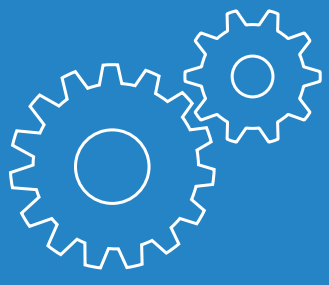
SICHER. **VERNETZT.**

LANCOM Wireless ePaper Server
Rendering Reference



LANCOM
Systems

Inhalt

1 Inhalt

Diese Dokumentation enthält Informationen zum Rendering von Grafiken über den LANCOM Wireless ePaper Server.

Sie ist in folgende Abschnitte unterteilt:

- **Allgemeine Information** - Einleitung und allgemeine Information zum Renderingprozess der Grafiken.
- **Elemente** - Beschreibung aller Elemente und Attribute, die dazu dienen, ein Template zu definieren.
- **Utils** - Eine Dokumentation über alle Util-Methoden, die innerhalb eines Templates aufgerufen werden können, um den Inhalt zu verändern.
- **Beispiele** - Mehrere Beispiele inklusive nutzbarer Templates zur Generierung von Grafiken.

2 Allgemein

Der Renderprozess von Grafiken basiert auf *XSL (Extensible Stylesheet Language)* Templates. Das Template wird zur Spezifizierung der Felder genutzt, mit denen das Bild gerendert werden soll. Das jeweilige Bild für das Display wird generiert, indem das Template für ein XML angewendet wird. Dies hängt von den Eigenschaften des Displays ab und wird erweitert durch ein kundenspezifisches XML des Nutzers.

Folgende Schritte werden beim Verarbeiten eines Templates ausgeführt:

1. Generieren eines XML-basierten Eintrags auf dem Display.
2. Anwendung des Templates (XSL) für den generierten XML-Eintrag. Daraus ergibt sich ein Dokument, welches Felder beinhaltet, die in diesem Verweis spezifiziert sind und Werte aus dem XML referenzieren.
3. Rendern des Bildes, basierend auf dem vorherigen Output.

XSL Template

Das XSL Template wird im XML-Eintrag angewendet. Nachdem das Template angewendet wird, wird eine Quelle zum Generieren des Bildes erzeugt. Die Struktur eines Templates sollte wie folgt aussehen (Beispiel):

Template Example

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

     <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" version="1.0">

     <xsl:template match="Record">

     Space for field definitions and XML record data referencing


     </xsl:template>

     </xsl:stylesheet>
```

XML-Eintrag

Bevor ein Bild gerendert wird, wird automatisch ein XML-Eintrag generiert. Das XML weist die folgende Struktur auf:

Record Example

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

     <Record>

      <Label>

       <Id>B1000000</Id>

       <LabelType>iMAGOTAG 2.7</LabelType>

       <DisplayHeight>176</DisplayHeight>

       <DisplayWidth>264</DisplayWidth>

      </Label>

      <Task>

       <CurrentDate>11.09.2012</CurrentDate>
```

```
      <CurrentTime>15:13:53</CurrentTime>
      <ExternalId>0</ExternalId>
      <Page>0</Page>
     </Task>
    </Record>
```

Das vom User spezifizierte XML wird nach dem „Auftrag“ Element hinzugefügt.

Dies bedeutet, dass dieses XML jedem Feld im XSL Template zugewiesen werden kann, z.B.

<xsl:value select="Label/Id"/>

weist die ID des Displays zu.

Utils

Das Template Utils wurde erstellt, um Einfluss auf die Textinhalte für das generierte Bild zu haben. Nähere Informationen über die verschiedenen unterstützten Methoden, deren Parameter und Outputs finden Sie im Abschnitt *Utils*.

Unterstützung von CSS

Die Templates unterstützen das Importieren von einer oder mehreren CSS-Dateien. Damit werden die Attribute der verschiedenen Felder festgelegt. Jedes Tag, außer dem Haupt-Image-Tag, unterstüzt HTML-Attribute wie „ID“ und „Klasse“ aus der CSS-Datei.

Der LANCOM Wireless ePaper Server unterstützt die Definition von CSS1. Die in der CSS-Datei verwendeten Attribute haben die gleiche Bezeichnung wie die Template-Attribute (z.B. Template: Schriftgröße=“12“, in CSS: Schriftgröße: 12;)

Um eine CSS-Datei zu importortieren, muss ein <css>-Tag und das “href“-Attribut im Haupt-Image-Tag definiert sein. Das “href“-Attribut bestimmt den relativen Pfad zur CSS-Datei von der Template-Datei.

Für die Anwendung siehe *2,7“* auf Seite 74 im Abschnitt "Beispiel CSS".

3 Elemente

Dieses Kapitel beschreibt alle Elemente und Attribute, die dazu dienen, ein Template zu definieren.

3.1 Das Element <image>

Beschreibung:

Das Basiselement für die Formatvorlage eines Bildes. Es bestimmt sowohl die wesentlichen Merkmale (z.B. Größe, Hintergrund) eines zu generierenden Bildes als auch die Standardwerte für Schrift und Farbe.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
colors	Die Farbe des Elements oder des vorhandenen Texts.	falsch	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	falsch	*2,7"* auf Seite 74
invert	Invertiert die Farbe des Bildes.	falsch	wahr | falsch	falsch	-
rotation	Legt das Drehen des auszugebenden Bildes fest. Das Drehen des Bildes ist nur um 0, 90, 180 und 270 Grad möglich.	falsch	0|90|180|270	0	-
background-image	Bestimmt das Hintergrundbild eines Elements. Der Pfad zu dem Bild sollte relativ zu dem genutzten Template angegeben sein. Ist das Bild zu klein, wird es in die obere linke Ecke gesetzt.	falsch	Ein Pfad relativ zur Template-Datei oder zu einer URL (z.B. "../test/image.png oder url.to/image.png").	Kein Default-Wert	-
width	Die Breite des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
height	Die Höhe des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	*4,4"* auf Seite 108
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	*2,7"* auf Seite 74
clip-height	Die Höhe des Clipping-Bereichs (Anzeigebereichs).	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
clip-width	Die Breite des Clipping-Bereichs (Anzeigebereichs).	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
clip-x	Die x-Koordinaten für den Clipping-Bereich.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
clip-y	Die y-Koordinaten für den Clipping-Bereich.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
dithering	Aktiviert oder deaktiviert Dithering für das entstehende Bild.	falsch	wahr | falsch	falsch	-

Untergeordnete Elemente

<field>	siehe Kapitel *Das Element <field>*
<rect>	siehe Kapitel *Das Element <rect>*
<line>	siehe Kapitel *Das Element <line>*
<triangle>	siehe Kapitel *Das Element <triangle> (Dreieck)*
<ellipse>	siehe Kapitel *Das Element <ellipse>*
<polygon>	siehe Kapitel *Das Element <polygon>*

3.2 Das Element <field>

Beschreibung:

Das Element `field` (Feld) bestimmt einen rechteckigen Container, in den ein Bild, Text oder Barcode positioniert werden kann. Die Position des Feldes kann entweder über die x- und y-Koordinate der oberen linken Ecke bestimmt werden oder indem nur die x-Koordinate angegeben werden. Im zweiten Fall wird das Feld genau nach dem letzten Feld positioniert, das in der y-Koordinate festgelegt war (oder 0, für den Fall, dass kein Feld in der y-Koordinate festgelegt war).

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
rotation	Ein positiver oder negativer Drehwinkel, um den entsprechenden Feldinhalt zu drehen.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	0	*4,4"* auf Seite 108
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
x	Die Position des Elements auf der x-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
y	Die Position des Elements auf der y-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
width	Die Breite des Elements, angegeben in Pixeln.	wahr	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
height	Die Höhe des Elements, angegeben in Pixeln.	wahr	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
max-height	Legt die maximale Höhe des Feldes fest. Dieses Attribut wird in Kombination mit der relativen y-Positionierung verwendet. Die Höhe des gerenderten Feldes wird kleiner oder identisch mit der festgelegten	falsch	Ein positiver Integer-Wert (>0)	Kein Default-Wert	*2,7"* auf Seite 74

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	maximalen Höhe sein, abhänging von dem Inhalt des Feldes.				
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
border	Legt die Dicke des Rahmens um das Element fest. Der Wert 0 bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	0	*4,4"* auf Seite 108
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border-style	Legt die Linienart des Rahmens fest.	falsch	gepunktet | gestrichelt | durchgezogen	Durchgezogen	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
valign	Legt die vertikale Ausrichtung des Elements/Textes fest.	wahr	Oben | Mitte | Unten	Oben	-
rounded-corners	Legt fest, welche Ecken abgerundet werden sollen. Mehrere Ecken können, getrennt durch ein Komma (,), ausgewählt werden.	falsch	Oben_links | Oben_rechts | Unten_links | Unten_rechts	Kein Default-Wert	*7,4"* auf Seite 122
corner-radius	Radius der abgerundeten Ecken des Rechtecks/Feldes. Ein Radius mit dem Wert Null bedeutet keine abgerundeten Ecken. Der Radius der Ecken wird automatisch angepasst, sollte er zu groß sein. Der	falsch	Die Größe, Wert angegeben in Pixeln.	0	*7,4"* auf Seite 122

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	maximale Radius beträgt (Min(Breite, Höhe) - 1) / 2.				
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-

Untergeordnete Elemente

<field>	siehe Kapitel *Das Element <field>* auf Seite 8
<table>	siehe Kapitel *Das Element <table>*
<barcode>	siehe Kapitel *Das Element <barcode>*
<img>	siehe Kapitel *Das Element <img>*
<label>	siehe Kapitel *Das Element <label>*
<text>	siehe Kapitel *Das Element <text>*
<rect>	siehe Kapitel *Das Element <rect>*
<line>	siehe Kapitel *Das Element <line>*
<triangle>	siehe Kapitel *Das Element <triangle> (Dreieck)*
<ellipse>	siehe Kapitel *Das Element <ellipse>*
<polygon>	siehe Kapitel *Das Element <polygon>*

Beispiele

- *2,7"* Beispiel verschachtelte Felder

3.3 Das Element <table>

Beschreibung:

Das Element `table` kennzeichnet den Anfang einer Tabelle, es folgen Tabellenzeilen und -zellen. Als Standard übernimmt die Tabelle die Breite des Feldes, in das die Tabelle gesetzt wird. Die Höhe der Tabelle hängt von der Höhe der Tabellenzeilen innerhalb der Tabelle ab.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
padding-bottom	Definiert den Innenabstand in Pixeln von unten.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-left	Definiert den Innenabstand in Pixeln von links.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-right	Definiert den Innenabstand in Pixeln von rechts.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-top	Definiert den Innenabstand in Pixeln von oben.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen,	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	durchgestrichen oder keine).				
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt	wahr	wahr | falsch	falsch	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	gerendert werden soll.				
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
cellpadding	Bestimmt den Abstand um den Textinhalt einer Tabelle, angegeben in Pixeln.	wahr	Ein positiver Integer-Wert (>=0)	2	-
valign	Legt die vertikale Ausrichtung des Elements/Textes fest.	wahr	Oben | Mitte | Unten	Oben	-
width	Die Breite des Elements, angegeben in Pixeln.	wahr	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
height	Die Höhe des Elements, angegeben in Pixeln.	wahr	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
max-height	Legt die maximale Höhe des Feldes fest. Dieses Attribut wird in Kombination mit der relativen y-Positionierung verwendet. Die Höhe des gerenderten Feldes wird kleiner oder identisch mit der festgelegten maximalen Höhe sein, abhänging von dem Inhalt des Feldes.	falsch	Ein positiver Integer-Wert (>0)	Kein Default-Wert	, *4,4"* auf Seite 108*2,7"* auf Seite 74

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
table-border	Bestimmt die Rahmenbreite der gesamten Tabelle mit Ausnahme der Kopfzeile und/oder -spalte. Die Breite der Kopfzeile/-zelle wird bestimmt durch das Attribut Tabellenrahmen. Ein Rahmen mit dem Wert Null bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	1	
header-border	Bestimmt die Rahmenbreite der Kopfzeile und -spalte. Dieses Attribut hat nur Auswirkung wenn „Titelzeile“ und/oder „Titelspalte“ deaktiviert ist.	falsch	Ein positiver Integer-Wert (>0)	2	
header-column	Die Kopfspalte ist die erste Spalte einer Tabelle. Die Rahmendicke dieser Spalte kann festgelegt werden als „Titelrahmen“, wenn es aktiviert ist.	falsch	wahr | falsch	falsch	
header-row	Die Kopfzeile ist die erste Zeile einer Tabelle. Die Rahmendicke dieser Zeile kann festgelegt werden als „Titelrahmen“, wenn es aktiviert ist.	falsch	wahr | falsch	falsch	
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-

Untergeordnete Elemente

<row> siehe Kapitel *Das Element <row> (Zeile)*

Beispiele

- *4.4"* Beispiel Tabelle

3.4 Das Element <row> (Zeile)

Beschreibung:

Das Element `row` (Zeile) kennzeichnet eine neue Zeile innerhalb einer Tabelle. Die Zeilenhöhe und - breite kann festgelegt werden. Ist keine Höhe und Breite festgelegt, wird die maximale Breite und Höhe verwendet, die der Zeileninhalt benötigt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
padding-bottom	Definiert den Innenabstand in Pixeln von unten.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-left	Definiert den Innenabstand in Pixeln von links.	wahr	Ein positiver oder negativer Integer-Wert.	0	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
padding-right	Definiert den Innenabstand in Pixeln von rechts.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-top	Definiert den Innenabstand in Pixeln von oben.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
table-height	Bestimmt die Höhe des aktuellen Tabellenelements.	falsch	Ein positiver Integer-Wert (>=0)	Kein Default-Wert	
valign	Legt die vertikale Ausrichtung des Elements/Textes fest.	wahr	Oben | Mitte | Unten	Oben	-
cellpadding	Bestimmt den Abstand um den Textinhalt einer Tabellenzelle, angegeben in Pixeln.	wahr	Ein positiver Integer-Wert (>=0)	2	-

Untergeordnete Elemente

<cell> siehe Kapitel *Das Element <cell>*

Beispiele

- *4.4"* Beispiel Tabelle

3.5 Das Element <cell>

Beschreibung:

Das Element `cell` (Zelle) steht für eine Tabellenzelle innerhalb einer Tabellenreihe. Es ist möglich, eine feste Breite zu bestimmen. Ist eine feste Breite festgelegt, wird dieser Wert auf die Spaltengröße für die gesamte Tabelle übernommen. Bestimmen mehrere Zellen die Breite dieser Spalte, wird der maximale Wert übernommen. Sind keine Werte festgelegt, wird die Breite der Zelle gleichmäßig verteilt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
padding-bottom	Definiert den Innenabstand in Pixeln von unten.	wahr	Ein positiver oder negativer Integer-Wert.	0	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
padding-left	Definiert den Innenabstand in Pixeln von links.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-right	Definiert den Innenabstand in Pixeln von rechts.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-top	Definiert den Innenabstand in Pixeln von oben.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch	wahr	Eine oder mehrere Gleitkommazahl(en),	1,0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	mit dem bestmöglichen Komprimierungsfaktor komprimiert.		getrennt durch einen Doppelpunkt.		
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb	wahr	wahr | falsch	falsch	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	oder außerhalb des entsprechenden Textes gelegt werden soll.				
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-
table-width	Die Breite des Clipping-Bereichs (Anzeigebereichs).	falsch	Ein positiver Integer-Wert (>=0)	Kein Default-Wert	
cellpadding	Bestimmt den Abstand um den Textinhalt einer Tabelle, angegeben in Pixeln.	wahr	Ein positiver Integer-Wert (>=0)	2	-
valign	Legt die vertikale Ausrichtung des Elements/Textes fest.	wahr	Oben | Mitte | Unten	Oben	-

Untergeordnete Elemente

<label> siehe Kapitel *Das Element <label>*

<text> siehe Kapitel *Das Element <text>*

Beispiele

- *4.4"* Beispiel Tabelle

3.6 Das Element <img>

Beschreibung:

Dieser Tag wird genutzt, um ein Bild vor einem Hintergrund einzufügen. Entweder wird ein relativer Pfad des Templates zu einer Bilddatei ausgewählt oder ein Base64-kodierter String eines Bildes. Vorsicht bei Nutzung eines Pfades für ein Bild: Wird das Template auf dem Server verarbeitet, kann sich der Pfad möglicherweise verändert haben oder das Bild existiert nicht mehr auf dem Server.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*
src	Definiert den Pfad oder die URL zu einem Bild. Der Pfad muss relativ zur Template-Datei sein.	falsch	Ein Pfad relativ zur Template-Datei oder zu einer URL (z.B. "../test/image.png oder url.to/image.png")	Kein Default-Wert	*4,4"* auf Seite 108
data	Legt einen Base64 kodierten String als Bildquelle fest.	falsch	Ein Base64 kodierter String	Kein Default-Wert	*4,4"* auf Seite 108
autoscale	Aktiviert/deaktiviert automatische Skalierung des Elementes. Das Element wird auf die Feldgröße skaliert.	falsch	wahr | falsch	falsch	-
cut-bottom	Legt die Pixelanzahl für jede Seite des Ursprungsbildes fest,	falsch	Ein positiver Integer Wet (>=0)	0	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	die abgeschnitten werden soll.				
cut-left	Legt die Pixelanzahl für jede Seite des Ursprungsbildes fest, die abgeschnitten werden soll.	falsch	Ein positiver Integer Wet (>=0)	0	-
cut-right	Legt die Pixelanzahl für jede Seite des Ursprungsbildes fest, die abgeschnitten werden soll.	falsch	Ein positiver Integer Wet (>=0)	0	-
cut-top	Legt die Pixelanzahl für jede Seite des Ursprungsbildes fest, die abgeschnitten werden soll.	falsch	Ein positiver Integer Wet (>=0)	0	-

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Bild

3.7 Das Element <barcode>

Beschreibung:

Das Element `barcode` legt die Eigenschaften fest, um einen Barcode auf dem Bild auszugeben.

