Chapter 3

Getting Started

This chapter explains the components included with MapXtreme Java Edition, how to install them, and how to initialize the components.

Overview

MapXtreme Java Edition ships with two CDs, MapXtreme and HAHTsite, this Developer's Guide and the MapXtreme Object Model poster. The MapXtreme Java Edition CD contains the MapXtremeServlet, MapJ API, code libraries, sample applications, map data, and help files.

The HAHTsite CD contains the complete HAHTsite IDE package for developing and programming Web applications and the HAHTsite Application Server. Use of the HAHTsite software is optional. See page 29 for more information on HAHTsite.

If you are already developing Web applications in another development environment, such as JDeveloper or Visual Cafe, you will not have to install HAHTsite. HAHTsite is included as a convenience to you. If you do not currently have an IDE or an Application Server, you can use HAHTsite. MapXtreme Java Edition was built to be used by a multitude of IDEs.

System Requirements

MapXtreme Java Edition was created to allow development of mapping applications on any platform that supports a Java virtual machine. The following requirements are the minimum necessary to implement your mapping applications:

- Web server that supports servlets or web server with plug-in to support servlets
- A video card installed on the server
- A Java 2 Platform compatible virtual machine 1.2.2 or higher (no longer supporting Java 1.1)
- 30 megabytes hard drive space for MapXtreme Java (~40 MB for installation)
- 250 megabyte hard disk space for sample map data
- 64 megabytes of RAM available for MapXtreme.

Installation

MapXtreme Java can be installed on any system that supports a Java 2 VM. The general procedure is:

- Install MapXtreme Java with or without a 1.2.2 VM
- Install sample map data
- Install Tomcat and integrate it with MapXtreme Java and Apache Web Server (optional)
- Install HAHTsite 4.0 IDE and Application Server (optional)

Once you have completed the installation, be sure and go to the section "Setting Up Your Servlet Container" on page 33 for instructions on configuring MapXtreme Java with your servlet container.

Each installation is described below.

MapXtreme Java Installation

To install MapXtreme Java, open the install.htm in a Java compatible browser and follow the prompts (steps are below).You can choose to install a 1.2.2 Java VM with MapXtreme Java or choose the noVM option where MapXtreme will not install a VM on your system. If you choose the latter option you will be prompted for the location of your existing VM during the installation. Be sure to specify a Java 2 VM (1.2.2 or higher).

The installation installs Java class files, samples, help files and (optionally) a Servlet Wizard Addin for JDeveloper 3.0 for Windows. If you want the Addin installed, be sure you have already installed JDeveloper 3.0 for Windows on your system. (Note this is for Windows only.)

To install MapXtreme Java:

1. Run install.htm from the MapXtreme Java CD. Select the installation package for your platform. Choose to install with or without a VM. The installation is copied to a temporary location on your system.

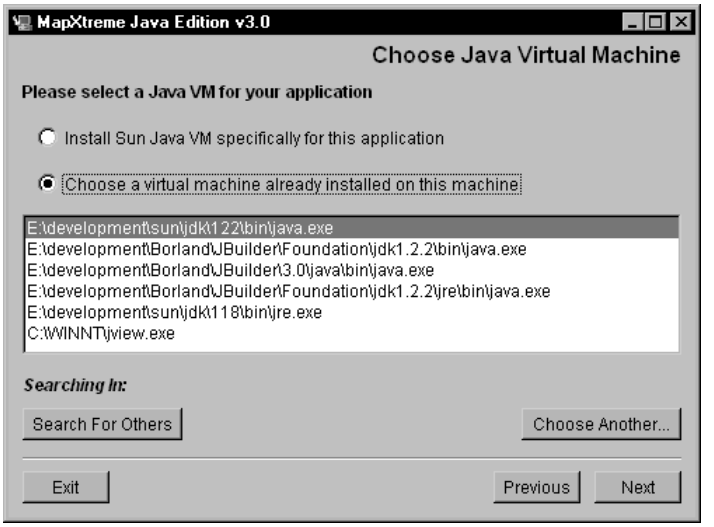
   Alternatively, you can bypass the browser. Go to the appropriate OS folder on the product CD under \InstData and run the install executable.

   Note: Choose the NoVM option if you already have a VM installed and do not want MapXtreme Java to install one for you. Choose the VM option if you wish to install an VM along with MapXtreme Java.

2. At the main MapXtreme screen, choose a language and click OK.
3. At the Introduction dialog, click Next.
4. At the Important Information, read the information and check the box indicating that you have read it and understand it. The Next button will not be activated until you check the box. Click Next.
5. Choose the location for MapXtreme Java by accepting the default location, clicking the Browse button and navigating to the location, or by manually typing in the location. Click Next.
6. Choose the location for the shortcuts. Click Next.
7. At the Choose Java Virtual Machine dialog, choose either:

   - Install Sun Java VM specifically for this install

   - Choose a Virtual Machine already installed on this machine



If you wish to choose your own VM, highlight one on the presented list or click Choose Another, which will display an Open dialog for you to locate the VM. Be sure to select a Java 2 compatible VM. Note: Clicking the Search button will initiate a lengthy search of your entire system.

