

Common-Controls Tools ResourceFactory

Version 1.4.055 - Date: 31. Oktober 2004

Published by:

SCC Informationssysteme GmbH
64367 Mühlthal (Germany)

Tel: +49 (0) 6151 / 13 6 31 0
Internet www.scc-gmbh.com

Product Site

<http://www.common-controls.com>

Copyright © 2000 - 2004 SCC Informationssysteme GmbH.
All rights reserved. Published 2003

No part of this publication may be stored in a retrieval system, transmitted, or reproduced in any way without the prior agreement and written permission of SCC Informationssysteme GmbH.

Sun, Sun Microsystems, the Sun Logo, Java, JavaServer Pages are registered trademarks of Sun Microsystems Inc in the U.S.A. and other Countries.

Microsoft, Microsoft Windows or other Microsoft Produkte are a registered trademark of Microsoft Corporation in the U.S.A. and other Countries.

Netscape, Netscape Navigator is a registered trademark of Netscape Communications Corp in the U.S.A. and other Countries.

All other product names, marks, logos, and symbols may be trademarks or registered trademarks of their respective owners.

Table of content

1	Introduction.....	1
2	Description of the Resources to be Generated.....	2
2.1	Example of the structure of the resource file.....	2
2.2	Document Type Definition (DTD).....	6
2.3	The environment.....	10
2.3.1	Path name.....	10
2.3.2	Expressions.....	10
2.3.3	Default identifier.....	12
2.4	Elemente der Ressourcen XML Datei.....	13
2.4.1	<environment> Element.....	13
2.4.2	<property> Element.....	13
2.4.3	<definitions> Element.....	13
2.4.4	<color> Element.....	14
2.4.5	 Element.....	14
2.4.6	<resources> Element.....	14
2.4.7	<stylebook> Element.....	15
2.4.8	<textfile> Element.....	15
2.4.9	<button> Element.....	16
2.4.10	<menu> Element.....	17
2.4.11	<bundle> Element.....	18
2.4.12	<resourcekey> Element.....	19
2.4.13	<value> Element.....	19
3	Starting the build process.....	20
3.1	Requirements.....	20
3.2	Using the comand line.....	20
3.3	Using the ant tasks.....	20
3.3.1	Setting the build properties.....	20
3.3.2	<fontlist> Task.....	20
3.3.3	<resourcefactory> Task.....	21
3.3.4	<version> Task.....	22
4	Outputting the available Fonts.....	24
5	Support.....	25

1 Introduction

The ResourceFactory Tool is used for the automatic generation of resources for the Common Controls Framework. The present version supports (see Figure 1):

- The automatic generation of button images for the DefaultPainter (Def) and DefaultPainter2 (Def2)
- The generation of StyleSheet files for customizing the color schemes of the DefaultPainters to an individual StyleSheet.
- StyleBook → Description of a StyleSheet in an HTML file.
- The generation of the Java ColorPalette class for a Resource Table derived from the DefResourceMap.
- Generation and management von Resource Bundels

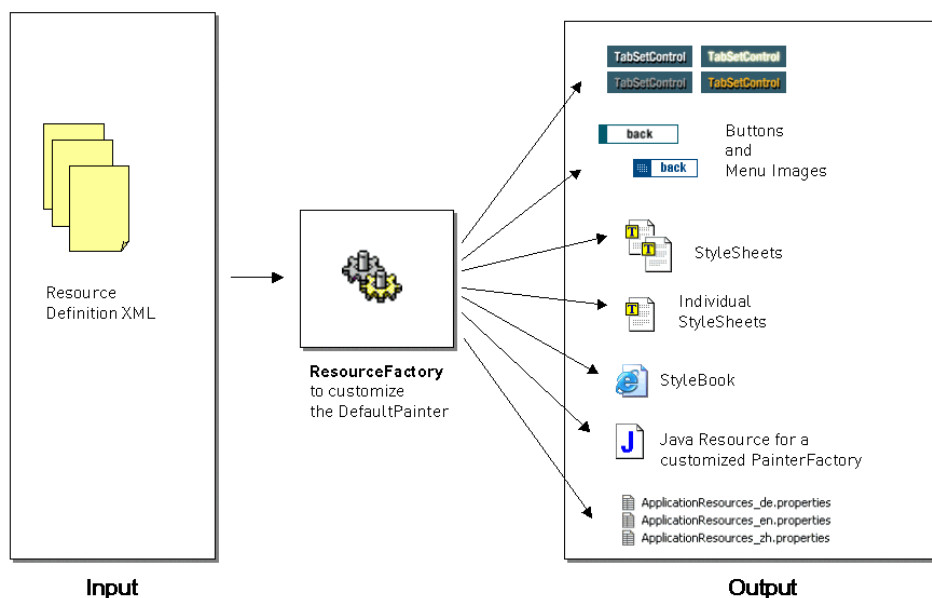


Figure 1: ResourceFactory

The following steps are required for using the tool:

1. Description of the resources in an XML file
2. Setting the Build Properties
3. Starting the Build process

Important note:

In the generation of graphics with text, serious quality variations have been observed depending on the JDK version employed. Please use the JDK Version 1.4.1 to achieve the best possible results.

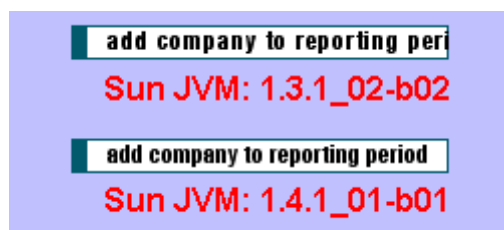


Figure 2: Effects of the JDK Version in the generation of buttons

2 Description of the Resources to be Generated

To start with, the resources to be generated have to be defined in an XML-file. To do so, a new file can be created or the accompanying file can be customized as required.

