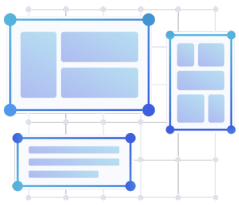
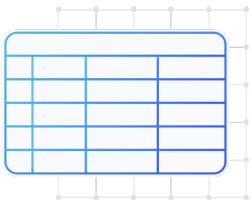
UniPDF

TEMPLATES DOCUMENTATION






POWERED BY

UniDOC

Table of Contents

Table of contents ⇑

INTRODUCTION

Templates provide a new way of building PDF files using the components available in the **creator** package. The templates are defined using an XML based markup language and they are parsed and translated into creator components at runtime by the internal template processor.

The templates have two processing phases. First, they are executed as text/template#Template instances, which allows actions to be executed and data to be injected. The second processing phase takes the output of the first and parses it into components which are then rendered.

Components are split into **data** and **container** components. Data components produce some form of content like text, images, geometric shapes. Container components hold or group other components (e.g. tables, lists).

The aim of templates is to make creating complex layouts a lot easier, faster, and by writing less code. In fact, this document was written entirely using templates. This documentation and other examples of templates in action, including their source code, can be found in our examples repository.

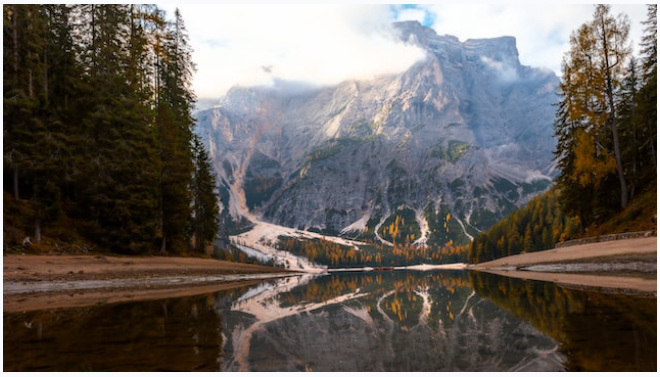
Let's dive right into it. Here's a simple example of a **division** containing a **paragraph**, a **line** and an **image**.

TEMPLATE

```
<division padding="15 10" margin="3">
    <background border-color="#333333" border-size="0.5"
        fill-color="#f9fafe"></background>

    <paragraph>
        <text-chunk>Far far away, behind the word mountains, </text-chunk>
        <text-chunk>far from the countries Vokalia and </text-chunk>
        <text-chunk>Consonantia, there live the blind texts. </text-chunk>
        <text-chunk>Separated they live in Bookmarksgrove </text-chunk>
        <text-chunk>right at the coast of the Semantics, </text-chunk>
        <text-chunk>a large language ocean.</text-chunk>
    </paragraph>

    <line position="relative" fit-mode="fill-width"
        thickness="0.5" color="#333333" margin="5 0"></line>

    <image fit-mode="fill-width"
        src="path('templates/res/images/sample-image.jpg')"></image>
</division>
```

RESULT

Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean.



DATA COMPONENTS

Data components are used to store and render some form of content. Most creator **data** components can be used in templates, with support for more to come in the near future.

Here is the list of currently supported components:

- paragraph
  - text-chunk
- image
- rectangle
- ellipse
- line
- chart
- page-break

PARAGRAPHS

The paragraph is the primary component used to render blocks of text. A paragraph consists of one or more text chunks, which are the main building blocks of the component. A text chunk is a piece of text with a particular style, similar to a **HTML span**. The component is represented in templates using the <paragraph> tag. Paragraphs start on a new line and any components rendered after a paragraph start on a new line as well.

Basic syntax of paragraphs:

TEMPLATE

```
<paragraph>
    <text-chunk>A single text chunk paragraph.</text-chunk>
</paragraph>
```

RESULT

A single text chunk paragraph.

TEMPLATE

```
<paragraph>
    <text-chunk>A paragraph with </text-chunk>
    <text-chunk color="#0000ff">three </text-chunk>
    <text-chunk>text chunks.</text-chunk>
</paragraph>
```

RESULT

A paragraph with three text chunks.

Supported attributes:

» text-align

Default: `left`. Valid values: `left`, `right`, `center`, `justify`.

The `text-align` attribute aligns the paragraph text chunks horizontally based on the specified option.

TEMPLATE

```
<paragraph>
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph text-align="right">
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph text-align="center">
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» vertical-text-align

Default: `baseline`. Valid values: `baseline`, `center`.

The `vertical-text-align` attribute specifies the reference point used for vertically aligning the paragraph text.

TEMPLATE

```
<paragraph>
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph vertical-text-align="center">
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» line-height

Default: 1.0.

The `line-height` represents a scale factor for the vertical space a text line takes. Basically, the line height attribute controls the amount of vertical space between paragraph text lines.

TEMPLATE

```
<paragraph>
    <text-chunk>Sample
text</text-chunk>
</paragraph>
```

RESULT

Sample
text

TEMPLATE

```
<paragraph line-height="1.5">
    <text-chunk>Sample
text</text-chunk>
</paragraph>
```

RESULT

Sample
text

» x

Default: 0.

The `x` attribute sets the X coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» y

Default: 0.

The `y` attribute sets the Y coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<paragraph>
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph margin="10 0 0 50">
    <text-chunk>Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» enable-wrap

Default: true.

The `enable-wrap` attribute controls whether the content of the paragraph should be split into lines based on the available space.

TEMPLATE

```
<paragraph margin="0 20">
    <text-chunk>Sample text ..........................</text-chunk>
    <text-chunk>.....................................</text-chunk>
</paragraph>
```

RESULT

Sample text .....
.....

TEMPLATE

```
<paragraph margin="0 20" enable-wrap="false">
    <text-chunk>Sample text ..........................</text-chunk>
    <text-chunk>.....................................</text-chunk>
</paragraph>
```

RESULT

Sample text .....

» enable-word-wrap

Default: `false`.

The `enable-word-wrap` attribute controls whether the content of the paragraph should be split into lines based on the available space.

TEMPLATE

```
<paragraph margin="0 40">
    <text-chunk>Far far away, behind the word mountains, </text-chunk>
    <text-chunk>far from the countries Vokalia and </text-chunk>
    <text-chunk>Consonantia, there live the blind texts.</text-chunk>
</paragraph>
```

RESULT

Far far away, behind the word mountains, f
ar from the countries Vokalia and Consona
ntia, there live the blind texts.

TEMPLATE

```
<paragraph margin="0 40" enable-word-wrap="true">
    <text-chunk>Far far away, behind the word mountains, </text-chunk>
    <text-chunk>far from the countries Vokalia and </text-chunk>
    <text-chunk>Consonantia, there live the blind texts.</text-chunk>
</paragraph>
```

RESULT

Far far away, behind the word mountains,
far from the countries Vokalia and
Consonantia, there live the blind texts.

» text-overflow

Default: `visible`. Valid values: `visible`, `hidden`.

The `text-overflow` attribute controls the bevahior of content which does not fit in the available space and overflows. The attribute has no effect if the `enable-wrap` attribute is set to `true` because in that case the content that does not fit is moved on a new line.

TEMPLATE

```
<paragraph margin="0 20" enable-wrap="false">
    <text-chunk>Sample text ..........................</text-chunk>
    <text-chunk>.....................................</text-chunk>
</paragraph>
```

RESULT

Sample text .....

TEMPLATE

```
<paragraph margin="0 20" enable-wrap="false" text-overflow="hidden">
    <text-chunk>Sample text ..........................</text-chunk>
    <text-chunk>.....................................</text-chunk>
</paragraph>
```

RESULT

Sample text .....

» angle

Default: `0`.

The `angle` attribute represents an angle specified in degrees to rotate the paragraph content by. The rotation is applied anti-clockwise.

TEMPLATE

```
<paragraph margin="0 5" angle="90">
    <text-chunk>T</text-chunk>
</paragraph>
```

RESULT

⊢

TEMPLATE

```
<paragraph margin="0 5" angle="180">
    <text-chunk>T</text-chunk>
</paragraph>
```

RESULT

⊥

PARAGRAPH » TEXT CHUNKS

The content of paragraphs is constructed using text chunks. Text chunks are bits of text with a particular style, configured using attributes. The text chunk component is represented in templates using the <text-chunk> tag.

Supported attributes:

» font

Default: helvetica.

The `font` attribute specifies the font used for rendering the text chunk.
For more information see chapter 4.1. Fonts.

TEMPLATE

```
<paragraph>
    <text-chunk font="courier">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» font-size

Default: 10.

The `font-size` attribute specifies the size of the font used by the text chunk.

TEMPLATE

```
<paragraph>
    <text-chunk font-size="12">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» character-spacing

Default: 0.

The `character-spacing` attribute is used to change the distance between individual characters.

TEMPLATE

```
<paragraph>
    <text-chunk character-spacing="6">Sample text</text-chunk>
</paragraph>
```

RESULT

S a m p l e t e x t

» horizontal-scaling

Default: 100.

The `horizontal-scaling` attribute is used to scale the text horizontally by the specified percent.

TEMPLATE

```
<paragraph>
    <text-chunk horizontal-scaling="200">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph>
    <text-chunk horizontal-scaling="75">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» color

Default: #000000.

The `color` attribute is used to specify the text color.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<paragraph>
    <text-chunk color="#00695c">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» link

Default: "".

The `link` attribute is used to turn the text chunk into an internal or external link. The destination of internal links is a page or part of a page within the current document, whereas external links send users to external URLs.

TEMPLATE

```
<paragraph>
    <text-chunk>Go to the start of </text-chunk>
    <text-chunk link="page(8)">page 8</text-chunk>
    <text-chunk>.</text-chunk>
</paragraph>
```

RESULT

Go to the start of page 8.

TEMPLATE

```
<paragraph>
    <text-chunk>Go to </text-chunk>
    <text-chunk link="page(8, 0, 50)">page 8</text-chunk>
    <text-chunk>, at coordinates (0, 50).</text-chunk>
</paragraph>
```

RESULT

Go to page 8, at coordinates (0, 50).

TEMPLATE

```
<paragraph>
    <text-chunk>Visit </text-chunk>
    <text-chunk link="url('https://unidoc.io')">unidoc.io</text-chunk>
    <text-chunk>.</text-chunk>
</paragraph>
```

RESULT

Visit unidoc.io.

» underline

Default: false. Valid values: true, false.

The `underline` attribute controls whether the text chunk text is underlined.

TEMPLATE

```
<paragraph>
    <text-chunk underline="true">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» underline-color

Default: color of the text.

The `underline-color` attribute configures the color of the decoration line.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<paragraph>
    <text-chunk underline="true"
        underline-color="#ff00ff">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» underline-offset

Default: 0.

The `underline-offset` attribute configures the offset of the decoration line.

TEMPLATE

```
<paragraph>
    <text-chunk underline="true"
        underline-offset="3">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» underline-thickness

Default: 1.

The `underline-thickness` attribute configures the thickness of the decoration line.

TEMPLATE

```
<paragraph>
    <text-chunk underline="true" underline-offset="2"
        underline-thickness="2">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» text-rise

Default: 0.

The `text-rise` attribute represents a vertical offset applied to the text chunk. Text rise values can be either positive or negative.

