SCC
INFORMATIONSSYSTEME

COMMON CONTROLS

Quickstart

Adding the Common Controls to an existing project

SCC Informationssysteme GmbH

Version 1.6 · As of: 14. Januar 2006

Impressum

PUBLISHED BY

SCC Informationssysteme GmbH
Drosselweg 13
64367 Mühltal
+49 6151 13 63 10
www.scc-gmbh.com

Contents

1 **Quickstart**

1.1 Setting up the project

1.1.1 Copying the JAR files

1.1.2 Copying the resources

1.1.3 Copying the tag library descriptors (TLDs)

1.2 Declaring the tag libraries

1.3 Adding the tag libraries to the deployment descriptor

1.4 Registering the resource servlet (optional)

1.5 Registering the painter factory

1.5.1 Using an action servlet

1.5.2 Using a Struts plug-in

1.6 Structure of the JSP page

1.7 Examples

2 **Appendix: sample web.xml**

1 Quickstart

This document describes the basic steps for putting the Common Controls to use. The Common Controls support application developers in building dynamic HTML user interfaces while maintaining a strict separation between the presentation layer and the business logic.

1.1 Setting up the project

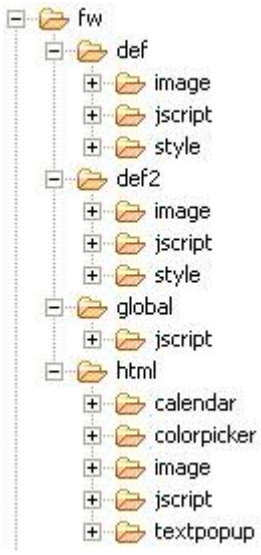
1.1.1 Copying the JAR files

To use the Common Controls, copy the following JAR files into the lib directory of the project:[1]

- antlr.jar
- common-controls-1.x.x.jar
- commons-beanutils.jar
- commons-collections.jar
- commons-digester.jar
- commons-el.jar
- commons-el-ext.jar[2]
- commons-fileupload.jar
- commons-logging.jar
- commons-validator.jar
- ecs.jar
- jakarta-oro.jar
- jakarta-regexp.jar
- log4j.jar
- struts.jar

1.1.2 Copying the resources

Rendering the user interface also requires the images and style sheets that ship with the Common Controls. To provide them, copy the fw directory into the subdirectory that serves as the root directory of the web application.



1.1.3 Copying the tag library descriptors (TLDs)

The following tag library descriptors (TLDs) also need to be copied into the project:

Common Controls:	Base elements
cc-base.tld	Control elements
cc-controls.tld	Type converters
cc-converter	Form elements
cc-forms.tld	Menu
cc-menu.tld	Permission system
cc-security.tld	SVG tags
cc-svg.tld	JSP templates
cc-template.tld	Utility
cc-utility.tld	Bean tags
Struts:	HTML tags
struts-bean	Logic tags
struts-html	Nested tags
struts-logic	Struts Tiles
struts-nested	
struts-tiles	

The following chapter assumes that the TLDs are placed in the subdirectory /WEB-INF/tlds/. If a different directory is chosen, the paths given below have to be adjusted accordingly.

Before the JSP tags can be used, the tag library descriptors have to be registered in the deployment descriptor of the application (web.xml); see chapter 1.2.

Note: as of JSP specification 1.2, the tag library descriptors packed into a JAR file can also be accessed directly. In that case neither registering nor copying them is necessary.

1.2 Declaring the tag libraries

To use the Common Controls tags on a JSP page, the corresponding tag library has to be declared at the top of the page.

With the TLDs registered in the deployment descriptor (web.xml):

```jsp
<%@ taglib uri="/WEB-INF/tlds/cc-controls.tld" prefix="ctrl" %>
```

or when using the TLDs directly from common-controls.jar:

```jsp
<%@ taglib uri="http://www.common-controls.com/cc/tags-ctrl" prefix="ctrl" %>
```

The Common Controls can then be referenced with the prefix <ctrl:tagname />. The prefix can be chosen freely. Example:

```jsp
<%@ taglib uri="/WEB-INF/tlds/cc-controls.tld" prefix="ctrl" %>
<%@ taglib uri="/WEB-INF/tlds/cc-utility.tld" prefix="util" %>

<html>
<head>
    <%-- Framework includes --%>
    <util:jsp  directive="includes"/>
</head>

<body leftmargin="0" topmargin="0" onLoad="init();">

<ctrl:tree
    name="products"
    action="sample201/producttreeBrowse"
    root="true"
    linesAtRoot="true"
    labelProperty="name"
    imageProperty="type"
    expandMode="multiple"
    groupselect="true"/>

</body>
</html>

<%-- Framework cleanup processing --%>
<util:jsp  directive="endofpage"/>
```