2.1 Example of the structure of the resource file

The following illustration shows an example of the structure of a resource file. The color values for the StyleSheet of the DefaultPainter and the buttons for a German and English language version are defined within the file. The resources are saved to different sub-directories when they are generated.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE resource-factory PUBLIC
  "-//common-controls//DTD ResourceFactory 1.1//EN"
  "http://www.common-controls.com/dtds/resource-factory_1_1.dtd">

<resource-factory version="1.1">

  <!--
  use color macros to reduce the number of
  different color values so it's more easy
  to change the entire stylesheet
  -->

  <property name="col00" value="#000000"/>
  <property name="col01" value="#0000ff"/>
  <property name="col02" value="#005a6b"/>
  <property name="col03" value="#80adba"/>
  <property name="col04" value="#84adbd"/>
  <property name="col05" value="#87b1ba"/>
  <property name="col06" value="#8d9da1"/>
  <property name="col07" value="#a5c4cb"/>
  <property name="col08" value="#a7c2c7"/>
  <property name="col09" value="#b4ced4"/>
  <property name="col10" value="#bdbdbd"/>
  <property name="col11" value="#c1d6db"/>
  <property name="col12" value="#c4c8c9"/>
  <property name="col13" value="#c7003c"/>
  <property name="col14" value="#cecece"/>
  <property name="col15" value="#dadfe0"/>
  <property name="col16" value="#dce8eb"/>
  <property name="col17" value="#edeff0"/>
  <property name="col18" value="#efefef"/>
  <property name="col19" value="#f3f4f5"/>
  <property name="col20" value="#f57e17"/>
  <property name="col21" value="#fae4c2"/>
  <property name="col22" value="#fea217"/>
  <property name="col23" value="#ffa510"/>
  <property name="col24" value="#ffd3d3"/>
  <property name="col25" value="#ffffe1"/>
  <property name="col26" value="#ffffff"/>

  <environment>

    <!-- ===== -->
    <!-- Style Definitions -->
    <!-- ===== -->

    <definitions code="form" name="Forms: Formulare">
      <definitions code="color" name="Colortable">
        <color code="bg.field" name="Background field" value="{ col18}"/>
        <color code="bg.header" name="Background caption" value="{ col02}"/>
        <color code="bg.label" name="Background label" value="{ col15}"/>
        <color code="bg.section" name="Background section" value="{ col08}"/>
        <color code="bg" name="Background color" value="{ col15}"/>
        <color code="border.item" name="Border row" value="{ col10}"/>
        <color code="border.section" name="Border section" value="{ col02}"/>
        <color code="border" name="Border" value="{ col02}"/>
        <color code="text.caption" name="Text color caption" value="{ col26}"/>
        <color code="text.detail" name="Text color detail" value="{ col26}"/>
        <color code="text.header" name="Text color header" value="{ col26}"/>
      </definitions>
    </definitions>
  </environment>
</resource-factory>
```

```

    <color code="text.section"    name="Text color section"    value="{col02}" />
  </definitions>
</definitions>

<definitions code="error" name="Forms: Error form">
  <definitions code="color" name="Colortable">
    <color code="bg.header"    name="Background caption"    value="{col13}" />
    <color code="bg.body"      name="Background body"      value="{col24}" />
    <color code="border"       name="Border color"         value="{col13}" />
    <color code="text"         name="Text color"           value="{col13}" />
  </definitions>
</definitions>

<definitions code="info" name="Forms: Informationsformular">
  <definitions code="color" name="Colortable">
    <color code="bg.header"    name="Background caption"    value="{col07}" />
    <color code="bg.body"      name="Background body"      value="{col16}" />
    <color code="border"       name="Border color"         value="{col07}" />
    <color code="text"         name="Text color"           value="{col02}" />
  </definitions>
</definitions>

<definitions code="head" name="Forms: Header Form">
  <definitions code="color"    name="Colortable">
    <color code="bg"           name="Background"           value="{col15}" />
    <color code="border"       name="Border color"         value="{col02}" />
    <color code="separator"    name="Seperator"           value="{col26}" />
  </definitions>
</definitions>

<definitions code="hmenu" name="Controls: Horizontal Menu">
  <definitions code="color"    name="Colortable">
    <color code="bg"           name="Background"           value="{col09}" />
  </definitions>
</definitions>

<definitions code="vmenu1" name="Controls: Vertical menu level 1">
  <definitions code="color"    name="Colortable">
    <color code="bg"           name="Background"           value="{col05}" />
    <color code="bg.sel"       name="Background selected"  value="{col05}" />
    <color code="border"       name="Border level 1"       value="{col26}" />
    <color code="text.A"       name="Text hyperlink"       value="{col02}" />
    <color code="text.A.sel"   name="Text hyperlink sel."  value="{col02}" />
  </definitions>
</definitions>

<definitions code="vmenu2" name="Controls: Vertical menu level 2">
  <definitions code="color"    name="Colortable">
    <color code="bg"           name="Background"           value="{col11}" />
    <color code="bg.sel"       name="Background selected"  value="{col22}" />
    <color code="border"       name="Border 2"             value="{col26}" />
    <color code="text.A"       name="Text hyperlink"       value="{col02}" />
    <color code="text.A.sel"   name="Text hyperlink sel."  value="{col26}" />
    <color code="text"         name="Text"                 value="{col02}" />
    <color code="text.sel"     name="Text selected"        value="{col26}" />
  </definitions>
</definitions>

<definitions code="vmenu3" name="Controls: Vertical menu level 3">
  <definitions code="color"    name="Colortable">
    <color code="bg"           name="Background"           value="{col21}" />
    <color code="bg.sel"       name="Background selected"  value="{col21}" />
    <color code="border"       name="Border"                 value="{col26}" />
    <color code="text"         name="Text"                 value="{col00}" />
    <color code="text.A"       name="Text hyperlink"       value="{col00}" />
    <color code="text.A.sel"   name="Text hyperlink sel."  value="{col00}" />
  </definitions>
</definitions>