TEMPLATE

```
<paragraph>
    <text-chunk >Sample </text-chunk>
    <text-chunk text-rise="5">text</text-chunk>
</paragraph>
```

RESULT

Sample text

» rendering-mode

Default: fill. Valid values: fill, stroke, fill-stroke, invisible.

The `rendering-mode` attribute determines whether showing text shall cause character outlines to be stroked, filled or a combination of the two. The attribute can also be used to render invisible text.

TEMPLATE

```
<paragraph>
    <text-chunk rendering-mode="fill">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph>
    <text-chunk rendering-mode="stroke"
        outline-size="0.1">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

TEMPLATE

```
<paragraph>
    <text-chunk rendering-mode="fill-stroke"
        outline-size="0.1" color="#ffff00">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» outline-size

Default: 0.

The `outline-size` attribute is used to specify the outline size of the text. By default, outlines are not rendered. That behavior can be changed by specifying an appropriate `rendering-mode` value such as `fill-stroke`.

TEMPLATE

```
<paragraph>
    <text-chunk rendering-mode="fill-stroke"
        outline-size="0.01" color="#ffffff">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

» outline-color

Default: #ffffff.

The `outline-color` attribute is used to specify the color of the text outline. By default, outlines are not rendered. That behavior can be changed by specifying an appropriate `rendering-mode` value such as `fill-stroke`. For more information see chapter 4.3. Colors.

TEMPLATE

```
<paragraph>
    <text-chunk rendering-mode="fill-stroke" outline-size="0.01"
        outline-color="#ff0000" color="#ffff00">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

IMAGES

Images can be rendered using the **image** component. The component is represented in templates using the <paragraph> tag.

Basic syntax of images:

TEMPLATE

```
<image src="path('templates/res/images/sample-image.jpg')"
    fit-mode="fill-width"></image>
```

RESULT

Supported attributes:

» src

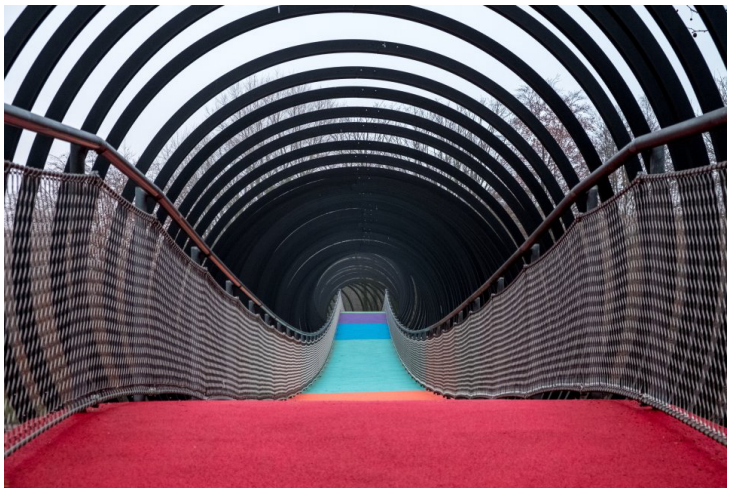
Default: empty string.

The `src` attribute is used to specify the source of the image. The source image can be specified by path in the template, in which case it will be loaded, or it can be specified by name if the image is loaded in the image map of the options used to draw the template.
For more information see chapter 4.2. Images.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-4.jpg')"
    fit-mode="fill-width"></image>
```

RESULT



» fit-mode

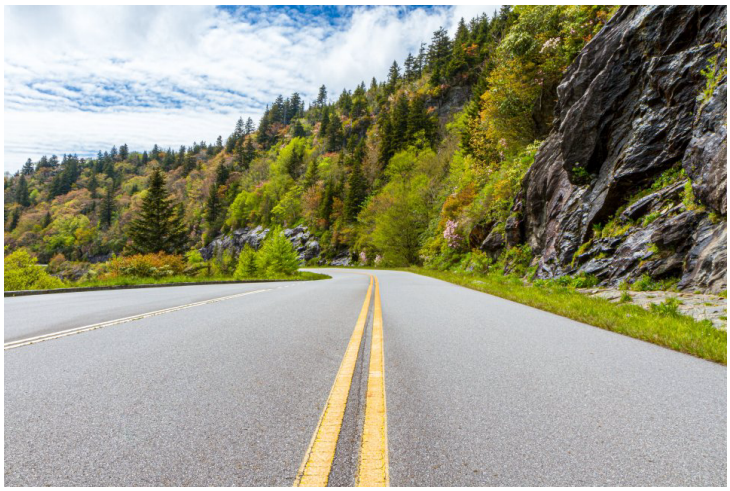
Default: `none`. Valid values: `none`, `fill-width`.

The `fit-mode` attribute controls the sizing of the component relative to the available space. When the attribute value is set to `fill-width`, the component is scaled so that it occupies the entire available width, preserving the original aspect ratio.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    fit-mode="fill-width"></image>
```

RESULT



» align

Default: `left`. Valid values: `left`, `right`, `center`.

The `align` attribute aligns the image in the available space, based on the specified option.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-2.jpg')"
    width="160" height="90"></image>
```

RESULT

TEMPLATE

```
<image src="path('templates/res/images/sample-image-2.jpg')"
    width="160" height="90" align="center"></image>
```

RESULT

TEMPLATE

```
<image src="path('templates/res/images/sample-image-2.jpg')"
    width="160" height="90" align="right"></image>
```

RESULT

» width

Default: original image width.

The `width` attribute sets the width of the rendered image.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-3.jpg')"
    width="160" height="90"></image>
```

RESULT

» height

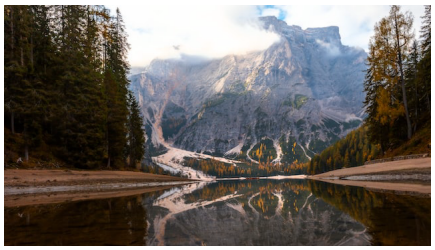
Default: original image height.

The `height` attribute sets the height of the rendered image.

TEMPLATE

```
<image src="path('templates/res/images/sample-image.jpg')"
    width="160" height="90"></image>
```

RESULT



» opacity

Default: 1.0. Valid values: any value between 0 and 1.

The `opacity` attribute allows setting the transparency of the image.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-2.jpg')"
    width="160" height="90"></image>
```

RESULT

TEMPLATE

```
<image src="path('templates/res/images/sample-image-2.jpg')"
    width="160" height="90" opacity="0.5"></image>
```

RESULT

Table of contents ⇑

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-3.jpg')"
    width="160" height="90"></image>
```

RESULT

TEMPLATE

```
<image src="path('templates/res/images/sample-image-3.jpg')"
    width="160" height="90" margin="10 0 0 50"></image>
```

RESULT

» x

Default: 0.

The `x` attribute sets the X coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» y

Default: 0.

The `y` attribute sets the Y coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» angle

Default: 0.

The `angle` attribute represents an angle specified in degrees to rotate the image by. The rotation is applied anti-clockwise.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    width="160" height="90" angle="180"></image>
```

RESULT

RECTANGLES

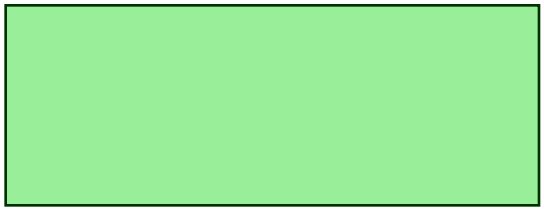
Rectangles can be rendered using the **rectangle** component. The component is represented in templates using the <rectangle> tag.

Basic syntax of rectangles:

TEMPLATE

```
<rectangle position="relative" width="200" height="75"
    fill-color="#98ee99" border-color="#003300"></rectangle>
```

RESULT



Supported attributes:

» width

Default: 0.

The `width` attribute sets the width of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="75" height="50"></rectangle>
```

RESULT

» height

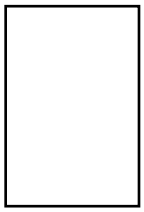
Default: 0.

The `height` attribute sets the height of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="50" height="75"></rectangle>
```

RESULT



» fit-mode

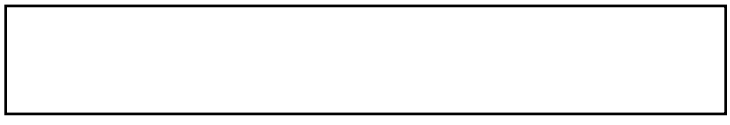
Default: none. Valid values: none, fill-width.

The `fit-mode` attribute controls the sizing of the component relative to the available space. When the attribute value is set to `fill-width`, the component is scaled so that it occupies the entire available width, preserving the original aspect ratio. The `width` and `height` of the rectangle must be specified in order to calculate the original aspect ratio of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="1" height="0.15"
    fit-mode="fill-width"></rectangle>
```

RESULT



» margin

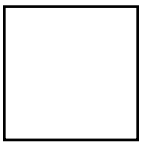
Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<rectangle position="relative" width="50" height="50"
    margin="10 0 10 100"></rectangle>
```

RESULT



» position

Default: absolute. Valid values: relative, absolute.

The `position` attribute controls whether the component uses relative or absolute position. In absolute mode, user position the component manually using the `x` and `y` attributes. In relative position mode, the component is positioned relative to the already rendered components.

» x

Default: 0.

The `x` attribute sets the X coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» y

Default: 0.

The `y` attribute sets the Y coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» fill-color

Default: #ffffff.

The `fill-color` attribute sets the fill color of the rectangle.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<rectangle position="relative" width="100" height="50"
    fill-color="#e64a19"></rectangle>
```

RESULT

» fill-opacity

Default: 1.0. Valid values: any value between 0 and 1.

The `fill-opacity` attribute allows setting the transparency of the rectangle background.

TEMPLATE

```
<rectangle position="relative" width="100" height="50"
    fill-color="#e64a19" fill-opacity="0.5"></rectangle>
```

RESULT

» border-width

Default: 1.

The `border-width` attribute sets the width of the rectangle border.

TEMPLATE

```
<rectangle position="relative" width="200" height="75"
    border-width="5"></rectangle>
```

RESULT

» border-color

Default: #000000.

The `border-color` attribute sets the border color of the rectangle.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<rectangle position="relative" width="100" height="75" border-width="10"
    border-color="#5c007a"></rectangle>
```

RESULT

» border-opacity

Default: 1.0. Valid values: any value between 0 and 1.

The `border-opacity` attribute allows setting the transparency of the rectangle border.

TEMPLATE

```
<rectangle position="relative" width="100" height="75" border-width="10"
    border-color="#5c007a" border-opacity="0.5"></rectangle>
```

RESULT

» border-radius

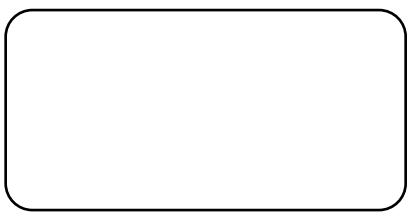
Default: 0.

The `border-radius` attribute sets the radius used when rendering the corners of the rectangle.
For more information see chapter 4.5. Border Radius.

TEMPLATE

```
<rectangle position="relative" width="150" height="75"
    border-radius="10"></rectangle>
```

RESULT



» border-top-left-radius

Default: 0.

The `border-top-left-radius` attribute sets the radius of the top left corner of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="150" height="50" border-width="3"
    border-color="#283593" border-top-left-radius="10"></rectangle>
```

RESULT

» border-top-right-radius

Default: 0.

