7.4. RELEASE-NOTES-2.0-M4.TXT

```
Release Notes -- Apache Geronimo -- Version 2.0 - Milestone 4

Geronimo URLs
-------------
Home Page:     http://geronimo.apache.org/
Downloads:     http://geronimo.apache.org/downloads.html
Documentation: http://geronimo.apache.org/documentation.html
Mailing Lists: http://geronimo.apache.org/mailing.html
Source Code:   http://geronimo.apache.org/svn.html
Bug Tracking:  http://issues.apache.org/jira/browse/GERONIMO
Wiki:          http://cwiki.apache.org/geronimo


IMPORTANT
---------
This is a Milestone release, that means that is not the final version of
Apache Geronimo v2.0 Take a look at "Known Issues and Limitations" section for
further details.

Updated Information
-------------------
Please see http://cwiki.apache.org/GMOxDOC20/release-notes-20-m3txt.html for the latest information
on this release.

System Requirements
-------------------
You need a platform that supports the Sun JDK 5.0+ (J2SE 1.5.0+).

Most testing has been done on Linux, Mac OS X, and Windows.

Significant Changes in the 2.0-M4 Release
-----------------------------------------



Functional Characteristics for 2.0-M4
-------------------------------------



Significant Changes in the 2.0-M3 Release
-----------------------------------------
Overall this new milestone release is more stable and includes additional and enhanced support
to the features integrated in the previous releases.

Some of the new features added in this release include:

- JavaServer Faces 1.2 (via Apache MyFaces v1.2)
- Common Annotations for the Java Platform (partial support)
- Enterprise JavaBeans 3.0 (via Apache OpenEJB 3.0, partial support)
- Web Services Implementation (partial support)

*See "Functional Characteristics" section below for more details on the new functions.

Functional Characteristics for 2.0-M3
-------------------------------------

- Web Services (Apache Axis2)
  - JAX-WS 2.0 limited support (POJO only)
  - Web Services Metadata 2.0 limited support

- Web Services (Apache CXF)
  - JAX-WS 2.0 support for POJO and EJB
  - Web Services Metadata 2.0 limited support

- Limitations:
  - Redeploy does not work on Windows. Undeploy leaves behind files in the $geronimo_home/repository which are
```

```
locked.
    This prevents the same configuration from being deployed again. The server has to be stopped
    and the files deleted before the configuration can be deployed again.

Features and functions supported in the 2.0-M2 Release
------------------------------------------------------

- EJB 3.0 (via Apache OpenEJB project)
  Supported:
     - JPA (Custom Provider, App-managed, Container-managed) (Also supported in 2.0-M1)
     - POJO as a Business Interface (both local and remote)
     - POJO Session EJBs
     - geronimo-openejb.xml file not required unless you have geronimo specific information to configure
     - Deployment of annotated beans (@Stateful and @Stateless)
     - Injection via deployment descriptor
     - @Resource injection for env-entries, resource-refs, message-destinations, service-refs,  most resource-
env-refs
     - @EJB injection of ejb-refs and ejb-local-refs (Tomcat)
     - @PersistenceContext injection
     - @PersistenceUnit injection
     - JNDI references to the ejb
     - JNDI references from the ejb
     - Transaction support
     - Legacy component (i.e. home) interfaces on a Pojo session bean
     - Xml-based *and* annotation-based injection for ejbs, except for message-destinations,
       or SessionContext when the field or setter is not named setSessionContext
     - References to business interfaces, local or remote, from a servlet or an ejb
     - References to home interfaces, local or remote, from a servlet or an ejb
     - Extended JNDI and DI types
     - Deploy and Undeploy

     Simple EJB 3.0 example application available at:
       http://cwiki.apache.org/GMOxDOC20/using-some-of-ejb-30-functionalities.html

  Limitations:
     - No support for MDBs.
     - @EJB injection of ejb-refs and ejb-local-refs (Jetty)


- Web Services (Apache CXF)
  Supported:
     - Deployng a JAX-WS based POJO service that leverages annotations to simplify the creation of the Web
service.
     - Deploying a traditional JAX-RPC based POJO service
     - @Resource Annotations are fully supported provided they are also defined in the web.xml deployment
descriptor
     - The @Resource WebServiceContext annotation is fully supported.

     Simple Web Services example available at: http://cwiki.apache.org/GMOxDOC20/simple-web-service-with-jax-ws.
html

  Limitations:
     - No EJB support for Web Services in 2.0-M2
     - <service-ref> elements in the deployment descriptor or @WebServiceRef annotations are not processed.