<definitions code="list" name="Controls: Liste">
  <definitions code="color"    name="Colortable">
    <color code="bg.body"      name="Background body"      value="{col15}" />
    <color code="bg.colheader" name="Bg column header"    value="{col03}" />
    <color code="bg.even"      name="Background even Rows" value="{col17}" />
    <color code="bg.header"    name="Background header"    value="{col02}" />
    <color code="bg.INPUT"     name="BG input field"        value="{col26}" />
    <color code="bg.odd"       name="Background odd Rows"  value="{col26}" />
  </definitions>
</definitions>

```

```

    <color code="border.body"      name="Border body"      value="{col02}" />
    <color code="border.INPUT"    name="Border input field" value="{col02}" />
    <color code="text.A"          name="Text color Hyperlink" value="{col01}" />
    <color code="text.colheader"  name="Text column label"  value="{col26}" />
    <color code="text.caption"    name="Text color caption" value="{col26}" />
    <color code="text.detail"     name="Text detail text"   value="{col26}" />
    <color code="text.header"     name="Text color header"  value="{col26}" />
  </definitions>
</definitions>

<definitions code="tree" name="Controls: Tree">
  <definitions code="color" name="Colortable">
    <color code="bg.even"        name="Background even row" value="{col19}" />
    <color code="bg.odd"         name="Background odd row"  value="{col26}" />
    <color code="bg.sel"         name="BG selected row"     value="{col22}" />
    <color code="border"         name="Border row"         value="{col16}" />
    <color code="text.A"         name="Text hyperlink"     value="{col00}" />
    <color code="text.A.sel"     name="Text hyperl.sel. row" value="{col26}" />
  </definitions>
</definitions>

<definitions code="tabset" name="Controls: Tabset">
  <definitions code="color" name="Colortable">
    <color code="bg.disabled"    name="Bg inactiv Tab"     value="{col06}" />
    <color code="bg.level1"      name="Background level 1" value="{col18}" />
    <color code="bg.level2"      name="Background level 2" value="{col15}" />
    <color code="bg.level3"      name="Background level 3" value="{col12}" />
    <color code="border"         name="Border"             value="{col02}" />
    <color code="text"           name="Text color"         value="{col00}" />
    <color code="text.sel"       name="Text selected Tab"  value="{col00}" />
    <color code="text.unsel"     name="Text unselected Tab" value="{col26}" />
  </definitions>
</definitions>

<definitions code="header" name="Controls: Header">
  <definitions code="color" name="Colortable">
    <color code="border"         name="Border"             value="{col03}" />
    <color code="text.caption"    name="Text caption"       value="{col20}" />
    <color code="text.detail"     name="Text detail"        value="{col03}" />
  </definitions>
</definitions>

<definitions code="help" name="Controls: Help">
  <definitions code="color" name="Colortable">
    <color code="bg"             name="Background"         value="{col25}" />
    <color code="border"         name="Border"             value="{col00}" />
    <color code="text"           name="Text"                 value="{col00}" />
  </definitions>
</definitions>

<!-- ===== -->
<!-- Button Style Definitions -->
<!-- ===== -->

<definitions code="button" name="Buttons">
  <definitions code="color" name="Colortable">
    <color code="bg"             name="Background"         value="{col26}" />
    <color code="border"         name="Border"             value="{col02}" />
    <color code="text.active"     name="text"                 value="{col00}" />
    <color code="text.inactive"   name="text inactive"       value="{col14}" />
    <color code="state.active"    name="State activ (id=1)"  value="{col02}" />
    <color code="state.inactive"  name="State inactiv (id=2)" value="{col14}" />
    <color code="state.hover"     name="State hover (id=3)"  value="{col23}" />
    <color code="state.down"     name="State down (id=4)"  value="{col04}" />
  </definitions>

  <!-- ===== -->
  <!-- Use the fontlist task to view all -->
  <!-- available fonts. Note: If you specify a -->
  <!-- font which does not exists on -->
  <!-- your system a default font will be used -->
  <!-- ===== -->

```

```

    <definitions code="font" name="Font list">
        <font code="face"
            family="HelveticaNeue LT 57 Cn"
            style="bold"
            size="12"/>
    </definitions>
</environment>

<!-- ===== -->
<!-- Button Resources -->
<!-- ===== -->

<resources destdir="images/buttons/def">

    <!-- ===== -->
    <!-- German resources -->
    <!-- ===== -->
    <resources destdir="de">
        <button name="About" label="über" />
        <button name="Back" label="zurück" />
        <button name="Continue" label="weiter" />
        <button name="Copy" label="kopieren" />
        <button name="Close" label="schliessen" />
        <button name="Execute" label="ausführen" />
        <button name="Import" label="importieren" />
        <button name="Logon" label="anmelden" />
        <button name="Save" label="speichern" />
        <button name="Search" label="suchen" />
        <button name="Open" label="öffnen" />
    </resources>

    <!-- ===== -->
    <!-- English resources -->
    <!-- ===== -->
    <resources destdir="en">
        <button name="About" label="about" />
        <button name="Back" label="back" />
        <button name="Continue" label="continue" />
        <button name="Copy" label="copy" />
        <button name="Close" label="close" />
        <button name="Print" label="print" />
        <button name="Import" label="import" />
        <button name="Logon" label="logon" />
        <button name="Save" label="save" />
        <button name="Search" label="search" />
        <button name="Open" label="open" />
    </resources>
</resources>

<!-- ===== -->
<!-- Cascading Stylesheet Resources -->
<!-- ===== -->

<resources destdir="fw/def/docs">
    <stylebook name="stylebook" title="Color palette DefaultPainter (Def)" />
</resources>

<resources destdir="fw/def/styles">
    <textfile name="default.css" template="templates/def/default.tem" />

