

Developer Guide

Develop applications using RichFaces 4

by Sean Rogers (Red Hat)

1. Introduction	1
2. Getting started with RichFaces	3
2.1. Technical Requirements	3
2.2. Development environments	3
2.3. Setting up RichFaces	3
2.4. Creating a project with JBoss Tools	4
2.5. Creating a project with Maven	5
2.5.1. Setting up Maven	5
2.5.2. Using the RichFaces project archetype	5
2.6. Using RichFaces in existing JSF 2 projects	8
3. RichFaces overview	11
3.1. Full technical requirements	11
3.1.1. Server requirements	11
3.1.2. Client requirements	11
3.1.3. Development requirements	12
3.2. Architecture	12
3.2.1. Ajax Action Components	12
3.2.2. Ajax Containers	13
3.2.3. Ajax Output	13
3.2.4. Skins and theming	13
3.2.5. RichFaces Ajax Extensions	13
3.3. Technologies	13
3.4. Differences between JSF and RichFaces mechanisms	13
3.5. Restrictions	14
3.6. Compiling from source code	14
4. Basic concepts	17
4.1. Sending an Ajax request	17
4.2. Partial tree processing	17
4.3. Partial view updates	18
4.4. Component overview	18
5. Advanced features	19
5.1. JSF 2 integration	19
5.2. Error handling	19
5.2.1. Client-side errors	19
5.2.2. Server-side errors	19
5.3. Other functions	19
6. Skinning and theming	21
6.1. What are skins?	21
6.2. Using skins	21
6.3. Skinning overview	22
6.3.1. Skin parameter tables	22
6.3.2. ECSS files	24
6.4. Customizing skins	25
6.4.1. Creating a new skin	26

6.5. Changing skins at runtime	27
6.6. Skinning standard controls	28
6.6.1. Automatic skinning	28
6.6.2. Skinning with the rfs-ctn class	29
A. Style classes and skin parameters	31
A.1. Processing management	31
A.1.1. <a4j:log>	31
A.2. Rich inputs	33
A.2.1. <rich:autocomplete>	33
A.2.2. <rich:calendar>	34
A.2.3. <rich:fileUpload>	40
A.2.4. <rich:inplaceInput>	42
A.2.5. <rich:inplaceSelect>	44
A.2.6. <rich:inputNumberSlider>	46
A.2.7. <rich:inputNumberSpinner>	48
A.2.8. <rich:select>	49
A.3. Panels and containers	50
A.3.1. <rich:panel>	50
A.3.2. <rich:accordion>	51
A.3.3. <rich:collapsiblePanel>	52
A.3.4. <rich:popupPanel>	53
A.3.5. <rich:tabPanel>	55
A.4. Tables and grids	56
A.4.1. <rich:dataTable>	56
A.4.2. <rich:collapsibleSubTable>	59
A.4.3. <rich:collapsibleSubTableToggler>	61
A.4.4. <rich:extendedDataTable>	61
A.4.5. <rich:dataGrid>	64
A.4.6. <rich:list>	65
A.4.7. <rich:dataScroller>	66
A.5. Trees	67
A.5.1. <rich:tree>	67
A.5.2. <rich:treeNode>	68
A.6. Menus and toolbars	69
A.6.1. <rich:dropDownMenu>	69
A.6.2. <rich:panelMenu>	71
A.6.3. <rich:toolbar>	76
A.7. Output and messages	76
A.7.1. <rich:message>	76
A.7.2. <rich:messages>	77
A.7.3. <rich:progressBar>	78
A.7.4. <rich:tooltip>	78
A.8. Drag and drop	79
A.8.1. <rich:dropTarget>	79

A.8.2. <rich:dragIndicator>	79
B. Revision History	81

Introduction

The RichFaces framework is a rich component library for JavaServer Faces (JSF). The framework extends the JSF framework's Ajax capabilities with advanced features for enterprise web application development.

RichFaces leverages several parts of the JSF 2 framework including lifecycle, validation, conversion facilities, and management of static and dynamic resources. The RichFaces framework includes components with built-in Ajax support and a customizable look-and-feel that can be incorporated into JSF applications.

RichFaces provides a number of advantages for enterprise web application development:

- Create complex application views using out-of-the-box components. The RichFaces user interface (UI) library (`rich`) contains components for adding rich interactive features to JSF applications. It extends the RichFaces framework to include a large set of Ajax-enabled components that come with extensive skinning support. Additionally, the RichFaces framework is designed to be used seamlessly with other 3d-party libraries on the same page, so you have more options for developing applications.
- Write your own customized rich components with built-in Ajax support. The Component Development Kit (CDK), used for the RichFaces UI library creation, includes a code-generation facility and a templating facility using XHTML (extended hyper-text markup language) syntax.
- Generate binary resources on the fly. Extensions to JSF 2 resource-handling facilities can generate images, sounds, **Microsoft Excel** spreadsheets, and more during run-time.
- Create a modern rich user-interface with skinning technology. RichFaces provides a skinning feature that allows you to define and manage different color schemes and other parameters of the look and feel. It is possible to access the skin parameters from page code during run-time. RichFaces comes packaged with a number of skins to get you started, but you can also easily create your own customized skins too.

Getting started with RichFaces

Follow the instructions in this chapter to configure the RichFaces framework and get started with application development.

If you have existing projects that use a previous version of RichFaces, refer to the *RichFaces Migration Guide*.

2.1. Technical Requirements

The minimum technical requirements needed to get started with RichFaces are outlined below.

- Java Development Kit (JDK) 1.5 or higher
- An application server compliant with Java Platform, Enterprise Edition 5 (JEE5), such as JBoss Application Server 6 or Apache Tomcat 6.
- A compliant web browser, such as Firefox 3.5 or Internet Explorer 7

2.2. Development environments

RichFaces applications can be developed using a range of tools, including integrated development environments (IDEs). This chapter covers only two such environments in detail:

- JBoss Tools, as described in [Section 2.4, “Creating a project with JBoss Tools”](#).
- Maven, as described in [Section 2.5, “Creating a project with Maven”](#).

Other environments, such as Idea or NetBeans, could also be used for RichFaces development, but are not detailed in this book.

2.3. Setting up RichFaces

Follow the instructions in this section to set up the RichFaces framework and begin building applications.

1. Download RichFaces archive

Download RichFaces from the JBoss RichFaces Downloads area at <http://www.jboss.org/richfaces/download.html>. The binary files (available in `.bin.zip` or `.bin.tar.gz` archives) contain the following:

- compiled, ready-to-use Java Archives (JAR files) of the RichFaces libraries
- library source JAR files
- documentation, including Java documentation and JavaScript documentation

- archetypes
- example source code
- **Compiling from source**

Instead of using the pre-compiled binaries, you can compile the source files yourself. Refer to [Section 3.6, “Compiling from source code”](#) for further instructions.

2. Unzip archive

Create a new directory named `RichFaces`, then unzip the archive containing the binaries there.

2.4. Creating a project with JBoss Tools

Follow the procedure in this section to create a new RichFaces application with JBoss Tools. Ensure you are using the latest version of JBoss Tools.

1. Create a new project

Create a new project based on the JSF 2 environment using the RichFaces 4 template. In JBoss Tools, select **File** → **New** → **JSF Project** from the menu. Name the project, select **JSF 2** from the **JSF Environment** drop-down box, and click the **Finish** button to create the project.

If necessary, update the JSF 2 JAR files to the latest versions.

2. Add the RichFaces libraries to the project

Add `richfaces-core-api.jar`, `richfaces-core-impl.jar`, `richfaces-components-api.jar`, and `richfaces-components-ui.jar` into your project. Copy them from the location where you unzipped the RichFaces archive to the `WebContent/WEB-INF/lib/` directory of your project in **JBoss Tools**.

3. Reference the tag libraries

The RichFaces tag libraries need to be referenced on each XHTML page in your project:

```
<ui:composition xmlns:a4j="http://richfaces.org/a4j"
                xmlns:rich="http://richfaces.org/rich">
    ...
</ui:composition>
```

You are now ready to begin constructing your RichFaces applications. RichFaces components can be dragged and dropped into your application's XHTML pages from the RichFaces palette in JBoss Tools.

2.5. Creating a project with Maven

Apache Maven is a build automation and project management tool for Java projects. Follow the instructions in this section to create a Maven project for RichFaces.

2.5.1. Setting up Maven

Maven can be downloaded and installed from Apache's website at <http://maven.apache.org/download.html>. Due to the use of dependency importing, Maven version 2.0.9 or above is required.

Once Maven has been installed, no further configuration is required to begin building Maven projects.

2.5.2. Using the RichFaces project archetype

A Maven archetype is a template for creating projects. Maven uses an archetype to generate a directory structure and files for a particular project, as well as creating `pom.xml` files that contain build instructions.

The RichFaces Component Development Kit includes a Maven archetype named `richfaces-archetype-simpleapp` for generating the basic structure and requirements for a RichFaces application project. Maven can obtain the archetype from the JBoss repository at <https://repository.jboss.org/nexus/content/groups/public/>. The archetype is also included with the RichFaces source code in the `archetypes` directory. Follow the procedure in this section to generate a new Maven-based RichFaces project using the archetype.

1. Add required repository

The details for the JBoss repository need to be added to Maven so it can access the archetype. Add a profile in the `maven_installation_folder/conf/settings.xml` file under the `<profiles>` element:

```
<profiles>
  ...
  <profile>
    <id>jboss-public-repository</id>
    <repositories>
      <repository>
        <id>jboss-public-repository-group</id>
        <name>JBoss Public Maven Repository Group</name>
        <url>https://repository.jboss.org/nexus/content/groups/public/</url>
        <layout>default</layout>
        <releases>
          <enabled>true</enabled>
```

```
        <updatePolicy>never</updatePolicy>
      </releases>
      <snapshots>
        <enabled>true</enabled>
        <updatePolicy>never</updatePolicy>
      </snapshots>
    </repository>
  </repositories>
  <pluginRepositories>
    <pluginRepository>
      <id>jboss-public-repository-group</id>
      <name>JBoss Public Maven Repository Group</name>
      <url>https://repository.jboss.org/nexus/content/groups/
public/</url>
      <layout>default</layout>
      <releases>
        <enabled>true</enabled>
        <updatePolicy>never</updatePolicy>
      </releases>
      <snapshots>
        <enabled>true</enabled>
        <updatePolicy>never</updatePolicy>
      </snapshots>
    </pluginRepository>
  </pluginRepositories>
</profile>
</profiles>
```

The profile then needs to be activated in the `<activeProfiles>` element:

```
<activeProfiles>
  <activeProfile>jboss-public-repository</activeProfile>
</activeProfiles>
```

For further details, refer to the *JBoss RichFaces Wiki*.

2. Generate the project from the archetype

The project can now be generated with the `richfaces-archetype-simpleapp` archetype. Create a new directory for your project, then run the following Maven command in the directory:

```
mvn archetype:generate -DarchetypeGroupId=org.richfaces.archetypes
-DarchetypeArtifactId=richfaces-archetype-simpleapp -
DarchetypeVersion=4.0.0-SNAPSHOT -DgroupId=org.docs.richfaces -
DartifactId=new_project
```

The following parameters can be used to customize your project:

`-DgroupId`
Defines the package for the Managed Beans

`-DartifactId`
Defines the name of the project

The command generates a new RichFaces project with the following structure:

```
new_project
### pom.xml
### readme.txt
### src
### main
### java
#   ### org
#       ### docs
#           ### richfaces
#               ### RichBean.java
### webapp
### index.xhtml
### templates
#   ### template.xhtml
### WEB-INF
### faces-config.xml
### web.xml
```

3. Add test dependencies (optional)

Your root directory of your project contains a project descriptor file, `pom.xml`. If you wish to include modules for test-driven JSF development, add any dependencies for the tests to the `pom.xml` file. For full details on how to use the `jsf-test` project, refer to <http://community.jboss.org/wiki/TestDrivenJSFDevelopment> [<http://community.jboss.org/docs/DOC-13155>].