8. At the Select Install Options dialog, choose from:

   - MapXtreme Core Components

   - Sample Applications

   - Documentation / Help

   - JDeveloper Addins (Win32 only)

9. Click Next. If you choose the JDeveloper Addin options, a dialog displays prompting you to specify the location of your installation of JDeveloper 3.x.

10. Click Next. The installation proceeds to completion.

Note: During the installation, files are copied to a temporary directory (IA_Installers). It is safe to delete them when the installation is complete.

To uninstall MapXtreme Java from Windows, click on the unistall shortcut on the Program menu under MapXtreme Java 3.0 or run the Uninstall MapXtremeJava30.exe, found in the UninstallerData directory after installation.

To uninstall MapXtreme Java from UNIX, run the uninstall script provided in the UninstalllerData data.

Sample Map Data Installation

A wide variety of sample map data is provided for your use in building and testing your application.You can choose to install all the data or one or more of five regions: North America, South America, Europe, Asia, and Australia.You will need at least the world.gst from one of the geographic regions to run the sample servlet and applet.

The total data set requires approximately 220 MB of disk storage. The installer requires about 350 MB to carry out the installation.

To install the map data:

1. On the MapXtreme Java CD, go to the \maps\InstData*yourplatform* directory and run the install executable under the VM folder (if you do not already have a VM on your system) or the noVm folder (the installer will run using the VM on your system).

2. At the initial screen, choose the appropriate language and click OK.

3. At the Introduction dialog, click Next to continue or Exit to leave the installer.

4. At the Choose Install Folder dialog, specify the location you wish for the data. This location can be any directory you choose. It does not have to be associated with the MapXtreme Java software. Specify the complete path for the location. If the folder does not exist it will be created. Click Next.
5. At the Choose Install Set dialog, click the Complete button and then click Install to install the entire sample data set.

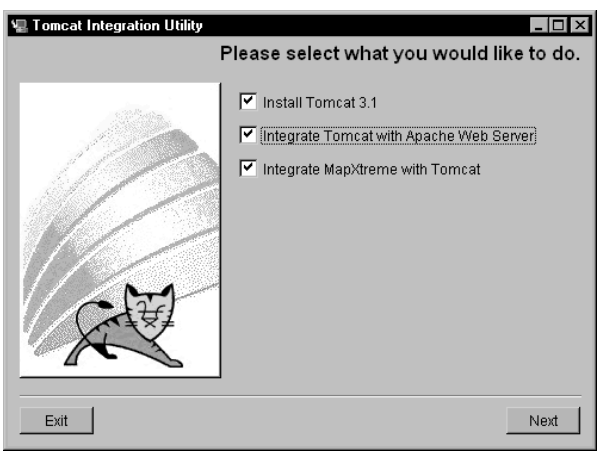
   To install a subset of the data, click the Customize button to display a list of geographic regions. Check or clear any or all of the checkboxes as desired.
6. Click Install and the process proceeds to completion. Click Exit, then Done to leave the install program.

To uninstall the sample data from Windows, run the Uninstall MapData.exe, found in the UninstallerData directory after installation.

To uninstall MapXtreme Java from UNIX, run the uninstall script provided in the UninstalllerData directory.

Tomcat Installation (Optional)

MapXtreme Java requires a web server / servlet container to run, such as Apache / Tomcat, JavaWebServer, JRun, etc. As a convenience to users who do not have a servlet container, we provide Tomcat as an optional installation. This installation will not only install Tomcat, but integrate it with an existing Apache Web Server (1.3.9 or higher) and MapXtreme Java 3.0. You can find a copy of Apache Web Server on the MapXtreme CD.



To install / integrate Tomcat:

1. Shut down Apache and / or Tomcat if they are already running.
2. Run install.htm from the \tomcat directory on the MapXtreme Java CD.

3. At theWelcome dialog, click Next.
4. At the Important Information dialog, review the information and check the box that you have read and understand the information. Click Next.
5. At the Please select what you would like to do dialog, choose any or all options:

   - Install Tomcat 3.1

   - Integrate Tomcat with Apache Web Server

   - Integrate MapXtreme with Tomcat
6. If you did not check Install Tomcat 3.1, the Find Current Tomcat Location dialog displays. Specify its location and click Next.
7. If you are installing Tomcat, the Choose an Install Location dialog displays. Choose a location where Tomcat will be installed.A Tomcat folder will automatically be created at this location. Click Next.
8. If you are configuring Tomcat with Apache Web Server, at the Find Current Apache Web Server Location dialog, specify the Apache location. Note: Apache Web Server 1.3.9 or higher is required. Click Next.
9. At the MapXtreme Servlet Sample Application Configuration dialog, specify a folder that contains your map data (world.gst). Enter a URL to find the sample servlet or accept the default `http://localhost:8080`. Click Next.
10. At the Installation Summary dialog, review the installation options. Click Next.
11. If you are configuring Tomcat with MapXtreme, provide the location where your map data exists and enter the URL to find MapXtremeServlet. Click Next. The installation proceeds to completion.

