.appendChild(document.createTextNode(i));
     oRow.appendChild(oCell);
     }

var oTableContainer = document.getElementById("oTableContainer");
oTableContainer.appendChild(oTable);
```

Appendix B

M-Business JavaScript engine reference

Contents

Overview of M-Business JavaScript engine reference

Organization of the M-Business JavaScript engine reference material

This reference material in this guide only documents the features of M-Business JavaScript engine that are different from standard JavaScript. M-Business JavaScript engine does not implement the entire standard JavaScript feature set, and it adds a few features that are not part of that standard.

The list below indicates where to look for different types of reference material related to M-Business JavaScript engine:

- **Supported and unsupported features**: For details on the specific features from standard JavaScript that are not supported by M-Business Client, see "JavaScript" on page 320.
- **Added features**: For documentation of the features that M-Business Anywhere has added to standard JavaScript, see "M-Business JavaScript engine proprietary JavaScript objects" on page 335.
- **Features implemented through PODS**: For a list of JavaScript engine features, both standard and M-Business-specific, that are implemented through PODS, see "Standard JavaScript objects implemented through PODS" on page 334. Detailed documentation of these features is available, beginning with "PODS API mechanics" [*M-Business Anywhere API Reference*].
- **Standard JavaScript features**: For detailed documentation of standard JavaScript features, refer to the JavaScript sources listed in "Related publications" on page xi.

Note
JavaScript engine is case sensitive. Be sure to capitalize your JavaScript code exactly as it is shown in this reference material and examples in this guide.

Displaying and hiding JavaScript errors

When you are troubleshooting a JavaScript problem, you can programmatically turn the display of JavaScript errors on just before the section of JavaScript where you believe the problem is occurring, then programmatically turn the display of JavaScript errors off right after that point. Off is the default value for this setting.

The `PODSPrefs` object functions allow you to get and set the values of a variety of preferences for M-Business Client. Use the `ShowJSErrors` constant to reference the `ShowJSErrors` preference with these functions.

For more information on the functions available, and the synopsis to access them from JavaScript, see "PODSPrefs object" [*M-Business Anywhere API Reference*].

M-Business JavaScript engine additions to standard JavaScript

JavaScript engine contains a number of features that extend the basic JavaScript feature set in ways that are specific to the M-Business Client environment. “M-Business JavaScript engine additions to standard JavaScript” on page 333 lists the objects which have been added.

Note
In addition to the JavaScript objects listed in “M-Business JavaScript engine additions to standard JavaScript” on page 333, JavaScript engine supports a robust DOM implementation. The PODS “ADOMDOMImplementation object” [*M-Business Anywhere API Reference*] is directly accessible from JavaScript engine.

All of these objects are accessible from PODS, and the material in the PODS reference sections includes information about the JavaScript synopsis to access the methods and attributes of these objects.

Table 1. M-Business JavaScript engine additions to standard JavaScript

Object	Description
`avantgo`	Contains the “PODSAvantGo object” [*M-Business Anywhere API Reference*], which provides features unique to M-Business Client. See “avantgo object” on page 335.
`button`	Contains the “PODSButton object” [*M-Business Anywhere API Reference*], which allows limited management of buttons on the M-Business Client toolbar. See “button object” on page 336. Note that this is separate from the `Button` object in standard JavaScript.
`exception`	Contains the “PODSException object” [*M-Business Anywhere API Reference*], which provides a method to get the message associated with an exception. See “exception object” on page 336.

Standard JavaScript objects implemented through PODS

Some of JavaScript engine's basic JavaScript objects, defined in the standard, are actually implemented through the M-Business client extension API (PODS). The material in the PODS reference sections includes information about the JavaScript synopsis to access the methods and attributes of these objects.

Table 2. Standard JavaScript objects implemented through PODS

Object	PODS Reference Section
`History`	See "PODSHistory object" [*M-Business Anywhere API Reference*].
`Location`	See "PODSLocation object" [*M-Business Anywhere API Reference*].
`navigator`	See "PODSNavigator object" [*M-Business Anywhere API Reference*].
`window`	See "PODSWindow object" [*M-Business Anywhere API Reference*].

M-Business JavaScript engine proprietary JavaScript objects

These custom objects are M-Business-specific extensions to JavaScript and are non-standard JavaScript elements. All of these objects are imported from the corresponding PODS objects. For documentation of the objects' methods, follow the cross references to the corresponding PODS reference section.

> **Note**
> The standard JavaScript objects, `Window`, `History`, `Location`, `Screen`, and `Navigator` are also imported from the corresponding PODS objects, `PODSWindow`, `PODSHistory`, `PODSLocation`, `PODSScreen`, and `PODSNavigator`. However, the implementations follow standard JavaScript, so JavaScript programmers can obtain more useful information from a standard JavaScript reference than from the corresponding PODS object reference section. In the Recommended references section, see "JavaScript" on page xiii.

avantgo object

Description

Contains the "PODSAvantGo object" [*M-Business Anywhere API Reference*], which provides features unique to M-Business Client.

Property of

`window` object

Properties

- **preferences** Contains the "PODSPrefs object" [*M-Business Anywhere API Reference*], which allows you to get and set the user's M-Business Client preferences. See "preferences object" on page 337.
- **submissionMgr** Contains the "PODSSubmissionMgr object" [*M-Business Anywhere API Reference*], which manages the creation and monitoring of all form submissions from M-Business Client. See "submissionMgr object" on page 338.
- **symbolScanner** Contains the "PODSSymbolScanner object" [*M-Business Anywhere API Reference*], which accesses the API for working with a Symbol Technologies bar-code scanner device. See "symbolScanner object" on page 338.

Methods

For additional properties and methods, see "PODSAvantGo object" [*M-Business Anywhere API Reference*].

Event handlers

None

button object

Description

Contains the “PODSToolbar object” [*M-Business Anywhere API Reference*], which allows limited management of buttons on the M-Business Client toolbar.

Property of

`toolbar` object

Properties and methods

See “PODSToolbar object” [*M-Business Anywhere API Reference*].

Event handlers

None

exception object

Description

Contains the “PODSException object” [*M-Business Anywhere API Reference*], which provides a method to get the message associated with an exception.

Methods

See “PODSException object” [*M-Business Anywhere API Reference*].

Event handlers

None

podsArray object

Description

Contains the “PODSArray object” [*M-Business Anywhere API Reference*], which supports passing arrays between PODS and JavaScript engine.

Property of

`window` object

Properties and methods

See “PODSArray object” [*M-Business Anywhere API Reference*].

Note: You can reference a `podsArray` object from JavaScript, but you must create a `PODSArray object` from PODS.

Event handlers

None

preferences object

Description

Contains the "PODSPrefs object" [*M-Business Anywhere API Reference*], which supports passing arrays between PODS and JavaScript engine.

Property of

`avantgo` object

Properties and methods

See "PODSPrefs object" [*M-Business Anywhere API Reference*].

Event handlers

None

submission object

Description

Contains the "PODSSubmission object" [*M-Business Anywhere API Reference*], which manages submissions (forms) and their associated form elements from M-Business Client.

Properties and methods

See "PODSSubmission object" [*M-Business Anywhere API Reference*].

Event handlers

None

submissionElement object

Description

Contains the "PODSSubmissionElement object" [*M-Business Anywhere API Reference*], which provides read/write access to the name and value of each element in a form submission from M-Business Client.

Properties and methods

See "PODSSubmissionElement object" [*M-Business Anywhere API Reference*].

Event handlers

None

submissionMgr object

Description

Contains the “PODSSubmissionMgr object” [*M-Business Anywhere API Reference*], which manages the creation and monitoring of all form submissions from M-Business Client.

Property of

`avantgo` object

Properties and methods

See “PODSSubmissionMgr object” [*M-Business Anywhere API Reference*].

Event handlers

None

symbolScanner object

Description

Contains the “PODSSymbolScanner object” [*M-Business Anywhere API Reference*], which accesses the API for working with a Symbol Technologies bar-code scanner device.

Property of

`avantgo` object

Properties and methods

See “PODSSymbolScanner object” [*M-Business Anywhere API Reference*].

Event handlers

None

toolbar object

Description

Contains the “PODSToolbar object” [*M-Business Anywhere API Reference*], which supports customizing the M-Business Client toolbar and reading its buttons' properties.

Property of

`window` object

Properties and methods

See “PODSToolbar object” [*M-Business Anywhere API Reference*].

Event handlers

None

M-Business JavaScript engine proprietary JavaScript properties

M-Business JavaScript engine extends the standard HTML `Form` tag with properties that allow you to control additional aspects of the behavior of the `submit()` method. They are non-standard elements.

Form extensions

> **Note**
> Only the properties that M-Business Anywhere has added to the `Form` object are listed below. For complete documentation of the standard features of the `Form` object, refer to JavaScript sources listed in "Related publications" on page xi.

M-Business Anywhere JavaScript properties

- **agsubmitDiscardResponse**   Delete the response?

  Default: `FALSE`

- **agsubmitDisplayDefaultMessage**   Display a default message when submitting offline?

  Offline default: `TRUE`

  Online default: `FALSE`

- **agsubmitHidden**   Submission is hidden from the Forms Manager?

  Default: `FALSE`

- **agsubmitIncludeImages**   Allow response to contain images?

  Offline default: `TRUE`

  Online default: from application preferences

- **agsubmitLinkDepth**   Link depth of the response.

  Default: 0 (zero)

- **agsubmitMessage**   Message text to display in an alert after submitting offline.

  Default: "" (empty string); no message displayed

- **agsubmitMultiple**   Multiple offline submissions allowed?

  Default: `TRUE`

- **agsubmitOffsiteLinks**   Allow response to contain offsite links?

  Default: `TRUE`

- **agsubmitSize** Maximum size of the response in multiples of 1K.

  Default: 100

Sample code

Here is sample code that uses the M-Business Anywhere Form extensions.

```
<form action=http://search.yahoo.com/bin/search
    agsubmitMessage=Thanks
    agsubmitOffsiteLinks=true
    agsubmitIncludeImages=true
    agsubmitLinkDepth=2
    agsubmitSize=200
    agsubmitDiscardResponse=true
    agsubmitMultiple=false
    agsubmitHidden=false
    agsubmitDisplayDefaultMessage=true>

          ...

    <input size=15 name=p><BR>
    <INPUT TYPE=submit VALUE="Search">
    <INPUT TYPE=reset>
</form>
```

Accessing an M-Business JavaScript engine event object

In order for your JavaScript event handler function to access the event object, you must explicitly assign the event handler at runtime within your JavaScript code. You can do this using one of the following two methods:

```
objInput.onclick = myEventHandler(evt) {
    alert(this.name + " you clicked on (X,Y)= ("
    + evt.clientX + ", " + evt.clientY + ")"); return; };
```

Or

```
objInput.onclick = myEventHandler;
```

For both methods, the event object is always passed as the first argument.

M-Business JavaScript engine supports the following event handlers:

- **onClick**
- **onKeyDown** - except with form controls
- **onKeyPress** - except with form controls
- **onKeyUp** - except with form controls
- **onMouseDown** - except with form controls
- **onMouseMove** - except with form controls
- **onMouseOut** - except with form controls
- **onMouseUp** - except with form controls

> **Note**
> You need to write a PODSEventHandler and register it with the PODSEventMgr. The PODSEventHandler has a single handleEvent() method that takes a PODSEvent object. A PODSEvent is a simple wrapper on the native event type for each platform.
> See the ObjectSrc sample POD in the *pods.zip* file, available from http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere/, for a similar example of creating a PODSObjectSrc and registering it with the PODSObjectMgr. You can download the *pods.zip* file from:
> http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere

M-Business JavaScript engine methods

The following methods are M-Business-specific extensions to standard JavaScript objects. They are non-standard JavaScript elements.

These methods are now deprecated. In place of these methods, you should try to use the `Form` object's standard `submit()` method with the extended M-Business properties listed under "Form extensions" on page 340.

submitForm( )

Description

Submits a form.

Property of

`Form` object

JavaScript synopsis

submitForm(
 confirmMessage,
 followOffsiteLinks,
 includeImages,
 linkDepth,
 maxSize,
 trashResponse,
 isMulti,
 isHidden,
 displayDefaultMsg)

JavaScript Parameters

- **confirmMessage** Displays a message to the user in a confirmation dialog. This argument can be `NULL`, which suppresses the display of the dialog to the user.

  Note: The user cannot cancel, only confirm.

- **followOffsiteLinks** If `true`, the response page can be outside the web site's domain. If `false`, offsite links are disallowed. The default is `false`.

- **includeImages** If `true`, the response can contain images. Otherwise `false`. The default is `true`.

- **linkDepth** An integer specifying the number of levels of links to follow when downloading the page. Typically, this is 0, which just downloads the main response page. A value of 1 indicates that M-Business Client should traverse all of the links on the response page and download those pages as well. The default is 0.

- **maxSize** Specifies maximum size in kilobytes of the response.

- **trashResponse** If `true`, do not display the response page. As soon as the user successfully submits the form, delete the form submission from the user's device.

- **isMulti** If `true`, the user can submit the same form many times. If `false`, the user can submit only one form request at a time against a particular web page.
- **isHidden** If `true`, the form should not be listed in the Forms Manager page. Setting this argument to `true` and `trashResponse` to `true` allows you to handle form requests silently, without the user having to know that an HTML form was used.
- **displayDefaultMsg** If `true`, a default message appears. If `false`, the message does not appear.

submitNoResponse( )

Description

Submits a form that requires no response.

Property of

`Form` object

JavaScript synopsis

submitNoResponse(
 confirmMessage,
 isHidden,
 isMulti)

JavaScript Parameters

- **confirmMessage** Displays a message to the user in a confirmation dialog. It can be `NULL`.
- **isHidden** If `true`, the form should not be listed in the Forms Manager page.
- **isMulti** If `true`, the user can submit the same form many times. If `false`, the user can submit only one form request at a time against a particular web page.

submitWithResponse( )

Description

Submits a form and displays the response.

Property of

`Form` object

JavaScript synopsis

submitWithResponse(
confirmMessage,
linkDepth,
isMulti)

Parameters

- **confirmMessage** Displays a message to the user in a confirmation dialog. This argument can be `NULL`.
- **linkDepth** An integer specifying the number of levels of links to follow when downloading the page. Typically, this is 0, which just downloads the main response page. A value of 1 indicates that M-Business Client should traverse all of the links on the response page and download those pages as well. The default is 0.
- **isMulti** If `true`, the user can submit the same form many times. If `false`, the user can submit only one form request at a time against a particular web page.

Resource allocation limitations

JavaScript programs can allocate four types of values as they run: objects, properties, strings, and floating-point numbers.

For example, the following script allocates an object and two properties:

```
var myobj = new Object;
myobj.color = "red";
myobj.size = "large";
```

The following script allocates a string:

```
var mystr = "cruella" + "deville";
```

This script allocates a floating-point number:

```
var x = 3.14159 - 2.71828;
```

JavaScript automatically garbage collects values that are no longer in use.

Allocating objects

An M-Business Client JavaScript program can allocate an unlimited number of values when only a fixed number of values are in use at any one time. For example, consider the following loop:

```
for (var x = 0 ; x < 100000 ; ++x) {
  var obj = new Object;
  obj.foo = 4;
}
```

This loop will run without problems in M-Business JavaScript (although it may take some time to complete!). The loop allocates 100,000 objects, but each allocated object becomes garbage as soon as the next loop iteration begins, so only one object is actually in use at any given time. The following loop, on the other hand, may run out of memory:

```
var a = new Array;
for (var x = 0 ; x < 1000 ; ++x) {
  var obj = new Object;
  obj.foo = 4;
  a[a.length] = obj;
}
```

This loop allocates 1000 objects and keeps them all in use by storing them in an array.

Number of objects in use

The number of objects that can be used at one time in M-Business JavaScript engine depends on:

- How much memory a mobile device has.
- The device features being used, including TCP/IP and/or SSL (via M-Business Client's SSL feature).

- The PODS that are loaded.
- Any other applications present on the device.

Appendix C

M-Business JavaScript engine sample code

Contents

Downloading and accessing the JavaScript engine sample files

The sample files for M-Business JavaScript engine use DHTML to demonstrate what you can do with these tools in a typical business application.

These sample files are contained in a file called *dhtml.zip*.

♦ To download the dhtml.zip file:

1. Navigate to:

   http://www.ianywhere.com/developer/product_manuals/mbusiness_anywhere.

2. Click the **DHTML sample code** link.
3. Unzip all of the files, including images, into one directory.
4. To run the demo, open the home page, *index.html*.

Home page sample: expanding and collapsing hierarchical text

A home page is the first page users see when they run your application. The home page in this example, *index.html*, presents some current information and provides access to the other pages in the application. The workings of this page are described in detail in "Home page: expanding and collapsing hierarchical text" on page 138.

The full source code for *index.html* is listed below. For instructions on downloading this source code, see "Downloading and accessing the JavaScript engine sample files" on page 350.