Eigenschaften

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiel:
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiel:
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108*Text*
type	Der Barcode Typ, der erzeugt werden soll.	falsch	codabar | code128 | code39 | ean13 | ean8 | ean-128 | upc-a | upc-e | intl2of5 | itf-14 | postnet | royal-mail-cbc | usps4cb | pdf417 | datamatrix | qr	Kein Default-Wert	-

Barcode-Typen

Tabelle 1: Codabar

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadablePlacement	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
displayStartStop	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | toben | keine	unten

Name	Beschreibung	Mögliche Werte	Default-Wert
displayStartStop	Aktiviert/deaktiviert Anzeige von Start und Stop Kennzeichen für Codabar und Code-39 Barcodes.	wahr | falsch	falsch

Tabelle 2: Code 128

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
codesets	Setzt den Zeichensatz für Code-128 Barcodes.	Eine String-Kombination für Zeichensätze A, B und C.	abc

Tabelle 3: Code39

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6

Name	Beschreibung	Mögliche Werte	Default-Wert
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
displayStartStop	Aktiviert/deaktiviert Anzeige von Start und Stop Kennzeichen für Codabar und Code-39 Barcodes.	wahr | falsch	falsch
displayChecksum	Anzeigen/Verbergen der Prüfsumme	wahr | falsch	falsch
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Ignorieren
intercharGapWidth	Legt die Breite der Ruhezone des Code-39 Barcodes fest.	Ein positiver Integer-Wert (>0)	1
extendedCharset	Aktiviert/deaktiviert erweiterten Zeichensatz für Code-39 Barcodes.	wahr | falsch	falsch

Tabelle 4: DataMatrix

Name	Beschreibung	Mögliche Werte	Default-Wert
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	1
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	wahr
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
autoscale	Aktiviert/deaktiviert automatische Skalierung von	wahr | falsch	falsch

Name	Beschreibung	Mögliche Werte	Default-Wert
	Objekten. Das Objekt wird auf die Feldgröße skaliert.		
shape	Legt die Form des DataMatrix Barcodes fest.	Keine|Rechteck|Form	Keine
padded	Aktiviert/deaktiviert Abstand	wahr | falsch	falsch
minSymbolSizeHeight	Legt die Mindesthöhe des Symbols fest.	Ein positiver Integer Wet (>=0)	1
minSymbolSizeWidth	Legt die Mindestbreite des Symbols fest.	Ein positiver Integer Wet (>=0)	1
maxSymbolSizeHeight	Legt die Maximalhöhe des Symbols fest.	Ein positiver Integer Wet (>=0)	1
maxSymbolSizeWidth	Legt die Maximalbreite des Symbols fest.	Ein positiver Integer Wet (>=0)	0

Tabelle 5: EAN-128

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
template	Legt ein optionales Template für den EAN-128 Barcode fest.	String	Kein Default-Wert

Name	Beschreibung	Mögliche Werte	Default-Wert
omitBrackets	Aktiviert/deaktiviert Weglassen der Klammer für EAN-128 Barcode.	wahr | falsch	falsch

Tabelle 6: EAN-13

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Automatisch

Tabelle 7: ean8

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6

Name	Beschreibung	Mögliche Werte	Default-Wert
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Automatisch

Tabelle 8: intl2of5

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von	wahr | falsch	falsch

Name	Beschreibung	Mögliche Werte	Default-Wert
	Objekten. Das Objekt wird auf die Feldgröße skaliert.		
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
wideFactor	Definiert den Faktor für die Berechnung der Lücken zwischen den Barcode Streifen.	Eine positive Gleitkommazahl (>0)	3,0
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Ignorieren
displayChecksum	Anzeigen/Verbergen der Prüfsumme	wahr | falsch	falsch

Tabelle 9: ITF-14

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
bearerBarWidth	Definiert die Breite (in Pixeln) des Rahmen rundum den Barcode für ITF-14 Barcodes.	Ein positiver Integer Wet (>=0)	20

Name	Beschreibung	Mögliche Werte	Default-Wert
bearerBox	Definiert die Breite (in Pixeln) des Rahmen rundum den Barcode für ITF-14 Barcodes.	Ein positiver Integer Wet (>=0)	wahr
wideFactor	Definiert den Faktor für die Berechnung der Lücken zwischen den Barcode Streifen.	Eine positive Gleitkommazahl (>0)	2,0
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Automatisch
displayChecksum	Anzeigen/Verbergen der Prüfsumme	wahr | falsch	falsch

Tabelle 10: PDF417

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
rowHeightRatio	Definiert das Größenverhältnis der Zeilen für den PDF417 Barcode.	Eine positive Gleitkommazahl (>0)	3,0
columns	Definiert die Anzahl der Spalten. Kann deaktiviert werden, indem ein Nullwert verwendet wird. In diesem Fall wird es automatisch erkannt.	Ein positiver Integer-Wert (>0)	0
minColumns	Definiert die Mindestanzahl der Spalten.	Ein positiver Integer-Wert (>0)	2

Name	Beschreibung	Mögliche Werte	Default-Wert
minColumns	Definiert die maximale Anzahl der Spalten.	Ein positiver Integer-Wert (>0)	2
ecLevel	Definiert die Mindestanzahl der Zeilen.	Ein positiver Integer-Wert (>0)	0
widthToHeightRatio	Definiert das Verhältnis zwischen Breite und Höhe.	Eine positive Gleitkommazahl (>0)	3,0
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	2
verticalQuietZoneFactor	Faktor zur Berechnung der vertikalen Ruhezone.	Ein positiver Integer-Wert (>0)	6
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	wahr
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
padded	Aktiviert/deaktiviert Abstand	wahr | falsch	falsch
minRows	Definiert die Mindestanzahl der Zeilen.	Ein positiver Integer-Wert (>0)	3
maxRows	Definiert die Mindestanzahl der Zeilen.	Ein positiver Integer-Wert (>0)	90
autoMode	Aktiviert/deaktiviert den Auto Mode von PDF417 Barcodes.	wahr | falsch	wahr

Tabelle 11: PostNet

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr

Name	Beschreibung	Mögliche Werte	Default-Wert
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten

Tabelle 12: QR Code

Name	Beschreibung	Mögliche Werte	Default-Wert
padded	Aktiviert/deaktiviert Abstand	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	2
errorCorrectionLevel	Definiert den Fehlerkorrektur-Level eines QR-Codes.	l | m | q | h	h
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch

Tabelle 13: Royal Mail CBC

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von	wahr | falsch	falsch

Name	Beschreibung	Mögliche Werte	Default-Wert
	Objekten. Das Objekt wird auf die Feldgröße skaliert.		
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten

Tabelle 14: UPC-A

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Automatisch

Tabelle 15: UPC-E

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6

Name	Beschreibung	Mögliche Werte	Default-Wert
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von Objekten. Das Objekt wird auf die Feldgröße skaliert.	wahr | falsch	falsch
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten
checksumMode	Legt die Handhabung der Prüfsumme fest.	Hinzufügen | Prüfen | Ignorieren | Automatisch	Automatisch

Tabelle 16: USPS4CB

Name	Beschreibung	Mögliche Werte	Default-Wert
fontSizePt	Legt eine Reihe von Schriftgrößen fest, getrennt durch Semikolon. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch ein Semikolon als alternative Schriftgrößen.	6
fontName	Legt die Schrift des Textinhalts des Barcodes fest.	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Helvetica
quietZoneFactor	Faktor zur Berechnung der Ruhezone eines Barcodes.	Ein positiver Integer-Wert (>0)	10
quietZoneEnabled	Falls eine Ruhezone aktiviert/deaktiviert werden soll.	wahr | falsch	falsch
scale	Festlegen einer Skala für den Barcode selbst.	Ein positiver Integer Wet (>=0)	1
humanReadableEnabled	Aktiviert oder deaktiviert die Anzeige des Textinhalts des Barcodes.	wahr | falsch	wahr
autoscale	Aktiviert/deaktiviert automatische Skalierung von	wahr | falsch	falsch

Name	Beschreibung	Mögliche Werte	Default-Wert
	Objekten. Das Objekt wird auf die Feldgröße skaliert.		
humanReadablePlacement	Definiert, an welcher Stelle der Textinhalt im Verhältnis zum Barcode angezeigt werden soll.	unten | oben | keine	unten

Untergeordnete Elemente

<text> siehe Kapitel *Das Element <text>*

Beispiele

- *4,4"* Beispiel Barcode

3.8 Das Element <rect>

Beschreibung:

Definiert ein Rechteck, das auf ein Bild übertragen werden soll. Das Rechteck wird hinter die Feldelemente gesetzt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
x	Die Position des Elements auf der x-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
y	Die Position des Elements auf der	falsch	Ein Integer-Wert >= 0 und kleiner als die	Kein Default-Wert	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	y-Achse, angegeben in Pixeln.		festgelegte Bildgröße.		
width	Die Breite des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
height	Die Höhe des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border	Legt die Dicke des Rahmens um das Element fest. Der Wert 0 bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	0	*4,4"* auf Seite 108
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border-style	Legt die Linienart des Rahmens fest.	falsch	Gepunktet | Gestrichelt | Durchgezogen	Durchgezogen	-
fill	Legt fest, ob das Element mit der gewählten Farbe ausgefüllt werden soll.	falsch	wahr | falsch	wahr	-
rounded-corners	Legt fest, welche Ecken abgerundet werden sollen. Mehrere Ecken können, getrennt durch ein Komma (,), ausgewählt werden.	falsch	Oben_links | Oben_rechts | Unten_links | Unten_rechts	Kein Default-Wert	*7,4"* auf Seite 122
corner-radius	Radius der abgerundeten Ecken des Rechtecks/Feldes. Ein Radius mit dem Wert Null bedeutet keine abgerundeten Ecken. Der Radius der Ecken wird	falsch	Die Größe, Wert angegeben in Pixeln.	0	*7,4"* auf Seite 122

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	automatisch angepasst, sollte er zu groß sein. Der maximale Radius beträgt (Min(Breite, Höhe) - 1) / 2.				

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Abbildung
- *7,4"* Erweitertes Abbildungsbeispiel

3.9 Das Element <line>

Beschreibung:

Bestimmt die Eigenschaften einer Linie, die in einem Bild eingefügt werden soll. Die Linie wird hinter die Feldelemente gesetzt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
x-from	Bestimmt die Anfangsposition auf der x-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
y-from	Bestimmt die Anfangsposition auf der y-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*4,4"* auf Seite 108
x-to	Bestimmt die Endposition auf der x-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*4,4"* auf Seite 108
y-to	Bestimmt die Endposition auf der y-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*4,4"* auf Seite 108
thickness	Bestimmt die Dicke, angegeben in Pixeln.	falsch	Ein positiver Integer-Wert (>=0)	1	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Abbildung
- *7,4"* Erweitertes Abbildungsbeispiel

3.10 Das Element <triangle> (Dreieck)

Beschreibung:

Definiert ein Dreieck, das auf ein Bild übertragen werden soll. Das Element `triangel` wird hinter die Feldelemente gesetzt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	Mehrfachtags angewendet werden.				
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
x1	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
x2	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
x3	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
y1	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
y2	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
y3	Koordinate, zur Bestimmung einer Ecke eines Dreiecks.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	*7,4"* auf Seite 122
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border	Legt die Dicke des Rahmens um das Element fest. Der Wert 0 bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	0	*4,4"* auf Seite 108
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon	Schwarz	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
			getrennter RGBA String mit Werten zwischen 0 und 255		
border-style	Legt die Linienart des Rahmens fest.	falsch	Gepunktet | Gestrichelt | Durchgezogen	Durchgezogen	-
fill	Legt fest, ob das Element mit der gewählten Farbe ausgefüllt werden soll.	falsch	wahr | falsch	wahr	-

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Abbildung
- *7,4"* Erweitertes Abbildungsbeispiel

3.11 Das Element <ellipse>

Beschreibung:

Definiert eine Ellipse, die auf ein Bild übertragen werden soll. Die Ellipse wird hinter das Feldelement gesetzt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
x	Die Position des Elements auf der x-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
y	Die Position des Elements auf der y-Achse, angegeben in Pixeln.	falsch	Ein Integer-Wert >= 0 und kleiner als die festgelegte Bildgröße.	Kein Default-Wert	-
width	Die Breite des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
height	Die Höhe des Elements, angegeben in Pixeln.	falsch	Die Größe, Wert angegeben in Pixeln.	Kein Default-Wert	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border	Legt die Breite des Rahmens um das Element fest. Der Wert 0 bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	0	*4,4"* auf Seite 108
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border-style	Legt die Linienart des Rahmens fest.	falsch	gepunktet | gestrichelt | durchgezogen	Durchgezogen	-
fill	Legt fest, ob das Element mit der gewählten Farbe ausgefüllt werden soll.	falsch	wahr | falsch	wahr	-

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Abbildung
- *7,4"* Erweitertes Abbildungsbeispiel

3.12 Das Element <polygon>

Beschreibung:

Definiert ein Polygon, das auf ein Bild übertragen werden soll. Das Polygon wird hinter die Feldelemente gesetzt.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
points	Eine durch Leerzeichen getrennte Liste mit Koordinaten, die zum Zeichnen eines Polygons genutzt werden soll.	falsch	Eine durch Leerzeichen getrennte Liste mit Koordinaten. Koordinaten sind getrennt durch Doppelpunkt, z.B. 12,12,221,152	Kein Default-Wert	*7,4"* auf Seite 122
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
border	Legt die Dicke des Rahmens um das Element fest. Der Wert 0 bedeutet kein Rahmen.	falsch	Ein positiver Integer-Wert (>0)	0	*4,4"* auf Seite 108
border-color	Legt die Farbe des ausgewählten Rahmens fest.	falsch	schwarz | weiß | transparent | durch Semikolon	Schwarz	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
			getrennter RGBA String mit Werten zwischen 0 und 255		
border-style	Legt die Linienart des Rahmens fest.	falsch	Gepunktet | Gestrichelt | Durchgezogen	Durchgezogen	-
fill	Legt fest, ob das Element mit der gewählten Farbe ausgefüllt werden soll.	falsch	wahr | falsch	wahr	-

Untergeordnete Elemente

Keine untergeordneten Elemente

Beispiele

- *4,4"* Beispiel Abbildung
- *7,4"* Erweitertes Abbildungsbeispiel

3.13 Das Element <text>

Beschreibung:

Das Element `text` legt die Eigenschaften des enthaltenen Textes fest. Das Feldelement kann mehrere Textelemente enthalten. Die Schrifteigenschaften des Textelements werden vom Haupt-Image-Tag übernommen. Sollen mehrere Textelemente in ein Feld passen, muss das Attribut y-Position weggelassen werden. Dann wird ein Text unter den anderen positioniert.

Vorsicht beim Attribut Automatischer Abstand. Wird dynamische Positionierung verwendet, nutzt das erste Textelement den gesamten Platz des Feldes.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
padding-bottom	Definiert den Innenabstand in Pixeln von unten.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-left	Definiert den Innenabstand in Pixeln von links.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-right	Definiert den Innenabstand in Pixeln von rechts.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
padding-top	Definiert den Innenabstand in Pixeln von oben.	wahr	Ein positiver oder negativer Integer-Wert.	0	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten.	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-

Untergeordnete Elemente

<#text> siehe Kapitel *Das Element <#text>*

 siehe Kapitel *Das Element
*

 siehe Kapitel *Das Element *

 siehe Kapitel *Das Element <ul> (Ungeordnete Liste)*

 siehe Kapitel *Das Element *

Beispiele

- *4,4"* Beispiel Einzel- und Mehrfachzeilen

3.14 Das Element <label>

Beschreibung:

Das Element `label` definiert einen einzeiligen Text. Der enthaltene Text wird abgeschnitten, wenn er zu lang ist. Es erfolgt kein Zeilenumbruch.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
align	Legt die horizontale Ausrichtung des Elements/Textes fest.	wahr	Linksbündig | Rechtsbündig | Zentriert | Blocksatz	Linksbündig	-
padding-bottom	Definiert den Innenabstand in Pixeln von unten.	wahr	A positive or negative Integer-Wert	0	-
padding-left	Definiert den Innenabstand in Pixeln von links.	wahr	A positive or negative Integer-Wert	0	-
padding-right	Definiert den Innenabstand in Pixeln von rechts.	wahr	A positive or negative Integer-Wert	0	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
padding-top	Definiert den Innenabstand in Pixeln von oben.	wahr	A positive or negative Integer-Wert	0	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Fließzahlen, getrennt durch einen Doppelpunkt.	1,0	
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den	wahr	Eine positive oder negative Gleitkommazahl.	0,0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	Abstand zwischen den Buchstaben innerhalb eines Textes.				
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Dicke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixel.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-

Untergeordnete Elemente

<#text> siehe Kapitel *Das Element <#text>*

 siehe Kapitel *Das Element *

Beispiele

- *4,4"* Beispiel Einzel- und Mehrfachzeilen

3.15 Das Element

Beschreibung:

Das Element span ist innerhalb eines Textelements festgelegt, um den enthaltenen Text unterschiedlich zu formatieren.

