



**SIMPLY RICH**

**ZK™**

# **The Quick Start Guide**

**Version 3.0.0 RC**

September 2007

**Potix Corporation**

**Copyright © Potix Corporation. All rights reserved.**

The material in this document is for information only and is subject to change without notice. While reasonable efforts have been made to assure its accuracy, Potix Corporation assumes no liability resulting from errors or omissions in this document, or from the use of the information contained herein.

Potix Corporation may have patents, patent applications, copyright or other intellectual property rights covering the subject matter of this document. The furnishing of this document does not give you any license to these patents, copyrights or other intellectual property.

Potix Corporation reserves the right to make changes in the product design without reservation and without notification to its users.

The Potix logo and ZK are trademarks of Potix Corporation.

All other product names are trademarks, registered trademarks, or trade names of their respective owners.

# Table of Contents

<b>Before You Start.....</b>	<b>4</b>
New to the Servlet Container (aka., Java Web Server).....	4
New to Java Language.....	4
New to Java Integrated Development Environment (IDE).....	4
<b>1. What to Download.....</b>	<b>5</b>
<b>2. Run the Demo.....</b>	<b>6</b>
<b>3. Installation.....</b>	<b>7</b>
Alternative 1: Bundle ZK into Your Web Application.....	7
Alternative 2: Install ZK to Be Shared by All Web Applications.....	7
Working with MySQL.....	9
<b>4. The Content of Binary Distribution.....</b>	<b>10</b>
/doc.....	10
/dist/lib.....	10
/dist/lib/zkforge.....	10
/dist/lib/ext.....	10
/dist/src.....	11
/dist/xsd.....	11
/dist/WEB-INF.....	11
<b>5. The Content of Demo Distribution.....</b>	<b>12</b>
/.....	12
/zkdemo.....	12
/MyApp.....	12
<b>6. My First ZK Application.....</b>	<b>13</b>
My First Hello World.....	15

# Before You Start

---

## New to the Servlet Container (aka., Java Web Server)

Before developing Web applications in Java (and running ZK demo in your machine), you have to install a Servlet container first. Apache Tomcat is one of the most popular Servlet containers. It is easy to install and use.

Download      Installer for Windows:  
                    [apache-tomcat-5.5.20.exe](#)

                    A list of all available versions:  
                    <http://tomcat.apache.org/download-55.cgi>

Documentation    <http://tomcat.apache.org/tomcat-5.5-doc/index.html>

## New to Java Language

You don't need to know Java to use ZK, since all rich user interface can be implemented in HTML-like markup language called ZUML. However, to complete a Web application, you or teammate need some basic knowledge about Java. Here are some good tutorial.

Language Basic    <http://java.sun.com/docs/books/tutorial/java/nutsandbolts/index.html>

Class and Object   <http://java.sun.com/docs/books/tutorial/java/concepts/index.html>  
                          <http://java.sun.com/docs/books/tutorial/java/javaOO/index.html>  
                          <http://java.sun.com/docs/books/tutorial/java/IOandI/index.html>

## New to Java Integrated Development Environment (IDE)

Eclipse is one of the most popular Java IDEs. With IDE, it is easier to develop and debug your Web applications. Moreover, you can understand ZK better by debugging through ZK's source codes.

The step-by-step setup guide can be found in one of our small talks:

<http://www.zkoss.org/smalltalks/eclipse/ek.html>

# 1. What to Download

---

File	Description
zk-bin-3.0.0-RC.tar.gz zk-bin-3.0.0-RC.zip	<p>The binary distribution of ZK, including ZK libraries and the third-party libraries.</p> <p>It is all you need to enrich your Web applications with ZK.</p>
zk-demo-3.0.0-RC.zip	<p>Standalone <code>zkdemo</code> application in WAR and EAR format, and the source codes of <code>zkdemo</code>.</p> <p>It is the fastest way to test drive the features of ZK without configuring your Web server.</p>
zk-javadoc-3.0.0-RC.zip	The Java API document of ZK framework.
zk-src-3.0.0-RC.tar.gz	The source codes of ZK framework.