4. Build the project

Build the project from the command line by entering the `mvn install` command.

The `BUILD SUCCESSFUL` message indicates the project has been assembled and is ready to import into an IDE (integrated development environment), such as JBoss Tools.

5. Import the project into an IDE

To import the project into Eclipse and JBoss Tools, use the JBoss Maven Integration plug-ins. These plug-ins work with plug-ins from the M2Eclipse project to import Maven projects.

a. Install the plug-ins

- i. Choose **Help** → **Install New Software** from the Eclipse menu.
- ii. Select the JBoss Tools update site to use, then open the **Maven Support** group and select the **JBoss Maven Integration** and **JBoss Maven JSF Configurator** plug-ins.
- iii. Follow the prompts to install the integration plug-ins. The installation will automatically include the transitive dependencies **Maven Integration for Eclipse** and **Maven Integration for WTP**. Both of these dependencies are from the M2Eclipse project.
- iv. Restart Eclipse to finish the installation.

b. Open the importing wizard

With the plug-ins installed, open the importing wizard by choosing **File** → **Import** from the menu.

c. Select the project

Select **Maven** → **Existing Maven Projects** as the import source and choose the directory with the `pom.xml` file for your project.



Exporting from Maven

The ability to prepare the project for Eclipse and export it using Maven is deprecated in RichFaces 4.0. The process does not support JBoss integration-specific features, such as JSF Facets.

Your project is now ready to use. Once components and functionality have been added, you can run the application on a server and access it through a web browser at the address `http://localhost:8080/jsf-app/`.

2.6. Using RichFaces in existing JSF 2 projects

RichFaces can be added to existing JSF 2 projects by adding the new RichFaces libraries. Refer to [Step 2](#) and [Step 3](#) in [Section 2.4, “Creating a project with JBoss Tools”](#) for details.



Application-level settings

In RichFaces 4, it is not necessary to add any extra settings to the `web.xml` or `config.xml` settings files to use the framework.

RichFaces overview

Read this chapter for technical details on the RichFaces framework.

3.1. Full technical requirements

RichFaces has been developed with an open architecture to be compatible with a wide variety of environments.

3.1.1. Server requirements

RichFaces 4 requires either of the following server technologies:

- An application server compliant with Java Platform, Enterprise Edition 5 or 6 (JEE5 or JEE6), such as JBoss Application Server 6.
- A major servlet container, such as Jetty 6 or Apache Tomcat 6.

3.1.2. Client requirements

Clients accessing RichFaces applications require a web browser. Richfaces supports the following web browsers:

Linux environments

- Firefox 3.0 and higher
- Google Chrome 1.0 and higher
- Opera 9.5 and higher

Mac OS environments

- Firefox 3.5 and higher
- Google Chrome 1.0 and higher
- Safari 3.0 and higher

Microsoft Windows environments

- Firefox 3.0 and higher
- Google Chrome 1.0 and higher
- Internet Explorer 7.0 and higher
- Opera 9.5 and higher

- Safari 4.0 and higher

3.1.3. Development requirements

Developing applications with the RichFaces framework requires the Java Development Kit (JDK), an implementation of JavaServer Faces (JSF), and a development environment.

Java Development Kit (JDK)

RichFaces supports the following JDK versions:

- JDK 1.5 and higher

JavaServer Faces (JSF)

RichFaces supports the following JSF implementations and frameworks:

- MyFaces 2 and higher
- Seam 3 and higher
- Mojarra 2 and higher

Development environment

RichFaces can be developed using most Java development environments. The following are recommended, and used for examples in this guide:

- JBoss Tools 3.1 and higher
- Maven 2.0.9 and higher (2.2.1 recommended)

3.2. Architecture

The important elements of the RichFaces framework are as follows:

- Ajax Action Components
- Ajax Containers
- Ajax Output
- Skins and Theming
- RichFaces Ajax Extensions

Read this section for details on each element.

3.2.1. Ajax Action Components

The RichFaces framework includes several Ajax Action Components and Submitting Behaviors: `<a4j:commandButton>`, `<a4j:commandLink>`, `<a4j:poll>`, `<a4j:ajax>`, and more. Use Ajax Action Components to send Ajax requests from the client side.

3.2.2. Ajax Containers

`AjaxContainer` is an interface that marks part of the JSF tree that is decoded during an Ajax request. It only marks the JSF tree if the component or behavior sending the request does not explicitly specify an alternative. `AjaxRegion` is an implementation of this interface.

3.2.3. Ajax Output

`AjaxContainer` is an interface that marks part of the JSF tree that will be updated and rendered on the client for every Ajax request. It only marks the JSF tree if the component or behavior sending the request does not explicitly turn off automatic updates.

3.2.4. Skins and theming

RichFaces includes extensive support for application skinning. Skinning is a high-level extension to traditional CSS (Cascading Style Sheets) which allows the color scheme and appearance of an application to be easily managed. The skins simplify look-and-feel design by allowing multiple elements of the interface to be handled as manageable features, which have associated color palettes and styling. Application skins can additionally be changed on the fly during run-time, allowing user experiences to be personalized and customized.

For full details on skinning and how to create skins for the components in your application, refer to [Chapter 6, Skinning and theming](#).

3.2.5. RichFaces Ajax Extensions

The RichFaces Ajax Extensions plug in to the standard JSF 2 Ajax script facility. They extend the script facility with new features and options.

3.3. Technologies

RichFaces 4 features full JSF 2 integration and uses standard web application technologies such as JavaScript, XML (Extensible Markup Language), and XHTML (Extensible Hypertext Markup Language).

3.4. Differences between JSF and RichFaces mechanisms

JavaServer Faces 2 evaluates Ajax options, such as `execute` and `render`, while rendering a page. This allows any parameters to be sent directly from the client side.

RichFaces evaluates the options when the current request is sent. This increases both the security of the data and the convenience for evaluating parameters.

For example, binding Ajax options to Java Bean properties in RichFaces allows you to evaluate the options dynamically for the current request, such as defining additional zones to render.

Parameters changed manually on the client side will not influence the request processing. With JSF 2, the options have evaluated during the previous page rendering would need to be used.

3.5. Restrictions

The following restrictions apply to applications implementing the RichFaces framework:

- As with most Ajax frameworks, you should not attempt to append or delete elements on a page using RichFaces Ajax, but should instead replace them. As such, elements that are rendered conditionally should not be targeted in the `render` attributes for Ajax controls. For successful updates, an element with the same identifier as in the response must exist on the page. If it is necessary to append code to a page, include a placeholder for it (an empty element).
- JSF 2 does not allow resources such as JavaScript or Cascading Style Sheets (CSS) to be added if the element requiring the resource is not initially present in the JSF tree. As such, components added to the tree via Ajax must have any required resources already loaded. In RichFaces, any components added to the JSF tree should have components with corresponding resources included on the main page initially. To facilitate this, components can use the `rendered="false"` setting to not be rendered on the page.
- Switching RichFaces skins via Ajax during runtime should be avoided, as this requires all the stylesheets to be reloaded.

3.6. Compiling from source code

The source code for the RichFaces framework can be compiled manually rather than using the pre-compiled binaries.

1. Acquiring the source code

- **JBoss RichFaces Downloads area**

Download the RichFaces distribution as described in [Section 2.3, “Setting up RichFaces”](#). The distribution bundles contain the full RichFaces source code.

- **Anonymous SVN repository**

Alternatively, the source files can be checked out from the anonymous SVN repository at <http://anonsvn.jboss.org/repos/richfaces/branches/4.0.X/> using the following command:

```
svn co http://anonsvn.jboss.org/repos/richfaces/branches/4.0.X/
```

2. Unzip archive

If using the downloaded distribution, create a new directory named `RichFaces`, then unzip the archive containing the source code there.

3. Configure Maven for RichFaces

To compile the RichFaces source code, Maven requires access to the JBoss repositories. Refer to [Step 1](#) of the procedure in [Section 2.5.2, “Using the RichFaces project archetype”](#) to configure the required repositories.

4. Compile using Maven

In the root directory of the unzipped source code, enter the following command to compile and build the RichFaces framework:

```
mvn clean install
```

Add any of the following options after the command to customize the build:

`-P release,docs`

Maven additionally builds the documentation and release artifacts.

`-D skipTests=true`

Maven skips the unit and functional tests, which completes the build faster.

`-D checkstyle.skip=true`

Maven skips the scans for checking style, which completes the build faster.

`-D skip-source`

Maven does not produce any source `jar` files from the build.

`-D skip-enforce`

Maven does not check for the correct JDK version, Maven version, and `SNAPSHOT` plugin, and forces the build.

`-D jsf-profile=implementation`

Maven tests builds and tests against the specified JSF implementation. The *implementation* value can be any of the following:

- `jsf_ri`: The latest release of JSF RI (Mojarra). This is the default implementation.
- `jsf_ri_javax`: The latest release of `javax.faces`.
- `jsf_ri_snapshot`: The latest snapshot version of JSF RI (Mojarra).
- `myfaces`: The latest release of MyFaces.
- `myfaces`: The latest snapshot version of MyFaces.

5. Access compiled binaries

The compiled libraries are located in the following sub-directories of the root RichFaces directory:

- `/ui/dist/richfaces-components-api/target/richfaces-components-api-version-SNAPSHOT.jar`
- `/ui/dist/richfaces-components-ui/target/richfaces-components-ui-version-SNAPSHOT.jar`
- `/core/api/target/richfaces-core-api-version-SNAPSHOT.jar`
- `/core/impl/target/richfaces-core-impl-version-SNAPSHOT.jar`

You can now add the compiled libraries to your project as described in [Step 2](#) of [Section 2.4](#), “*Creating a project with JBoss Tools*”.

Basic concepts

Read this chapter for the basic concepts of using RichFaces in conjunction with Ajax and JavaServer Faces.

4.1. Sending an Ajax request

Many of the tags in the `a4j` and `rich` tag libraries are capable of sending Ajax requests from a JavaServer Faces (JSF) page.

- The `<a4j:commandButton>` and `<a4j:commandLink>` tags are used to send an Ajax request on the `click` JavaScript event.
- The `<a4j:poll>` tag is used to send an Ajax request periodically using a timer.
- The `<a4j:ajax>` tag allows you to add Ajax functionality to standard JSF components and send Ajax request on a chosen JavaScript event, such as `keyup` or `mouseover`, for example.
- Most components in the `rich` tag library have built-in Ajax support. Refer to the *RichFaces Component Reference* for details on the use of each component.

4.2. Partial tree processing

Use the `execute` attribute to specify which parts of the JSF tree to process during an Ajax request. The `execute` attribute can point to an `id` identifier of a specific component to process. Components can also be identified through the use of Expression Language (EL).

Alternatively, the `execute` attribute accepts the following keywords:

`@all`

Every component is processed.

`@none`

No components are processed.

`@this`

The requesting component with the `execute` attribute is processed.

`@form`

The form that contains the requesting component is processed.

`@region`

The region that contains the requesting component is processed. Use the `<a4j:region>` component as a wrapper element to specify regions.

Some components make use of additional keywords. These are detailed under the relevant component entry in the *RichFaces Component Reference*.

4.3. Partial view updates

Use the `render` attribute to specify which components to render for an Ajax update. The `render` attribute can point to an `id` identifier of a specific component to update. Components can also be identified through the use of Expression Language (EL).

Alternatively, the `render` attribute accepts the following keywords:

`@all`

Every component is updated.

`@none`

No components are updated.

`@this`

The requesting component with the `execute` attribute is updated.

`@form`

The form that contains the requesting component is updated.

