

Common-Controls Tools ResourceFactory

Version 1.4.055 - Stand: 31. Oktober 2004

Herausgeber:

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.

Inhaltsverzeichnis

1	Einleitung	1
2	Beschreibung der zu erzeugenden Ressourcen	2
2.1	Aufbau einer Ressourcen XML Datei	2
2.2	Document Type Definition (DTD)	6
2.3	Das Environment.....	10
2.3.1	Pfadnamen	10
2.3.2	Ausdrücke.....	10
2.3.3	Vordefinierte Bezeichner	12
2.4	Elemente der Ressourcen XML Datei.....	13
2.4.1	Definitionselement <environment>.....	13
2.4.2	Definitionselement <property>	13
2.4.3	Definitionselement <definitions>	13
2.4.4	Definitionselement	14
2.4.5	Definitionselement <color>.....	14
2.4.6	Ressourcen Element <resources>.....	14
2.4.7	Ressourcen Element <stylebook>.....	15
2.4.8	Ressourcen Element <textfile>	15
2.4.9	Ressourcen Element <button>	16
2.4.10	Ressourcen Element <menu>	17
2.4.11	Ressourcen Element <bundle>	18
2.4.12	Ressourcen Element <resourcekey>.....	19
2.4.13	Ressourcen Element <value>.....	19
3	Starten des Build Vorganges	20
3.1	Voraussetzungen	20
3.2	Start über die Kommandozeile.....	20
3.3	Verwendung der Ant-Tasks.....	20
3.3.1	Setzen der Build Properties.....	20
3.3.2	Der <fontlist> Task	21
3.3.3	Der <resourcefactory> Task.....	21
3.3.4	Der <version> Task	22
4	Ausgabe der verfügbaren Schriftarten	24
5	Support	25
6	Abkürzungsverzeichnis	Fehler! Textmarke nicht definiert.

1 Einleitung

Das ResourceFactory Tool dient zur automatischen Erzeugung von Ressourcen für das Common Controls Framework. Die gegenwärtige Version unterstützt (vgl. Abbildung 1):

- Die automatische Erstellung von unterschiedlichen Bildern (für Buttons, Menüs, etc.) für den DefaultPainter (Def) und DefaultPainter2 (Def2)
- Die Erstellung von Cascading StyleSheet Dateien zur Anpassung des Farbschemas des DefaultPainters an einen individuellen StyleSheet.
- StyleBook → Beschreibung eines StyleSheets in einer HTML Datei.
- Die Erzeugung von lokalisierten PropertyResourceBundle Dateien sowie die Erzeugung einer Interface Klasse mit symbolischen Konstanten für ausgewählte Property Schlüssel
- Die Erstellung der Java ColorPalette Klasse für eine von der `DefResourceMap` abgeleiteten Ressourcen Tabelle.
- Erstellung/Management von Resource Bundels

