Deployment Guide
Witango for J2EE

June 2004

Witango Technologies Pty Ltd
Level 1,
44 Miller Street,
North Sydney, NSW, 2060
Australia

Telephone:+612 9460 0500
Fax:+612 9460 0502

Email:info@witango.com
Web: www.witango.com

Table of Contents

Witango for J2EE v1.5

Deployment instructions for Witango for J2EE v1.5

About This Guide

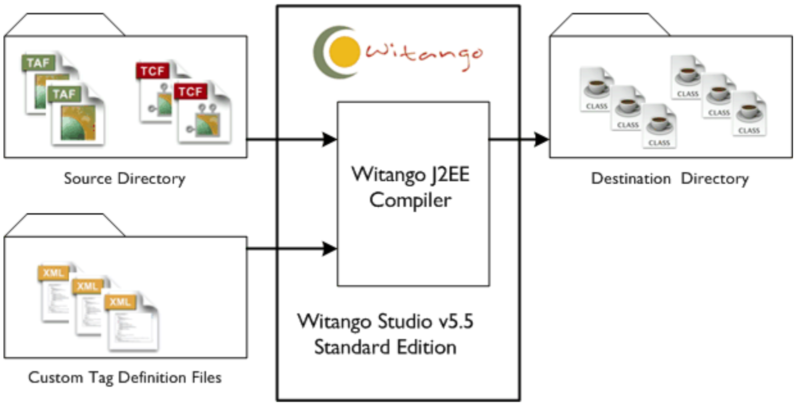
This guide outlines the instructions to deploy a compiled Witango application onto a J2EE compliant web server.

About Witango Compiler For Java

Witango Compiler for Java compiles Witango application files (which can be executed on Witango application server) into java servlets which can be executed on J2EE compliant web servers.

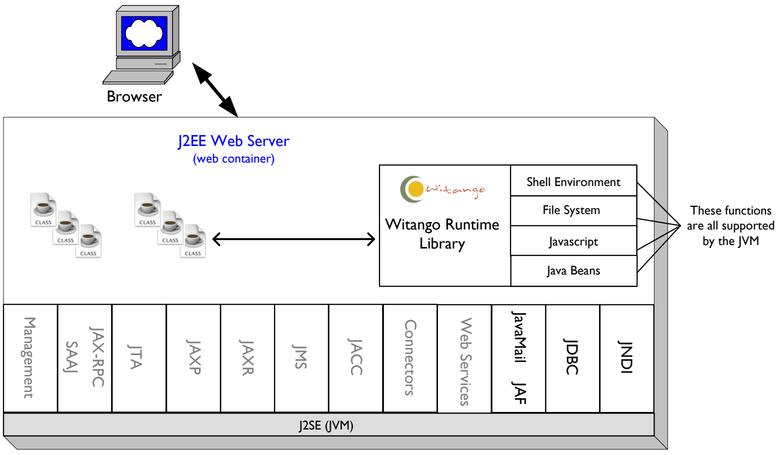
Witango Compiler for Java is written in java language and consists of two parts:

- A compiler which is integrated in the Witango Studio 5.5 Standard Edition. As shown in the figure below the Witango Studio compile functionality compiles Witango application files (.taf files and .tcf files) into java servlet files (.class files).



- Witango for J2EE, a runtime library which resides on the J2EE compliant web server of your choice. The compiled java servlets MUST have this runtime library in order to execute on a J2EE compliant web server.

  The diagram shown below illustrates the role the Witango for J2EE plays in the J2EE compliant web server.


Browser
J2EE Web Server
(web container)
CLASS
Witango Runtime Library
Shell Environment
File System
Javascript
Java Beans
These functions are all supported by the JVM
Management
JAX-RPC SAAJ
JTA
JAXP
JAXR
JMS
JACC
Connectors
Web Services
JavaMail JAF
JDBC
JNDI
J2SE (JVM)

Minimum System Specifications

Witango Technologies strongly recommends the following as minimum setup standards for Witango Application compiled to Java:

Web Server	J2EE compliant web server
Hardware	500 MHz CPU 128 Mb RAM 30 Mb hard disk space

Other Requirements

The Witango software is supplied to you via download from Witango's web site found at `http://www.witango.com/downloads` . The software installer is in a compressed archive format and will appear in the location that you choose during the download process.

Note The number which appears in the archive file name will vary in line with the current software version being supplied.

It is strongly recommended that you shut down your J2EE application server prior to installing Witango for J2EE.

Installing J2SE

In order to run Witango for J2EE, you need to have J2SE version 1.4.1 (JDK 1.4.1) or higher installed on your computer.

You can download it from:

`http://java.sun.com/j2se/index.jsp`, or

`http://java.sun.com`

Download the version suitable for your computer and operating system.

Note Documentation on installation will be available after you unpack the downloaded archive.

After you have downloaded and installed the J2SE 1.4.1 JDK, you will have the **java** and **javac** executables available on your computer. You will then need to do the following:

1 Create a new environment variable `JAVA_HOME` that points to the directory where you have installed the JDK.

Append `<JAVA_HOME>/bin` to your PATH environment variable under the `<JAVA_HOME>/bin` directory.You will find **java** and **javac** executables in this directory.

2 Verify that you have successfully installed the JDK 1.4 on your computer by opening a command line terminal and typing:

```
java -server -version
```

The system will tell you the java virtual machine you are using is of version 1.4.1.

3 Type into the command line terminal:

```
javac -help
```

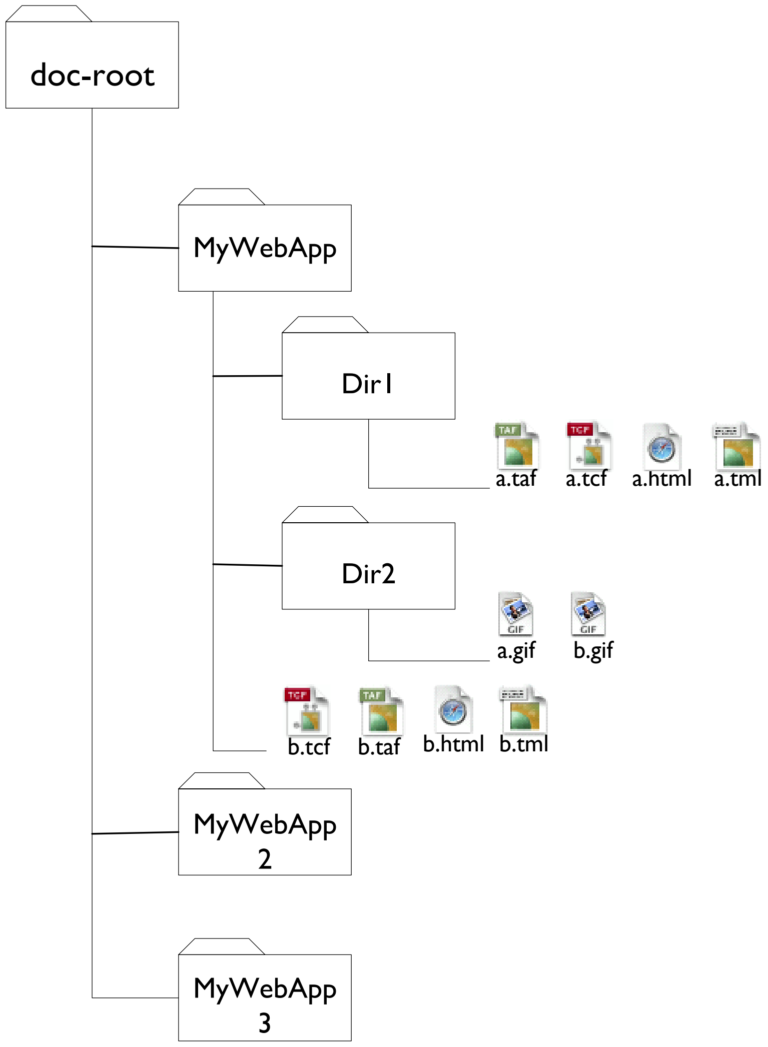
The system will display the **javac** program help information.

Note You must have both **java** and **javac** in your PATH environment variable. Otherwise you may encounter errors when you compile and run your J2EE web application.