The `border-top-right-radius` attribute sets the radius of the top right corner of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="150" height="50" border-width="3"
    border-color="#009688" border-top-right-radius="10"></rectangle>
```

RESULT

» border-bottom-left-radius

Default: 0.

The `border-bottom-left-radius` attribute sets the radius of bottom left corner of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="150" height="50" border-width="3"
    border-color="#ffeb3b" border-bottom-left-radius="10"></rectangle>
```

RESULT

» border-bottom-right-radius

Default: 0.

The `border-bottom-right-radius` attribute sets the radius of the bottom right corner of the rectangle.

TEMPLATE

```
<rectangle position="relative" width="150" height="50" border-width="3"
    border-color="#f44336" border-bottom-right-radius="10"></rectangle>
```

RESULT

Table of contents ⇑

ELLIPSES

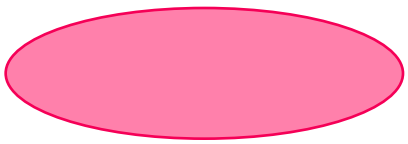
Ellipses can be rendered using the **ellipse** component. The component is represented in templates using the <ellipse> tag.

Basic syntax of ellipses:

TEMPLATE

```
<ellipse position="relative" width="150" height="50"
    fill-color="#ff80ab" border-color="#f50057"></ellipse>
```

RESULT



Supported attributes:

» width

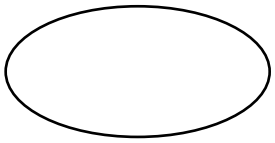
Default: `0`.

The `width` attribute sets the width of the ellipse.

TEMPLATE

```
<ellipse position="relative" width="100" height="50"></ellipse>
```

RESULT



» height

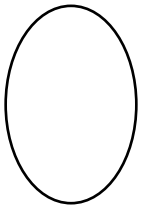
Default: `0`.

The `height` attribute sets the height of the ellipse.

TEMPLATE

```
<ellipse position="relative" width="50" height="75"></ellipse>
```

RESULT



» fit-mode

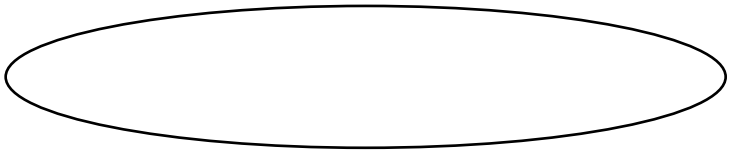
Default: `none`. Valid values: `none`, `fill-width`.

The `fit-mode` attribute controls the sizing of the component relative to the available space. When the attribute value is set to `fill-width`, the component is scaled so that it occupies the entire available width, preserving the original aspect ratio. The `width` and `height` of the ellipse must be specified in order to calculate the original aspect ratio of the ellipse.

TEMPLATE

```
<ellipse position="relative" width="1" height="0.2"
fit-mode="fill-width"></ellipse>
```

RESULT



» margin

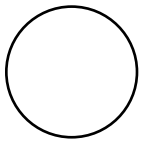
Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<ellipse position="relative" width="50" height="50"
    margin="10 0 10 100"></ellipse>
```

RESULT



» position

Default: absolute. Valid values: relative, absolute.

The `position` attribute controls whether the component uses relative or absolute position. In absolute mode, users position the component manually using the `x` and `y` attributes. In relative position mode, the component is positioned relative to the already rendered components.

» cx

Default: 0.

The `cx` attribute sets the X coordinate of the center of the component. The attribute is used only when manually positioning the component.

» cy

Default: 0.

The `cy` attribute sets the Y coordinate of the center of the component. The attribute is used only when manually positioning the component.

» fill-color

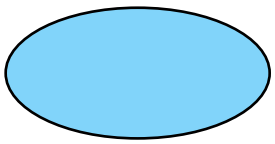
Default: #ffffff.

The `fill-color` attribute sets the fill color of the ellipse.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<ellipse position="relative" width="100" height="50"
    fill-color="#81d4fa"></ellipse>
```

RESULT



» fill-opacity

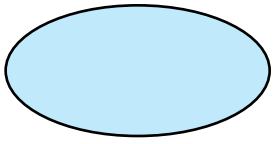
Default: 1.0. Valid values: any value between 0 and 1.

The `fill-opacity` attribute allows setting the transparency of the ellipse background.

TEMPLATE

```
<ellipse position="relative" width="100" height="50"
    fill-color="#81d4fa" fill-opacity="0.5"></ellipse>
```

RESULT



» border-width

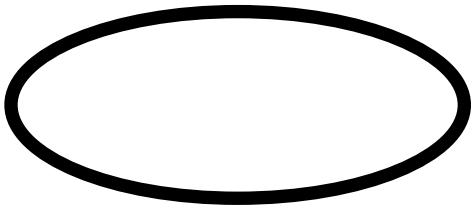
Default: 1.

The `border-width` attribute sets the width of the ellipse border.

TEMPLATE

```
<ellipse position="relative" width="175" height="75"
    border-width="5"></ellipse>
```

RESULT



» border-color

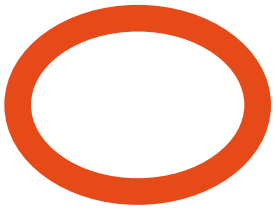
Default: #000000.

The `border-color` attribute sets the border color of the ellipse.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<ellipse position="relative" width="100" height="75" border-width="10"
    border-color="#e64a19"></ellipse>
```

RESULT



» border-opacity

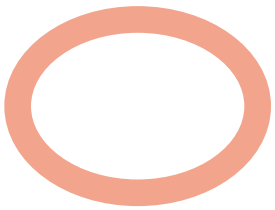
Default: 1.0. Valid values: any value between 0 and 1.

The `border-opacity` attribute allows setting the transparency of the ellipse border.

TEMPLATE

```
<ellipse position="relative" width="100" height="75" border-width="10"
    border-color="#e64a19" border-opacity="0.5"></ellipse>
```

RESULT



LINES

Lines can be rendered using the **line** component. The component is represented in templates using the <line> tag.

Basic syntax of lines:

TEMPLATE

```
<line position="relative" x1="10" y1="10" x2="280" y2="30"
    color="#dd2c00"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    color="#1b5e20"></line>
```

RESULT

Supported attributes:

» fit-mode

Default: `none`. Valid values: `none`, `fill-width`.

The `fit-mode` attribute controls the sizing of the component relative to the available space. When the attribute value is set to `fill-width`, the component is scaled so that it occupies the entire available width, preserving the original orientation of the line. In `fill-width` mode, if (`x1`, `y1`) and (`x2`, `y2`) are specified, they are used only to determine orientation of the line.

TEMPLATE

```
<line position="relative" fit-mode="fill-width"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    x1="0" y1="20" x2="0" y2="0"></line>
```

RESULT

» color

Default: `#000000`.

The `color` attribute sets the color of the line.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    color="#283593"></line>
```

RESULT

» opacity

Default: `1.0`. Valid values: `any value between 0 and 1`.

The `opacity` attribute allows setting the transparency of the line.

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    color="#283593" opacity="0.5"></line>
```

RESULT

» thickness

Default: `1`.

The `thickness` attribute sets the thickness of the line.

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#ad1457"
    thickness="5"></line>
```

RESULT

» style

Default: solid. Valid values: solid, dashed.

The `style` attribute sets the style of the line.

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    color="#7f0000"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width"
    color="#7f0000" style="dashed"></line>
```

RESULT

» dash-array

Default: [1].

The `dash-array` attribute controls the pattern of dashes and gaps used to paint the line. It specifies both the pattern and the lengths of the dashes and gaps. The attribute has no effect if the `style` attribute is not set to `dashed`.

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#00b8d4"
    style="dashed" dash-array="4 1"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#00b8d4"
    style="dashed" dash-array="1 4"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#00b8d4"
    style="dashed" dash-array="4 1 2"></line>
```

RESULT

» dash-phase

Default: 0.

The `dash-phase` attribute controls the distance into the dash pattern at which to start the dash. The attribute has no effect if the `style` attribute is not set to `dashed`.

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#8b6b61"
    style="dashed" dash-array="1 3 5 7"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#8b6b61"
    style="dashed" dash-array="1 3 5 7" dash-phase="3"></line>
```

RESULT

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#8b6b61"
    style="dashed" dash-array="1 3 5 7" dash-phase="6"></line>
```

RESULT

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<line position="relative" fit-mode="fill-width" color="#1565c0"
    margin="5 25"></line>
```

RESULT

» position

Default: absolute. Valid values: relative, absolute.

The `position` attribute controls whether the component uses relative or absolute position. In absolute mode, users position the component manually using the (`x1`, `y1`) and (`x2`, `y2`) pairs of attributes. In relative position mode, the component is positioned relative to the already rendered components.

» x1

Default: 0.

The `x1` attribute sets the X coordinate of the starting point of the line. The attribute is mainly used when manually positioning the component.

» y1

Default: 0.

The `y1` attribute sets the Y coordinate of the starting point of the line. The attribute is mainly used when manually positioning the component.

» x2

Default: 0.

The `x2` attribute sets the X coordinate of the ending point of the line. The attribute is mainly used when manually positioning the component.

» y2

Default: 0.

The `y2` attribute sets the Y coordinate of the ending point of the line. The attribute is mainly used when manually positioning the component.

CHARTS

Charts can be rendered using the **chart** component. The component is represented in templates using the <chart> tag. The charts are created using unichart, our in-house charting library. For more information, please visit github.com/unidoc/unichart.

Supported chart types:

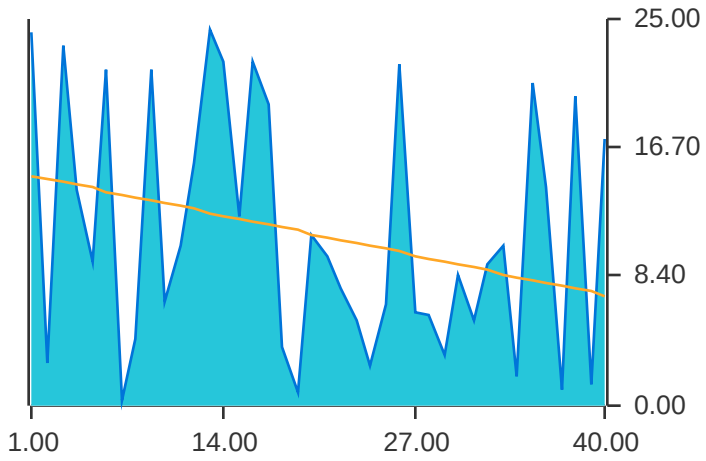
- Line chart
- Bar chart
- Stacked bar chart
- Pie chart
- Donut chart
- Scatter chart

Basic syntax of charts:

TEMPLATE

```
<chart src="line-chart-1" height="175"></chart>
```

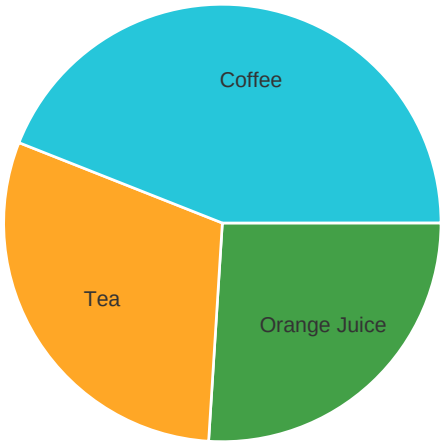
RESULT



TEMPLATE