## 2. Run the Demo

---

The simplest way to test drive the power of ZK is to visit <http://www.zkoss.org/zkdemo/userguide>.

If you want to run the demo in your local server, you can follow the following steps.

1. Download `zk-demo-3.0.0-RC.zip` from here ([sourceforge.net](http://sourceforge.net)).
2. After uncompressing the file, you can deploy `zkdemo-all.war` or `zkdemo-all.ear` to your Web or application server.

Most Web or application servers have a management console that allows you to deploy an application painlessly. Consult the corresponding manuals, or you can visit [http://en.wikibooks.org/wiki/ZK/How-Tos#Installation\\_and\\_Configuration](http://en.wikibooks.org/wiki/ZK/How-Tos#Installation_and_Configuration), where the ZK community maintains the installation guides for many Web or application servers.

For Tomcat server, you can copy it directly to the `$TOMCAT_DIR/webapps` directory, and then Tomcat will start the deployment automatically.

3. After `zkdemo-all.war` is deployed you can visit it at, say, <http://localhost:8080/zkdemo-all/userguide>. The port number depends on how you installed your Web or application servers. Some application servers deploy only the EAR file. That is, you have to deploy `zkdemo-all.ear` instead.

## 3. Installation

---

ZK consists of several libraries (the JAR files). There are two ways to install them. First, bundle them with your Web application. In other words, copy them to the `WEB-INF/lib` directory of your Web application. The benefit is that you can deploy your Web application to any Web server without configuring the Web server.

Second, install them to your Web or application server. In other words, copy them to the shared directory of the Web or application server (for Tomcat, it is `shared/lib`). The benefit is that all Web applications can use ZK without bundled with any ZK library.

### Alternative 1: Bundle ZK into Your Web Application

By copying ZK libraries to your Web application, you can deploy your Web application to any Web or application server without configuration.

#### To bundle ZK libraries with the WAR file (the Web application)

1. Uncompress `zk-3.0.0-RC.zip` or `zk-3.0.0-RC.tar.gz`
2. Copy `dist/lib/*.jar`, `dist/zkforge/*.jar`, and `dist/lib/ext/*.jar` to the `/WEB-INF/lib` directory in your Web application.  
All libraries in `dist/zkforge` and most of libraries in `dist/lib/ext` are optional. If you want to minimize the footprint of your Web application, refer to the next chapter for the description of these libraries.

#### To bundle ZK libraries with the EAR file (the Java EE application)

1. Uncompress `zk-3.0.0-RC.zip` or `zk-3.0.0-RC.tar.gz`
2. Copy `dist/lib/*.jar`, `dist/zkforge/*.jar`, and `dist/lib/ext/*.jar` to the root directory in your Java EE application.  
All libraries in `dist/zkforge` and most of libraries in `dist/lib/ext` are optional. If you want to minimize the footprint of your Java EE application, refer to the next chapter for the description of these libraries.

### Alternative 2: Install ZK to Be Shared by All Web Applications

The way to install ZK libraries to the shared directory varies from one Web server to another.

## Install ZK on Tomcat

1. Download Tomcat from <http://tomcat.apache.org> and install it, if you haven't installed it yet.
2. Stop Tomcat.
3. Uncompress `zk-3.0.0-RC.zip` or `zk-3.0.0-RC.tar.gz`
4. Copy `dist/lib/*.jar`, `dist/zkforge/*.jar`, and `dist/lib/ext/*.jar` to `$TOMCAT_HOME1/shared/lib`  
All libraries in `dist/zkforge` and most of libraries in `dist/lib/ext` are optional. If you want to minimize the footprint of your Web server, refer to the next chapter for the description of these libraries.
5. Re-start Tomcat.

Then, you can deploy your Web application, e.g., `demo/bin/zkdemo.war`, to Tomcat. It can be done by use of the Tomcat manager, or by copying it to `$TOMCAT_HOME/webapps` directly. If you prefer copying directly, you have to stop Tomcat first. After deployed, you can run your Web application by visiting the corresponding URL, e.g., <http://localhost:8080/zkdemo/userguide>, depending on the configuration of your Web server.