HAHTsite Installation (Optional)

The HAHTsite CD includes a full IDE and Application Server. It is included as a convenience if you are not already using an IDE or an Application Server. If you are already developing on an IDE and have an Application Server, you do not need to install HAHTsite.You will still have access to all of the features of MapXtreme Java Edition.

If you choose to install the HAHTsite components, you may install the IDE or the Application Server. The IDE is built to run on Windows systems. The Application Server will run on Windows NT, Solaris, and HPUX.

For installation instructions refer to the HAHTsite IDE and IP Installation Guide and the HAHTsite Application Server Installation Guide. See also page 47 for more information about the HAHTsite documentation set.

Installed Components

The following explains some of the key MapXtreme Java files that are installed under the \mapxtremejava\server directory on your system:

Jar Files

- mxtj30.jar - core MapXtreme Java class files.
- raster30.jar - class files for handling raster images.
- mistyles30.jar - contains pen and brush styles for map display
- dpext30.jar - class files necessary for JDBC connections.
- devsup.jar - class files for MapToolkit (Servlet utility library).
- mxloc30.jar - localized class files for U. S. and non-U.S. locations.
- xml4j_1_1_16.jar - XML parser

Although MapXtreme Java does not install JDBC drivers, be sure if you plan to access data in an RDBMS that you include the appropriate driver file along with the jar files in your classpath. For example, to access Oracle8*i*, you will need classes12.zip.

Fonts and Symbols

MapXtreme Java places these TrueType symbol font sets on your system. You will need to register these fonts with your operating system in order for them to be accessible in MapXtreme Java.

- MapInfo Cartographic
- MapInfo Transportation
- MapInfo Real Estate
- MapInfo Miscellaneous
- MapInfo Oil & Gas
- MapInfo Weather

- MapInfo Arrows
- MapInfo Shields
- MapInfo Symbols
- Map Symbols

Additionally, MapXtreme Java installs a folder of custom symbol GIF images. For a thumbnail view of each image, see Appendix E.

Nadcon

MapXtreme Java places a \nadcon directory of North American Datum Conversion (NADCON) files on your system. These files give more accurate Coordinate Transforms when converting between NAD83 and NAD27 datums.

Map Definition Manager

This is a GUI tool that allows you to create and load map definitions that describe the layers that make up your map.

To run Map Definition Manager on Windows, use the shortcut provided on the Start menu. The shortcut runs an executable file called Map Definition Manager.exe, found in the MapXtreme Java installation directory. The executable obtains its settings from the Map Definition Manager.lax file. The following are important configuration settings stored in the .lax file:

- lax.class.path – add any JDBC drivers you need to the list of jar files here. For example, to access Oracle8*i*, add classes12.zip to the classpath.
- lax.stderr.redirect – for debugging, put "`=console`" at the end to send the information to a console window.
- lax.stdout.redirect – put "`=console`" at the end for debugging, as above.
- lax.nl.current.vm – check the version of the VM, if it's not working.

For those developing on platforms other than Windows, there are no shortcuts. You must use the command line or script files to start Map Definition Manager. At the command line type:

```
java com.mapinfo.mapdefman.MapDefManager
```

For a complete description of Map Definition Manager, see Chapter 14.

Connections Manager

MapXtreme Java provides a tool to assist you in managing JDBC connections to RDBMS's. To run Connections Manager, from the Windows Start menu, use the shortcut provided. The executable gets its configuration settings from the Connections Manager.lax. It is important that you include any JDBC drivers in the classpath in order for Connections Manager to create the proper connections.

For developers on systems other than Windows, at the command line type:

```
java com.mapinfo.dp.util.ConnectionsManager
```

For more on Connections Manager and accessing remote databases, see Chapter 10.

Installation Troubleshooting

If you are unable to launch the installer using the instructions in the previous sections, you may directly access the installation program from the MapXtreme Java Edition CD. To directly access the installer go to the \install\ directory of the CD. Then choose the directory for your system. Run the install program in this directory and follow the directions in the MapXtreme Java Installation section.

Configuration Files

Depending on which install options you select, the installer may modify configuration files (*.ini, *.properties, *.conf) of other software on your system. For example, if you choose to install the optional JDeveloper Wizard, the installer will modify the following JDeveloper files: jdeveloper.ini, gallery.ini, and jdeveloper.properties. If you installTomcat and choose to configure Apache with Tomcat, the Installer will modify the Apache file httpd.conf.

Note that the installer makes backup copies of these configuration files before modifying them. For example, the installer will create httpd.conf.bak before modifying httpd.conf. Therefore, if you ever need to manually remove the MapXtreme changes to those configuration files, you can use the backup files as a reference.