```
<html>

<head>

<title>Quick Update</title>

<meta name="HandheldFriendly" content="True">

<script language="JavaScript1.2">

<!--

//
// FILE: index.html
//
// Revision Date: March 15, 2002
//
// Copyright (c) 2006, iAnywhere Solutions, Inc.
//

//-->

<!--

    //
    // FUNCTION: InitializeNav()
    //
    function InitializeNav()
    {

        if (arguments.length != 0)
            throw new Error("function InitializeNav() incorrectly called");

        //
        // Navigation bar initially not displayed.
        //
        HideNav();

        //
        // Show the navigation URLs that are at the bottom of the HTML
        document.
        //
        document.getElementById("navurls").style.display = "block";

        return;

    } //InitializeNav
```

```
        //
        // FUNCTION: ShowNav()
        //
        // Purpose: Show the dropdown navigation bar.
        //
        function ShowNav()
        {

            if (arguments.length != 0)
                throw new Error("function ShowNav() incorrectly called");

            document.getElementById('openNav').style.display="block";
            document.getElementById('closedNav').style.display="none";
            document.getElementById('nav').style.display="block";

            return;

        } //ShowNav

        //
        // FUNCTION: HideNav()
        //
        // Purpose: Hide the dropdown navigation bar.
        //
        function HideNav()
        {

            if (arguments.length != 0)
                throw new Error("function HideNav() incorrectly called");

            document.getElementById('openNav').style.display="none";
            document.getElementById('closedNav').style.display="block";
            document.getElementById('nav').style.display="none";

            return;
        } //HideNav

        //
        // FUNCTION: ExpandDisp()
        //
        function ExpandDisp(id)
        {

            if (arguments.length != 1)
                throw new Error("function ExpandDisp() incorrectly called");

            document.getElementById(id).style.display =
                ((document.getElementById(id).style.display == "none")
                    ? "block" : "none");

            return;

        } //ExpandDisp

    //-->

    </script>

    </head>

    <body onload="InitializeNav();"
        style="FONT-SIZE: 8px; COLOR: #000000;
            FONT-FAMILY: verdana,geneva,arial;
            MARGIN-TOP: 0px; MARGIN-LEFT: 0px">
```

```
<table border="0" width="144" style="font-size: 8px;
    COLOR: #000000;">

<tr>
    <td valign="top" width="128">
    <IMG height=34 src="logo.gif" width=118 border=0>
    <IMG height=5 src="darkdotline.gif" width=118 border=0>
    QUICK UPDATE
    <br>
    <IMG height=5 src="darkdotline.gif" width=118 border=0>
    <br>

    <!-- If you want the sections to be      -->
    <!-- initially displayed, within the DIV -->
    <!-- tags, change display:none to        -->
    <!-- display:block -->


    <!-- Quick Update 1 -->

    <IMG height=11 src="pointer1.gif" width=16 border=0>
    <b> <A onclick="return ExpandDisp('item1Text');" href ="#">
    ROADMAP
    </A>
    </b>

    <div ID="item1Text" style="display:none;">
    Supply issues in the Singapore plant have not affected the
        company's Q2 release schedule.
    </div>

    <!-- Quick Update 2 -->
    <br>
    <IMG height=11 src="pointer1.gif" width=16 border=0>
    <b>
    <A onclick="return ExpandDisp('item2Text');" href ="#">
    PERSONNEL
    </A>
    </b>

    <div ID="item2Text" STYLE="display:none;">
    Donald Morgan named acting PM of space component project.
    </div>

    <!-- Quick Update 3 -->
    <br>
    <IMG height=11 src="pointer1.gif" width=16 border=0>
    <b>
    <A onclick="return ExpandDisp('item3Text');" href ="#">
    COMPETITION
    </A>
    </b>

    <div ID="item3Text" style="display:none;">
    United Aerospace has inside track on Boeing deal.
    </div>
    </td>

    <!-- Navigation Bar -->
    <td valign="top" align="left" width="16">

    <div id="openNav" onclick="HideNav()">
    <IMG height=11 src="open.gif" width=16 border=0>
    </div>
    <div id="closedNav" onclick="ShowNav()">
    <IMG height=11 src="closed.gif" width=16 border=0>
    </div>

    <div id="nav">
    <IMG height=92 src="home_nav.gif" width=16 useMap=#nav border=0>
    </div>

    </td>

</tr>

</table>

<br>

<map name="nav">
<area shape="RECT" coords="0,0,15,15" href="index.html">
<area shape="RECT" coords="0,19,15,34" href="calendar.html">
<area shape="RECT" coords="0,38,15,53" href="chart.html">
<area shape="RECT" coords="0,57,15,72" href="mail.html">
<area shape="RECT" coords="0,76,15,91" href="leads.html">
</map>

<div ID="navurls">
<table border="0" width="120"
    style="FONT-SIZE: 8px; PADDING-BOTTOM: 1px; PADDING-TOP: 1px;
        FONT-FAMILY: verdana,geneva,arial">

    <tr align="center" valign="top" >
    <td><A href="index.html">Home</A></td>
    <td><A href="calendar.html">Cal</A></td>
    <td><A href="chart.html">Chart</A></td>
    <td><A href="mail.html">Mail</A></td>
    <td><A href="leads.html">Leads</A></td>

    </tr>
</table>
</div>

</body>
</html>
```

Calendar page sample: hiding and displaying page elements

Many applications involve some sort of calendar information. The calendar page in this sample application, *calendar.html*, receives event information from a server application and allows the user to view the schedule details for any particular day by tapping that day in the calendar. The workings of this page are described in detail in "Calendar page: hiding and displaying page elements" on page 141.

The full source code for *calendar.html* is listed below. For instructions on downloading this source code, see "Downloading and accessing the JavaScript engine sample files" on page 350.

```
<html>
<head>

<title>My Calendar</title>
<meta name="HandheldFriendly" content="True">

<script language="JavaScript1.2">

<!--
//
// FILE: calendar.html
//
// Revision Date: March 15, 2002
//
// Copyright (c) 2006, iAnywhere Solutions, Inc.
//
//-->

<!--
    //
    // Globals
    //
    var oldTD = null;

    //
    // FUNCTION: InitializeNav()
    //
    function InitializeNav()
    {
        //
        // Initialize the calendar.
        //
        InitializeCalAppts();

        //
        // By default, the navigation bar is not displayed.
        //
        HideNav();

        //
        // Show the navigation URLs that are at the bottom of the HTML
        document.
        //
        document.getElementById("navurls").style.display = "block";

        return;

    } //InitializeNav
```

```
//
// FUNCTION: InitializeCalAppts()
//
function InitializeCalAppts()
{
    var objAppointments;

    if (arguments.length != 0)
        throw new Error("function InitializeCalAppts()
            incorrectly called");

    for (i=1; i<=31; i++)
    {
        //
        // Hide all appointments for the month.
        //
        var objAppointments = document.getElementById("dateD" + i);

        //
        // If the day has appointments, it will have a
        // non-null appointments object.
        //
        if (objAppointments != null)
            objAppointments.style.display = "none";

    } // for

    return;

} // InitializeCalAppts

//
// FUNCTION: ShowNav()
//
// Purpose: Show the dropdown navigation bar.
//
function ShowNav()
{

    if (arguments.length != 0)
        throw new Error("function ShowNav()
            incorrectly called");

    document.getElementById('openNav').style.display="block";
    document.getElementById('closedNav').style.display="none";
    document.getElementById('nav').style.display="block";
    return;

} // ShowNav

//
// FUNCTION: HideNav()
//
// Purpose: Hide the dropdown navigation bar.
//
function HideNav()
{

    if (arguments.length != 0)
        throw new Error("function HideNav()
            incorrectly called");

    document.getElementById('openNav').style.display="none";
    document.getElementById('closedNav').style.display="block";
```

```
        document.getElementById('nav').style.display="none";

        return;

    } // HideNav

    //
    // FUNCTION: ShowDate()
    //
    function ShowDate(newTD) {
        var oldDisp = null;
        var newDisp = null;

        if (arguments.length != 1)
            throw new Error("function ShowDate()
                incorrectly called");

        //
        // First hide any existing appointments that are being shown.
        //
        if (oldTD != null) {
            oldDisp = document.getElementById("date" + oldTD.id);

            if (oldDisp != null)
                oldDisp.style.display = "none";

            oldTD.style.backgroundColor="#ffffff";

            } // if

        //
        // Process the new selected date.
        //
        newTD.style.backgroundColor="#ffff00";

        newDisp = document.getElementById("date" + newTD.id);

        if (newDisp != null)
            newDisp.style.display = "block";

        oldTD = newTD;

        return false;

    } // ShowDate

//-->

</script>

</head>


<body style="FONT-SIZE: 8px; COLOR: #000000; FONT-FAMILY:
        verdana,geneva,arial;
    MARGIN-TOP: 0px; MARGIN-LEFT: 0px" onload="InitializeNav()">


<table border="0" width="144" style="font-size: 8px; COLOR: #000000;">
    <tr>
        <td valign="top" width="128">
        <IMG height=34 src="logo.gif" width=118 border=0>
        <IMG height=5 src="darkdotline.gif" width=118 border=0>
        <br>
        <span style="COLOR: #333333; FONT-SIZE: 8px;">
        MY CALENDAR
        </span>

        <br>
        <IMG height=5 src="darkdotline.gif" width=118 border=0>
        <br>
        <span style="FONT-SIZE: 9px;">
        <b> September 2001 </b>
        </span>
        <br>
        <br>

        <table border="1" cellspacing="1" ID="Calendar" style="FONT-SIZE: 
8px;
            BACKGROUND-COLOR: #ffffff;" width="115">

    <tr align="center">
        <th width="16">Su</th>
        <th width="16">Mo</th>
        <th width="16">Tu</th>
        <th width="16">We</th>
        <th width="16">Th</th>
        <th width="16">Fr</th>
        <th width="16">Sa</th>
    </tr>

    <tr align="center">
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)" id=D1>1</td>
    </tr>

    <tr align="center">
        <td onclick="ShowDate(this)" id=D2>2</td>
        <td onclick="ShowDate(this)" id=D3>
            <font style="COLOR: #cc0000"> <b>3</b> </font></td>
        <td onclick="ShowDate(this)" id=D4>
            <font style="COLOR: #cc0000"> <b>4</b> </font></td>
        <td onclick="ShowDate(this)" id=D5>5</td>
        <td onclick="ShowDate(this)" id=D6>6</td>
        <td onclick="ShowDate(this)" id=D7>7</td>
        <td onclick="ShowDate(this)" id=D8>8</td>
    </tr>

    <tr align="center">
        <td onclick="ShowDate(this)" id=D9>9</td>
        <td onclick="ShowDate(this)" id=D10>10</td>
        <td onclick="ShowDate(this)" id=D11>11</td>
        <td onclick="ShowDate(this)" id=D12>12</td>
        <td onclick="ShowDate(this)" id=D13>13</td>
        <td onclick="ShowDate(this)" id=D14>14</td>
        <td onclick="ShowDate(this)" id=D15>15</td>

    </tr>
        <td onclick="ShowDate(this)" id=D16>16</td>
        <td onclick="ShowDate(this)" id=D17>17</td>
        <td onclick="ShowDate(this)" id=D18>18</td>
        <td onclick="ShowDate(this)" id=D19>19</td>
        <td onclick="ShowDate(this)" id=D20>20</td>
        <td onclick="ShowDate(this)" id=D21>21</td>
        <td onclick="ShowDate(this)" id=D22>22</td>
    </tr>

    <tr align="center">
        <td onclick="ShowDate(this)" id=D23>23</td>
        <td onclick="ShowDate(this)" id=D24>24</td>
        <td onclick="ShowDate(this)" id=D25>25</td>
        <td onclick="ShowDate(this)" id=D26>26</td>
        <td onclick="ShowDate(this)" id=D27>27</td>
        <td onclick="ShowDate(this)" id=D28>28</td>
        <td onclick="ShowDate(this)" id=D29>29</td>
    </tr>

    <tr align="center">
        <td onclick="ShowDate(this)" id=D30>30</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
        <td onclick="ShowDate(this)">&nbsp;</td>
    </tr>

    </table>


<div class="dataElement" ID="dateD3">
<br>
<table width="150" cellspacing="1" style="FONT-SIZE: 8px;
    BACKGROUND-COLOR: #cccccc;" border="1">
<tr>
    <td nowrap valign=top bgcolor="#ffff00"><b> 8:00 </b></td>
    <td>Meeting with Phil Condit. Remember stealth module swap arrays.</td>
</tr>
</table>

<br>

<table width="150" cellspacing="1" style="FONT-SIZE: 8px;
    BACKGROUND-COLOR: #cccccc;" border="1">
<tr>
    <td nowrap valign=top bgcolor="#ffff00"> <b>9:00</b> </td>
    <td> Conference call with near east manufacturing facilities
        managers </td>
</tr>
</table>
</div>

<div class="dataElement" ID="dateD4">
<br>
<table width="150" cellspacing="1" style="FONT-SIZE: 8px;
    BACKGROUND-COLOR: #cccccc;" border="1">
<tr>
    <td nowrap valign=top bgcolor="#ffff00"><b>10:00</b></td>
    <td>
    Exec staff - Reports from Davis (finance), Whittaker
        (Engineering), Gardner (Program Management)
    </td>
</tr>
</table>

<br>
```

```
<table width="150" cellspacing="1" style="FONT-SIZE: 8px;
    BACKGROUND-COLOR: #cccccc;" border="1">
<tr>
    <td nowrap valign=top bgcolor="#ffff00"><b>12:00</b></td>
    <td>Lunch with Trudy</td>
</tr>
</table>

</div>

</td>

    <td valign="top" align="right" width="16">
    <div id="openNav" onclick="HideNav()">
    <IMG height=11 src="open.gif" width=16 border=0>
    </div>

    <div id="closedNav" onclick="ShowNav()">
    <IMG height=11 src="closed.gif" width=16 border=0>
    </div>

    <div id="nav">
    <IMG id=IMG1 height=92 src="calendar_nav.gif" width=16
        useMap=#nav border=0 >
    </div>
    </td>

</tr>

</table>

<br>

<map name="nav">
<area shape="RECT" coords="0,0,15,15" href="index.html">
<area shape="RECT" coords="0,19,15,34" href="calendar.html">
<area shape="RECT" coords="0,38,15,53" href="chart.html">
<area shape="RECT" coords="0,57,15,72" href="mail.html">
<area shape="RECT" coords="0,76,15,91" href="leads.html">
</map>

<div id="navurls">
<table border="0" width="120"
    style="font-family:verdana,geneva,arial;font-size:8px;
    padding-bottom:1px;padding-top:1px;">

    <tr align="center" valign="top" >
        <td><A href="index.html">Home</A></td>
        <td><A href="calendar.html">Cal</A></td>
        <td><A href="chart.html">Chart</A></td>
        <td><A href="mail.html">Mail</A></td>
        <td><A href="leads.html">Leads</A></td>
    </tr>

</table>
</div>

</body>

</html>
```

Chart page sample: dynamically changing graphic information

Many applications involve some sort of graphic display of information. The chart page in this sample application, *chart.html*, presents a bar chart with data for four quarters and allows the user to adjust any quarter's value up or down by tapping an up or down arrow for that quarter. The workings of this page are described in detail in "Chart page: dynamically changing graphic information" on page 142.

The full source code for *chart.html* is listed below. For instructions on downloading this source code, see "Downloading and accessing the JavaScript engine sample files" on page 350.

```
<html>

<head>

<title>Today's Charts</title>

<meta name="HandheldFriendly" content="True">

<script language="JavaScript1.2">

<!--

//
// FILE: chart.html
//
// Revision Date: March 15, 2002
//
// Copyright (c) 2006, iAnywhere Solutions, Inc.
//

//-->

<!--

    //
    // Globals
    //
    var intQ1 = 0;
    var intQ2 = 1;
    var intQ3 = 2;
    var intQ4 = 3;

    //
    // [bar chart color, initial revenue in millions]
    //
    var arrQ = [["red", 7],
        ["blue", 12],
        ["yellow", 15],
        ["green", 21]];

    var intQuarters = 4;

    //
    // Indexes into the above array.
    //
    var intQ_Color = 0;
    var intQ_Rev = 1;
```

```
//
// FUNCTION: InitializeNav()
//
function InitializeNav()
{

    if (arguments.length != 0)
        throw new Error("function InitializeNav()
            incorrectly called");

    //
    // By default, the navigation bar is not displayed.
    //
    HideNav();

    //
    // Show the navigation URLs that are at the bottom of the HTML
    document.
    //
    document.getElementById("navurls").style.display = "block";

    for (i = 0; i < intQuarters; i++)
    {
        var obj;

        obj = document.getElementById("Q" + (i + 1) + "_Bar");

        obj.style.height = arrQ[i][intQ_Rev] + "px";

        obj.style.backgroundColor = arrQ[i][intQ_Color];

    }

    return;

} // InitializeNav

//
// FUNCTION: ShowNav()
//
// Purpose: Show the dropdown navigation bar.
//
function ShowNav()
{

    if (arguments.length != 0)
        throw new Error("function ShowNav()
            incorrectly called");

    document.getElementById('openNav').style.display="block";
    document.getElementById('closedNav').style.display="none";
    document.getElementById('nav').style.display="block";

    return;

} // ShowNav

//
// FUNCTION: HideNav()
//
// Purpose: Hide the dropdown navigation bar.
//
function HideNav()
{
    if (arguments.length != 0)
        throw new Error("function HideNav()
            incorrectly called");

    document.getElementById('openNav').style.display="none";
    document.getElementById('closedNav').style.display="block";
    document.getElementById('nav').style.display="none";

    return;

} // HideNav

//
// FUNCTION: BumpDown()
//
function BumpDown(intQ_IN)
{
    var obj;

    if (arguments.length != 1)
        throw new Error("function BumpDown() incorrectly called");

    obj = document.getElementById("Q" + (intQ_IN + 1) + "_Bar");

    if (arrQ[intQ_IN][intQ_Rev] > 7)
    {
        arrQ[intQ_IN][intQ_Rev] -= 6;

        obj.style.height = arrQ[intQ_IN][intQ_Rev] + "px";
        obj.style.width = "20px";
        obj.style.backgroundColor = arrQ[intQ_IN][intQ_Color];
    }

    return;

} // BumpDown
//
// FUNCTION: BumpUp()
//
function BumpUp(intQ_IN)
{

    if (arguments.length != 1)
        throw new Error("function BumpUp()
            ncorrectly called");

    var obj = document.getElementById("Q" + (intQ_IN + 1) +  "_Bar");

    if (arrQ[intQ_IN][intQ_Rev] < 53)
    {
        arrQ[intQ_IN][intQ_Rev] += 6;
        obj.style.height = arrQ[intQ_IN][intQ_Rev] + "px";
        obj.style.width = "20px";
        obj.style.backgroundColor = arrQ[intQ_IN][intQ_Color];
    }

    return;