Nach einem span-Element erfolgt kein automatischer Zeilenumbruch.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:**	wahr	Normal | Keine	Normal	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.				
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-

Untergeordnete Elemente

<#text> siehe Kapitel *Das Element <#text>*

Beispiele

- *4,4"* Span-Beispiele

3.16 Das Element <ul> (Ungeordnete Liste)

Beschreibung:

Das Element `ul` bestimmt eine ungeordnete Liste, vergleichbar mit dem HTML-Tag <ul>. Die Elemente, die als "li"-Elemente definiert sind, werden dann in eine ungeordnete Liste formatiert.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
wrap	Ändert den Zeilenumbruch eines Textfelds. **Normal:** Zeilenumbruch bei Leerzeichen und Bindestrichen. **Keine:** Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-
bullet-font-family	Legt die Schrift des Aufzählungszeichens fest.	falsch	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Kein Default-Wert	
bullet-font-family	Legt die Schrift des Aufzählungszeichens fest.	falsch	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Kein Default-Wert	
padded	Manuelles Festlegen des Abstands, in Pixeln	falsch	Ein positiver Integer-Wert (>0)	Kein Default-Wert	
type	Bestimmt welche Art von Aufzählungszeichen bei einer Aufzählung genutzt werden soll.	falsch	gefüllter Kreis (•) | Strich (-) | Punkt (.) | Doppelpunkt (:) | Klammer ()) | Punkt_Klammer (.)) | Kreis ( ) | Quadrat ( ) | gebräuchlichste Aufzählungszeichen	Strich	
float	Wird genutzt, um Elemente um eine Liste umfließen zu lassen. In einer umfließenden Liste sind Umbrüche wie in einem normalen Text, es gibt nicht nach jedem Element einen Zeilenumbruch. Der Zeilenumbruch in einer Liste ist nicht nach einem Aufzählungszeichen und nicht vor dem ersten Buchstaben.	falsch	wahr | falsch	falsch	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
wrapped-line-spacing	Bestimmt den Zeilenabstand für automatische Zeilenumbrüche innerhalb einer Liste (negative Werte verkleinern den Zeilenabstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
bullet-image	Pfad, der für ein Bild als Aufzählungszeichen genutzt wird.	falsch	Ein Pfad relativ zur Template-Datei oder zu einer URL (z.B. "../test/image.png oder url.to/image.png")	Kein Default-Wert	

Untergeordnete Elemente

 siehe Kapitel *Das Element *

Beispiele

- *4,4"* Beispiel Liste

3.17 Das Element

Beschreibung:

Das Element `ol` bestimmt eine geordnete Liste, vergleichbar mit dem HTML-Tag <ol>. Die Elemente, die als "li"- Elemente definiert sind, werden dann in eine geordnete Liste formatiert.

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	Dieser Identifier muss eindeutig sein.				
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Gleitkommazahl(en), getrennt durch einen Doppelpunkt.	1,0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
wrap	Ändert den Zeilenumbruch eines Textfelds. Normal: Zeilenumbruch bei Leerzeichen und Bindestichen. Keine: Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Stärke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
shadow-size	Die Größe des Schattens, in Pixeln.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-
padded	Manuelles Festlegen des Abstands, in Pixeln	falsch	Ein positiver Integer-Wert (>0)	Kein Default-Wert	
type	Bestimmt welche Art von Aufzählungszeichen bei einer Aufzählung genutzt werden soll.	falsch	gefüllter Kreis (•) | Strich (-) | Punkt (.) | Doppelpunkt (:) | Klammer ()) | Punkt_Klammert (.)) | Kreis, nur Rahmen ( ) | Quadrat ( ) | gebräuchlichste Aufzählungszeichen	.	
float	Wird genutzt, um Elemente um eine Liste umfließen zu lassen. In einer umfließenden Liste sind Umbrüche wie in einem normalen Text, es gibt nicht nach jedem Element einen Zeilenumbruch. Der Zeilenumbruch in einer Liste ist nicht nach einem Aufzählungszeichen und nicht vor dem ersten Buchstaben.	falsch	wahr | falsch	falsch	-
wrapped-line-spacing	Bestimmt den Zeilenabstand für automatische Zeilenumbrüche	wahr	Ein positiver oder negativer Integer-Wert.	0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	innerhalb einer Liste (negative Werte verkleinern den Zeilenabstand).				
bullet-image	Pfad, der für ein Bild als Aufzählungszeichen genutzt wird.	falsch	Ein Pfad relativ zur Template-Datei oder zu einer URL (z.B. "../test/image.png oder url.to/image.png")	Kein Default-Wert	

Untergeordnete Elemente

 siehe Kapitel *Das Element *

Beispiele

- *4,4"* Beispiel Liste

3.18 Das Element

Beschreibung:

Das Element `li` bestimmt ein Aufzählungslisten-Element, vergleichbar mit dem HTML-Tag "li".

Merkmale

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
class	Identifier der CSS-Klasse. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Der gleiche Klassen-Identifier soll bei Mehrfachtags angewendet werden.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108
id	Identifier eines Tags. Dieser Identifier kann mit Hilfe einer CSS-Datei referenziert werden. Dieser Identifier muss eindeutig sein.	falsch	String	Kein Default-Wert	*4,4"* auf Seite 108

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
font-family	Der Name der Schrift, die gewählt werden soll.	wahr	Ein Schriftname (z.B. Arial, Tahoma, Times New Roman, ect.)	Arial	-
background-color	Die Hintergrundfarbe des Elements.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Transparent	-
text-decoration	Definiert die Textdekoration für einen vorhandenen Text (unter- oder überstreichen, durchgestrichen oder keine).	wahr	Unterstrichen | Durchgestrichen | Keine	Keine	
font-size	Legt eine Reihe von Schriftgrößen fest, getrennt durch Doppelpunkt. Der Text wird mit der größtmöglichen (und festgelegten) Schriftgröße gerendert.	wahr	Ein positiver Integer gibt die Schriftgröße in pt an, oder mehrere Integer, getrennt durch einen Doppelpunkt als alternative Schriftgrößen.	12	-
font-type	Legt fest, ob ein Text normal oder kursiv gedruckt werden soll.	wahr	Normal | Kursiv	Normal	-
font-weight	Legt fest, ob ein Text normal oder fett gedruckt werden soll.	wahr	Normal | Fett	Normal	-
color	Die Farbe des Elements oder des vorhandenen Texts.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
condense	Ist der Text zu lang, wird er automatisch mit dem bestmöglichen Komprimierungsfaktor komprimiert.	wahr	Eine oder mehrere Fließzahlen, getrennt durch einen Doppelpunkt.	1,0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen, in Pixel (negative	wahr	Ein positiver oder negativer Integer-Wert.	0	

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
	Werte verringern den Abstand).				
wrap	Ändert den Zeilenumbruch eines Textfelds. Normal: Zeilenumbruch bei Leerzeichen und Bindestichen. Keine: Keine Zeilenumbrüche.	wahr	Normal | Keine	Normal	-
letter-spacing	Das Attribut Zeichenabstand vergrößert oder verkleinert den Abstand zwischen den Buchstaben innerhalb eines Textes.	wahr	Eine positive oder negative Gleitkommazahl.	0,0	
superscript	Wird zur Hochstellung oder Tiefstellung eines Textes verwendet.	wahr	Hochstellung | Tiefstellung | Normal	Normal	
outline	Legt fest, ob um einen Text eine Kontur gelegt werden soll.	wahr	wahr | falsch	falsch	-
outline-color	Bestimmt die Farbe der Textkontur.	wahr	schwarz | weiß | transparent | durch Semikolon getrennter RGBA String mit Werten zwischen 0 und 255	Schwarz	-
outline-thickness	Bestimmt die Dicke der Textkontur.	wahr	Ein positiver Integer-Wert (>0)	1	-
outline-inside	Dieser Wert bestimmt, ob die Kontur innerhalb oder außerhalb des entsprechenden Textes gelegt werden soll.	wahr	wahr | falsch	falsch	-
shadow	Aktiviert/deaktiviert Schatten	wahr	wahr | falsch	falsch	-
shadow-alpha	Der Alpha-Wert (Transparenz) des Schattens.	wahr	Eine positive Gleitkommazahl >= 0 und <= 1	0,5	-
shadow-color	Bestimmt die Farbe des Schattens.	wahr	schwarz | weiß | transparent | durch	Schwarz	-

Name	Beschreibung	Vererbt	Mögliche Werte	Default-Wert	Beispiele
			Semikolon getrennter RGBA String mit Werten zwischen 0 und 255		
shadow-size	Die Größe des Schattens, in Pixel.	wahr	Ein positiver Integer-Wert (>0)	2	-
shadow-blur	Ob ein Schatten mit oder ohne ein Weichzeichnen-Effekt gerendert werden soll.	wahr	wahr | falsch	falsch	-
wrapped-line-spacing	Bestimmt den Zeilenabstand für automatische Zeilenumbrüche innerhalb einer Liste (negative Werte verkleinern den Zeilenabstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	
line-spacing	Bestimmt den Abstand zwischen den Textzeilen in Pixel (negative Werte verkleinern den Abstand).	wahr	Ein positiver oder negativer Integer-Wert.	0	

Untergeordnete Elemente

<#text> siehe Kapitel *Das Element <#text>*

 siehe Kapitel *Das Element
*

 siehe Kapitel *Das Element *

Beispiele

- *4,4"* Beispiel Liste

3.19 Das Element

Beschreibung:

Das Element `br` ist innerhalb eines Textelements positioniert und erzwingt einen Zeilenumbruch. Es erfordert keine Attribute.

Mehrfaches Auftreten erlaubt: wahr

Attribute

Keine

Untergeordnete Elemente

Keine

3.20 Das Element <#text>

Beschreibung:

Klartext, der genutzt wird, um einen Barcode zu generieren oder der in einem generierten Bild angezeigt wird.

Mehrfaches Auftreten erlaubt: wahr

Attribute

Keine

Untergeordnete Elemente

Keine

4 Utils

Das Template Utils wurde erstellt, um Textinhalte für generierte Bilder verändern zu können.

Um solch eine Methode aufzurufen, muss es im Template definiert sein.

```
Utils Example
<utils method="toUpperCase">text</utils>
```

Als Ergebnis wird das generierte Bild „TEXT“ anstatt „text“ anzeigt.

Für den Fall, dass die Methode verschiedene Parameter beinhaltet, müssen diese als Attribut im Tag Utils definiert sein.

```
Utils Example with multiple arguments
<utils method="replace" arg1="x" arg2="s">text</utils>
```

Der Output dieser Methode ist „test“ anstatt „text“.

4.1 DefaultTemplateUtils

replace

Ersetzt alle Strings eines Textelements durch einen anderen String.

Tabelle 17: Parameter für replace

Name	Typ	Beschreibung
Textinhalt	String	Text für Suchen und Ersetzen.
arg1	String	Der String, nach dem gesucht werden soll.
arg2	String	Der String, der eingefügt werden soll.

contains

Prüft, ob ein Textelement einen gesuchten String enthält.

Tabelle 18: Parameter für contains

Name	Typ	Beschreibung
Textinhalt	String	Zu prüfender Text
arg1	String	Zu suchender Text

split

Teilt ein Textelement.

Tabelle 19: Parameter für split

Name	Typ	Beschreibung
Textinhalt	String	Zu trennender Text

Name	Typ	Beschreibung
arg1	String	Zu nutzendes Trennzeichen
arg2	String	Textstelle für Zeilenumbruch

toLowerCase

Stellt einen Textinhalt auf Kleinbuchstaben um.

Tabelle 20: Parameter für toLowerCase

Name	Typ	Beschreibung
Textinhalt	String	String zur Umwandlung in Kleinbuchstaben.

toUpperCase

Stellt einen Textinhalt auf Großbuchstaben um.

Tabelle 21: Parameter für toUpperCase

Name	Typ	Beschreibung
Textinhalt	String	String zur Umwandlung in Großbuchstaben.

format

Formatiert den vorgesehenen String mit dem festgelegten Format. Die Formatierung nutzt die Java-Implementierung der Methode `String.format()`.

Tabelle 22: Parameter für format

Name	Typ	Beschreibung
Textinhalt	String	Der String zum Formatieren
arg1	String	Der Format-String

lowerCase

Stellt einen Textinhalt auf Kleinbuchstaben um.

Tabelle 23: Parameter für lowerCase

Name	Typ	Beschreibung
Textinhalt	String	Der String für Kleinbuchstaben

capitalize

Stellt den ersten Buchstaben eines Textinhalts auf Großschreibung um. Keine Auswirkung auf die folgenden Buchstaben.

Tabelle 24: Parameter für capitalize

Name	Typ	Beschreibung
Textinhalt	String	Der String für Großschreibung

formatNumber

Formatiert die vorgesehene Zahl mit dem festgelegten Format. Die Formatierung nutzt die Java-Implementierung der Methode `String.format()`. Für weiterführende Informationen, siehe: http://docs.oracle.com/javase/tutorial/java/data/numberformat.html

Tabelle 25: Parameter für formatNumber

Name	Typ	Beschreibung
Textinhalt	String	Zu formatierende Zahl
arg1	String	Der Format-String

deleteWhitespace

Entfernt alle Leerzeichen eines Strings.

Tabelle 26: Parameterfür deleteWhitespace

Name	Typ	Beschreibung
Textinhalt	String	Der String, bei dem Leerzeichen gelöscht werden sollen

abbreviate

Kürzt einen String durch Verwenden von Auslassungen. Aus „Jetzt ist die Zeit gekommen für alle guten Menschen" wird „Jetzt ist die Zeit gekommen für...".

Tabelle 27: Parameter für abbreviate

Name	Typ	Beschreibung
Textinhalt	String	Zu kürzender Text
arg1	String	Maximale Länge des Ergebnis-Strings (Minimum 4)

getCurrentDate

Gibt das aktuelle Datum im festgelegten Format zurück.

Tabelle 28: Parameter für getCurrentDate

Name	Typ	Beschreibung
Textinhalt	String	Die Beschreibung des gewünschten Datumformats

getNonBreakingSpace

Gibt ein geschütztes Leerzeichen zurück, um automatische Umbrüche zu vermeiden (char code: 160).

Keine Parameter

getOperatingSystem

Gibt das aktuelle Betriebssystem zurück, auf dem der Wireless ePaper Server installiert ist. (WINDOWS|LINUX|MAC/ANDERE)

Keine Parameter

isFontAvailable

Gibt zurück, ob die vorgegebenen Schriftart im aktuellen System vorhanden ist (wahr | falsch).

Tabelle 29: Parameter für isFontAvailable

Name	Typ	Beschreibung
Textinhalt	String	Der Schriftname als String

getCurrentWorkingDirectory

Gibt das aktuelle Arbeitsverzeichnis zurück.

Keine Parameter

replaceUmlauts

Ersetzt alle Umlaute im angegebenen Textinhalt (ä->ae, ö->oe, ü->ue).

Tabelle 30: Parameter für replaceUmlauts

Name	Typ	Beschreibung
Textinhalt	String	Text für Suchen und Ersetzen

insertUmlauts

Ersetzt die ausgetauschten Umlaute wieder mit Umlauten (ae->ä, oe->ö, ue->ü).

Tabelle 31: Parameter für insertUmlauts

Name	Typ	Beschreibung
Textinhalt	String	Text für Suchen und Ersetzen

formatDate

Parst den vorhandenen Datum-String, indem das vorhande Eingabeformat durch das gewünschte Ausgabeformat ersetzt wird. y: Jahr, D: Tag des Jahres, M: Monat des Jahres, d: Tag des Monats, H: Stunde des Tages, m: Minute der Stunde, s: Sekunde der Minute, z: Zeitzone, E: Tag der Woche, w: Woche des Jahres

Tabelle 32: Parameter für formatDate

Name	Typ	Beschreibung
Textinhalt	String	Zu formatierender Datum-String
arg1	String	Das Format zum Parsen des Eingabedatum-Strings
arg2	String	Das Format für den Ausgabedatum-String

5 Beispiele

Im Folgenden wird anhand mehrerer Beispiele die Generierung von Grafiken auf Wireless ePaper Displays veranschaulicht. Die verwendeten Templates können für die Praxis eingesetzt werden.