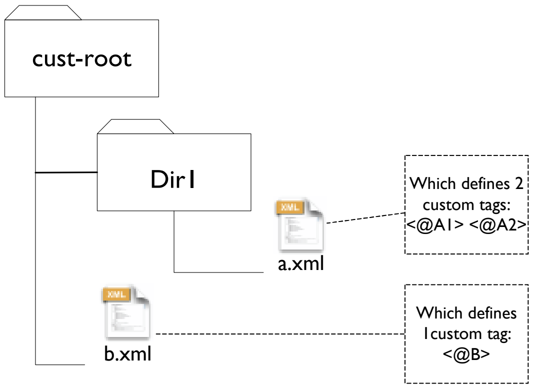
Example Witango Application

Throughout this guide the following application will be used as an example. It will be compiled into java servlets and then the compiled servlets will be deployed onto several different J2EE compliant web servers. You can apply the same steps to compile your own Witango application into a J2EE web application and run it.

1 Assume you have developed a Witango web application **MyWebApp** with following directory structure:



2 Assume the custom tags for this application are stored in the following directory structure:


cust-root
Dir1
a.xml
b.xml
Which defines 2 custom tags: <@A1> <@A2>
Which defines 1custom tag: <@B>

3 You would invoke the web application with URLs similar to:

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf
http://www.abc.com:port/MyWebApp/Dir1/a.html
http://www.abc.com:port/MyWebApp/b.taf
http://www.abc.com:port/MyWebApp/b.html
```

Note You will get an error if you try to invoke TCF files via HTTP requests directly.

Compile Sample Witango Application into Java Servlets

Before You Begin

Before you begin the compilation, please check:

- Your Witango Studio is of version 5.5 Standard edition or newer
- You have the following jar files installed in the Witango Studio Library Directory:

  `wicompiler.jar`

  `wiruntime.jar`

  (These files should have been copied during the installation process of your Witango Studio 5.5)

Note The compiler functionality is only available in the Standard edition of the Witango Studio. A trial key is available from the `witango.com` website.

To Compile your Witango Application

The compilation process works around 3 main steps being conducted on a source directory which contains a Witango web application. The steps are:

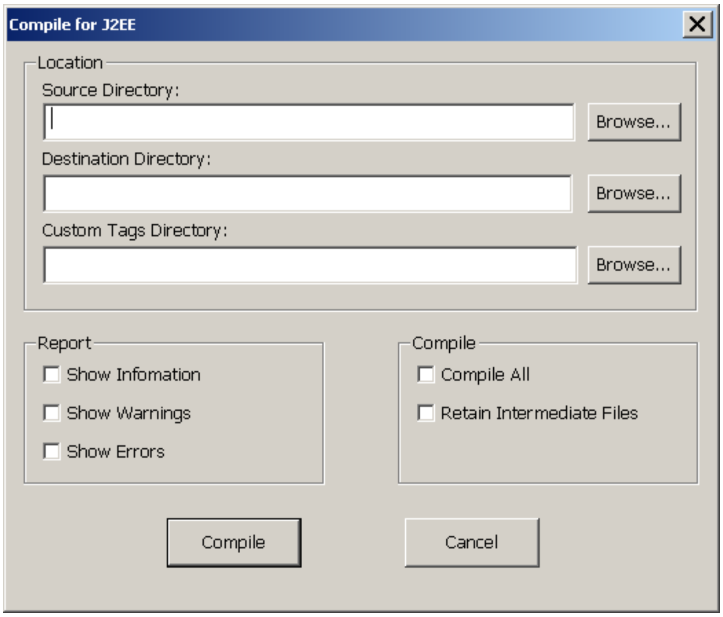
1. **Syntax Checking -** which checks all the `taf` and `tcf` files which are located in the specified source directory (and sub-directories contained therein) to ensure that all Witango meta tags exhibit correct syntax.
2. **Compiling for J2EE -** which takes all the `taf` and `tcf` files that are located in the specified source directory (and sub-directories contained therein) and compiles them into java class files and javabeans. The files that are created are located in a destination directory as specified by the user.
3. **Cleaning the Project -** which removes all of the machine generated files, created during the compilation process, from the destination directory as specified by the user.

Note Step 2 includes step 1, however, it is recommended that as a matter of good practice, step 1 is performed prior to step 2.

These steps all take place in the Witango Studio 5.5 Standard Edition. Full details of how this functionality works is outlined in the Witango 5.5 Studio User Guide - Chapter 23.

Specifying Directories during the Witango compile process

During the compile process the Witango Studio will pop a window as shown below where the user enters details of all source and destination directories.



For our example web application use the following directories:

Directory	**Example Data**
Source Directory	<doc-root>/MyWebApp
Destination Directory	<target-doc-root>
Custom Tags Directory	<cust-root>

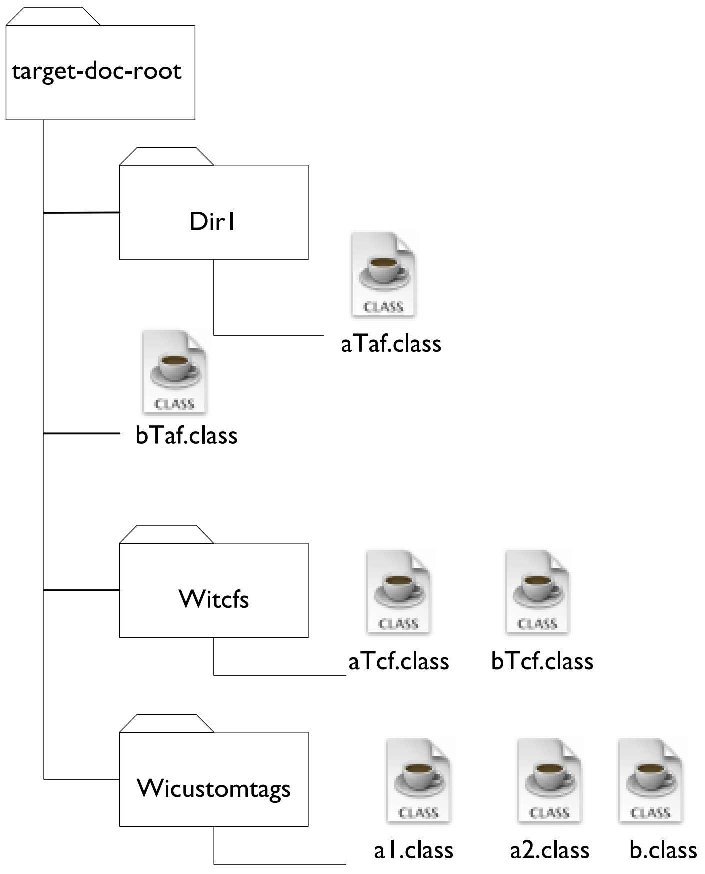
Unsucessful Compilation

If the compile process is not successful, you will be presented with error information in a Report. You will need to correct these errors within the Witango application file(s) and execute the compile process again.

Note The Report produced by the Witango Studio will help you to identify and locate the errors in the Witango application file(s). See the Witango Users Guide for further explanation of these features.

Successful Compilation

If the process is successful, you will have directory structure under <target-doc-root> as shown below:



Note If you have checked `Retain Intermediate Files` during the compilation process in the Witango Studio, intermediate files may exist in `Wijavasource` directory under <target-doc-root> with java files in it. These files are not necessary to deploy and run your compiled application.

Deployment of Compiled J2EE Web Application

Once you have your Witango J2EE web application you will need to deploy it to your J2EE web server. The following sections will outline how to deploy and run your java application on a number of different J2EE web servers.

Before You Start

Before you start, please check you have following jar files provided by Witango Technologies:

```
cryptix-jce-provider.jar
wiruntime.jar
js.jar
jakarta-oro-2.0.6.jar
log4j-1.2.6.jar
xercesImpl.jar
xalan.jar
xml-apis.jar
```

If you do not have them, you will not be able to run your compiled web application. These files supply additional functionality required by the compiled application.

These files should be installed in the WEB-INF /lib directory of each web application.

Some J2EE Web Servers (eg Apache Tomcat) do allow you to install these libraries (with the exception of `wiruntime.jar`) at server level instead of at each web application level. Thus all deployed web applications can see them.

Some J2EE Web Servers (eg Resin) allow you to install them as symbolic links so that th actual libraries can be stored in a single place. Then all deployed web applications can point to them.

Note When you install either the actual files or the symbolic links under WEB-INF/lib directory, some J2EE Web Servers (eg IBM Websphere) need to be told/configured to that these Witango libraries will take precedence over other server libraries installed at server level. Thus if there is a class name conflict, the Witango libraries will be used.

JDBC Drivers

During the compile process, all ODBC datasources will be converted to JDBC calls. This removes the need to have an ODBC to JDBC bridge.

Since we are using JDBC drivers to communicate with data source(s), you will need JDBC drivers installed and configured to run your compiled J2EE web application if it communicates with a database.

For example:

DataSource	JDBC Drivers
Microsoft SQL	`msbase.jar mssqlserver.jar msutil.jar`
Oracle	`ojdbc14.jar`
MySQL	`mysql-connector-java-3.0.6-stable-bin.ja`
OpenBase	`OpenBaseJDBC.jar`

Many JDBC drivers are freely available from database vendors. Make sure you get the correct JDBC driver for your computer platform where the J2EE web server is running.

Customized Java Beans

If your Witango web application uses customized java beans, your compiled Witango application will need to use them as well. These java beans are best packed as jar files e.g. `MyBeans.jar`.

This guide outlined instructions for the deployment of compiled Witango applications on following J2EE compliant web servers:

- Apache Tomcat Server (Free), running on a UNIX system
- Resin J2EE Server (Free), running on a UNIX system
- Jboss Server (Free), running on a UNIX system
- WebLogic Server, running on a UNIX system
- Sun ONE Application Server, running on a Windows system

The compiled web application is deployed as a J2EE web application in an exploded directory structure. If you need to deploy the application as a `WAR` archive, you can compress the directory structure by using the JDK jar utility and then deploy the resulting archive.

When you deploy a WAR archive, you must tell the J2EE server to decompress the archive during deployment.

Deployment on Apache Tomcat J2EE Server

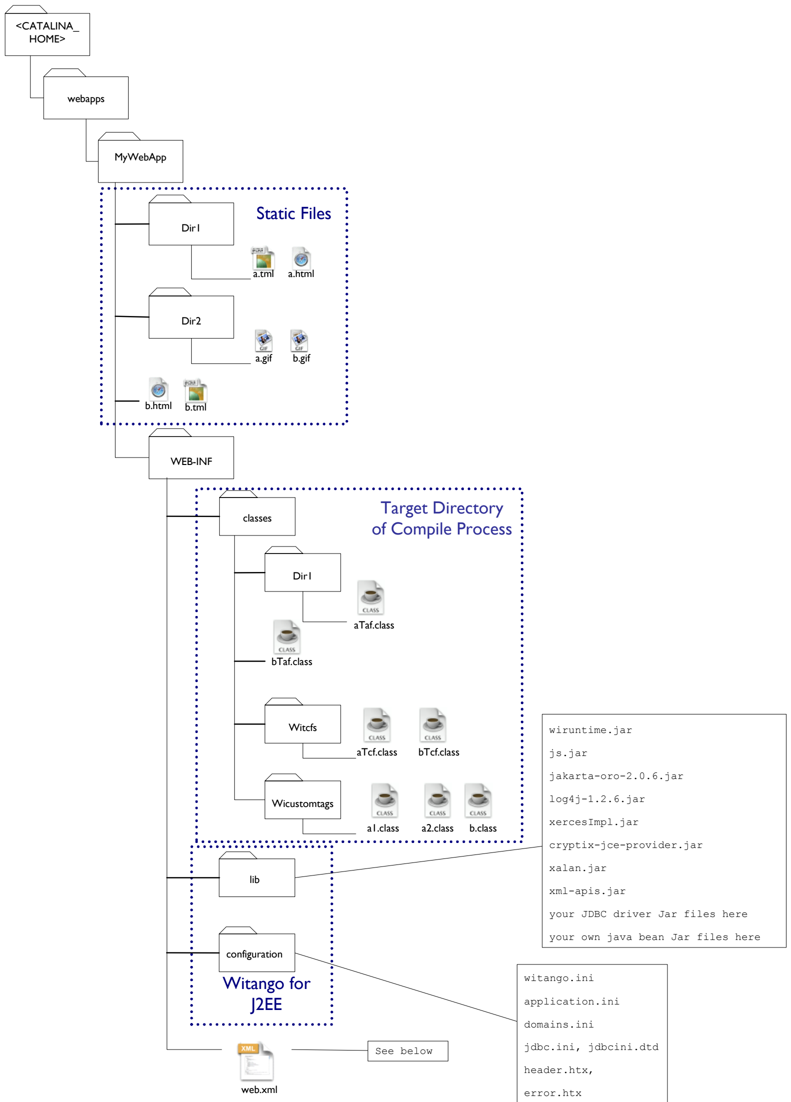
Deployment Process

1. Download Apache Tomcat Server from `http://jakarta.apache.org`. Use the latest stable release.
2. When you unpack the downloaded archive, the Tomcat server is virtually installed.
3. You must create a new environment variable CATALINA_HOME. This environment variable points to the directory where you have installed the Apache Tomcat Server.
4. You can verify your installation by going to `<CATALINA_HOME>/bin` and run `startup.sh`. Wait about 10-15 seconds. Then launch and point your browser to "http://hostname:8080". You will see Tomcat Server respond with its default page.
5. You can run `shutdown.sh` in the same directory in the startup window to stop the server.
6. To deploy your compiled web application, make a directory structure by:
   - Moving the contents of the **<target-doc-root>** which was the output of the compilation process to:

     `<CATALINA_HOME>/webapps/MyWebApp/WEB-INF/classes`
   - Moving the static **files which support your application** (.html, .gif, .tml, .thtml, etc) to:

     `<CATALINA_HOME>/webapps/MyWebApp`
   - Moving the **Witango for J2EE** downloaded from the Witango site to:

     `<CATALINA_HOME>/webapps/MyWebApp/WEB-INF/lib.`



The table below sets out the purpose of each configuration files.

Configuration File	Notes
witango.ini	which defines all Witango system scope variables using ‘name=value’ syntax

Configuration File	Notes
application.ini	optional. which defines all Witango application scope variables using ‘name=value’ syntax
domains.ini	optional. this file is identical to Witango server’s domains.ini
jdbc.ini jdbcini.dtd	optional. You must have these files if you have data source(s) your application communicates with
header.htx error.htx	optional. These files are identical to Witango server’s header.htx and error.htx
license.ini	You must have this file in order to run the compiled Witango web application

7 In order for your compiled Witango application to run a `web.xml` file must exist in the WEB-INF directory. This document is used by the J2EE web server to recognise servlets. A sample `web.xml` file is set out below.

Sample web.xml file

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!DOCTYPE web-app PUBLIC '-//Sun Microsystems, Inc.//DTD Web
Application 2.3//EN'
    'http://java.sun.com/dtd/web-app_2_3.dtd'>
<web-app>
    <listener>
        <listener-class>wiruntime.core.ServerInformation</listener-
class>
    </listener>
    <servlet>
        <servlet-name>TafServlet</servlet-name>
        <servlet-class>wiruntime.core.TafServlet</servlet-class>
        <init-param>
            <param-name>web-app-name</param-name>
            <param-value>MyWebApp</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet>
        <servlet-name>TcfServlet</servlet-name>
        <servlet-class>wiruntime.core.TcfServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>TafServlet</servlet-name>
        <url-pattern>*.taf</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>TafServlet</servlet-name>
        <url-pattern>*.tml</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>TafServlet</servlet-name>
        <url-pattern>*.thtml</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>TcfServlet</servlet-name>
        <url-pattern>*.tcf</url-pattern>
    </servlet-mapping>
</web-app>
```

Caution The text that you enter in <param-value> in above web.xml file (here is **MyWebApp**) will be used by the log files and appear in the event log file.

Note You may further modify above `web.xml` according to `http://java.sun.com/dtd/web-app_2_3.dtd`

Modifying the Server Configuration

1 You must modify the server configuration file

```
<CATALINA_HOME>/conf/server.xml
```

to include a definition for your new web applications. Open and search this XML file for an element `<Context>`. You will find something like:

```
<Context path='/examples' docBase='examples' debug='0'
reloadable='true' crossContext='true'>...</Context>
```

This is the pre-deployed `examples` web application.