Depending on which functionality is used, further tag libraries may have to be declared. Which tags are available in which tag library is described in the documentation of the Common Controls tag library.

```jsp
```

```
<%@ taglib uri="/WEB-INF/tlds/cc-base.tld"        prefix="base" %>
<%@ taglib uri="/WEB-INF/tlds/cc-controls.tld"    prefix="ctrl" %>
<%@ taglib uri="/WEB-INF/tlds/cc-converter.tld"   prefix="convert" %>
<%@ taglib uri="/WEB-INF/tlds/cc-forms.tld"       prefix="forms" %>
<%@ taglib uri="/WEB-INF/tlds/cc-menu.tld"        prefix="menu" %>
<%@ taglib uri="/WEB-INF/tlds/cc-security.tld"    prefix="sec" %>
<%@ taglib uri="/WEB-INF/tlds/cc-svg.tld"         prefix="svg" %>
<%@ taglib uri="/WEB-INF/tlds/cc-template.tld"    prefix="template" %>
<%@ taglib uri="/WEB-INF/tlds/cc-utility.tld"     prefix="util" %>
```

Or:

JSP

```
<%@ taglib uri="http://www.common-controls.com/cc/tags-base"     prefix="base" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-controls" prefix="ctrl" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-converter" prefix="convert" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-forms"    prefix="forms" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-menu"     prefix="menu" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-security" prefix="sec" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-svg"      prefix="svg" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-template" prefix="template" %>
<%@ taglib uri="http://www.common-controls.com/cc/tags-utility"  prefix="util" %>
```

1.3 Adding the tag libraries to the deployment descriptor

To add the tag libraries to the WEB-INF/web.xml file, the following section can simply be copied:

XML

```
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-base.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-base.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-controls.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-controls.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-converter.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-converter.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-forms.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-forms.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-menu.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-menu.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-security.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-security.tld</taglib-location>
</taglib>
<taglib>
	<taglib-uri>/WEB-INF/tlds/cc-svg.tld</taglib-uri>
	<taglib-location>/WEB-INF/tlds/cc-svg.tld</taglib-location>
</taglib>
<taglib>
```

```
        <taglib-uri>/WEB-INF/tlds/cc-template.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-template.tld</taglib-location>
</taglib>
<taglib>
        <taglib-uri>/WEB-INF/tlds/cc-utility.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-utility.tld</taglib-location>
</taglib>
```

1.4 Registering the resource servlet (optional)

The resource servlet provides caching and grants access to the resources (GIFs, SVG graphics) registered in the ResourceManager. When an HTML page is reloaded, the cache yields a noticeable gain in performance.

The resource servlet is registered in web.xml (the deployment descriptor) as follows.

JAVA

```
<servlet>
        <servlet-name>res</servlet-name>
        <display-name>Ressource Servlet</display-name>
        <description>Provides access to cached Ressources</description>
        <servlet-class>com.cc.framework.resource.ResourceServlet</servlet-class>
</servlet>

<servlet-mapping>
        <servlet-name>res</servlet-name>
        <url-pattern>*.res</url-pattern>
</servlet-mapping>
```

1.5 Registering the painter factory

The PainterFactory can be registered inside a servlet or by means of a Struts plug-in.

1.5.1 Using an action servlet

The Common Controls provide the programmer with a basic design for the user interface that can be extended to suit individual needs.[3]

The design of the application is determined by so-called painter factories. A painter is responsible for producing one interface design. By using several painters, different layouts can be used side by side. The painter factory can be registered application-wide, for instance in the init() method of the front controller servlet.

JAVA

```
import javax.servlet.ServletException

import org.apache.struts.action.ActionServlet;

import com.cc.framework.ui.painter.PainterFactory
import com.cc.framework.ui.painter.def.DefPainterFactory;
import com.cc.framework.ui.painter.html.HtmlPainterFactory;

public class MyFrontController extends ActionServlet {

    public void init() throws ServletException {

```

```
        super.init();

        // Register the Painter Factory with the prefered GUI-Design
        // In this case we use the Default-Design provided by the DefPainter.
        // If you want to use the Def2-Painter just register
        // the Def2PainterFactory or register your own Painterfactory
        PainterFactory.registerApplicationPainters();
        PainterFactory.registerApplicationPainter (
            getServletContext (), DefPainterFactory.instance());
    }
}
```

The action servlet then still has to be registered in the web.xml file.

This completes the basic steps for putting the Common Controls to use. Working with the individual control elements is described in the corresponding separate documents, the guided tours.

1.5.2 Using a Struts plug-in

When the Struts framework is used, a plug-in can take the place of a servlet of one's own. To do so, derive a class from org.apache.struts.action.PlugIn and register the PainterFactory in its init() method.