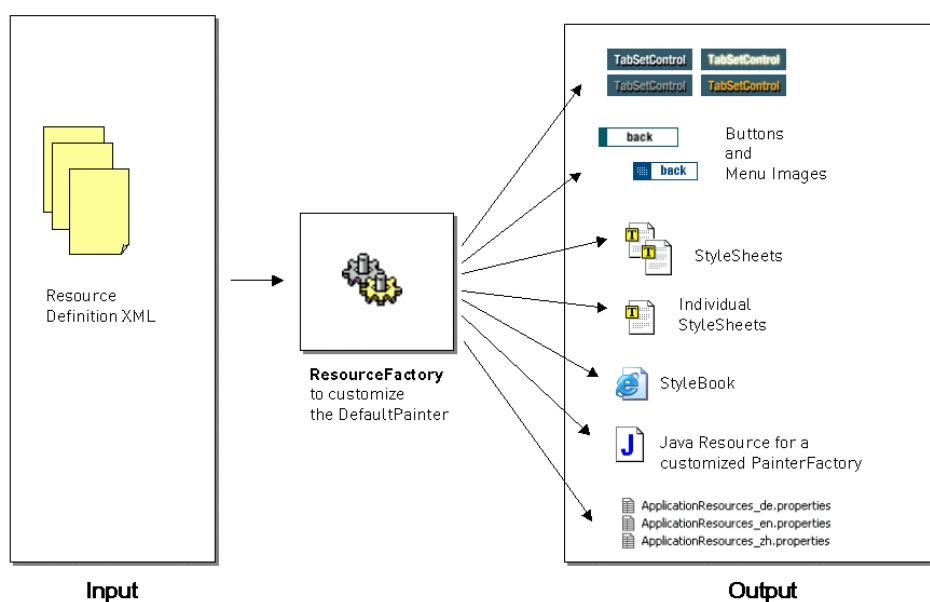


Abbildung 1: ResourceFactory

Für den Einsatz des Tools sind die folgenden Schritte notwendig.

1. Beschreibung der Ressourcen in einer oder mehreren XML-Dateien
2. Setzen der Build Properties (Ein- und Ausgabeverzeichnis)
3. Starten des Buildvorganges

Wichtige Anmerkung:

Bei der Generierung von Grafiken mit Texten haben sich gravierende Qualitätsunterschiede je nach der eingesetzten JDK Version gezeigt. Um möglichst gute Ergebnisse zu erzielen, sollte das JDK Version 1.4.1 verwendet werden.

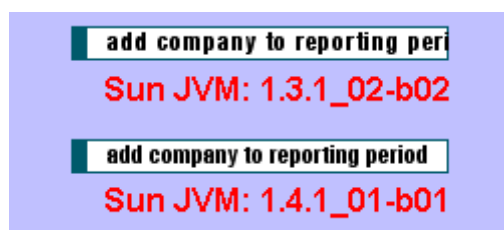


Abbildung 2: Auswirkungen der JDK Version bei der Erzeugung von Schaltflächen

2 Beschreibung der zu erzeugenden Ressourcen

Zu Beginn müssen die zu erzeugenden Ressourcen in einer oder mehreren XML-Dateien definiert werden. Hierzu kann eine neue Datei angelegt oder die mitgelieferte Datei kopiert und angepasst werden.

Die Ressourcen XML Dateien dienen dem ResourceFactory Tool als Eingabe.

2.1 Aufbau einer Ressourcen XML Datei

Die nachfolgende Abbildung zeigt einen exemplarischen Aufbau einer Ressourcen Datei. Innerhalb der Datei werden die Farbwerte für den StyleSheet des DefaultPainters und die Buttons für eine deutsche und englische Sprachversion definiert. Die Ressourcen werden bei der Generierung in verschiedenen Unterverzeichnissen abgelegt.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE resource-factory PUBLIC
  "-//common-controls//DTD ResourceFactory 1.2//EN"
  "http://www.common-controls.com/dtds/resource-factory_1_2.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}"/>
      </definitions>
    </definitions>
  </environment>
</resource-factory>
```

```

    <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}" />
    <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}" />
  </definitions>
</definitions>

```

```

    <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}" />
    <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">
    <javasource package="com.cc.framework.ui.painter.def" />
</resources>

</resource-factory>

```


2.2 Document Type Definition (DTD)

Mit der Document Type Definition (DTD) wird die Struktur einer Ressourcen XML Datei festgelegt:

```
<?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 hirachical 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 Das Environment

Das Environment ist ein hierarchischer Namensraum mit Definitionsobjekten (Farben und Schriften) und einfachen Properties (Schlüssel - Wert Paare). Jedes Element des Environments ist über seinen Pfadnamen eindeutig adressierbar. Der Zugriff auf die Properties eines Definitionsobjektes ist über die Punkt Notation möglich.