After Installation

Once you have installed MapXtreme Java and the sample data and, optionally, Tomcat, the next step is to configure MapXtremeServlet with your servlet container Instructions are provided in the next section.

Setting Up Your Servlet Container

MapXtreme Java now uses a servlet model to service requests for maps. This requires that MapXtremeServlet run in a servlet container, such as Apache/Tomcat, JRun or JavaWebServer.

This section describes the steps necessary to set up MapXtremeServlet with a servlet container. The procedure is described first in general terms that apply to any servlet container, followed by specific steps for setting up MapXtremeServlet with Apache/Tomcat, Allaire JRun, and JavaWebServer 2.0.

Note: If you chose to integrate Tomcat and MapXtreme using the MapXtreme Tomcat installer, you do not need to manually configure Tomcat to run MapXtreme. You can proceed to page 36 to test the setup.

General Procedure

The exact process of setting up a servlet container varies, depending on which product (which container) you use. However, regardless of which servlet container you use, there are basic set-up tasks that must be performed:

1. Set up the servlet container so that it's aware of the MapXtreme jar files
2. Make the servlet container aware of the directory containing MapXtreme's .properties files.
3. Create a deployment descriptor for the class com.mapinfo.mapxtreme.MapXtremeServlet. This involves assigning a registered name by which the servlet will be called.
4. Deploy your own Java code that displays a map (e.g., applet, application, or servlet)
5. Test the setup to ensure MapXtremeServlet is running and you are using the correct URL for MapXtremeServlet.

Each step is discussed below:

1. Make Servlet Container Aware of MapXtreme Jar Files

Depending on the servlet container, you might simply copy the jar files into a designated directory, where they will be automatically loaded by the servlet container. In other cases, you may want to edit the container's classpath to reference the jar files in their original location.

You will need to make the servlet container aware of the following jar files:

- mxtj30.jar, raster30.jar, devsup30.jar, mistyles30.jar, xml4j_1_1_16.jar (MapXtreme files)
- dpext30.jar (if accessing an RDBMS)
- classes12.zip (for Oracle users)
- mxtjservletsamples30.jar (to use the pre-compiled samples)

2. Make Servlet Container Aware of Properties Files

MapXtreme properties files, such as rasterhandlerfactory.properties, reside in the MapXtreme\server directory after installation. Make the servlet container aware of this location by adding a reference to the classpath setting used by the servlet container.

3. Create a Deployment Descriptor for MapXtremeServlet

Although your servlet container might not require you to do so, it is a good idea to create a deployment descriptor for the class **com.mapinfo.mapxtreme.MapXtremeServlet**, which is provided in mxtj30.jar. Some servlet containers provide you with an Administrator dialog box, which assists you in creating deployment descriptors; other containers require you to create and/or edit an XML file to create a deployment descriptor.

By creating a deployment descriptor, you allow your programs to reference the server using a much shorter, registered name (e.g. your programs can specify the server as "mapxtreme" instead of "com.mapinfo.mapxtreme.MapXtremeServlet"). Also, once you have set up a deployment descriptor, you have greater control over the behavior of the servlet; for example, you can specify whether the servlet is loaded at startup (when the servlet container starts up). The table below identifies the init parameters for MapXtremeServlet that you can control.

Parameter	Value	Example
nadcon	The directory in which the North American Datum Conversion (NADCON) files are located. These are used to give more accurate Coordinate Transforms when converting between NAD83 and ND27 datums.	/opt/mapxtremejava/nadcon
jpegQuality	A value between 0 and 100 that specifies the quality of JPEG images exported by MapXtremeServlet. The default value is 75. Higher values produce better quality images and larger file sizes.	85

4. Deploy Your Java Code

Before your users can view maps generated by MapXtremeServlet, you must deploy your own Java code (e.g. an applet, application, or servlet that you have written to display a map). MapXtreme provides you with various JavaBeans and sample code to simplify this process.

Note: The MapXtremeServlet does not have any user interface, so users do not interact with MapXtremeServlet directly. Instead, a user might load a web page containing an applet, and the applet would communicate with MapXtremeServlet.

Instead of an applet, you might write and deploy a custom servlet that communicates with MapXtremeServlet. MapXtreme Java comes with an example of this type of servlet, called **HTMLEmbeddedMapServlet**. It comes as a pre-compiled sample contained in the mxtjservletsamples30.jar. Keep in mind that this servlet example is separate from MapXtremeServlet. MapXtremeServlet acts as a stateless server, with no user interface; HTMLEmbeddedMapServlet is a "sample client servlet" that you can customize, and it does have a user interface — it displays a map in a form in the user's browser.

5. Test the Setup

To test that you have set up your servlet container and MapXtreme correctly, start a browser and go to the URL you set during installation. For example:

```
http/localhost:8080/mapxtreme/servlet/htmlmapservlet
```

Typing this URL into your browser starts the sample client servlet, which in turn gets the map from the MapXtremeServlet (com.mapinfo.mapxtreme.MapXtremeServlet). The map is displayed in an HTML form along with several map items that allow you to change the map width, zoom in and out, pan, and control the display of map layers.