`@region`

The region that contains the requesting component is updated. Use the `<a4j:region>` component as a wrapper element to specify regions.

Some components make use of additional keywords. These are detailed under the relevant component entry in the *RichFaces Component Reference*.

Use the `<a4j:outputPanel>` component with the `ajaxRendered="true"` setting to always update a section irrespective of the requesting component's `render` attribute. The `<rich:message>` and `<rich:messages>` components are based on the `<a4j:outputPanel>` component, and as such will also always be updated. To override this behavior, use the `limitRender="true"` setting on the requesting component.

4.4. Component overview

The RichFaces framework is made up of two tag libraries: the `a4j` library and the `rich` library. The `a4j` tag library represents *Ajax4jsf*, which provides page-level Ajax support with core Ajax components. This allows developers to make use of custom Ajax behavior with existing components. The `rich` tag library provides Ajax support at the component level instead, and includes ready-made, self-contained components. These components don't require additional configuration in order to send requests or update.

For details on the use of the various components, refer to *RichFaces Component Reference*.

Advanced features

Read this chapter for details on some of the advanced features and configuration possibilities for the RichFaces framework.

5.1. JSF 2 integration

JavaServer Faces (JSF) is the Java-based web application framework upon which the RichFaces framework has been built. RichFaces is now integrated with JSF 2, which features several improvements to the framework.

- The standard display technology used by JSF 1 was JavaServer Pages (JSP). With JSF 2, the standard display technology has been changed to Facelets, which is a more powerful and more efficient View Declaration Language (VLD) than JSP.

5.2. Error handling

RichFaces allows standard handlers to be defined for processing different application exceptions. Custom JavaScript can be executed when these exceptions occur.

5.2.1. Client-side errors

JSF provides a global `onError` handler on the client. The handler provides the relevant error code and other associated data. The RichFaces Ajax components provide the `error` attribute if extra functionality needs to be defined.

Additional processing is available through a number of components, such as the following:

- The `<a4j:status>` component has an additional error state.
- The `<a4j:queue>` component can be used to process errors.

5.2.2. Server-side errors

Use the JSF 2 `ExceptionHandler` class to handle server-side errors such as session expiration.

5.3. Other functions

RichFaces provides a number of advanced functions, such as managing user roles and identifying elements. Refer to the *Functions* chapter in the *RichFaces Component Reference* for further details.

Skinning and theming

Read this chapter for a guide to skinning and theming RichFaces applications, including how to implement themes, and details on customizing and extending skins.

6.1. What are skins?

Application skins are used with the RichFaces framework to change the appearance of an application through setting the colors and decoration of controls and components. Typically the appearance of web applications is handled through the CSS (Cascading Style Sheet) files associated with the application, but skinning allows the settings in a CSS file to be abstracted and easily edited. Skins consist of a small, generalized set of font and color parameters that can be applied to multiple different styles. This avoids repetitive coding and duplication in CSS files. CSS files are not completely replaced: skins work as a high-level extension to standard CSS.

Each skin has a set of `skin-parameters`, which are used to define the theme palette and other elements of the user interface. These parameters work together with regular CSS declarations, and can be referred to from within CSS using JavaServer Faces Expression Language (EL).

The skinning feature of RichFaces also allows skins to be changed at runtime, so users can personalize an application's appearance on the fly.

6.2. Using skins

RichFaces includes a number of predefined skins. These skins can be used in RichFaces web applications by specifying the skin name in the `org.richfaces.skin` context parameter in the `web.xml` settings file. The predefined skins are as follows:

- `DEFAULT`
- `plain`, which contains no skin parameters and is intended for embedding RichFaces components into existing projects with their own styles.
- `emeraldTown`
- `blueSky`
- `wine`
- `japanCherry`
- `ruby`
- `classic`
- `deepMarine`

To add one of these skins to your application, add the `org.richfaces.SKIN` context parameter to the `web.xml` configuration file:

```
<context-param>
  <param-name>org.richfaces.skin</param-name>
  <param-value>skin_name</param-value>
</context-param>
```

6.3. Skinning overview

RichFaces skins are implemented using the following three-level scheme:

Component stylesheets

Stylesheets are provided for each component. CSS style parameters map to skin parameters defined in the skin property file. This mapping is accomplished through the use of ECSS files. Refer to [Section 6.3.2, “ECSS files”](#) for details on ECSS files.

Skin property files

Skin property files map skin parameters to constant styles. Skin properties are defined in `skin.properties` files. Refer to [Section 6.3.1, “Skin parameter tables”](#) for a listing of the skin parameters used in a typical skin.

Custom style classes

Individual components can use the `styleClass` attribute to redefine specific elements. These components then use the styles defined in a CSS file instead of the standard look for components as defined by the ECSS stylesheets.

6.3.1. Skin parameter tables

[Table 6.1, “Parameter settings for the blueSky skin”](#) lists the default values for the parameter settings in the `blueSky` skin. These values are all listed in the `blueSky.skin.properties` file, which can be customized and extended as described in [Section 6.4, “Customizing skins”](#).

Table 6.1. Parameter settings for the `blueSky` skin

Parameter name	Default value
<code>headerBackgroundColor</code>	<code>#BED6F8</code>
<code>headerGradientColor</code>	<code>#F2F7FF</code>
<code>headTextColor</code>	<code>#000000</code>
<code>headerWeightFont</code>	<code>bold</code>
<code>generalBackgroundColor</code>	<code>#FFFFFF</code>
<code>generalTextColor</code>	<code>#000000</code>
<code>generalSizeFont</code>	<code>11px</code>
<code>generalFamilyFont</code>	<code>Arial, Verdana, sans-serif</code>
<code>controlTextColor</code>	<code>#000000</code>
<code>controlBackgroundColor</code>	<code>#FFFFFF</code>

Parameter name	Default value
<i>additionalBackgroundColor</i>	#ECF4FE
<i>shadowBackgroundColor</i>	#000000
<i>shadowOpacity</i>	1
<i>panelBorderColor</i>	#BED6F8
<i>subBorderColor</i>	#FFFFFF
<i>calendarWeekBackgroundColor</i>	#F5F5F5
<i>calendarHolidaysBackgroundColor</i>	#FFEEDA
<i>calendarHolidaysTextColor</i>	#FF7800
<i>calendarCurrentBackgroundColor</i>	#FF7800
<i>calendarCurrentTextColor</i>	#FFEEDA
<i>calendarSpecBackgroundColor</i>	#E4F5E2
<i>calendarSpecTextColor</i>	#000000
<i>editorBackgroundColor</i>	#F1F1F1
<i>editBackgroundColor</i>	#FEFFDA
<i>errorColor</i>	#FF0000
<i>gradientType</i>	plain
<i>tabBackgroundColor</i>	#C6DEFF
<i>tabDisabledTextColor</i>	#8DB7F3
<i>tableHeaderBackgroundColor</i>	#D6E6FB
<i>tableSubHeaderBackgroundColor</i>	#ECF4FE
<i>tableBorderWidth</i>	1px
<i>tableHeaderTextColor</i>	#0B356C
<i>trimColor</i>	#D6E6FB
<i>tipBackgroundColor</i>	#FAE6B0
<i>tipBorderColor</i>	#E5973E
<i>selectControlColor</i>	#E79A00
<i>generalLinkColor</i>	#0078D0
<i>hoverLinkColor</i>	#0090FF
<i>visitedLinkColor</i>	#0090FF
<i>headerSizeFont</i>	11px
<i>headerFamilyFont</i>	Arial, Verdana, sans-serif
<i>tabSizeFont</i>	11px
<i>tabFamilyFont</i>	Arial, Verdana, sans-serif
<i>buttonSizeFont</i>	11px

Parameter name	Default value
<i>buttonFamilyFont</i>	Arial, Verdana, sans-serif
<i>tableBackgroundColor</i>	#FFFFFF
<i>tableFooterBackgroundColor</i>	#CCCCCC
<i>tableSubfooterBackgroundColor</i>	#F1F1F1
<i>tableBorderColor</i>	#C0C0C0
<i>warningColor</i>	#FF6600
<i>warningBackgroundColor</i>	#FF0000

6.3.2. ECSS files

RichFaces uses ECSS files to add extra functionality to the skinning process. ECSS files are CSS files which use Expression Language (EL) to connect styles with skin properties.

Example 6.1. ECSS style mappings

The ECSS code for the `<rich:panel>` component contains styles for the panel and its body:

```
.rf-p{
    background-color: '#{richSkin.generalBackgroundColor}';
    color: '#{richSkin.panelBorderColor}';
    border-width: 1px;
    border-style: solid;
    padding: 1px;
}

.rf-p-b{
    font-size: '#{richSkin.generalSizeFont}';
    color: '#{richSkin.generalTextColor}';
    font-family: '#{richSkin.generalFamilyFont}';
    padding: 10px;
}
```

`.rf-p` defines the panel styles

- The `background-color` CSS property maps to the `generalBackgroundColor` skin parameter.
- The `color` CSS property maps to the `panelBorderColor` skin parameter.

`.rf-p-b` defines the panel body styles

- The `font-family` CSS property maps to the `generalFamilyFont` skin parameter.
- The `font-size` CSS property maps to the `generalSizeFont` skin parameter.

- The color CSS property maps to the *generalTextColor* skin parameter.

6.4. Customizing skins

Skins in RichFaces can be customized on each of the three levels:

Skin property files

Application interfaces can be modified by altering the values of skin parameters in the skin itself. Edit the constant values defined in the `skin.properties` file to change the style of every component mapped to that skin property.

Component stylesheets

Mappings and other style attributes listed in a component's ECSS file can be edited. Edit the ECSS file to change the styles of all components of that type.

Custom style classes

Individual components can use the `styleClass` attribute to use a unique style class. Add the new style class to the application CSS and reference it from an individual component with the `styleClass` attribute.

Example 6.2. Simple skinning example

Using any component, such as a panel, without specifying a `styleClass` will use the default skin parameters for that component.

```
<rich:panel>This is a panel without a header</rich:panel>
```

When rendered for display, the panel consists of two HTML elements: a wrapper `<div>` element and a `<div>` element for the body of the panel. The wrapper element for a panel without a specified `styleClass` is rendered as follows:

```
<div id="..." class="rf-p">
  <div id="..." class="rf-p-b">
    This is a panel without a header
  </div>
</div>
```

To customize the panel appearance according to the three-level scheme, adjust the styles according to the following approach:

1. Change the definitions for the *generalBackgroundColor* or *panelBorderColor* parameters in the skin. This will cause all panels in the application to change to the new settings.

2. Redefine the `rf-p` class in the application CSS. This will also cause all panels in the application to change to the new settings, though the skin itself has not been altered. Any properties not mapped to skin parameters should be redefined in this way.
3. Specify a different `styleClass` attribute to style the individual component. If a `styleClass` attribute is used, the specified style class is applied to the component, which could extend or override the default styles.

```
<rich:panel styleClass="customClass">...</rich:panel>
```

The `customClass` style is added to the CSS, and is applied to the component when it is rendered for display:

```
<div class="rf-p customClass">
    ...
</div>
```

6.4.1. Creating a new skin

1. Create the skin file

The name of the skin file should follow the format `new_skin_name.skin.properties` and is placed in either the `META-INF/skins/` directory or the classpath directory of your project.

2. Define the skin constants

- **Define all the skin constants**

Add skin parameter constants and values to the file. All the skin parameters listed in [Table 6.1, “Parameter settings for the blueSky skin”](#) should be included in the skin file, with settings relevant to your new skin.

Example 6.3. `blueSky.skin.properties` file

Open the `blueSky.skin.properties` file from the `/META-INF/skins` directory in the `richfaces-impl-version.jar` package. The file lists all the skin parameter constants shown in [Table 6.1, “Parameter settings for the blueSky skin”](#).

You can use the `blueSky.skin.properties` file as a template for your new skin.