5.1 2,7"

2,7" Beispielbild - Coca Cola

Abbildung 1: Generiertes Bild

- Template:

```
2.7" record example - Coca Cola (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image font-family="Arial" height="176" width="264" >
      <line thickness="5" x-from="0" x-to="264" y-from="2" y-to="2" >
      </line>
      <field height="81" width="175" x="5" y="10" >
        <label condense="1.0,0.8,0.6,0.5" font-size="34,33,32,30,28,26,24,22,20"
font-weight="bold" >
          <xsl:value-of select="article/field[@key='name']/@value" >
          </xsl:value-of>
        </label>
        <text condense="1.0,0.8,0.6,0.5" font-size="20,19,18,16,12" padding-top="5" >
          <xsl:value-of select="article/field[@key='description']/@value" >
          </xsl:value-of>
        </text>
        <text condense="1.0,0.8,0.6,0.5" font-size="12" padding-top="3" >
          <ul float="true" type="disc" >
            <li>
              <span font-weight="bold" >
                Einheit:
              </span>
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
              <xsl:value-of select="article/field[@key='pricePerUnit']/@value" >
              </xsl:value-of>
            </li>
          </ul>
        </text>
      </field>
      <field height="65" width="65" x="189" y="10" >
```

```
        <barcode autoscale="true" type="qr" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </barcode>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="92" y-to="92" >
      </line>
      <field height="79" valign="center" width="264" x="0" y="95" >
        <label align="center" condense="1.0,0.8,0.6,0.5"
font-size="70,68,66,64,62,60,58,56,54,52,50,48,46,44,42,40,38" font-style="italic"
font-weight="bold" >
          €
          <xsl:value-of select="article/field[@key='price']/@value" >
          </xsl:value-of>
        </label>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="173" y-to="173" >
      </line>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Coca Cola (Record)
<article articleNumber="5449000000996" >
  <field key="pricePerUnit" value="1.65" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.55" >
  </field>
  <field key="description" value="Dose 0,33 l" >
  </field>
  <field key="name" value="Coca Cola" >
  </field>
</article>
```

2,7" Beispielbild - Chili con carne

Abbildung 2: Generiertes Bild

- Template:

```
2.7" record example - Chili con carne (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image font-family="Arial" height="176" width="264" >
      <line thickness="5" x-from="0" x-to="264" y-from="2" y-to="2" >
      </line>
      <field height="81" width="175" x="5" y="10" >
        <label condense="1.0,0.8,0.6,0.5" font-size="34,33,32,30,28,26,24,22,20"
font-weight="bold" >
```

```
          <xsl:value-of select="article/field[@key='name']/@value" >
          </xsl:value-of>
        </label>
        <text condense="1.0,0.8,0.6,0.5" font-size="20,19,18,16,12" padding-top="5" >
          <xsl:value-of select="article/field[@key='description']/@value" >
          </xsl:value-of>
        </text>
        <text condense="1.0,0.8,0.6,0.5" font-size="12" padding-top="3" >
          <ul float="true" type="disc" >
            <li>
              <span font-weight="bold" >
                Einheit:
              </span>
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
              <xsl:value-of select="article/field[@key='pricePerUnit']/@value" >
              </xsl:value-of>
            </li>
          </ul>
        </text>
      </field>
      <field height="65" width="65" x="189" y="10" >
        <barcode autoscale="true" type="qr" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </barcode>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="92" y-to="92" >
      </line>
      <field height="79" valign="center" width="264" x="0" y="95" >
        <label align="center" condense="1.0,0.8,0.6,0.5"
font-size="70,68,66,64,62,60,58,56,54,52,50,48,46,44,42,40,38" font-style="italic"
font-weight="bold" >
          €
          <xsl:value-of select="article/field[@key='price']/@value" >
          </xsl:value-of>
        </label>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="173" y-to="173" >
      </line>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Chili con carne (Record)
<article articleNumber="9000275639319" >
  <field key="pricePerUnit" value="4.98" >
  </field>
  <field key="unit" value="1 kg" >
  </field>
  <field key="price" value="2.49" >
  </field>
  <field key="description" value="Dose 500 g" >
  </field>
  <field key="name" value="Chili con Carne" >
  </field>
</article>
```

2,7" Beispielbild - Red Bull

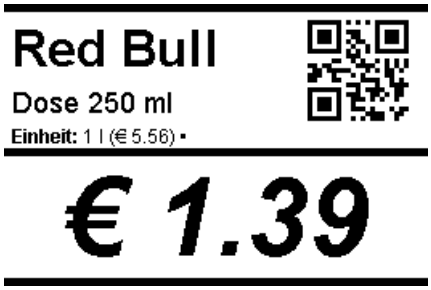


Abbildung 3: Generiertes Bild

- Template:

```
2.7" record example - Red Bull (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image font-family="Arial" height="176" width="264" >
      <line thickness="5" x-from="0" x-to="264" y-from="2" y-to="2" >
      </line>
      <field height="81" width="175" x="5" y="10" >
        <label condense="1.0,0.8,0.6,0.5" font-size="34,33,32,30,28,26,24,22,20"
font-weight="bold" >
          <xsl:value-of select="article/field[@key='name']/@value" >
          </xsl:value-of>
        </label>
        <text condense="1.0,0.8,0.6,0.5" font-size="20,19,18,16,12" padding-top="5" >
          <xsl:value-of select="article/field[@key='description']/@value" >
          </xsl:value-of>
        </text>
        <text condense="1.0,0.8,0.6,0.5" font-size="12" padding-top="3" >
          <ul float="true" type="disc" >
            <li>
              <span font-weight="bold" >
                Einheit:
              </span>
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
              <xsl:value-of select="article/field[@key='pricePerUnit']/@value" >
              </xsl:value-of>
            </li>
          </ul>
        </text>
      </field>
      <field height="65" width="65" x="189" y="10" >
        <barcode autoscale="true" type="qr" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </barcode>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="92" y-to="92" >
      </line>
      <field height="79" valign="center" width="264" x="0" y="95" >
        <label align="center" condense="1.0,0.8,0.6,0.5"
font-size="70,68,66,64,62,60,58,56,54,52,50,48,46,44,42,40,38" font-style="italic"
font-weight="bold" >
          €
          <xsl:value-of select="article/field[@key='price']/@value" >
          </xsl:value-of>
        </label>
      </field>
```

```
      <line thickness="5" x-from="0" x-to="264" y-from="173" y-to="173" >
      </line>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Red Bull (Record)
<article articleNumber="9002490100070" >
  <field key="pricePerUnit" value="5.56" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="1.39" >
  </field>
  <field key="description" value="Dose 250 ml" >
  </field>
  <field key="name" value="Red Bull" >
  </field>
  <field key="sale" value="1" >
  </field>
</article>
```

2,7" Beispielbild - Vöslauer

Abbildung 4: Generiertes Bild

- Template:

```
2.7" record example - Vöslauer (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image font-family="Arial" height="176" width="264" >
      <line thickness="5" x-from="0" x-to="264" y-from="2" y-to="2" >
      </line>
      <field height="81" width="175" x="5" y="10" >
        <label condense="1.0,0.8,0.6,0.5" font-size="34,33,32,30,28,26,24,22,20"
font-weight="bold" >
          <xsl:value-of select="article/field[@key='name']/@value" >
          </xsl:value-of>
        </label>
        <text condense="1.0,0.8,0.6,0.5" font-size="20,19,18,16,12" padding-top="5" >
          <xsl:value-of select="article/field[@key='description']/@value" >
          </xsl:value-of>
        </text>
        <text condense="1.0,0.8,0.6,0.5" font-size="12" padding-top="3" >
          <ul float="true" type="disc" >
            <li>
              <span font-weight="bold" >
                Einheit:
              </span>
              <xsl:value-of select="article/field[@key='unit']/@value" >
```

```
              </xsl:value-of>
              <xsl:value-of select="article/field[@key='pricePerUnit']/@value" >
              </xsl:value-of>
            </li>
          </ul>
        </text>
      </field>
      <field height="65" width="65" x="189" y="10" >
        <barcode autoscale="true" type="qr" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </barcode>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="92" y-to="92" >
      </line>
      <field height="79" valign="center" width="264" x="0" y="95" >
        <label align="center" condense="1.0,0.8,0.6,0.5"
font-size="70,68,66,64,62,60,58,56,54,52,50,48,46,44,42,40,38" font-style="italic"
font-weight="bold" >
          €
          <xsl:value-of select="article/field[@key='price']/@value" >
          </xsl:value-of>
        </label>
      </field>
      <line thickness="5" x-from="0" x-to="264" y-from="173" y-to="173" >
      </line>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Vöslauer (Record)
<article articleNumber="9009700145104" >
  <field key="pricePerUnit" value="0.35" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.35" >
  </field>
  <field key="description" value="Vöslauer prickelnd, 1 l PET" >
  </field>
  <field key="name" value="Vöslauer Mineralwasser" >
  </field>
</article>
```

2,7" Beispielbild - Chili con carne

Abbildung 5: Generiertes Bild

- Template:

```
2.7" record example - Chili con carne (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <xsl:variable name="price" >
      <xsl:value-of select="article/field[@key='price']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="name" >
      <xsl:value-of select="article/field[@key='name']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="description" >
      <xsl:value-of select="article/field[@key='description']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <image height="176" width="264" font-family="Verdana" >
      <field height="80" width="225" x="5" y="7" >
        <text font-size="16" font-weight="bold" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$name" >
          </xsl:value-of>
        </text>
        <text font-size="14" font-weight="normal" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$description" >
          </xsl:value-of>
        </text>
      </field>
      <field height="100" width="140" x="0" y="75" >
        <text align="right" font-family="Lucida Sans" font-size="80" font-weight="bold"
condense="1.0" >
          <xsl:value-of select="substring-before($price, '.')" >
          </xsl:value-of>
        </text>
      </field>
      <field height="80" width="20" x="135" y="95" >
       <text align="center" font-family="Lucida Sans" font-size="60" font-weight="bold"
condense="1.0" >
          .
        </text>
      </field>
      <field height="90" width="80" x="135" y="79" >
        <label align="left" font-family="Lucida Sans" font-size="52" font-weight="bold"
condense="0.9" >
          <xsl:value-of select="substring-after($price, '.')" >
          </xsl:value-of>
        </label>
      </field>
     <field align="center" valign="center" rotation="270" height="25" width="150" x="230"
 y="17" >
        <barcode scale="1" humanReadableEnabled="false" type="code128" >
          1234567890ABCD
        </barcode>
      </field>
      <field height="14" width="100" y="160" x="5" >
        <label align="left" font-size="10" font-weight="normal" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </label>
      </field>
    </image>
```

```
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Chili con carne (Record)
<article articleNumber="9000275639319" >
  <field key="pricePerUnit" value="4.98" >
  </field>
  <field key="unit" value="1 kg" >
  </field>
  <field key="price" value="2.49" >
  </field>
  <field key="description" value="Dose 500 g" >
  </field>
  <field key="name" value="Chili con Carne" >
  </field>
</article>
```

2,7" Beispielbild - Red Bull

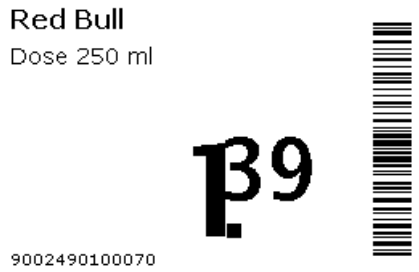


Abbildung 6: Generiertes Bild

- Template:

```
2.7" record example - Red Bull (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <xsl:variable name="price" >
      <xsl:value-of select="article/field[@key='price']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="name" >
      <xsl:value-of select="article/field[@key='name']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="description" >
      <xsl:value-of select="article/field[@key='description']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <image height="176" width="264" font-family="Verdana" >
      <field height="80" width="225" x="5" y="7" >
        <text font-size="16" font-weight="bold" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$name" >
          </xsl:value-of>
        </text>
        <text font-size="14" font-weight="normal" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$description" >
          </xsl:value-of>
        </text>
      </field>
```

```
      <field height="100" width="140" x="0" y="75" >
        <text align="right" font-family="Lucida Sans" font-size="80" font-weight="bold"
condense="1.0" >
          <xsl:value-of select="substring-before($price, '.')" >
          </xsl:value-of>
        </text>
      </field>
      <field height="80" width="20" x="135" y="95" >
        <text align="center" font-family="Lucida Sans" font-size="60" font-weight="bold"
condense="1.0" >
          .
        </text>
      </field>
      <field height="90" width="80" x="135" y="79" >
        <label align="left" font-family="Lucida Sans" font-size="52" font-weight="bold"
condense="0.9" >
          <xsl:value-of select="substring-after($price, '.')" >
          </xsl:value-of>
        </label>
      </field>
     <field align="center" valign="center" rotation="270" height="25" width="150" x="230"
 y="17" >
        <barcode scale="1" humanReadableEnabled="false" type="code128" >
          1234567890ABCD
        </barcode>
      </field>
      <field height="14" width="100" y="160" x="5" >
        <label align="left" font-size="10" font-weight="normal" >
          <xsl:value-of select="article/@articleNumber" >
          </xsl:value-of>
        </label>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Red Bull (Record)
<article articleNumber="9002490100070" >
  <field key="pricePerUnit" value="5.56" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="1.39" >
  </field>
  <field key="description" value="Dose 250 ml" >
  </field>
  <field key="name" value="Red Bull" >
  </field>
  <field key="sale" value="1" >
  </field>
</article>
```

2,7" Beispielbild - Vöslauer

Abbildung 7: Generiertes Bild

- Template:

```
2.7" record example - Vöslauer (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <xsl:variable name="price" >
      <xsl:value-of select="article/field[@key='price']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="name" >
      <xsl:value-of select="article/field[@key='name']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="description" >
      <xsl:value-of select="article/field[@key='description']/@value" >
      </xsl:value-of>
    </xsl:variable>
    <image height="176" width="264" font-family="Verdana" >
      <field height="80" width="225" x="5" y="7" >
        <text font-size="16" font-weight="bold" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$name" >
          </xsl:value-of>
        </text>
        <text font-size="14" font-weight="normal" condense="1.0, 0.9, 0.8, 0.7"
line-spacing="-1" padding-bottom="5" >
          <xsl:value-of select="$description" >
          </xsl:value-of>
        </text>
      </field>
      <field height="100" width="140" x="0" y="75" >
        <text align="right" font-family="Lucida Sans" font-size="80" font-weight="bold"
condense="1.0" >
          <xsl:value-of select="substring-before($price, '.')" >
          </xsl:value-of>
        </text>
      </field>
      <field height="80" width="20" x="135" y="95" >
        <text align="center" font-family="Lucida Sans" font-size="60" font-weight="bold"
condense="1.0" >
          .
        </text>
      </field>
      <field height="90" width="80" x="135" y="79" >
        <label align="left" font-family="Lucida Sans" font-size="52" font-weight="bold"
condense="0.9" >
          <xsl:value-of select="substring-after($price, '.')" >
          </xsl:value-of>
        </label>
```

```
      </field>
     <field align="center" valign="center" rotation="270" height="25" width="150" x="230"
 y="17" >
       <barcode scale="1" humanReadableEnabled="false" type="code128" >
         1234567890ABCD
       </barcode>
     </field>
     <field height="14" width="100" y="160" x="5" >
       <label align="left" font-size="10" font-weight="normal" >
         <xsl:value-of select="article/@articleNumber" >
         </xsl:value-of>
       </label>
     </field>
   </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
2.7" record example - Vöslauer (Record)
<article articleNumber="9009700145104" >
  <field key="pricePerUnit" value="0.35" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.35" >
  </field>
  <field key="description" value="Vöslauer prickelnd, 1 l PET" >
  </field>
  <field key="name" value="Vöslauer Mineralwasser" >
  </field>
</article>
```

2,7" Beispiel verschachtelte Felder

Ein 2,7"-Template, das die Verwendung von verschachtelten Feldern zeigt.

