

# PTViewerME: Immersive Panoramas for PDA and Smartphone

Helmut Dersch  
FH Furtwangen  
der@fh-furtwangen.de

May 20, 2004

## Abstract

We have ported PTViewer, a Java-based spherical panorama viewer, to J2ME, Sun's microedition runtime for the Java language. A prepackaged version for PalmOS 5 is distributed together with a generic jar-version suitable for any CLDC-1.1/MIDP-2.0 compliant Java runtime.

## 1 Introduction

Handheld computers like personal digital assistants (PDAs) and smartphone devices are getting increasingly more powerful and now have reached levels making them suitable for simple image processing tasks. A useful yet demanding application is the interactive display of immersive panoramas. These panoramas are often used for virtual tours of properties or scenic locations. In connection with a handheld device they could be used for real-time guides and navigational aids.

Panorama viewers for handheld devices have been demonstrated [5]. However, this viewer displays just a set of prerendered images and does not provide true interactive navigation (pan,tilt,zoom) like the well known viewers [1],[4] for personal computers do. Our main objective is to demonstrate the feasibility of such a viewer and provide a platform for experimenting with new applications for interactive panoramas.

## 2 Program Structure and Techniques

PTViewerME (PTViewer *Microedition*) is based on the panorama viewer applet PTViewer [4] developed and published by the author of this article. PTViewer is Java based and requires the standard Java runtime J2SE [7]. It has also been ported to the limited API of the GNU Java-compiler [2] and runs as a native application on platforms supported by GJC. The functionality of this program and usage instructions are described elsewhere [4].

Less powerful Java runtimes have been developed for handheld devices. Sun Microsystems has defined a standard (J2ME [8]) for these reduced APIs, and compliant runtimes are distributed for many operating systems. The most recent version of this standard (CLDC1.1/MIDP2) provides enough high level functionality to approach the current task. We have based our work on the IBM-WebSphere environment [9], which contains compliant Java runtimes for PalmOS 5 and PocketPC.

The CLDC1.1/MIDP2 standard lacks many important features of J2SE. As a consequence, for PTViewerME we either had to incorporate additional code to provide these features, or reduce its functionality. The most important features which have been added to the standard PTViewer are:

- JPEG support is not included in J2ME, but crucial for the viewer. A pure-Java JPEG-decoder has been written, which basically is a port of the C++-decoder published by Bee-Chung Chen [6]. Only the panoramic image may use JPEG format, all other images of a virtual tour must use PNG.
- Many high level math functions (atan, exp, ln) are missing in J2ME and have been added using public domain sources.
- Large arrays commonly used to hold pixel data are not allowed due to heap size limitations. E.g., the Palm Tungsten E provides 32 MByte RAM, but only 2 MByte may be used as heap memory. J2ME offers the concept of RecordStores to access the additional memory in 60 kByte chunks. Large arrays are implemented by swapping these chunks.
- A couple of more standard Java classes have been coded separately.

Features which have been omitted from PTViewer due to the limited API are:

- GIF support. J2ME provides a PNG-decoder, and all images excluding the panorama itself should use this format. This includes wait- and frame-

images (see PTVIEWER documentation[4] about details). Watch out for file-size (see below).

- Mouse support is exchanged for pointer support. While the concept is similar, it is not the same.
- File system and network access is missing. This may be added later. Currently, all files used for virtual tours must be prepackaged with the viewer.
- No keyboard support. Only the left and right buttons are linked to left- and right pan.
- No PTVIEWER extensions (E.g. QTVR etc.).
- Many features of PTVIEWER have not been tested yet.

An important limitation which only refers to PalmOS devices is the file size limit of 64 kByte. No individual file in the JAR-package may exceed this size *including the panoramic images*. Providing a  $800 \times 400$  pixel image in JPEG format, compression level 60 – 70%, usually works and provides enough detail for the small screen size.

## 3 Installation and Usage

### 3.1 Palm Devices

Most Arm-processor equipped Palm-PDAs should be suitable. We have tested the Palm 5 emulator successfully, and I am personally running the software on a Tungsten E.

To install the sample tour *Marburg* from the PTVIEWERME-distribution [3] follow these steps:

- Download the latest IBM Java Runtime [10]. To my knowledge all other Java runtimes for PalmOS do not comply with CLDC-1.1/MIDP-2.0 specs.
- Install the Java Runtime on your Palm as described in the respective documentation.
- In the general settings dialog on the Palm, select IBM Java VM and set stack size and memory both to their maximum value.