2 Add a line similar to the pre-deployed example for your web application. For the example web application this would look like:

```
<Context path='/MyWebApp' docBase='MyWebApp'
debug='0' reloadable='true' crossContext='true' />
```

3 Now start the Tomcat server again and you can invoke the web application using URLs of similar format to:

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf
http://www.abc.com:port/MyWebApp/Dir1/a.html
http://www.abc.com:port/MyWebApp/b.taf
http://www.abc.com:port/MyWebApp/b.html
```

Note Apache Tomcat standalone Server default port is 8080.

Event Logs

When you run your compiled web application, an event log file and an access log file will be generated under directory

```
<CATALINA_HOME>/webapps/MyWebApp/WEB-INF/logs
```

CautionIf you have more than one web application deployed on the same server, you must assign unique `web-app-name` to each web application.

Deployment on Resin J2EE Server

Deployment Process

1 Download Resin J2EE Server from `http://www.caucho.com`.

2 When you unpack the downloaded archive, the Resin server is virtually installed.

3 Create a new environment variable `RESIN_HOME`. It should point to the directory where you have installed Resin J2EE Server.

4 You can verify your installation by going to `<RESIN_HOME>/bin` and run `httpd.sh`. Wait until you see the server tells you it is listening on port 8080. Then launch and point your browser to `http://hostname:8080` .The Resin Server will respond with its default page.

5 To stop the server focus the startup window and press `Ctrl-C` on keyboard.

6 To deploy your compiled web application, make a directory structure by:

- Moving the contents of the **<target-doc-root>** which was the output of the compilation process to:

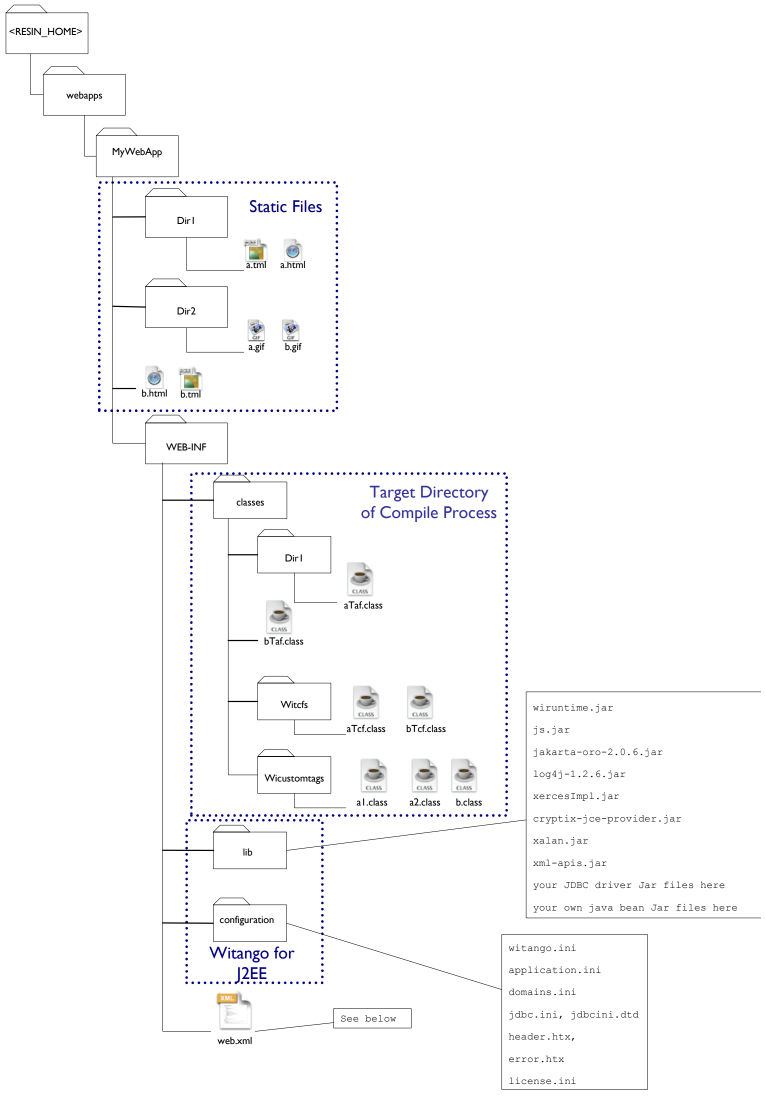
  `<RESIN_HOME>/webapps/MyWebApp/WEB-INF/classes`

- Moving the **static files** which support your application (.html,.tml, .gif, .thtml, etc) to:

  `<RESIN_HOME>/webapps/MyWebApp`

- Moving the **Witango Runtime library** downloaded from the Witango site to:

  `<RESIN_HOME>/webapps/MyWebApp/WEB-INF/lib`



7 In order for your compiled Witango application to run a `web.xml` file must exist in the WEB-INF directory. This document is used by the J2EE web server to recognise servlets. A sample `web.xml` file can be found on page 14.

8 Start the Resin Server again.

9 Now you can invoke the web application using the URLs of similar format to:

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf
http://www.abc.com:port/MyWebApp/Dir1/a.html
http://www.abc.com:port/MyWebApp/b.taf
http://www.abc.com:port/MyWebApp/b.html
```

Note Resin standalone server default port is 8080.

Event Logs

When you run your compiled web application, an event log file and an access log file will be generated under directory

```
<RESIN_HOME>/webapps/MyWebApp/WEB-INF/logs
```

Deployment on Jboss J2EE Server

Deployment Process

1 Download Jboss J2EE Server from `http://www.jboss.org`.

2 When you unpack the archive, the server is virtually installed.

3 Create a new environment variable `JBOSS_HOME`. It should point to the directory where you have installed the Jboss J2EE Server.

4 You can verify your installation by going to `<JBOSS_HOME>/bin` and run `run.sh`. Wait until the messages being printed to your screen are complete. Then launch and point your browser to `http://hostname:8080`. You will see a page like `HTTP ERROR: 404 / Not Found RequestURI=/`. This is because you have not deployed anything onto the server yet and the server does not provide any default contents.

5 To stop the server focus the startup window and press Ctrl-C on keyboard.

6 To deploy your compiled web application, make a directory structure by:

- Moving the contents of the **<target-doc-root>** which was the output of the compilation process to:

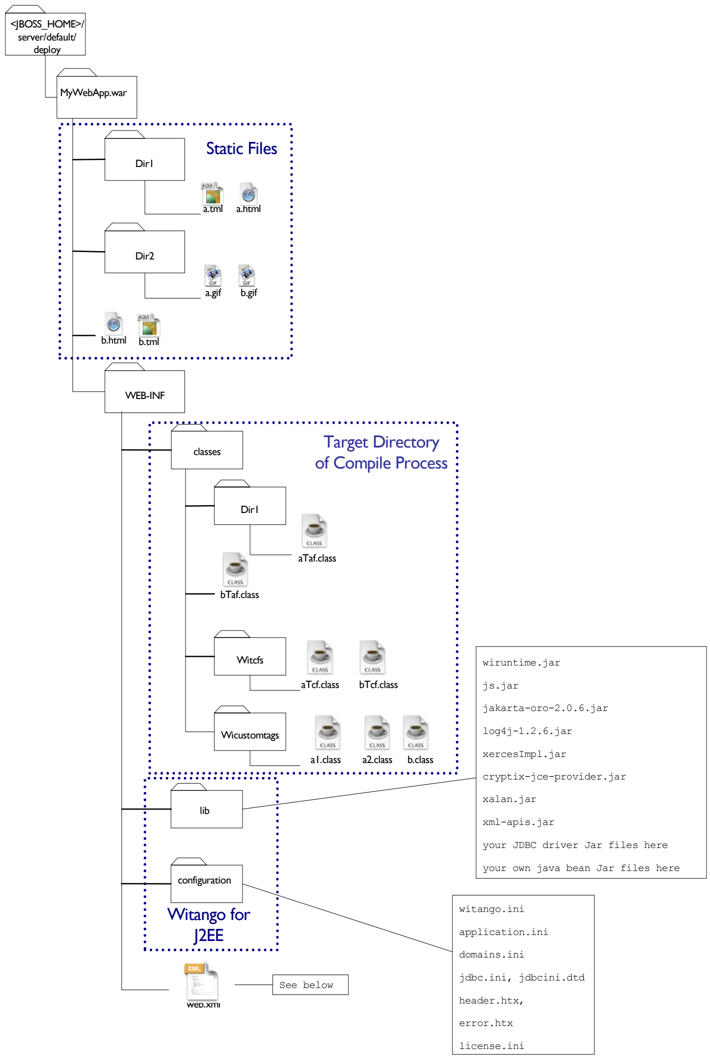
  <JBOSS_HOME>`/server/default/deploy/MyWebApp.war/WEB-INF/classes`

- Moving the **static files** which support your application (.html, .tml .gif, .thtml, etc) to:

  <JBOSS_HOME>`/server/default/deploy/MyWebApp.war`

- Moving the **Witango Runtime library** downloaded from the Witango site to:

  <JBOSS_HOME>`/server/default/deploy/MyWebApp.war/WEB-INF/lib`



Note Notice the directory name here is `MyWebApp.war` instead of `MyWebApp`

---

7 In order for your compiled Witango application to run a web.xml file must exist in the WEB-INF directory. This document is used by the J2EE web server to recognise servlets. A sample `web.xml` file can be found on page 14.