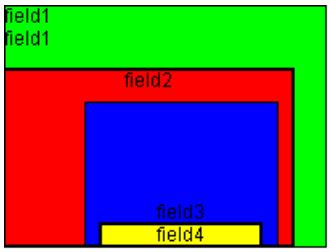


Abbildung 8: Generiertes Bild

- Template:

```
2,7" nested field example (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <image height="176" width="264" colors="true" >
      <field x="0" y="0" width="200" height="150" align="left" border="1"
background-color="green" >
       <text>
         field1
       </text>
       <text>
         field1
       </text>
       <field x="0" y="40" width="180" height="110" border="2" align="center"
```

```
background-color="red" >
          <text>
            field2
          </text>
          <field x="50" y="20" width="120" height="90" border="1" align="center"
valign="bottom" background-color="blue" >
            <text>
              field3
            </text>
            <field x="0" width="100" max-height="30" border="2" align="center"
background-color="yellow" >
              <text>
                field4
              </text>
            </field>
          </field>
        </field>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Rendern von Grafiken für Aktions- bzw. nicht-Aktions-Artikel

Dieses Beispiel zeigt, wie dynamische Einstellungen angewendet werden können, um verschiedene Grafiken für Produkte als nicht-Aktions- bzw. Aktions-Artikel zu rendern.

Abbildung 9: Generiertes Bild

- Template (nicht-Aktions-Artikel):

```
Dynamic Sale/No Sale Image Rendering (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <image height="176" width="264" >
      <xsl:choose>
        <xsl:when test="article/field[@key='Sale']/@value='1'" >
          <rect color="black" height="126" width="264" x="0" y="0" >
          </rect>
          <field height="37" width="90" x="160" y="10" >
           <text color="white" font-family="Comic Sans MS" font-size="26" font-weight="bold"
 >
              Aktion!
            </text>
          </field>
        </xsl:when>
        <xsl:otherwise>
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
        </xsl:otherwise>
      </xsl:choose>
      <field height="40" width="150" x="9" y="10" >
        <xsl:choose>
```

```
          <xsl:when test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="width" >
              150
            </xsl:attribute>
          </xsl:when>
          <xsl:otherwise>
            <xsl:attribute name="width" >
              240
            </xsl:attribute>
          </xsl:otherwise>
        </xsl:choose>
        <text font-family="Tahoma" font-size="14" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <span font-weight="bold" >
            <utils method="toUpperCase" >
              <xsl:value-of select="article/field[@key='ProductDescription']/@value" >
              </xsl:value-of>
            </utils>
          </span>
        </text>
        <label font-family="Tahoma" font-size="16" padding-top="3" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="article/field[@key='ProductName']/@value" >
          </xsl:value-of>
        </label>
      </field>
      <field height="20" width="50" x="10" y="100" >
        <text align="left" font-family="Verdana" font-size="13" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="article/field[@key='TotalAmount']/@value" >
          </xsl:value-of>
        </text>
      </field>
      <field height="83" width="190" x="60" y="50" >
        <text align="right" condense="1,0.8,0.6,0.4" font-family="Impact"
font-size="68,64,60,58,56" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:if test="article/field[@key='MainPrice']/@value!=''" >
           <xsl:value-of select="floor(number(article/field[@key='MainPrice']/@value) div
 100)" >
            </xsl:value-of>
           <xsl:value-of select="number(article/field[@key='MainPrice']/@value) mod 100"
 >
            </xsl:value-of>
          </xsl:if>
        </text>
      </field>
      <field height="14" width="70" x="180" y="131" >
```

```
       <text align="right" font-family="Verdana" font-size="11" >
         <xsl:if test="article/field[@key='UsedUnit']/@value!='' and
article/field[@key='UsedUnit']/@value!='0'" >
           <span>
             per
           </span>
           <xsl:if test="article/field[@key='UsedDimension']/@value='2'" >
             <span>
               100
             </span>
           </xsl:if>
         </xsl:if>
         <span>
           <xsl:choose>
             <xsl:when test="article/field[@key='UsedUnit']/@value='1'" >
               g
             </xsl:when>
             <xsl:when test="article/field[@key='UsedUnit']/@value='2'" >
               kg
             </xsl:when>
             <xsl:when test="article/field[@key='UsedUnit']/@value='3'" >
               l
             </xsl:when>
             <xsl:when test="article/field[@key='UsedUnit']/@value='4'" >
               ml
             </xsl:when>
           </xsl:choose>
         </span>
       </text>
     </field>
     <field height="24" width="70" x="180" y="146" >
       <text align="right" font-family="Tahoma" font-size="20" >
         <xsl:if test="article/field[@key='ReferencePrice']/@value!='" >
          <xsl:value-of select="floor(number(article/field[@key='ReferencePrice']/@value)
 div 100)" >
           </xsl:value-of>
          <xsl:value-of select="number(article/field[@key='ReferencePrice']/@value) mod
 100" >
           </xsl:value-of>
         </xsl:if>
       </text>
     </field>
     <field align="center" height="20" width="160" x="10" y="136" >
       <barcode autoscale="false" humanReadableEnabled="false" type="code128" >
         <xsl:value-of select="Label/Id" >
         </xsl:value-of>
       </barcode>
     </field>
     <field height="12" width="160" x="10" y="158" >
       <text align="center" font-family="Tahoma" font-size="10" >
         <span>
           <xsl:value-of select="number(article/field[@key='EAN']/@value)" >
           </xsl:value-of>
         </span>
         <span>
           <xsl:value-of select="article/field[@key='Location']/@value" >
           </xsl:value-of>
         </span>
         <xsl:if test="article/field[@key='Clearance']/@value='1'" >
           <span>
           </span>
           <span font-weight="bold" >
             R
           </span>
```

```
          </xsl:if>
          <span>
          </span>
          <span>
            <xsl:value-of select="article/field[@key='UpdateTime']/@value" >
            </xsl:value-of>
          </span>
        </text>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic Sale/No Sale Image Rendering (Record)
<article>
  <field key="LineID" value="1234567890" >
  </field>
  <field key="Version" value="1" >
  </field>
  <field key="Operator" value="1" >
  </field>
  <field key="CostCenter" value="1" >
  </field>
  <field key="NAN" value="0000000412342" >
  </field>
  <field key="EAN" value="3330001299874" >
  </field>
  <field key="MainPrice" value="999" >
  </field>
  <field key="UsedUnit" value="1" >
  </field>
  <field key="UsedDimension" value="2" >
  </field>
  <field key="ReferencePrice" value="1998" >
  </field>
  <field key="ProductDescription" value="Extra big schoko" >
  </field>
  <field key="ProductName" value="Corny" >
  </field>
  <field key="TotalAmount" value="50 g" >
  </field>
  <field key="Sale" value="0" >
  </field>
  <field key="Clearance" value="1" >
  </field>
  <field key="Location" value="08L03b" >
  </field>
  <field key="UpdateTime" value="135200" >
  </field>
  <field key="TagSize" value="1" >
  </field>
  <field key="MultiLocation" value="" >
  </field>
  <field key="Reserved" value="" >
```

```
  </field>
</article>
```

Abbildung 10: Generiertes Bild

- Template (Aktions-Artikel):

```
Dynamic Sale/No Sale Image Rendering (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <image height="176" width="264" >
      <xsl:choose>
        <xsl:when test="article/field[@key='Sale']/@value='1'" >
          <rect color="black" height="126" width="264" x="0" y="0" >
          </rect>
          <field height="37" width="90" x="160" y="10" >
           <text color="white" font-family="Comic Sans MS" font-size="26" font-weight="bold"
 >
              Aktion!
            </text>
          </field>
        </xsl:when>
        <xsl:otherwise>
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
        </xsl:otherwise>
      </xsl:choose>
      <field height="40" width="150" x="9" y="10" >
        <xsl:choose>
          <xsl:when test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="width" >
              150
            </xsl:attribute>
          </xsl:when>
          <xsl:otherwise>
            <xsl:attribute name="width" >
              240
            </xsl:attribute>
          </xsl:otherwise>
        </xsl:choose>
        <text font-family="Tahoma" font-size="14" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <span font-weight="bold" >
            <utils method="toUpperCase" >
              <xsl:value-of select="article/field[@key='ProductDescription']/@value" >
              </xsl:value-of>
            </utils>
          </span>
        </text>
```

```
        <label font-family="Tahoma" font-size="16" padding-top="3" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="article/field[@key='ProductName']/@value" >
          </xsl:value-of>
        </label>
      </field>
      <field height="20" width="50" x="10" y="100" >
        <text align="left" font-family="Verdana" font-size="13" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="article/field[@key='TotalAmount']/@value" >
          </xsl:value-of>
        </text>
      </field>
      <field height="83" width="190" x="60" y="50" >
        <text align="right" condense="1,0.8,0.6,0.4" font-family="Impact"
font-size="68,64,60,58,56" >
          <xsl:if test="article/field[@key='Sale']/@value='1'" >
            <xsl:attribute name="color" >
              white
            </xsl:attribute>
          </xsl:if>
          <xsl:if test="article/field[@key='MainPrice']/@value!=''" >
           <xsl:value-of select="floor(number(article/field[@key='MainPrice']/@value) div
 100)" >
            </xsl:value-of>
           <xsl:value-of select="number(article/field[@key='MainPrice']/@value) mod 100"
 >
            </xsl:value-of>
          </xsl:if>
        </text>
      </field>
      <field height="14" width="70" x="180" y="131" >
        <text align="right" font-family="Verdana" font-size="11" >
          <xsl:if test="article/field[@key='UsedUnit']/@value!='' and
article/field[@key='UsedUnit']/@value!='0'" >
            <span>
              per
            </span>
            <xsl:if test="article/field[@key='UsedDimension']/@value='2'" >
              <span>
                100
              </span>
            </xsl:if>
          </xsl:if>
          <span>
            <xsl:choose>
              <xsl:when test="article/field[@key='UsedUnit']/@value='1'" >
                g
              </xsl:when>
              <xsl:when test="article/field[@key='UsedUnit']/@value='2'" >
                kg
              </xsl:when>
              <xsl:when test="article/field[@key='UsedUnit']/@value='3'" >
                l
              </xsl:when>
              <xsl:when test="article/field[@key='UsedUnit']/@value='4'" >
```

```
              ml
            </xsl:when>
          </xsl:choose>
        </span>
      </text>
    </field>
    <field height="24" width="70" x="180" y="146" >
      <text align="right" font-family="Tahoma" font-size="20" >
        <xsl:if test="article/field[@key='ReferencePrice']/@value!='" >
         <xsl:value-of select="floor(number(article/field[@key='ReferencePrice']/@value)
div 100)" >
          </xsl:value-of>
         <xsl:value-of select="number(article/field[@key='ReferencePrice']/@value) mod
100" >
          </xsl:value-of>
        </xsl:if>
      </text>
    </field>
    <field align="center" height="20" width="160" x="10" y="136" >
      <barcode autoscale="false" humanReadableEnabled="false" type="code128" >
        <xsl:value-of select="Label/Id" >
        </xsl:value-of>
      </barcode>
    </field>
    <field height="12" width="160" x="10" y="158" >
      <text align="center" font-family="Tahoma" font-size="10" >
        <span>
          <xsl:value-of select="number(article/field[@key='EAN']/@value)" >
          </xsl:value-of>
        </span>
        <span>
          <xsl:value-of select="article/field[@key='Location']/@value" >
          </xsl:value-of>
        </span>
        <xsl:if test="article/field[@key='Clearance']/@value='1'" >
          <span>
          </span>
          <span font-weight="bold" >
            R
          </span>
        </xsl:if>
        <span>
        </span>
        <span>
          <xsl:value-of select="article/field[@key='UpdateTime']/@value" >
          </xsl:value-of>
        </span>
      </text>
    </field>
  </image>
 </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic Sale/No Sale Image Rendering (Record)
<article>
 <field key="LineID" value="1234567890" >
 </field>
 <field key="Version" value="1" >
 </field>
 <field key="Operator" value="1" >
 </field>
 <field key="CostCenter" value="1" >
 </field>
```

```
  <field key="NAN" value="0000000412342" >
  </field>
  <field key="EAN" value="3330001299874" >
  </field>
  <field key="MainPrice" value="799" >
  </field>
  <field key="UsedUnit" value="1" >
  </field>
  <field key="UsedDimension" value="2" >
  </field>
  <field key="ReferencePrice" value="1598" >
  </field>
  <field key="ProductDescription" value="Extra big schoko" >
  </field>
  <field key="ProductName" value="Corny" >
  </field>
  <field key="TotalAmount" value="50 g" >
  </field>
  <field key="Sale" value="1" >
  </field>
  <field key="Clearance" value="0" >
  </field>
  <field key="Location" value="08L03b" >
  </field>
  <field key="UpdateTime" value="135200" >
  </field>
  <field key="TagSize" value="1" >
  </field>
  <field key="MultiLocation" value="" >
  </field>
  <field key="Reserved" value="" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 2,7"

Dieses Beispiel basiert auf einem dynamischen Template und dem Coca Cola-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

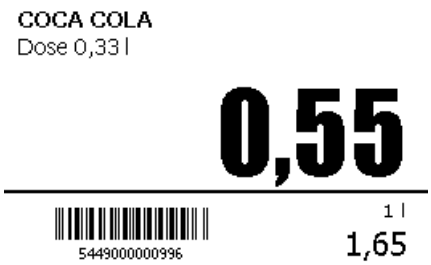


Abbildung 11: Generiertes Bild

- Template:

```
Dynamic template using default record - 2.7" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
```

```
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
','')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
```

```
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
             <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="728" >
```

```
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 2.7" (Record)
<article articleNumber="5449000000996" >
  <field key="pricePerUnit" value="1.65" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.55" >
  </field>
  <field key="description" value="Dose 0,33 l" >
  </field>
  <field key="name" value="Coca Cola" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 2,7"

Dieses Beispiel basiert auf einem dynamischen Template und dem Chili Con Carne-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 12: Generiertes Bild

- Template:

```
Dynamic template using default record - 2.7" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
```

```
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
```

```
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 2.7" (Record)
<article articleNumber="9000275639319" >
  <field key="pricePerUnit" value="4.98" >
  </field>
  <field key="unit" value="1 kg" >
  </field>
  <field key="price" value="2.49" >
  </field>
  <field key="description" value="Dose 500 g" >
  </field>
  <field key="name" value="Chili con Carne" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 2,7"

Dieses Beispiel basiert auf einem dynamischen Template und dem Red Bull-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

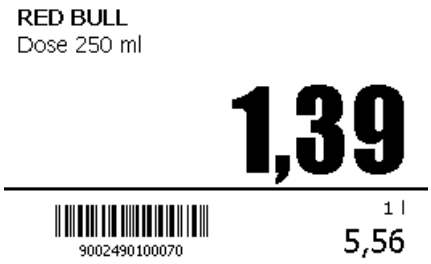


Abbildung 13: Generiertes Bild

- Template:

```
Dynamic template using default record - 2.7" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
```

```
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
```

```
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 2.7" (Record)
<article articleNumber="9002490100070" >
  <field key="pricePerUnit" value="5.56" >
```

```
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="1.39" >
  </field>
  <field key="description" value="Dose 250 ml" >
  </field>
  <field key="name" value="Red Bull" >
  </field>
  <field key="sale" value="1" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 2,7"

Dieses Beispiel basiert auf einem dynamischen Template und dem Vöslauer-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 14: Generiertes Bild

- Template:

```
Dynamic template using default record - 2.7" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
```

```
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
             <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
             <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
```

```
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
```

```
            </xsl:value-of>
          </text>
        </field>
      </image>
    </xsl:when>
  </xsl:choose>
 </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 2.7" (Record)
<article articleNumber="9009700145104" >
 <field key="pricePerUnit" value="0.35" >
 </field>
 <field key="unit" value="1 l" >
 </field>
 <field key="price" value="0.35" >
 </field>
 <field key="description" value="Vöslauer prickelnd, 1 l PET" >
 </field>
 <field key="name" value="Vöslauer Mineralwasser" >
 </field>
</article>
```

Angepasste Grafik - 2,7"

Abbildung 15: Generiertes Bild

- Template:

```
Custom Record 2.7" Example (Template)
<xsl:stylesheet version="2.0" >
 <xsl:template match="Record" >
  <xsl:variable name="price" >
   <xsl:value-of select="translate(Article/Price,'.','')" >
   </xsl:value-of>
  </xsl:variable>
  <image height="176" width="264" >
   <field height="20" width="256" x="8" y="8" >
    <text font-family="Verdana" font-size="10" text-decoration="underline" >
     <xsl:value-of select="Article/Category" >
     </xsl:value-of>
    </text>
   </field>
   <field height="19" width="256" x="8" y="28" >
    <text font-family="Verdana" font-size="16" font-weight="bold" >
     <utils method="toUpperCase" >
      <xsl:value-of select="Article/Manufacturer" >
      </xsl:value-of>
     </utils>
    </text>
   </field>
```

```
      <field height="19" width="256" x="8" y="48" >
        <text font-family="Verdana" font-size="16" >
          <xsl:value-of select="Article/Name" >
          </xsl:value-of>
        </text>
      </field>
      <xsl:choose>
        <xsl:when test="substring-after($price,',') = '00'" >
          <field height="78" width="250" x="0" y="83" >
            <text align="right" font-family="Tahoma" font-size="65" font-style="italic"
font-weight="bold" >
              <xsl:value-of select="substring-before($price,',')" >
              </xsl:value-of>
            </text>
          </field>
        </xsl:when>
        <xsl:otherwise>
          <field height="78" width="210" x="0" y="83" >
            <text align="right" font-family="Tahoma" font-size="65" font-style="italic"
font-weight="bold" >
              <xsl:value-of select="substring-before($price,',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="78" width="64" x="200" y="83" >
            <text align="left" font-family="Tahoma" font-size="35" font-style="italic"
font-weight="bold" >
              <xsl:value-of select="substring-after($price,',')" >
              </xsl:value-of>
            </text>
          </field>
        </xsl:otherwise>
      </xsl:choose>
      <field height="8" width="105" x="4" y="165" >
        <barcode humanReadableEnabled="false" type="code128" >
          <xsl:value-of select="Article/ArticleNumber" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="13" width="60" x="3" y="153" >
        <text font-family="Tahoma" font-size="11" >
          <xsl:value-of select="Article/ArticleNumber" >
          </xsl:value-of>
        </text>
      </field>
      <field align="right" height="8" width="146" x="114" y="165" >
        <barcode humanReadableEnabled="false" type="code128" >
          <xsl:value-of select="Label/Id" >
          </xsl:value-of>
        </barcode>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Custom Record 2.7" Example (Record)
<Article>
  <ArticleNumber>
    1183821
  </ArticleNumber>
  <Name>
    Coolpix S9200 Silber
  </Name>
```