} // BumpUp

//
// FUNCTION: ShowSales()
//
function ShowSales(intQ_IN)
{
```

```
            var obj;
            if (arguments.length != 1)
                throw new Error("function ShowSales()
                    incorrectly called");

            obj = document.getElementById("Q" + (intQ_IN + 1) + "_Bar");

            alert("Q" + (intQ_IN + 1) + " sales are $" + arrQ[intQ_IN]
             [intQ_Rev] + "MM");

            return;

        } // ShowSales

    //-->
    </script>

    </head>

    <body onload="InitializeNav();"
        style="FONT-SIZE: 8px; COLOR: #000000; FONT-FAMILY:
    verdana,geneva,arial;
        MARGIN-TOP: 0px; MARGIN-LEFT: 0px">

    <table border="0" width="144" style="font-size: 8px; COLOR: #000000;">

        <!-- start only row in parent table -->
        <tr>

        <!--- begin column 1 -->
        <td valign="top" width="128">
        <img height="34" src="logo.gif" width="118" border="0">
        <img height="5" src="darkdotline.gif" width="118" border="0">

        <span style="COLOR: #333333; FONT-SIZE: 8px;">
        TODAY'S CHARTS
        </span>

        <br>
        <img height="5" src="darkdotline.gif" width="118" border="0">
        <br>

        <!-- begin the second table for the actual chart -->
        <table border="1" width="100" style="font-size: 8px;">

        <tr bgColor="white">

        <td width="23" height="65" align="center" valign="bottom"
            onclick="ShowSales(intQ1)">
        <img id="Q1_Bar" alt="Q1" align="bottom" style="WIDTH: 20px;
            HEIGHT: 10px; BACKGROUND-COLOR: red;" WIDTH="20" HEIGHT="1">
        </td>

        <td width="23" height="65" align="center" valign="bottom"
            onclick="ShowSales(intQ2)">
        <img id="Q2_Bar" alt="Q2" align="bottom" style="WIDTH: 20px;
            HEIGHT: 20px; BACKGROUND-COLOR: blue;" WIDTH="20" HEIGHT="1">
        </td>

        <td width="23" height="65" align="center" valign="bottom"
            onclick="ShowSales(intQ3)">
        <img id="Q3_Bar" alt="Q3" align="bottom" style="WIDTH: 20px;
            HEIGHT: 30px; BACKGROUND-COLOR: yellow;" WIDTH="20" HEIGHT="1">
        </td>
        <td width="23" height="65" align="center" valign="bottom"
            onclick="ShowSales(intQ4)">
        <img id="Q4_Bar" alt="Q4" align="bottom" style="WIDTH: 20px;
            HEIGHT: 40px; BACKGROUND-COLOR: green;" WIDTH="20" HEIGHT="1">
        </td>

        </tr>

        <tr>

            <td align="center" onclick="BumpUp(intQ1)">
            <img src="open.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpUp(intQ2)">
            <img src="open.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpUp(intQ3)">
            <img src="open.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpUp(intQ4)">
            <img src="open.gif" border="0" width="16" height="11">
            </td>

        </tr>

        <tr>

            <td align="center">Q1</td>
            <td align="center">Q2</td>
            <td align="center">Q3</td>
            <td align="center">Q4</td>

        </tr>

        <tr>

            <td align="center" onclick="BumpDown(intQ1)">
            <img src="closed.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpDown(intQ2)">
            <img src="closed.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpDown(intQ3)">
            <img src="closed.gif" border="0" width="16" height="11">
            </td>

            <td align="center" onclick="BumpDown(intQ4)">
            <img src="closed.gif" border="0" width="16" height="11">
            </td>

        </tr>

    </table>

    <!-- end table for chart table -->

    </td>

    <!--- end column 1 -->
```

```
<!-- start column 2 -->

<td valign="top" align="right" width="16">

<div id="openNav" onclick="HideNav()">
<img height="11" src="open.gif" width="16" border="0">
</div>

<div id="closedNav" onclick="ShowNav()">
<img height="11" src="closed.gif" width="16" border="0">
</div>

<div id="nav">
<img height="92" src="chart_nav.gif" width="16" useMap="#nav" border="0">
</div>

<!-- end column 2 -->
</td>

<!-- end only row in parent table -->
</tr>

<!-- end table for enclosing table -->
</table>

<br>

<map name="nav">
<area shape="RECT" coords="0,0,15,15" href="index.html">
<area shape="RECT" coords="0,19,15,34" href="calendar.html">
<area shape="RECT" coords="0,38,15,53" href="chart.html">
<area shape="RECT" coords="0,57,15,72" href="mail.html">
<area shape="RECT" coords="0,76,15,91" href="leads.html">
</map>

<div ID="navurls">
<table border="0" width="120"
    style="FONT-SIZE: 8px; PADDING-BOTTOM: 1px; PADDING-TOP: 1px;
    FONT-FAMILY: verdana,geneva,arial">

    <tr align="center" valign="top" >
        <td><A href="index.html">Home</A></td>
        <td><A href="calendar.html">Cal</A></td>
        <td><A href="chart.html">Chart</A></td>
        <td><A href="mail.html">Mail</A></td>
        <td><A href="leads.html">Leads</A></td>
    </tr>

</table>
</div>

</body>
</html>
```

Mail page sample: a form submission

Many applications involve some sort of form submission. The mail page in this sample application, *mail.html*, submits a form. The page displays messages and allows the user to reply by tapping a Reply link. The Reply link displays the form. The workings of this page are described in detail in "Mail page: a form submission" on page 144.

The full source code for *mail.html* is listed below. For instructions on downloading this source code, see "Downloading and accessing the JavaScript engine sample files" on page 350.

```
<html>

<head>

<title>New Mail</title>

<meta name="HandheldFriendly" content="True">

<script language="JavaScript1.2">

<!--

//
// FILE: mail.html
//
// Revision Date: March 15, 2002
//
// Copyright (c) 2006, iAnywhere Solutions, Inc.
//
//-->
    <!--
    //
    // FUNCTION: InitializeNav()
    //
    function InitializeNav()
    {

        if (arguments.length != 0)
            throw new Error("function InitializeNav()
                incorrectly called");


        initializeDisp();

        //
        // By default, the navigation bar is not displayed.
        //
        HideNav();

        //
        // Show the navigation URLs that are at the bottom of the HTML
        document.
        //
        document.getElementById("navurls").style.display = "block";

        return;

    } // InitializeNav

    // Purpose: Show the dropdown navigation bar.
    //
```

```
function ShowNav()
{

    if (arguments.length != 0)
        throw new Error("function ShowNav()
            incorrectly called");

    document.getElementById('openNav').style.display="block";
    document.getElementById('closedNav').style.display="none";
    document.getElementById('nav').style.display="block";

    return;

} // ShowNav

//
// FUNCTION: HideNav()
//
// Purpose: Hide the dropdown navigation bar.
//
function HideNav()
{

    if (arguments.length != 0)
        throw new Error("function HideNav()
        incorrectly called");

    document.getElementById('openNav').style.display="none";
    document.getElementById('closedNav').style.display="block";
    document.getElementById('nav').style.display="none";

    return;

} // HideNav

//
// FUNCTION: InitializeDisp()
//
function InitializeDisp()
{

    var obj;

    if (arguments.length != 0)
        throw new Error("function InitializeDisp()
            incorrectly called");

    obj = document.getElementById("item1Text");
    if (obj)
        obj.style.display = "none";

    obj = document.getElementById("item2Text");
        if (obj)
        obj.style.display = "none";

    document.getElementById("navurls").style.display = "none";
    return;

} // InitializeDisp

//
// FUNCTION: ExpandDisp()
//
function ExpandDisp(id)
```

```
    {
        var style;

        if (arguments.length != 1)
            throw new Error("function ExpandDisp()
                incorrectly called");

        style = document.getElementById(id).style;
        style.display = ((style.display == "none") ? "block" : "none" );

        return false;

    } // ExpandDisp
//-->
</script>

</head>

<body onload="InitializeNav();"
    style="FONT-SIZE: 8px; COLOR: #000000; FONT-FAMILY:
        verdana,geneva,arial;
    MARGIN-TOP: 0px; MARGIN-LEFT: 0px">

<table border="0" width="144" style="font-size: 8px; COLOR: #000000;">

<tr>
    <td valign="top" width="128">
    <IMG height=34 src="logo.gif" width=118 border=0>
    <IMG height=5 src="darkdotline.gif" width=118 border=0>

    <span style="COLOR: #333333">
    NEW MAIL
    </span>
    <br>
    <IMG height=5 src="darkdotline.gif" width=118 border=0>
    <br>

    <span style="COLOR: #333333">
    <b>FROM:</b>Fred Jones, Boeing<br>
    Did you mail me the driving instructions to your facility?<br>
    <A onclick="return ExpandDisp('item2Text');" href ="#">Reply</A>
    </span>

    <div class="dataElement" ID="item2Text">
    <form action="mailsent.html" id=form1 name=form1>
    <TEXTAREA id=textarea style="FONT-SIZE: 8px; WIDTH: 106px;
        HEIGHT: 86px" name=textarea rows=6 cols=17>
    </TEXTAREA>
    <br>
    <input type=submit value=send id=submit1 name=submit1>
    </form>
    </div>

    <br>

    </td>

    <td valign="top" align="right" width="16">
    <div id="openNav" onclick="HideNav()">
    <IMG height=11 src="open.gif" width=16 border=0>
    </div>

    <div id="closedNav" onclick="ShowNav()">
    <IMG height=11 src="closed.gif" width=16 border=0>
    </div>
    <div id="nav">
    <IMG height=92 src="mail_nav.gif" width=16 useMap=#nav border=0>
    </div>

    </td>

</tr>

</table>

<br>

<map name="nav">
<area shape="RECT" coords="0,0,15,15" href="index.html">
<area shape="RECT" coords="0,19,15,34" href="calendar.html">
<area shape="RECT" coords="0,38,15,53" href="chart.html">
<area shape="RECT" coords="0,57,15,72" href="mail.html">
<area shape="RECT" coords="0,76,15,91" href="leads.html">
</map>


<div ID="navurls">
<table border="0" width="120"
    style="FONT-SIZE: 8px; PADDING-BOTTOM: 1px; PADDING-TOP: 1px;
    FONT-FAMILY: verdana,geneva,arial">

<tr align="center" valign="top" >
    <td><A href="index.html">Home</A></td>
    <td><A href="calendar.html">Cal</A></td>
    <td><A href="chart.html">Chart</A></td>
    <td><A href="mail.html">Mail</A></td>
    <td><A href="leads.html">Leads</A></td>
</tr>

</table>

</div>


</body>

</html>
```

Leads page sample: dynamically creating and sorting a table

If your application is data intensive, and you want the user to be able to sort the data after it is received, then using the full power of DHTML to create, update, and rearrange HTML elements on a page can be superior to dynamically generating a page on the server that becomes a static page on the user's mobile device. The page in this sample application, *leads.html*, dynamically creates and sorts four tables. The workings of this page are described in detail in “Leads page: dynamically creating and sorting a table” on page 146.

The full source code for *leads.html* is listed below. For instructions on downloading this source code, see “Downloading and accessing the JavaScript engine sample files” on page 350.

```
<html>

<head>

<title>Lead Manager</title>

<meta name="HandheldFriendly" content="True">

<script language="JavaScript1.2">

<!--

//
// FILE: leads.html
//
// Revision Date: March 15, 2002
//
// Copyright (c) 2006, iAnywhere Solutions, Inc.
//
//-->

<!--
    //
    // Globals
    //
    var oldTD = null;

    //
    // Lead Manager: Leads tab data
    //
    var intLM_LeadsTab = 1;
    var arrLM_Leads =[
        ["Name", "Company", "Status"],
        ["Chambers, B.", "Starco", "Open"],
        ["Johnson, E.", "J & A Inc.", "Closed"],
        ["Perez, R.", "Lux. P&T", "Working"],
        ["Smyth, S.", "Gen. Comp.", "Contacted"],
        ["Weller, P", "Lexin Ind.", "Open"],
        ["Williams, M.", "Telco GBMH", "Working"],
        ["Rock, F.", "US Airways", "Open"],
        ["Webb, S.", "Air, Inc.", "Closed"]
    ];

    //
    // Lead Manager: Accounts tab data
    //
    var intLM_AccountsTab = 2;
    var arrLM_Accounts =[
        ["Name", "Region", "Phone", "Owner"],
        ["BBI", "Europe", "44 2345...", "JB"],
        ["FTel", "US West", "1 310 55...", "DB"],
        ["IBT Co.", "Japan", "81 4567...", "JB"],
        ["KDNY", "US East", "1 212 55..", "DB"],
        ["Tel Ital", "Europe", "39 4567...", "AF"],
        ["ZedCo", "US East", "1 617 55...", "JB"],
        ["SingP", "Asia", "39 4567...", "AF"],
        ["ManilaTel", "Asia", "44 3454...", "JB"]
    ];

    //
    // Lead Manager: Contacts tab data
    //
    var intLM_ContactsTab = 3;
    var arrLM_Contacts =[["Name", "Company", "Status"],
        ["Chambers, B.", "Starco", "Open"],
        ["Johnson, E.", "J & A Inc.", "Closed"],
        ["Perez, R.", "Lux. P&T", "Working"],
        ["Smyth, S.", "Gen. Comp.", "Contacted"],
        ["Weller, P.", "Lexin Ind.", "Open"],
        ["Williams, M.", "Telco GBMH", "Working"],
        ["Rock, F.", "US Airways", "Open"],
        ["Webb, S.", "Air, Inc.", "Closed"]
    ];

    //
    // Lead Manager: Opportunities tab data
    //
    var intLM_OppsTab = 4;
    var arrLM_Opps =[["Name", "Amount ($)", "Close Date"],
        ["InfoCorp", "35k", "6/12/01"],
        ["BBI", "170k", "Closed"],
        ["BBI", "390k", "6/17/01"],
        ["FTEL", "1000k", "8/23/01"],
        ["ZBT Corp", "48k", "10/14/01"],
        ["IndiaTel", "275k", "10/20/01"],
        ["British Telecom", "1135k", "12/12/01"],
        ["Lucent", "35k", "12/19/01"]
    ];

    //
    // FUNCTION: Initialize()
    //
    function Initialize()
    {

        var i;
        var objMasterTabsDIV;
        var objDIV;
        var objTABLE;
        var objTBODY;
        var objTR;
        var objTD;
        var objBR;

        if (arguments.length != 0)
            throw new Error("function Initialize()
                incorrectly called");

        //
        // Setup the four Lead Manager tabs.
        //
```

```
objTABLE = document.getElementById("TableTabs");
objTR = objTABLE.firstChild.childNodes[0];
objTR.align = "center";
objTD = objTR.childNodes[0];
oldTD = objTD;
SetTabColors(objTD, 1);

//
// Initially the first Lead Manager tab is selected.
//
for (i = 1; i <= 3; i++)
SetTabColors(objTR.childNodes[i], 0);

//
// Dynamically generate the four Lead Manager tables.
//
objMasterTabsDIV = document.getElementById("TabTablesDiv");

for (i = 1; i <= 4; i++)
{
    //
    // DIV
    //
    objDIV = document.createElement("DIV");
    objMasterTabsDIV.appendChild(objDIV)
    objDIV.id="MainT" + i + "Div";
    objDIV.style.display = (i == 1) ? "block" : "none";

    //
    // DIV => TABLE
    //
    objTABLE = document.createElement("TABLE");
    objDIV.appendChild(objTABLE);
    objTABLE.id = "Table" + i;
    objTABLE.border = "1";
    objTABLE.cellSpacing = "0";
    objTABLE.width = "134px";
    objTABLE.style.fontSize = "8px";

    //
    // DIV => TABLE => TBODY
    //
    objTBODY = document.createElement("TBODY");
    objTABLE.appendChild(objTBODY);

    //
    // DIV => TABLE => TBODY
    //     => BR
    //
    objBR = document.createElement("BR");
    objDIV.appendChild(objBR);
} // for

//
// Dynamically load the data into the Lead
// Manager tables.
//
// Adds the TR and TD elements to the above
// tables.
//

LoadLeadManagerTable("Table1", arrLM_Leads);
LoadLeadManagerTable("Table2", arrLM_Accounts);
LoadLeadManagerTable("Table3", arrLM_Contacts);
LoadLeadManagerTable("Table4", arrLM_Opps);
```

```
    return;
} // Initialize

//
// FUNCTION: InitializeNav()
//
function InitializeNav()
{

    if (arguments.length != 0)
        throw new Error("function InitializeNav()
            incorrectly called");

    Initialize();

    //
    // By default, the navigation bar is not displayed.
    //
    HideNav();

    //
    // Show the navigation URLs that are at the bottom of the HTML
    document.
    //
    document.getElementById("navurls").style.display = "block";

    return;

} // InitializeNav

//
// FUNCTION: ShowNav()
//
// Purpose: Show the dropdown navigation bar.
//
function ShowNav()
{

    if (arguments.length != 0)
        throw new Error("function ShowNav()
            incorrectly called");

    document.getElementById('openNav').style.display="block";
    document.getElementById('closedNav').style.display="none";
    document.getElementById('nav').style.display="block";

    return;

} // ShowNav

//
// FUNCTION: HideNav()
//
// Purpose: Hide the dropdown navigation bar.
//
function HideNav()
{

    if (arguments.length != 0)
        throw new Error("function HideNav()
            incorrectly called");

    document.getElementById('openNav').style.display="none";
    document.getElementById('closedNav').style.display="block";
    document.getElementById('nav').style.display="none";
    return;

