WebSphere Micro Environment for palmOne Devices
MIDP 2.0 Porting Guide (GM Final)
March 2004
www.palmone.com/java

This document provides detailed information on the WebSphere Micro Environment implementation for palmOne devices. In particular, it reviews the components of the MIDP and CLDC standards that could vary between a standard MIDP handset-type device and a palmOne device.

Standards and Devices

The new WebSphere Micro Environment J2ME runtime implements the MIDP 2.0 and CLDC 1.1 standards:
http://jcp.org/en/jsr/detail?id=118
http://jcp.org/en/jsr/detail?id=139

The CLDC 1.1 Floating Point Support is IEEE-754 compliant:
http://grouper.ieee.org/groups/754/

It is supported on Palm OS 5.2 and beyond devices, including Tungsten C, Tungsten T2, Tungsten T3, Tungsten E, Zire 71, and the Treo 600.

The original Tungsten T is not officially supported due to it having a small dynamic heap and Palm OS 5.0, though it can be used for development.

The Zire 21 is not supported.

Tungsten W (Palm OS 4.1) is supported by the previously released WebSphere Micro Environment 5.6 68k/PACE runtime, which supports MIDP 1.0 and CLDC 1.0:
http://jcp.org/en/jsr/detail?id=30
http://jcp.org/en/jsr/detail?id=3

Memory Availability and Limitations

The WebSphere Micro Environment runtime has access to all available dynamic and storage heap on the palmOne device. There is no seperate memory for Java applications.

Device	Dynamic Heap	Storage Heap
Treo 600	3mb	32mb
Tungsten T3	4mb	51mb
Tungsten C	4mb	51mb
Tungsten E	1mb	32mb
Tungsten T2	1mb	32mb
Zire 71	1mb	16mb
Tungsten W	1mb	16mb

The WebSphere Micro Environment calls to totalMemory() and freeMemory() represent the total memory allocated by the virtual machine for the current MIDlet. WebSphere Micro Environment will dynamically allocate more memory as needed up to the maximum available heap on the device. The returned value from totalMemory() will change throughout the lifecycle of the application.

Also, the total amount of available dynamic heap is shared between the MIDlet Suite application, the WebSphere Micro Environment runtime, and any other system service currently running in memory. For instance, calls to the network stack will load the TCP/IP stack as well as underlying radio stack (Wifi, Bluetooth, CDMA, etc.) into memory.

Dynamic heap on devices such as the Zire 71 and Tungsten E is very limited, and any MIDlet Application should be well tested on these device to assure it works within the more limited memory context.

The only limit for total MIDlet suite size is the available storage heap on a device.

There are two virtual machines implemented for WebSphere Micro Environment for Palm OS 5; the 68k/PACE implementation, and the ARM implementation. The two implementations have different limits even on the same device, as the memory management is quite different. The two VMs also support different configuration and profile versions. The 68k/PACE VM supports CLDC 1.0/MIDP 1.0 and the ARM VM support CLDC 1.1/MIDP 2.0.

Arguably the most critical memory component that determines what applications can be run on a Palm OS device is the amount of dynamic heap. The dynamic heap is the space where the virtual machine will load classes, create objects, allocate images, etc. The 68k implementation makes use of the storage heap. This allows for all of the available storage memory on the

device to be used to run Java programs. This is done at the expense of interoperability with other Palm OS applications. For this reason applications such as a background MP3 player do not work well while the 68k VM is running.

The ARM implementation makes use of only the dynamic heap, the size of the dynamic heap varies from device to device. On ARM devices fairly large dynamic heaps are available, whereas traditionally on the 68k devices much smaller dynamic heaps made it impractical for java. Since the normal dynamic heap is being used, there are no interoperability issues other than the expected contention for limited dynamic heap space – some java developers may be surprised that less memory is available to their java programs even though the storage heap size is larger on the ARM device.

As a rule of thumb, the larger and more complex the application, the more dynamic heap will be required. There are several factors to be taken into account:

* number and size of classes to be loaded
* number of instances to be created
* number and size of images to be created and drawn onto the screen

As an example, a very small application can very easily run out of memory. Creating many images and drawing them is enough to run out of dynamic heap on some devices.

Observations using the ARM virtual machine

It has been observed that the ARM VM requires a minimum of 440k available dynamic heap to start up and run a very simple, form-based MIDlet application. (This was determined using Memory Hog to limit the available dynamic heap on a device) However, 440k is not sufficient to run any meaningful application.