```
<chart src="pie-chart-1" height="175"></chart>
```

RESULT



Supported attributes:

» src

Default: empty string.

The `src` attribute is used to specify the name of the chart to render. The referenced source chart is loaded from the chart map of the options used to draw the template.
For more information see chapter 5.0. Golang Interface.

TEMPLATE

```
<chart src="line-chart-1"></chart>
```

» x

Default: 0.

The `x` attribute sets the X coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» y

Default: 0.

The `y` attribute sets the Y coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» width

Default: all available space.

The `width` attribute sets the width of the rendered chart. The attribute is ignored if the chart is rendered in relative position mode (i.e. it is not manually positioned by the user using the `x` and `y` attributes).

TEMPLATE

```
<chart src="pie-chart-1" width="1024"></chart>
```

» height

Default: 400.

The `height` attribute sets the height of the rendered chart.

TEMPLATE

```
<chart src="donut-chart-1" height="250"></chart>
```

» margin

Default: 0.

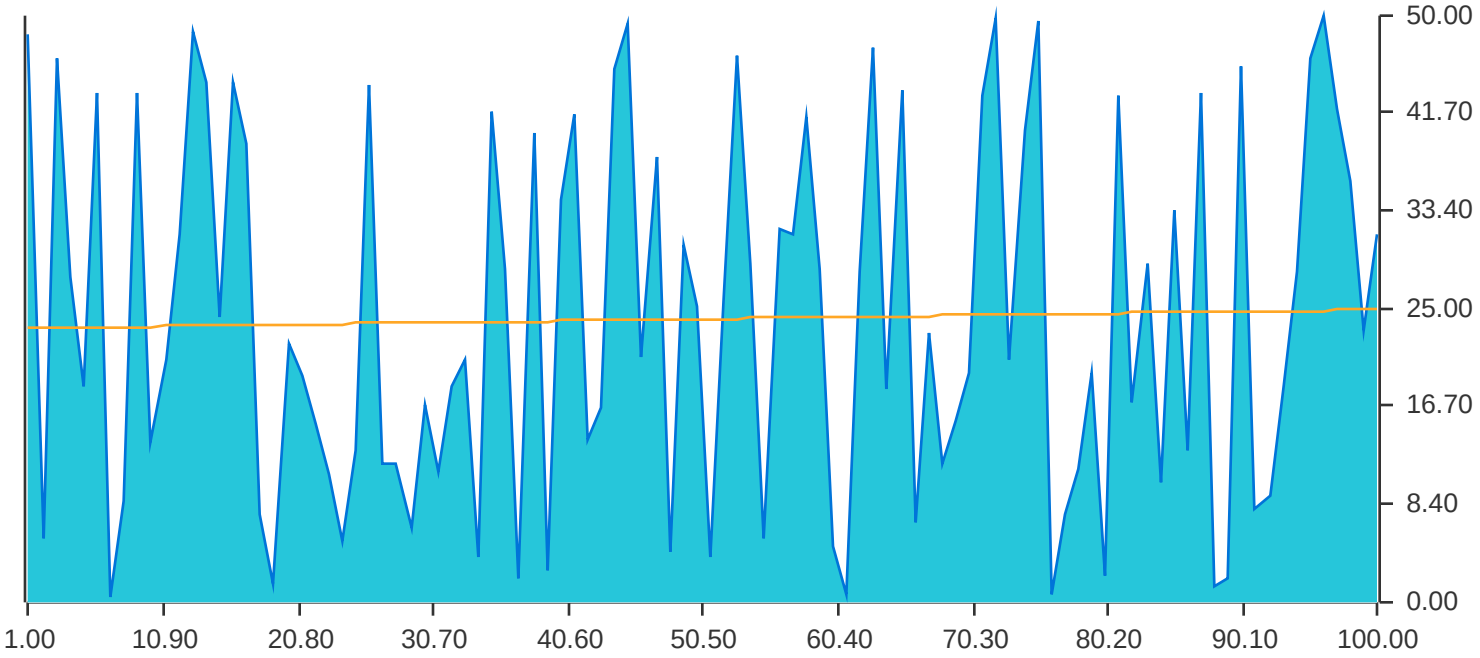
The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

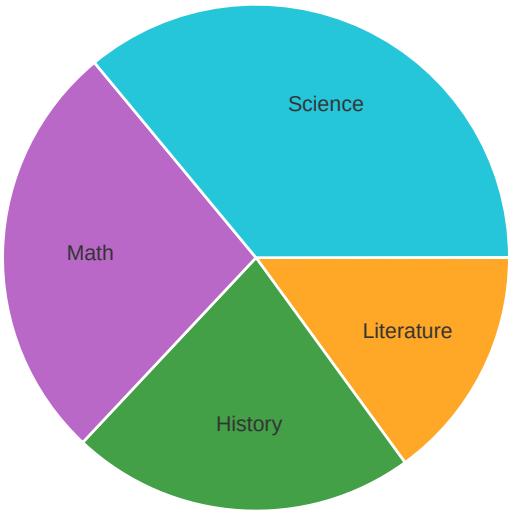
TEMPLATE

```
<chart src="bar-chart-1" margin="10 5 20 5"></chart>
```

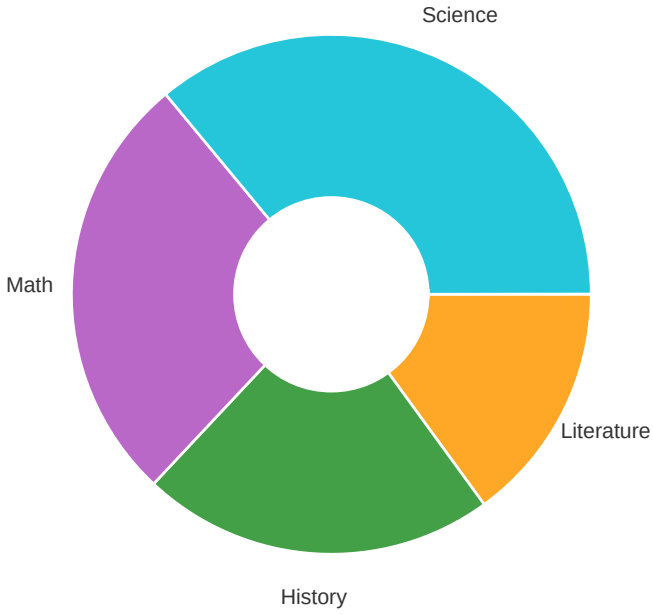
Chart types showcase:

Line Chart



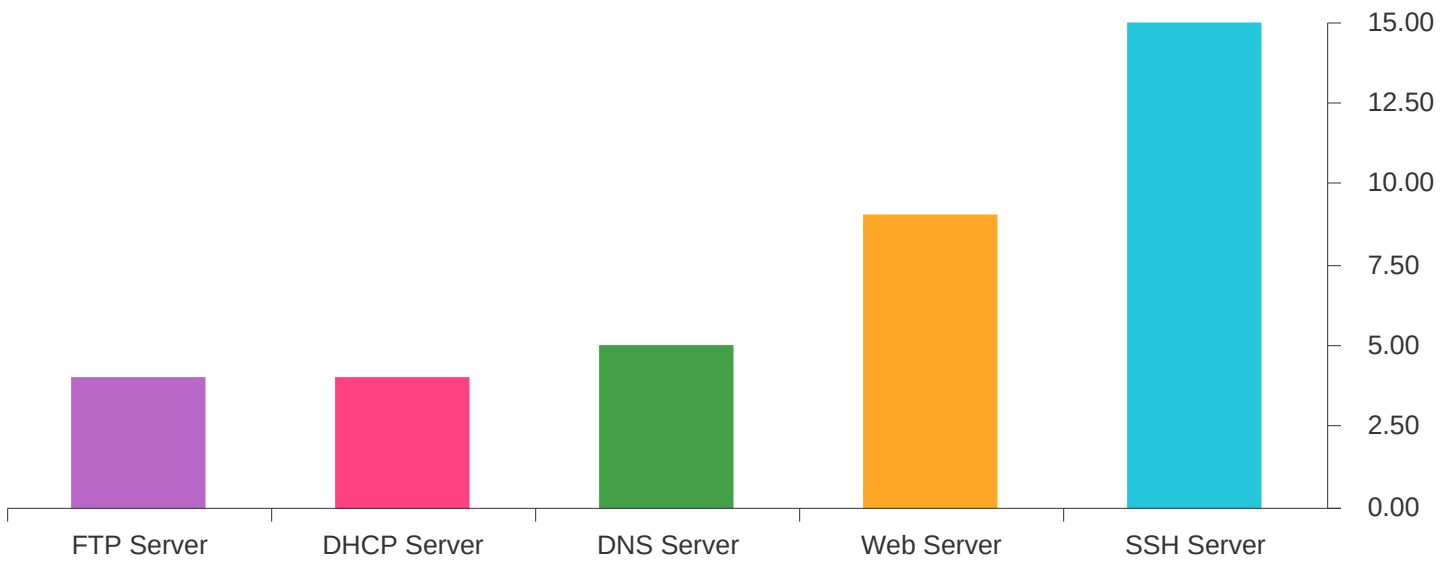


Pie Chart

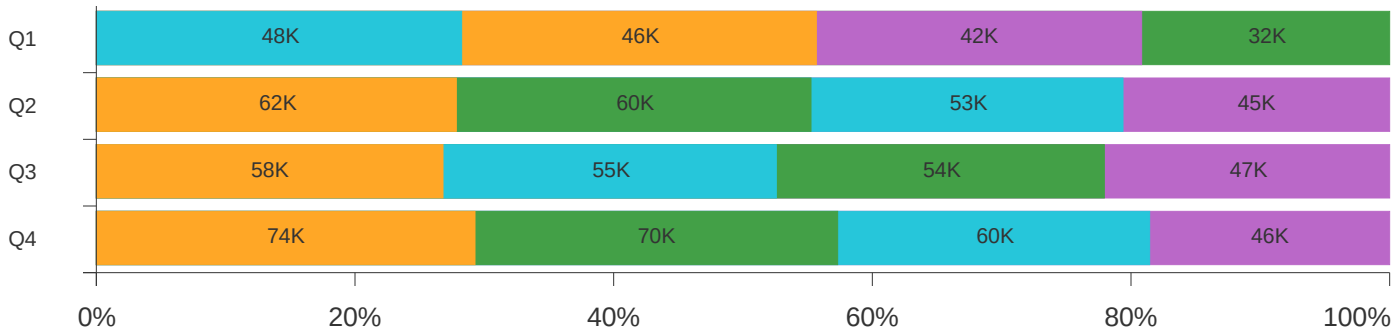


Donut Chart

Bar Chart



Stacked Bar Chart



Full documentation and more examples of charts can be found at github.com/unidoc/unichart.

PAGE BREAKS

The **page break** component has a single purpose and that is to end the current page and start a new one. The component is represented in templates using the <page-break> tag.

Basic syntax of page breaks:

TEMPLATE