} // HideNav

function ArrayBubbleSort(arrValuesIN, intColumn)
{

    var temp;
    var i;
    var j;
    var intLength;

    if (arguments.length != 2)
        throw new Error("function ArrayBubbleSort()
            incorrectly called");

    intLength = arrValuesIN.length

    //
    // Skip the first row: we do not want to sort the table
    column headings.
    //
    for (i = 1; i < (intLength-1); i++) {

        for (j = i + 1; j < intLength; j++) {

            if (arrValuesIN[j][intColumn] < arrValuesIN[i][intColumn])
            {
                temp = arrValuesIN[i];
                arrValuesIN[i] = arrValuesIN[j];
                arrValuesIN[j] = temp;
            } // if

        } // for j

    } // for i

    return;

} // FUNCTION: ArrayBubbleSort


//
// FUNCTION: SortLeadManagerTableElements
//
function SortLeadManagerTableElements(objTargetTBODY,
    arrValues, intColumn)
{
    var i;
    var j;
    var intColumns;
    var intRows;

    if (arguments.length != 3)
        throw new Error("function SortLeadManagerTableElements()
            incorrectly called");

    ArrayBubbleSort(arrValues, intColumn);

    intColumns = arrValues[0].length;
    intRows = arrValues.length;

    //
    // Reload the table with the sorted values.
```

```
        //
        for (i = 1; i < intRows; i++)
        {

            var objRow = objTargetTBODY.childNodes[i];
            for (j = 0; j < intColumns; j++)
            {

                var objCell = objRow.childNodes[j];

                var objText = objCell.childNodes[0];

                objText.nodeValue = arrValues[i][j];

            } /// for j (each column in the row)

        } // for i (each row)

        return;

    } // SortLeadManagerTableElements

    //
    // FUNCTION: SortLeadManagerTable
    //
    function SortLeadManagerTable(event)
    {

        var objTD;
        var objTBODY;
        var intColumn;
        var strTableID;

        //
        // HTML event handlers assigned using
        // JavaScript are only passed 1 argument:
        // an event object.
        //
        if (arguments.length > 1)
            throw new Error("function SortLeadManagerTable()
                incorrectly called");

        //
        // Identify the column and table that needs
        // to be sorted.
        //
        objTD = this;

        intColumn = objTD.cellIndex;

        //
        // TD => TR => TBODY => TABLE => TABLE.id
        //
        objTBODY = objTD.parentNode.parentNode;
        strTableID = objTBODY.parentNode.id;

        //
        // Sort the correct table.
        //
        switch (strTableID) {

            case "Table1":
                SortLeadManagerTableElements(objTBODY,
                arrLM_Leads, intColumn); break;
            case "Table2":
                SortLeadManagerTableElements(objTBODY,
                arrLM_Accounts, intColumn); break;

            case "Table3":
                SortLeadManagerTableElements(objTBODY,
                arrLM_Contacts, intColumn); break;

            case "Table4":
                SortLeadManagerTableElements(objTBODY,
                arrLM_Opps, intColumn); break;

            default:
                alert("Demo Application Error:
                SortLeadManagerTable - switch"); break;

        } // switch

    return;

    } // SortLeadManagerTable

    //
    // FUNCTION: LoadLeadManagerTable()
    //
    function LoadLeadManagerTable(strTableName, arrTableData)
    {

        var objTable;
        var eleTR;
        var fChangeColor;
        var intTableRows;
        var intTableCols;
        var i;

        if (arguments.length != 2)
            throw new Error("function LoadLeadManagerTable()
                incorrectly called");

        fChangeColor = 0;

        objTable = document.getElementById(strTableName);

        intInitialRows = objTable.rows.length

        //
        // Delete the existing rows in the table.
        //
        intTableRows = objTable.ROWS;

        for (i = 0; i < intTableRows; i++)
            objTable.deleteRow(0);

        //
        // Add the records to the table.
        //
        fChangeColor = 0;

        //
        // How many columns does this table have?
        //
        if (arrTableData.length > 0)
        intTableCols = arrTableData[0].length
```

```
        //
        // Load the data into the table.
        //
        for (i = 0; i < arrTableData.length; i++)
        {

            //
            // Insert a new table row for each record: adds a TR element.
            //
            eleTR = objTable.insertRow(i);

            if (i == 0) {
                eleTR.style.color = "white";
                eleTR.align = "center";
                eleTR.valign = "top";
              }
            else {
                eleTR.style.color = "black";
                }

        //
        // Add cells to the row: add TD elements.
        //
        for (j = 0; j < intTableCols; j++)
        {

            cell = eleTR.insertCell(j);
            cell.border = 1;

            if (i == 0) {
                cell.onclick = SortLeadManagerTable;
                cell.style.backgroundColor = "navy";
            }
            else {
                cell.style.backgroundColor =
                    (fChangeColor == 1) ? "aqua" : "white";
            }

            cell.appendChild(
                document.createTextNode(
                    arrTableData[i][j]));

        } // for

        //
        // Create the alternating row colors.
        //
        if (fChangeColor == 1)
            fChangeColor = 0;
        else
            fChangeColor = 1;

        } // for add rows

        return;

    } // LoadLeadManagerTable

    //
    // FUNCTION: SetTabColors()

    //
    function SetTabColors(inTD, fState)
    {
        if (fState == 1)
            inTD.style.backgroundColor = "white";
        else
            inTD.style.backgroundColor = "silver";

        return;

    } // SetTabColors

    //
    // FUNCTION: ShowMain()
    //
    function ShowMain(newTD)
    {
        if (arguments.length != 1)
            throw new Error("function ShowMain()
                incorrectly called");

        if (newTD == oldTD)
            return;

    //
    // Hide the parent DIV element for the previously selected table.
    //
    if (oldTD != null) {
        document.getElementById('Main' + oldTD.id +
        'Div').style.display = 'none';
        SetTabColors(oldTD, 0);
        SetTabColors(newTD, 1);
    } // if

    //
    // Show the parent DIV element for the selected table.
    //
    document.getElementById('Main' + newTD.id + 'Div').style.display
    = 'block';

    oldTD = newTD;
        return;

    } // ShowMain

//-->

</script>

</head>

<body onload="InitializeNav()"
    style="font-family:verdana,geneva,arial;font-size:8px;color:#000000;
    margin-left:0px; margin-top:0px;">

<table border="0" width="144" style="font-size: 8px; COLOR: #000000;">

<tr>
    <td valign="top" width="128">
    <img src="logo.gif" width="118" height="34" border="0">
    <img src="darkdotline.gif" width="118" height="5" border="0">
    <br>

    <span style="COLOR: #333333; FONT-SIZE: 8px;">
    LEAD MANAGER
    </span>
    <br>
    <img src="darkdotline.gif" width="118" height="5" border="0">
    <!-- Tabs -->

    <table id="TableTabs" border="1px" cellspacing="0px"
       width="120" style="font-size:8px;">

    <tr>

        <td onclick="ShowMain(this)" id=T1>Leads</td>
        <td onclick="ShowMain(this)" id=T2>Accts</td>
        <td onclick="ShowMain(this)" id=T3>Contacts</td>
        <td onclick="ShowMain(this)" id=T4>Opps</td>
    </tr>

    </table>

    <div id="TabTablesDiv">

    <!-- ----------- START TABS CONTENT ----------- -->

    </div>

    </td>


    <td valign="top" align="right" width="16">

    <div id="openNav" onclick="HideNav()">
    <img src="open.gif" width="16" height="11" border="0">
    </div>

    <div id="closedNav" onclick="ShowNav()">
    <img src="closed.gif" width="16" height="11" border="0">
    </div>

    <div id="nav">
    <img src="docs_nav.gif" width="16" height="92" usemap="#nav" border="0">
    </div>

    </td>

</tr>

</table>

<map name="nav">
<area shape="RECT" coords="0,0,15,15" href="index.html">
<area shape="RECT" coords="0,19,15,34" href="calendar.html">
<area shape="RECT" coords="0,38,15,53" href="chart.html">
<area shape="RECT" coords="0,57,15,72" href="mail.html">
<area shape="RECT" coords="0,76,15,91" href="leads.html">
</map>

<div ID="navurls">

<table border="0" width="120"
    style="FONT-SIZE: 8px; PADDING-BOTTOM: 1px; PADDING-TOP: 1px;
    FONT-FAMILY: verdana,geneva,arial">

    <tr align="center" valign="top" >
        <td><A href="index.html">Home</A></td>
        <td><A href="calendar.html">Cal</A></td>
        <td><A href="chart.html">Chart</A></td>
        <td><A href="mail.html">Mail</A></td>
        <td><A href="leads.html">Leads</A></td>
    </tr>
```

```
</table>
</div>

</body>
</html>
```

Appendix D

Sample code

Contents

Overview of sample code

This appendix contains complete code listings for all of the examples described in the text of this guide. For the code listings that are also available online, a link is provided immediately below the main heading for the sample in this appendix.

For a description of how each listing is used in the example, follow the link provided above the first listing in each section.

Note
These listings are also cross referenced from the section of the text where they are described.

Online sample code

Some of the sample code that is currently available for M-Business Anywhere are can be downloaded off the iAnywhere Solutions web site. This section lists those samples, along with a brief description and the URL from which you can download the files.

CustDB sample UltraLite application

Uses a customer database sample application to illustrate some key features of UltraLite. Available online at:

http://www.ianywhere.com/developer/product_manuals/sqlanywhere/0902/en/html/ulfoen9/00000017.htm

Custom launcher

Provides examples of how to create a custom launcher for M-Business Client: on Palm OS devices without having to use a 3GL compiler; on Microsoft OS devices using a C compiler. Available online at:

http://www.ianywhere.com/developer/code_samples/mbus_customlauncher.html

Date/Time Picker

For the text describing this sample code, see “Using the Date/Time Picker” on page 214.

Demonstrates use of the Date/Time Picker to easily add user-friendly date and time selection to your M-Business Client application. Available online at:

http://www.ianywhere.com/developer/code_samples/dtpicker.html

Dynamic HTML - DocWrite

Dynamic HTML sample code that dynamically creates and displays elements using DOM 1.0, HTML 4.0, CSS 1.0, and JavaScript. Available online at:

http://www.ianywhere.com/developer/code_samples/mbus_dhtml2.html

Dynamic HTML - ExecDash

For the text describing this sample code, see “Using DHTML” on page 138.

This sample application illustrates some of the features that you can implement with DHTML, including expanding/collapsing menus and text hierarchies, an interactive calendars and charts, form submissions, and dynamically sorting a table. Available online at:

http://www.ianywhere.com/developer/code_samples/mbus_dhtml.html

Master-Detail page

For the text describing this sample code, see:

- “Using the List Viewer with M-Business XML datastore” on page 118
- “Using the PODSPrefs object” on page 124

A master page provides summary level information on the customer, account, product, or whatever category of information is represented. From this master page, the user can drill down and see a more detailed level of information for a selected entry on the master page. In desktop applications, the detail page may be displayed in a special area at the bottom of the master page; in mobile applications, the detail page typically has to be a completely separate page, and therefore you have to maintain the application state as the user switches between the master and detail pages. Available online at:

http://www.ianywhere.com/developer/code_samples/master_detail_mbus.html

One-button synchronization of UltraLite and M-Business Anywhere using MobiLink Redirector

For the text describing this sample code, see “Sample application using UltraLite for M-Business Anywhere” on page 250.

This sample application illustrates use of UltraLite for M-Business Anywhere for on-device data. It uses the MobiLink Redirector and one-button synchronization features of M-Business Client, version 5.5 and higher, along with MobiLink in Adaptive Server Anywhere, version 9.02. Available online at:

http://www.ianywhere.com/developer/code_samples/btn_redir_sample.html

Session POD sample binary files for Pocket PC and Palm

Provides an additional options for developers who need to pass values between Dynamic HTML documents. Once the application terminates, all values stored within the session POD are deleted. Available online at:

http://www.sybase.com/detail?id=1027590

Serializing form submissions with M-Business Client

For the text describing this sample code, see “Serializing forms: breaking one form into several pages” on page 195.

This pair of code samples illustrate two approaches to breaking a single form into multiple logical pages: DHTML to selectively hide and display sections of a single form; and DOM to build sections of a form in the Forms Manger. Available online at:

http://www.ianywhere.com/developer/code_samples/mbus_serializing_forms.html

Form submission via HTML - complex example

For a simple example of form submission via HTML, see "Example: setting forms property attributes" on page 328.

```
<html>
<head>
<title>submitForm1_new</title>
<meta NAME="HandheldFriendly" CONTENT="True">
</head>
<body>

<h3>Testing: agsubmitMessage, agsubmitOffsiteLinks, agsubmitIncludeImages,
    agsubmitLinkDepth, agsubmitSize, agsubmitDiscardResponse,
        agsubmitMultiple,
    agsubmitHidden, agsubmitDisplayDefaultMessage.</h3>

<p>What to expect:<br>


1. agsubmitMessage=null <br>
2. agsubmitOffsiteLinks=false<br>
3. agsubmitIncludeImages=false: No image should be downloaded.<br>
4. agsubmitLinkDepth=0: all links are dead from the response page<br>
5. agsubmitSize=100<br>
6. agsubmitDiscardResponse=false: response pages available<br>
7. agsubmitMultiple=true: the text field will not retain the old value<br>
8. agsubmitHidden=false: form will appear in the Forms Manager<br>
9. agsubmitDisplayDefaultMessage=false: No default form submission
        message.<br>

<form action=http://search.yahoo.com/bin/search
agsubmitMessage=null
agsubmitOffsiteLinks=false
agsubmitIncludeImages=false
agsubmitLinkDepth=0
agsubmitSize=100
agsubmitDiscardResponse=false
agsubmitMultiple=true
agsubmitHidden=false
agsubmitDisplayDefaultMessage=false>

<input size=15 name=p><BR>
<INPUT TYPE=submit VALUE="Search">
<INPUT TYPE=reset>
</form>

1. Do offline submit. You should see a "null" alert.

</body>
</html>
```

Movie Review Channel example

Basic version

For the text describing these sample files, see "Cookie personalization example: a Movie Review Channel" on page 196.

frontpage.pl, basic version

```
#!D:\Perl\bin\perl.exe -w
use CGI qw(:standard);
use CGI::Cookie;


# custom content for each M-Business Anywhere user.
#
# This is your basic Movie News front page. It looks up
# the user's preferences for movies he likes, and
# provides links to, different movie reviews if they
# fit into his list of categories. If this is the user's
# first time here, it provides him with some default
# categories.
#

# The basic strategy is this: We look for a cookie called
# moviereviewID.
#
# If it exists, we take the value of that
# cookie (the user's ID) and look it up in a database.
# That database contains a list of the user's preferred
# channels.
#

# If we don't find a cookie, we just give them some
# default content.
#
# PLEASE NOTE: This is a script running under ActivePerl
# on Win2k. Please change the first line to reflect
# whatever system you're running Perl under. Please read
# the legal message at the bottom.
#


# Prints out some standard HTTP headers.

print header(-'Cache-Control'=>'private');

# Gets all the cookies and puts them into a hash.

%cookies = fetch CGI::Cookie;


if ($cookies{'moviereviewID'} ) {


    # If a cookie exists, we get its value (which is the
    # user's ID number)
    $myID = $cookies{'moviereviewID'} -> value;

dbmopen(%prefs,"movieuserprefs",0644) ||
        die "Can't open movieuserprefs! $!";


    # We look up the user's preferences based on his ID
    # in the database.
    #
    # The record itself is a big string of values
    # separated by ~:~ characters.
    # The "split" function breaks up that string and
    # places the resulting values into the variable
    # "propername" and the array "prefs"

    ($propername, @prefs) = split(/~:~/, $prefs{$myID});

    dbmclose(%prefs);

} else {

    # We make prefs an array of default values if the
    # user's cookie does not exist.

    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);

    # Flag indicates we're using default values.

    $default = 1;

}


# Now we print out the beginning of the web page.

print <<END_of_Start;

<HTML>
    <HEAD>
    <TITLE>Movie reviews</title>
    <META NAME="HandheldFriendly" content="True">
    </head>
    <body>
    <Center><h2>The movie review channel!</h2></center>

END_of_Start


print "<p>Welcome, $propername </p>" unless $default;
print "<p>Here are a list of movies you might be
    interested in...</p>\n <ul>\n";


dbmopen(%movies,"moviereviews",0644) || die "Can't open moviereviews! $!";


# We iterate over movie ID within the moviereviews
# database.

foreach $filmnumber (keys (%movies)) {

    # Again, breaking up a string into the variable
    # "filmname" and the "genres" array using a split
    # command.
    ($filmname, @genres) = split(/~:~/, $movies{$filmnumber});

    $match = 0;
    for ($i =0; $i < 11; $i++) {
        if ($genres[$i] && $prefs[$i]) {
            $match = 1;     # We've found a match!
            last;           # Break out of the loop.
                            # no need to keep looking.
        }
    }

    # If this film has a match, include a link to it.

    if ($match) {
        print "<li><A href=\"reviews/${filmnumber}.html\">$filmname</a>\n";
    }

}


dbmclose(%movies);

print "</ul>\n<p>Tune in next time for more movie reviews!<p>\n";

if ($default) {
     # Informative message telling them to personalize
     # their content
    print "<center>Want personalized content? Of course you do!\n";
    print "<br><a href=\"/cgi-bin/movies/editprefsform.pl\
        ">Edit your preferences</a></center>\n";

} else {

     # They've already personalized, so we can be a little
     # more understated here.
    print "<a href=\"/cgi-bin/movies/editprefsform.pl\
        ">Edit preferences</a>\n";
}

print "</body>\n</html>\n";


# And now, the important legal message...
#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

editprefsform.pl, basic version

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;
```

```
#        editprefsform.pl
#
# Sample code to display an adjust-movie-preferences form
#
# This displays a form where the user can pick and choose
# what movie categories he would like to subscribe to.
#
# The reason this is a perl script (instead of a static
# page) is that we can check by default the categories the
# person is already subscribed to.
#


%cookies = fetch CGI::Cookie;


if ($cookies{'moviereviewID'} ) {

    # If there is a cookie for this person, then he's
    # already made some selections. We grab this info out
    # of the database.