The amount of dynamic heap available to the virtual machine will vary depending on the state of other applications or services. For example, the dynamic heap available on the Treo 600 varies depending on the radio module activity. The amount of available dynamic heap decreases once when the radio is turned on and further when data is being transmitted. Running an application in the background (e.g. an MP3 player) will also decrease the amount of dynamic heap available to the VM.

If a device lacks sufficient dynamic heap the virtual machine will not be able run the requested application. As described above, what is "sufficient" varies greatly from application to application. If less than 440k of dynamic heap is available the VM will not be able to startup. The VM will either immediately exit back to the Palm OS launcher (reporting OutOfMemoryError to standard err) or display a dialog, "java.lang.Error: Unable to create offscreen window". Similarly, if the application is running and the VM cannot allocate the required memory the same failure will occur.

Available Dynamic Heap	**Device Tested**	**Observations**
680kb	Tungsten T2	Very few MIDlets will execute to completion. Those that do are small form-based MIDlets and simple graphics MIDlets (i.e. games).
930kb	Zire 71	Many common MIDlets will execute to completion. MIDlets creating many images or loading large images still fail.
1960kb	Tungsten E	All common MIDlets which were tested execute to completion. However, this did not include any MIDlets using network connections.
2900kb	Treo 600	All common MIDlets which were tested execute to completion. This did include testing MIDlets using network connections.

RMS Memory Limits

Each record in an RMS store can be a maximum of about 64kb.

The total size of an RMS store and the total number of RMS stores is limited by the total device Storage Heap.

Media and Resources

PNG images supported from 8 to 24 bit depth with transparency. Actual displayed colors will be limited by the device color depth capability.

WAV audio supported from 8khz, mono up to 44.1khz, stereo.

If an application is provisioned via the Palm Desktop HotSync mechanism, maximum individual JAR resources of any format are 64kb. If an application is provisioned via web download or an SD card, there is no limit, other than the device memory. However, an application with resources larger than 64kb will not be able to be backed up using the Palm Desktop HotSync mechanism.

Resources loaded through a GCF Connection, like HTTP, can be any size within the limits of the available memory.

There is currently no support for loading local resources from an SD memory card.

Display Resolution and Color

Device	Resolution(s)	Color Depth
Treo 600	160x160	12-bit
Tungsten T3	320x320, 320x480, 480x320	16-bit
Tungsten C	320x320, 160x160	16-bit
Tungsten E	320x320, 160x160	16-bit
Tungsten T2	320x320, 160x160	16-bit
Zire 71	320x320, 160x160	16-bit
Tungsten W	160x160	16-bit

Tungsten T3 supports dynamic resizing and rotating of MIDP 2.0 midlets through use of its “Dynamic Input Area” expanding and collapsing and rotation slip control. Change in screen height and width can be detected by a MIDP 2.0 Canvas by overriding the method “sizeChanged (int width, int height)”.

Devices which support 320x320 High Res mode can choose to run MIDlets in either that mode, or in low res 160x160 mode. This choice is made with the JarToPrc converter tool or as a user preference on the device.

The entire screen resolution is available if there are no on-screen Commands being displayed. If there are Commands showing for the Screen, they will take up approximately 16 (low res) or 32 (high res) pixels of spaces on the bottom of the screen.

Icons

Icons can be set for MIDlet suites converted to PRCs, using the JarToPrc utility. Icons should be in BMP, PNG, GIF, or JPG format.

For high-resolution or “double density” displays, such as on Tungsten and Zire devices, the icon sizes are: Large Icon View=44x44, Small List View=18x30

For low-resolution or “single density” displays, such as on Treo 600, the icon sizes are: Large Icon View=22x22, Small List View=9x15

Fonts

Devices which include recent version of Versamail (Tungsten C), include a special high resolution Font library. If this library is present, the Java runtime will be able to take advantage of it, and provide a richer mapping between the MIDP Font types and underlying native rendering. More information on this mapping will be provided in the future.

On devices without Versamail, Java fonts are mapped to the limited Palm OS fonts, which are unable to fully support the full combination of faces, styles, and sizes. Monospace and Proportional are the same font. Small and Medium are the same size. There is no italic support.

Command Placement

Commands are represented either as buttons placed in a row at the bottom of the screen, or as items in the Palm OS native menus (activated by the Menu silkscreen button or through a key on the keyboard). If onscreen button Commands overflow the bottom area of the screen, they will be placed in a Menu. Priority of Commands is respected for ordering purposes.