    <textfile name="controls.css" template="templates/def/controls.tem" />
    <textfile name="forms.css" template="templates/def/forms.tem" />
    <textfile name="menu.css" template="templates/def/menu.tem" />
</resources>

<!-- ===== -->
<!-- Java Painter Classes -->
<!-- ===== -->

<resources destdir="WEB-INF/source">
    <javacode package="com.cc.framework.ui.painter.def" />
</resources>
</resource-factory>

```


2.2 Document Type Definition (DTD)

The document type definition (DTD) specifies the structure of a resource xml file:

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT resource-factory (property*, environment?, resources*)>
<!ATTLIST resource-factory
    version          CDATA #REQUIRED
>

<!--
Properties can be defined everywhere.
The value of a property can be accessed with ${propertyname}
code.....: Unique property name
value.....: The property value
-->
<!ELEMENT property EMPTY>
<!ATTLIST property
    name             CDATA #REQUIRED
    value            CDATA #REQUIRED
>

<!-- ===== -->
<!-- ==           == -->
<!-- == Environment Definitions == -->
<!-- ==           == -->
<!-- ===== -->

<!--
An Environment spans up a hierarchical namespace
with definitions. A single definition can be accessed
by its unique path name.
-->
<!ELEMENT environment ((property|definitions)*)>

<!--
Creates a directory in the environment namespace.
Definitions can be defined recursively.
code.....: Segment of the namespace path
           = name of the directory
name.....: Optional user friendly name
           of the directory
-->
<!ELEMENT definitions ((property|definitions|font|color)*)>
<!ATTLIST definitions
    code             CDATA #REQUIRED
    name             CDATA #IMPLIED
>

<!--
Creates a font definition in the painterfactory environment
code.....: The unique identifier for this definition
name.....: Optional user friendly name
family.....: The font family name
style.....: Font style
size.....: Font size

You can access the font properties in Expressions:
Example:  ${myfont.family} or ${myfont.size}

You can use expressions to make calculations
Example:  ${(int) myfont.size + 5}
-->
<!ELEMENT font EMPTY>
<!ATTLIST font
    code             CDATA #REQUIRED
    name             CDATA #IMPLIED
    family           CDATA #REQUIRED
    style            (bold|plain|italic|bolditalic) #IMPLIED
    size            CDATA #REQUIRED
>

<!--
Creates a color definition in the painterfactory environment
code.....: The unique identifier for this definition
```

```

name.....: Optional user friendly name
value.....: The color value (name or #RRGGBB)
-->
<!ELEMENT color EMPTY>
<!ATTLIST color
    code          CDATA #REQUIRED
    name          CDATA #IMPLIED
    value         CDATA #REQUIRED
>

<!-- ===== -->
<!-- ==           == -->
<!-- == Resource Definitions == -->
<!-- ==           == -->
<!-- ===== -->

<!--
Creates a output directory for nested resources.
Resources-Tags can be nested
destdir.....: Name of the output directory
overwrite....: A value of false directs the resource
                factory not to overwrite existing
                files for the nested resources
                (default: overwrite="true")
-->
<!ELEMENT resources      ((property|resources|menu|button|textfile|stylebook|
                           javasource|bundle)*)>
<!ATTLIST resources
    destdir          CDATA #REQUIRED
    overwrite        (true | false) #IMPLIED
>

<!--
Creates a button resource
destdir.....: Name of the output directory
name.....: The name of the resulting image files
              (a state number will be appended)
label.....: The button label text
overwrite....: A value of false directs the resource
                factory not to overwrite existing
                files
                (default: overwrite="true")
generate.....: Enables or disables the generation
                of this resource
                (default: generate="true")
-->
<!ELEMENT button EMPTY>
<!ATTLIST button
    destdir          CDATA #IMPLIED
    name             CDATA #REQUIRED
    label            CDATA #REQUIRED
    overwrite        (true | false) #IMPLIED
    generate         (true | false) #IMPLIED
>

<!--
Creates a menu button resource
destdir.....: Name of the output directory
name.....: The name of the resulting image files
              (a state number will be appended)
label.....: The label text
overwrite....: A value of false directs the resource
                factory not to overwrite existing
                files
                (default: overwrite="true")
generate.....: Enables or disables the generation
                of this resource
                (default: generate="true")
-->
<!ELEMENT menu EMPTY>
<!ATTLIST menu
    destdir          CDATA #IMPLIED
    name             CDATA #REQUIRED
    label            CDATA #REQUIRED
    overwrite        (true | false) #IMPLIED
    generate         (true | false) #IMPLIED
>

```

```

<!--
Transforms a template into a text file
destdir.....: Name of the output directory
name.....: The name of the resulting text files
template.....: The name of the template file
                (the template file can access any
                definition or property in the environment)
overwrite.....: A value of false directs the resource
                factory not to overwrite existing
                files
                (default: overwrite="true")
generate.....: Enables or disables the generation
                of this resource
                (default: generate="true")
-->
<!ELEMENT textfile EMPTY>
<!ATTLIST textfile
    destdir          CDATA #IMPLIED
    name             CDATA #REQUIRED
    template         CDATA #REQUIRED
    overwrite        (true | false) #IMPLIED
    generate         (true | false) #IMPLIED
>

<!--
Creates a HTML Stylebook
destdir.....: Name of the output directory
name.....: The name of the resulting stylebook
                files
title.....: The stylebook Title
overwrite.....: A value of false directs the resource
                factory not to overwrite existing
                files
                (default: overwrite="true")
generate.....: Enables or disables the generation
                of this resource
                (default: generate="true")
-->
<!ELEMENT stylebook EMPTY>
<!ATTLIST stylebook
    destdir          CDATA #IMPLIED
    name             CDATA #REQUIRED
    title            CDATA #IMPLIED
    overwrite        (true | false) #IMPLIED
    generate         (true | false) #IMPLIED
>

<!--
Creates a predefined Java File
destdir.....: Name of the output directory
type.....: The class to create
name.....: The Java classname
package.....: The java package
title.....: The stylebook Title
overwrite.....: A value of false directs the resource
                factory not to overwrite existing
                files
                (default: overwrite="true")
generate.....: Enables or disables the generation
                of this resource
                (default: generate="true")
-->
<!ELEMENT javasource EMPTY>
<!ATTLIST javasource
    destdir          CDATA #IMPLIED
    type             (colorpalette) #IMPLIED
    name             CDATA #REQUIRED
    package          CDATA #REQUIRED
    overwrite        (true | false) #IMPLIED
    generate         (true | false) #IMPLIED
>

<!--
Creates a property resource bundle
destdir.....: Name of the output directory
name.....: The Java classname