The following sections describe setup / test procedures for Tomcat, JRun, and JavaWebServer.

Setting up MapXtreme with Tomcat

The MapXtreme Tomcat installer performs the necessary setup to automatically integrate Tomcat and MapXtreme. If you haven't already installed Tomcat, follow the instructions on page 28.You do not need to manually integrate these products.

To test the setup using the sample HTMLEmbeddedMapServlet:

1. Start Apache if you have chosen to integrate Tomcat with Apache.
2. Start Tomcat by running startup.bat or startup.sh. Once Tomcat is initialized, MapXtremeServlet is ready to service requests.
3. Open a browser and go to the designated URL.

   ```
   http/localhost:8080/mapxtreme/servlet/htmlmapservlet
   ```

   A map with basic map navigation tools displays.

Note that the servlet name is case-sensitive. This is the registered name for HTMLEmbeddedMapServlet that was set during the integration process by the Tomcat installer. The registration information is contained in Tomcat's web.xml file, located in \{tomcat directory}\mapxtreme\Web-inf\web.xml.

Setting up MapXtreme with Allaire JRun

The following procedure tells you how to set up MapXtreme with Allaire JRun 2.3.3. This discussion assumes you have already installed JRun and, if desired, you have already integrated JRun with your web server (if any).

To set up MapXtreme with JRun:

1. If you have not already done so, install MapXtreme. Make a note of where the MapXtreme jar files (mxtj30.jar, etc.) are installed.
2. Edit the classpath that is used by JRun. You can edit this classpath in the JRun administrator, or you can edit the classpath setting in a text editor, by editing the java.classpath setting in the file jsm.properties. Note that JRun may install different jsm.properties files in different directories; you should edit jsm.properties file under jrun/jsm-default/properties. For example, if you install JRun on a Windows system, under JRun, you would edit the file:

   ```
   C:\jrun\jsm-default\properties\jsm.properties
   ```

   Expand the classpath to include references to: The mxtj30.jar file; the dpext30.jar file; the raster30.jar file; the xml4j_1_1_16.jar file; the devsup30.jar file; the mistyles30.jar, and mxtjservletsamples30.jar and the server directory where MapXtreme's .properties files exist (e.g. rasterhandlerfactory.properties). For example, if you installed MapXtreme on a Windows system, in C:\mxt, and MapXtreme's .properties files are in C:\mxt\server, then you would insert the following text at the start of the JRun classpath:

   ```
   C:/mxt/server/mxtj30.jar;C:/mxt/server/dpext30.jar;C:/
      mxt/server/raster30.jar;C:/mxt/server/devsup30.jar;C:/
      mxt/server;C:/mxt/server/xml4j_1_1_16.jar;C:/mxt/
      server/mistyles30.jar;C:/mxt/server/
      mxtjservletsamples30.jar
   ```

   NOTE: You should insert the MapXtreme classpath additions at the start of the classpath, to avoid conflicts between the XML parser used by MapXtreme (xml4j_1_1_16.jar) and a similar library that is included with JRun by default (xml4j.jar).

   Note also that if your miconnections.properties file contains database definitions (set up using the Connections Manager), then your classpath

should also include any archives needed for that database connection. For example, Oracle connections require classes12.zip.

3. Run the JRun administrator to set up a deployment descriptor (a set of initialization parameters, etc.) for HTMLEmbeddedMapServlet.

   If the JRun Service Manager tab appears, select "jsm-default" and click the Configure button.

   On the jsm-default panel, in the Services tab, select "jseweb" and click the Service Config button.

   On the jseweb panel, go to the Aliases tab. Click the Add button to add a deployment descriptor for the HTMLEmbeddedMapServlet. Under Name, specify a brief name such as mxtclient. Under ClassName, enter HTMLEmbeddedMapServlet -- and pay close attention to capitalization, as class names are case-sensitive. Then double-click the Init Arguments box to enter initialization parameters. At a minimum, enter the following initialization parameters:

Name	Value	Example
mapxtremeurl	The URL to your MapXtremeServlet. The exact host name or IP address and exact port number will depend on your configuration.	http://199.99.9.999/servlet/com.mapinfo.mapxtreme.MapXtremeServlet
filetoload	The path and filename of a geoset (.gst) or map definition (.mdf) to load.	C:\mxt\maps\asia.gst

4. To test the setup, start JRun and launch a browser.Type in an appropriate URL. For example, if you entered "mxtclient" as the name of your deployment descriptor, then your URL might be:

   `http://localhost/servlet/mxtclient`

   A map displays in an HTML form along with several map items that allow you to change the map width, zoom in and out, pan, and control the display of map layers.

Setting up MapXtreme with JavaWebServer 2.0

The following procedure tells you how to set up MapXtreme Java with Sun's JavaWebServer 2.0. This discussion assumes JavaWebServer is already installed.