- **Extend a base skin**

Instead of redefining an entire new skin, your skin can use an existing skin as a base on which to build new parameters. Specify a base skin by using the `baseSkin` parameter in the skin file, as shown in [Example 6.4, “Using a base skin”](#).

Example 6.4. Using a base skin

This example takes the `blueSky` skin as a base and only changes the `generalSizeFont` parameter.

```
baseSkin=blueSky
generalSizeFont=12pt
```

3. Reference the skin definition

Add a skin definition `<context-param>` to the `web.xml` settings file of your application:

```
<context-param>
  <param-name>org.richfaces.skin</param-name>
  <param-value>new_skin_name</param-value>
</context-param>
```

6.5. Changing skins at runtime

To allow users to change skins at runtime, use a managed bean to access the skin.

1. Create the skin bean

The skin bean is a simple interface to manage the skin:

```
public class SkinBean {

    private String skin;

    public String getSkin() {
        return skin;
    }

    public void setSkin(String skin) {
        this.skin = skin;
    }

}
```

2. Reference the skin bean

Add the `@ManagedBean` and `@SessionScoped` references to the class.

- Alternatively, use EL (Expression Language) to reference the skin bean from the `web.xml` settings file.

```
<context-param>
  <param-name>org.richfaces.skin</param-name>
  <param-value>#{skinBean.skin}</param-value>
</context-param>
```

3. Set initial skin

The application needs an initial skin to display before the user chooses an alternative skin. Specify the skin in your class with `@ManagedProperty`.

```
@ManagedProperty(value="blueSky")
private String skin;
```

- Alternatively, specify the initial skin in the `web.xml` configuration file.

```
<managed-bean>
  <managed-bean-name>skinBean</managed-bean-name>
  <managed-bean-class>SkinBean</managed-bean-class>
  <managed-bean-scope>session</managed-bean-scope>
  <managed-property>
    <property-name>skin</property-name>
    <value>blueSky</value>
  </managed-property>
</managed-bean>
```

6.6. Skinning standard controls

Standard HTML controls used alongside RichFaces components are also themed to create a cohesive user interface.

6.6.1. Automatic skinning

The skinning style properties are automatically applied to controls based on their element names and attribute types. If the HTML elements are referenced in the standard skin stylesheets, the controls will be styled according to the mapped skin properties.

Standard HTML controls are skinned in this way by default. To override this behavior, set the `org.richfaces.enableControlSkinning` context parameter in the `web.xml` configuration file to `false`:

```
<context-param>
  <param-name>org.richfaces.enableControlSkinning</param-name>
```

```
<param-value>false</param-value>
</context-param>
```

6.6.2. Skinning with the `rfs-ctn` class

The skinning style properties can be determined through a separate CSS class. This method is not available by default, but is enabled through the `org.richfaces.enableControlSkinningClasses` context parameter in the `web.xml` configuration file:

```
<context-param>
  <param-name>org.richfaces.enableControlSkinningClasses</param-name>
  <param-value>true</param-value>
</context-param>
```

When enabled, a stylesheet with predefined classes offers a special CSS class named `rfs-ctn`. Reference the `rfs-ctn` class from any container element (such as a `<div>` element) to skin all the standard HTML controls in the container.

Standard HTML controls can also be specifically defined in the CSS. Refer to the `/core/impl/src/main/resources/META-INF/resources/skinning_both.ecss` file in the `richfaces-ui.jar` package for examples of specially-defined CSS classes with skin parameters for HTML controls.

Appendix A. Style classes and skin parameters

Each of the RichFaces components are listed below, along with their style classes and skin parameters. For further details on each component, refer to the relevant section in the *RichFaces Component Reference*.

A.1. Processing management

A.1.1. <a4j:log>

Table A.1. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-log</code> This class defines styles for the log.	<code>generalTextColor</code>	color
<code>.rf-log-popup</code> This class defines styles for the log when it appears as a pop-up.	No skin parameters.	
<code>.rf-log-popup-cnt</code> This class defines styles for the content of the log pop-up.	No skin parameters.	
<code>.rf-log-inline</code> This class defines styles for the log when it appears in-line.	No skin parameters.	
<code>.rf-log-contents</code> This class defines styles for the log contents.	No skin parameters.	
<code>.rf-log-entry-lbl</code> This class defines styles for a label in the log.	No skin parameters.	
<code>.rf-log-entry-lbl-debug</code> This class defines styles for the debug label in the log.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-log-entry-lbl-info</code> This class defines styles for the information label in the log.	No skin parameters.	
<code>.rf-log-entry-lbl-warn</code> This class defines styles for the warning label in the log.	No skin parameters.	
<code>.rf-log-entry-lbl-error</code> This class defines styles for the error label in the log.	No skin parameters.	
<code>.rf-log-entry-msg</code> This class defines styles for a message in the log.	No skin parameters.	
<code>.rf-log-entry-msg-debug</code> This class defines styles for the debug message in the log.	No skin parameters.	
<code>.rf-log-entry-msg-info</code> This class defines styles for the information message in the log.	No skin parameters.	
<code>.rf-log-entry-msg-warn</code> This class defines styles for the warning message in the log.	No skin parameters.	
<code>.rf-log-entry-msg-error</code> This class defines styles for the error message in the log.	No skin parameters.	
<code>.rf-log-entry-msg-xml</code> This class defines styles for an XML message in the log.	No skin parameters.	

A.2. Rich inputs

A.2.1. <rich:autocomplete>

Table A.2. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-au-fnt This class defines styles for the auto-complete box font.	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-au-inp This class defines styles for the auto-complete input box.	<i>controlBackgroundColor</i>	background-color
.rf-au-fld This class defines styles for the auto-complete field.	<i>panelBorderColor</i>	border-color
	<i>controlBackgroundColor</i>	background-color
.rf-au-fld-btn This class defines styles for a button in the auto-complete field.	No skin parameters.	
.rf-au-btn This class defines styles for the auto-complete box button.	<i>headerBackgroundColor</i>	background-color
	<i>panelBorderColor</i>	border-left-color
.rf-au-btn-arrow This class defines styles for the button arrow.	No skin parameters.	
.rf-au-btn-arrow-dis This class defines styles for the button arrow when it is disabled.	No skin parameters.	
.rf-au-lst-scr1 This class defines styles for the scrollbar in the auto-complete list.	No skin parameters.	
.rf-au-itm This class defines styles for an item in the auto-complete list.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-au-itm-sel</code> This class defines styles for a selected item in the auto-complete list.	<code>headerBackgroundColor</code>	background-color
	<code>generalTextColor</code>	border-color
<code>.rf-au-shdw</code> This class defines styles for the auto-complete box shadow.	No skin parameters.	
<code>.rf-au-shdw-t</code> , <code>.rf-au-shdw-l</code> , <code>.rf-au-shdw-r</code> , <code>.rf-au-shdw-b</code> These classes define styles for the top, left, right, and bottom part of the auto-complete box shadow.	No skin parameters.	
<code>.rf-au-tbl</code> This class defines styles for a table in the auto-complete box.	No skin parameters.	

A.2.2. <rich:calendar>

Table A.3. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-cal-extr</code> This class defines the styles for a pop-up calendar exterior.	<code>panelBorderColor</code>	border-color
<code>.rf-cal-btn</code> This class defines styles for a calendar button.	No skin parameters.	
<code>.rf-cal-hdr</code> This class defines the styles for a calendar header.	<code>panelBorderColor</code>	border-bottom-color
	<code>additionalBackgroundColor</code>	background-color
	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
<code>.rf-cal-hdr-optnl</code> This class defines the styles for an optional header.	<code>panelBorderColor</code>	border-bottom-color
	<code>additionalBackgroundColor</code>	background-color
	<code>generalSizeFont</code>	font-size

Class (selector)	Skin Parameters	Mapped CSS properties
	<i>generalFamilyFont</i>	font-family
.rf-cal-hdr-month This class defines the styles for the month header.	<i>headerBackgroundColor</i>	background-color
	<i>headerSizeFont</i>	font-size
	<i>headerFamilyFont</i>	font-family
	<i>headerWeightFont</i>	font-weight
	<i>headerTextColor</i>	color
.rf-cal-ftr This class defines the styles for a calendar footer.	<i>panelBorderColor</i>	border-right-color, border-bottom-color
	<i>additionalBackgroundColor</i>	background
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
.rf-cal-ftr-optnl This class defines the styles for an optional footer.	<i>panelBorderColor</i>	border-right-color, border-bottom-color
	<i>additionalBackgroundColor</i>	background
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
.rf-cal-tl This class defines the styles for calendar toolbars.	<i>headerBackgroundColor</i>	background-color
	<i>headerSizeFont</i>	font-size
	<i>headerFamilyFont</i>	font-family
	<i>headerWeightFont</i>	font-weight
	<i>headerTextColor</i>	color
.rf-cal-tl-ftr This class defines the styles for a toolbar item in the calendar footer.	<i>additionalBackgroundColor</i>	background
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
.rf-cal-tl-btn This class defines styles for a toolbar button.	No skin parameters.	
.rf-cal-tl-btn-dis This class defines styles for a disabled toolbar button.	No skin parameters.	
.rf-cal-tl-btn-hov This class defines the styles for toolbar items when it is hovered over with the mouse cursor.	<i>calendarWeekBackgroundColor</i>	background-color
	<i>generalTextColor</i>	color
	<i>tableBackgroundColor</i>	border-color

Class (selector)	Skin Parameters	Mapped CSS properties
	<i>panelBorderColor</i>	border-right-color, border-bottom-color
<code>.rf-cal-tl-btn-press</code> This class defines the styles for toolbar items when it is pressed.	<i>panelBorderColor</i>	border-color
	<i>panelBorderColor</i>	border-right-color, border-bottom-color
<code>.rf-cal-tl-close</code> This class defines styles for a Close button in a toolbar.	No skin parameters.	
<code>.rf-cal-c</code> This class defines the styles for regular calendar cells.	<i>panelBorderColor</i>	border-bottom-color, border-right-color
	<i>tableBackgroundColor</i>	background-color
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
<code>.rf-cal-c-cnt</code> This class defines styles for the content of a cell.	No skin parameters.	
<code>.rf-cal-today</code> This class defines the styles for the cell representing today's date.	<i>calendarCurrentBackgroundColor</i>	background-color
	<i>calendarCurrentTextColor</i>	color
<code>.rf-cal-sel</code> This class defines the styles for the selected day.	<i>headerBackgroundColor</i>	background-color
	<i>headerTextColor</i>	color
<code>.rf-cal-hov</code> This class defines the styles for a cell when it is hovered over with the mouse cursor.	<i>calendarSpecBackgroundColor</i>	background-color
	<i>calendarSpecTextColor</i>	color
<code>.rf-cal-week</code> This class defines the styles for week numbers.	<i>panelBorderColor</i>	border-bottom-color, border-right-color
	<i>calendarWeekBackgroundColor</i>	background-color
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
<code>.rf-cal-holiday</code> This class defines the styles for weekends and holidays.	<i>calendarHolidaysBackgroundColor</i>	background-color
	<i>calendarHolidaysTextColor</i>	color

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-cal-boundary-day This class defines styles for an active boundary button.	No skin parameters.	
.rf-cal-sp-inp This class defines the styles for a spinner input field in the pop-up element for time selection.	<i>buttonSizeFont</i>	font-size
	<i>buttonFamilyFont</i>	font-family
.rf-cal-sp-inp-cntr This class defines the styles for a wrapper <td> element for a spinner input field in the pop-up element for time selection.	<i>controlBackgroundColor</i>	background-color
	<i>panelBorderColor</i>	border-color
	<i>subBorderColor</i>	border-right-color, border-bottom-color
.rf-cal-sp-btn This class defines the styles for a wrapper <td> element for spinner buttons in the pop-up element for time selection.	<i>headerBackgroundColor</i>	background-color, border-color
.rf-cal-sp-up This class defines styles for the Up spinner button.	No skin parameters.	
.rf-cal-sp-down This class defines styles for the Down spinner button.	No skin parameters.	
.rf-cal-sp-press This class defines styles for a spinner button when it is pressed.	No skin parameters.	
.rf-cal-edtr-shdw This class defines the styles for the calendar editor shadow.	<i>tableBackgroundColor</i>	background
.rf-cal-edtr-layout-shdw This class defines the styles for the layout shadow of a calendar editor.	<i>shadowBackgroundColor</i>	background-color