Im folgenden Beispiel wurde eine Schrift wie folgt definiert:

```
<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 Pfadnamen

Das Schrift Objekt ist dann unter dem folgenden Pfadnamen erreichbar:

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

Der Zugriff auf einzelne Properties ist durch die zusätzliche Angabe des Property Namens möglich:

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

`#{list.font.myFont.style}` → „bold“

`#{list.font.myFont.size}` → „12“

2.3.2 Ausdrücke

Innerhalb der `#{}` Markierung können beliebige Ausdrücke angegeben werden:

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

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

Es gelten die folgenden Syntax Regeln:

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

identifier
  : 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 Vordefinierte Bezeichner

Die folgenden Bezeichner sind vordefiniert :

build.factory	Versionsnummer der ResourceFactory
build.dir	Das Ausgabeverzeichnis
build.file	Der Name der Ressourcen XML Datei
build.date	Das aktuelle Tagesdatum
painter	Der eingestellte Painter
painter.name	Benutzerfreundlicher Name des Painters
user.name	Die Benutzerkennung des Anwenders

Zusätzlich kann auf alle Einträge des Java System Environments (`System.getProperties()`) zugegriffen werden.

Mit diesen Informationen kann innerhalb eines Templates beispielsweise ein Header für die Ausgabedatei erzeugt werden.

Beispiel Template:

```
/* =====
** 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!
** =====
*/
```

Ausgabe:

```
/* =====
** 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 Definitionselement <environment>

Leitet den Environment block der Ressourcen Definition ein. Das Environment kann auf mehrere XML Dateien verteilt werden.

2.4.2 Definitionselement <property>

Definiert ein Schlüssel/Werte Paar. Unter dem Schlüssel kann ein Wert abgelegt und später referenziert werden. Properties werden derzeit für die globale Definition von Farbwerten verwendet. Der Zugriff auf ein Property erfolgt über die folgende Syntax: **#{pfad.propertyname}**. Properties können dabei auch innerhalb von Zeichenketten verwendet werden.

Attribut	Bedeutung
name	Name des Properties
value	Wert

2.4.3 Definitionselement <definitions>

Das <definitions> Element dient zur Gruppierung und insbesondere dem Aufbau eines hierarchischen Schlüssels. Das Element kann beliebig geschachtelt werden.

Attribut	Bedeutung
code	Teil des Schlüssels
name	Anzeigename zum Beispiel im StyleBook

Ein Schlüssel wird über die Deklarationshierarchie hinweg gebildet. Beispiel:

```
<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>
```

Erzeugt den Schlüssel: `error.color.bg.header` und weist diesem den Farbwert `#c7003c` zu.

Alternative Darstellungsvarianten:

A) Mit nur einer Gruppe:

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

B) Ohne Gruppe:

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

Die bereits definierten Schlüssel werden bei der Generierung der StyleSheets benötigt. Die StyleSheets werden anhand von Templatedatei erstellt, in denen auf die Schlüssel Bezug genommen wird. Bei der Erzeugung des StyleSheets wird der Schlüssel, durch den ihm zugeordneten Farbwert ersetzt. Durch diesen Mechanismus können die bestehenden Templates bei Bedarf erweitert oder um neue Templates ergänzt werden.

Anmerkung: Über die Gruppierung wird Einfluss auf den Aufbau des StyleBooks genommen. Wird die standardmäßig angebotene Gruppierung geändert, ändert sich auch der Aufbau in der erzeugten HTML-Datei.

2.4.4 Definitionselement

Mit dem Element wird die Schriftfamilie und die Schriftgröße festgelegt, mit der die Schaltflächen erstellt werden.

Attribut	Bedeutung
code	Eindeutiger Bezeichner
name	Benutzerfreundlicher Name
family	Schriftfamilie
style	Schrift Stiel. Mögliche Ausprägungen <ul style="list-style-type: none"> - plain - italic - bold - bolditalic
size	Schriftgröße in Points

Beispiel:

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

2.4.5 Definitionselement <color>

Mit dem <color> Element wird ein Farbwert festgelegt

Attribut	Bedeutung
code	Eindeutiger Bezeichner
name	Benutzerfreundlicher Name
value	Farbwert im Format #RRGGBB oder als Frabname angegeben werden

Beispiel:

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

2.4.6 Ressourcen Element <resources>

Das <resources> Element dient zur logischen Gruppierung von Ressourcen in Verzeichnissen.