    $myID = $cookies{'moviereviewID'} -> value;

    dbmopen(%prefs,"movieuserprefs",0644) || die "Can't
        open movieprefs! $!";

    ($propername, @prefs) = split(/~:~/, $prefs{$myID});

} else {

    # OK, no cookie available. We could either have
    # none of the categories checked, or the default
    # ones. Here, I'm going with the default ones.

    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);

    $default = 1;

}

print header(-'Cache-Control'=>'private');

print <<END_of_FormTop;
<HTML>
    <HEAD>
        <TITLE>Get customized</TITLE>
        <META NAME="HandheldFriendly" content="True">
    </HEAD>
<BODY>
    <p>Please indicate what movies you would like to receive
        reviews for:</p>
    <form name="choosemovies" action="/cgi-bin/movies/setprefs.pl" >
    <p>

END_of_FormTop

# We iterate through each category, providing a checkbox
# for each. If the user already has this category selected
# in his preferences, we add a "checked" to the form.

@genrenames = ( "Action/Adventure",
    "Art House/Indie",
    "Classics",
    "Comedy",
    "Drama",
    "Horror and Suspense",
    "Kids and Family",
    "Musicals",
    "Sci-Fi/Fantasy",
    "Special Interest",
    "TV/Documentary"
    );

for ($i = 0; $i<11; $i++) {
    print "<input type=\"checkbox\" name=\"$i\" value=\"1\"";
    if ($prefs[$i]) {
        print " checked";
    }
    print ">\n$genrenames[$i]<br>\n";
}

print "<p>\nPlease enter your first name.\n";
print "<input type=\"text\" name=\"firstname\" size=\"15\"
    maxlength=\"15\" value=\"$propername\">\n </p>\n"

# That's it! Now we just print the rest of the page.

print <<END_of_Page;


    <input type="submit" name="submit" value="Submit">

    </form>

    </BODY>
    </HTML>

END_of_Page


# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

setprefs.pl, basic version

```
#!D:\Perl\bin\perl.exe
use CGI qw/:standard/;
use CGI::Cookie;

#
#            setprefs.pl
#
# Sample Perl script that processes the result of
# editprefsform.pl to set preferences for each
# M-Business Anywhere user.
#
# We grab the username and preferences from the form
# submitted to us and we place the results in our
# database.
#
# We also create a cookie with the user's uniqueID and set
# that, just in case.

srand;

# The first thing we do is grab the preferences from the
# submitted form and store them in an array.

for ($i=0; $i<11; $i++) {
    $prefs[$i] = param($i) || "0";
}

# We also grab the user's firstname and strip out any
# weird pattern-breaking characters.

$username = param("firstname");

$username =~ s/~:~/~;~/g;

# If they have an ID already, we use it. Otherwise, we
# give them one.

%cookies = fetch CGI::Cookie;
if ($cookies{'moviereviewID'} ) {
    $myID = $cookies{'moviereviewID'}->value;
} else {
    $myID = getrandomID();
}

# We create the cookie, with a name, ID, and expires time
# of 10 years.

$tkcookie = new CGI::Cookie(-name=>'moviereviewID',-value=>$myID,
          -expires=>'+10y');

# We write the user's preferences into the database.

dbmopen(%prefs, "movieuserprefs", 0644) || die "cannot open DBM $!";

$prefstring = $username. "~:~". join ("~:~", @prefs);

$prefs{$myID} = $prefstring;

dbmclose(%prefs);

# Now, we use the Set-Cookie HTTP heading to actually set
# the cookie. This is what the M-Business Sync Server needs
# to set the value of a cookie.

print "Content-type: text/html\n";
print "Set-Cookie: $tkcookie\n";
print "Cache-Control: private\n";

# Nobody should ever see this form, becuase we're sending
# this with submitnoresponse. But I keep it around for
# debugging purposes.

print <<END_of_Response;
```

```
<HTML>
    <HEAD>
    <TITLE>Cookie has been set</TITLE>
    </HEAD>
    <BODY>
    Your cookie has been set. <p>

END_of_Response

print "This is debug information you'll never see. \n";
print "Your prefs string is $prefstring <p> \n";
print "And your ID is $myID \n </body></HTML>";

# This used to be a bad idea because srand used time() as
# its seed. In later versions of Perl, it doesn't.

sub getrandomID {
    return (time(). sprintf("%07d",(int(rand(5000000)))));
}

# And now, the important legal message...
#
# Copyright (c) 2000, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

Version 2, supporting online users

For the text describing these sample files, see "Accommodating online users" on page 182.

frontpage.pl, version supporting online users

```
#!D:\Perl\bin\perl.exe -w
use CGI qw(:standard);
use CGI::Cookie;


#
#            frontpage.pl
#
# Sample Perl script that uses cookies
# to create custom content for each M-Business Anywhere user.
#
# This is your basic Movie Reviews front page. It looks up
# the user's preferences for movies he likes, and
# provides links to, different movie reviews if they
# fit into his list of categories. If this is the user's
# first time here, it provides him with some default
# categories.
#
```

```
# The basic strategy is this: We look for a cookie called
# moviereviewID.
#
# If it exists, we take the value of that cookie (the
# user's ID) and look it up in a database. That database
# contains a list of the user's preferred channels.
#
# If we don't find a cookie, we just give them some
# default content.
#
# PLEASE NOTE: This is a script running under ActivePerl
# on Win2k. Please change the first line to reflect
# whatever system you're running Perl under. Please read
# the legal message at the bottom.
#


# Prints out some standard HTTP headers.

print header(-'Cache-Control'=>'private');

# Gets all the cookies and puts them into a hash.

%cookies = fetch CGI::Cookie;

if ($cookies{'moviereviewID'} ) {

# If a cookie exists, we get its value (which is the
    # user's ID number) and look up his preferences in a
    # database.
    #
    # The preferences are pretty kludgy. We basically
    # grab a list of six strings split apart by colons and
    # stick them into an array.

    $myID = $cookies{'moviereviewID'} -> value;
    dbmopen(%prefs,"movieuserprefs",0644) || die "
         Can't open movieprefs! $!";
    ($propername, @prefs) = split(/~:~/, $prefs{$myID});
    dbmclose(%prefs);
} else {
    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);
    $default = 1;
}


# Now, we print out the beginning of the web page.

print <<END_of_Start;


<HTML>
    <HEAD>
    <TITLE>Movie reviews</title>
    <META NAME="HandheldFriendly" content="True">
    </head>
    <body>
    <Center><h2>The movie review channel!</h2></center>
    <p>Version2.0</p>


END_of_Start
```

```
#  print "<p>Your Id is $myID\n";
#  print "<p>Your prefs is @prefs\n";

print "<p>Welcome, $propername </p>" unless $default;
print "<p>Here are a list of movies you might be interested
        in...</p>\n <ul>\n";

dbmopen(%movies,"moviereviews",0644) || die "Can't open movieprefs! $!";

foreach $filmnumber (keys (%movies)) {
    ($filmname, @genres) = split(/~:~/, $movies{$filmnumber});
    $match = 0;
    for ($i =0; $i < 11; $i++) {
        if ($genres[$i] && $prefs[$i]) {
            $match = 1;
        }
    }

    if ($match) {
        print "<li><A href=\"reviews/${filmnumber}.html\">$filmname</a>\n";
    }
}


dbmclose(%movies);

print "</ul>\n<p>Tune in next time for more movie reviews!<p>\n";

if ($default) {

# Informative message telling them to personalize
    # their content
    print "<center>Want personalized content? Of course you do!\n";
    print "<br><a href=\"/cgi-bin/movies2/editprefsform.pl\">
        Edit your preferences</a></center>\n";

} else {

    # They've already personalized, so we can be a little
    # more understated here.
    print "<a href=\"/cgi-bin/movies2/editprefsform.pl\">
        Edit preferences</a>\n";

}

print "</body>\n</html>\n";


# And now, the important legal message...

#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

editprefsform.pl, version supporting online users

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;


#
#        editprefsform.pl
#
# Sample code to display an adjust-movie-preferences form
#
# This displays a form where the user can pick and choose
# what movie categories he would like to subscribe to.
#

# The reason this is a perl script (instead of a static
# page) is that we can check by default the categories the
# person is already subscribed to.
#

%cookies = fetch CGI::Cookie;

if ($cookies{'moviereviewID'} ) {

    # If there is a cookie for this person, then he's
    # already made some selections. We grab this info out
    # of the database.
    $myID = $cookies{'moviereviewID'} -> value;
    dbmopen(%prefs,"movieuserprefs",0644) || die "
         Can't open movieprefs! $!";
    ($propername, @prefs) = split(/~:~/, $prefs{$myID});

} else {

    # OK, no cookie available. We could either have
    # none of the categories checked, or the default
    # ones. Here, I'm going with the default ones.
    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);
    $default = 1;

}

print header(-'Cache-Control'=>'private');

print <<END_of_FormTop;
<HTML>
    <HEAD>
        <TITLE>Get customized</TITLE>
        <META NAME="HandheldFriendly" content="True">
    </HEAD>
<BODY>

<p>Please indicate what movies you would like to receive news for:</p>
<form name="choosemovies" action="/cgi-bin/movies2/setprefs.pl" >
<p>

END_of_FormTop

# We iterate through each category, providing a checkbox
# for each. If the user already has this category selected
# in his preferences, we add a "checked" to the form.

@genrenames = ( "Action/Adventure",
    "Art House/Indie",
    "Classics",
    "Comedy",
    "Drama",
    "Horror and Suspense",
    "Kids and Family",
    "Musicals",
    "Sci-Fi/Fantasy",
    "Special Interest",
    "TV/Documentary"
    );

for ($i = 0; $i<11; $i++) {
    print "<input type=\"checkbox\" name=\"$i\" value=\"1\"";
    if ($prefs[$i]) {
        print " checked";
    }
    print ">\n$genrenames[$i]<br>\n";
}

print "<p>\nPlease enter your first name.\n";
print "<input type=\"text\" name=\"firstname\" size=\"15\"
    maxlength=\"15\" value=\"$propername\">\n </p>\n";


# That's it! Now we just print the rest of the page.

# Note that our submit button is a little special here.
# We use a button that calls a "submitNoResponse"
# function, which means the results returned by the script
# are not sent to the Forms Manager. (This is a good
# thing, since all we care about in this next form is
# setting cookies and internal databases.)
#

# The first argument to this function is the text that the
# dialog box will display. The second argument tells us
# "Yes, I really do want you to hide the form from the
# Forms Manager". The third argument tells us "No, you can
# only submit one version of this form per sync" — i.e.
# if I fill out this form again, it will only submit the
# most recent one the next time I sync.

print <<END_of_Page;

    <input type="button" name="submit" value="Submit"

    onClick="document.forms[0].submitNoResponse(
        'Your updated categories will be here next time you sync',
        true,false);back();">

</form>
</BODY>
</HTML>

END_of_Page

# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
```

```
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

setprefs.pl, version supporting online users

```
#!D:\Perl\bin\perl.exe
use CGI qw/:standard/;
use CGI::Cookie;


#
#            setprefs.pl
#
# Sample Perl script that processes the result of
# editprefsform.pl to set preferences for each M-Business Anywhere
# user. #

# We grab the username and preferences from the form
# submitted to us and we place the results in our
# database.
#
# We also create a cookie with the user's uniqueID and set
# that, just in case.

srand;

# The first thing we do is grab the preferences from the
# submitted form and store them in an array.

for ($i=0; $i<11; $i++) {
    $prefs[$i] = param($i) || "0";
}

$username = param("firstname");
$username =~ s/~:~/~;~/g;

# If they have an ID already, we use it. Otherwise, we
# give them one.

%cookies = fetch CGI::Cookie;

if ($cookies{'moviereviewID'} ) {
    $myID = $cookies{'moviereviewID'}->value;
} else {
    $myID = getrandomID();
}

# We create the cookie, with a name, ID, and expires time
# of 10 years.

$tkcookie = new CGI::Cookie(-name=>'moviereviewID',
    -value=>$myID,-expires=>'+10y');

# We write the user's preferences into the database.

dbmopen(%prefs, "movieuserprefs", 0644)
    || die "cannot open DBM $!";
$prefstring = $username. "~:~". join ("~:~", @prefs);
$prefs{$myID} = $prefstring;

dbmclose(%prefs);
```

```
# Now, we use the Set-Cookie HTTP heading to actually set
# the cookie. This is what the  M-Business Sync Server needs
# to set the value of a cookie.

print "Content-type: text/html\n";
print "Set-Cookie: $tkcookie\n";

# Nobody should ever see this form, becuase we're sending
# this with SubmitNoResponse. But I keep it around for
# online users.

print <<END_of_Response;

<HTML>
    <HEAD>
    <META name="HandheldFriendly" content="true">
    <TITLE>Cookie has been set</TITLE>
    </HEAD>
    <BODY>
    Thanks for your submission!
    <p>
    Please go back to the
        <a href="/cgi-bin/movies2/frontpage.pl">
        Front Page to read more movie reviews.
    <p>

END_of_Response

# This used to be a bad idea because srand used time() as
# its seed. In later versions of Perl, it doesn't.

sub getrandomID {
    return (time(). sprintf("%07d",(int(rand(5000000)))));
}


# And now, the important legal message...
#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

Version 3, no database, preferences stored in cookie

frontpage.pl, no-database version

```
#!D:\Perl\bin\perl.exe -w
use CGI qw(:standard);
use CGI::Cookie;
```

```
#
#            frontpage.pl
#
# Sample Perl script that uses cookies
# to create custom content for each M-Business Anywhere user.
#
# This is your basic Movie Reviews front page. It looks up
# the user's preferences for movies he likes, and
# provides links to, different movie reviews if they
# fit into his list of categories. If this is the user's
# first time here, it provides him with some default
# categories.

#
# The basic strategy is this: We look for a cookie called
# moviereviewID.
#
# If it exists, we take the value of that cookie (the
# user's ID) and look it up in a database. That database
# contains a list of the user's preferred channels.
#

# If we don't find a cookie, we just give them some
# default content.
#
# PLEASE NOTE: This is a script running under ActivePerl
# on Win2k. Please change the first line to reflect
# whatever system you're running Perl under. Please read
# the legal message at the bottom.
#

# Prints out some standard HTTP headers.

print header(-'Cache-Control'=>'private');

# Gets all the cookies and puts them into a hash.

%cookies = fetch CGI::Cookie;

if ($cookies{'moviereviewprefs'} ) {

    # If a cookie exists, we get its value (which is the
    # user's ID number) and look up his preferences in a
    # database.
    #
    # The preferences are pretty kludgy. We basically
    # grab a list of six strings split apart by colons and
    # stick them into an array.

    ($propername, @prefs) = split(/~:~/,
         $cookies{'moviereviewprefs'} -> value);

} else {

    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);
    $default = 1;

}

# Now, we print out the beginning of the web page.

print <<END_of_Start;
```

```
<HTML>
    <HEAD>
    <TITLE>Movie reviews</title>
    <META NAME="HandheldFriendly" content="True">
    </head>
    <body>
    <Center><h2>The movie review channel!</h2></center>
    <p>Version2.0</p>

END_of_Start


#  print "<p>Your Id is $myID\n";
#  print "<p>Your prefs is @prefs\n";

print "<p>Welcome, $propername </p>" unless $default;
print "<p>Here are a list of movies you might be interested
        in...</p>\n <ul>\n";

dbmopen(%movies,"moviereviews",0644) || die "Can't open movieprefs! $!";

foreach $filmnumber (keys (%movies)) {
    ($filmname, @genres) = split(/~:~/, $movies{$filmnumber});
    $match = 0;
    for ($i =0; $i < 11; $i++) {
        if ($genres[$i] && $prefs[$i]) {
        $match = 1;
        }
    }

    if ($match) {
        print "<li><A href=\"reviews/${filmnumber}.html\">$filmname</a>\n";
    }
}

dbmclose(%movies);

print "</ul>\n<p>Tune in next time for more movie reviews!<p>\n";

if ($default) {

    # Informative message telling them to personalize their content
    print "<center>Want personalized content? Of course you do!\n";
    print "<br><a href=\"/cgi-bin/movies3/editprefsform.pl\">
        Edit your preferences</a></center>\n";

} else {

    # They've already personalized, so we can be a little
    # more understated here.
    print "<a href=\"/cgi-bin/movies3/editprefsform.pl\">
        Edit preferences</a>\n";

}

print "</body>\n</html>\n";


# And now, the important legal message...
#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

editprefsform.pl, no-database version

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;


#
#        editprefsform.pl
#
# Sample code to display an adjust-movie-preferences form
#
# This displays a form where the user can pick and choose
# what movie categories he would like to subscribe to.
#

# The reason this is a perl script (instead of a static
# page) is that we can check by default the categories the person
# is already subscribed to.
#

%cookies = fetch CGI::Cookie;

if ($cookies{'moviereviewID'} ) {

    # If there is a cookie for this person, then he's
    # already made some selections. We grab this info out
    # of the database
    ($propername, @prefs) = split(/~:~/, $cookies{'moviereviewprefs'} ->
value);

} else {

    # OK, no cookie available. We could either have
    # none of the categories checked, or the default
    # ones. Here, I'm going with the default ones.
    @prefs = (1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0);
    $default = 1;

}

print header(-'Cache-Control'=>'private');

print <<END_of_FormTop;

<HTML>
    <HEAD>
        <TITLE>Get customized</TITLE>
        <META NAME="HandheldFriendly" content="True">
    </HEAD>

<BODY>
    <p>Please indicate what movies you would like to receive news for:</p>
    <form name="choosemovies" action="/cgi-bin/movies3/setprefs.pl" >
    <p>

END_of_FormTop


# We iterate through each category, providing a checkbox
# for each. If the user already has this category selected
# in his preferences, we add a "checked" to the form.