JavaWebServer provides you with various ways of configuring the server. For example, given a jar that you want to use, you can copy the jar into JavaWebServer's designated /lib directory, or you can specify the jar through a classpath (with a -cp argument, in the batch file or script that you use to start the server). The following describes one way of configuring JavaWebServer, but not the only way.

1. If you have not already done so, install MapXtreme. Make a note of where the MapXtreme jar files (mxtj30.jar, etc.) are installed.
2. Copy all of the MapXtreme jar files into the /lib directory inside your JavaWebServer directory.
3. Edit the batch file or script that you use to start JavaWebServer, to add or modify the -cp argument so that it includes the directory where MapXtreme's .properties files exist (e.g. rasterhandlerfactory.properties). For example, if you installed MapXtreme on aWindows system, in C:\mxt, and MapXtreme's .properties files are in C:\mxt\server, then you could use the following command to start JavaWebServer:

   ```
   httpdnojre -javahome C:\jdk\122 -cp .;C:\mxt\server
   ```

   Note the use of the -javahome argument, which specifies the location of the VM that will be used to run JavaWebServer. This example is designed to use a Java 1.2 VM installed in C:\jdk\122.

   You do not need to add MapXtreme Java jar files, such as mxtj30.jar, to the classpath that you specify on the command line. This is because JavaWebServer automatically loads any .jar files located in the /lib folder. However, JavaWebServer does not automatically load .zip files. This is relevant to Oracle8*i* users, since classes12.zip is required for accessing Oracle8*i* layers. If you plan to use map layers stored in Oracle8*i*, then you must add classes12.zip to the classpath in your command line, like so:

   ```
   httpdnojre -javahome C:\jdk\122 -cp .;C:\mxt\server;
      C:\zips\classes12.zip
   ```

4. Start the JavaWebServer and its administrator (e.g. by browsing the URL http://localhost:9090/).
5. Click the Manage button, then click the Servlets icon (in the WebService dialog).
6. To add a new servlet definition for MapXtremeServlet, in the list at the left, click Add:

   For Servlet Name, type a brief name such as: mapxtreme

   For Servlet Class, type: com.mapinfo.mapxtreme.MapXtremeServlet

   Click the Add button to save the new deployment descriptor.

   Now that you have created a deployment descriptor, a client application or applet can use your MapXtremeServlet service, using the name that you entered:

   `http://yourhost:8080/servlet/mapxtreme`

   The exact host name or IP address and port number will depend on your configuration of JavaWebServer.
7. If you want to modify or customize the sample servlet, HTMLEmbeddedMapServlet.java, then compile the servlet and copy all of its class files into the JavaWebServer\servlets directory. However, if you simply want to run the sample servlet in its original form, you do not need to compile it or copy its class files — those class files are already provided for you, in mxtjsampleservlets.jar.
8. Return to the Web Service dialog, and click Add to add a new servlet definition — this time, for the sample HTMLEmbeddedMapServlet:

   For Servlet Name, type a brief name such as: mxtclient

   For Servlet Class, type: HTMLEmbeddedMapServlet

   Click the Add button to save the new servlet definition.
9. To complete the deployment descriptor, enter initialization parameter values to specify what MapXtreme URL and what map file the servlet should use. To define initialization parameters, go to the Properties tab, then click Add. At a minimum, enter the following initialization parameters:

Name	Value	Example
mapxtremeurl	The URL to your MapXtremeServlet (Will depend on your particular configuration).	http://199.99.9.999/servlet/mapxtreme
filetoload	The path and filename of a geoset (.gst) or map definition (.mdf) to load.	C:\mxt\maps\asia.gst

10. To test the setup, open a browser and type in the sample servlet's name that you entered:

    `http://yourhost:8080/servlet/mxtclient`

    The exact host name or IP address and port number will depend on your configuration of JavaWebServer.

    You should run the servlet using the brief name instead of the explicit classname — otherwise, the servlet container will not apply any init parameters that you specified.

    Note that the URL is case-sensitive.

    A successful test using HTMLEmbeddedMapServlet will display a map with basic map navigation tools.

Configuration Troubleshooting

Once you have installed MapXtreme and set it up in a servlet container such as Tomcat, you might encounter a configuration problem that prevents MapXtremeServlet from servicing mapping requests. For example, if you try to deploy the sample applet, SimpleMap, the applet might report a FileNotFoundException referring to the URL of your MapXtremeServlet.

If you experience such an error, there are two main possibilities:

- MapXtremeServlet might not be configured correctly.
- You might not be specifying the correct URL to access MapXtremeServlet.

To troubleshoot this type of problem, run a "debug" URL that will display a status page if MapXtremeServlet is running and the URL is correct.

In your browser's URL text field, type in the URL to your MapXtremeServlet, followed by "?debug=true". For example:

```
http://localhost:8080/mapxtreme/servlet/
   mapxtreme?debug=true
```

The status page will show basic diagnostic information, including the version of the Java VM and the version of MapXtreme in use.

If you do not see this status page, either the servlet is not running or the URL is not correct.