8 Now start the Jboss server again.

9 Now you may invoke the web application using the URLs of format similar to:

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf
http://www.abc.com:port/MyWebApp/Dir1/a.html
http://www.abc.com:port/MyWebApp/b.taf
http://www.abc.com:port/MyWebApp/b.html
```

Note Jboss standalone server default port is 8080.

Event Logs

When you run your compiled web application, an event log file and an access log file will be generated under the following directory:

```
<JBOSS_HOME>/server/default/deploy/MyWebApp.war/WEB-INF/logs
```

Special Notes for Jboss version 3.2.3 or higher

1 If you deploy more than one Witango web application on the same server i.e. under `<JBOSS_HOME>/server/default/deploy`, you need to edit the file `<JBOSS_HOME>/server/default/deploy/jbossweb-tomcat41.sar/META-INF/jboss-service.xml`. Search for

```
<server>
   <mbean code="org.jboss.web.tomcat.tc4.EmbeddedTomcatService"
name="jboss.web:service=WebServer">
      ...
      <attribute name="UseJBossWebLoader">true</attribute>
      ...
   </mbean>
</server>
```

and change that attribute into false: `<attribute name="UseJBossWebLoader">false</attribute>`. This will isolate the web applications.

2 If you use DOM tags or ELEMENT tags, you need to overwrite `<JBOSS_HOME>/lib/xml-apis.jar` and `<JBOSS_HOME>/lib/xercesImpl.jar` with the jar files supplied by Witango download.

Note You might need to perform similar steps for Jboss version higher than 3.2.3. This is because Jboss default configuration does not fully conform to J2EE Servlet Specification.

Deployment on WebLogic J2EE Server

1 WebLogic J2EE Server is a commercial product, and therefore you cannot download a free copy. You can however download an evaluation version and try it out. Download WebLogic Server from `http://www.bea.com`.

Note If you downloaded an installer e.g. `net_server810_linux32.bin` run the installer on your platform. Make sure you download the correct version of the installer for your computer platform.

2 Follow the instructions during installation being guided by the installer, the installer may download further files from BEA website.

3 After installation create the following environment variables :

Environment Vars	Notes
`BEA_HOME`	points to your root installation directory, you can customize it during installation
`WL_HOME`	points to where you have installed the WebLogic Server. It is probably `<BEA_HOME>/weblogic81`
`PATHSEP`	your system PATH delimiter. On Unix it is ":" and on Windows it is ";"
`CLASSPATHSEP`	your system CLASSPATH delimiter. On Unix it is ":" and on Windows it is ";"

Note If you do not set these enivironmnet variables up manually, the running scripts provided by WebLogic should set them up.

4 You can verify your installation by going to `<WL_HOME>/samples/domains/examples` and run `startExamplesServer.sh`. Wait until the messages have finished printing to your screen. Then launch and point your browser to `http://hostname:7001`. You will see WebLogic Server respond with its example application.

5 To stop the server focus the startup window and press `Ctrl-C` on keyboard.

6 Now create a new domain of your own. Go to `<WL_HOME>/common/bin` and run `config.sh`. Follow the instructions to create a new domain and server(s) in this domain. For example, you could create a domain called `MyDomain` under directory `<BEA_HOME>/user_projects`. Then you will have a directory `<BEA_HOME>/user_projects/MyDomain` created for you and all shell scripts and configuration files automatically generated for you under that directory.

7 Then create a new directory structure under the domain directory for your web application and deploy your compiled web application as follows:

- Moving the contents of the **<target-doc-root>** which was the output of the compilation process to:

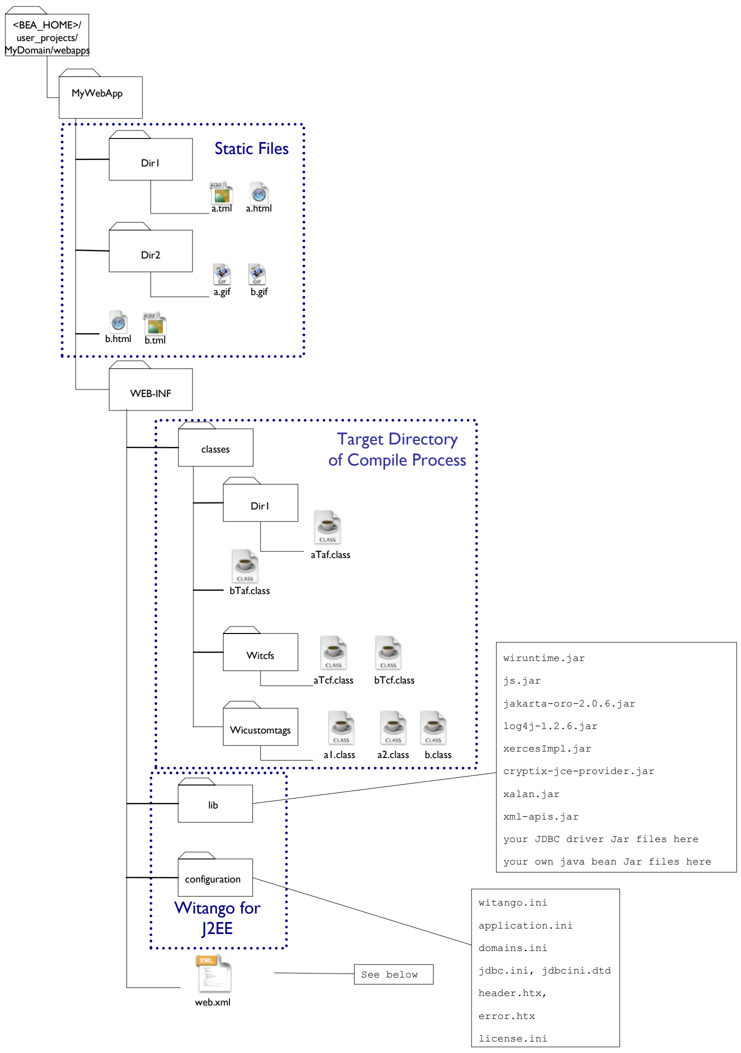
  ```
  <BEA_HOME>/user_projects/MyDomain/webapps/
  MyWebApp/WEB-INF/classes
  ```

- Move the **static files** which support your application (.html, .gif, .tml, .thtml, etc) to:

  ```
  <BEA_HOME>/user_projects/MyDomain/webapps/
  MyWebApp/
  ```

- Move the **Witango Runtime library** downloaded from the Witango site to:

  ```
  <BEA_HOME>/user_projects/MyDomain/webapps/
  MyWebApp/WEB-INF/lib
  ```



8 In order for your compiled Witango application to run a web.xml file must exist in the WEB-INF directory. This document is used by the J2EE web server to recognise servlets. A sample `web.xml` file is set out on page 14.

9 Now go to `<BEA_HOME>/user_projects/MyDomain` and run `startWeblogic.sh` to start the weblogic server instance under the domain MyDomain. The `startWebLogic.sh` is created

automatically for you when you create **MyDomain** at Step 6. You can further modify the shell script if required.

10 Invoke the administration console of MyDomain to deploy your compiled web application **MyWebApp** under MyDomain. To do that, launch and point your browser to `http://hostname:7001/console`. **hostname** is where the WebLogic J2EE server is running. You will see a login window for doing administration work.

11 Login using the username and password you provide when you create the domain at Step 6.

It is not enough to simply copy the files onto the server, the application must be deployed, which means, the server must have the necessary information recorded in its configuration file.

12 Then you will see a graphic administration window. You will need to read and follow the documentation provided by WebLogic (`http://edocs.bea.com or http://edocs.bea.com/wls/docs81/index.html`) on how to deploy and undeploy a web application on WebLogic Server 8.1. Deploy the compiled web application in exploded directory structure under

```
<BEA-HOME>/user-projects/MyDomain/webapps/MyWebApp
```

13 If for example you have deployed the web application using `MyWebApp`, you may invoke the compiled web application using a URLs of a format similar to:

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf

http://www.abc.com:port/MyWebApp/Dir1/a.html

http://www.abc.com:port/MyWebApp/b.taf

http://www.abc.com:port/MyWebApp/b.html
```

Note WebLogic Server default port is 7001.