@genrenames = ( "Action/Adventure",
    "Art House/Indie",
    "Classics",
    "Comedy",
    "Drama",
    "Horror and Suspense",
    "Kids and Family",
    "Musicals",
    "Sci-Fi/Fantasy",
    "Special Interest",
    "TV/Documentary"
    );

for ($i = 0; $i<11; $i++) {
    print "<input type=\"checkbox\" name=\"$i\" value=\"1\"";
    if ($prefs[$i]) {
        print " checked";
    }
    print ">\n$genrenames[$i]<br>\n";
}

print "<p>\nPlease enter your first name.\n";
print "<input type=\"text\" name=\"firstname\" size=\"15\"
    maxlength=\"15\" value=\"$propername\">\n </p>\n";

# That's it! Now we just print the rest of the page.

# Note that our submit button is a little special here.
# We use a button that calls a "submitNoResponse"
# funtion, which means the results returned by the script
# are not sent to the Forms Manager. (This is a good
# thing, since all we care about in this next form is
# setting cookies and internal databases.)
#

# The first argument to this function is the text that the
# dialog box will display. The second argument tells us
# "Yes, I really do want you to hide the form from the
# Forms Manager". The third argument tells us "No, you can
# only submit one version of this form per sync" – i.e.
# if I fill out this form again, it will only submit the
# most recent one the next time I sync.

print <<END_of_Page;

<input type="button" name="submit" value="Submit"
onClick="document.forms[0].submitNoResponse(
    'Your updated categories will be here next time you sync',
    true,false);back();" >
</form>

</BODY>
</HTML>

END_of_Page
```

```
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

setprefs.pl, no-database version

```
#!D:\Perl\bin\perl.exe
use CGI qw/:standard/;
use CGI::Cookie;


# The first thing we do is grab the preferences from the
# submitted form and store them in an array.

for ($i=0; $i<11; $i++) {
    $prefs[$i] = param($i) || "0";
}

$username = param("firstname");
$username =~ s/~:~/~;~/g;

# We create the prefs string out of our form results.
$prefstring = $username. "~:~". join ("~:~", @prefs);
# We create the cookie, with a name, preferences string,
# and expires time of 10 years.

$tkcookie = new CGI::Cookie(-name=>'moviereviewprefs',
    -value=>$prefstring,-expires=>'+10y');

# Now, we use the Set-Cookie HTTP heading to actually set
# the cookie. This is what the  M-Business Sync Server needs
# to set the value of a cookie.

print "Content-type: text/html\n";
print "Set-Cookie: $tkcookie\n";

# Nobody should ever see this form, becuase we're sending
# this with submitnoresponse. But I keep it around for
# debugging purposes.
#

print <<END_of_Response;

<HTML>
    <HEAD>
    <META name="HandheldFriendly" content="true">
    <TITLE>Cookie has been set</TITLE>
    </HEAD>
    <BODY>
    Thanks for your submission!
    <p>
    Please go back to the <a href="/cgi-bin/movies3/frontpage.pl">
        Front Page to read more movie reviews.
    <p>
```

```
END_of_Response


# And now, the important legal message...
#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

Wine Finder Channel example

frontpage.pl

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;


#
#            frontpage.pl
#
# Sample script for the findagoodwine.com channel that
# creates the normal front page. We look up the user's
# last three searches in our database, and provide links
# to them. When the  M-Business Sync Server traverses the
# channel, it will follow these links and that's where the
# search actually gets conducted.

# Prints out some standard HTTP headers.

print header(-'Cache-Control'=>'private');

# Gets all the cookies and puts them into a hash.

%cookies = fetch CGI::Cookie;

if ($cookies{'findagoodwineID'} ) {

# If a cookie exists, we get its value (which is the
    # user's ID number) and look up his preferences in a
    # database.

$myID = $cookies{'findagoodwineID'} -> value;
    dbmopen(%history,"winehistory",0644) || die "Can't
         open wineprefs! $!";
    @lastsearch = split(/~:~/, $history{$myID});
    dbmclose(%history);

}

# Now, we print out the beginning of the web page.

print <<END_of_Start;

<HTML>
    <HEAD>
    <TITLE>Find a good wine</title>
    <META NAME="HandheldFriendly" content="True">
    </head>
    <body>
    <Center><h2>The find a good wine channel!</h2></center>

    <ul>
    <li><a href="/examples/findagoodwine/news.html">Wine News</a>
    <li><a href="/examples/findagoodwine/row.html">Review of the Week</a>
    <li><a href="/examples/findagoodwine/searchform.html">Find a wine</a>
    </ul>

END_of_Start
```

```
if (@lastsearch) {
    print "<p>Here are the results of your last three searches...</p>\n";
    print "<ul>";

    foreach $searchstring (@lastsearch) {
        ($searchurl, $searchdesc) = split(/~::~/, $searchstring);
        print "<li><a href=\"$searchurl\">$searchdesc</a>\n";
    }

    print "</ul>\n";

}

print "</body>\n</html>\n";


#
# Copyright (c) 2000, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

searchform.html

```
<html>
<head>
<title>M-Business</title>

    <style type="text/css">
    <!--
    select  {  font-family: Verdana, Arial, Helvetica; font-size: 10px }
    textarea  {  font-family: Verdana, Arial, Helvetica; font-size: 10px }
    input   {  font-family: Verdana, Arial, Helvetica; font-size: 10px }
    -->
    </style>

    <link rel="SHORTCUT ICON" href="/images/ami/favicon.ico">
    <link rel=stylesheet type=text/css href="/dhtml/style.css">

    <script language="JavaScript" TYPE="text/javascript">
    <!--

    if(window.event + "" == "undefined") event = null;
        function HM_f_PopUp(){return false};
        function HM_f_PopDown(){return false};
        popUp = HM_f_PopUp;
        popDown = HM_f_PopDown;

    //-->

    </SCRIPT>
    <SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
    <!--

    HM_PG_MenuWidth = 150;
    HM_PG_FontFamily = "verdana,geneva,arial,sans serif";
    HM_PG_FontSize = 8;
    HM_PG_FontBold = 0;
    HM_PG_FontItalic = 0;
    HM_PG_FontColor = "#003366";
    HM_PG_FontColorOver = "#ffffff";
    HM_PG_BGColor = "#e5e5e5";
    HM_PG_BGColorOver = "#336699";
    HM_PG_ItemPadding = 3;

    HM_PG_BorderWidth = 1;
    HM_PG_BorderColor = "#cccccc";
    HM_PG_BorderStyle = "solid";
    HM_PG_SeparatorSize = 1;
    HM_PG_SeparatorColor = "#cccccc";
    HM_PG_ImageSrc = "/images/nav/arrow.gif";
    HM_PG_ImageSrcLeft = "/images/nav/arrow_r.gif";

    HM_PG_ImageSize = 11;
    HM_PG_ImageHorizSpace = 0;
    HM_PG_ImageVertSpace = 2;

    HM_PG_KeepHilite = true;
    HM_PG_ClickStart = 0;
    HM_PG_ClickKill = 0;
    HM_PG_ChildOverlap = 25;
    HM_PG_ChildOffset = 5;
    HM_PG_ChildPerCentOver = null;
    HM_PG_TopSecondsVisible = 0;
    HM_PG_StatusDisplayBuild =0;
    HM_PG_StatusDisplayLink = 1;
    HM_PG_UponDisplay = null;
    HM_PG_UponHide = null;
    HM_PG_RightToLeft = false;

    HM_PG_CreateTopOnly = 1;
    HM_PG_ShowLinkCursor = 1;
    HM_PG_NSFontOver = true;

    //HM_a_TreesToBuild = [1,2];

    //-->
    </script>
    <script language="JavaScript"><!--
    name = 'mbusinessmainwindow';
    //--></script>

</head>

<body marginwidth="0" marginheight="0" leftmargin="0" topmargin="0"
    bgcolor="#ffffff" link="#993300" vlink="#cc6600" alink="#ff9933" >

<table border="0" cellpadding="0" cellspacing="0" width="780">
    <tr>
        <td colspan="8">&nbsp;</td>
        <td><img src="/images/transparent.gif" width="17" height="1"
            alt="" border="0">
            <img src="/images/nav/avgo_logo.gif" width="165" height="67"
            alt="M-Business" border="0">
            <img src="/images/transparent.gif" width="17" height="1"
            alt="" border="0">
        </td>
    </tr>

    <tr>
        <td><a href="/frontdoor/corpfrontdoor.html">
```

```
            <img src="/images/nav/home.gif" width="46" height="20"
            alt="HOME" border="0"></a>
        </td>

        <td><a href="/products/" onMouseOver="popUp('HM_Menu1',event)"
            onMouseOut="popDown('HM_Menu1')">
            <img src="/images/nav/products_services.gif"
        width="134" height="20"
            alt="PRODUCTS & SERVICES" border="0"></a>
        </td>

        <td><a href="/support/" onMouseOver="popUp('HM_Menu2',event)"
            onMouseOut="popDown('HM_Menu2')">
            <img src="/images/nav/support.gif" width="67" height="20"
            alt="SUPPORT" border="0"></a></td>

        <td><a href="/partners/" onMouseOver="popUp('HM_Menu3',event)"
            onMouseOut="popDown('HM_Menu3')">
            <img src="/images/nav/partners.gif" width="72" height="20"
            alt="PARTNERS" border="0"></a>
        </td>

        <td><a href="http://www.ianywhere.com/developer
            /mbusiness_anywhere.html" onMouseOver="popUp('HM_Menu4',event)"
            onMouseOut="popDown('HM_Menu4')">
            <img src="/images/nav/developers.gif" width="79" height="20"
            alt="DEVELOPERS" border="0"></a>
        </td>

        <!-- <td><a href="/news/" onMouseOver="popUp('HM_Menu4',event)"
            onMouseOut="popDown('HM_Menu4')">
            <img src="/images/nav/news.gif" width="51" height="20"
            alt="NEWS" border="0"></a>
        </td> -->

        <td>
            <a href="/events/" onMouseOver="popUp('HM_Menu5',event)"
                onMouseOut="popDown('HM_Menu5')">
                <img src="/images/nav/events.gif" width="60" height="20"
                alt="EVENTS" border="0"></a>
        </td>

        <td><a href="/company/" onMouseOver="popUp('HM_Menu6',event)"
            onMouseOut="popDown('HM_Menu6')">
            <img src="/images/nav/company.gif" width="67" height="20"
            alt="COMPANY" border="0"></a></td>

        <td><a href="/store/"><img src="/images/nav/store.gif" width="54"
            height="20" alt="HOW TO BUY" border="0"></a>
        </td>

        <!-- <td><a href="/store/"><img src="/images/nav/how_to_buy.gif"
            width="82" height="20" alt="HOW TO BUY" border="0"></a></td> -->

        <td>&nbsp;</td>

    </tr>

    <html>

<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta name="HandheldFriendly" content="true">
</head>
```

```
<body bgcolor="#FFFFFF">
<p>Search for your favorite wine</p>
<form method="GET" name="winesearch" action="/cgi-bin/findagoodwine/
winesearch.pl">

<p>Variety:

    <select name="variety">
        <option value="All" selected>All varieties</option>
        <option value="red">Any Red</option>
        <option value="rcab">Cabernet Sauvignon</option>
        <option value="rmer">Merlot</option>
        <option value="rsir">Petite Sirah</option>
        <option value="rsan">Sangiovese</option>
        <option value="rzin">Zinfandel</option>
        <option value="wht">Any White</option>
        <option value="wchr">Chardonnay</option>
        <option value="wblc">Pinot Blanc</option>
        <option value="wgrs">Pinot Gris</option>
        <option value="wsau">Sauvignon Blanc</option>
    </select>

</p>
<p>Cost

    <select name="cost">
        <option value="0-150" selected>All prices</option>
        <option value="0-10">less than $10</option>
        <option value="10-15">$10 - $15</option>
        <option value="15-10">$15 - $20</option>
        <option value="20-40">$20 - $40</option>
        <option value="40-75">$40 - $75</option>
        <option value="75-150">$75 and up</option>
    </select>

</p>
<p>Region

    <select name="region" multiple size="3">
        <option value="any">Any region</option>
        <option value="cal">California</option>
        <option value="calnap">- Napa/Sonoma</option>
        <option value="calcen">- Central valley</option>
        <option value="calsud">- South coast</option>
        <option value="uswest">Other US West Coast</option>
        <option value="otherus">Other US</option>
        <option value="can">Canada</option>
        <option value="chile">Chile</option>

        <option value="frc">France</option>
        <option value="frcbor">- Bordeaux</option>
        <option value="frebur">- Burgundy</option>
        <option value="frcrho">- Rhone</option>
        <option value="frcsub1">- (Lots of subcategories</option>
        <option value="frcsub2">- go here)</option>
        <option value="ity">Italy</option>
        <option value="itysub1">- Italy subcategories</option>
        <option value="por">Portugal</option>
        <option value="spn">Spain</option>
    </select>

</p>
<p>Food
```

```
        <select name="food">
            <option value="any" selected>Any food</option>
            <option value="non">No food</option>
            <option value="hor">Hors d'oeuvres</option>
            <option value="sal">Salad</option>
            <option value="bef">Beef</option>
            <option value="por">Pork</option>
            <option value="chk">Chicken</option>
            <option value="duk">Duck</option>
            <option value="tof">Tofu</option>

            <option value="bur">Burgers R' Us</option>
            <option value="sol">Sole</option>
            <option value="dab">Sand Dabs</option>
            <option value="sal">Salmon</option>
            <option value="tun">Tuna Steak</option>
            <option value="etc">and so on</option>
            <option value="des">Dessert</option>
        </select>

    </p>

    <input type="button" name="submit" value="Submit"
        onClick="document.forms[0].submitNoResponse(
        'Check back for search results next time you sync!',
        true, true);back();" >

    </form>

    <p>&nbsp;</p>
    <p>&nbsp;</p>
    </body>
    </html>

    <tr>
        <td colspan="10">
        <img src="/images/transparent.gif" width="1" height="40">
        </td>

    </tr>

    <tr>
        <td colspan="2"><img src="/images/transparent.gif"
        width="1" height="1" alt="" border="0"></td>

        <td colspan="8" background="/images/dotline.gif">
        <img src="/images/transparent.gif" width="1" height="1"
        alt="" border="0"></td>

    </tr>

    <tr>

        <td colspan="2"><img src="/images/transparent.gif"
            width="1" height="1" alt="" border="0"></td>

        <td colspan="8"><font class="footer">

        <br>Your use of the web site is subject to and
        constitutes acceptance of the
        <a href="/company/legal/terms.html">Terms of Use</a> and
        <a href="/company/legal/privacy.html">Privacy<br>Statement</a>
        » <a href="/company/legal/">Legal</a> - Modified 12/11/02
        » Copyright 1998-2003 iAnywhere Solutions, Inc. All rights reserved.
        <br><br><br></font>
        </td>

    </tr>

    </table>

    <SCRIPT LANGUAGE="JavaScript1.2" SRC="/dhtml/HM_Loader.js"
        TYPE='text/javascript'></SCRIPT>

    </body>
    </html>
```

winesearch.pl

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);
use CGI::Cookie;


#         winesearch.pl
#
# Sample code used to generate a wine search. Note that it
# does not actually conduct any sort of search. It just
# enters the search into the user's preferences.
#
# The search itself is conducted by the realsearch.pl
# script.

srand;

%cookies = fetch CGI::Cookie;

if ($cookies{'findagoodwineID'} ) {
    $myID = $cookies{'findagoodwineID'}->value;
} else {
    $myID = getrandomID();
}

# We create the cookie, with a name, ID, and expires time
# of 10 years.

$winecookie = new CGI::Cookie(-name=>'findagoodwineID',
    -value=>$myID,-expires=>'+10y');

# We get the query string and append it on to the
# _real_ cgi-bin script

$searchURL = '/cgi-bin/findagoodwine/realsearch.pl?'.
    $ENV{'QUERY_STRING'};

# and we get the time and use that as our search
# description

$searchDesc = "Search submitted ". localtime();
$bigstring = $searchURL. '~::~'. $searchDesc;

dbmopen(%history, "winehistory", 0644) || die "cannot open DBM $!";

@lastsearch = split(/~:~/, $history{$myID});
```

```
$lastsearch[2] = $lastsearch[1];
$lastsearch[1] = $lastsearch[0];
$lastsearch[0] = $bigstring;

$finallist = join("~:~", @lastsearch);

$history{$myID} = $finallist;

dbmclose(%history);

print "Content-type: text/html\n";
print "Set-Cookie: $winecookie\n";
print "Cache-Control: private\n";

print <<"End_of_Page";

<HTML>
    <HEAD>
    <META name="HandheldFriendly" content="True">
    <TITLE>Search has been entered</TITLE>
    </head>
    <BODY>

    Thanks for your search. Go back and reload the
    <a href="/cgi-bin/findagoodwine/frontpage.pl">Front page</a>
    to see the results of your new search.

    <p>
    More debug info:
    <p>

    searchURL = $searchURL <br>
    searchDesc = $searchDesc <br>
    lastsearch2 = $lastsearch[2] <br>
    lastsearch1 = $lastsearch[1] <br>
    lastsearch0 = $lastsearch[0] <br>
    <p>
    finallist = $finallist
    </body>

</html>

End_of_Page

sub getrandomID {
    return (time(). sprintf("%07d",(int(rand(5000000)))));
}


#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

realsearch.pl