Verifying that MapXtremeServlet is Running

In addition to running the debug URL, above, you can check if MapXtremeServlet is running by watching the startup process. As your servlet container launches, watch for any indication of errors. For example, if your servlet container has a console window or log that you can view, view the console or log for any messages that refer to "MapXtremeServlet" or "mapxtreme".

If your servlet container provides an administrator tool, you should be able to check the status of MapXtremeServlet by running your administrator. If your servlet container does not provide an administrator tool, try viewing a console window or log to see whether MapXtremeServlet started.

The exact text that appears in your console window or log will depend in part on which servlet container you use. In general, if MapXtremeServlet is initialized

successfully, you should see an "init" message. For example, if MapXtreme has been correctly configured with Tomcat 3.1, the Tomcat console or log file shows the following text:

```
Context log path="/mapxtreme"  :mapxtreme: init
```

These messages indicate that the web.xml file for MapXtreme was processed, and the MapXtremeServlet was initialized correctly. Note that status message above says "mapxtreme: init" instead of "com.mapinfo.mapxtreme.MapXtremeServlet: init"; this is because MapXtreme's Tomcat installer assigns the MapXtremeServlet a registered name of "mapxtreme"; see the MapXtreme web.xml file.

Similarly, the JavaWebServer 2.0 console shows the following message when MapXtremeServlet is successfully initialized:

```
javawebserver: [Tue Apr 11 13:24:03 EDT 2000]
   com.mapinfo.mapxtreme.MapXtremeServlet: init
```

If MapXtremeServlet does not initialize when you start your servlet container, it does not necessarily mean that there is a problem. It might simply mean that the MapXtremeServlet configuration did not set the Load On Startup option to True (and hence, MapXtremeServlet has not yet been initialized because there have not yet been any requests for it). MapXtreme's Tomcat installer does set Load On Startup toTrue, so Tomcat users should see an "init" message, as shown above, when Tomcat is started.

If you are having difficulty verifying that MapXtreme is configured correctly, you should configure MapXtreme so that it is set to Load On Startup, and then restart your servlet container. In Tomcat, the Load On Startup option is specified in mapxtreme\WEB-INF\web.xml, using the <load-on-startup> tag. If your servlet container provides an administrator tool, you can probably set the Load On Startup option in the administrator.

If your MapXtremeServlet deployment descriptor is set to Load On Startup, and you still do not see a MapXtremeServlet "init" message in the servlet container's console or log, then there is probably an error in how MapXtreme is configured; in this case, review the steps that you used to configure MapXtreme, to make sure all configuration steps were performed correctly.

Verifying the Correct URL for MapXtremeServlet

Before you can take advantage of your MapXtremeServlet, you need to know its precise URL. For example, before you can run the sample applet, SimpleMap, you need to edit the .html file that loads the applet, and specify the URL to your MapXtremeServlet.

You should determine the precise URL to your MapXtremeServlet, and write down that URL for future reference:

MapXtremeServlet URL: http://_____

Note that URLs are case-sensitive.

The general form of a MapXtremeServlet URL is as follows:

```
http://hostname:portnumber/path/registeredname
```

hostname: This depends on the name or IP address of your server. If you are running everything on one computer (i.e. for testing or development), you might use "localhost" for your hostname.

portnumber: This depends on how you set up your servlet container. If you ran MapXtreme's Tomcat installer and accepted the defaults, this should be port 8080. If you have configured your servlet container to use port 80, you can omit the port number (and the colon) when you specify the MapXtreme URL.

path: This depends on your servlet container, and may also depend on what sub-directories you created. See examples below.

registeredname: This is a name that you can specify, such as "mapxtreme", which acts as a shortcut to the com.mapinfo.mapxtreme.MapXtremeServlet class. It is not a strict requirement that you use one; you can specify com.mapinfo.mapxtreme.MapXtremeServlet as the final part of your URL. However if you have set up a registered name for MapXtremeServlet, then you should use that registered name. (Any configuration options that you specified in your deployment descriptor, such as Load On Startup options, are associated with the registered name; if you use an URL that specifies the class name rather than the registered name, then those options will not take effect.)

Example: Tomcat

If you used MapXtreme'sTomcat installer to install Tomcat, and you accepted the defaults, then your MapXtremeServlet URL should be:

```
http://localhost:8080/mapxtreme/servlet/mapxtreme
```

In this example, the first "/mapxtreme" signifies the mapxtreme directory created inside the Tomcat directory, and the ending "/mapxtreme" refers to the registered name that the installer places into {tomcat directory}/mapxtreme/web-inf/web.xml, using the <servlet-name> tag.

Example: JavaWebServer

If you set up MapXtreme under Sun's JavaWebServer, the exact URL will depend partly on how you configured JavaWebServer. If you chose port 8080, and if you chose "mapxtreme" as the registered name when you created the MapXtremeServlet deployment descriptor, then your MapXtremeServlet URL might look like this:

```
http://localhost:8080/servlet/mapxtreme
```

If You See a Blank Map Frame

If the HTMLEmbeddedMapServlet runs but does not display a map, it probably means that you did not set the initialization parameters correctly (e.g., filetoload setting), or that you called the servlet using its class name instead of its registered name. You must call a servlet by its registered name for its initialization parameters to take effect.

