



**SIMPLY RICH**

**ZK™**

**クイックスタートガイド**

**Version 5.0.4**

August 2010

**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

はじめに.....	4
サーブレットコンテナ (Java Web Server)のご紹介.....	4
Java 言語のご紹介.....	4
統合開発環境 (IDE)のご紹介.....	4
1. デモファイルを実行してみよう.....	5
2. インストール.....	6
方法 1: ZK をアプリケーションに添付する.....	6
方法 2: ZK ライブラリーを共有フォルダに置く.....	6
MySQL を使う.....	7
3. ZK パッケージ.....	8
/doc.....	8
/dist/lib.....	8
/dist/lib/zkforge.....	8
/dist/lib/ext.....	9
/dist/src.....	10
/dist/xsd.....	10
/dist/WEB-INF.....	10
4. デモパッケージ.....	11
/.....	11
/zkdemo.....	11
/MyApp.....	11
5. 初めての ZK アプリ.....	12
初めての Hello World.....	14

# はじめに

## サーブレットコンテナ (Java Web Server)のご紹介

Java でウェブアプリケーションを開発するにはサーブレットコンテナをインストールしなければなりません。Apache Tomcat は最も普及しているサーブレットコンテナの一つです。Tomcat をインストールする方法はとても簡単です。

Prerequisites	Description
Download	Installer for Windows:apache-tomcat-5.5.20.exe  A list of all available versions: <a href="http://tomcat.apache.org/download-55.cgi">http://tomcat.apache.org/download-55.cgi</a>
Documentation	<a href="http://tomcat.apache.org/tomcat-5.5-doc/index.html">http://tomcat.apache.org/tomcat-5.5-doc/index.html</a>

## Java 言語のご紹介

HTML に類似する ZUML という言語を使えば、Java が分からなくても ZK でリッチなユーザーインターフェースを実現することは可能です。但し、一つのウェブアプリケーションを完成するのに、Java の基本知識は必要です。Java のチュートリアルをいくつかリストアップしました。

Java	URLs
Language Basic	<a href="http://java.sun.com/docs/books/tutorial/java/nutsandbolts/index.html">http://java.sun.com/docs/books/tutorial/java/nutsandbolts/index.html</a>
Class and Object	<a href="http://java.sun.com/docs/books/tutorial/java/concepts/index.html">http://java.sun.com/docs/books/tutorial/java/concepts/index.html</a>  <a href="http://java.sun.com/docs/books/tutorial/java/javaOO/index.html">http://java.sun.com/docs/books/tutorial/java/javaOO/index.html</a>  <a href="http://java.sun.com/docs/books/tutorial/java/1andI/index.html">http://java.sun.com/docs/books/tutorial/java/1andI/index.html</a>

## 統合開発環境 (IDE)のご紹介

Eclipse は最もよく知られている Java 統合開発環境の一つです。IDE の使用により、ウェブアプリケーションの開発・デバッグは簡単になります。さらに、ZK のソースコードをデバッグする作業を通して ZK への理解を深めることができます。

ステップ・バイ・ステップガイドは”small talks”より確認できます。

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

# 1. デモファイルを実行してみよう

---

ZK の能力を試すのであれば、下記の URL を参照しましょう。

<http://www.zkoss.org/zkdemo/userguide>

また、ローカルサーバでデモを実行する方法は次の通りです。

1. SourceForge.net から次のファイルをダウンロードします: zk-demo-5.0.4.zip
2. ファイルを解凍し、zkdemo.war か zkdemo.ear をウェブサーバまたはアプリケーションサーバにデプロイします。

ほとんどのサーバは管理用のコンソールを持ち、それを使えば簡単にアプリケーションをデプロイすることができます。手順は関連するマニュアルを参照するか、下記 ZK コミュニティの URL をご確認ください:

[http://en.wikibooks.org/wiki/ZK/How-Tos#Installation\\_and\\_Configuration](http://en.wikibooks.org/wiki/ZK/How-Tos#Installation_and_Configuration)