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-cal-edtr-btn This class defines styles for a button in the calendar editor.	No skin parameters.	
.rf-cal-edtr-btn-over This class defines the styles for the calendar editor button when it is hovered over with the mouse cursor.	<i>panelBorderColor</i>	border-color
	<i>calendarSpecBackgroundColor</i>	background
.rf-cal-edtr-btn-sel This class defines the styles for the calendar editor button when it is selected.	<i>calendarCurrentBackgroundColor</i>	background-color
	<i>calendarCurrentTextColor</i>	color
.rf-cal-edtr-tl-over This class defines the styles for a toolbar item in the calendar editor when it is hovered over with the mouse cursor.	<i>additionalBackgroundColor</i>	background
	<i>tableBackgroundColor</i>	border-color
	<i>panelBorderColor</i>	border-right-color, border-bottom-color
.rf-cal-edtr-tl-press This class defines the styles for a toolbar item in the calendar editor when it is pressed.	<i>additionalBackgroundColor</i>	background
	<i>panelBorderColor</i>	border-color
	<i>tableBackgroundColor</i>	border-right-color, border-bottom-color
.rf-cal-time-inp This class defines styles for the time input field.	No skin parameters.	
.rf-cal-time-btn This class defines the styles for a button in the pop-up element for the calendar's time section.	<i>tableBackgroundColor</i>	border-color
	<i>panelBorderColor</i>	border-right-color, border-bottom-color
.rf-cal-time-btn-press This class defines the styles for a pressed button in the pop-up element for the calendar's time section.	<i>tableBackgroundColor</i>	border-right-color, border-bottom-color
	<i>panelBorderColor</i>	border-color
	<i>calendarWeekBackgroundColor</i>	background-color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-cal-timepicker-cnt</code> This class defines the styles for the content of the pop-up element during time selection.	<code>panelBorderColor</code>	<code>border-color</code>
	<code>additionalBackgroundColor</code>	<code>background</code>
	<code>generalSizeFont</code>	<code>font-size</code>
	<code>generalFamilyFont</code>	<code>font-family</code>
<code>.rf-cal-timepicker-inp</code> This class defines the styles for an input field in the time picker.	<code>generalSizeFont</code>	<code>font-size</code>
	<code>generalFamilyFont</code>	<code>font-family</code>
<code>.rf-cal-timepicker-ok</code> This class defines styles for the OK button in the time picker.	No skin parameters.	
<code>.rf-cal-timepicker-cancel</code> This class defines styles for the Cancel button in the time picker.	No skin parameters.	
<code>.rf-cal-monthpicker-cnt</code> This class defines the styles for the content of the pop-up element during month or year selection.	<code>panelBorderColor</code>	<code>border-color</code>
	<code>tableBackgroundColor</code>	<code>background</code>
	<code>generalSizeFont</code>	<code>font-size</code>
	<code>generalFamilyFont</code>	<code>font-family</code>
<code>.rf-cal-monthpicker-ok</code> This class defines the styles for the OK button for the month picker.	<code>additionalBackgroundColor</code>	<code>background</code>
	<code>panelBorderColor</code>	<code>border-top-color</code>
<code>.rf-cal-monthpicker-cancel</code> This class defines the styles for the Cancel button for the month picker.	<code>additionalBackgroundColor</code>	<code>background</code>
	<code>panelBorderColor</code>	<code>border-top-color</code>
<code>.rf-cal-monthpicker-split</code> This class defines the styles for the splitter in the month picker.	<code>panelBorderColor</code>	<code>border-right-color</code>

A.2.3. `<rich:fileUpload>`

Table A.4. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-fu</code> This class defines styles for the file upload control.	<code>generalBackgroundColor</code>	background-color
	<code>panelBorderColor</code>	border-color
<code>.rf-fu-hdr</code> This class defines styles for the header of the file upload control.	<code>headerBackgroundColor</code>	background-color, border-color
<code>.rf-fu-lst</code> This class defines styles for lists in the file upload control.	No skin parameters.	
<code>.rf-fu-cntr-hdn</code> This class defines styles for the file upload container when it is hidden.	No skin parameters.	
<code>.rf-fu-btns-lft</code> , <code>.rf-fu-btns-rgh</code> These classes define styles for buttons on the left and right of the file upload control.	No skin parameters.	
<code>.rf-fu-btn-add</code> This class defines styles for the Add button in the file upload control.	<code>trimColor</code>	background-color
	<code>panelBorderColor</code>	border-color
<code>.rf-fu-btn-cnt-add</code> This class defines styles for the content of the Add button in the file upload control.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-fu-btn-add-dis</code> This class defines styles for the Add button in the file upload control when it is disabled.	<code>tableFooterBackgroundColor</code>	background-color
	<code>tableFooterBackgroundColor</code>	border-color

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-fu-btn-cnt-add-dis This class defines styles for the content of the Add button in the file upload control when it is disabled.	tabDisabledTextColor	color
	generalFamilyFont	font-family
	generalSizeFont	font-size
.rf-fu-btn-upl This class defines styles for the Upload button in the file upload control.	trimColor	background-color
	panelBorderColor	border-color
.rf-fu-btn-cnt-upl This class defines styles for the content of the Upload button in the file upload control.	generalTextColor	color
	generalFamilyFont	font-family
	generalSizeFont	font-size
.rf-fu-btn-clr This class defines styles for the Clear button in the file upload control.	trimColor	background-color
	panelBorderColor	border-color
.rf-fu-btn-cnt-clr This class defines styles for the content of the Clear button in the file upload control.	generalTextColor	color
	generalFamilyFont	font-family
	generalSizeFont	font-size
.rf-fu-itm This class defines styles for an item in the file upload control.	panelBorderColor	border-bottom-color
.rf-fu-itm-lft, .rf-fu-itm-rgh These classes define styles for items on the left and right of the file upload control.	No skin parameters.	
.rf-fu-itm-lbl This class defines styles for the label of an item in the file upload control.	generalTextColor	color
	generalFamilyFont	font-family
	generalSizeFont	font-size
.rf-fu-itm-st This class defines styles for the status of an item in the file upload control.	generalTextColor	color
	generalFamilyFont	font-family
	generalSizeFont	font-size

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-fu-itm-lnk</code> This class defines styles for a link item in the file upload control.	<code>generalLinkColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-fu-inp</code> This class defines styles for the input field in the file upload control.	No skin parameters.	
<code>.rf-fu-inp-cntr</code> This class defines styles for the input field container in the file upload control.	No skin parameters.	

A.2.4. `<rich:inplaceInput>`

Table A.5. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ii-d-s</code> This class defines styles for the in-place input when it is in the default state.	<code>editorBackgroundColor</code>	background-color
	<code>generalTextColor</code>	border-bottom-color
<code>.rf-ii-e-s</code> This class defines styles for the in-place input when it is in the editing state.	No skin parameters.	
<code>.rf-ii-c-s</code> This class defines styles for the in-place input when it is in the changed state.	No skin parameters.	
<code>.rf-ii-dis-s</code> This class defines styles for the in-place input when it is in the disabled state.	No skin parameters.	
<code>.rf-ii-flt</code> This class defines styles for the in-place input field.	<code>editBackgroundColor</code>	background-color, border-bottom-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size

Class (selector)	Skin Parameters		Mapped CSS properties
.rf-ii-dflt-lbl This class defines styles for the default label of the in-place input.	No skin parameters.		
.rf-ii-edit This class defines styles for the in-place input when it is being edited.	No skin parameters.		
.rf-ii-btn This class defines styles for the buttons for the in-place input.	<i>tabBackgroundColor</i>		background-color
	<i>panelBorderColor</i>		border-color
.rf-ii-btn-p This class defines styles for the buttons for the in-place input when they are pressed.	<i>tabBackgroundColor</i>		background-color
	<i>panelBorderColor</i>		border-color
.rf-ii-btn-set, .rf-ii-btn-prepos, .rf-ii-btn-pos These classes define the positioning of the buttons.	No skin parameters.		
.rf-ii-btn-shdw This class defines styles for the button shadows for the in-place input.	No skin parameters.		
.rf-ii-btn-shdw-t, .rf-ii-btn-shdw-b, .rf-ii-btn-shdw-l, .rf-ii-btn-shdw-r These classes define the top, bottom, left, and right edge of the button shadows.	No skin parameters.		
.rf-ii-none This class defines styles for the in-place input when it cannot be edited.	No skin parameters.		

A.2.5. `<rich:inplaceSelect>`

Table A.6. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-is-d-s</code> This class defines styles for the in-place select when it is in the default state.	<code>editorBackgroundColor</code>	background-color
	<code>generalTextColor</code>	border-bottom-color
<code>.rf-is-e-s</code> This class defines styles for the in-place select when it is in the editing state.	No skin parameters.	
<code>.rf-is-c-s</code> This class defines styles for the in-place select when it is in the changed state.	No skin parameters.	
<code>.rf-is-dis-s</code> This class defines styles for the in-place select when it is in the disabled state.	No skin parameters.	
<code>.rf-is-fl-d</code> This class defines styles for the in-place select field.	<code>editBackgroundColor</code>	background
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-is-opt</code> This class defines styles for an option for the in-place select.	<code>generalTextColor</code>	border-color
<code>.rf-is-sel</code> This class defines styles for the selected option of the in-place select.	<code>generalTextColor</code>	border-color
<code>.rf-is-lbl</code> This class defines styles for the label of the in-place select.	No skin parameters.	

Class (selector)	Skin Parameters		Mapped CSS properties
<code>.rf-is-dflt-lbl</code> This class defines styles for the default label of the in-place select.	No skin parameters.		
<code>.rf-is-edit</code> This class defines styles for the in-place select when it is being edited.	No skin parameters.		
<code>.rf-is-btn</code> This class defines styles for the buttons for the in-place select.	<code>tabBackgroundColor</code>		background-color
	<code>panelBorderColor</code>		border-color
<code>.rf-is-btn-p</code> This class defines styles for the buttons for the in-place select when they are pressed.	<code>tabBackgroundColor</code>		background-color
	<code>panelBorderColor</code>		border-color
<code>.rf-is-btn-set, .rf-is-btn-prepos, .rf-is-btn-pos</code> These classes define the positioning of the buttons.	No skin parameters.		
<code>.rf-is-lst-pos</code> This class defines the positioning of the list.	No skin parameters.		
<code>.rf-is-lst-dec</code> This class defines styles for a decreasing list for the in-place select.	<code>editBackgroundColor</code>		background-color
	<code>panelBorderColor</code>		border-color
<code>.rf-is-lst-scr1</code> This class defines styles for the list scrollbar.	No skin parameters.		
<code>.rf-is-shdw</code> This class defines styles for the in-place select shadow.	No skin parameters.		

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-is-shdw-t, .rf-is-shdw-b, .rf-is-shdw-l, .rf-is-shdw-r</code> These classes define the top, bottom, left, and right edge of the in-place select shadows.	No skin parameters.	
<code>.rf-is-btn-shdw</code> This class defines styles for the button shadows for the in-place select.	No skin parameters.	
<code>.rf-is-none</code> This class defines styles for the in-place select when it cannot be edited.	No skin parameters.	