MapXtreme Information Resources

Now that you have installed MapXtreme Java and configured it with your servlet container, and perhaps even displayed a sample map, take a minute to review the sources of information provided in this product.

MapXtreme Java Edition Developer's Guide

Use this as your primary hard copy reference for MapXtreme. Before trying to develop your own MapXtreme application, you should read the Chapter 1: Introduction, Chapter 3: Getting Started, and Chapter 4: PlanningYour Application.

MapJ API Specifications

For further information on the MapJ API, use the Javadoc-generated HTML Reference Guide. The online help contains detailed descriptions of the classes and methods of MapJ. This is installed in your MapXtreme directory under the \help\api directory. Choose index.htm to go to the opening page of the reference.

MapXtreme Web-based Developer's Guide

Use this HTML version of the Developer's Guide to access the most recently updated information on MapXtreme. This Web-based documentation also gives you the advantage of using links to access areas of related information. View this Developer's Guide on the MapXtreme Web site (http://www.mapxtreme.com/).

MapXtreme KnowledgeBase

The MapXtreme KnowledgeBase is a database of collected information on MapXtreme topics and other information that will be relevant to your application. You should review this before running your application.You may access the KnowledgeBase on the MapInfo Test Drive Center (http://testdrive.mapinfo.com/).

MapXtreme Discussion Area

The MapXtreme Discussion Area is available over theWeb and is hosted by the MapInfo Test Drive Center (http://testdrive.mapinfo.com/).

HAHTsite 4.0 Documentation Set

Included with MapXtreme Java Edition is HAHTsite 4.0 IDE and Application Server. This is an optional installation, provided for customers who do not already have a development environment and application server. The documentation set provided in PDF on the HAHTsite CD consists of the following:

- Application Server Installation Guide
- Application Server Administrator Guide
- IDE and IDE Installation Guide
- IDE Programmer's Guide
- IDE and IP User's Guide
- Widget Programming Guide

Additionally, HAHTsite provides an online help system covering IDE functions.

Sample Applications

There are several sample applications included with MapXtreme, some of which have already been introduced in the servlet container setup instructions. These applications are an excellent way to learn how to program MapXtreme applications. Each one includes an HTML tutorial that explains the features and the code that makes them possible. The sample applications are pre-compiled and located in **mxtjservletsamples30.jar** in the MapXtreme\server directory.

The sample applications may also be used as a base for your own application. You can simply modify the existing application to suit your solution needs. If you choose to modify one of the sample applications, you should make a copy of the entire directory and change only the copy. Any sample applications that require specific instructions for setup have included the information in their HTML tutorial.

Servlet example: HTMLEmbeddedMapServlet.java

This is a "client servlet" (a customizable servlet which in turn uses the MapXtremeServlet) that generates map images, which the end user views in an HTML form. This example demonstrates basic map navigation (e.g. click the map to pan/zoom, or type in a new zoom width to zoom), use of Layer Control (provided through an HTML form), display of a scalebar, and the ability to toggle the map image size. Server-side, the MapJ object and the business logic are managed by a

customizable servlet (which in turn uses MapXtremeServlet); client-side, the user runs a browser that displays HTML forms.

This sample is further discussed in Chapter 7:WritingYour Own Servlets.

Servlet/AWT Applet example: MapperServlet

This is a "client servlet" (a customizable servlet which in turn uses the MapXtremeServlet) that generates map images, which the end user views in an applet. Server-side, the MapJ object and business logic are managed by a customizable servlet (which in turn uses MapXtremeServlet); client-side, the user runs a thin applet. The applet is AWT-based, with no Swing or Java 2 Platform requirements, and no JavaBeans are involved.

Swing Applet example: SimpleMap

This sample is an applet that demonstrates using our JavaBeans (VisualMapJ, MapToolbar) to provide basic map display, map navigation, and theme/legend display using the Add ThemeWizard and LegendContainer. This applet communicates directly with MapXtremeServlet — no application server is involved. Advanced user interface features, such as Layer Control dialog, are provided through MapXtreme's JavaBeans; as a result, this applet requires Swing. The MapJ object and business logic reside on the client, in the applet.

This sample is further discussed in Chapter 6: MapXtreme JavaBeans

HAHTsite 4.0 example: JavaHelloWorld

This example demonstrates basic map navigation (e.g. click the map to pan/zoom, or type in a new zoom width to zoom), use of Layer Control (provided through an HTML form), and display of a scalebar.Server-side, the MapJ object and the business logic are managed by the HAHTsite Server application (which in turn uses MapXtremeServlet); client-side, the user runs a browser that displays HTML forms. This example is written using only Java code and does not show use of HAHTBasic.