```
#!D:\Perl\bin\perl.exe
use CGI qw(:standard);


#           realsearch.pl
#
# Sample code for conducting a search for the perfect
# wine. Since I don't actually have any access to real
# wines, I just created some random responses.

print header(-'Cache-Control'=>'max-age=604800');
srand;

# pseudo-real response

$winename{'all'} = "Some wine";
$winename{'red'} = "Some red wine";
$winename{'rcab'} = "Cabernet";
$winename{'rmer'} = "Merlot";
$winename{'rsir'} = "Petite Sirah";
$winename{'rsan'} = "Sangiovese";

$winename{'rzin'} = "Zinfandel";
$winename{'wht'} = "Some white wine";
$winename{'wchr'} = "Chardonnay";
$winename{'wblc'} = "Pinot Blanc";
$winename{'wgrs'} = "Pinot Gris";
$winename{'wsau'} = "Sauvignon Blanc";

@vineyard = ("Fru-fru gardens", "Grapes R Us",
    "Da Wine Haus", McWine", "Squashed Grape",
    "Wine Hut", "Wine-in-a-box", "Fancy pants",
    "Grapealicious!");

@descriptions = qw(Dry Fruity Sweet Full-bodied
    Light-bodied Tart Flavorful Good Excellent
    tasty yummy icky);

@odors = qw(courant blackberry peach pear cherry
    orange sulfur apple lemon raspberry);

print <<End_of_Beginning;

<HTML>
    <HEAD>
    <META name="HandheldFriendly" content="true">
    <TITLE>Search Results</TITLE>
    </HEAD>
    <BODY>
    Results from your search:


End_of_Beginning

# Create three wines...

for ($i = 0; $i < 3; $i++) {

    print '<table width="100%" border="0"
        cellspacing="1" cellpadding="0">'. "\n";
    print '<tr>      <td colspan="2">'. "\n";
    print '<div align="center"><b>';
    # Print the vineyard, type of wine, and year
    print $vineyard[int(rand(@vineyard))];
    print ' '. $winename{param('variety')}. ', '.
        (int(rand(50)) + 1950). '</b></div>'. "\n";
    print '</td></tr>'. "\n";

    # Print a random price based on the "cost" parameter
    $_ = param('cost');
    ($low, $hi) = /(\d+)-(\d+)/;
    print '<tr><td width="20%">Cost</td><td width="80%"> $';
    print (int(rand($hi - $low)) + $low). '</td></tr>'. "\n";

    # Generate some random adjectives to the wine's odor
    # and overall description.

    print '<tr><td width="20%">Descriptions</td><td width="80%"> ';
    print $descriptions[int(rand(@descriptions))]. ', ';
    print $descriptions[int(rand(@descriptions))]. '</td></tr>'. "\n";
    print '<tr><td width="20%">Odors</td><td width="80%">';
    print $odors[int(rand(@odors))]. ', '.
        $odors[int(rand(@odors))]. '</td></tr>'. "\n";
    print '</tr> </table>'. "\n";

    print '</BODY></HTML>'. "\n\n";

}


#
# Copyright (c) 2007, iAnywhere Solutions, Inc.,
# all rights reserved.
#
# IANYWHERE MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT
# THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED
# WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
# PURPOSE, OR NON-INFRINGEMENT. IANYWHERE SHALL NOT BE
# LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT
# OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR
# ITS DERIVATIVES.
```

Using a .mal file to subscribe users to a channel

If your channel is open to anyone in your organization and subscribing to it is optional, you can create a *.mal* file that users can access from your organization's intranet to subscribe to the channel.

Below is a sample *.mal* file:

```
<?xml version="1.0"?>

<MALServer

    hostname="shaft"
    port="80"
    username="tip"
    password="t"
    passwordIsHashed="FALSE"

    passwordIsEncoded="FALSE"
    nonce=""
    disabled="FALSE"
    friendlyName="Shaft-bld128"
    userUrl="http://SHAFT/"

    description="The server is running."
    confirmationcaption="Connected to Shaft-bld128"
    serverUri="/sync"
    sendDeviceInfo="TRUE"
    hashPassword="TRUE"

    requestPassword="FALSE"
    ConnectSecurely="FALSE"
    AllowSecureConnection="TRUE"
    ConnectSecureOnly="FALSE"
    DeviceSecureOverride="FALSE"

    confirmation="Your desktop software is now ready to sync to
        Shaft-bld128 [username: tip].">

</MALServer>
```

Master-Detail page example

Note
The source files for this example are available online. You can download them from http://www.ianywhere.com/developer/code_samples/master_detail_mbus.html.

The master-detail page sample illustrates the common situation where one a master page provides summary level information on customers. A master page could deal with accounts, products, or any other category of information. From this master page, the user can drill down and see the detail level of information for any selected entry on the master page.

This sample code illustrates use of:

- On-device data to dynamically generate pages
- List Viewer to display a table of information and sort the table on a column by clicking the column header
- M-Business database API to filter the information displayed
- `PODSPrefs` object to maintain the application state when switching between pages

The following source files are provided with this sample code:

- "Master page" on page 419 — presents summary information on customers, from which user can select a particular customer for which to see detailed information; user can also filter and sort the customer summary information.
- "Detail page" on page 432 — presents detailed information on the customer selected on the master page.
- "Pages to download supporting binaries" on page 436 — shows how the `AG_DEVICEOS` macro is used to download the correct binaries for either Microsoft OS or Palm OS devices.
- "Customer data XML file" on page 436 — shows the structure of the XML data that was downloaded into the on-device datastore in order to dynamically generate the master and detail pages.
- "Customer XML data schema file" on page 437 — shows the schema file associated with the XML data.

Master page

The master page uses the List Viewer to dynamically generate the table of customer summary information from on-device data. It also uses the List Viewer to allow the user to sort the customer information on any one of three columns by tapping on a column header. To make it easier for the user to locate the customer of interest, it also supports filtering the customer list, When the user taps within a row for a particular customer, information necessary to maintain the application state is passed to the detail page, which displays more detailed information on the selected customer.

CustomersList.html

```
<html>
<head>
<TITLE>Customer Listview</TITLE>
<META name="HandheldFriendly" content="True">

<!--
Demonstration of data driven master-detail page
development with M-Business Client.

Originally authored by John Karabaic and extended by
Yadong Liu of iAnywhere Solutions.

Copyright, 2007. iAnywhere Solutions.
-->

<script>

var dbmgr;
var accountDb;
var accountSortOrderArray = new Array(3);
var accountCols = new Array(3);

function initPage()
{

    var list;
    var dbname = "Customers";

    // Open database

    if (undefined == dbmgr) {
        dbmgr = CreateObject('avantgo.db');
    }

    if (null == dbmgr) {
        alert('Error: unable to create avantgo.db');
        return false;
    }

    accountDb = dbmgr.open(dbname, "r");

    if (null == accountDb) {
        alert("failed to open db");
        return;
    }

    // Save dbname as session info in avantgo preferences object.

    avantgo.preferences.setStringValueForKey("ListPage_dbname", dbname)

    // Set attributes, and create event handler for accountTable

    list = document.getElementById("accountTable");
    list.setDbSet(accountDb);
    list.onclick = onClickHandler;

    // Initialize vars

    accountCols[0] = "ContactName";
    accountCols[1] = "CompanyName";
    accountCols[2] = "PostalCode";
```

```
        accountSortOrderArray[0] = 0;
        accountSortOrderArray[1] = 0;
        accountSortOrderArray[2] = 0;

        // Set attributes

        list.setColumnCount(3);

        list.setColumnTitle(0, "Customer");
        list.setColumnWidth(0, 60);
        list.setColumnDSName(0, "ContactName");
        list.setColumnHeaderImage(0, 1);

        list.setColumnTitle(1, "Company");
        list.setColumnWidth(1, 60);
        list.setColumnDSName(1, "CompanyName");
        list.setColumnHeaderImage(1, 1);

        list.setColumnTitle(2, "Postal Code");
        list.setColumnWidth(2, 40);
        list.setColumnDSName(2, "PostalCode");

    }

    function exitPage()
    {
        if(accountDb) accountDb.close();
    }

    function onClickHandler(e)
    {

        var list;
        var component;
        var i;
        var index;
        var data;
        var colId;

        list = document.getElementById("accountTable");

        // Figure out where the click happened
        // Passing (x, y) in document coordinate system

        component = list.getComponent(e.clientX, e.clientY)

        // Possible component values:
        //  TARGET_NONE = 0,
        //  TARGET_PREV_PAGE_BTN = 1,
        //  TARGET_NEXT_PAGE_BTN = 2,
        //  TARGET_HEADER = 3,
        //  TARGET_ITEM = 4,
        //  TARGET_SCROLL_UP_BTN = 5,
        //  TARGET_SCROLL_DOWN_BTN = 6,
        //  TARGET_SCROLLBAR_BUBBLE = 7,

        if (4 == component) {
            index = list.getSelectionDbSetIndex();

            //check if the user hits just the pixel between
            //rows which gives me a hit results, but no selected row.

            if(index < 1 || index > accountDb.nrows) {
                return true;
            }
            accountDb.moveTo(index);
            data = accountDb.getStringField("ContactName");

            // You can optionally display the selected ContactName
            // alert("row selected - " + data);

            // Don't call default handler after this since we are done.

            gotoDetailPage("Customers", accountDb.CustomerID,
                "CustomerDetail.html");
            return false;

        }
        else if (3 == component) {
            var hasSelection = false;
            var selectionDbIndex = -1;
            var selectionId = null;

            // save current selection

            index = list.getSelectionIndex();
            if (-1 != index) {
                hasSelection = true;
                selectionDbIndex = list.getSelectionDbSetIndex();
                accountDb.moveTo(selectionDbIndex);
                selectionId = accountDb.getStringField("ContactName");
            }

        // Figure out which column the user clicked on

        colId = list.getColumnId(e.clientX, e.clientY);

        //  SORT_NONE = 0,
        //  SORT_ASCENDING = 1,
        //  SORT_DESCENDING = 2,

        for (i = 0; i < 3; i++) {

            if (colId == i) {
                if (1 == accountSortOrderArray[i]) {
                    accountSortOrderArray[i] = 2;
                }

                else {
                    accountSortOrderArray[i] = 1;
                }
            }

            else {
                accountSortOrderArray[i] = 0;
            }
        }

        // Set sort order for the db set

        accountDb.setSort(accountCols[colId], accountSortOrderArray[colId] == 1);

        // Set sort order for the column

        list.setColumnSortOrder(colId, accountSortOrderArray[colId] == 1);

        // Refresh mimelist to use the modified dbset

        list.refreshDbSet();
```

```
    // Set mimelist selection

    if (hasSelection) {
        var filter = "ContactName = " + selectionId;
        var search = accountDb.createSearch(filter);
        accountDb.moveFirst();

        if (accountDb.find(search)) {
            var index = accountDb.index;
            list.setSelectionDbSetIndex(index);
        }
    }

    // Force a redraw of the mimelist

    list.refreshScreen();

    // Don't call default handler after this since we are done.

    return false;

    }
    else {
        // Call default handler since we haven't handled this event
        return true;
    }

}

function setFilter()
{

    var list;
    var filterInput;
    var filter;
    var search;

    list = document.getElementById("accountTable");
    filterInput = document.getElementById("filterInput");

    if ((null == filterInput.value) ||
            (0 == filterInput.value.length)) {
        search = accountDb.setFilter(null);
    }
    else {

        // Set filter for the db set
        filter = "ContactName ~ " + filterInput.value;
        accountDb.moveTo(0);
        search = accountDb.createSearch(filter);
        accountDb.setFilter(search);
    }

    // Refresh mimelist to use the modified dbset
    list.refreshDbSet();

    // Force a redraw of the mimelist
    list.refreshScreen();

}

function gotoDetailPage(dbname, id, pageurl)
{
    avantgo.preferences.setStringValueForKey(
        "detailPage_"+dbname+"Id", id);
    window.location = pageurl;
}

</script>
</head>

<body onload="initPage()" onUnload="exitPage()">

<form>
<table width="220" cellspacing="0" cellpadding="0" border="0">

<tr>
    <td align="right">
        <input style="background-color:#ffffcc;font-family:verdana;
            font-size:9px;padding-top:2px;padding-bottom:1px;"
            type="text" name="filter" id="filterInput" value=""
        <input type="button" value="Search" onclick="setFilter()">
    </td>
</tr>

</table>
</form>

<form>

<object id="accountTable" type="application/avantgo-mime-list"
    width="220" height="240" style="vertical-align:baseline;
    padding-left:3px;border-color:black;border-width:1;
    background-color:white;color:black;font-size:9px;
    font-family:verdana;">

    <param name="numRows" value="10"></param>
    <param name="rowMargin" value="2"></param>
    <param name="colMargin" value="4"></param>
    <param name="headerBgColor" value="#66cc99"></param>
    <param name="headerBdColor" value="#000000"></param>

    <param name="cellBgColor" value="#f9f9f9"></param>
    <param name="cellBdColor" value="#cecece"></param>
    <param name="cellSelectBgColor" value="#cecece"></param>
    <param name="cellSelectBdColor" value="#f9f9f9"></param>

    <param name="scrollbarBdColor" value="#000000"></param>
    <param name="scrollbarBgColor" value="#666666"></param>
    <param name="scrollbarButtonColor" value="#66cc99"></param>
    <param name="scrollbarArrowColor" value="#000000"></param>
    <param name="scrollbarBubbleColor" value="#66cc99"></param>

</object>

</form>
    <a href="CustomerDetail.html"></a>
    <a href="PODS/AG_DEVICEOS/pods.htm"></a>

</body>
</html>
```

Master page for Microsoft Smartphone

The M-Business List Viewer API contains Javascript functions `pageUp()`, `pageDown()`, `scrollUp()`, and `scrollDown()`. By setting an `onKeyPress` handler for the list, you can allow users on Microsoft Smartphone devices to navigate list data using keys on the device. For sample code illustrating this functionality, see the listing that follows — “CustomersList_MSP.html” on page 425. This is basically the same sample code in “CustomersList.html” on page 420, modified to use device keys to navigate the list data.

Comments in the listing explain which keys are being set to which functions. For a description of keys and functions, see “Using a Microsoft Smartphone device with the List Viewer” on page 226.

CustomersList_MSP.html

```
<html>
<head>
<TITLE>Customer Listview</TITLE>
<META name="HandheldFriendly" CONTENT="true">
<META HTTP-EQUIV="content-type" CONTENT="text/html; charset=UTF-8">
<!-- Demonstration of data driven master-detail page development with M-
Business client.
Originally authored by John Karabaic and extended by Yadong Liu of iAnywhere
Solutions.

Extended by Doug Lowder to add key press handler and enable Smartphone usage,
2006

Copyright, 2004, 2005, 2006. iAnywhere Solutions. -->


<script>

var dbmgr;
var accountDb;
var accountSortOrderArray = new Array(3);
var accountCols = new Array(3);


function initPage()
{
  var list;
  var dbname = "Customers";

  // Initialize vars

  accountCols[0] = "ContactName";
  accountCols[1] = "CompanyName";
  accountCols[2] = "PostalCode";

  accountSortOrderArray[0] = 0;
  accountSortOrderArray[1] = 0;
  accountSortOrderArray[2] = 0;


// Set attributes, and create event handler for accountTable

  list = document.getElementById("accountTable");

  list.onclick = onClickHandler;
  // Here we register the key press handler that allows Smartphone to work
  list.onkeypress = onKeyPressHandler;


// Set attributes
```

```
    list.setColumnCount(3);

    list.setColumnTitle(0, "Customer");
    list.setColumnWidth(0, 60);
    list.setColumnDSName(0, "ContactName");
    list.setColumnHeaderImage(0, 1);


    list.setColumnTitle(1, "Company");
    list.setColumnWidth(1, 60);
    list.setColumnDSName(1, "CompanyName");
    list.setColumnHeaderImage(1, 1);

    list.setColumnTitle(2, "Postal Code");
    list.setColumnWidth(2, 40);
    list.setColumnDSName(2, "PostalCode");


    // Open database
    if (undefined == dbmgr) {
        dbmgr = CreateObject('avantgo.db');
    }
    if (null == dbmgr) {
        alert('Error: unable to create avantgo.db');
        return false;
    }


    accountDb = dbmgr.open(dbname, "r");
    if (null == accountDb) {
        alert("failed to open db");
        return false;
    }


    // Save dbname as session info in AvantGo preferences object.
    avantgo.preferences.setStringValueForKey("ListPage_dbname", dbname)

    list.setDbSet(accountDb);

  }

  function exitPage()
  {
    if(accountDb) accountDb.close();
  }

  // This key press handler allows the user to use arrow keys to navigate
  into
  // and out of the mimelist.
  // Up and down arrow keys and the numbers 2 and 8 on the keypad scroll up
  and
  // down the list.
  // Numbers 3 and 9 on the keypad page up and down the mimelist.
  // Pressing 5, or the center button of the Dpad, selects the highlighted row
  // and opens the CustomerDetail.html page.
  // Pressing 0 sorts the mimelist; pressing 0 repeated cycles through
  ascending
  // and descending sorts on different columns.


  function onKeyPressHandler(e)
  {
      var list;
      var component;
      var i;
      var index;
      var data;
      var colId;
      var j;