## Install ZK on Jetty

1. Download Jetty from <http://www.mortbay.org/jetty/index.html> and install it<sup>2</sup>, if you haven't installed it yet.
2. Stop Jetty.
3. Uncompress `zk-3.0.0-RC.zip` or `zk-3.0.0-RC.tar.gz`
6. Copy `dist/lib/*.jar`, `dist/zkforge/*.jar`, and `dist/lib/ext/*.jar` to `$JETTY_HOME/ext`  
All libraries in `dist/zkforge` and most of libraries in `dist/lib/ext` are optional. If you want to minimize the footprint of your Web server, refer to the next chapter for the description of these libraries.
4. Re-start Jetty.

Then, you can deploy your Web application, e.g., `demo/bin/zkdemo.war`, to Jetty by copying it to `$JETTY_HOME/webapps` directly. After deployed, you can run your Web

---

<sup>1</sup> `$TOMCAT_HOME` is where you installed Tomcat.

<sup>2</sup> Refer to <http://www.mortbay.org/jetty/tut/GettingStarted.html>



application by visiting the corresponding URL, e.g.,  
`http://localhost:8080/zkdemo/userguide`, depending on the configuration of your Web server.

## **Working with MySQL<sup>3</sup>**

To open the connection under zscript, you have to put MySQL JDBC driver (`mysql-connector-*.jar`) under the `$TOMCAT_DIR/common/lib` directory.

---

<sup>3</sup> <http://www.mysql.com>

## 4. The Content of Binary Distribution

---

This chapter describes the content of zk-3.0.0-RC.zip.

### **/doc**

This directory holds the documents including copyrights and release notes.

### **/dist/lib**

This directory holds the binary libraries required to run ZK.

### **/dist/lib/zkforge**

This directory holds the components from ZK Forge, such as FCKeditor and Dojo. It is optional depending on whether you need them.

File	Description
zuljsp.jar	Required if you want to use ZUL JSP tags. In other words, it allows developers to use ZUL tags directly in JSP pages.
dojoz.jar	Required if you want to use ZK DOJO components.
fckez.jar	Required if you want to use ZK FCKeditor components.
gmapsz.jar	Required if you want to use ZK Google Maps components.
timelinez.jar json_simple.jar	Required if you want to use ZK Timeline components.

### **/dist/lib/ext**

This directory holds the external libraries required to run ZK. Since these libraries are common, you might have installed them in your container.

Here are optional jar files. You can choose whether to copy depending on your requirements.

File	Description
commons-fileupload.jar commons-io.jar	Required if you want to upload files with them. Version: Commons Fileupload 1.2 and Commons IO 1.3.1
jcommon.jar jfreechar.jar	Required if you want to use ZUL's chart component. Version: JFreeChart 1.0.5 and JCommon 1.0.9

File	Description
bsh.jar	Required if you want scripting in Java interpreter (BeanShell). Version: BeanShell 2.0b4
js.jar	Required if you want scripting in JavaScript (Rhino). Version: Rhino 1.6R5
groovy.jar	Required if you want scripting in Groovy. Version: Groovy 1.0 (groovy-all)
jruby.jar	Required if you want scripting in Ruby (JRuby). Version: JRuby 1.0 (jruby-complete)
Filters.jar	Required if you want to use the <code>captcha</code> component. Version: JH Labs Java Image Filters

## **/dist/src**

This directory holds the source codes in JAR format. These JAR files are used for debugging in IDE, such as Eclipse. You cannot build the binary libraries from these. Rather, download and uncompress `zk-src-3.0.0-RC.tar.gz`.

## **/dist/xsd**

This directory holds the XSD files that might be useful to develop ZK applications.

## **/dist/WEB-INF**

This directory holds the TLD files. These TLD files are part of JAR files so they are loaded automatically. We put them here mainly for your reference only.

## 5. The Content of Demo Distribution

---

This chapter describes the content of zk-demo-3.0.0-RC.zip.