Figure 1: PTViewerME running in Palm Emulator: Displaying progress bar during JPEG-decoding (*left*) and displaying rendered view (*right*). A background image covers unused portions of the screen. The actual screen image is clearer than it appears in this PDF-document.

- Install the file `Marburg.prc` from the PTViewerME-distribution [3] using HotSync.

### 3.2 Other Mobile Devices

PTViewerME has been tested to run on the Sun CLDC-1.1/MIDP-2.0 emulator. Please check to see whether a compliant Java runtime is available for your particular device and refer to the documentation of this runtime about how to install midlets. One or both of the files `PTViewer.jar` and `PTViewer.jad` may be required. I would be glad to hear from successful installs on other systems and include a compatibility list with later releases.

Figure 3.2 shows two screenimages taken from the PalmOS simulator. Navigation is similar to the standard version of PTViewer. The buttons in the viewer window are programmable by the author of the virtual tour, see the HTML-file below and the PTViewer documentation. In this case buttons for loading new panoramas, starting autopans, zooming and making hotspots visible are provided. Panning in the viewer window is possible by dragging the pointer. An additional

set of buttons is provided outside the viewer. Be patient when issuing commands: the viewer takes some time to respond, see below.

## 4 Performance

On a Tungsten E (32 MByte RAM, 120 MHz) using a  $200 \times 140$  pixel viewer window we get approx. 1 Frame/sec for nearest neighbor interpolation and 0.5 for bilinear interpolation. This is significantly worse than what is provided by a PC with similar resources. The emulator is approximately 3 times faster.

The reason for the poor performance is probably the inefficient memory management ( $\rightarrow$  PalmOS, see above) and graphics routines ( $\rightarrow$  Java, no access to screen memory) which act as bottleneck. More powerful PDAs may perform better and I would like to get results from other devices.

Nevertheless, providing immersive images on handheld devices is a new experience with many novel applications. The performance of the host device will inevitably reach the level of current PCs in the near future.

## 5 Authoring Virtual Tours

### 5.1 Creating the Midlet

The following tutorial explains how to package virtual tours and the PTVIEWERME program into an installable object. I suggest to work with the provided tour *Marburg* before proceeding with your own images. Locate the file `PTViewer.jar` which contains all necessary components. Create an empty folder `work` and move this file into `work` for the next steps.

To disassemble the tour and later assemble new ones, the J2SE Java development kit from Sun [7] is needed. It is a free download available for almost any platform (Linux, MacOS, Windows, ...). After installation of the Java tools, open a DOS/Shell-window in the folder containing the file `PTViewer.jar` and type `jar xf PTViewer.jar`. Now you can see and study all the files required for a virtual tour which consist of the usual HTML- and image-files as well as the Java-classes. To build your own tour, you would now exchange the HTML and image file while leaving the Java files at their places.

To package these files back into a JAR file proceed as follows:

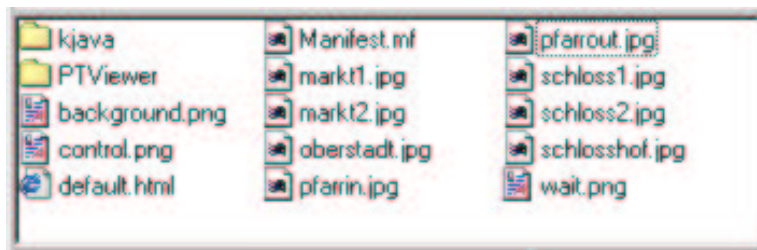


Figure 2: Components required for virtual tours: HTML-file and image files, Java-Classes and manifest.

- Move the file `Manifest.mf` out of the folder `Meta-inf` into `work`. Then remove the now empty folder `Meta-inf` and also the Jar-file `PTViewer.jar`. The folder `work` should then contain the items shown in figure 5.1.
- Type the command `jar cf PTViewer.jar -m Manifest.mf *`. This recreates the file `PTViewer.jar`. On some devices this midlet may be directly installable and executable.
- On PalmOS it is more convenient to wrap this file into a palm database and install via HotSync. The IBM WebSphere Micro Environment Toolkit is required for this task (only Windows). It can be obtained from Palm [9]. You have to register as a developer to get a free copy. After installation of this toolkit, drag the file `PTViewer.jar` onto the icon of the tool `jartoprc_w.exe` which is located in the `bin` directory of the Websphere installation. The dialog shown in figure 5.1 appears.