Attribut	Bedeutung
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen. Das Hauptverzeichnis wird in der Datei build.properties angegeben.
overwrite	Legt für alle untergeordneten Elemente fest ob bestehende Dateien im ausgabeverzeichnis überschrieben werden dürfen. Wird das Attribut nicht angegeben, so dürfen Dateien überschrieben werden

Das Ressource Element lässt sich beliebig tief schachteln, wobei die Verzeichnisnamen immer über die Deklarationshierarchie hinweg aneinandergehängt werden. Damit lässt sich jede gewünschte Verzeichnisstruktur zur Ablage der Ressourcen abbilden.

```
<resources destdir="images/buttons" overwrite="false">
  <resources destdir="de">
  </resources>

  <resources destdir="en">
  </resources>
</resources>
```

Pfade: images/buttons/de
images/buttons/en

2.4.7 Ressourcen Element <stylebook>

Erzeugt eine HTML Datei mit allen Farbwerten für den Painter. Die Datei wird unter dem im <resource> Element spezifizierten Unterverzeichnisses abgelegt.

Attribut	Bedeutung
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen.
name	Name der zu erzeugenden Datei
title	Titel des Stylebooks
overwrite	Gibt an ob eine bestehende Datei überschrieben werden soll
generate	Gibt an ob das Element überhaupt eine Ausgabedatei erzeugen soll

2.4.8 Ressourcen Element <textfile>

Mit Hilfe des <textfile> Elementes kann eine Template-Datei in eine Ausgabedatei übersetzt werden.

Bei der Template-Datei handelt es sich um eine einfach Text Dateien, welche mit Hilfe von **`\${}** **Markierungen** auf die Definitionen im Environment zugreift.