```

```
package.....: The java package
defaultlocale.....: The default locale
overwrite.....: A value of false directs the resource
                 factory not to overwrite existing
                 files
                 (default: overwrite="true")
generate.....: Enables or disables the generation
                 of this resource
                 (default: generate="true")
-->
<!ELEMENT bundle (resourcekey*)>
<!ATTLIST bundle
  package          CDATA #REQUIRED
  name             CDATA #REQUIRED
  defaultlocale    CDATA #IMPLIED
  overwrite        (true | false) #IMPLIED
  generate         (true | false) #IMPLIED
>

<!--
creates a resource key within a property resource bundle.
Resource keys can be nested
code.....: The key name
public.....: A value of true directs the resource
             factory to put this key in the Java
             interface that will be created
-->
<!ELEMENT resourcekey ((resourcekey|value)*)>
<!ATTLIST resourcekey
  code            CDATA #REQUIRED
  public         (true|false) #IMPLIED
>

<!--
Property Value
locale.....: The locale
-->
<!ELEMENT value (#PCDATA)>
<!ATTLIST value
  locale         CDATA #REQUIRED
>
```

2.3 The environment

The environment sets up a hierarchical name space with objects like fonts, colors and simple properties (key value pairs). Each element within the name space can be addresses by a unique path.

The following examples defines a font:

```
<resource-factory version="1.1">
  <environment>
    <definitions code="list" name="Buttons">
      <definitions code="font" name="Font list">
        <font code="myFont"
          family="HelveticaNeue LT 57 Cn"
          style="bold"
          size="12"/>
      </definitions>
    </definitions>
  </environment>
</resource-factory>
```

2.3.1 Path name

The font can then be used like:

`#{list.font.myFont}` → „HelveticaNeue LT 57 Cn“

Also the single properties can be used if the property Name is specified like:

`#{list.font.myFont.family}` → „HelveticaNeue LT 57 Cn“

`#{list.font.myFont.style}` → „bold“

`#{list.font.myFont.size}` → „12“

2.3.2 Expressions

The `#{}` macro allows to specify expressions like:

`#{(int) list.font.myFont.size * 2}` → 24 (as Integer!)

`#{(list.font.myFont.size == „12“) ? „14“ : „17“}` → „14“

The following syntax can be used:

```
type_name
  : "char"
  | "string"
  | "int"
  | "long"
  | "double"
  | "boolean"

idetifier
  : IDENTIFIER

unary_operator
  : '+' | '-' | '~' | '!'

primary_expr
  : IDENTIFIER
  | LITERAL
  | '(' expr_list ')'

postfix_expr
  : primary_expr
implemented
  | postfix_expr '(' ')'
  | postfix_expr '(' expr_list ')'

unary_expr
  : postfix_expr
  | unary_operator cast_expr
```

```
cast_expr
: unary_expr
| '(' type_name ')' cast_expr

multiplicative_expr
: cast_expr
| multiplicative_expr '*' cast_expr
| multiplicative_expr '/' cast_expr
| multiplicative_expr '%' cast_expr

additive_expr
: multiplicative_expr
| additive_expr '+' multiplicative_expr
| additive_expr '-' multiplicative_expr

shift_expr
: additive_expr
| shift_expr LEFT_OP additive_expr
| shift_expr RIGHT_OP additive_expr

relational_expr
: shift_expr
| relational_expr '<' shift_expr
| relational_expr '>' shift_expr
| relational_expr LE_OP shift_expr
| relational_expr GE_OP shift_expr

equality_expr
: relational_expr
| equality_expr EQ_OP relational_expr
| equality_expr NE_OP relational_expr

and_expr
: equality_expr
| and_expr '&' equality_expr

exclusive_or_expr
: and_expr
| exclusive_or_expr '^' and_expr

inclusive_or_expr
: exclusive_or_expr
| inclusive_or_expr '|' exclusive_or_expr

logical_and_expr
: inclusive_or_expr
| logical_and_expr AND_OP inclusive_or_expr

logical_or_expr
: logical_and_expr
| logical_or_expr OR_OP logical_and_expr

conditional_expr
: logical_or_expr
| logical_or_expr '?' expr_list ':' conditional_expr

expr_list
: conditional_expr
| expr_list ',' conditional_expr

constant_expr
: conditional_expr

...
```

2.3.3 Default identifier

The following default identifier exist:

build.factory	Version of the ResourceFactory
build.dir	Name of the build directory
build.file	Name of the xml file
build.date	The current date
painter	The selected painter
painter.name	User frindly name for the painter
user.name	User name

To access Java System Environments variables use `System.getProperties()`.

You can use the identifiers to generate a head section for the template files.

