potix

SIMPLY RICH

ZK™

クイックスタートガイド

Version 3.0.4

March 2008

Potix Corporation

Table of Contents

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: http://tomcat.apache.org/download-55.cgi
Documentation	http://tomcat.apache.org/tomcat-5.5-doc/index.html

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	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/IandI/index.html

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. ダウンロード

File	Description
zk-bin-std-3.0.4.tar.gz zk-bin-std-3.0.4.zip	Zk を使用するのに、必要なバイナリファイルと最低必要なライブラリーです。 スタンダード版です。
zk-bin-prof-3.0.4.tar.gz zk-bin-prof-3.0.4.zip	ZK のバイナリパッケージです。ZK スタンダード版にあるすべてのファイルと、JFreeChart・GoogleMaps・Jasper Reports・レイアウトコンポーネンツが含まれています。 Pro 版です。
zk-bin-3.0.4.tar.gz zk-bin-3.0.4.zip	ZK のバイナリパッケージです。ZK Pro 版にあるすべてのファイル、アクセシビリティ関係、パフォーマンス向上ファイル(zkmax.jar)、Ruby/Groovy/JavaScript/MVEL/OGNL のためのプラグインなどが入っています。 エンタープライズ版です。
Zk-demo-3.0.4.zip	独立した zkdemo アプリケーションを WAR と EAR のフォーマットで提供します。デモアプリケーションのソースコードも含まれています。 デモ用サンプルを見るにはウェブサーバ等の設定は一切不要です。
Zk-javadoc-3.0.4.zip	ZK フレームワークにて Java API を使用するマニュアルです。
Zk-src-3.0.4.tar.gz	ZK フレームワークのソースコードです。

2. 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-3.0.4.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 コミュニティーの URL をご確認ください:
   `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。

3. 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-3.0.4.zip` または `zk-bin-3.0.4.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し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-3.0.4.zip` か `zk-bin-3.0.4.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し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-3.0.4.zip もしくは zk-bin-3.0.4.tar.gz を解凍します。

4. dist/lib/*.jar と dist/zkforge/*.jar と dist/lib/ext/*.jar を $TOMCAT_HOME[1]/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し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[2]します。

2. Jetty を停止します。

3. zk-bin-3.0.4.zip もしくは zk-bin-3.0.4.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し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[3]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。

1 $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。
2 http://docs.codehaus.org/display/JETTY/Jetty+Documentation
3 http://www-jp.mysql.com/

4. ZK パッケージ

この章は zk-bin-3.0.4.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

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

File	Description
dojoz.jar	ZK DOJO コンポーネントを使う場合に必要 Version: 0.4.1_1
fckez.jar	ZK FCKeditor コンポーネントを使う場合に必要 Version: 2.5.1_1
gmapsz.jar	ZK Google Maps コンポーネントを使う場合に必要 Version: 2.0_7
timelinez.jar	ZK Timeline コンポーネントを使う場合に必要 Version: 1.2_1

/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 1.0.9 and JCommon 1.0.12

File	Description
	[スタンダード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	jasperreport コンポーネントを使用する場合に使う Version: Jasper Reports 2.0.4 (itext: 1.3.1, commons-collections: 2.1, commons-logging: 1.0.2, jxl: 2.6, poi: 3.0.1) 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。 [スタンダード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 1.6R5 [スタンダード版・Pro 版には入っておりません]
groovy.jar	Groovy でスクリプトを作成する場合に使う Version: Groovy 1.5.1 (groovy-all) [スタンダード版・Pro 版には入っておりません]
jruby.jar	Ruby (JRuby) でスクリプトを作成する場合に使う Version: JRuby 1.0.1 (jruby-complete) [スタンダード版・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 2.2.1 [スタンダード版・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 [スタンダード版には入っておりません]
mvel.jar	MVEL を使用する場合に使う Version: MVEL 1.2.21 (Java1.4 用) [スタンダード版・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 [スタンダード版・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-3.0.4.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。

5. デモパッケージ

この章は zk-demo-3.0.4.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。

6. 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>Listener for cleanup when a session is destroyed</description>
        <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>
```

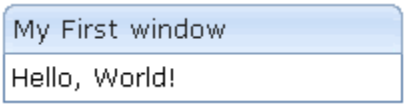
```
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.htm</welcome-file>
    </welcome-file-list>
</web-app>
```

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:



ここで注意しなければならないのは、hello.zul は XUL[4]と類似していますが、それは XUL ではなく ZUML です。ZL ローダーは 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