A.2.6. <rich:inputNumbersSlider>

Table A.7. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-insl</code> This class defines styles for the number slider itself.	No skin parameters.	
<code>.rf-insl-trc</code> This class defines styles for the number slider track.	<code>controlBackgroundColor</code>	background-color
	<code>panelBorderColor</code>	border-bottom-color
<code>.rf-insl-trc-cntr</code> This class defines styles for the container of the number slider track.	No skin parameters.	
<code>.rf-insl-mn</code> This class defines styles for the minimum label on the number slider.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
	<code>generalTextColor</code>	color
	<code>panelBorderColor</code>	border-left-color
<code>.rf-insl-mx</code> This class defines styles for the maximum label on the number slider.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
	<code>generalTextColor</code>	color
	<code>panelBorderColor</code>	border-right-color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-insl-inp</code> This class defines styles for the input field on the number slider.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
	<code>generalTextColor</code>	color
<code>.rf-insl-inp-cntr</code> This class defines styles for the container of the input field.	No skin parameters.	
<code>.rf-insl-hnd</code> This class defines styles for the handle on the number slider.	No skin parameters.	
<code>.rf-insl-hnd-cntr</code> This class defines styles for the container of the handle.	No skin parameters.	
<code>.rf-insl-hnd-sel</code> This class defines styles for the handle when it is selected.	No skin parameters.	
<code>.rf-insl-hnd-dis</code> This class defines styles for the handle when it is disabled.	No skin parameters.	
<code>.rf-insl-dec</code> , <code>.rf-insl-inc</code> These classes define styles for the step controls to decrease and increase the number.	No skin parameters.	
<code>.rf-insl-dec-sel</code> , <code>.rf-insl-inc-sel</code> These classes define styles for the step controls when they are selected.	No skin parameters.	
<code>.rf-insl-dec-dis</code> , <code>.rf-insl-inc-dis</code> These classes define styles for the step controls when they are disabled.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-insl-tt</code> This class defines styles for the tool-tip on the number slider.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
	<code>generalTextColor</code>	color
	<code>tipBorderColor</code>	border
	<code>tipBackgroundColor</code>	background-color

A.2.7. `<rich:inputNumberSpinner>`

Table A.8. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-insp</code> This class defines styles for the number spinner itself.	<code>panelBorderColor</code>	border-color
<code>.rf-insp-inp</code> This class defines styles for the input field on the number spinner.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family
	<code>generalTextColor</code>	color
	<code>controlBackgroundColor</code>	background-color
<code>.rf-insp-btns</code> This class defines styles for the buttons on the number spinner.	<code>headerBackgroundColor</code>	background-color
	<code>panelBorderColor</code>	border-left-color
<code>.rf-insp-dec, .rf-insp-inc</code> These classes define styles for the step controls to decrease and increase the number.	No skin parameters.	
<code>.rf-insp-dec-dis, .rf-insp-inc-dis</code> These classes define styles for the step controls when they are disabled.	No skin parameters.	

A.2.8. <rich:select>

Table A.9. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-sel This class defines styles for the select control itself.	No skin parameters.	
.rf-sel-cntr This class defines styles for the container of the select control.	<i>panelBorderColor</i>	border-color
.rf-sel-inp This class defines styles for the select control input field.	<i>controlBackgroundColor</i>	background-color
.rf-sel-fld-err This class defines styles for the input field when an error occurs.	No skin parameters.	
.rf-sel-opt This class defines styles for an option in the select control.	<i>generalTextColor</i>	color
	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
.rf-sel-sel This class defines styles for the selected option of the select control.	<i>generalTextColor</i>	border-color
.rf-sel-dflt-lbl This class defines styles for the default label of the select control.	No skin parameters.	
.rf-sel-btn This class defines styles for the button of the select control.	<i>headerBackgroundColor</i>	background-color
	<i>panelBorderColor</i>	border-left-color
.rf-sel-btn-arrow This class defines styles for the arrow on the button.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-sel-btn-dis</code> This class defines styles for the button of the select control when it is disabled.	No skin parameters.	
<code>.rf-sel-lst-scr1</code> This class defines styles for the list scrollbar.	No skin parameters.	
<code>.rf-sel-shdw</code> This class defines styles for the select control shadow.	No skin parameters.	
<code>.rf-sel-shdw-t</code> , <code>.rf-sel-shdw-b</code> , <code>.rf-sel-shdw-l</code> , <code>.rf-sel-shdw-r</code> These classes define the top, bottom, left, and right edge of the select control shadows.	No skin parameters.	

A.3. Panels and containers

A.3.1. <rich:panel>

Table A.10. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-p</code> This class defines styles for the panel itself.	<code>generalBackgroundColor</code>	background-color
	<code>panelBorderColor</code>	color
<code>.rf-p-hdr</code> This class defines styles for the header of a panel.	<code>headerBackgroundColor</code>	background-color, border-color
	<code>headerTextColor</code>	color
	<code>headerSizeFont</code>	font-size
	<code>headerWeightFont</code>	font-weight
	<code>headerFamilyFont</code>	font-family
<code>.rf-p-b</code> This class defines styles for the body of a panel.	<code>generalTextColor</code>	color
	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family

A.3.2. <rich:accordion>

Table A.11. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ac</code> This class defines styles for the accordion control itself.	<code>panelBorderColor</code>	border-color
	<code>generalBackgroundColor</code>	background
<code>.rf-ac-itm-hdr</code> This class defines styles for the header of an accordion item.	<code>panelBorderColor</code>	border-bottom-color
	<code>headerBackgroundColor</code>	background-color
	<code>headerTextColor</code>	color
	<code>headerWeightFont</code>	font-weight
	<code>headerFamilyFont</code>	font-family
	<code>headerSizeFont</code>	font-size
<code>.rf-ac-itm-hdr-act, .rf-ac-itm-hdr-inact</code> These classes define styles for the header when the item is either active (expanded) or inactive (collapsed).	No skin parameters.	
<code>.rf-ac-itm-hdr-dis</code> This class defines styles for the header when it is disabled.	<code>tabDisabledTextColor</code>	color
<code>.rf-ac-itm-gr</code> This class defines styles for an item group.	No skin parameters.	
<code>.rf-ac-itm-cnt</code> This class defines styles for the content of an accordion item.	<code>panelBorderColor</code>	border-bottom-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-ac-itm-ico</code> This class defines styles for the item icon.	No skin parameters.	
<code>.rf-ac-itm-exp-ico</code> This class defines styles for the expanded icon for an item.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ac-itm-ico-act, .rf-ac-itm-ico-inact</code> These classes define styles for the icon when the item is either active (expanded) or inactive (collapsed).	No skin parameters.	
<code>.rf-ac-itm-lbl</code> This class defines styles for the item label.	No skin parameters.	
<code>.rf-ac-itm-lbl-act, .rf-ac-itm-lbl-inact</code> These classes define styles for the label when the item is either active (expanded) or inactive (collapsed).	No skin parameters.	

A.3.3. <rich:collapsiblePanel>

Table A.12. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-cp</code> This class defines styles for the collapsible panel itself.	<code>panelBorderColor</code>	color
	<code>generalBackgroundColor</code>	background
<code>.rf-cp-hdr</code> This class defines styles for the header of a collapsible panel.	<code>headerBackgroundColor</code>	background-color, border-color
	<code>headerTextColor</code>	color
	<code>headerWeightFont</code>	font-weight
	<code>headerFamilyFont</code>	font-family
	<code>headerSizeFont</code>	font-size
<code>.rf-cp-hdr-exp, .rf-cp-hdr-colps</code> These classes define styles for the header when the item is either expanded or collapsed.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-cp-gr This class defines styles for a collapsible panel group.	No skin parameters.	
.rf-cp-b This class defines styles for the body of a collapsible panel.	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-cp-ico This class defines styles for the panel icon.	No skin parameters.	
.rf-cp-exp-ico This class defines styles for the expanded icon for a panel.	No skin parameters.	
.rf-cp-ico-exp, .rf-cp-ico-colps These classes define styles for the icon when the panel is either expanded or collapsed.	No skin parameters.	
.rf-cp-lbl This class defines styles for the panel label.	No skin parameters.	
.rf-cp-lbl-exp, .rf-cp-lbl-colps These classes define styles for the label when the panel is either expanded or collapsed.	No skin parameters.	

A.3.4. <rich:popupPanel>

Table A.13. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-pp-btn This class defines styles for the pop-up panel button.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-pp-shade This class defines styles for the shading that covers the page when presenting a modal pop-up panel.	No skin parameters.	
.rf-pp-cntr This class defines styles for the container for the pop-up panel.	<i>panelBorderColor</i>	border
	<i>generalBackgroundColor</i>	background
.rf-pp-hdr This class defines styles for the header of the pop-up panel.	<i>headerBackgroundColor</i>	background
.rf-pp-hdr-cnt This class defines styles for the content of the header.	<i>headerTextColor</i>	color
	<i>headerWeightFont</i>	font-weight
	<i>headerFamilyFont</i>	font-family
	<i>headerSizeFont</i>	font-size
.rf-pp-cnt This class defines styles for the content of the pop-up panel.	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-pp-cnt-sclrlr This class defines styles for the scroll bars of the pop-up panel.	<i>generalBackgroundColor</i>	background
.rf-pp-hndlr This class defines styles for borders of the pop-up panel. The border handler is used to re-size the panel.	No skin parameters.	
.rf-pp-hndlr-t, .rf-pp-hndlr-b, .rf-pp-hndlr-l, .rf-pp-hndlr-r, .rf-pp-hndlr-tl, .rf-pp-hndlr-tr, .rf-pp-hndlr-bl, .rf-pp-hndlr-br These classes define styles for the top, bottom, left, right, top-left, top-	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
right, bottom-left, and bottom-right edges of the border handler.		

A.3.5. <rich:tabPanel>

Table A.14. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-tab-hdr This class defines styles for a tab header.	<i>panelBorderColor</i>	border
	<i>tabBackgroundColor</i>	background-color
	<i>generalTextColor</i>	color
.rf-tab-hdr-act This class defines styles for a tab header when it is active.	<i>additionalBackgroundColor</i>	background-color
.rf-tab-hdr-inact This class defines styles for a tab header when it is inactive.	No skin parameters.	
.rf-tab-hdr-dis This class defines styles for a tab header when it is disabled.	<i>tabDisabledTextColor</i>	color
.rf-tab-hdr-tabline-vis This class defines styles for the header tab line when it is visible.	<i>additionalBackgroundColor</i>	background-color
	<i>panelBorderColor</i>	border-color
.rf-tab-hdr-tabs This class defines styles for the tabs in the header.	No skin parameters.	
.rf-tab-hdr-spcr This class defines styles for the tab header spacer.	<i>panelBorderColor</i>	border-bottom
.rf-tab-lbl This class defines styles for the tab label.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-tab-hdn This class defines styles for the tab when it is hidden.	No skin parameters.	
.rf-tab-hdr-scr1-lft, .rf-tab-hdr-scr1-rgh These classes define styles for the left and right controls for the tab header scroller.	<i>additionalBackgroundColor</i>	background
	<i>panelBorderColor</i>	border
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-tab-hdr-tablst This class define styles for the tab header list.	<i>additionalBackgroundColor</i>	background
	<i>panelBorderColor</i>	border
	<i>generalFamilyFont</i>	font-family
.rf-tab-hdr-brd This class define styles for the tab header border.	<i>tabBackgroundColor</i>	background
	<i>panelBorderColor</i>	border
.rf-tab-cnt This class define styles for the content of the tab panel.	<i>generalBackgroundColor</i>	background
	<i>panelBorderColor</i>	border
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