Template example:

```
/* =====
** THIS FILE IS GENERATED WITH
** THE CC-RESOURCEFACTORY TOOL.
**
** USER.....: ${user.name}
** PAINTER.....: ${painter}
** BUILD-FACTORY..: ${build.factory}
** BUILD-DATE.....: ${build.date}
** BUILD-DIRECTORY: ${build.dir}
** BUILD-FILE.....: ${build.file}
**
** DO NOT MODIFY!
** =====
*/
```

Output:

```
/* =====
** THIS FILE IS GENERATED WITH
** THE CC-RESOURCEFACTORY TOOL.
**
** USER.....: P001001
** PAINTER.....: def
** BUILD-FACTORY..: v1.4.040
** BUILD-DATE.....: Sun Oct 24 18:39:26 CEST 2004
** BUILD-DIRECTORY: C:\eclipse\workspace\cc-framework
** BUILD-FILE.....: C:\eclipse\workspace\cc-framework\build\resources\resources.xml
**
** DO NOT MODIFY!
** =====
*/
```

2.4 Elemente der Ressourcen XML Datei

2.4.1 <environment> Element

Defines an environment section within the resource definition. The environment can be split into several xml files.

2.4.2 <property> Element

Defines a key/values pair. A value can be stored under the key and referenced later. Properties are currently used for the global definition of color values. A property is accessed with the following syntax: **`${propertyname}`**. Properties can also be used within character strings.

Attribut	Description
name	Name of the properties
value	Value

2.4.3 <definitions> Element

The <definitions> element serves for the grouping and especially for the construction of a hierarchical key. The element can be nested at will.

Attribut	Description
code	Part of the keyTeil
name	Display name as in the StyleBook

A key is formed across the declaration hierarchy. Example:

```
<property name="coll3" value="#c7003c"/>

<definitions code="error" name="Forms: Error form">
  <definitions code="color" name="Colortable">
    <color code="bg.header" name="Background caption" value="${coll3}"/>
  </definitions>
</definitions>
```

Generates the key: `error.color.bg.header` and assigns to it the color value `#c7003c`.

Alternative depiction variants:

A) With one group:

```
<definitions code="error.color" name="Colortable">
  <color code="bg.header" name="Background caption" value="${coll3}"/>
</definitions>
```

B) Without a group:

```
<color code="error.color.bg.header" name="Background caption" value="${coll3}"/>
```

The keys that have already been defined are required during the generation of the StyleSheets. The StyleSheets are generated with the help of a template file, in which a reference is made to the key. During the generation of the StyleSheet, the key is replaced by the color value assigned to it. Using this mechanism, the existing templates can be expanded if required, or enhanced with new templates.

Remark: The grouping is used to influence the construction of the StyleBook. If the grouping that is offered by default is changed, the structure in the generated HTML file also changes.

2.4.4 <color> Element

The element is used to define a color value.

Attribut	Description
code	Unique identifier
name	display name
value	value for the color. The color name or a hex value #RRGGBB

Example:

```
<color code="bg.field" name="Background field" value="{col18}"/>
```

2.4.5 Element

The element is used to define the font family and the font size with which the buttons are to be generated.

Attribut	Description
family	Font family
style	Font style: <ul style="list-style-type: none"> - plain - italic - bold - bolditalic
size	Font size (points)

Example:

```
<font family="HelveticaNeue LT 57 Cn" style="bold" size="12"/>
```

2.4.6 <resources> Element

The <resources> element is used for the logical grouping of resources in directories.

Attribut	Description
destdir	Relative directory in which the resources are to be saved. The main directory will be specified in the file build.properties.
overwrite	Overwrite existing files

The Resource element can be nested to any desired depth, with the directory names being appended to one another across the declaration hierarchy. In this manner, it is possible to depict any desired directory structure for storage of the resources.

```
<resources destdir="images/buttons">
  <resources destdir="de">
  </resources>
  <resources destdir="en">
  </resources>
</resources>
```

Directory: images/buttons/de
images/buttons/en

2.4.7 <stylebook> Element

Generates an HTML file with all the color values for the Painter. The file is saved in the sub-directory specified in the <resource> element.

Attribut	Description
destdir	Relative directory in which the resources are to be saved. The main directory will be specified in the file build.properties.
name	Name of the file to be generated
title	Title of the Stylebooks
overwrite	Overwrite existing files
generate	Specifies if the element should be generated

2.4.8 <textfile> Element

Is used for the specification of a file to be generated.

A template file is a simple text file which uses some **`\${}` macros** to access definitions whin the environnement.

Example for a template file:













```
.mystyle1 {
    font-weight: bold;
    background-color: ${panel.color.bg.header};
    padding-bottom: 2px;
    border-top: 1px solid ${panel.color.border.header};
    border-bottom: 1px solid ${panel.color.border.header};
    color: ${panel.color.text.header};
}

.mystyle2 {
    font-family: ${panel.font};
    font-size: ${int} panel.font.size + 2}pt;
    background-color: ${panel.color.bg.body};
    border: 2px solid ${panel.color.border.body};
}
```

Attribut	Description
destdir	Relative directory in which the resources are to be saved. The main directory will be specified in the file build.properties.
name	Name of the file to be generated
template	The template to be used for the generation of the file. In the case of a StyleSheet, the color values are used in the templates for the keys defined above.
overwrite	Overwrite existing files
generate	Specifies if the element should be generated





2.4.9 <button> Element

Defines a button resource.

Attribut	Meaning												
name	<p>Name of the image under which the button is generated. Buttons always start with the string btn and have a number at the end that shows the state of the button. The following apply:</p> <table border="0"> <tr> <td>1 = Active</td> <td></td> <td>btnExecute1.gif</td> </tr> <tr> <td>2 = Inactiv</td> <td></td> <td>btnExecute2.gif</td> </tr> <tr> <td>3 = Actively selected</td> <td></td> <td>btnExecute3.gif</td> </tr> <tr> <td>4 = Pressed</td> <td></td> <td>btnExecute4.gif</td> </tr> </table> <p>The “btn” prefix and the status number are <u>not</u>, however, specified in the name-attribute. Thus, the following setting is sufficient for the example shown: name=“Execute“</p>	1 = Active		btnExecute1.gif	2 = Inactiv		btnExecute2.gif	3 = Actively selected		btnExecute3.gif	4 = Pressed		btnExecute4.gif
1 = Active		btnExecute1.gif											
2 = Inactiv		btnExecute2.gif											
3 = Actively selected		btnExecute3.gif											
4 = Pressed		btnExecute4.gif											
label	Label of the button												
overwrite	Overwrite existing files												
generate	Specifies if the element should be generated												

The button painter needs informations about the font and the colors. Therefore the following keys will be used.