Event Logs

When you run your compiled web application, an event log file and an access log file will be generated under the following directory:

```
<BEA-HOME>/user-projects/MyDomain/webapps/MyWebApp/WEB-
INF/logs
```

Deployment on Sun[tm] ONE Application Server 7

Note (Example OS is Windows)

1 Sun[tm] ONE Application Server 7 Standard Edition can be downloaded from `http://www.sun.com`. It supports Solaris 8 & 9 platform and Windows 2000 & XP platform. The download comes with a free license for development and production deployment of applications. It can be upgraded to a fully featured Sun ONE Application Server Standard Edition with a license key.

2 Install the server after you obtain a distribution. You can find a full documentation of Sun ONE Application Server at `http://docs.sun.com`. Follow the documentation to install the server and familiarise yourself with it. You may reuse your current java JDK installation during server installation. Assume you have installed Sun ONE Application Server under `C:\Sun\AppServer7` on a Windows platform and have added `C:\Sun\AppServer7\bin` into your PATH.

3 Start the server using

```
asadmin start-domain --domain domain1
```

in a command line window.

4 Launch and point your browser to `http://hostname:4848` (4848 is the default administration port) and type in administration username and password you supplied during installation. Go through the pages to get familiar with the configuration tasks.

5 The server can be stopped by typing:

```
asadmin stop-domain --domain domain1
```

in the same command line window.

6 After installation you will have one domain called **domain1**. Under **domain1** you have an **Admin Server** and an application server **server1**. Under the application server **server1** you have a virtual server called again **server1** and an HTTP listener called **http-listener-1** which will listen on port 8080. The virtual server **server1** is assigned HTTP listener **http-listener-1**. This acts as a basic default configuration for the Sun ONE Web Server.

7 To deploy your compiled web application, make a directory structure by:

- Moving the contents of the **<target-doc-root>** which was the output of the compilation process to:

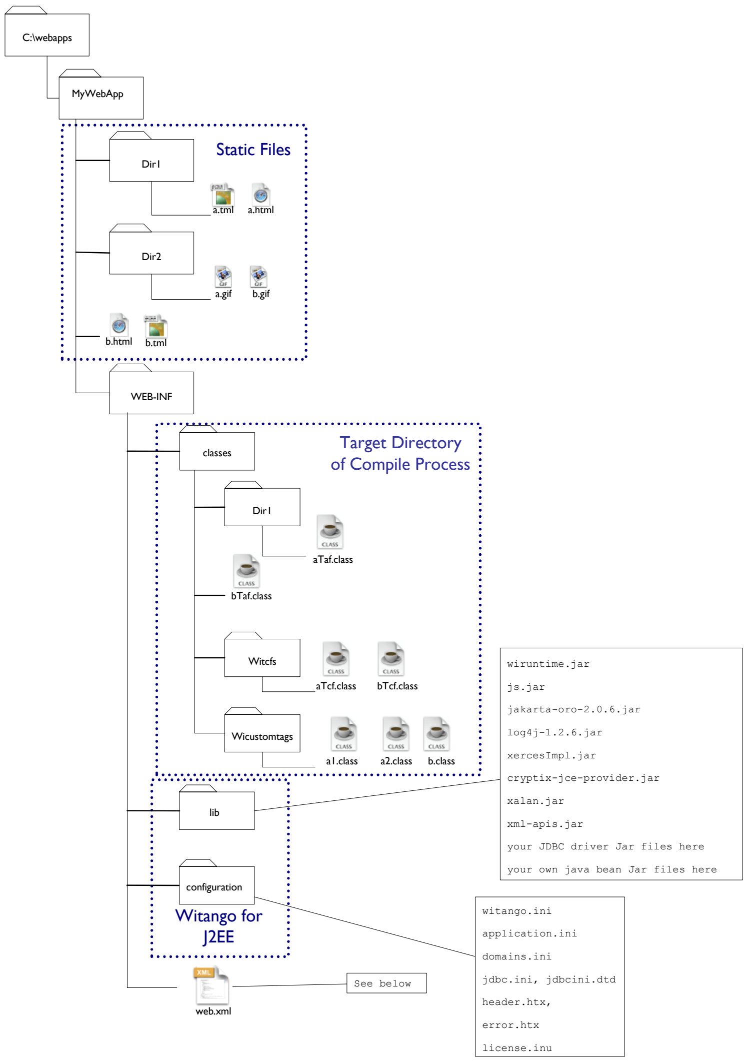
  `C:\webapps\MyWebApp\WEB-INF\classes`

- Moving the **static files** which support your application (.html, .gif, .thtml, etc) to:

  `C:\webapps\MyWebApp`

- Moving the **Witango Runtime library** downloaded from the Witango site to:

  `C:\webapps\MyWebApp\WEB-INF\lib`



```
wiruntime.jar
js.jar
jakarta-oro-2.0.6.jar
log4j-1.2.6.jar
xercesImpl.jar
cryptix-jce-provider.jar
xalan.jar
xml-apis.jar
your JDBC driver Jar files here
your own java bean Jar files here
```

```
witango.ini
application.ini
domains.ini
jdbc.ini, jdbcini.dtd
header.htx,
error.htx
license.inu
```

8 This directory structure is made outside the Sun ONE Application Server installation directory.

Note The name `webapps` and `MyWebApp` in above structure is not required, it is shown here as an example.

9 In order for your compiled Witango application to run a web.xml file must exist in the WEB-INF directory. This document is used by the J2EE web server to recognise servlets. A sample `web.xml` file is set out on page 14.

10 Open a command line terminal and type:

```
cd C:\webapps\MyWebApp
```

and then type

```
jar cvf MyWebApp.war *
```

Under `C:\webapps\MyWebApp` you will now find a file called `MyWebApp.war`

11 Start the server and browse the administration page `http://hostname:4848` to deploy the WAR archive MyWebApp.war onto the server.

Note You may need to tell the server to decompress the WAR archive during deployment - this is usually indicated through the presence of a configuration item on the administration page.

12 You can now use URL's such as those set out below to invoke the web application

```
http://www.abc.com:port/MyWebApp/Dir1/a.taf
http://www.abc.com:port/MyWebApp/Dir1/a.html
http://www.abc.com:port/MyWebApp/b.taf
http://www.abc.com:port/MyWebApp/b.html
```

Note If you need to modify security policy of application server **server1**, look at the file `C:\Sun\AppServer7\domains\domain1\server1\config\server.policy`. This file defines what java security permissions your code will have.

Event Logs

When you run your compiled web application, an event log file and an access log file will be generated under the following directory:

```
C:\Sun\AppServer7\domains\domain1\server1\applications\
j2ee-modules\MyWebApp_1\WEB-INF\logs
```

Entering Your License Information

The Witango for J2EE license information is made up of two parts:

- a license key; and
- a license file.

You will receive both of these via email when you activate your PIN on the witango.com web site. More information on receiving your PIN is available on page 35 .

Entering the Witango License Key

To enter your license key follow the steps below:

1. Open the `witango.ini` file in a text editor. It is located in the following directory:

   ```
   .../WEB-INF/configuration/witango.ini
   ```

2. Search the line `LICENSE=` and paste your serial number immediately after “=” .

   Note If you can not find such a line in the file, simply add a new line `LICENSE=your-serial_number` into the file.

3. **Save** the changes made to your `witango.ini` file.

Installing the Witango License File

To install the Witango License File copy the `license.ini` file to the configuration directory for your application:

```
.../WEB-INF/configuration/license.ini
```

Changing Witango License Information

You may at some time need to change your Witango for J2EE license information, for example, going from Trial to Standard mode. To do this:

1. Open the `witango.ini` file in a text editor. It is located in the following directory:

   ```
   .../WEB-INF/configuration/witango.ini
   ```

2. Search the line `LICENSE=` and paste your new serial number immediately after “=” . Then Save your changes.

3. Copy your new `license.ini` file over the old file in the configuration directory for your application:

   ```
   .../WEB-INF/configuration/license.ini
   ```

Witango Log Files

When you run your compiled Witango web application, event log file and access log file will be generated under directory

```
    .../WEB-INF/Logs
```

on the J2EE complaint web server. And the text that you enter in <param-value> in ".../WEB-INF/web.xml" file (in the sample `web.xml` on page 14 the text used is "MyWebApp") will be used by the log files and appear in the event log file.