```
  <Manufacturer>
    Nikon
  </Manufacturer>
  <Category>
    Digitalkamera
  </Category>
  <Price>
    129,90
  </Price>
  <Specification>
    <Field>
      Fotoauflösung: 16 Megapixel max.
    </Field>
    <Field>
      Zoom-Bereich: 25 bis 450 mm
    </Field>
    <Field>
      Speichermedium: SD/SDHC/SDXC
    </Field>
    <Field>
      Mögliche Dateiformate: JPEG, MOV, MPEG4
    </Field>
    <Field>
      Videoauflösung: 1980 x 1080 Pixel
    </Field>
    <Field>
      Zoomfaktor: 18-fach
    </Field>
    <Field>
      optischer Bildstabilisator
    </Field>
    <Field>
      Lichtstärke: F 1:3,5 bis 5,9
    </Field>
    <Field>
      Lichtempfindlichkeit: 3200 ISO
    </Field>
  </Specification>
  <Packaging>
    <Item>
      Trageschlaufe
    </Item>
    <Item>
      USB-Kabel
    </Item>
    <Item>
      Netzadapter
    </Item>
  </Packaging>
  <Accessories>
    <Item>
      Akku EN-EL12
    </Item>
    <Item>
      SDHC-Karte
    </Item>
  </Accessories>
</Article>
```

5.2 4,4"

Beispiel Liste

Ein Beispiel, das die verschiedenen Verwendungsmöglichkeiten des Listen-Tags zeigt.

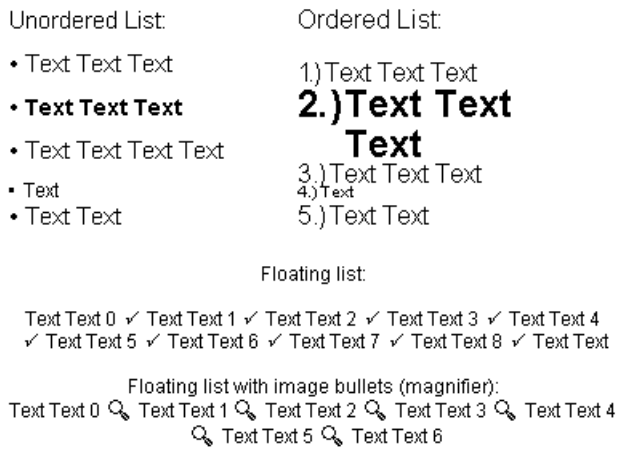


Abbildung 16: Generiertes Bild

- Template:

```
List Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field height="150" width="150" x="10" y="10" >
        <text font-family="Arial" font-size="15" line-spacing="10" >
          Unordered List:
          <ul line-spacing="10" spacing="5" type="disc" wrapped-line-spacing="1" >
            <li>
              Text Text Text
            </li>
            <li>
              <span font-size="18,16,14" font-weight="bold" >
                Text Text Text
              </span>
            </li>
            <li wrapped-line-spacing="15" >
              Text Text Text Text
            </li>
            <li font-size="12" line-spacing="0" >
              Text
            </li>
            <li>
              Text Text
            </li>
          </ul>
        </text>
      </field>
      <field height="150" width="150" x="190" y="10" >
        <text font-family="Arial" font-size="16" line-spacing="15" >
          Ordered List:
          <ol line-spacing="-4" spacing="2" type="DOT_BRACKET" wrapped-line-spacing="1" >
```

```
          <li>
            Text Text Text
          </li>
          <li font-size="24,22,20,18" font-weight="bold" wrapped-line-spacing="-3" >
            Text Text Text
          </li>
          <li wrapped-line-spacing="20" >
            Text Text Text
          </li>
          <li font-size="10" line-spacing="0" >
            Text
          </li>
          <li>
            Text Text
          </li>
        </ol>
      </text>
    </field>
    <field height="56" width="380" x="10" y="170" >
      <text align="center" font-size="12" >
        Floating list:
        <br>
        </br>
        <ul float="true" bullet-font-family="Wingdings" spacing="5" type="ü" >
          <li>
            Text Text 0
          </li>
          <li>
            Text Text 1
          </li>
          <li>
            Text Text 2
          </li>
          <li>
            Text Text 3
          </li>
          <li>
            Text Text 4
          </li>
          <li>
            Text Text 5
          </li>
          <li>
            Text Text 6
          </li>
          <li>
            Text Text 7
          </li>
          <li>
            Text Text 8
          </li>
          <li>
            Text Text
          </li>
        </ul>
      </text>
    </field>
    <field height="56" width="380" x="10" y="240" >
      <text align="center" font-size="12" >
        Floating list with image bullets (magnifier):
        <ul float="true" spacing="4" bullet-image="../images/magnifier.png" >
          <li>
            Text Text 0
          </li>
```

```
            <li>
              Text Text 1
            </li>
            <li>
              Text Text 2
            </li>
            <li>
              Text Text 3
            </li>
            <li>
              Text Text 4
            </li>
            <li>
              Text Text 5
            </li>
            <li>
              Text Text 6
            </li>
          </ul>
        </text>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel relative Positionierung

Ein Beispiel, das die relative Positionierung von Feldern zeigt und die Verwendung vom Attribut `max-height`.

Abbildung 17: Generiertes Bild

- Template:

```
Relative Positioning Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image font-size="16" height="300" width="400" >
      <field max-height="200" width="200" x="10" y="20" >
        <text>
          Fixed Position
        </text>
      </field>
      <field max-height="100" width="200" x="10" >
        <label>
          Relative Position
        </label>
```

```
      </field>
      <field max-height="100" width="200" x="10" y="200" >
        <text>
          Second fixed position
        </text>
      </field>
      <field background-color="black" max-height="50" width="100" x="10" >
        <text color="white" >
          Relative 1
        </text>
      </field>
      <field background-color="black" max-height="50" width="100" x="10" >
        <text color="white" >
          Relative 2
        </text>
      </field>
      <field max-height="50" width="100" x="10" >
        <text>
          Relative 3
        </text>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel Einzel- und Mehrfachzeilen

Dieses Beispiel zeigt den Unterschied zwischen dem Label-Tag (nur Einzelzeilen) und dem Text-Tag (Mehrfachzeilen mit automatischem Zeilenumbruch). Es zeigt außerdem, wie der Buchstabenabstand die Textausgabe verändert.

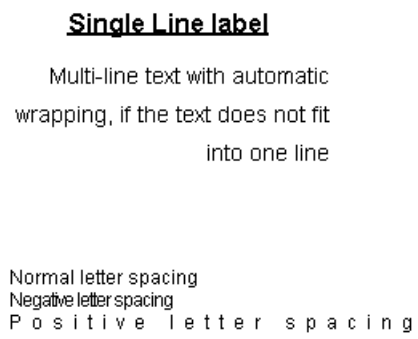


Abbildung 18: Generiertes Bild

- Template:

```
Single and Multi Line Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field x="10" y="20" width="200" height="105" >
       <label font-size="16" font-weight="bold" align="center" text-decoration="underline"
 padding-bottom="15" >
          Single Line label
        </label>
        <text font-size="14" align="right" line-spacing="8" >
          Multi-line text with automatic wrapping, if the text does not fit into one line
        </text>
```

```
      </field>
      <field x="10" y="180" width="300" height="100" >
        <text letter-spacing="0" >
          Normal letter spacing
        </text>
        <text letter-spacing="-0.08" >
          Negative letter spacing
        </text>
        <text letter-spacing="0.5" >
          Positive letter spacing
        </text>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel Abbildung

Dieses Beispiel zeigt verschiedene Arten von Abbildungen mit verschiedenen Rändern, Hintergrundfarben und Dicken. Es zeigt außerdem, wie Felder auf die abgebildeten Elemente, wie z.B. Linien oder Rechtecke, positioniert werden.

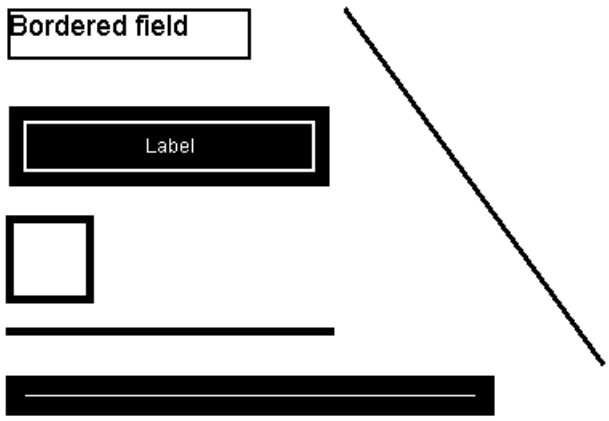


Abbildung 19: Generiertes Bild

- Template:

```
Drawing Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field border="2" height="30" width="150" x="10" y="20" >
        <label font-size="18" >
          Bordered field
        </label>
      </field>
      <rect height="50" width="200" x="10" y="80" >
      </rect>
      <field background-color="black" border="2" border-color="white" height="30"
valign="center" width="180" x="20" y="90" >
        <label align="center" color="white" >
          Label
        </label>
      </field>
      <rect border="5" fill="false" height="50" width="50" x="10" y="150" >
      </rect>
      <line thickness="5" x-from="10" x-to="210" y-from="220" y-to="220" >
      </line>
```

```
      <rect border="5" height="20" width="300" x="10" y="250" >
      </rect>
      <line color="white" x-from="20" x-to="300" y-from="260" y-to="260" >
      </line>
      <line thickness="3" x-from="220" x-to="380" y-from="20" y-to="240" >
      </line>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel Bild

Dieses Beispiel zeigt die Verwendung des Bild-Tags, um Bilder in das Template einzufügen, entweder mithilfe eines relativen Pfads oder durch Einfügen eines Base64-Strings im Template.

Abbildung 20: Generiertes Bild

- Template:

```
Image Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field align="center" height="80" width="380" x="10" y="10" >
        <img autoscale="true" src="../images/imagotag.png" >
        </img>
      </field>
      <field align="center" height="190" width="380" x="10" y="100" >
        <img
data="iVBORw0KGgoAAAANSUhEUgAAAQgAAACwAQAAAADbY83+AAALVElEQVR42u2YfVxTVR/ABByIISpvx
gIrplmWImvj4miJe0seQLCaZYvLBqWm+O58UMWBcFA0RX8iXtHhwhBaCAVYKKcJEUCzC+VYzFSYK
DaWx4Rhju7vnPOecy4Ah0efT3+6P+zm7n+/9vZ9zfufw4N/9eM+J50TXnwJCU8+EDsJN1CLW+wN
bqAiYp4lzGN5g/eTUQOER7ojLvOhOZKMmO4ttYZCV+hM4aGpey0GIZ0J9Ro8ZLu3dFcspVBXu3eE
pCthSVWqb0BjDY1EWBBBP0PUt4aH6yCQSSAE4d3KSG3updyYVxe3Eo0LIDz3DAGSh4bkPVXU0qno
j7o7S1un+e+F2dNXMHrOW8MzRO2HVR5Mjk/N2ic08Vb5DPGVXBczCQotwnwhEaK2J1gI8mobslRq
vk6iyUMvokAXX/TQnJIqb6SNavMQ8y0cr67Ebrp5hVgGi8E0kDbjMXpR1SUeYDhdszDA2qSDqRGH
I5rRm8YuRKsD/cP8zxWatXcLqzz1LI53kX08Wl6mxxTPZwZYUhNVoBagAmDS7Imyl+OSPxjEKlRQ
ri43Q+Quy9oTKdOAF98RxvxOh+Qd8ZLgStTaE4GnrboLiYdbIyATNya9CtUxbLIjQGiGZcvlUCu8
IJFtUI42PepUQBxh5l1s0k52rKpiPSM/91tnwt+r7bzVOx9q2B6dWR2h82nO1Z1U/Ag7CpkjWnoH
BjWnZ4CJmyOa1jf+QmJhsstc/YFX15atW7zVb2uRkf8wJGwxCpkBCdnWTlydtu0FUUHBn48ZYWWQ
9UPTEUSYTRDcpW3E9TPveH6XFmDhGcRDrgFxQSoqZisDGW+5jdi+i/fqIotkkeogv/Ji0ZqAUmwD
DbW/+tmIjB938VJ/Wr1BdItvroAxitJy9LIY3og7YiMuGkcMv/2+v/FQqVIxs8x34hHsqppufnjK
RgR/1ccreXd21JVBLVLXfG/FnSgcEEVCtNFGvP1Q4Drh7LqdZwbTMA4my+fh2WRQx/aeYCPcbgr6
OX2jkPluU+vo6CmT+2fj4m7sO3aBjZizo3/gsOBIreYRmxdn2TA84gQKBID+99fRbUSvS5Gim+ac
aL5Bu7/6k7rL9Y44beKdjeE2Yk+f953c97mupszWKwCIm1Y+wsE8NghGthHAIThsxqCP1x6lmABK
cxPUic142l1xjiqi2og9G+/49q30eqBgztItfuZ0YRPWontNWG8jRIKTovzCneWQOXs4v8EYqmgQ
```

```
4/znVShtxIyl5plzNl+KlFkUTzT+QQ/o1jBceXlzrgg5gp3tMDA4OezfWvio6JyiaFM2vQStuLBZ
OfmJnCOsriVbHCTqeUrm/LWD8tyhGbL3cNhbyiljG9H8mkPfF2//nEPBmJ9hIS15uGnRRLw2hrv8
0UbUHztQ+q8RoyQWyjAgjbkRvgBkRGPl58QP93PEpSHh1CzFhVwIQ+k6mTZpNBs4FSuPKNW2EV/F
Op/asezKMasi4etNzOD6N0D9WFzL4oaaNmImWFCRvktX/Ov0L4roqD8D7hj7jEOE0afBICYEcNn6
iTCjKPouoB9diNFm556rSvJCQQXfqw18TobXGb+88/kQ6GOPu7iDkqcTrUkmvIScK2jiCHbok6P8
+Cl00d7Ipg/gvaL3Ci4PN6P5YJ5pYjnCEqsPleROaY66ajixUAfls+/dEeSgQFhpKeQsbZ6nXUvv
HwAtZWCjm3HP/d+jnjq9gz5mAyJaVIR4Wu76qS5D7R9y07i68beYUOmAh4JVYuzBMUs5Iawjkubn
qRaV16f4359QmRPZpMl3jg5CHrwoXeZCiAeO/XLgnXDleVHM+v5LVMCou+49EG8yu0yaNoK3KIuW
BEkztkt/mT+VrqhWXh8hwnt6CVNTQIir3y16DN3fbclylT641DdEeh3+1lvA7fralYQoaRxJR4wO
8U5/derpJY4/NDVITjo4rCYroCGREAaLlN4ycNYWQbrssGCX3Cituu30xctExiucHZllQsX9cZ5U
PDWhpWS2+7wzivujB/oS4hsuYiXGXAs0L93cC9LLY5asiL1FnXYULefscFFj4mC6B1Ndv3EgHwaE
lS5zKV8uWzPt2wuE+CmTEC4puxfTD76/CmFclDV72MhP6Ks1wYvJjn2b1Ac4LAhLjEu49AqE46Ww
8mbaN1UC55e8CXGPIz7rv0aYf2oGSkGcFDJbeRZ45sCnYtJknCC+gP5L0qNuN7zuA+F0MbTClPUt
Ih+PWYTwJJayHq0XlX8ox6BcxSDnhsoiwMlJbu/ipRRG8jFhfWPu9jkrRvdBAmm0fuqezNMG9u/1
C15SwZQhRMto54itsxZjgsJ78ebExvhxK8pwci3LhyoQYfF4afrJbXNC0feHhcR+36SPcj49joZ1
ekK0upkjL/a+OxbVnQIvLBoNWyMKTB+PhpXfCmksY9l7zKHj9bOR6+OweazJ4Dt41fY30NDbKsV2
WEJrjW47GnhIxnhinlL1tihLOZSGQLWQEODFivcPDBoW6IdWULJ1/W/LAyf3F/wSoebsURMmmvod
HO749ZuxfGhlfTChZpdc186lxvKmm4YRovmD1+sXaZP2oYhznd7v7EifMXUBVPm9uzUGTOjd5u8O
PD38v3xoZJMxUXy61CkpTTiq6M/CBIZoWTc12Ts9LV6MJPiTZs5SFvmft2AFMEnLCaF/c/WcDQmy
8ygn00k7AG5s9hrWGO1Z6SUVczIE37n1d2d7URDIMdHasiclZbmj9NxKJl9Cdg/9cb13ShQ7E6X7
JzNqsSosJdcnHthrPKX32UxxhOvIVW/uh4E44LSZgkJ2bv6+wI/YU4V5N7n9Vp+QcHIvZfanADQt
hvGAhbwRN7d9SY8qpD/j9mzD12/pz0BmtARtwY3Ilh2SuX5Oy64FZcPJCW3EnBkzMqExnjJBSyMy
dKgs0Fsg8JySCze6cIT19I6FqLtaKpSjzY/MEoHjXAfepBP0fIojgPQyKhxQK6EhW0wipq+qHtZ7
7r4Qka27WIrLSb0QfWAlud2fQ43f6bZzzc9Bdr1lYXwceuaQ3GbQtasFosDblB1RdZG1tVjqOvS6
n5Noin1/Kg1l8AwiZwbcSrXyeJQ90YIbCfY4af113Z092OJMVD0syb4mpjuCsfLxM5HUWHR3hMkq
RLKZlYXYUrW8Oxl6bFfzbJ4jhNfUhq4E8lP3MU26ZdiKtKgBXUbby+BDUIe7b/w6EY5SQB1dbk+g
Db8EEVZcpK0KWoHmxSN7ohSSNzhmQHEsH4mqZe2JJkAnooaUGChhryFC7dnVF8BHBDkfxawMxsZf
60Iw1TgE5CwxcxmeeGB5F0L3BD9JoN5dR6rT2oW49z2NXCV9r4Mv4HdD4FjRkKQ7YdWCxG4IcSx+
EuLqiAPxdOdzEtdrw8doAeK06D12fIvVaDoRUvh53i0IaMjlorW32wMKLUSdZVgbYSZWbMtW/DQ+
Ojp0OldqLGoUCksHgfy2KkgO2og6JhNizZ2I6rFybJiNMKmiTEK0RIEOQuWgaP8AEcyXF13kSG0n
og74tB/osKWnXMrlyP/qjsozRk5uP59gIgKa1ShczR1E09YCUlhtBPgB1SGyy9jpqCuQtkceEwWQ
1aEAyToItii8c16AFAId8r4QAsp2kt1SbJc5MWzRKTHBtqfLyy5zQIwLFWezpf3DcrsZBSTcBcF9
XljnuUZ30sJp16p1mi4XHF0JCTT81d0FVwpZso7TZudrg45TdNbKwD/+SgaDjTJmUX99Q2LBRhlP
9HCHYsbEWdeeCJymsx49EThNh3x6IKyLkW9H5T0QYMt+yOzp8bZn0j2oH9wj4aWSLVP1SGy/v3mm
uEfizvrXBlI93zo5Jwf/zb1Uazz9/DbwOfGPiP8D6F7mPMdSRz4AAAAASUVORK5CYII=" >
        </img>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel Barcode

Dieses Beispiel zeigt die Verwendung verschiedener Barcode-Typen.