Keys used by the **Def** Button painter:

Key	Definition type	Description
button.font.face		Font used for the label
button.color.border	<color>	Border color
button.color.bg	<color>	Background color
button.color.text.active	<color>	Color used by the active state for the label
button.color.text.inactive	<color>	Color used by the inactive state for the label
button.color.state.active	<color>	Color for the image which indicates the active state 
button.color.state.inactive	<color>	Color for the image which indicates the inactive state 
button.color.state.hover	<color>	Color for the image which indicates the hover state 
button.color.state.down	<color>	Color for the image which indicates the pressed state 
















Keys used by the **Def2** Button painter:

Key	Definition type	Description
button.font.face		Font used for the label
button.color.border	<color>	Border color
button.color.border.inactive	<color>	Inactiv border color
button.color.bg	<color>	Background color
button.color.text.active	<color>	Color used by the active state for the label
button.color.text.inactive	<color>	Color used by the inactive state for the label
button.color.state.active	<color>	Color for the image which indicates the active state
button.color.state.inactive	<color>	Color for the image which indicates the inactive state
button.color.state.hover	<color>	Color for the image which indicates the hover state

button.color.state.down	<color>	Color for the image which indicates the pressed state
button.color.pattern.active	<color>	Color for the pattern which is used by the active state
button.color.pattern.inactive	<color>	Color for the pattern which is used by the inactive state
button.color.pattern.hover	<color>	Color for the pattern which is used by the hover state
button.color.pattern.down	<color>	Color for the pattern which is used by the pressed state

2.4.10 <menu> Element

Defines a menu resource.

Attribut	Description															
destdir	Relative directory in which the resources are to be saved. The main directory will be specified in the file build.properties.															
name	<p>Name of the image under which the menu button is generated. Menu buttons always start with the string btn and have a number at the end that shows the state of the menu button. The following apply:</p> <table border="0"> <tr> <td>1 = Active</td> <td></td> <td>btnLogout1.gif</td> </tr> <tr> <td>2 = Inactiv</td> <td></td> <td>btnLogout2.gif</td> </tr> <tr> <td>3 = Active Hover</td> <td></td> <td>btnLogout3.gif</td> </tr> <tr> <td>5 = Actively selected</td> <td></td> <td>btnLogout5.gif</td> </tr> <tr> <td>6 = Actively selected Effekt</td> <td></td> <td>btnLogout6.gif</td> </tr> </table> <p>The "btn" prefix and the status number are <u>not</u>, however, specified in the name-attribute. Thus, the following setting is sufficient for the example shown: name="Lgout"</p>	1 = Active		btnLogout1.gif	2 = Inactiv		btnLogout2.gif	3 = Active Hover		btnLogout3.gif	5 = Actively selected		btnLogout5.gif	6 = Actively selected Effekt		btnLogout6.gif
1 = Active		btnLogout1.gif														
2 = Inactiv		btnLogout2.gif														
3 = Active Hover		btnLogout3.gif														
5 = Actively selected		btnLogout5.gif														
6 = Actively selected Effekt		btnLogout6.gif														
label	Label of the button															
overwrite	Overwrite existing files															
generate	Specifies if the element should be generated															

The menu painter needs informations about the font and the colors. Therefore the following keys will be used.

Keys for the **Def** Menü painter:

Key	Definition type	Description
menu.font.face		Font used for the menu caption
menu.color.bg	<color>	Background color
menu.color.text.active	<color>	Color used by the active state for the label
menu.color.text.inactive	<color>	Color used by the inactive state for the label
menu.color.glow	<color>	Color used by the glow effect
menu.color.shadow	<color>	Color used by the shadow
menu.shadow.x	Int-property	X-offset for the shadow
menu.shadow.y	Int-property	Y-offset for the shadow
menu.backlit.radius	Int-property	Radius backlighting

Keys for the Def2 Menü painter::

Key	Definition type	Description
menu.font.face		Font used for the menu caption
menu.color.bg	<color>	Background color
menu.color.text.active	<color>	Color used by the active state for the label
menu.color.text.inactive	<color>	Color used by the inactive state for the label
menu.color.glow	<color>	Color used by the glow effect
menu.color.shadow	<color>	Color used by the shadow
menu.shadow.x	Int-property	X-offset for the shadow
menu.shadow.y	Int-property	Y-offset for the shadow
menu.backlit.radius	Int-property	Radius backlighting

2.4.11 <bundle> Element

The <bundle> element is used for the definition of localized PropertyResource Bundles.

Attribut	Description
destdir	Relative directory in which the resources are to be saved. The main directory will be specified in the file build.properties.
package	Name of the java package in which the resources (*.properties files) are to be saved.
name	Name of the resource bundle
defaultlocale	Specifies the default locale. For the default locale also a *.propertiesfile is generated without the locale suffix.
overwrite	Overwrite existing files
generate	Specifies if the element should be generated

The <bundle> element generates:

- Localized `<package>\<name>_<locale>.properties` files.
- For the default locale an additional `<package>\<name>.properties` file
- A java interface class named `<package>.<name>.java` which includes all keys used within the Property Resource Bundle

Example:

```
<resource-factory version="1.2">
  <resources destdir="source">
    <bundle
      package="com.cc.bulletin" name="ApplicationResources" defaultlocale="en">

      <resourcekey code="errors">
        <resourcekey code="required">
          <value locale="de">Eingabe erwarted in Feld: {0}</value>
          <value locale="en">{0} is required.</value>
        </resourcekey>

        <resourcekey code="invalid.retype">
          <value locale="de">Ungültige Kennwort Wiederholung</value>
          <value locale="en">password retype does not match</value>
        </resourcekey>
      </resourcekey>

      <resourcekey code="user" public="false">
        <resourcekey code="list">
          <resourcekey code="caption">
            <value locale="de">Benutzer</value>
          </resourcekey>
        </resourcekey>
      </resourcekey>
    </bundle>
  </resources>
</resource-factory>
```


3 Starting the build process

3.1 Requirements

To use the ResourceFactory tool, the shipped jar files must be added to the classpath

- **cc-resfactory-1.4.055.jar** The ResourceFactory.
- **glf.jar** The graphics layer framework from SUN, which is used to generate complex images.