```
<page-break></page-break>
```

CONTAINER COMPONENTS

Container components are used to group other components, usually with the goal of presenting them in a structured way. Most creator **container** components can be used in templates, with support for more to come in the near future.

Here is the list of currently supported components:

- division
  - background
- table
  - table-cell
- list
  - list-item
  - list-marker
- chapter
  - chapter-heading

DIVISIONS

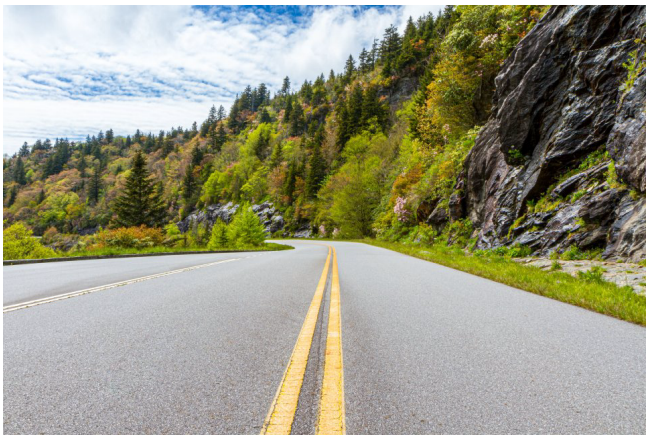
The **division** component is used to group a collection of components, stacking them vertically. Optionally, divisions can have configurable backgrounds. Basically, a background can be assigned to all supported components by including them a division. The component is represented in templates using the <division> tag.

Basic syntax of divisions:

TEMPLATE

```
<division padding="10" margin="5">
    <background border-color="#0772cd" border-size="0.5"
        border-radius="5"></background>

    <image fit-mode="fill-width"
        src="path('templates/res/images/sample-image-5.jpg')"></image>

    <line position="relative" fit-mode="fill-width"
        thickness="0.5" color="#333333" margin="5 0"></line>

    <paragraph text-align="center">
        <text-chunk>A seemingly endless road...</text-chunk>
    </paragraph>
</division>
```

RESULT



A seemingly endless road...

Supported attributes:

» enable-page-wrap

Default: `true`. Valid values: `true`, `false`.

The `enable-page-wrap` attribute controls whether the division is wrapped across pages. Page wrapping is enabled by default. When the attribute is set to `false`, the division is moved in its entirety on a new page, if it does not fit in the available height. If the height of the division is larger than an entire page, wrapping is enabled automatically in order to avoid unwanted behavior.

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<division>
    <background border-color="#cccccc" border-size="0.5"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

Sample text

TEMPLATE

```
<division margin="10">
    <background border-color="#cccccc" border-size="0.5"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

Sample text

» padding

Default: 0.

The `padding` attribute allows setting a configurable amount of space inside the component, applied around its children. The background of the division is not affected by the padding attribute.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<division>
    <background border-color="#cccccc" border-size="0.5"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

Sample text

TEMPLATE

```
<division padding="10">
    <background border-color="#cccccc" border-size="0.5"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

Sample text

DIVISION » BACKGROUNDS

A background can be added to a division by assigning it a **background** component. The background component is represented in templates using the <background> tag.

Supported attributes:

» fill-color

Default: #ffffff.

The `fill-color` attribute sets the fill color of the division background.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<division padding="5" margin="5">
    <background fill-color="#00695c"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk color="#ffffff">Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

Sample text

» border-size

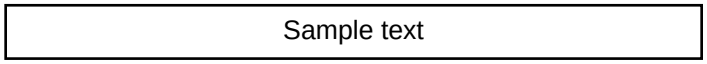
Default: 0.

The `border-color` attribute sets the border size of the division background.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="1" border-color="#000000"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT



» border-color

Default: #ffffff.

The `border-color` attribute sets the border color of the division background.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="1" border-color="#5c007a"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

» border-radius

Default: 0.

The `border-radius` attribute sets the radius used when rendering the corners of the division background.
For more information see chapter 4.5. Border Radius.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="3" border-color="#999999"
        border-radius="5"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

» border-top-left-radius

Default: 0.

The `border-top-left-radius` attribute sets the radius of the top left corner of the division background.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="3" border-color="#283593"
        border-top-left-radius="10"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

» border-top-right-radius

Default: 0.

The `border-top-right-radius` attribute sets the radius of the top right corner of the division background.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="3" border-color="#009688"
        border-top-right-radius="10"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT

» border-bottom-left-radius

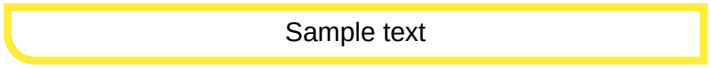
Default: 0.

The `border-bottom-left-radius` attribute sets the radius of bottom left corner of the division background.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="3" border-color="#ffeb3b"
        border-bottom-left-radius="10"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT



» border-bottom-right-radius

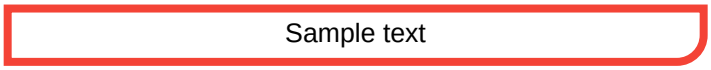
Default: 0.

The `border-bottom-right-radius` attribute sets the radius of bottom right corner of the division background.

TEMPLATE

```
<division padding="5" margin="5">
    <background border-size="3" border-color="#f44336"
        border-bottom-right-radius="10"></background>

    <paragraph text-align="center" vertical-text-align="center">
        <text-chunk>Sample text</text-chunk>
    </paragraph>
</division>
```

RESULT



TABLES

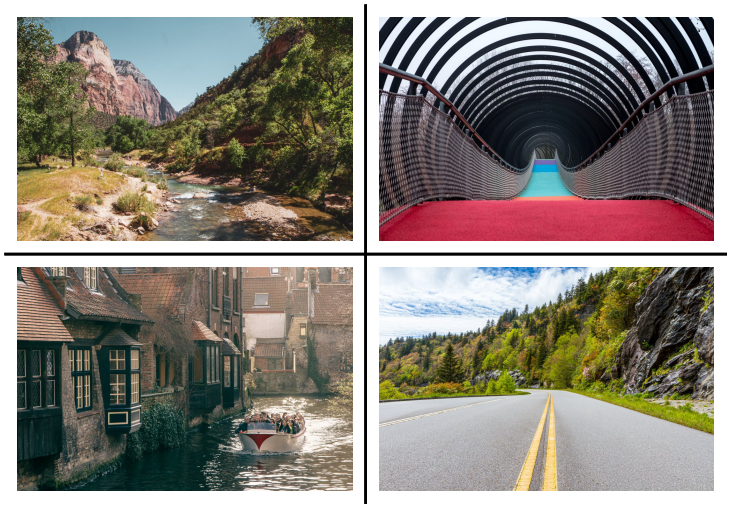
Tables can be created using the **table** component, which is somewhat similar to a **HTML table**. Tables are comprised of **cell** components, which are grouped into **rows** based on the user specified number of **columns**. The component is represented in templates using the <table> tag.

Basic syntax of tables:

TEMPLATE

```
<table columns="2">
    <table-cell indent="0" border-width-right="1" border-width-bottom="1">
        <image src="path('templates/res/images/sample-image-2.jpg')"
            fit-mode="fill-width" margin="5"></image>
    </table-cell>
    <table-cell indent="0" border-width-left="1" border-width-bottom="1">
        <image src="path('templates/res/images/sample-image-4.jpg')"
            fit-mode="fill-width" margin="5"></image>
    </table-cell>
    <table-cell indent="0" border-width-right="1" border-width-top="1">
        <image src="path('templates/res/images/sample-image-3.jpg')"
            fit-mode="fill-width" margin="5"></image>
    </table-cell>
    <table-cell indent="0" border-width-left="1" border-width-top="1">
        <image src="path('templates/res/images/sample-image-5.jpg')"
            fit-mode="fill-width" margin="5"></image>
    </table-cell>
</table>
```

RESULT



TEMPLATE

```
<table columns="3">
    <table-cell border-width="1" align="center" colspan="3">
        <paragraph>
            <text-chunk font="helvetica-bold">Warm colors</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#d32f2f">Red</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#e64a19">Orange</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#ffc107">Yellow</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" colspan="3">
        <paragraph>
            <text-chunk font="helvetica-bold">Cool colors</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#1565c0">Blue</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#1b5e20">Green</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk color="#8e24aa">Purple</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Warm colors		
Red	Orange	Yellow
Cool colors		
Blue	Green	Purple

Supported attributes:

» columns

Default: 1.

The `columns` attribute sets the number of columns the table has. The attribute basically controls how the cells are split into rows.

TEMPLATE

```
<table>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>D</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A
B
C
D

TEMPLATE

```
<table columns="2">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>D</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B
C	D

TEMPLATE

```
<table columns="4">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>D</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B	C	D

» column-widths

Default: 1.0 / column count.

The `column-widths` attribute allows setting the sizes of the columns. The column sizes are specified as an array of ratios between `0.0` and `1.0` and the sum of the elements must equal `1.0`. If the attribute is not provided, all columns are considered equal.

TEMPLATE

```
<table columns="3">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B	C

TEMPLATE

```
<table columns="3" column-widths="0.2 0.5 0.3">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B	C

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<table columns="3">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B	C

TEMPLATE

```
<table columns="3" margin="30 20 0 20">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

A	B	C

» header-start-row

Default: 0.

The `header-start-row` attribute sets the starting row of the optional table header. The header rows are repeated on every page the table spans. If the `header-end-rows` attribute is not specified or its value is 0, the attribute has no effect.

TEMPLATE

```
<table columns="2" header-start-row="1" header-end-row="1">
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk font="helvetica-bold">Header column 1</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph>
            <text-chunk font="helvetica-bold">Header column 2</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Header column 1	**Header column 2**
A	B

» header-end-row

Default: 0.

The `header-end-row` attribute sets the ending row of the optional table header. The header rows are repeated on every page the table spans. If the `header-start-rows` attribute is not specified or its value is 0, the attribute has no effect.

TEMPLATE

```
<table columns="2" header-start-row="1" header-end-row="2">
    <table-cell border-width="1" align="center" colspan="2">
        <paragraph>
            <text-chunk font="helvetica-bold">Header row 1</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" colspan="2">
        <paragraph>
            <text-chunk font="helvetica-bold">Header row 2</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Header row 1	
Header row 2	
A	B

» x

Default: 0.

The `x` attribute sets the X coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» y

Default: 0.

The `y` attribute sets the Y coordinate of the top left corner of the component. The attribute is used only when manually positioning the component.

» enable-page-wrap

Default: true. Valid values: true, false.

The `enable-page-wrap` attribute controls whether the table is wrapped across pages. Page wrapping is enabled by default. When the attribute is set to `false`, the table is moved in its entirety on a new page if it does not fit in the available height. If the height of the table is larger than an entire page, wrapping is enabled automatically in order to avoid unwanted behavior.

» enable-row-wrap

Default: false. Valid values: true, false.

The `enable-page-wrap` attribute controls whether the individual table rows are wrapped across pages, basically splitting their content. Row wrapping is currently available for table cells consisting of paragraph and division components. The behavior is disabled by default.

TABLE » CELLS

Table cells are the main building blocks of the **table** component. The cells are arranged into **rows** based on the number of **columns** the parent **table** has. The component is represented in templates using the <table-cell> tag.