Where more than one application is deployed

If you have more than one web application deployed on the same server, you must assign a unique "web-app-name" to each web application.

Backing up

Log files are backed up daily.

Modification of TAF, TCF, Custom Tags

Recompiling modified web applications

If you modify your .taf file(s), .tcf file(s) or custom tag definition .xml file(s), you need re-compile your web application.

Changes affecting only file which is modified

If the change is limited within the modified file(s) and does not affect other Witango application file(s), you may uncheck "Compile All" during re-compilation to only re-compile the modified file(s).

Changes affecting other unmodified files

If the change affects other unmodified file(s) e.g. you have modified an action name in one file and this action is the branch target action of some other Witango application file(s), you need to check "Compile All" to rebuild the whole web application.

Hot deployment of modified web applications

Witango for J2EE supports hot deployment, so that modified compiled web applications can be automatically re-deployed. You will need to consult your J2EE web server documentation whether your web server supports this functionality and if so, how it is implemented.

If your web server does not support hot deployment you will need to un-deploy your old version of the compiled web application before re-deploying the modified application.

Witango for J2EE Registration

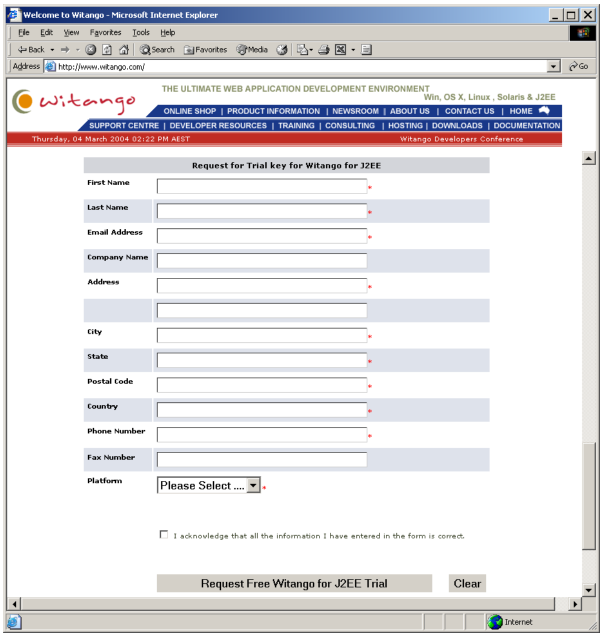
The Witango for J2EE Installer will install a fully featured edition of Witango for J2EE by default. This product is capable of running in 2 modes:

- **TRIAL MODE** - a 30 day time limited fully featured edition of Witango for J2EE. This mode puts a Trial Banner at the bottom of every page served.
- **APPLICATION MODE** - a fully featured Witango for J2EE which is NOT time limited but only allows one web application to be deployed and run on a J2EE web server.
- **STANDARD MODE** - a fully featured Witango for J2EE which is NOT time limited.

The mode in which Witango for J2EE will run in depends on the License Key entered.

Obtaining a Trial Key

To request a Trial License Key for Witango for J2EE, visit `witango.com` and follow the 30 DAY TRIAL KEY button. You will be taken to a page which outlines the steps to trial Witango for J2EE. Step 5 includes the form to request a trial key. Complete the form and you will be emailed a trial license key and license file.


Welcome to Witango - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.witango.com/
Go
THE ULTIMATE WEB APPLICATION DEVELOPMENT ENVIRONMENT
Win, OS X, Linux , Solaris & J2EE
ONLINE SHOP | PRODUCT INFORMATION | NEWSROOM | ABOUT US | CONTACT US | HOME
SUPPORT CENTRE | DEVELOPER RESOURCES | TRAINING | CONSULTING | HOSTING | DOWNLOADS | DOCUMENTATION
Thursday, 04 March 2004 02:22 PM AEST
Witango Developers Conference
Internet

Request for Trial key for Witango for J2EE

First Name	*
Last Name	*
Email Address	*
Company Name	
Address	*
City	*
State	*
Postal Code	*
Country	*
Phone Number	*
Fax Number	
Platform	Please Select .... *

☐ I acknowledge that all the information I have entered in the form is correct.

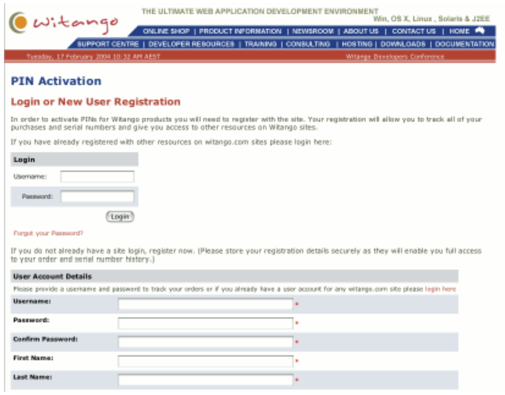
Request Free Witango for J2EE Trial | Clear

Note This request is NOT immediately processed as Witango Technologies reserves the right to refuse any request for a trial license key. Any request that is made with erroneous details or uses an anonymous email address (e.g. yahoo.com or hotmail.com), will be automatically rejected. As a general rule only one request for a trial key per person will be processed. Each request will be processed on the next working day (Australian Time).

Obtaining a Standard Key

Standard keys for your Witango for J2EE may be purchased onine from `witango.com`. Upon purchase you will be issued with a PIN. This PIN can then be activated on the `witango.com` website to receive your license key and license file.

To activate your PIN, follow the ACTIVATE PIN button located on the homepage. You will then be taken to a login page where you can register as a user on the site, or, login using your existing registration details.


witango
THE ULTIMATE WEB APPLICATION DEVELOPMENT ENVIRONMENT
Win, OS X, Linux , Solaris & J2EE
ONLINE SHOP | PRODUCT INFORMATION | NEWSROOM | ABOUT US | CONTACT US | HOME
SUPPORT CENTRE | DEVELOPER RESOURCES | TRAINING | CONSULTING | HOSTING | DOWNLOADS | DOCUMENTATION
Tuesday, 17 February 2004 10:32 AM AEST
Witango Developers Conference
Login
Forgot your Password?

PIN Activation

Login or New User Registration

In order to activate PINs for Witango products you will need to register with the site. Your registration will allow you to track all of your purchases and serial numbers and give you access to other resources on Witango sites.

If you have already registered with other resources on witango.com sites please login here:

Login

Username:

Password:

If you do not already have a site login, register now. (Please store your registration details securely as they will enable you full access to your order and serial number history.)

User Account Details

Please provide a username and password to track your orders or if you already have a user account for any witango.com site please login here

Username: *

Password: *

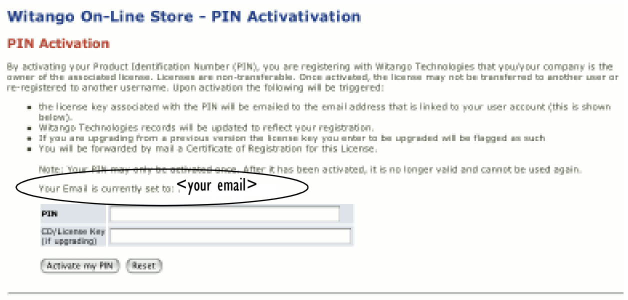
Confirm Password: *

First Name: *

Last Name: *

Note It is important that you use your correct details as this information is used to record ownership details of the product licensed andverify ownership for future upgrades. The email address you enter is where your license key will be sent to. Details cannot be changed online. If you require a change in registration details you will need to notify `info@witango.com` accordingly.

Upon successful login or registration you will be taken to a PIN Activation page where you simply enter your PIN. Then push the **Activate my PIN** button and you will be sent via email your license key and license file. You will also be mailed by post a Certificate of Registration.



Witango On-Line Store - PIN Activativation

PIN Activation

By activating your Product Identification Number (PIN), you are registering with Witango Technologies that you/your company is the owner of the associated license. Licenses are non-transferable. Once activated, the license may not be transferred to another user or re-registered to another username. Upon activation the following will be triggered:

- the license key associated with the PIN will be emailed to the email address that is linked to your user account (this is shown below).
- Witango Technologies records will be updated to reflect your registration.
- If you are upgrading from a previous version the license key you enter to be upgraded will be flagged as such
- You will be forwarded by mail a Certificate of Registration for this License.