Beispiel für eine Template-Datei:

```
.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	Bedeutung
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen.

name	Name der zu erzeugenden Datei
template	Das zu verwendende Template.
overwrite	Gibt an ob eine bestehende Datei überschrieben werden soll
generate	Gibt an ob das Element überhaupt eine Ausgabedatei erzeugen soll

2.4.9 Ressourcen Element <button>

Definiert eine Button-Ressource

Attribut	Bedeutung												
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen.												
name	Name des Images, unter dem der Button generiert wird. Buttons beginnen immer mit der Zeichenfolge btn und erhalten am Ende eine Ziffer, die den Zustand des Buttons kennzeichnet. Dabei gilt: <table border="0" style="margin-left: 20px;"> <tr> <td>1 = normaler Zustand</td> <td></td> <td>btnExecute1.gif</td> </tr> <tr> <td>2 = inaktiver Zustand</td> <td></td> <td>btnExecute2.gif</td> </tr> <tr> <td>3 = Hover Effekt</td> <td></td> <td>btnExecute3.gif</td> </tr> <tr> <td>4 = gedrückter Zustand</td> <td></td> <td>btnExecute4.gif</td> </tr> </table> Das „btn“ Präfix und die Statusnummer werden jedoch im name-Attribut <u>nicht</u> angegeben! Für das gezeigte Beispiel reicht also die folgende Einstellung: name="Execute"	1 = normaler Zustand		btnExecute1.gif	2 = inaktiver Zustand		btnExecute2.gif	3 = Hover Effekt		btnExecute3.gif	4 = gedrückter Zustand		btnExecute4.gif
1 = normaler Zustand		btnExecute1.gif											
2 = inaktiver Zustand		btnExecute2.gif											
3 = Hover Effekt		btnExecute3.gif											
4 = gedrückter Zustand		btnExecute4.gif											
label	Beschriftung der Schaltfläche												
overwrite	Gibt an ob eine bestehende Datei überschrieben werden soll												
generate	Gibt an ob das Element überhaupt eine Ausgabedatei erzeugen soll												

Der Button Painter benötigt Informationen über Farben und Schriften zur Darstellung der Schaltflächen. Er verwendet hierzu vordefinierte Schlüssel welche über das Environment auf konkrete Definitionsobjekte abgebildet werden:

Schlüssel des **Def** Button Painters:

Schlüssel	Definitionstyp	Beschreibung
button.font.face		Schaltflächenbeschriftung
button.color.border	<color>	Rahmen
button.color.bg	<color>	Hintergrundfarbe
button.color.text.active	<color>	Beschriftung
button.color.text.inactive	<color>	Beschriftung inaktive Schaltfläche
button.color.state.active	<color>	Zustandsindikator aktiv
button.color.state.inactive	<color>	Zustandsindikator inaktiv
button.color.state.hover	<color>	Zustandsindikator hover
button.color.state.down	<color>	Zustandsindikator down







Schlüssel des **Def2** Button Painters:

Schlüssel	Definitionstyp	Beschreibung
button.font.face		Schaltflächenbeschriftung
button.color.border	<color>	Rahmen
button.color.border.inactive	<color>	Rahmen für inactive Schaltflächen

button.color.bg	<color>	Hintergrund
button.color.text.active	<color>	Beschriftung
button.color.text.inactive	<color>	Beschriftung inaktive Schaltfläche
button.color.state.active	<color>	Zustandsindikator aktiv
button.color.state.inactive	<color>	Zustandsindikator inaktiv
button.color.state.hover	<color>	Zustandsindikator hover
button.color.state.down	<color>	Zustandsindikator down
button.color.pattern.active	<color>	Muster aktiv
button.color.pattern.inactive	<color>	Muster inaktiv
button.color.pattern.hover	<color>	Muster hover
button.color.pattern.down	<color>	Muster down

2.4.10 Ressourcen Element <menu>

Definiert eine Menü-Schaltflächen

Attribut	Bedeutung
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen.
name	<p>Name des Images, unter dem der Menü button generiert wird. Menü Buttons beginnen immer mit der Zeichenfolge btn und erhalten am Ende eine Ziffer, die den Zustand des Menü Buttons kennzeichnet. Dabei gilt:</p> <p>1 = normaler Zustand  btnLogout1.gif</p> <p>2 = inaktiver Zustand  btnLogout2.gif</p> <p>3 = Hover Effekt  btnLogout3.gif</p> <p>4 = gedrückter Zustand  btnLogout4.gif</p> <p>5 = Selektierter Zustand  btnLogout5.gif</p> <p>6 = Hover Effekt  btnLogout6.gif</p> <p>Das „btn“ Präfix und die Statusnummer werden jedoch im name-Attribut <u>nicht</u> angegeben! Für das gezeigte Beispiel reicht also die folgende Einstellung: name="Logout"</p>
label	Beschriftung der Menü Schaltfläche
overwrite	Gibt an ob eine bestehende Datei überschrieben werden soll
generate	Gibt an ob das Element überhaupt eine Ausgabedatei erzeugen soll

Der Menü Painter benötigt Informationen über Farben und Schriften zur Darstellung der Menü Schaltflächen. Er verwendet hierzu vordefinierte Schlüssel welche über das Environment auf konkrete Definitionsobjekte abgebildet werden:

Schlüssel des **Def** Menü Painters:

Schlüssel	Definitionstyp	Beschreibung
menu.font.face		Menü Beschriftung
menu.color.bg	<color>	Hintergrundfarbe
menu.color.text.active	<color>	Beschriftung
menu.color.text.inactive	<color>	Beschriftung inaktives Menü
menu.color.glow	<color>	Farbe für den Glow Effekt

menu.color.shadow	<color>	Farbe des Schattens
menu.shadow.x	Int-property	Der X-Offset des Schattens
menu.shadow.y	Int-property	Der Y-Offset des Schattens
menu.backlit.radius	Int-property	Der Radius der Hintergrundbeleuchtung

Schlüssel des **Def2** Menü Painters:

Schlüssel	Definitionstyp	Beschreibung
menu.font.face		Menü Beschriftung
menu.color.bg	<color>	Hintergrundfarbe
menu.color.text.active	<color>	Beschriftung
menu.color.text.inactive	<color>	Beschriftung inaktives Menü
menu.color.glow	<color>	Farbe für den Glow Effekt
menu.color.shadow	<color>	Farbe des Schattens
menu.shadow.x	Int-property	Der X-Offset des Schattens
menu.shadow.y	Int-property	Der Y-Offset des Schattens
menu.backlit.radius	Int-property	Der Radius der Hintergrundbeleuchtung

2.4.11 Ressourcen Element <bundle>

Das <bundle> Element dient zur Definition eines lokalisierten PropertyResource Bundles.

Attribut	Bedeutung
destdir	Relatives Verzeichnis, in welchem die Ressourcen abgelegt werden sollen.
package	Gibt den Namen des Java Paketes an in welchem die *.properties Dateien abgelegt werden sollen
name	Gibt den Namen des Resourcebundles an
defaultlocale	Gibt das Default Locale an. Für das Default Locale wird eine zusätzliche *.properties Datei ohne Locale Suffix erzeugt
overwrite	Gibt an ob eine bestehende Datei überschrieben werden soll
generate	Gibt an ob das Element überhaupt eine Ausgabedatei erzeugen soll

Das <bundle> Element erzeugt die folgenden Ausgaben:

- Lokalisierte <package>\<name>_<locale>.properties Dateien
- Für das default Locale wird eine zusätzliche <package>\<name>.properties Datei erzeugt
- Erzeugung einer Java Interface Klasse mit dem Namen <package>.<name>.java welche alle öffentlichen Schlüssel des Property Resource Bundle als symbolische Konstanten enthält

Beispiel:

```
<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 erwartet in Feld: {0}</value>
          <value locale="en">{0} is required.</value>
        </resourcekey>
      <resourcekey code="invalid.retype">
```


3 Starten des Build Vorganges

3.1 Voraussetzungen

Damit das ResourceFactory Tool gestartet werden kann müssen die mitgelieferten Java Bibliotheken im Klassenpfad liegen:

- **cc-resfactory-1.4.055.jar** Die ResourceFactory selbst
- **glf.jar** Das Graphics Layer Framework von SUN wird zur Erstellung von komplexen Grafiken verwendet

Zur Verwendung des Ant Tasks ist zusätzlich die folgende Bibliothek notwendig:

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

Das ResourceFactory Tool erzeugt Grafiken über das Java Abstract Windowing Toolkit (AWT). Dabei handelt es sich lediglich um eine Schnittstellen Spezifikation (Java Interfaces). Die tatsächliche Implementierung ist von der verwendeten Java Virtual Machine und dem Betriebssystem abhängig. Bei den meisten Betriebssystemen werden die Grafikbefehle an das darunterliegende Grafik Subsystem weitergeleitet:

- Microsoft Windows → GDI
- Linux → X-Windows

Die meisten Server Betriebssysteme verfügen über keine Grafik Unterstützung (sogenannte „headless“ Systeme)

- OS390
- Unix

Auf diesen „headless“ Systemen kann die ResourceFactory nur eingeschränkt verwendet werden. Grafiken können hier erst durch die Installation eines zusätzlichen Grafikpaketes erzeugt werden. Als beispiel sei hier das kostenfreie „Pure Abstract Windowing Toolkit“ (**PJA**) genannt (<http://www.eteks.com/pja/en>).

3.2 Start über die Kommandozeile

Die Erzeugung der Ressourcen wird durch Aufruf der Batchdatei **build.bat** gestartet.

3.3 Verwendung der Ant-Tasks

Das ResourceFactory Tool enthält mehrere Ant Tasks (<http://ant.apache.org>) welche in einem Ant Build-Script verwendet werden können. Damit kann die Erzeugung der aktuellen Ressourcen in den Build-Prozess einer Anwendung integriert werden.

3.3.1 Setzen der Build Properties

Die Datei build.properties dient zur Festlegung folgender Einstellungen:

Variable	Beschreibung
build.home	Gibt das Verzeichnis an, in dem die Ressourcen abgelegt werden.
build.src	Name der Datei, welche die zu generierenden Ressourcen definiert. Der Ausdruck <code>\${basedir}</code> spezifiziert dabei das Verzeichnis, in welchem das ResourceFactory Tool installiert bzw. entpacket wurde.

Beispiel:

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

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

Abbildung 3: Die Datei build.properties

3.3.2 Der <fontlist> Task

3.3.2.1 Beschreibung

Der <fontlist> Task gibt eine Liste der auf dem System installierten Schriftarten auf der Konsole aus.

3.3.2.2 Voraussetzungen

Der Task muss zuerst mit dem internen <taskdef> Tag bekanntgemacht werden:

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

3.3.2.3 Attribute

keine

3.3.2.4 Beispiel

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

3.3.3 Der <resourcefactory> Task

3.3.3.1 Beschreibung

3.3.3.2 Voraussetzungen

Der Task muss zuerst mit dem internen <taskdef> Tag bekanntgemacht werden:

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

3.3.3.3 Attribute

Attribut	Beschreibung	Pflicht
destdir	Gibt das Ausgabeverzeichnis an in welches die Ressourcen generiert werden sollen	Ja
srcfile	Gibt die Ressourcen XML Datei an. Alternativ können auch mehrere Ressourcen XML Dateien über geschachtelte <fileset> Tags angegeben werden. In Diesem Fall kann das „srcfile“ Attribut	Ja oder <fileset>

	weggelassen werden	
severity	Das ResourceFactory Tool überprüft die Konsistenz der Ressourcen XML Datei(en). Dieses Attribut gibt die Fehlerklasse an ab welcher Konsistenz Prüfungen ausgegeben werden. Gültige Wert sind: <ul style="list-style-type: none"> • none • question • information • warning • error • fatal 	Nein Voreinstellung: question
generate	Gibt an ob Ressourcen erzeugt werden sollen. Gültige werte sind: <ul style="list-style-type: none"> • true • false 	Nein Voreinstellung: true
painter	Gibt den painter an mit welchem die Ressourcen gezeichnet werden sollen. Gültige Werte sind: <ul style="list-style-type: none"> • def • def2 	Ja

3.3.3.4 Beispiel