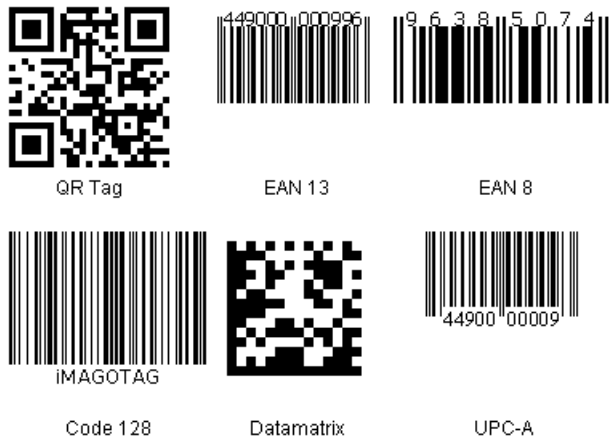


Abbildung 21: Generiertes Bild

- Template:

```
Barcode Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field height="100" width="100" x="10" y="10" >
        <barcode autoscale="true" type="qr" >
          <xsl:value-of select="Barcodes/Qr" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="100" x="10" y="110" >
        <label align="center" >
          QR Tag
        </label>
      </field>
      <field height="60" width="100" x="140" y="10" >
       <barcode fontName="Tahoma" humanReadableEnabled="true" humanReadablePlacement="top"
type="ean13" >
          <xsl:value-of select="Barcodes/Ean13" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="100" x="140" y="110" >
        <label align="center" >
          EAN 13
        </label>
      </field>
      <field height="60" width="140" x="250" y="10" >
       <barcode fontName="Tahoma" humanReadableEnabled="true" humanReadablePlacement="top"
scale="2" type="ean8" >
          <xsl:value-of select="Barcodes/Ean8" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="100" x="270" y="110" >
        <label align="center" >
          EAN 8
        </label>
```

```
      </field>
      <field height="100" width="125" x="10" y="150" >
        <barcode autoscale="true" type="code128" >
          <xsl:value-of select="Barcodes/Code128" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="125" x="10" y="260" >
        <label align="center" >
          Code 128
        </label>
      </field>
      <field height="100" width="100" x="140" y="150" >
        <barcode autoscale="true" type="datamatrix" >
          <xsl:value-of select="Barcodes/DataMatrix" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="100" x="140" y="260" >
        <label align="center" >
          Datamatrix
        </label>
      </field>
      <field height="60" width="100" x="270" y="150" >
        <barcode fontName="Times New Roman" humanReadableEnabled="true"
humanReadablePlacement="bottom" type="upc-a" >
          <xsl:value-of select="Barcodes/Upca" >
          </xsl:value-of>
        </barcode>
      </field>
      <field height="20" valign="bottom" width="100" x="270" y="260" >
        <label align="center" >
          UPC-A
        </label>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Barcode Example (Record)
<Barcodes>
  <Qr>
    iMAGOTAG
  </Qr>
  <Ean13>
    5449000000996
  </Ean13>
  <Ean8>
    96385074
  </Ean8>
  <Code128>
    iMAGOTAG
  </Code128>
  <DataMatrix>
    iMAGOTAG
  </DataMatrix>
  <Upca>
    544900000093
  </Upca>
</Barcodes>
```

Span-Beispiele

Dieses Beispiel zeigt die Verwendung des Span-Tags für die Textformatierung, einschließlich Komprimierung, verschiedener Schriftgrößen und Textdekorationen.

- Template:

```
Span Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field height="71" width="380" x="10" y="10" >
        <text font-family="Times New Roman" font-size="30" >
          Text
          <span font-family="Tahoma" font-size="20" font-weight="bold"
text-decoration="overline" >
            Span
          </span>
          <span superscript="superscript" font-size="24" >
            super
          </span>
          <span font-size="24" >
            normal
          </span>
          <span font-size="24" superscript="subscript" >
            subscript
          </span>
        </text>
        <label font-family="Verdana" font-size="18" >
          Label with
          <span font-size="26" font-style="italic" text-decoration="underline" >
            Span
          </span>
          <span font-family="Arial" font-size="14" text-decoration="linethrough" >
            Span
          </span>
        </label>
      </field>
      <field height="60" width="380" x="10" y="90" >
        <text condense="1,0.8,0.6,0.4" >
          Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text Text
 Text
        </text>
        <label font-size="16,12" >
          Label Label Label Label Label Label Label Label Label
        </label>
        <label condense="1,0.8,0.6,0.4" font-size="16" >
          Label Label Label Label Label Label Label Label Label
        </label>
      </field>
      <field height="140" width="180" x="10" y="150" >
        <text condense="1,0.8,0.6,0.4" font-size="24" >
          <ul>
            <li>
              Text Text Text Text
            </li>
            <li condense="1" >
              Text Text Text Text
            </li>
            <li>
              Text Text Text Text
            </li>
            <li condense="1" font-size="24,22,20,18,16,14,12" >
              Text Text Text Text
            </li>
```

```
          </ul>
        </text>
      </field>
      <field height="140" width="190" x="200" y="150" >
        <text condense="1,0.8,0.6,0.4" font-size="24" >
          <ol>
            <li>
              Text Text Text Text
            </li>
            <li condense="1" >
              Text Text Text Text
            </li>
            <li>
              Text Text Text Text
            </li>
            <li condense="1" font-size="24,22,20,18,16,14,12" >
              Text Text Text Text
            </li>
          </ol>
        </text>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel CSS

Dieses Beispiel verwendet eine CSS-Datei zum Formatieren und Festlegen des Templates.

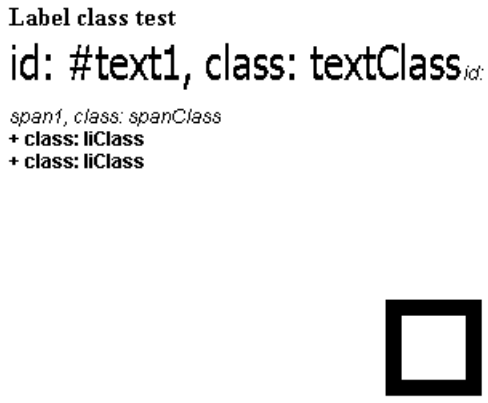


Abbildung 22: Generiertes Bild

- Template:

```
CSS Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <css href="../css/example.css" >
      </css>
      <field max-height="290" width="300" x="10" y="10" >
        <label class="labelClass" font-size="16" >
          Label class test
        </label>
        <text class="textClass" id="text1" >
          id: #text1, class: textClass
          <span class="spanClass" id="span1" >
```

```
            id: span1, class: spanClass
          </span>
          <ul class="listClass" id="ul1" >
            <li id="liClass" >
              class: liClass
            </li>
            <li id="liClass" >
              class: liClass
            </li>
          </ul>
        </text>
      </field>
      <rect class="rectClass" id="rect1" >
      </rect>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Beispiel Tabelle

Dieses Beispiel zeigt die Verwendung einer Tabelle.

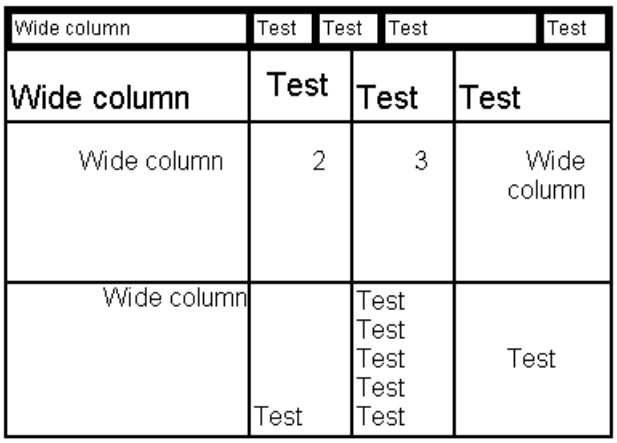


Abbildung 23: Generiertes Bild

- Template:

```
Table Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field max-height="280" width="380" x="10" y="10" >
        <table table-border="2" header-border="5" header-row="true" valign="bottom" >
          <row>
            <cell table-width="40%" >
              <text>
                Wide column
              </text>
            </cell>
            <cell>
              <text>
                Test
              </text>
            </cell>
            <cell>
              <label>
                Test
```

```
            </label>
          </cell>
          <cell>
            <label>
              Test
            </label>
          </cell>
          <cell>
            <label>
              Test
            </label>
          </cell>
        </row>
        <row>
          <cell>
            <text font-size="20" >
              Wide column
            </text>
          </cell>
          <cell cellpadding="10" >
            <label font-size="20" >
              Test
            </label>
          </cell>
          <cell>
            <text font-size="20" >
              Test
            </text>
          </cell>
          <cell>
            <label font-size="20" >
              Test
            </label>
          </cell>
        </row>
        <row cellpadding="15" table-height="100px" valign="top" align="right" >
          <cell>
            <text font-size="16" >
              Wide column
            </text>
          </cell>
          <cell>
            <label font-size="16" >
              2
            </label>
          </cell>
          <cell>
            <text font-size="16" >
              3
            </text>
          </cell>
          <cell table-width="100px" >
            <text font-size="16" >
              Wide column
            </text>
          </cell>
        </row>
        <row>
          <cell align="right" cellpadding="0" valign="top" >
            <text font-size="16" >
              Wide column
            </text>
          </cell>
          <cell>
```

```
            <label font-size="16" >
              Test
            </label>
          </cell>
          <cell>
            <text font-size="16" >
              Test Test Test Test Test
            </text>
          </cell>
          <cell align="center" table-width="100px" valign="center" >
            <label font-size="16" >
              Test
            </label>
          </cell>
        </row>
      </table>
    </field>
  </image>
 </xsl:template>
</xsl:stylesheet>
```

Beispiel Drehen

Dieses Beispiel zeigt die Verwendung des Attributs Drehen.

Abbildung 24: Generiertes Bild

- Template:

```
Rotation Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="300" width="400" >
      <field height="80" rotation="-45" width="80" x="0" y="0" >
        <label font-size="20" font-weight="bold" >
          Aktion!
        </label>
      </field>
      <field height="100" rotation="-90" width="100" x="150" y="0" >
        <barcode type="ean13" >
          5449000000996
        </barcode>
      </field>
      <field height="80" rotation="180" width="200" x="0" y="100" >
        <label font-size="16" font-weight="bold" >
          Test Test Test Test Test
```

```
        </label>
      </field>
      <field height="50" rotation="270" width="100" x="300" y="0" >
        <label font-size="16" font-weight="bold" >
          Test Test
        </label>
      </field>
      <field height="70" rotation="-10" width="200" x="100" y="190" >
        <img autoscale="true" src="../images/imagotag.png" >
        </img>
      </field>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

5.3 7,4"

Erweitertes Abbildungsbeispiel

Dieses Beispiel zeigt die Verwendung von Abbildungselementen wie Dreiecken, Ellipsen und Polygonen.

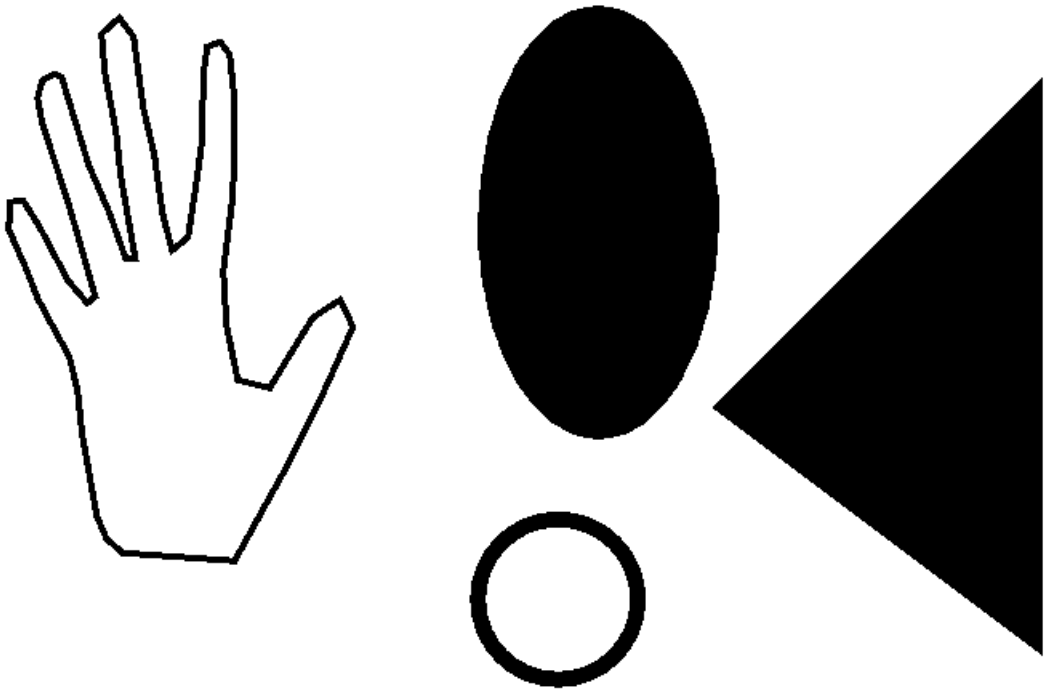


Abbildung 25: Generiertes Bild

- Template:

```
Advanced Drawing Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="800" rotation="270" width="480" >
      <polygon border="4" fill="false" points="104,248 160,279 208,303 250,322 267,314
256,297 235,284 212,270 217,250 251,243 285,241 332,247 367,248 392,248 420,245 428,239
```

```
425,231 411,229 385,228 368,228 339,224 306,219 298,209 321,203 357,198 387,191 431,185
443,176 433,165 409,167 372,174 344,177 293,186 293,180 321,170 351,157 406,140 408,136
404,128 393,125 377,128 328,144 311,149 269,161 265,156 279,144 313,126 329,116 328,108
311,107 286,118 268,125 231,145 210,150 182,153 160,157 132,162 118,168 109,178" >
      </polygon>
      <triangle border="5" x1="50" x2="200" x3="400" y1="750" y2="550" y3="750" >
      </triangle>
      <ellipse height="150" width="270" x="180" y="400" >
      </ellipse>
      <ellipse border="10" fill="false" height="100" width="100" x="30" y="400" >
      </ellipse>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Abgerundetes Rechteck/Feldbeispiel

Dieses Beispiel zeigt die Verwendung von abgerundeten Ecken in Rechtecken und Feldern mit Text.