      // alert("e.keyVal = " + e.keyVal);

      list = document.getElementById("accountTable");

      if (50 == e.keyVal || 101 == e.keyVal || 69 == e.keyVal) { // scroll up
  when key pressed is 2, up arrow, or 'E'

          index = list.getSelectionDbSetIndex();
          if(index == 1) // we are at the top of the list, so return true and
  allow us to navigate out of the mimelist
              return true;

          list.scrollUp();
          // Dont call default handler after this since we are done.
          return false;
      }

      else if (56 == e.keyVal || 120 == e.keyVal || 88 == e.keyVal ) { // scroll
  down when key pressed is 8, down arrow, or 'X'

          index = list.getSelectionDbSetIndex();
          if(index == accountDb.nrows) // we are at the bottom of the list, so
  return true and allow us to navigate out of the mimelist
              return true;

          list.scrollDown();
          // Dont call default handler after this since we are done.
          return false;
      }

      else if (51 == e.keyVal || 114 == e.keyVal || 82 == e.keyVal) { // page
  up when '3' or 'R' is pressed
          list.pageUp();
          // Dont call default handler after this since we are done.
          return false;
      }
      else if (57 == e.keyVal || 99 == e.keyVal || 67 == e.keyVal) { // page
  down when '9' or 'C' is pressed
          list.pageDown();
          // Dont call default handler after this since we are done.
          return false;
      }

      else if (53 == e.keyVal || 13 == e.keyVal || 68 == e.keyVal) {  // hit '5'
  or return or 'D', we selected a row, go to Customer detail page
          index = list.getSelectionDbSetIndex();
          if(index < 1 || index > accountDb.nrows) {
              index = 1;
          }

          accountDb.moveTo(index);
          data = accountDb.getStringField("ContactName");

          // You can optionally display the selected ContactName
          // alert("row selected - " + data);
          gotoDetailPage("Customers", accountDb.CustomerID,
  "CustomerDetail.html");
          // Dont call default handler after this since we are done.
          return false;
      }

      else if (48 == e.keyVal) { // hit '0', rotate through possible column
  sorts

          var hasSelection = false;
          var selectionDbIndex = -1;
          var selectionId = null;


          // save current selection
          index = list.getSelectionIndex();
          if (-1 != index) {
              hasSelection = true;
              selectionDbIndex = list.getSelectionDbSetIndex();
              accountDb.moveTo(selectionDbIndex);
              selectionId = accountDb.getStringField("ContactName");
          }


          //  SORT_NONE = 0,
          //  SORT_ASCENDING = 1,
          //  SORT_DESCENDING = 2,

          for (i=0; i<3; i++) {
              if (1 == accountSortOrderArray[i]) {
                  accountSortOrderArray[i] = 2;
                  colId = i;
                  i=3;
              }

              else if (2 == accountSortOrderArray[i]) {
                  j = i + 1;
                  if(j==3) j=0;
                  accountSortOrderArray[i] = 0;
                  accountSortOrderArray[j] = 1;
                  colId = j;
                  i=3;
              }
          }


          if ( (accountSortOrderArray[0] == 0)
                  && (accountSortOrderArray[1] == 0)
                  && (accountSortOrderArray[2] == 0) ) {
              accountSortOrderArray[0] = 1;
          }

          // alert("sortorderarray = " + accountSortOrderArray[0] + " " +
          //     accountSortOrderArray[1] + " " + accountSortOrderArray[2]  );

          // Set sort order for the db set
         accountDb.setSort(accountCols[colId], accountSortOrderArray[colId] ==
  1);

          // Set sort order for the column
          list.setColumnSortOrder(colId, accountSortOrderArray[colId] == 1);

          // Refresh mimelist to use the modified dbset
          list.refreshDbSet();

          // Set mimelist selection


          if (hasSelection) {
              var filter = "ContactName = " + selectionId;
              var search = accountDb.createSearch(filter);
              accountDb.moveFirst();
              if (accountDb.find(search)) {
                  var index = accountDb.index;
                      list.setSelectionDbSetIndex(index);
             }
          }


          // Force a redraw of the mimelist
          list.refreshScreen();

          // Don't call default handler after this since we are done.
          return false;
      }
      else {
          // Call default handler since we haven't handled this event
          return true;
      }
  }


  function onClickHandler(e)
  {
      var list;
      var component;
      var i;
      var index;
      var data;
      var colId;

      list = document.getElementById("accountTable");


      // Figure out where the click happened
      // Passing (x, y) in document coordinate system

      component = list.getComponent(e.clientX, e.clientY)


      // Possible component values:
      //  TARGET_NONE = 0,
      //  TARGET_PREV_PAGE_BTN = 1,
      //  TARGET_NEXT_PAGE_BTN = 2,
      //  TARGET_HEADER = 3,
      //  TARGET_ITEM = 4,
      //  TARGET_SCROLL_UP_BTN = 5,
      //  TARGET_SCROLL_DOWN_BTN = 6,
      //  TARGET_SCROLLBAR_BUBBLE = 7,


      if (4 == component) {
          index = list.getSelectionDbSetIndex();
          //check if the user hits just the pixel between
          //rows which gives me a hit results, but no selected row.
          if(index < 1 || index > accountDb.nrows) {
              return true;
          }

          accountDb.moveTo(index);
          data = accountDb.getStringField("ContactName");

          // You can optionally display the selected ContactName
          // alert("row selected - " + data);

          // Don't call default handler after this since we are done.
          gotoDetailPage("Customers", accountDb.CustomerID,
  "CustomerDetail.html");
          return false;
      }

      else if (3 == component) {

          var hasSelection = false;
          var selectionDbIndex = -1;
          var selectionId = null;

          // save current selection
          index = list.getSelectionIndex();
          if (-1 != index) {
              hasSelection = true;
              selectionDbIndex = list.getSelectionDbSetIndex();
              accountDb.moveTo(selectionDbIndex);
              selectionId = accountDb.getStringField("ContactName");
          }

          // Figure out which column the user clicked on
          colId = list.getColumnId(e.clientX, e.clientY);

          //  SORT_NONE = 0,
          //  SORT_ASCENDING = 1,
          //  SORT_DESCENDING = 2,

          for (i = 0; i < 3; i++) {
              if (colId == i) {
                  if (1 == accountSortOrderArray[i]) {
                      accountSortOrderArray[i] = 2;
                  }
                  else {
                      accountSortOrderArray[i] = 1;
                  }
              }

              else {
                  accountSortOrderArray[i] = 0;
              }
          }

          // Set sort order for the db set
         accountDb.setSort(accountCols[colId], accountSortOrderArray[colId] ==
  1);

          // Set sort order for the column
          list.setColumnSortOrder(colId, accountSortOrderArray[colId] == 1);

          // Refresh mimelist to use the modified dbset
          list.refreshDbSet();

          // Set mimelist selection


          if (hasSelection) {
              var filter = "ContactName = " + selectionId;
              var search = accountDb.createSearch(filter);
              accountDb.moveFirst();
              if (accountDb.find(search)) {
                  var index = accountDb.index;
                      list.setSelectionDbSetIndex(index);
             }
          }

          // Force a redraw of the mimelist
          list.refreshScreen();

          // Don't call default handler after this since we are done.
          return false;
      }
      else {
          // Call default handler since we haven't handled this event
          return true;
      }
  }


  function setFilter()
  {
      var list;
      var filterInput;
      var filter;
      var search;

      list = document.getElementById("accountTable");
      filterInput = document.getElementById("filterInput");

      if ((null == filterInput.value) ||
          (0 == filterInput.value.length)) {
          search = accountDb.setFilter(null);
      }

      else {
          // Set filter for the db set
          filter = "ContactName ~ " + filterInput.value;
          accountDb.moveTo(0);
          search = accountDb.createSearch(filter);
          accountDb.setFilter(search);
      }

      // Refresh mimelist to use the modified dbset
      list.refreshDbSet();
      // Force a redraw of the mimelist
      list.refreshScreen();
  }

  function gotoDetailPage(dbname, id, pageurl)
  {
      avantgo.preferences.setStringValueForKey("detailPage_"+dbname+"Id", id);
      window.location = pageurl;
  }

  </script>
  </head>

  <body onload="initPage()" onUnload="exitPage()" >

  <form>
  <table width="220" cellspacing="0" cellpadding="0" border="0">
```

```
<tr>
    <td align="right">
                <input style="background-color:#ffffcc;font-family:verdana;
                    font-size:9px;padding-top:2px;padding-bottom:1px;"
type="text"
                    name="filter" id="filterInput" value="" size="8">
                <input type="button" value="Search" onclick="setFilter()">
        </td>
</tr>
</table>
</form>

<form>

<object id="accountTable" type="application/avantgo-mime-list"
    width="220" height="180" style="vertical-align:baseline;padding-left:3px;
    border-color:black;border-width:1;background-color:white;color:black;
    font-size:9px;font-family:verdana;" onKeyPress="onKeyPressHandler()">

    <param name="numRows" value="6"></param>
    <param name="rowMargin" value="2"></param>
    <param name="colMargin" value="4"></param>
    <param name="headerBgColor" value="#66cc99"></param>
    <param name="headerBdColor" value="#000000"></param>
    <param name="cellBgColor" value="#f9f9f9"></param>
    <param name="cellBdColor" value="#cecece"></param>

    <param name="cellSelectBgColor" value="#cecece"></param>
    <param name="cellSelectBdColor" value="#f9f9f9"></param>
    <param name="scrollbarBdColor" value="#000000"></param>
    <param name="scrollbarBgColor" value="#666666"></param>
    <param name="scrollbarButtonColor" value="#66cc99"></param>
    <param name="scrollbarArrowColor" value="#000000"></param>
    <param name="scrollbarBubbleColor" value="#66cc99"></param>

</object>

</form>

<a href="CustomerDetail.html"></a>
<!-- Link to XMLDB PODS -->
<!-- For PalmOS devices, use href="PODS/AG_DEVICEOS/dbpod_qual.prc" instead
-->
<a href="PODS/AG_DEVICEOS/dbpod_quad.dll"></a>
<a href="PODS/AG_DEVICEOS/mimelist.dll"></a>
<a href="PODS/AG_DEVICEOS/dbpod_quad.prc"></a>
<a href="PODS/AG_DEVICEOS/mimelist.prc"></a>

</body>
</html>
```

Detail page

CustomerDetail.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >

<html>
<head>
```

```
<TITLE>Customer detail page</TITLE>
<META name="HandheldFriendly" content="True">

<!--
Demonstration of data driven master-detail page development
with M-Business Client. Originally authored by John Karabaic
and extended by Yadong Liu of iAnywhere Solutions.

Copyright, 2004. iAnywhere Solutions.
-->

<script>

var dbMgr = CreateObject("avantgo.db");
var dbset = null;

function onloadHandler() {

    // Retrieve dbname and CustomerID.

    var dbname =
        avantgo.preferences.getStringValueForKey(
        "ListPage_dbname");
    var id =
        avantgo.preferences.getStringValueForKey(
        "detailPage_"+dbname+"Id");

    dbset = dbMgr.open(dbname, "r");

    if(dbset == null) {
        alert("Could not open the '" + dbname + "' database.");
        return;
    }

    // Locate the customer record that matches CustomerID.

    var search = dbset.createSearch("CustomerID ==" + id);
    dbset.setFilter( search );
    dbset.moveFirst();

    // Bind customer data to detail page template.

    document.getElementById("customer_name")
     .firstChild.data = dbset.ContactName;
    document.getElementById("customer_title")
     .firstChild.data = dbset.ContactTitle;
    document.getElementById("customer_company")
     .firstChild.data = dbset.CompanyName;
    document.getElementById("customer_address")
     .firstChild.data = dbset.AddressLine1;
    document.getElementById("customer_city")
     .firstChild.data = dbset.City;

    document.getElementById("customer_zip")
     .firstChild.data = dbset.PostalCode;
    document.getElementById("customer_country")
     .firstChild.data = dbset.Country;
    document.getElementById("customer_phone")
     .firstChild.data = dbset.Phone;
    document.getElementById("customer_fax")
     .firstChild.data = dbset.Fax;

}

function onunloadHandler() {
    if( dbset ) dbset.close();
}

</script>
</head>

<body onload="onloadHandler()"
    onunload="onunloadHandler()"
    bgcolor="#ffffff"
    style="margin-left:0px;margin-top:0px;" >

<p><b>Sample Listviewer - Detail Page </b></p>

<table width="224">

<tr style="background-color:#66cc99;">

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Customer Name:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top">
    <span id="customer_name">_</span></td>

<tr>

<tr>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Customer Title:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
       padding-top:3px;" valign="top"><span id="customer_title">_</span></td>

<tr>

<tr style="background-color:#66cc99;">

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Company Name:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_company">_</span></
td>

<tr>

<tr>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Address:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_address">_</span></
td>

<tr>
```

```
<tr style="background-color:#66cc99;">

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>City:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_city">_</span></td>

<tr>

<tr>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Postal Code:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_zip">_</span></td>

<tr>

<tr style="background-color:#66cc99;">

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Country:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_country">_</span></
td>

<tr>

<tr>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Phone:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
       padding-top:3px;" valign="top"><span id="customer_phone">_</span></td>

<tr>

<tr style="background-color:#66cc99;">

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><b>Fax:</b> </td>

    <td style="padding-right:1px;padding-left:1px;
        font-size:9px;padding-bottom:1px;color:#666666;
        padding-top:3px;" valign="top"><span id="customer_fax">_</span></td>

<tr>

</table>

</body>
</html>
```

Pages to download supporting binaries

Binaries are platform specific; different binaries must be downloaded to run on Microsoft OS vs. Palm OS devices. The "Master page" on page 419 uses the `DEVICE_OS` URL macro to cause the correct *pods.htm* page to be downloaded — the line below is near the end of the file:

```
<a href="PODS/AG_DEVICEOS/pods.htm"></a>
```

When M-Business Server replaces `AG_DEVICE` with the value supplied by either a Microsoft OS or Palm OS device, the *pods.htm* page in turn causes the download of the correct binaries that have been compiled to run on that device OS — *.dll* files are downloaded to Microsoft OS devices and *.prc* files are downloaded to Palm OS devices. For more information on URL macros, see "Using server URL macros to customize content" on page 166.

Caution
The link from the master page, *CustomersList.html* to a *pods.htm* is at the same Link Depth required to download *CustomerDetail.html*. The link from either *pods.htm* file to the supporting binaries adds one to the Link Depth. If *CustomerDetail.html* is already at the limit of your Link Depth, the binaries will fail to download. For more information on how Link Depth is calculated, see the Link Depth topic in "Description of channel settings" on page 29. To see how to get around your Link Depth setting, if you do not want to increase it, see "Downloading a page beyond channel's Link Depth" on page 56.

WINCE_OS/pods.htm

```
<html>
    <a href="mimelist.dll></a>
    <a href="dbpod_quad.dll></a>
</html>
```

PALM_OS/pods.htm

```
<html>
    <a href="mimelist.prc></a>
    <a href="dbpod_quad.prc></a>
</html>
```

Customer data XML file

The XML data file is downloaded to the mobile device via the M-Business XML conduit. On the device, the XML data is transferred into the on-device datastore, from which it can be accessed by the M-Business XML POD.

Your back-end database on the server must produce XML data in this element-based format, or in an equivalent attribute-based format, in order for it to be transferred correctly into the on-device datastore. For more information on generating the XML data file, see "Generating the XML file" on page 278.

Only the data for the first and last customers is shown below. The complete file contains data on 91 customers. The complete file is available from http://www.ianywhere.com/developer/code_samples/master_detail_mbus.html.

Customers.xml

```
<?xml version="1.0" encoding="iso-8859-1"?>

<root>

    <Customers>
    <CustomerID>ALFKI</CustomerID>
    <CompanyName>Alfreds Futterkiste</CompanyName>
    <ContactName>Maria Anders</ContactName>
    <ContactTitle>Sales Representative</ContactTitle>

    <AddressLine1>Obere Str. 57</AddressLine1>
    <AddressLine2></AddressLine2>
    <City>Berlin</City>
    <Region></Region>
    <PostalCode>12209</PostalCode>

    <Country>Germany</Country>
    <Phone>030-0074321</Phone>
    <Fax>030-0076545</Fax>
    </Customers>

 ...

    <Customers>
    <CustomerID>WOLZA</CustomerID>
    <CompanyName>Wolski  Zajazd</CompanyName>
    <ContactName>Zbyszek Piestrzeniewicz</ContactName>
    <ContactTitle>Owner</ContactTitle>

    <AddressLine1>ul. Filtrowa 68</AddressLine1>
    <AddressLine2></AddressLine2>
    <City>Warszawa</City>
    <Region></Region>
    <PostalCode>01-012</PostalCode>

    <Country>Poland</Country>
    <Phone>(26) 642-7012</Phone>
    <Fax>(26) 642-7012</Fax>
    </Customers>

</root>
```

Customer XML data schema file

The XML data schema file is required to successfully download the XML data to the mobile device via the M-Business XML conduit.

Your back-end database on the server must produce an XML data schema file in this format. For more information on generating the XML data file, see “Generating the XML file” on page 278.

Customers.xsd

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

    <xsd:schema

        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:dt="urn:schemas-microsoft-com:datatypes"
        xmlns:msch="urn:schemas-microsoft-com:mapping-schema">

    <xsd:element name="Customers"

        msch:relation="Customers"
        type="Customers_type"/>

    <xsd:complexType name="Customers_type">

        <xsd:element name="CustomerID" type="xsd:string"/>
        <xsd:element name="CompanyName" type="xsd:string"/>
        <xsd:element name="ContactName" type="xsd:string"/>
        <xsd:element name="ContactTitle" type="xsd:string"/>
        <xsd:element name="AddressLine1" type="xsd:string"/>

        <xsd:element name="AddressLine2" type="xsd:string"/>
        <xsd:element name="City" type="xsd:string"/>
        <xsd:element name="Region" type="xsd:string"/>
        <xsd:element name="PostalCode" type="xsd:string"/>
        <xsd:element name="Country" type="xsd:string"/>

        <xsd:element name="Phone" type="xsd:string"/>
        <xsd:element name="Fax" type="xsd:string"/>
        <xsd:key name="PrimaryKey">
        <xsd:selector xpath="."/>
            <xsd:field xpath="@CustomerID"/>
        </xsd:key>

    </xsd:complexType>

</xsd:schema>
```

Index