**/**

This directory holds the executable: `zkdemo.war`, `zkdemo-all.war`, `zkdemos.ear` and `zkdemos-all.ear`. Refer to the **Installation** chapter for details.

**/zkdemo**

This directory holds the source codes of the live demo.

**/MyApp**

This directory holds an empty Web application which you can start your new Web application from.

## 6. My First ZK Application

---

### Prepare WEB-INF/web.xml

Copy or merge the following content to the `web.xml` in the `WEB-INF` directory in your application. This step must be done once each time you created a new Web applications. Then, what you need to do is to copy files with `.zul` or `.zhtml` extension to the proper directories in your Web applications.

```
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

    <!-- //// -->
    <!-- ZK -->
    <listener>
        <description>Used to cleanup when a session is
destroyed</description>
        <display-name>ZK Session Cleaner</display-name>
        <listener-
class>org.zkoss.zk.ui.http.HttpSessionListener</listener-class>
    </listener>
    <servlet>
        <description>ZK loader for evaluating ZK pages</description>
        <servlet-name>zkLoader</servlet-name>
        <servlet-class>org.zkoss.zk.ui.http.DHtmlLayoutServlet</servlet-
class>

        <!-- Must. Specifies URI of the update engine
(DHtmlUpdateServlet). -->
        <init-param>
            <param-name>update-uri</param-name>
            <param-value>/zkau</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup><!-- MUST -->
    </servlet>
    <servlet-mapping>
        <servlet-name>zkLoader</servlet-name>
        <url-pattern>*.zul</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>zkLoader</servlet-name>
        <url-pattern>*.zhtml</url-pattern>
    </servlet-mapping>
    </servlet>
```

```

        <description>The asynchronous update engine for ZK</description>
        <servlet-name>auEngine</servlet-name>
        <servlet-class>org.zkoss.zk.au.http.DHtmlUpdateServlet</servlet-
class>
    </servlet>
    <servlet-mapping>
        <servlet-name>auEngine</servlet-name>
        <url-pattern>/zkau/*</url-pattern>
    </servlet-mapping>
    <!-- //// -->

    <!-- MIME mapping -->
    <mime-mapping>
        <extension>gif</extension>
        <mime-type>image/gif</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>html</extension>
        <mime-type>text/html</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>htm</extension>
        <mime-type>text/html</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>jad</extension>
        <mime-type>text/vnd.sun.j2me.app-descriptor</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>jpeg</extension>
        <mime-type>image/jpeg</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>jpg</extension>
        <mime-type>image/jpeg</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>js</extension>
        <mime-type>application/x-javascript</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>png</extension>
        <mime-type>image/png</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>txt</extension>
        <mime-type>text/plain</mime-type>
    </mime-mapping>
    <mime-mapping>

```

```

        <extension>xml</extension>
        <mime-type>text/xml</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>zhtml</extension>
        <mime-type>text/html</mime-type>
    </mime-mapping>
    <mime-mapping>
        <extension>zul</extension>
        <mime-type>text/html</mime-type>
    </mime-mapping>

    <welcome-file-list>
        <welcome-file>index.zul</welcome-file>
        <welcome-file>index.zhtml</welcome-file>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.htm</welcome-file>
    </welcome-file-list>
</web-app>

```

## My First Hello World

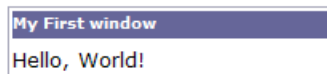
Create a file called `hello.zul` with the following content. Then, you could use the browser to see the result, say `http://localhost:8080/zkdemo/hello.zul`.

```

<window title="My First window" border="normal" width="200px">
    Hello, World!
</window>

```

Then, the result is depicted as follow.



Notice that, though the content of `hello.zul` is very similar to XUL<sup>4</sup>, it is actually written in ZUML. ZK Loader parses it into a valid HTML page which can be interpreted correctly by a regular browser, such as Internet Explorer and Mozilla Firefox. Refer to the Developer's Guide for more details.

---

<sup>4</sup> <http://xul.sourceforge.net/mozilla.html>