Supported attributes:

» colspan

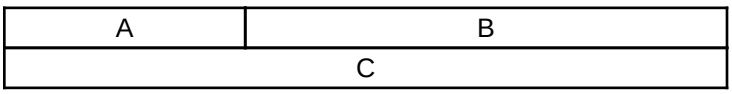
Default: 1.

The `colspan` attribute sets the number of columns a cell spans.

TEMPLATE

```
<table columns="3">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" colspan="2">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" colspan="3">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT



» rowspan

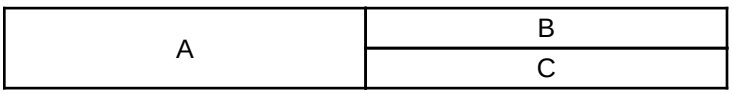
Default: 1.

The `rowspan` attribute sets the number of rows a cell spans.

TEMPLATE

```
<table columns="2">
    <table-cell border-width="1" align="center"
        vertical-align="middle" rowspan="2">
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT



» indent

Default: 5.

The `indent` attribute represents an horizontal offset applied on the left of the table cell content.

TEMPLATE

```
<table>
    <table-cell border-width="1">
        <paragraph>
            <text-chunk>A</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" indent="0">
        <paragraph>
            <text-chunk>B</text-chunk>
        </paragraph>
    </table-cell>
    <table-cell border-width="1" indent="15">
        <paragraph>
            <text-chunk>C</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

A
B
C

» align

Default: left. Valid values: left, center, right.

The `align` attribute sets the horizontal alignment of the cell content.

TEMPLATE

```
<table columns="3">
    <table-cell border-width="1">
        <paragraph><text-chunk>Left</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>Center</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="right">
        <paragraph><text-chunk>Right</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Left	Center	Right

» vertical-align

Default: top. Valid values: top, middle, bottom.

The `vertical-align` attribute sets the vertical alignment of the cell content.

TEMPLATE

```
<table columns="4">
    <table-cell border-width="1" align="center">
        <paragraph><text-chunk>Top</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" vertical-align="middle">
        <paragraph><text-chunk>Middle</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" align="center" vertical-align="bottom">
        <paragraph><text-chunk>Bottom</text-chunk></paragraph>
    </table-cell>
    <table-cell border-width="1" indent="0" align="center">
        <image src="path('templates/res/images/sample-image-2.jpg')"
            fit-mode="fill-width" margin="5 5 2 2"></image>
    </table-cell>
</table>
```

RESULT

Top	Middle	Bottom	

» border-width

Default: 0.

The `border-width` attribute sets the width of the cell border.

TEMPLATE

```
<table>
    <table-cell>
        <paragraph><text-chunk>No border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

No border

TEMPLATE

```
<table>
    <table-cell border-width="2">
        <paragraph><text-chunk>Border width 2</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Border width 2

» border-width-left

Default: 0.

The `border-width-left` attribute sets the width of the left cell border.

TEMPLATE

```
<table>
    <table-cell border-width="1" border-width-left="3">
        <paragraph><text-chunk>Left border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Left border

» border-width-right

Default: 0.

The `border-width-right` attribute sets the width of the right cell border.

TEMPLATE

```
<table>
    <table-cell align="right" border-width="1" border-width-right="3">
        <paragraph><text-chunk>Right border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Right border

» border-width-top

Default: 0.

The `border-width-top` attribute sets the width of the top cell border.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1" border-width-top="3">
        <paragraph><text-chunk>Top border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Top border

» border-width-bottom

Default: 0.

The `border-width-bottom` attribute sets the width of the bottom cell border.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1" border-width-bottom="3">
        <paragraph><text-chunk>Bottom border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Bottom border

» border-style

Default: single. Valid values: single, double.

The `border-style` attribute sets the style of the cell border.

TEMPLATE

```
<table>
    <table-cell border-width="1">
        <paragraph><text-chunk>Single border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Single border

TEMPLATE

```
<table>
    <table-cell border-width="1" border-style="double">
        <paragraph><text-chunk>Double border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Double border

» border-style-left

Default: single. Valid values: single, double.

The `border-style-left` attribute sets the style of the left cell border.

TEMPLATE

```
<table>
    <table-cell border-width="1" border-style-left="double">
        <paragraph><text-chunk>Left border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Left border

» border-style-right

Default: single. Valid values: single, double.

The `border-style-right` attribute sets the style of the right cell border.

TEMPLATE

```
<table>
    <table-cell align="right" border-width="1" border-style-right="double">
        <paragraph><text-chunk>Right border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Right border

» border-style-top

Default: single. Valid values: single, double.

The `border-style-top` attribute sets the style of the top cell border.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1" border-style-top="double">
        <paragraph><text-chunk>Top border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Top border

» border-style-bottom

Default: single. Valid values: single, double.

The `border-style-bottom` attribute sets the style of the bottom cell border.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1"
        border-style-bottom="double">
        <paragraph><text-chunk>Bottom border</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Bottom border

» border-color

Default: #000000.

The `border-color` attribute sets the color of the cell border.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell border-width="3">
        <paragraph>
            <text-chunk>Default border color</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Default border color

TEMPLATE

```
<table>
    <table-cell border-width="3" border-color="#ff6d00">
        <paragraph>
            <text-chunk>Custom border color</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Custom border color

» border-color-left

Default: #000000.

The `border-color-left` attribute sets the color of the left cell border.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell border-width="3" border-color="#cccccc"
        border-color-left="#004d40">
        <paragraph>
            <text-chunk>Border color left</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

» border-color-right

Default: #000000.

The `border-color-right` attribute sets the color of the right cell border.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell align="right" border-width="3" border-color="#cccccc"
        border-color-right="#fdd835">
        <paragraph>
            <text-chunk>Border color right</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

» border-color-top

Default: #000000.

The `border-color-top` attribute sets the color of the top cell border.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="3" border-color="#cccccc"
        border-color-top="#283593">
        <paragraph>
            <text-chunk>Border color top</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

» border-color-bottom

Default: #000000.

The `border-color-bottom` attribute sets the color of the bottom cell border.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="3" border-color="#cccccc"
        border-color-bottom="#c62828">
        <paragraph>
            <text-chunk>Border color bottom</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

» border-line-style

Default: solid. Valid values: solid, dashed.

The `border-line-style` attribute sets the line style of the cell border.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="3" border-color="#f50057">
        <paragraph>
            <text-chunk>Line style solid</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Line style solid

TEMPLATE

```
<table>
    <table-cell align="center" border-width="3" border-color="#f50057"
        border-line-style="dashed">
        <paragraph>
            <text-chunk>Line style dashed</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Line style dashed

» background-color

Default: #ffffff.

The `background-color` attribute sets the background color of the cell.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1">
        <paragraph><text-chunk>Default background</text-chunk></paragraph>
    </table-cell>
</table>
```

RESULT

Default background

TEMPLATE

```
<table>
    <table-cell align="center" border-width="1" border-color="#102027"
        background-color="#62727b">
        <paragraph>
            <text-chunk color="#ffffff">Custom background</text-chunk>
        </paragraph>
    </table-cell>
</table>
```

RESULT

Custom background

LISTS

The **list** component is used to create a list of items and is similar to a **HTML list**. Currently, the **paragraph** and **list** components are supported as list items. The component is represented in templates using the <list> tag.

Basic syntax of lists:

TEMPLATE

```
<list>
    <list-item>
        <paragraph><text-chunk>Coffee</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>Tea</text-chunk></paragraph>
    </list-item>
    <list-item>
        <list-marker> </list-marker>
        <list>
            <list-marker>» </list-marker>
            <list-item>
                <paragraph><text-chunk>Rooibos</text-chunk></paragraph>
            </list-item>
            <list-item>
                <paragraph><text-chunk>Oolong</text-chunk></paragraph>
            </list-item>
        </list>
    </list-item>
    <list-item>
        <paragraph><text-chunk>Milk</text-chunk></paragraph>
    </list-item>
</list>
```

RESULT

- Coffee
- Tea
  - » Rooibos
  - » Oolong
- Milk

Supported attributes:

» margin

```
Default: 0.
```

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<list margin="5 0 5 20">
    <list-item>
        <paragraph><text-chunk>Leonardo</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>Donatello</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>Raphael</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>Michelangelo</text-chunk></paragraph>
    </list-item>
</list>
```

RESULT

- Leonardo
- Donatello
- Raphael
- Michelangelo

» indent

```
Default: 0.
```

The `indent` attribute represents an horizontal offset applied on the left side of the list items. By default, for list items of type list, the `indent` attribute is 15. Otherwise, it defaults to 0.

TEMPLATE

```
<list indent="5">
    <list-item>
        <paragraph><text-chunk>Blue shades</text-chunk></paragraph>
    </list-item>
    <list-item>
        <list-marker> </list-marker>
        <list indent="10">
            <list-item>
                <paragraph>
                    <text-chunk color="#48aaad">Teal</text-chunk>
                </paragraph>
            </list-item>
            <list-item>
                <paragraph>
                    <text-chunk color="#241571">Berry</text-chunk>
                </paragraph>
            </list-item>
        </list>
    </list-item>
</list>
```

RESULT

- Blue shades
  - Teal
  - Berry

LIST » ITEMS

List items are the main building blocks of the **list** component. Currently, list items can be either **paragraphs** or **lists** . The component is represented in templates using the <list-item> tag.

Supported attributes:

List items have no attributes.

LIST » MARKERS

List markers are sequences of symbols which precede each **list item**. By default, the list marker is the **bullet (`•`)** symbol. The component is represented in templates using the <list-marker> tag. The tag is supported by both the **list** and **list item** components. All list items inherit the properties of the list marker defined by the list. However, individual list items can override all or some of the properties of the list marker.

Basic syntax of list markers:

TEMPLATE

```
<list>
    <list-item>
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </list-item>
</list>
```

RESULT

- A
- B
- C

TEMPLATE

```
<list>
    <list-marker font="zapf-dingbats" text-rise="4"> </list-marker>
    <list-item>
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </list-item>
    <list-item>
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </list-item>
</list>
```

RESULT

☞ A
☞ B
☞ C

TEMPLATE

```
<list>
    <list-item>
        <list-marker color="#c62828">1. </list-marker>
        <paragraph><text-chunk>A</text-chunk></paragraph>
    </list-item>
    <list-item>
        <list-marker color="#1b5e20">2. </list-marker>
        <paragraph><text-chunk>B</text-chunk></paragraph>
    </list-item>
    <list-item>
        <list-marker color="#0d47a1">3. </list-marker>
        <paragraph><text-chunk>C</text-chunk></paragraph>
    </list-item>
</list>
```

RESULT

1. A
2. B
3. C

Supported attributes:

List markers are basically **text chunks** so they share the same attributes. Please see chapter 2.1.1. Text Chunks in order to see the full list of supported attributes.

CHAPTERS

Chapters are used to group content into logical units, much like a book does. They are top-level components, the only valid parent a chapter can have is another chapter, thus making it a subchapter. **Chapters** automatically appear in the document's table of contents. The component is represented in templates using the <chapter> tag.