- Screen: at the bottom of the Screen
- Item: in the Palm OS Menus
- Ok: at the bottom of the Screen
- Back: in a seperate “Navigate” Palm OS Menu
- Help: in a seperate “Help” Palm OS Menu

Key and Game Events

Key Events can be generated through Graffiti text entry, the Onscreen Keyboard, or either an integrated or external hardware keyboard.

The Canvas direction game events and fire event are mapped to the palmOne device five-way navigation control.

The A, B, C, D events are mapped to those letter keys on devices with integrated keyboards.

Pointer Events

All pointer events are supported in Canvas subclasses.

The stylus can also be used with all other Screen types for selection and text entry (via Palm OS Graffiti or on-screen keyboard)

General Connection Framework

The following standard set of connection types are supported:

- HTTP = “http://”
- HTTPS = “https://”
- Secure Socket = “ssl://”
- Socket = “socket://”
- Datagram = “datagram://”
- Comm = “comm://”

The maximum number of simultaneous TCP/IP-based communication channels is limited by the Palm OS stack limitation. Currently, four simultaneous open connections should be possible.

Secure connections, such as HTTPS and SSL, utilize the certificate database provided in the Palm Web Browser certdb.pdb file. This is included standard with the Tungsten C devices. Other devices, such as the Tungsten T3 and Treo 600, utilize a different certificate format database. In these cases, the runtime utilizes its own certificate database, albeit a more limited one, provided by IBM. In either case, the RSA SSL library included in Palm OS 5.2 and beyond performs the SSL functions.

Available Comm Ports can be looked up using the standard MIDP 2.0 system property values. Support for Serial connections over IrDA and Bluetooth is included.

AMS and Over-The-Air Support

Supported for downloading of JAD and JAR files by a device's web browser is included. There is also a built-in "MIDlet HQ" application, which is the Java Application Manager (JAM). This allows for direction URL provisioning, and MIDlet permission managements.

The AMS system supports provisioning of JAD and JAR files through the Palm OS Exchange Manager, so theoretically any application that places these types of data into that OS subsystem can interact with the JAM. For instance, if a JAR or JAD file is emailed and received as an attachment by Versamail, it should be able to hand this off to the JAM via Exchange Manager. Also, if a JAD or JAR file is beamed or bluetoothed via OBEX to a device, it should be handled, as well.

Once a MIDlet is provisioned, it will appear on the Palm OS launcher as a standard Palm application. The MIDlet itself is stored as a Palm OS Launchable Database, which means that the JAM is invoked when the icon is selected on the launcher, and then the JAM loads the JVM and MIDlet bytecode for execution.

This is different than if the MIDlet is preconverted on the desktop to a PRC file. In that case, the JVM and MIDlets are invoked directly, without interaction from the JAM.

Please note: a HotSync operation does not by default place JAD and JAR files into the Palm OS Exchange Manager. MIDlets that will be provisioned through a HotSync operation must first be converted into a PRC file using the WebSphere Micro Environment Toolkit for Palm OS Developers (WebSphere Micro Environment Toolkit).

Platform Request

The MIDlet platformRequest() method is supported in this release. The set of protocols supported is determined by which applications on the device have registered with the Palm OS Exchange Manager. The Exchange Manager is a local device "bus" which allows for data to be passed between applications through URL protocols, mime-type and file extension registration.

Most Palm OS Web Browser will register to support the http:// and https:// protocols. Calling these will launch the browser, and exit the MIDlet. The Versamail email application will handle the mailto: protocol, opening a mail composition dialog as a sublaunch, and returning control to the MIDlet once

the message is sent or canceled. Other possible protocols that are supported include “sms” and “phoneto:”, though these are device specific.

A platformRequest sample application is provided on the www.palmone.com/java website “Lessons and Samples” section.

User and Developer Preferences

An end-user system preferences panel is included in the core WebSphere Micro Environment installation. This allows end-users to turn on and off double buffering, high resolution supports, and modify the amount of dynamic heap and stack size utilized by the Java environment.

A developer preference panel is included with the toolkit distribution. This panel is used to set command line options to the runtime, enable debugging options, and more.

Known Issues

- MIDI player and Tone Sequence support in the Media framework is not currently implemented
- Push Manager is not in the plan to be implemented
- Including classes larger than 64kb may generate hotsync issues
- MIDlets provisioned over the air are not currently backed up

Please see the WebSphere Micro Environment Toolkit Readme file for the most current listing of all known issues.