A.4. Tables and grids

A.4.1. <rich:dataTable>

Table A.15. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-dt This class defines styles for the table.	<i>tableBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-left-width, border-top-width
	<i>tableBorderColor</i>	border-left-color, border-top-color
.rf-dt-cap This class defines styles for the table caption.	No skin parameters.	
.rf-dt-r This class defines styles for a table row.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-dt-fst-r</code> This class defines styles for the first row in a table.	No skin parameters.	
<code>.rf-dt-c</code> This class defines styles for a table cell.	<code>tableBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dt-nd</code> This class defines styles for a node.	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dt-hdr</code> This class defines styles for a table header.	No skin parameters.	
<code>.rf-dt-hdr-fst</code> This class defines styles for the first header.	No skin parameters.	
<code>.rf-dt-hdr-c</code> This class defines styles for a header cell.	<code>tableHeaderBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>tableHeaderTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dt-shdr</code> This class defines styles for a table sub-header.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-dt-shdr-fst</code> This class defines styles for the first sub-header.	No skin parameters.	
<code>.rf-dt-shdr-c</code> This class defines styles for a sub-header cell.	<code>tableHeaderBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>tableHeaderTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dt-ftr</code> This class defines styles for a table footer.	No skin parameters.	
<code>.rf-dt-ftr-fst</code> This class defines styles for the first footer.	No skin parameters.	
<code>.rf-dt-ftr-c</code> This class defines styles for a footer cell.	<code>tableFooterBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dt-sftr</code> This class defines styles for a table sub-footer.	No skin parameters.	
<code>.rf-dt-sftr-fst</code> This class defines styles for the first sub-footer.	No skin parameters.	
<code>.rf-dt-sftr-c</code> This class defines styles for a sub-footer cell.	<code>tableFooterBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>generalTextColor</code>	color

Class (selector)	Skin Parameters	Mapped CSS properties
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

A.4.2. <rich:collapsibleSubTable>

Table A.16. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-cst</code> This class defines styles for the table.	No skin parameters.	
<code>.rf-cst-r</code> This class defines styles for a table row.	No skin parameters.	
<code>.rf-cst-fst-r</code> This class defines styles for the first row in a table.	No skin parameters.	
<code>.rf-cst-c</code> This class defines styles for a table cell.	<i>tableBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
<code>.rf-cst-hdr</code> This class defines styles for a table header.	No skin parameters.	
<code>.rf-cst-hdr-fst</code> This class defines styles for the first header.	No skin parameters.	
<code>.rf-cst-hdr-fst-r</code> This class defines styles for the first row in the header.	No skin parameters.	
<code>.rf-cst-hdr-c</code> This class defines styles for a header cell.	<i>tableSubHeaderBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width

Class (selector)	Skin Parameters	Mapped CSS properties
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-cst-shdr This class defines styles for a table sub-header.	No skin parameters.	
.rf-cst-shdr-fst This class defines styles for the first sub-header.	No skin parameters.	
.rf-cst-shdr-c This class defines styles for a sub-header cell.	<i>tableSubHeaderBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-cst-ftr This class defines styles for a table footer.	No skin parameters.	
.rf-cst-ftr-fst This class defines styles for the first footer.	No skin parameters.	
.rf-cst-ftr-c This class defines styles for a footer cell.	<i>tableSubFooterBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-cst-sftr This class defines styles for a table sub-footer.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-cst-sftr-fst</code> This class defines styles for the first sub-footer.	No skin parameters.	
<code>.rf-cst-sftr-c</code> This class defines styles for a sub-footer cell.	<code>tableSubFooterBackgroundColor</code>	background-color
	<code>tableBorderWidth</code>	border-bottom-width, border-right-width
	<code>tableBorderColor</code>	border-bottom-color, border-right-color
	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size

A.4.3. <rich:collapsibleSubTableToggler>

Style classes (selectors)

`.rf-csttg`

This class defines styles for a toggle control.

`.rf-csttg-exp`

This class defines styles for a toggle control which expands the sub-table.

`.rf-csttg-colps`

This class defines styles for a toggle control which collapses the sub-table.

A.4.4. <rich:extendedDataTable>

Table A.17. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-edt</code> This class defines styles for the table.	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	border
	<code>tableBackgroundColor</code>	background-color
<code>.rich-edt-cnt</code> This class defines styles for the table content.	No skin parameters.	
<code>.rf-edt-c</code> This class defines styles for a table cell.	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	border-bottom
	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	border-right

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-edt-c-cnt This class defines styles for the contents of a cell.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-edt-tbl-hdr This class defines styles for the table header.	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-bottom
	<i>tableHeaderTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
	<i>tableHeaderTextColor</i>	color
.rich-edt-hdr This class defines styles for a header.	No skin parameters.	
.rf-edt-hdr-c This class defines styles for a table header cell.	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-bottom
	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-right
.rf-edt-hdr-c-cnt This class defines styles for the contents of a header cell.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
	<i>tableHeaderTextColor</i>	color
.rf-edt-tbl-ftr This class defines styles for the table footer.	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-top
	<i>tableFooterBackgroundColor</i>	background-color
.rich-edt-ftr This class defines styles for a footer.	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-top
	<i>tableFooterBackgroundColor</i>	background-color
.rich-edt-ftr-cnt This class defines styles for the content of a footer.	No skin parameters.	
.rf-edt-ftr-c This class defines styles for a table footer cell.	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-bottom
	<i>tableBorderWidth</i> , <i>tableBorderColor</i>	border-right
.rf-edt-ftr-c-cnt This class defines styles for the contents of a footer cell.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
	<i>generalTextColor</i>	color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-edt-ftr-emp</code> This class defines styles for an empty footer cell.	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	<code>border-right</code>
<code>.rich-edt-ftr-fzn</code> This class defines styles for a frozen footer.	No skin parameters.	
<code>.rich-edt-b</code> This class defines styles for the body of the table.	No skin parameters.	
<code>.rf-edt-r-sel</code> This class defines styles for the selected row.	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	<code>border-right</code>
<code>.rich-edt-r-act</code> This class defines styles for the active row.	No skin parameters.	
<code>.rich-edt-rsz</code> This class defines styles for the table resizer.	No skin parameters.	
<code>.rich-edt-rsz-cntr</code> This class defines styles for the resize container.	No skin parameters.	
<code>.rich-edt-rsz-mkr</code> This class defines styles for the resize marker.	<code>generalTextColor</code>	<code>border-left</code>
<code>.rf-edt-rord</code> This class defines styles for the re-order functionality.	<code>tableBorderWidth</code> , <code>tableBorderColor</code>	<code>border</code>
	<code>tableHeaderBackgroundColor</code> / <code>tableBackgroundColor</code>	<code>background-color</code>
<code>.rich-edt-rord-mkr</code> This class defines styles for the re-order marker.	No skin parameters.	
<code>.rich-edt-spcr</code> This class defines a spacer for Internet Explorer 7 compatibility.	No skin parameters.	

A.4.5. <rich:dataGrid>

Table A.18. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-dg This class defines styles for the grid.	<i>tableBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-left-width, border-top-width
	<i>tableBorderColor</i>	border-left-color, border-top-color
.rf-dg-cap This class defines styles for the grid caption.	No skin parameters.	
.rf-dg-r This class defines styles for a grid row.	No skin parameters.	
.rf-dg-c This class defines styles for a grid cell.	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-dg-nd-c This class defines styles for a node cell.	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-dg-th This class defines styles for the grid header section.	<i>tableBorderWidth</i>	border-bottom-width
	<i>tableBorderColor</i>	border-bottom-color
.rf-dg-h This class defines styles for a grid header.	No skin parameters.	
.rf-dg-h-f This class defines styles for the first header.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-dg-h-r This class defines styles for a header row.	No skin parameters.	
.rf-dg-h-c This class defines styles for a header cell.	<i>tableHeaderBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>tableHeaderTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-dg-f This class defines styles for a grid footer.	No skin parameters.	
.rf-dg-f-f This class defines styles for the first footer.	No skin parameters.	
.rf-dg-f-c This class defines styles for a footer cell.	<i>tableFooterBackgroundColor</i>	background-color
	<i>tableBorderWidth</i>	border-bottom-width, border-right-width
	<i>tableBorderColor</i>	border-bottom-color, border-right-color
	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

A.4.6. <rich:list>

Table A.19. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-ulst-itm This class defines styles for an item in an unordered list.	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-olst-itm This class defines styles for an item in an unordered list.	<i>generalTextColor</i>	color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-dlst-trm</code> This class defines styles for the term of an item in a definition list.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-dlst-dfn</code> This class defines styles for the definition of an item in a definition list.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size

A.4.7. `<rich:dataScroller>`

Table A.20. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ds</code> This class defines styles for the data scroller.	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
	<code>tableBackgroundColor</code>	background
<code>.rf-ds-btn</code> This class defines styles for buttons in the data scroller.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
	<code>tableBorderColor</code>	border-color
	<code>headerBackgroundColor</code>	background-color
<code>.rf-ds-btn-first</code> This class defines styles for the first button.	No skin parameters.	
<code>.rf-ds-btn-fastrwd</code> This class defines styles for the fast rewind button.	No skin parameters.	
<code>.rf-ds-btn-prev</code> This class defines styles for the previous button.	No skin parameters.	
<code>.rf-ds-btn-next</code> This class defines styles for the next button.	No skin parameters.	
<code>.rf-ds-btn-fastfwd</code> This class defines styles for the fast forward button.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ds-btn-last</code> This class defines styles for the last button.	No skin parameters.	
<code>.rf-ds-nmb-btn</code> This class defines styles for page number buttons in the data scroller.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
	<code>tableBorderColor</code>	border-color
	<code>tableBackgroundColor</code>	background-color
<code>.rf-ds-press</code> This class defines styles for a data scroller when a control is pressed.	<code>tableBorderColor</code>	border-color
<code>tableBackgroundColor</code>	background	
<code>.rf-ds-act</code> This class defines styles for an active data scroller.	<code>tableBorderColor</code>	color
<code>.rf-ds-dis</code> This class defines styles for a disabled data scroller.	<code>tableBorderColor</code>	color

A.5. Trees

A.5.1. `<rich:tree>`

Style classes (selectors)

`.rf-tr-nd`

This class defines styles for the nodes in a tree.

`.rf-tr-nd-last`

This class defines styles for last node in a tree.

`.rf-tr-nd-colps`

This class defines styles for a collapsed tree node.

`.rf-tr-nd-exp`

This class defines styles for an expanded tree node.

A.5.2. <rich:treeNode>

Table A.21. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-trn This class defines styles for a tree node.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-trn-lbl This class defines styles for a tree node label.	No skin parameters.	
.rf-trn-cnt This class defines styles for tree node content.	No skin parameters.	
.rf-trn-sel This class defines styles for a selected tree node.	<i>additionalBackgroundColor</i>	background
.rf-trn-ldn This class defines styles for a tree node when it is loading.	<i>additionalBackgroundColor</i>	background
.rf-trn-hnd This class defines styles for a tree node handle.	No skin parameters.	
.rf-trn-hnd-lf This class defines styles for the handle of a leaf node.	No skin parameters.	
.rf-trn-hnd-colps This class defines styles for the handle of a collapsed node.	No skin parameters.	
.rf-trn-hnd-exp This class defines styles for the handle of an expanded node.	No skin parameters.	
.rf-trn-hnd-ldn-fct This class defines styles for the loading facet of a tree node handle.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-trn-ico</code> This class defines styles for tree node icon.	No skin parameters.	
<code>.rf-trn-ico-lf</code> This class defines styles for the icon of a leaf node.	No skin parameters.	
<code>.rf-trn-ico-colps</code> This class defines styles for the icon of a collapsed node.	No skin parameters.	
<code>.rf-trn-ico-exp</code> This class defines styles for the icon of an expanded node.	No skin parameters.	
<code>.rf-trn-ico-cst</code> This class defines styles for a custom node icon.	No skin parameters.	