Basic syntax of chapters:

TEMPLATE

```
<chapter>
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

TEMPLATE

```
<chapter>
    <chapter-heading>Chapter title</chapter-heading>
    <chapter>
        <chapter-heading>Subchapter title</chapter-heading>
    </chapter>
</chapter>
```

Supported attributes:

» show-numbering

Default: `true`. Valid values: `true`, `false`.

The `show-numbering` attribute controls whether the chapter numbers are displayed.

TEMPLATE

```
<chapter>
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

TEMPLATE

```
<chapter show-numbering="false">
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

» include-in-toc

Default: `true`. Valid values: `true`, `false`.

The `include-in-toc` attribute controls whether the chapter is included in the document's table of contents.

TEMPLATE

```
<chapter>
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

TEMPLATE

```
<chapter include-in-toc="false">
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

» margin

Default: `0`.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<chapter margin="5 10">
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

CHAPTER » HEADINGS

Chapter headings are used to display the title and optionally the number of a **chapter**. The component is represented in templates using the <chapter-heading> tag.

Supported attributes:

» font

Default: helvetica.

The `font` attribute specifies the font used for rendering the chapter heading.
For more information see chapter 4.1. Fonts.

TEMPLATE

```
<chapter>
    <chapter-heading font="courier">Chapter title</text-heading>
</chapter>
```

» font-size

Default: 10.

The `font-size` attribute specifies the size of the font used by the chapter heading.

TEMPLATE

```
<chapter>
    <chapter-heading font-size="12">Chapter title</chapter-heading>
</chapter>
```

» text-align

Default: left. Valid values: left, right, center, justify.

The `text-align` attribute aligns the chapter heading horizontally based on the specified option.

TEMPLATE

```
<chapter text-align="center">
    <chapter-heading>Chapter title</chapter-heading>
</chapter>
```

» line-height

Default: 1.0.

The `line-height` represents a scale factor for the vertical space a text line takes. Basically, the line height attribute controls amount of vertical space between the chapter heading text lines.

TEMPLATE

```
<chapter>
    <chapter-heading line-height="1.5">Chapter title</chapter-title>
</chapter>
```

» enable-wrap

Default: true.

The `enable-wrap` attribute controls whether the content of the chapter heading should be split into lines based on the available space.

TEMPLATE

```
<chapter enable-wrap="false">
    <chapter-heading>Long chapter title</chapter-heading>
</chapter>
```


Table of contents ⇑

» color

Default: #000000.

The `color` attribute is used to specify the text color.
For more information see chapter 4.3. Colors.

TEMPLATE

```
<chapter>
    <chapter-heading color="#00695c">Chapter title</chapter-heading>
</chapter>
```

» margin

Default: 0.

The `margin` attribute allows setting a configurable amount of space around the component.
For more information see chapter 4.4. Margin and Padding.

TEMPLATE

```
<chapter>
    <chapter-heading margin="5 25">Chapter title</chapter-heading>
</chapter>
```

» max-lines

Default: 0.

The `max-lines` attribute specifies the maximum number of lines before the chapter heading content is truncated.

TEMPLATE

```
<chapter>
    <chapter-heading max-lines="2">Chapter title</chapter-heading>
</chapter>
```

Please see chapter 2.1. Paragraphs for visual examples of most attributes enumerated in this section.

ATTRIBUTES AND RESOURCES

This chapter includes information regarding methods of loading resources (**fonts**, **images**, **colors**) and the usage of attributes which have shorthand forms (**margin**, **border-radius**).

FONTS

There are multiple ways of loading and using **fonts** in templates. This section presents all of them, with code examples. In most cases, **fonts** are specified using the font attribute.

Using standard fonts

A list of **standard fonts** is provided by the library. These fonts are always available and can be specified by their names. If no font specified, components default to using **helvetica**.

Here is the list of available standard fonts:

- helvetica
- **helvetica-bold**
- *helvetica-oblique*
- ***helvetica-bold-oblique***
- `courier`
- **`courier-bold`**
- *`courier-oblique`*
- ***`courier-bold-oblique`***
- times
- **times-bold**
- *times-italic*
- ***times-bold-italic***
- symbol
- zapf-dingbats

Usage

TEMPLATE

```
<paragraph>
    <text-chunk color="#388e3c" font="times">Times</text-chunk>
</paragraph>
```

RESULT

Times

TEMPLATE

```
<paragraph>
    <text-chunk color="#4dd0e1"
        font="helvetica-bold">Helvetica Bold</text-chunk>
</paragraph>
```

RESULT

Helvetica Bold

TEMPLATE

```
<paragraph>
    <text-chunk color="#ff5722"
        font="courier-oblique">Courier Oblique</text-chunk>
</paragraph>
```

RESULT

Courier Oblique

TEMPLATE

```
<paragraph>
    <text-chunk color="#880e4f">Zapf Dingbats: </text-chunk>
    <text-chunk color="#880e4f" font="zapf-dingbats"> </text-chunk>
</paragraph>
```

RESULT

Zapf Dingbats: ☞ ⑩☆

TEMPLATE

```
<paragraph>
    <text-chunk color="#ffc400">Symbol: </text-chunk>
    <text-chunk color="#ffc400" font="symbol">    </text-chunk>
</paragraph>
```

RESULT

Symbol: α ≥ π ∗ β

Loading fonts from paths

Fonts can be loaded at runtime, directly from templates, by specifying their paths. Optionally, there is an option to specify the type of font (regular, composite). The provided font paths can be absolute or relative to the application binary.

Usage

TEMPLATE

```
<paragraph>
    <text-chunk font="path('templates/res/fonts/DejaVuSansMono.ttf')"
        color="#00838f">DejaVu Sans Mono</text-chunk>
</paragraph>
```

RESULT

DejaVu Sans Mono

TEMPLATE

```
<paragraph>
    <text-chunk font="path('templates/res/fonts/Roboto-Regular.ttf') type('composite')"
        color="#2e7d32">Roboto Regular</text-chunk>
</paragraph>
```

RESULT

Roboto Regular

Using preloaded fonts

Fonts can be made available to a template by loading and adding them to the **font map** of the options used to render that template. The fonts are accessed in the templates using the names assigned to them in the **font map**. Preloading fonts is useful when they are used more than once throughout templates, in order to avoid loading them multiple times.

Please see chapter 5. Golang Interface for examples of how to use the **font map**.

Usage

TEMPLATE

```
<paragraph>
    <text-chunk font="deja-vu-sans-mono"
        color="#dd2c00">deja-vu-sans-mono was preloaded for this example.</text-chunk>
</paragraph>
```

RESULT

deja-vu-sans-mono was preloaded for this example.

Using font lists

Font lists are used in order to provide acceptable fallback fonts, in case some of the requested fonts may not be available for any reason. A list of fonts can be specified by using any of the previously described methods of loading/referencing fonts. It is basically an enumaration of font preferences, separated by commas. Each of the specified fonts in the list is attempted from left to right and the first available font is selected. If none of fonts in the list is valid, **helvetica** is used a fallback.

Usage

TEMPLATE

```
<paragraph>
    <text-chunk font="path('templates/res/fonts/DejaVuSansMono.ttf'), courier"
        color="#2962ff">Font order: DejaVuSansMono (load from path) > courier (standard).</text-chunk>
</paragraph>
```

RESULT

Font order: DejaVuSansMono (load from path) > courier (standard).

TEMPLATE

```
<paragraph>
    <text-chunk font="path('templates/res/fonts/Roboto-Regular.ttf') type('composite'), deja-vu-sans-mono, times"
        color="#558b2f">Font order: Roboto-Regular (load from path, composite) > deja-vu-sans-mono (preloaded) > times (standard).</text-chunk>
</paragraph>
```

RESULT

Font order: Roboto-Regular (load from path, composite) > deja-vu-sans-mono (preloaded) > times (standard).

IMAGES

There are multiple ways of loading and using **images** in templates. This section presents all of them, with code examples. Source images are specified in the src attribute of the **image** component.

Loading images from paths

Images can be loaded at runtime, directly from templates, by specifying their paths. The provided image paths can be absolute or relative to the application binary.

Usage

TEMPLATE

```
<image src="path('templates/res/images/sample-image-3.jpg')"
    width="215" height="150" align="center"></image>
```

RESULT

Using preloaded images

Images can be made available to a template by loading and adding them to the **image map** of the options used to render that template. The images are accessed in the templates using the names assigned to them in the **image map**. Preloading images is useful when they are used more than once throughout templates, in order to avoid loading them multiple times.

Please see chapter 5. Golang Interface for examples of how to use the **image map**.

Usage

TEMPLATE

```
<image src="sample"
    width="215" height="150" align="center"></image>
```

RESULT

COLORS

In this section, we will cover all the ways of defining and using **colors** in templates. Color usage is identical with all relevant attributes (e.g. background, colors, border colors) unless noted otherwise.

Defining colors using hexadecimal notation

Colors can be defined using hex values, directly from templates. Both the regular **six-digit notation** (e.g. #00aaff) and its shorter **three-digit notation** (e.g. #0af) are supported. Hex color values are preceded by # and they are case insensitive.

Usage

TEMPLATE

```
<rectangle width="6" height="1" position="relative" fit-mode="fill-width"
    border-width="0" fill-color="#ffc107"></rectangle>
```

RESULT

TEMPLATE

```
<rectangle width="6" height="1" position="relative" fit-mode="fill-width"
    border-width="3" border-color="#0bf"></rectangle>
```

RESULT

TEMPLATE

```
<rectangle width="6" height="1" position="relative" fit-mode="fill-width"
    border-width="3" border-color="#a0f" fill-color="#EA80FC"></rectangle>
```

RESULT

Defining linear gradient patterns

Linear gradients transition colors progressively along an imaginary line. They are defined by two or more **color stops** which consist of a color and an optional starting point for that color. The starting point is defined as a percentage (0-100) of the available space. Optionally, an **angle** specified in degrees (0-360) can be specified, which rotates the entire pattern.

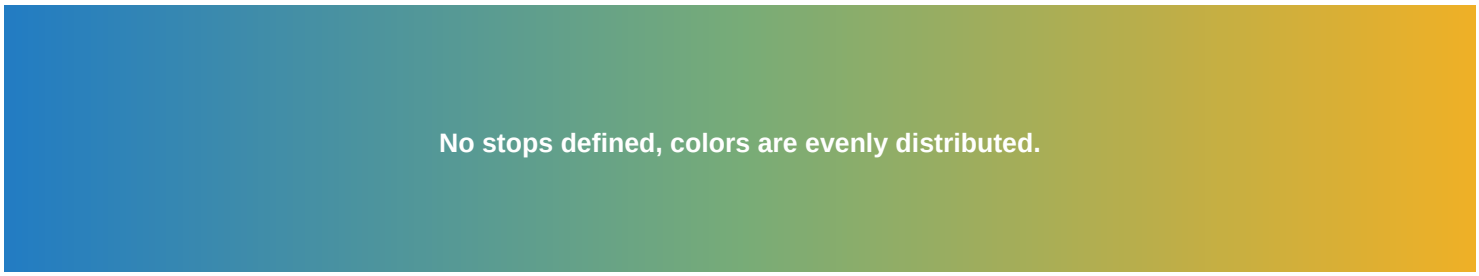
Note: currently, gradient patterns can only be applied to background colors of components. Border or outline gradient pattern colors are not supported yet.

Usage

TEMPLATE

```
<division>
    <background fill-color="linear-gradient(#227cc3, #78ac77, #f1b025)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk color="#fff" font="helvetica-bold">No stops defined, colors are evenly distributed.</text-chunk>
    </paragraph>
</division>
```

RESULT



TEMPLATE