Tomcat サーバの場合は、アプリケーションを \$TOMCAT\_DIR/webapps フォルダにコピーすれば Tomcat は自動的にデプロイしてくれます。

3. zkdemo.war のデプロイが完了しましたら、<http://localhost:8080/zkdemo/userguide> へアクセスしてみてください。ポート番号はウェブサーバの設定によって、必ずしも 8080 ではありません。ウェブサーバーにより、EAR ファイルしかデプロイできない場合があります。その場合は、zkdemo.ear を使いましょう。

## 2. インストール

---

ZK はライブラリー集(the JAR files)を持っています。これらのライブラリーを Tomcat サーバへインストールする方法は二つあります。一つ目はこれらのファイルをウェブアプリケーションに添付(bundle)します。つまり、アプリケーションの `WEB-INF/lib` フォルダにコピーします。この方法のメリットは今後アプリケーションを他のウェブサーバにデプロイする際、サーバでの設定は不要になります。

二つ目はライブラリーを使用するウェブサーバまたはアプリケーションサーバにインストールします。つまり、共有フォルダにコピーします。(Tomcat の場合は `shared/lib`)。この方法のメリットは共有ディレクトリにアクセスできるすべてのアプリケーションは ZK を使用することが可能になります。

### 方法 1: ZK をアプリケーションに添付する

ZK のライブラリーをウェブアプリケーションの適当なフォルダにコピーすることで、アプリケーションをサーバへデプロイする際、サーバ側の設定は一切不要になります。

#### ZK ライブラリーを WAR ファイルに添付する(ウェブアプリ)

1. `zk-bin-5.0.4.zip` または `zk-bin-5.0.4.tar.gz` を解凍します。
2. `dist/lib/*.jar`, `dist/zkforge/*.jar`, `dist/lib/ext/*.jar` をウェブアプリケーションの `/WEB-INF/lib` ディレクトリーにコピーします。

`dist/zkforge` の下にある全部のライブラリーと `dist/lib/ext` の下にある一部のライブラリーはオプションライブラリーです。オプションファイルについては次の章をご覧ください。

#### ZK ライブラリーを EAR ファイルに添付する(the Java EE application)

1. `zk-bin-5.0.4.zip` か `zk-bin-5.0.4.tar.gz` を解凍します。
2. `dist/lib/*.jar`, `dist/zkforge/*.jar`, `dist/lib/ext/*.jar` を Java Ee アプリケーションのホームディレクトリーにコピーします。

`dist/zkforge` の下にある全部のライブラリーと `dist/lib/ext` の下にある一部のライブラリーはオプションです。オプションファイルについては次の章をご覧ください。

### 方法 2: ZK ライブラリーを共有フォルダに置く

この方法は ZK ライブラリーを共有フォルダに置きます。やり方はウェブサーバによって違います。

#### ZK を Tomcat にインストール

1. <http://tomcat.apache.org> から Tomcat をダウンロードし、インストールします。
2. Tomcat を停止します。

3. zk-bin-5.0.4.zip もしくは zk-bin-5.0.4.tar.gz を解凍します。
4. dist/lib/\*.jar と dist/zkforge/\*.jar と dist/lib/ext/\*.jar を  
\$TOMCAT\_HOME<sup>1</sup>/shared/lib にコピーします。  
dist/zkforge の全てのライブラリーと dist/lib/ext にある一部のライブラリーはオプションです。オプションファイルについては次の章をご覧ください。
5. Tomcat を再起動します。

上記手順に従ってライブラリーのコピーが完了したら、ウェブアプリケーション(demo/bin/zkdemo-min.war)をTomcatにデプロイしましょう。Tomcat マネジャーを利用してデプロイするか、ファイルを \$TOMCAT\_HOME/webapps にコピーします。ファイルをコピーする場合、Tomcat を一旦停止しなければなりません。デプロイが完了したら <http://localhost:8080/zkdemo/userguide> より結果を確認できます。なお、上記のパスはウェブサーバの設定により異なります。