```
<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 Der <version> Task

3.3.4.1 Beschreibung

Der <version> Task erzeugt den Sourcecode für eine Java Klasse mit Versionsinformationen über die Anwendung.

3.3.4.2 Voraussetzungen

Der Task muss zuerst mit dem internen <taskdef> Tag bekanntgemacht werden:

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

3.3.4.3 Attribute

Attribut	Beschreibung	Pflicht
destdir	Gibt das Ausgabeverzeichnis an in welches die Ressourcen generiert werden sollen	Ja
packagename	Der Paketname der Java Klasse	Ja
classname	Der Klassenname der Java Versionsklasse	Ja
majorversion	Die Haupt Versionsnummer	
minorversion	Die Neben Versionsnummer	
buildnumber	Die buildnummer	
productname	Der Produktname	
productextension	Der Namenszusatz zum Produktnamen	
systemtype	Systemtyp Gültige Werte sind: <ul style="list-style-type: none"> • production • test • debug • develop • demo 	
vendor	Hersteller	
vendorsite	Website des Herstellers	

3.3.4.4 Beispiel

```
<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 Ausgabe der verfügbaren Schriftarten

Die auf dem System installierten Schriften können durch Aufruf der Batchdatei [fontlist.bat](#) auf dem Bildschirm ausgegeben werden. Die aufgeführten, d.h. installierten Schriftfamilien können für die Beschriftung von Schaltflächen herangezogen werden.

```
<!-- 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

Bei Fragen oder Problemen stehen wir Ihnen gerne zur Verfügung. Benutzen Sie bitte für Anfragen unser Service Formular auf unserer Homepage. Wir sind bemüht Ihre Anfragen zeitnah zu beantworten.

Wenn Sie Anregungen zu diesem Dokument oder Verbesserungsvorschläge haben, wären wir Ihnen dankbar, wenn Sie uns diese kurz mitteilen würden.