If the ant tasks should be used also the following jar is needed:

- **ant.jar** The Apache Ant Build System

3.2 Using the comand line

The generating of resources is started by calling the batch file **build.bat**.

3.3 Using the ant tasks

The ResourceFactory Tool provides several Ant (<http://ant.apache.org>) tasks, which can be used within your own build file. So the generation of the resources can be integrated in the build process of the application.

3.3.1 Setting the build properties

The file build.properties serves to determine the following settings:

Variable	Description
build.home	Specifies the directory in which the resources are stored.
build.src	Name of the file that defines the resources to be generated. Here, the expression <code>\${basedir}</code> specifies the directory in which the ResourceFactory Tool has been installed or unpacked.

Example:

```
# Output Directory
build.home=c:/_build/java

# Source file with the defined resources
build.src=${basedir}/build/resource-test.xml
```

Figure 3: The build.properties file

3.3.2 <fontlist> Task

3.3.2.1 Description

The <fontlist> task shows the fonts installed on the system.

3.3.2.2 Requirements

This task must be registered by the <taskdef> tag:

```
<taskdef name="resourcefactory" classname="com.cc.resourcefactory.ant.FontListTask"/>
```

3.3.2.3 Attribute

keine

3.3.2.4 Example

```

<target name="fontlist" description="Shows all available fonts">
    <taskdef
        name="fontlist"
        classname="com.cc.resourcefactory.ant.FontListTask" />
    <fontlist/>
</target>

```

3.3.3 <resourcefactory> Task

3.3.3.1 Description

3.3.3.2 Requirements

This task must be registered by the <taskdef> tag:

```
<taskdef name="resourcefactory" classname="com.cc.resourcefactory.ant.FontListTask"/>
```

3.3.3.3 Attribute

Attribut	Description	Required
destdir	Specifies the directory in which the resources are stored.	Yes
srcfile	Specifies the Ressourcen XML file To specify several resource files the <fileset> can be used. The <fileset> can be nested.	Yes or <fileset>
severity	The ResourceFactory Tool checks the consistency of the xml files. The attribute specifies which informations should be displayed Possible values: <ul style="list-style-type: none"> • none • question • information • warning • error • fatal 	No Default: question
generate	Specifies if the resource should be generated Possible values: <ul style="list-style-type: none"> • true • false 	No Default: true
painter	The painter which should be used to generate the	Yes

resources. Possible values:

- def
- def2

3.3.3.4 Example

```
<target name="build-res" description="Creates the resources">
  <taskdef
    name="resourcefactory"
    classname="com.cc.resourcefactory.ant.ResourceFactoryTask" />
  <resourcefactory
    destdir="${build.home}"
    painter="def"
    generate="true">
    <fileset dir="${basedir}/demo/input">
      <include name="**/*.xml" />
      <exclude name="**/*-def2.xml" />
    </fileset>
    <fileset dir="${basedir}/demo/test">
      <include name="**/*.xml" />
      <exclude name="**/*-def2.xml" />
    </fileset>
  </resourcefactory>
</target>
```

3.3.4 <version> Task

3.3.4.1 Description

The <version> task generates the java class which holds the version informations for the application.

3.3.4.2 Requirements

This task must be registered by the <taskdef> tag:

```
<taskdef name="resourcefactory" classname="com.cc.resourcefactory.ant.FontListTask"/>
```

3.3.4.3 Attribute

Attribut	Description	Required
destdir	Specifies the directory in which the resources are stored.	Yes
packagename	packagename for the java class	Yes
classname	The class name for the java versions class	Yes
majorversion	Majorversion	
minorversion	Minorversion	
buildnumber	Buildnummer	
productname	Productname	
productextension	Extension to the product name	
systemtype	Systemtype Valid attributes: <ul style="list-style-type: none"> • production 	

	<ul style="list-style-type: none">• test• debug• develop• demo	
vendor	Vendor	
vendorsite	Vendor Website	

3.3.4.4 Example

```
<target name="createVersion" description="creates the version class">
  <taskdef
    name="version"
    classname="com.cc.resourcefactory.ant.VersionTask" />
  <version
    destdir="${build.src}"
    packagename="com.elster.mis"
    classname="Version"
    majorversion="${project.major}"
    minorversion="${project.minor}"
    buildnumber="${project.build}"
    productname="${project.name}"
    productextension="for j2ee"
    systemtype="production"
    vendor="SCC Informationssysteme GmbH"
    vendorsite="www.scc-gmbh.com" />
</target>
```

4 Outputting the available Fonts

The fonts installed on the system can be output on the screen by calling the batch file [fontlist.bat](#). The font families listed can be used for labeling buttons.

```
<!-- Define the environment for the "def"-Painter -->  
<environment painter="def">  
  <buttons>  
    <font family="HelveticaNeue LT 57 Cn" style="bold" size="12"/>  
  </buttons>  
</environment>
```

5 Support

We would be happy to be of service if you have any questions or problems. Please use our Service Form on our homepage for your queries. We shall endeavor to answer your queries as quickly as possible