補足：Tomcat 6.x は、デフォルトでは、shared/libにあるクラスを読み込ません。読み込ませるには conf/catalina.properties ファイルで以下を指定します：

```
shared.loader=${catalina.base}/shared/lib/*.jar
```

## ZK を Jetty にインストール

1. <http://www.mortbay.org/jetty/index.html> から Jetty をダウンロードし、インストール<sup>2</sup>します。
2. Jetty を停止します。
3. zk-bin-5.0.4.zip もしくは zk-bin-5.0.4.tar.gz を解凍します。
4. dist/lib/\*.jar と dist/zkforge/\*.jar と dist/lib/ext/\*.jar を \$JETTY\_HOME/ext にコピーします。dist/zkforge にある全てのライブラリーと dist/lib/ext にある一部のライブラリーはオプションライブラリーです。オプションファイルについては次の章をご覧ください。
5. Jetty を再起動します。

上記手順に従ってライブラリーのコピーが完了したら、ウェブアプリケーション(demo/bin/zkdemo-min.war)を \$JETTY\_HOME/webapps にコピーしましょう。デプロイ成功したら <http://localhost:8080/zkdemo/userguide> より結果を確認できます。なお、上記のパスはウェブサーバの設定により異なります。

## MySQL<sup>3</sup>を使う

zscript にてコネクションを張るには MySQL JDBC ドライバ( mysql-connector-\*.jar ) を \$TOMCAT\_DIR/common/lib に置かなければなりません。

<sup>1</sup> \$TOMCAT\_HOME は Tomcat をインストールしたホームディレクトリーです。

<sup>2</sup> <http://docs.codehaus.org/display/JETTY/Jetty+Documentation>

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

### 3. ZK パッケージ

---

この章は zk-bin-5.0.4.zip について説明します。

#### /doc

コピーライト・リリースノート等ドキュメントはこのフォルダに入っています。

#### /dist/lib

ZK を実行するのに必要なライブラリーはこのフォルダにあります。

#### /dist/lib/zkforge

FCKeditor 等 ZK Forge のコンポーネントはこのフォルダに入っています。これらのコンポーネントを使用する場合のみ、下記のファイルが必要です。

File	Description
fckez.jar	ZK FCKeditor コンポーネントを使う場合に必要 Version: 2.6.5_50
ckez.jar	ZK CKeditor コンポーネントを使う場合に必要 Version: 3.3.2_50 [コミュニティ版には入っておりません]
gmapsz.jar	ZK Google Maps コンポーネントを使う場合に必要 Version: 2.0_50 [コミュニティ版・Pro 版には入っておりません]
timelinez.jar	ZK Timeline コンポーネントを使う場合に必要 Version: 2.3.1_50 [コミュニティ版には入っておりません]
timeplotz.jar	ZK Timeplot コンポーネントを使う場合に必要 Version: 1.1_50 [コミュニティ版には入っておりません]
zuljsp.jar	ZK JSP Tags コンポーネントを使う場合に必要 Version: 1.4.1 [コミュニティ版・Pro 版には入っておりません]