JAVA

```java
import javax.servlet.ServletException;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.apache.struts.action.ActionServlet;
import org.apache.struts.action.PlugIn;
import org.apache.struts.config.ModuleConfig;

import com.cc.framework.ui.painter.PainterFactory;
import com.cc.framework.ui.painter.html.HtmlPainterFactory;
import com.company.framework.ui.painter.app.AppPainterFactory;

/**
 * Struts - Application PlugIn
 */
public class ApplicationPlugin implements PlugIn {

       /**
        * Commons Logging instance.
        */
       private Log log = LogFactory.getLog(this.getClass());

       /**
        * Default Constructor
        */
       public ApplicationPlugin() {
           super();
       }

       /**
        * @see org.apache.struts.action.PlugIn#destroy()
        */
       public void destroy() {
       }

       /**
        * @see org.apache.struts.action.PlugIn#init(ActionServlet, ModuleConfig)
```

```
         */
        public void init(ActionServlet actionServlet, ModuleConfig moduleConfig)
                throws ServletException {

            // Enable resource key translation for all elements
            actionServlet.getServletContext().setAttribute(
                 com.cc.framework.Globals.LOCALENAME_KEY, "true");

            // Register the Painter Factory with the prefered GUI-Design
            // In this case we use the Default-Design provided by the DefPainter.
            // If you want to use the Def2-Painter just register
            // the Def2PainterFactory or register your own Painterfactory
            PainterFactory.registerApplicationPainters();
            PainterFactory.registerApplicationPainter (
                 actionServlet.getServletContext (), DefPainterFactory.instance());
        }
}
```

The plug-in then has to be registered in struts-config.xml.

XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"
"http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">

<struts-config>

       <!-- Data Sources -->
       <data-sources></data-sources>

       <!-- Form Beans -->
       <form-beans></form-bean>

       </form-beans>

       <!-- Global Exceptions -->
       <global-exceptions></global-exceptions>

       <!-- Global Forwards -->
       <global-forwards></global-forwards>

       <!-- Action Mappings -->
       <action-mappings></action-mappings>

       <!-- Message Resources -->
       <message-resources parameter="ApplicationResources"/>

       <!-- Application PlugIn -->
       <plug-in className="ApplicationPlugin"/>

</struts-config>

```

1.6 Structure of the JSP page

When building the JSP page, the tags shown in the figure below have to be included at the beginning and at the end. This gives the framework the opportunity to carry out its own initialisation.

The first tag, <util:jsp directive="includes"/>, inserts all HTML include directives the presentation framework needs; it includes the style sheets and JavaScript files required by the painters.

The tag <util:jsp directive="endofpage"/> makes sure that all necessary clean-up work is done at the end of the JSP page — emptying the error collection, for example.

To use hover effects on buttons, the JavaScript handler init() has to be added to the HTML body element.

JSP

```jsp
<%@ taglib uri="/WEB-INF/tlds/cc-utility.tld" prefix="util" %>

<html>
<head>
	<%-- Framework includes --%>
	<util:jsp  directive="includes"/>
</head>

<body leftmargin="0" topmargin="0" onload="init();">

<!--  Content -->

</body>
</html>

<%-- Framework cleanup processing --%>
<util:jsp  directive="endofpage"/>

```

1.7 Examples

Code examples are part of the demo application available for download on our website. It also contains configuration examples for the control elements that demonstrate various ways of using them.

2 Appendix: sample web.xml

JAVA

```java
<?xml version="1.0" encoding="UTF-8"?>
<!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 id="WebApp">
    <display-name>ccsamples</display-name>
    <description>Sample Application Common-Controls</description>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>com.cc.sampleapp.servlet.FrontController</servlet-class>

        <!—Struts default modul -->
        <init-param>
            <param-name>config</param-name>
            <param-value>WEB-INF/config/struts-config.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet>
        <servlet-name>res</servlet-name>
        <display-name>Ressourcen Servlet</display-name>
        <description>Verwaltung von Ressourcen</description>
        <servlet-class>com.cc.framework.resource.ResourceServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>res</servlet-name>
        <url-pattern>*.res</url-pattern>
    </servlet-mapping>

    <session-config>
        <session-timeout>10</session-timeout>
    </session-config>

    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>

    <!— Common Controls TagLibs-->
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-base.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-base.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-controls.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-controls.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-converter.tld</taglib-uri>
```

```
        <taglib-location>/WEB-INF/tlds/cc-converter.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-forms.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-forms.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-menu.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-menu.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-security.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-security.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-svg.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-svg.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-template.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-template.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/cc-utility.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/cc-utility.tld</taglib-location>
    </taglib>

    <!— Struts TagLibs-->
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-bean.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-bean.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-html.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-html.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-logic.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-logic.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-nested.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-nested.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-template.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-template.tld</taglib-location>
    </taglib>
    <taglib>
        <taglib-uri>/WEB-INF/tlds/struts-tiles.tld</taglib-uri>
        <taglib-location>/WEB-INF/tlds/struts-tiles.tld</taglib-location>
    </taglib>
</web-app>

```

[1] The JAR files are located in the lib directory of common-controls.zip. Some of them carry the version number in the file name as well.

[2] commons-el-ext.jar is only required if the application server in use does not support the Expression Language (EL).

[3] Further ones will ship with the next versions of the Common Controls, or can be developed in-house.