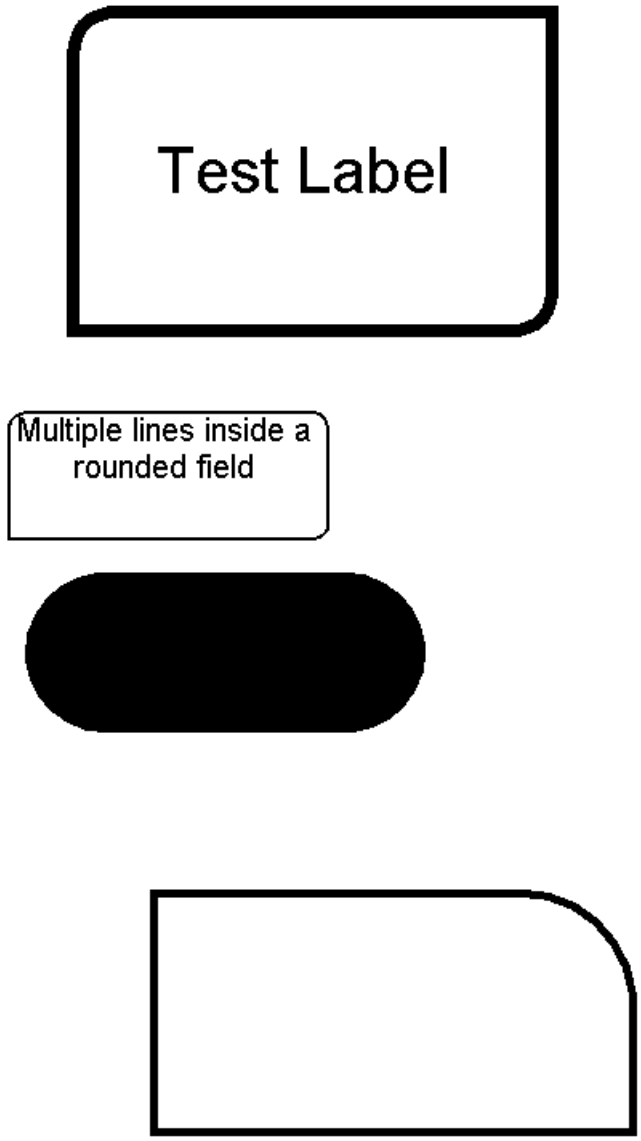


Abbildung 26: Generiertes Bild

- Template:

```
Rounded Rectangle/Field Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <image height="800" width="480" >
      <field border="8" corner-radius="25" height="200"
rounded-corners="TOP_LEFT,BOTTOM_RIGHT" valign="center" width="300" x="50" y="50" >
        <label align="center" font-size="40" >
          Test Label
        </label>
```

```
      </field>
      <field border="2" corner-radius="10" height="80"
rounded-corners="TOP_RIGHT,TOP_LEFT,BOTTOM_RIGHT" width="200" x="10" y="300" >
        <text align="center" font-size="20" >
          Multiple lines inside a rounded field
        </text>
      </field>
      <rect corner-radius="50" height="100"
rounded-corners="TOP_RIGHT,TOP_LEFT,BOTTOM_RIGHT,BOTTOM_LEFT" width="250" x="20" y="400"
>
      </rect>
      <rect border="5" corner-radius="70" fill="false" height="150"
rounded-corners="TOP_RIGHT" width="300" x="100" y="600" >
      </rect>
    </image>
  </xsl:template>
</xsl:stylesheet>
```

Dynamisches Template auf Basis einer Default-Grafik - 7,4"

Dieses Beispiel basiert auf einem dynamischen Template und dem Coca Cola-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 27: Generiertes Bild

- Template:

```
Dynamic template using default record - 7.4" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
```

```
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
','')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
```

```
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
```

```
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 7.4" (Record)
<article articleNumber="5449000000996" >
  <field key="pricePerUnit" value="1.65" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.55" >
  </field>
  <field key="description" value="Dose 0,33 l" >
  </field>
  <field key="name" value="Coca Cola" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 7,4"

Dieses Beispiel basiert auf einem dynamischen Template und dem Chili Con Carne-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 28: Generiertes Bild

- Template:

```
Dynamic template using default record - 7.4" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
```

```
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
```

```
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
```

```
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
             <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 7.4" (Record)
<article articleNumber="9000275639319" >
  <field key="pricePerUnit" value="4.98" >
  </field>
  <field key="unit" value="1 kg" >
  </field>
  <field key="price" value="2.49" >
  </field>
  <field key="description" value="Dose 500 g" >
  </field>
  <field key="name" value="Chili con Carne" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 7,4"

Dieses Beispiel basiert auf einem dynamischen Template und dem Red Bull-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 29: Generiertes Bild

- Template:

```
Dynamic template using default record - 7.4" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
```

```
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
```

```
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
```

```
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 7.4" (Record)
<article articleNumber="9002490100070" >
  <field key="pricePerUnit" value="5.56" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="1.39" >
  </field>
  <field key="description" value="Dose 250 ml" >
  </field>
  <field key="name" value="Red Bull" >
  </field>
  <field key="sale" value="1" >
  </field>
</article>
```

Dynamisches Template auf Basis einer Default-Grafik - 7,4"

Dieses Beispiel basiert auf einem dynamischen Template und dem Vöslauer-Beispiel. Die Bildgröße hängt vom Display-Typ ab.

Abbildung 30: Generiertes Bild

- Template:

```
Dynamic template using default record - 7.4" (Template)
<xsl:stylesheet version="1.0" >
  <xsl:template match="Record" >
    <xsl:choose>
      <xsl:when test="Label/DisplayHeight = 176 and Label/DisplayWidth = 264" >
        <image height="176" width="264" >
          <line thickness="2" x-from="0" x-to="264" y-from="126" y-to="126" >
          </line>
```

```
          <field height="36" width="245" x="9" y="10" >
            <text font-family="Tahoma" font-size="14" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="83" width="190" x="60" y="47" >
            <text align="right" font-family="Impact" font-size="68" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="180" y="131" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="180" y="146" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="136" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="158" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 300 and Label/DisplayWidth = 400" >
        <image height="300" width="400" >
          <line thickness="2" x-from="0" x-to="400" y-from="236" y-to="236" >
          </line>
          <field height="90" width="382" x="9" y="10" >
            <text font-family="Tahoma" font-size="24" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="18" >
              <br>
              </br>
```

```
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="122" width="350" x="40" y="100" >
            <text align="right" font-family="Impact" font-size="100" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
          </field>
          <field height="14" width="70" x="320" y="248" >
            <text align="right" font-family="Verdana" font-size="11" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="24" width="70" x="320" y="266" >
            <text align="right" font-family="Tahoma" font-size="20" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="260" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="282" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
      <xsl:when test="Label/DisplayHeight = 800 and Label/DisplayWidth = 480" >
        <image height="800" width="480" >
          <line thickness="2" x-from="0" x-to="480" y-from="700" y-to="700" >
          </line>
          <field height="400" width="460" x="10" y="10" >
            <text font-family="Tahoma" font-size="48" >
              <span font-weight="bold" >
                <utils method="toUpperCase" >
                  <xsl:value-of select="article/field[@key='name']/@value" >
                  </xsl:value-of>
                </utils>
              </span>
            </text>
            <text font-family="Tahoma" font-size="28" >
              <br>
              </br>
              <xsl:value-of select="article/field[@key='description']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="200" width="460" x="10" y="480" >
            <text align="right" font-family="Impact" font-size="120" >
              <xsl:value-of select="translate(article/field[@key='price']/@value, '.',
',')" >
              </xsl:value-of>
            </text>
```

```
          </field>
          <field height="32" width="100" x="370" y="728" >
            <text align="right" font-family="Verdana" font-size="26" >
              <xsl:value-of select="article/field[@key='unit']/@value" >
              </xsl:value-of>
            </text>
          </field>
          <field height="32" width="100" x="370" y="760" >
            <text align="right" font-family="Tahoma" font-size="26" >
              <xsl:value-of select="translate(article/field[@key='pricePerUnit']/@value,
'.', ',')" >
              </xsl:value-of>
            </text>
          </field>
          <field align="center" height="20" width="140" x="10" y="758" >
            <barcode autoscale="false" humanReadableEnabled="false" type="ean13" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </barcode>
          </field>
          <field height="12" width="140" x="10" y="780" >
            <text align="center" font-family="Tahoma" font-size="10" >
              <xsl:value-of select="article/@articleNumber" >
              </xsl:value-of>
            </text>
          </field>
        </image>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Dynamic template using default record - 7.4" (Record)
<article articleNumber="9009700145104" >
  <field key="pricePerUnit" value="0.35" >
  </field>
  <field key="unit" value="1 l" >
  </field>
  <field key="price" value="0.35" >
  </field>
  <field key="description" value="Vöslauer prickelnd, 1 l PET" >
  </field>
  <field key="name" value="Vöslauer Mineralwasser" >
  </field>
</article>
```

Angepasste Grafik 7,4"

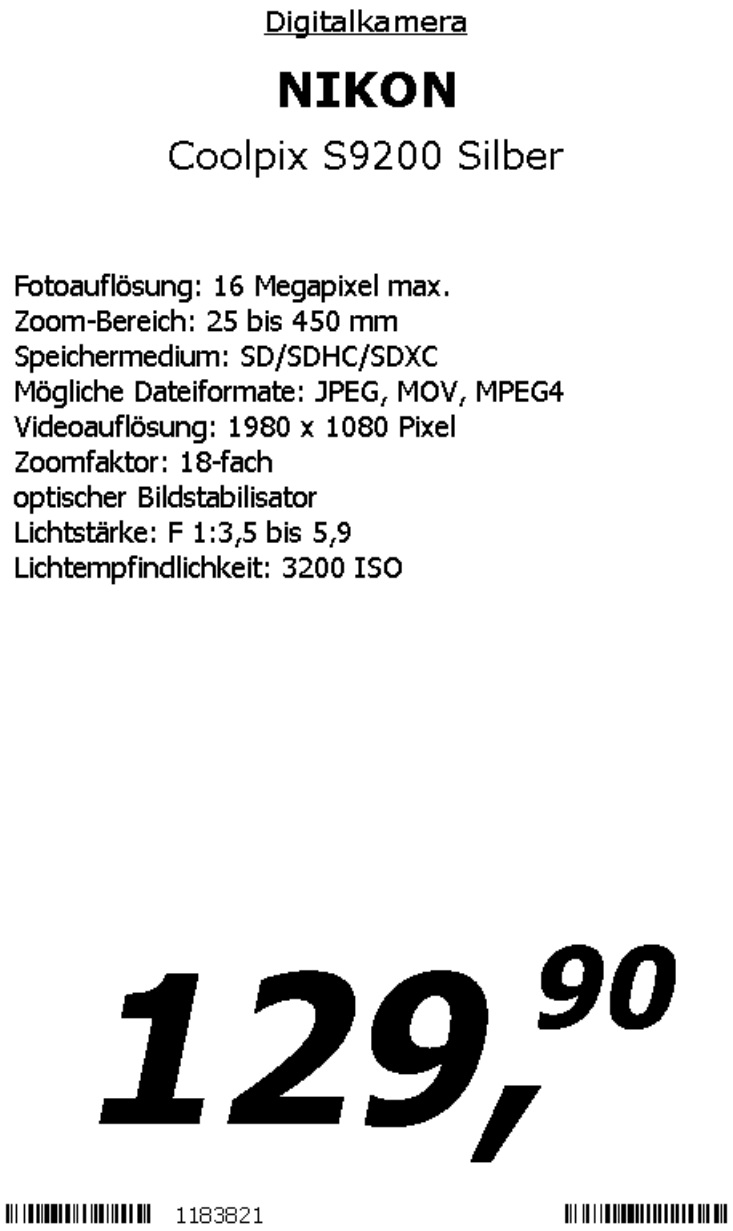


Abbildung 31: Generiertes Bild

- Template:

```
Custom Record 7.4" Example (Template)
<xsl:stylesheet version="2.0" >
  <xsl:template match="Record" >
    <xsl:variable name="price" >
      <xsl:value-of select="translate(Article/Price,'.','')" >
      </xsl:value-of>
    </xsl:variable>
    <xsl:variable name="maxProperties" select="8" >
    </xsl:variable>
    <xsl:variable name="maxSpecifications" select="7" >
    </xsl:variable>
```

```
    <image height="800" width="480" >
      <field height="22" width="480" x="0" y="30" >
       <text align="center" font-family="Verdana" font-size="18" text-decoration="underline"
 >
         <xsl:value-of select="Article/Category" >
         </xsl:value-of>
       </text>
      </field>
      <field height="36" width="480" x="0" y="65" >
        <text align="center" font-family="Verdana" font-size="30" font-weight="bold" >
          <utils method="toUpperCase" >
            <xsl:value-of select="Article/Manufacturer" >
            </xsl:value-of>
          </utils>
        </text>
      </field>
      <field height="60" width="450" x="15" y="110" >
        <text align="center" font-family="Verdana" font-size="24" >
          <xsl:value-of select="Article/Name" >
          </xsl:value-of>
        </text>
      </field>
      <field height="400" width="450" x="15" y="175" >
        <text align="center" font-family="Tahoma" font-size="15" font-weight="bold" >
          <ul spacing="0" type="" >
            <xsl:for-each select="Article/Properties/Field" >
              <li>
                <xsl:value-of select="." >
                </xsl:value-of>
              </li>
            </xsl:for-each>
          </ul>
        </text>
        <text align="left" font-family="Tahoma" font-size="18" font-weight="normal"
padding-left="5" padding-top="20" >
          <ul spacing="0" type="" >
            <xsl:for-each select="Article/Specification/Field" >
              <li>
                <xsl:value-of select="." >
                </xsl:value-of>
              </li>
            </xsl:for-each>
          </ul>
        </text>
      </field>
      <xsl:choose>
        <xsl:when test="number(substring-before($price, ',')) < 100" >
          <xsl:choose>
            <xsl:when test="substring-after($price,',') = '00'" >
              <field height="205" width="410" x="10" y="570" >
              <text align="right" font-family="Tahoma" font-size="170" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
            </xsl:when>
            <xsl:otherwise>
              <field height="205" width="330" x="10" y="570" >
              <text align="right" font-family="Tahoma" font-size="170" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
```

```
              </field>
              <field height="157" width="120" x="320" y="570" >
                <text font-family="Tahoma" font-size="85" font-style="italic"
font-weight="bold" >
                  <xsl:value-of select="substring-after($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
            </xsl:otherwise>
          </xsl:choose>
        </xsl:when>
        <xsl:when test="number(substring-before($price, ',')) < 1000" >
          <xsl:choose>
            <xsl:when test="substring-after($price,',') = '00'" >
              <field height="157" width="415" x="10" y="600" >
              <text align="right" font-family="Tahoma" font-size="130" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
            </xsl:when>
            <xsl:otherwise>
              <field height="157" width="340" x="10" y="600" >
              <text align="right" font-family="Tahoma" font-size="130" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
              <field height="157" width="100" x="340" y="600" >
                <text font-family="Tahoma" font-size="70" font-style="italic"
font-weight="bold" >
                  <xsl:value-of select="substring-after($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
            </xsl:otherwise>
          </xsl:choose>
        </xsl:when>
        <xsl:when test="number(substring-before($price, ',')) < 10000" >
          <xsl:choose>
            <xsl:when test="substring-after($price,',') = '00'" >
              <field height="157" width="435" x="10" y="600" >
              <text align="right" font-family="Tahoma" font-size="110" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
            </xsl:when>
            <xsl:otherwise>
              <field height="157" width="370" x="00" y="600" >
              <text align="right" font-family="Tahoma" font-size="110" font-style="italic"
 font-weight="bold" >
                  <xsl:value-of select="substring-before($price,',')" >
                  </xsl:value-of>
                </text>
              </field>
              <field height="157" width="100" x="360" y="600" >
                <text font-family="Tahoma" font-size="55" font-style="italic"
font-weight="bold" >
                  <xsl:value-of select="substring-after($price,',')" >
                  </xsl:value-of>
```

```
              </text>
            </field>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:when>
    </xsl:choose>
    <field align="left" height="12" width="100" x="15" y="780" >
      <barcode humanReadableEnabled="false" type="code128" >
        <xsl:value-of select="Article/ArticleNumber" >
        </xsl:value-of>
      </barcode>
    </field>
    <field height="17" width="80" x="120" y="778" >
      <text font-family="Tahoma" font-size="14" >
        <xsl:value-of select="Article/ArticleNumber" >
        </xsl:value-of>
      </text>
    </field>
    <field align="right" height="12" width="150" x="315" y="780" >
      <barcode humanReadableEnabled="false" type="code128" >
        <xsl:value-of select="Label/Id" >
        </xsl:value-of>
      </barcode>
    </field>
  </image>
 </xsl:template>
</xsl:stylesheet>
```

- Beispiel:

```
Custom Record 7.4" Example (Record)
<Article>
  <ArticleNumber>
    1183821
  </ArticleNumber>
  <Name>
    Coolpix S9200 Silber
  </Name>
  <Manufacturer>
    Nikon
  </Manufacturer>
  <Category>
    Digitalkamera
  </Category>
  <Price>
    129,90
  </Price>
  <Specification>
    <Field>
      Fotoauflösung: 16 Megapixel max.
    </Field>
    <Field>
      Zoom-Bereich: 25 bis 450 mm
    </Field>
    <Field>
      Speichermedium: SD/SDHC/SDXC
    </Field>
    <Field>
      Mögliche Dateiformate: JPEG, MOV, MPEG4
    </Field>
    <Field>
      Videoauflösung: 1980 x 1080 Pixel
    </Field>
    <Field>
      Zoomfaktor: 18-fach
```

```
    </Field>
    <Field>
      optischer Bildstabilisator
    </Field>
    <Field>
      Lichtstärke: F 1:3,5 bis 5,9
    </Field>
    <Field>
      Lichtempfindlichkeit: 3200 ISO
    </Field>
  </Specification>
  <Packaging>
    <Item>
      Trageschlaufe
    </Item>
    <Item>
      USB-Kabel
    </Item>
    <Item>
      Netzadapter
    </Item>
  </Packaging>
  <Accessories>
    <Item>
      Akku EN-EL12
    </Item>
    <Item>
      SDHC-Karte
    </Item>
  </Accessories>
</Article>
```