The application name as well as the signature should be changed to something unique and meaningful, then click `Generate PRC`.

## 5.2 Controlling the Midlet

An important goal of the current approach is to provide compatibility with other versions of `PTViewer`. Ideally, the same virtual tour could be displayed with minimal changes by `PTViewerME`.

The `PTViewer` documentation details how the applet (midlet in the case of `PTViewerME`) is controlled by the applet-tag in a HTML-file. This HTML file *must* be named `default.html`. These are the changes required to transform a VR-tour from standard `PTViewer` to `PTViewerME`:

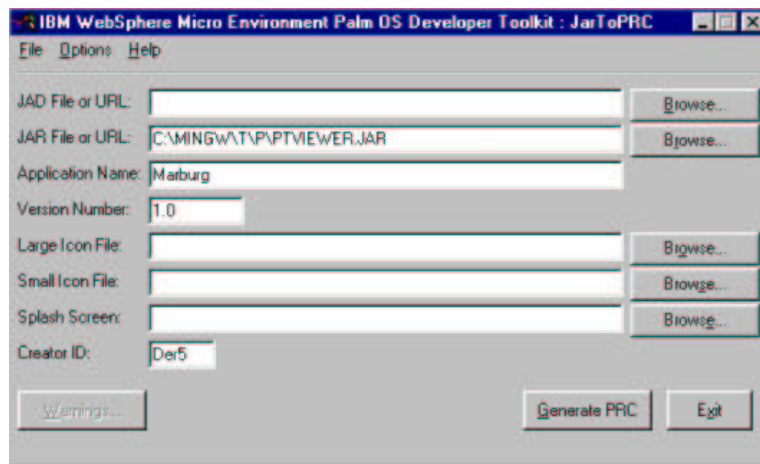


Figure 3: Wrapping the JAR-Midlet into a Palm database using the jartoprcw.exe tool.

- Reduce the size of the viewer window. 200x140 is the largest I can use on my Palm, although screen size is 320x320. This may vary depending on the particular PDA.
- Change the size of the image files: 64 kByte for PalmOS. This limit may not exist in other mobile devices.
- Change GIF to PNG. PNG works in standard PTViewer also, so there really is no need for GIF at all, except for animations.
- Remove features which do not work in PTViewerME: Extensions, warped hotspots, ...

One new tag has been added to PTViewerME: `background`, which can be used to specify a background image covering the unused parts of the screen, eg `<PARAM name=background value=background.png>`. See the tour *Marburg* and figure 3.2 for an example. The difference to the `frame` parameter is that `background` is painted just once at initialization time, and does not slow down drawing later. `Background` may therefore not protrude into the viewer. The `frame`, instead, is painted during each update, and may partially cover the viewer.

I suggest to view and study the provided tour *Marburg* and stepwise change this tour to what you want.

## 6 Building PTViewerME

To build PTViewerME from the included Java-sources, the following tools and programs are needed.

- The standard J2SE Java development kit from Sun [7].
- The microedition J2ME Java development kit from Sun [8]. This includes emulators for running the application. If you only want to create jar/jad files, this is all you need.
- For Palms, the MIDP 2.0 version of the WebSphere Micro Environment Toolkit is required. It can be obtained from Palm [9]. You have to register as a developer to get your free copy. You still need all the Sun tools above.

The sources are distributed under GNU-license, see the text of this license here: <http://www.fh-furtwangen.de/~dersch/Copying.html>.

## References

- [1] <http://www.quicktime.com>
- [2] <http://www.gnu.org>
- [3] <http://www.fh-furtwangen.de/~dersch/>
- [4] <http://www.fh-furtwangen.de/~dersch/>
- [5] <http://www.kinoma.com>
- [6] <http://www.cs.wisc.edu/~beechung/home/coding/index.html>
- [7] <http://java.sun.com/j2se/1.4.2/download.html>
- [8] <http://java.sun.com/j2me/index.jsp>
- [9] <http://www.palmone.com/java>
- [10] <http://www.palmone.com/support/jvm>