A.6. Menus and toolbars

A.6.1. `<rich:dropDownMenu>`

Table A.22. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ddm-lbl</code> This class defines styles for the label of the drop-down menu.	<i>headerFamilyFont</i>	font-family
<code>.rf-ddm-dis</code> This class defines styles for the drop-down menu when it is disabled.	<i>tabDisabledTextColor</i>	color
<code>.rf-ddm-lbl-dis</code> This class defines styles for the label of the drop-down menu when it is disabled.	<i>headerFamilyFont</i>	font-family

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-ddm-pos This class defines the positioning of the drop-down menu.	No skin parameters.	
.rf-ddm-lbl-unsel This class defines styles for the label of the drop-down menu when it is unselected.	No skin parameters.	
.rf-ddm-lst This class defines styles for the drop-down list.	<i>panelBorderColor</i>	border-color
	<i>additionalBackgroundColor</i>	background-color
.rf-ddm-lst-bg This class defines styles for the background of the drop-down list.	<i>additionalBackgroundColor</i>	border-color
.rf-ddm-sublst This class defines the positioning of the menu when used as a sub-menu.	No skin parameters.	
.rf-ddm-itm This class defines styles for a menu item.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-ddm-itm-sel This class defines styles for a menu item when it is selected.	<i>headerBackgroundColor</i>	border-color
	<i>tabBackgroundColor</i>	background-color
.rf-ddm-itm-unsel This class defines styles for a menu item when it is unselected.	No skin parameters.	
.rf-ddm-itm-dis This class defines styles for a menu item when it is disabled.	<i>tabDisabledTextColor</i>	color
.rf-ddm-itm-lbl This class defines styles for the label in a menu item.	<i>generalTextColor</i>	color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-ddm-itm-ic</code> This class defines styles for the icon in a menu item.	No skin parameters.	
<code>.rf-ddm-emptyIcon</code> This class defines styles for an empty icon in a menu item.	No skin parameters.	
<code>.rf-ddm-sep</code> This class defines styles for a menu separator.	<code>panelBorderColor</code>	<code>border-top-color</code>
<code>.rf-ddm-nd</code> This class defines styles for a menu node.	No skin parameters.	

A.6.2. <rich:panelMenu>

Table A.23. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-pm</code> This class defines styles for the panel menu itself.	No skin parameters.	
<code>.rf-pm-gr</code> This class defines styles for a panel menu group.	<code>panelBorderColor</code>	<code>border-top-color</code>
<code>.rf-pm-exp, .rf-pm-colps</code> These classes define styles for the panel menu when it is expanded or collapsed.	No skin parameters.	
<code>.rf-pm-ico</code> This class defines styles for the panel menu icons.	No skin parameters.	
<code>.rf-pm-ico-exp, .rf-pm-ico-colps</code> These classes define styles for the panel menu icons when they are expanded or collapsed.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-pm-hdr-exp, .rf-pm-hdr-colps</code> These classes define styles for the panel menu headers when they are expanded or collapsed.	No skin parameters.	
<code>.rf-pm-itm</code> This class defines styles for a panel menu item.	<code>panelBorderColor</code>	border-top-color
	<code>generalTextColor</code>	color
<code>.rf-pm-itm-gr</code> This class defines styles for a panel menu item as part of a panel menu group.	No skin parameters.	
<code>.rf-pm-itm: hover</code> This class defines styles for a panel menu item when the mouse hovers over it.	<code>additionalBackgroundColor</code>	background-color
<code>.rf-pm-itm-sel</code> This class defines styles for a panel menu item when it is selected.	No skin parameters.	
<code>.rf-pm-itm-dis</code> This class defines styles for a panel menu item when it is disabled.	<code>tabDisabledTextColor</code>	color
<code>.rf-pm-itm-ico</code> This class defines styles for the icon in a panel menu item.	No skin parameters.	
<code>.rf-pm-itm-exp-ico</code> This class defines styles for the icon in a panel menu item when it is expanded.	No skin parameters.	
<code>.rf-pm-itm-lbl</code> This class defines styles for the label in a panel menu item.	<code>generalSizeFont</code>	font-size
	<code>generalFamilyFont</code>	font-family

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-pm-gr This class defines styles for a panel menu group.	<i>panelBorderColor</i>	border-top-color
.rf-pm-gr-gr This class defines styles for a panel menu group as part of another panel menu group.	No skin parameters.	
.rf-pm-gr-sel This class defines styles for a panel menu group when it is selected.	No skin parameters.	
.rf-pm-gr-hdr This class defines styles for the header of a panel menu group.	<i>generalTextColor</i>	color
.rf-pm-gr-hdr:hover This class defines styles for the header of a panel menu group when the mouse hovers over it.	<i>additionalBackgroundColor</i>	background
.rf-pm-gr-hdr-dis This class defines styles for the header of a panel menu group when it is disabled.	<i>tabDisabledTextColor</i>	color
.rf-pm-gr-ico This class defines styles for the icon in a panel menu group.	No skin parameters.	
.rf-pm-gr-exp-ico This class defines styles for the icon in a panel menu group when it is expanded.	No skin parameters.	
.rf-pm-gr-lbl This class defines styles for the label in a panel menu group.	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-pm-gr-cnt This class defines styles for the content of a panel menu group.	No skin parameters.	
.rf-pm-top-itm This class defines styles for the top panel menu item.	<i>panelBorderColor</i>	border-color
	<i>generalTextColor</i>	color
.rf-pm-top-itm-gr This class defines styles for the top panel menu item as part of a panel menu group.	No skin parameters.	
.rf-pm-top-itm: hover This class defines styles for the top panel menu item when the mouse hovers over it.	<i>headerTextColor</i>	color
.rf-pm-top-itm-sel This class defines styles for the top panel menu item when it is selected.	No skin parameters.	
.rf-pm-top-itm-dis This class defines styles for the top panel menu item when it is disabled.	<i>tabDisabledTextColor</i>	color
.rf-pm-top-itm-ico This class defines styles for the icon in the top panel menu item.	No skin parameters.	
.rf-pm-top-itm-exp-ico This class defines styles for the icon in the top panel menu item when it is expanded.	No skin parameters.	
.rf-pm-top-itm-lbl This class defines styles for the label in the top panel menu item.	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-pm-top-gr This class defines styles for the top panel menu group.	<i>panelBorderColor</i>	border-color
.rf-pm-top-gr-gr This class defines styles for the top panel menu group as part of another panel menu group.	No skin parameters.	
.rf-pm-top-gr-sel This class defines styles for the top panel menu group when it is selected.	No skin parameters.	
.rf-pm-top-gr-hdr This class defines styles for the header of the top panel menu group.	<i>headerTextColor</i>	color
	<i>headerBackgroundColor</i>	background-color
.rf-pm-top-gr-hdr-dis This class defines styles for the header of the top panel menu group when it is disabled.	<i>tabDisabledTextColor</i>	color
	<i>additionalBackgroundColor</i>	background-color
.rf-pm-top-gr-ico This class defines styles for the icon in the top panel menu group.	No skin parameters.	
.rf-pm-top-gr-exp-ico This class defines styles for the icon in the top panel menu group when it is expanded.	No skin parameters.	
.rf-pm-top-gr-lbl This class defines styles for the label in the top panel menu group.	<i>generalSizeFont</i>	font-size
	<i>generalFamilyFont</i>	font-family
.rf-pm-top-gr-cnt This class defines styles for the content of the top panel menu group.	No skin parameters.	

A.6.3. <rich:toolbar>

Table A.24. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-tb This class defines styles for the toolbar itself.	<i>panelBorderColor</i>	border-color
	<i>headerTextColor</i>	color
	<i>headerBackgroundColor</i>	background-color
	<i>headerFamilyFont</i>	font-family
	<i>headerSizeFont</i>	font-size
	<i>headerWeightFont</i>	font-weight
.rf-tb-itm This class defines styles for an item in the toolbar.	No skin parameters.	
.rf-tb-sep This class defines styles for a separator in the toolbar.	No skin parameters.	
.rf-tb-sep-grid, .rf-tb-sep-line, .rf-tb-sep-disc, .rf-tb-sep-square These classes define styles for grid, line, disc, and square separators.	No skin parameters.	
.rf-tb-cntr This class defines styles for the container of the toolbar.	No skin parameters.	

A.7. Output and messages

A.7.1. <rich:message>

Table A.25. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-msg This class defines styles for the message itself.	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size
.rf-msg-err This class defines styles for an error message.	<i>errorColor</i>	color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-msg-ftl</code> This class defines styles for a fatal message.	<code>errorColor</code>	color
<code>.rf-msg-inf</code> This class defines styles for an information message.	<code>generalTextColor</code>	color
<code>.rf-msg-wrn</code> This class defines styles for a warning message.	<code>warningTextColor</code>	color
<code>.rf-msg-ok</code> This class defines styles for a basic OK message.	<code>generalTextColor</code>	color
<code>.rf-msg-sum, .rf-msg-det</code> These classes define styles for the summary or details of a message.	No skin parameters.	

A.7.2. <rich:messages>

Table A.26. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-msgs</code> This class defines styles for the message itself.	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size
<code>.rf-msgs-err</code> This class defines styles for an error message.	<code>errorColor</code>	color
<code>.rf-msgs-ftl</code> This class defines styles for a fatal message.	<code>errorColor</code>	color
<code>.rf-msgs-inf</code> This class defines styles for an information message.	<code>generalTextColor</code>	color
<code>.rf-msgs-wrn</code> This class defines styles for a warning message.	<code>warningTextColor</code>	color

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-msgs-ok</code> This class defines styles for a basic OK message.	<code>generalTextColor</code>	color
<code>.rf-msgs-sum</code> , <code>.rf-msgs-det</code> These classes define styles for the summary or details of a message.	No skin parameters.	

A.7.3. `<rich:progressBar>`

Table A.27. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-pb-lbl</code> This class defines styles for labels on the progress bar.	No skin parameters.	
<code>.rf-pb-prgs</code> This class defines styles for the progressed portion of the progress bar.	<code>panelBorderColor</code>	border-color
	<code>selectControlColor</code>	background-color
<code>.rf-pb-init</code> , <code>.rf-pb-fin</code> These classes define styles for the initial state and finished state.	<code>generalTextColor</code>	color
	<code>generalFamilyFont</code>	font-family
	<code>generalSizeFont</code>	font-size

A.7.4. `<rich:tooltip>`

Table A.28. Style classes (selectors) and corresponding skin parameters

Class (selector)	Skin Parameters	Mapped CSS properties
<code>.rf-tt</code> This class defines styles for the tool-tip itself.	No skin parameters.	
<code>.rf-tt-loading</code> This class defines styles for the tool-tip when it is loading.	No skin parameters.	
<code>.rf-tt-cnt</code> This class defines styles for the tool-tip content.	No skin parameters.	

Class (selector)	Skin Parameters	Mapped CSS properties
.rf-tt-cntr This class defines styles for the progressed portion of the progress bar.	<i>tipBorderColor</i>	border-color
	<i>generalFamilyFont</i>	font-family
	<i>generalSizeFont</i>	font-size

A.8. Drag and drop

A.8.1. <rich:dropTarget>

Style classes (selectors)

.rf-drp-hvr

This class defines styles for the drop target when a dragged item is hovering over it.

.rf-drp-hlight

This class defines styles for a highlighted drop target.

A.8.2. <rich:dragIndicator>

Style classes (selectors)

.rf-ind

This class defines styles for the drag indicator.

.rf-ind-drag.accept

This class defines styles for the indicator when it is over an acceptable drop target.

.rf-ind-drag.reject

This class defines styles for the indicator when it is over an unacceptable drop target.

.rf-ind-drag.default

This class defines styles for the indicator when it is being dragged, and is not over any drop targets.

Appendix B. Revision History

Revision History

Revision 1.0

Mon Apr 11 2011

SeanRogers

4.0.0.Final Release