Note: Your PIN may only be activated once. After it has been activated, it is no longer valid and cannot be used again.

Your Email is currently set to: <your email>

PIN	
CD/License Key (if upgrading)	

Note Please check that your email address shown on the PIN Activation screen is correct and current, as this is the email address where your license key will be sent.

Entering your Witango License Key

Before Witango for J2EE will execute an application a license key and license file must be correctly configured. Instructions to do this are outlined on page 31.

If the License Key is valid, Witango for J2EE will automatically launch in the mode corresponding the the license key entered.

Invalid License Keys

If the License Key you enter is not valid, a corresponding error message will appear in the log file for your J2EE Web Server. It will also be written to your Witango Events Log.

What mode is Witango J2EE running in ?

To detemine what mode your Witango for J2EE is operating in, review the `witangoevents.log` file located in your `WEB-INF/logs` directory.

Operating in Trial Mode

If you are operating in Trial mode, 'Trial' will be shown on the license type entry in the log. A sample extract from a `witangoevents.log` file is shown below:

```
[main] 2004-03-02 00:02:30.563MyWebAppName.WiRuntimeINFOProduct:
Witango Java Server version 150 (Mac OS X) [Standard Edition]

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeINFOProduct
Licensed To: Witango Technologies Test Lic

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeINFOLicense Type:
Trial

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeWARNTime Limited
License: Expires on 2004-04-15
```

```
[main] 2004-03-02 00:02:30.566MyWebAppName.TafServletINFOApplication
server started
```

Operating in Standard Mode

If you are operating in Standard mode, 'Full' will be shown on the license type entry in the log. A sample extract from a `witangoevents.log` file is shown below:

```
[main] 2004-03-02 00:02:30.563MyWebAppName.WiRuntimeINFOProduct:
Witango Java Server version 150 (Mac OS X) [Standard Edition]

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeINFOProduct
Licensed To: Witango Technologies Test Lic

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeINFOLicense Type:
Full

[main] 2004-03-02 00:02:30.565MyWebAppName.WiRuntimeWARNTime Limited
License: Expires on 2004-04-15

[main] 2004-03-02 00:02:30.566MyWebAppName.TafServletINFOApplication
server started
```

Changing your Witango License Key and license.ini file

You may wish to change your license key once entered, for instance, if you are going from LITE or Trial mode to Standard mode. To do this, see the instructions on page 31.

Where to get help

Documentation for Witango is available in both PDF and HTML formats. The HTML versions of the **Studio Users Guide** and the **Programmer's Guide** are installed as part of the Witango 5.5 Studio install. These options can be found in the Witango HELP menu. A full list of current documentation in PDF format can be found on `http://www.witango.com/documentation`.

Studio Users Guide

The Studio Users Guide can be accessed by via the HELP menu within the Studio. This Guide contains all the information you need to be able to effectively use the Studio. This manual will give you an understanding of the layout of the development studio and what function each action performs. The manual covers the following topics:

- Introduction
- Using Projects and Source Control
- Using Data Sources
- Using Snippets
- Setting Preferences
- Working with Meta Tags
- Working With Variables
- Building Actions Using Witango Builders
- Configuring the Search Builder
- Configuring the New Record Builder
- Using Actions
- Grouping Actions
- Using Basic Database Actions
- Using Control Actions
- Extending Witango Functionality
- Sending Electronic Mail From Witango
- Reading, Writing, and Deleting Files
- Using Advanced Database Actions
- Understanding Objects in Witango
- Using Objects
- Witango Class Files
- Compiling Witango Application Files

Programmer's Guide

The Programmers Guide can be accessed by via the HELP menu within the Studio. This manual contains the meta language reference and all information that is required for day to day programming. The manual covers the following topics:

- Introduction
- Witango Studio Basics
- Meta Tags
- Custom Meta Tags
- Working With Variables
- Document Object Model
- Configuration Variables
- Witango Server Error Codes
- <@CALC> Expression Operators
- Lists of Meta Tags
- Using DLLs With Witango

Tutorials

Tutorials have been made available for both the Windows and the OS X Studios. When you have completed the tutorials, you should have a good grasp of the basics required to create web applications with Witango.

They consist of a pdf document, sample database files, sample data, example taf files and instruction notes. These tutorials are available from the documentation section of witango.com.

The tutorials are designed for absolute beginners although a basic understanding of HTML, Datasources, Databases and Web Servers would be an advantage.

Course Content

- **Tutorial A -** The lessons in this tutorial introduce you to the basics of developing a Witango application file. The concepts you learn include creating a dynamic web page, passing values within an application file, and setting up a database search.
- **Tutorial B -** The lessons in this tutorial show you how to create a data source. You need to create this particular data source to complete subsequent tutorials.
- **Tutorial C -** The lessons in this tutorial teach you about Witango builders and projects. You work with the Search Builder and New Record Builder to create Witango applications in which you insert new records, search for them, and display results. You also learn how to organize your Witango application files into projects.

- **Tutorial D -** The lessons in this tutorial teach you about personalization and saving user information. You learn how to create form fields for collecting user information. Then, you assign this information to variables so that it can be stored and referenced over multiple web pages.
- **Tutorial E -** The lessons in this tutorial show you how to make functional and stylistic enhancements to your application.
- **Tutorial F -** The lessons in this tutorial teach you how to create a secure login model for users of your web site.
- **Tutorial G -** The lessons in this tutorial show you how to create a Witango class file, which can be used in a number of different Witango application files. You then learn how to call a Witango class file's methods into your login model to search the record list.

Developer Resources

The developer resources listed below are available from the Developer Section of the Witango web site.

Witango-Talk List

You are encouraged to join the witango-talk list. Witango-Talk is a mailing list that is the ideal place to discuss your ideas with other Witango users. It is hosted by Witango Technologies but is an unmoderated list for the Witango community. It is also monitored by our technical support staff. To subscribe/unsubscribe to witango-talk list click here section describes each list To obtain a list of commands for the witango-Talk mailing list, send an e-mail message with the subject help to listserv@witango.com.

Mail List Archives

A searchable archive of the mail list that you can view as a threaded forum.

Metatags Online

A full searchable list of all the metatags in the Witango script language.

Component Zone

The component zone page is a list of sample Witango code. Why code from scratch? - use one of the components to kick start your project. The Component Zone is the place for Witango developers to publish tools, utilities, demos of your applications, documents and more.

Sample Code Modules

Complete mini applications to install and run without any programming. The sample modules are all wrapped up in a simple but highly secure shell that has group based permissioning. Sample modules include:

- Email Client
- Survey System
- Content Publishing System
- Mini CRM
- Threaded Forum with email feed

Feature Request/Report a Bug

The bug reporting system is your interface for requesting new features or reporting a bug in existing functionality.

Configuring Witango Server

Configuration variables set options in the Witango Server that control it's behaviour. You can set up all of your configuration variables using the witango administration web application. You can download the administration web application from the witango web site. You must know the correct password, which can be found in the Witango Server's `witango.ini` file. It is strongly recommended that you change the password for reasons of security, before the server is exposed to an intranet or the internet. You can change the password by stopping Witango Server and editing the value of the configuration variable parameter, `CONFIGPASSWD=` in the `witango.ini` file, located in the configuration folder which is in the folder where Witango Server is installed. This change takes place when you restart the Witango Server.

Annual Developers Conference

Every year, Witango hosts an 2 day developers conference for the developer community to gather and learn. This conference is known as CORROBOREE. Training courses at all levels are run in conjunction with this conference.

Contacting Witango Technologies

You can get in touch with Witango Technologies by mail, telephone, fax, or e-mail:

Office Address

Witango Technologies Pty Ltd
(ABN 64 061 677 503)
Level One

44 Miller Street
NORTH SYDNEY NSW 2060
Australia

Postal Address

P.O.Box 4
WILLOUGHBY NSW 2068
Australia

Telephone

PH: + 61 2 9460 0500

Fax

FX: + 61 2 9460 0502

Email

General	info@witango.com
Sale Inquiries	sales@witango.com
Support for T2K, Trial and Lite versions of Witango	support@witango.com
Support for commercial versions of Witango 5.x	customer_support@witango.com
Web	http://www.witango.com/