     - Dynamically generated WSDL returned via ?wsdl request might be missing some information.
     - Dynamic clients (obtained using the javax.xml.ws.Service API) might not always work.


- JSTL 1.2
   Applications that use JSTL must add a dependency in their deployment plan to the jstl jar.
   Specifically:
    <dep:dependencies>
     <dep:dependency>
      <dep:groupId>jstl</dep:groupId>
      <dep:artifactId>jstl</dep:artifactId>
      </dep:dependency>
    </dep:dependencies>
   Alternatively, the jstl jar can be included in the application's WEB-INF/lib directory.


- JSF 1.2 - Not yet supported
```

```
 -JSF applications will deploy and start but will not execute yet. An updated MyFaces package
  will remedy this in the near future.

Features and functions supported in the 2.0-M1 Release
------------------------------------------------------

- Full Sun JDK 5.0+ (J2SE 1.5.0+)
- Servlet 2.5 (Tomcat)
- JSP 2.1 (Tomcat)
- JSP Debug 1.0 (Tomcat)
- Servlet 2.5 (Jetty)
- JSP 2.1 (Jetty - via Jasper)
- JSP Debug 1.0 (Jetty)
- JSF 1.2 (JSF applications won't execute)
- JSTL 1.2
- Common Annotations 1.0
- JAF 1.1
- JavaMail 1.4
- EJB 3.0 (JPA only)
- JTA 1.1
- JMS 1.1
- JACC 1.1

Installing & Starting Geronimo
------------------------------
To install, simply unpack the .zip (Windows) or tar.gz (Unix) file containing
Geronimo.

If you wish to modify the default ports that Geronimo will use, edit the file
<geronimo_home>/var/config/config.xml

Geronimo comes with batch and script files to control server start and stop
functions.  To see usage examples simply type geronimo.bat or geronimo.sh
command as appropriate for your platform.  It is necessary to set JAVA_HOME to
the copy of your Sun 5 JDK/JRE prior to executing the command.

Here is an example to set JAVA_HOME:

export JAVA_HOME=<JDK/JRE_home>

To see the available command options type:

<geronimo_home>/bin/geronimo.sh
or
<geronimo_home>\bin\geronimo.bat

The command will display help text instructing you as to how to start and stop
the Geronimo server.

If you prefer to start the server without a script file you can simply type:

java -jar <geronimo_home>/bin/server.jar

Once the server has started, you can access the Geronimo Administration Console
at http://localhost:8080/console/ . The default user name is "system" and the
default password is "manager".


Administration Console Security Configuration
---------------------------------------------
The default administration user/password for the Geronimo Administration Console
and command line deployment tool is system/manager.  You can change these defaults
directly from the Geronimo Administration Console by accessing Security -> Console
Realm and change the user name and password from the Console Realm Users portlet.

As an alternative, you can make the same changes by editing the
<geronimo_home>/var/security/users.properties and
<geronimo_home>/var/security/groups.properties files.


Deploying Applications
```

```
----------------------
Geronimo comes with deploy scripts and batch files to deploy J2EE modules or
applications. You can use the scripts or simply invoke the executable jar by
running the following command (note that you need to start Geronimo first):

<geronimo_home>/bin/java -jar deployer.jar deploy my-web-app.war [deploy plan]

You will need to use the username "system" and password "manager" unless you
customized those as described above.  The deployment plan argument is
optional -- you can pack a deployment plan into the application module, provide
it on the command line, or in some cases omit it entirely.

You can also use the "Login" command to avoid entering a user name and password
every time you use the deploy tool

For more information on the commands and options supported by the deploy tool,
run from within the Geronimo directory <geronimo_home>/bin:

java -jar deployer.jar help [command]

You can also graphically deploy applications and resources via the Geronimo
Administration Console available at http://localhost:8080/console/


Other Deployment Options
------------------------
As an alternative to the command-line deployer, you can copy application
modules into the <geronimo_home>/deploy/ directory and the
hot deployer service will deploy them automatically.  The command-line deployer
has some advantages, as it will output any
deployment errors to its own console rather than just the server log.

Additionally, Geronimo provides a Maven plugin that can deploy applications to
Geronimo as part of a Maven build.


Configuration
-------------
Most configuration attributes can be updated in the
<geronimo_home>/var/config/config.xml file.  The attributes most likely to be
changed are already included in the supplied config.xml file, while others may
need to be added manually.


Certification Status
--------------------
Apache Geronimo v2.0-M4, being a MILESTONE release is not yet certified.


Known Issues and Limitations
----------------------------
```