## /dist/lib/ext

ZKを実行するのに必要な外部ライブラリーはこちらになります。一般的なライブラリーですので、既にインストールされている可能性があります。

オプションのjarファイルは以下です。デザインにより、必要なものだけコピーしましょう。

File	Description
commons-fileupload.jar commons-io.jar	これらのファイルでアップロード機能を実現する場合に使う  Version: Commons Fileupload 1.2.1 and Commons IO 1.3.1
jcommon.jar jfreechar.jar	ZUL のチャートコンポーネントを使用する場合に使う。  Version: JFreeChart 1.0.13 and JCommon 1.0.16  [コミュニティ版には入っておりません]
jasperreports.jar itext.jar jxl.jar poi.jar commons-collections.jar commons-logging.jar commons-digester.jar	jasperreport コンポーネントを使用する場合に使う  Version: Jasper Reports 3.7.0 (itext: 2.1.7, commons-collections: 3.2.1, commons-logging: 1.1.1, commons-digester:2.0, jxl: 2.6.12, poi: 3.2)  補足: poi.jar は Apache POI からエクセルに変換する際に使います。また、jxl.jar は JExcelApi からエクセルに変換する際に必要になります。  [コミュニティ版には入っておりません]
bsh.jar	Java interpreter (BeanShell)でスクリプトを作成する場合に使う  Version: BeanShell 2.0b4
js.jar	JavaScript (Rhino)でスクリプトを作成する場合に使う  Version: Rhino 1.7R2 (for Java 5.0)
groovy.jar	Groovy でスクリプトを作成する場合に使う  Version: Groovy 1.7 (groovy-all)
jruby.jar	Ruby (JRuby) でスクリプトを作成する場合に使う  Version: JRuby 1.1.2 (jruby-complete)  [コミュニティ版・Pro 版には入っておりません]
jython.jar	Python (Jython) でスクリプトを作成する場合に使う  Version: Jython 2.5.0  [コミュニティ版・Pro 版には入っておりません]
Filters.jar	captcha コンポーネントを使用する場合に使う  Version: JH Labs Java Image Filters  [コミュニティ版には入っておりません]
mvel.jar	MVELを使用する場合に使う

File	Description
	Version: MVEL 1.2.21 (Java1.4 用) [コミュニティ版・Pro 版には入っておりません]
ognl.jar	OGNL を使用する場合に使う Version: OGNL 2.6.9 [コミュニティ版・Pro 版には入っておりません]

## /dist/src

Eclipse 等 IDE をデバッグするに使うソースコードは JAR ファイルの形で保存されています。これらのファイルからバイナリライブラリーを編集することはできません。編集する場合は `zk-src-5.0.4.tar.gz` をダウンロードし、解凍してください。

## /dist/xsd

ZK アプリケーションを開発する際便利な XSD ファイルはこちらです。

## /dist/WEB-INF

ZK アプリケーションを開発する際便利な TLD ファイルはこのフォルダに入っています。

## 4. デモパッケージ

---

この章は zk-demo-5.0.4.zip について説明します。

/

zkdemo.war、zkdemo-min.war、zkdemos.ear、zkdemos-min.ear 計4つの実行ファイルがこのフォルダに入っています。詳細は [3.インストール](#) を参考にしてください。

### /zkdemo

ライブデモのソースコードはこちらに入っています。

### /MyApp

このディレクトリーの下でウェブアプリケーションを新規作成しましょう。

## 5. 初めての ZK アプリ

### WEB-INF/web.xml を用意しましょう

まずは以下の内容を WEB-INF の下にある web.xml にコピーしましょう。新しいアプリケーションを開発するたびに必ずこれを実行しなければなりません。次に ZUL・ZHTML ファイルを適当なディレクトリーにコピーします。

```
<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>ZK listener for session cleanup</description>
        <listener-class>org.zkoss.zk.ui.http.HttpSessionListener</listener-class>
        <!-- For Servlet 2.3, use org.zkoss.zk.ui.http.HttpSessionListener23
            instead -->
    </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>

```

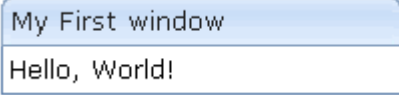
```
<welcome-file>index.zul</welcome-file>
<welcome-file>index.zhtml</welcome-file>
<welcome-file>index.html</welcome-file>
<welcome-file>index.htm</welcome-file>
</welcome-file-list>
</web-app>
```

## 初めての Hello World

以下のプログラムと同じ内容の `hello.zul` ファイルを作りましょう。そうすればブラウザ( `http://localhost:8080/zkdemo/hello.zul` )から結果が確認できます。パスはウェブサーバの設定により異なります。

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

結果は以下の通りです:

A screenshot of a web browser window. The title bar at the top says "My First window". The main content area of the window displays the text "Hello, World!".

ここで注意しなければならないのは、`hello.zul` は XUL<sup>4</sup>と類似していますが、それは XUL ではなく ZUML です。ZL ロードーは ZUML を IE 等のブラウザが読める HTML ページに変換します。詳しくは「開発者ガイド」をご覧ください。

---

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