```
<division>
    <background fill-color="linear-gradient(#227cc3, #78ac77 20%, #f1b025)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk color="#fff" font="helvetica-bold">First color stops at 20%, the rest are evenly distributed.</text-chunk>
    </paragraph>
</division>
```

RESULT

TEMPLATE

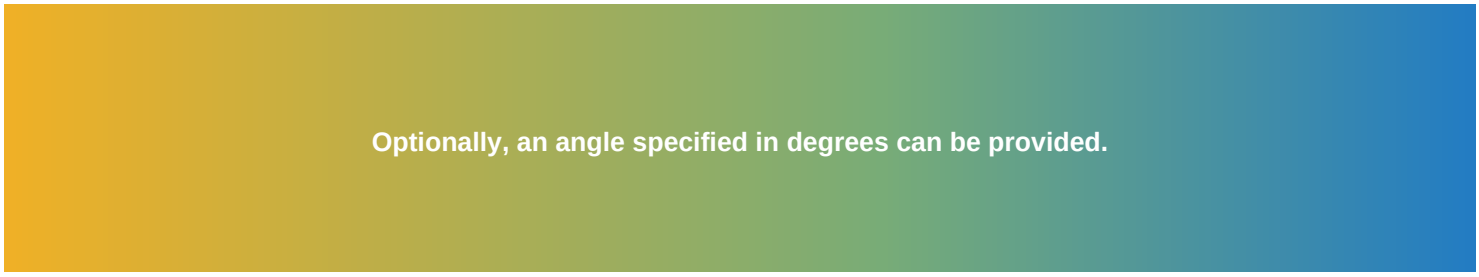
```
<division>
    <background fill-color="linear-gradient(#227cc3, #78ac77 20%, #f1b025 35%)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk color="#fff" font="helvetica-bold">First color stops at 20%, the second at 35%, the third takes the rest of the space.</text-chunk>
    </paragraph>
</division>
```

RESULT

TEMPLATE

```
<division>
    <background fill-color="linear-gradient(180deg, #227cc3, #78ac77 40%, #f1b025)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk color="#fff" font="helvetica-bold">Optionally, an angle specified in degrees can be provided.</text-chunk>
    </paragraph>
</division>
```

RESULT



Defining radial gradient patterns

Radial gradients transition colors progressively from the center of the assigned space. They are defined by two or more **color stops** which consist of a color and an optional starting point for that color. The starting point is defined as a percentage (0-100) of the available space.

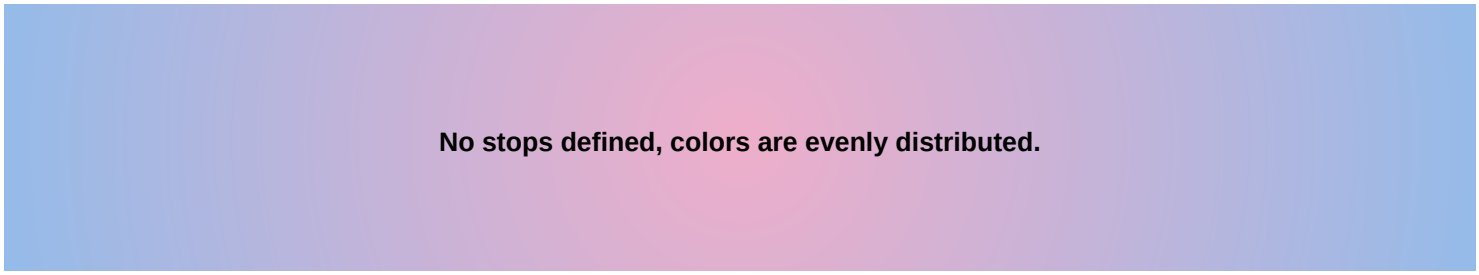
Note: currently, gradient patterns can only be applied to background colors of components. Border or outline gradient pattern colors are not supported yet.

Usage

TEMPLATE

```
<division>
    <background fill-color="radial-gradient(#eeaeca, #c3b4d9, #94bbe9)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk font="helvetica-bold">No stops defined, colors are evenly distributed.</text-chunk>
    </paragraph>
</division>
```

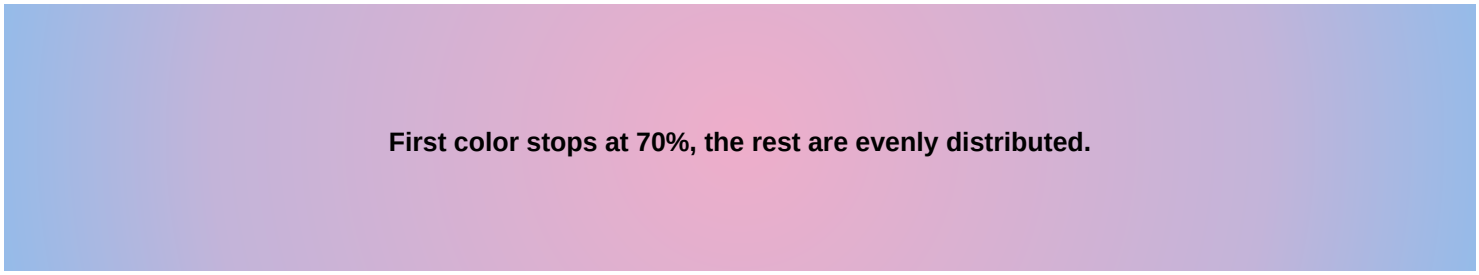
RESULT



TEMPLATE

```
<division>
    <background fill-color="radial-gradient(#eeaeca, #c3b4d9 70%, #94bbe9)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk font="helvetica-bold">First color stops at 70%, the rest are evenly distributed.</text-chunk>
    </paragraph>
</division>
```

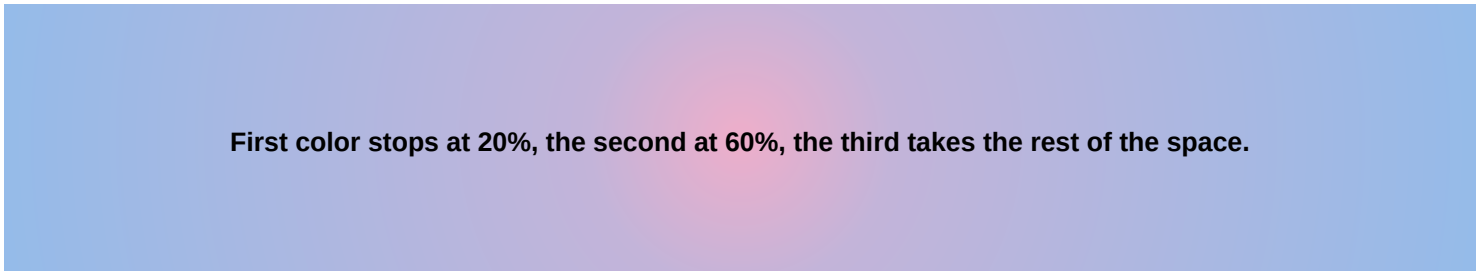
RESULT



TEMPLATE

```
<division>
    <background fill-color="radial-gradient(#eeaeca, #c3b4d9 20%, #94bbe9 60%)"></background>

    <paragraph text-align="center" margin="45 0">
        <text-chunk font="helvetica-bold">First color stops at 20%, the second at 60%, the third takes the rest of the space.</text-chunk>
    </paragraph>
</division>
```

RESULT



Using preloaded colors

Colors can be made available to a template by creating and adding them to the **color map** of the options used to render that template. The colors are accessed in the templates using the names assigned to them in the **color map**. Preloading colors is useful when they are used more than once throughout templates, in order to avoid defining them multiple times.

Please see chapter 5. Golang Interface for examples of how to use the **color map**.

Usage

TEMPLATE

```
<rectangle width="6" height="1" position="relative" fit-mode="fill-width"
    border-width="0" fill-color="secondary"></rectangle>
```

RESULT

TEMPLATE

```
<rectangle width="6" height="1" position="relative" fit-mode="fill-width"
    border-width="0" fill-color="primary-bg-gradient"></rectangle>
```

RESULT

MARGIN AND PADDING

The margin and padding attributes are used generate configurable amounts of space for a component. The difference between them is that **margin** generates space around the exterior of components, whereas **padding** generates space inside components, around their content. Almost all components have a margin attribute whereas padding is currently available only for **divisions**. The syntax of both attributes is identical.

Margin vs Padding

TEMPLATE

```
<division margin="20">
    <background border-size="5" border-color="#ffc400"></background>

    <image src="path('templates/res/images/sample-image.jpg')"
        margin="2" fit-mode="fill-width"></image>
</division>
```

RESULT

TEMPLATE

```
<division padding="20">
    <background border-size="5" border-color="#ffc400"></background>

    <image margin="2" src="path('templates/res/images/sample-image.jpg')"
        margin="2" fit-mode="fill-width"></image>
</division>
```

RESULT

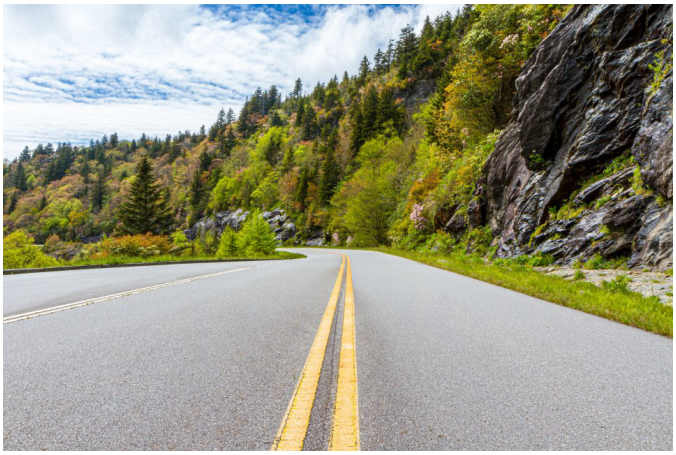
Syntax

Margin: top 10, right 10, bottom 10, left 10.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    fit-mode="fill-width" margin="10"></image>
```

RESULT

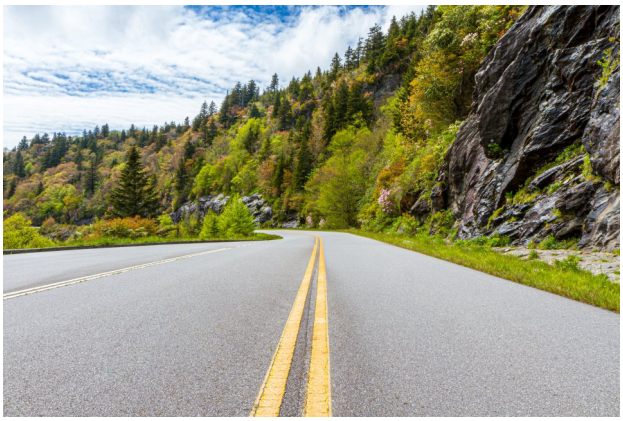


Margin: top 5, right 20, bottom 5, left 20.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    fit-mode="fill-width" margin="5 20"></image>
```

RESULT



Margin: top 0, right 20, bottom 5, left 20.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    fit-mode="fill-width" margin="0 20 5"></image>
```

RESULT

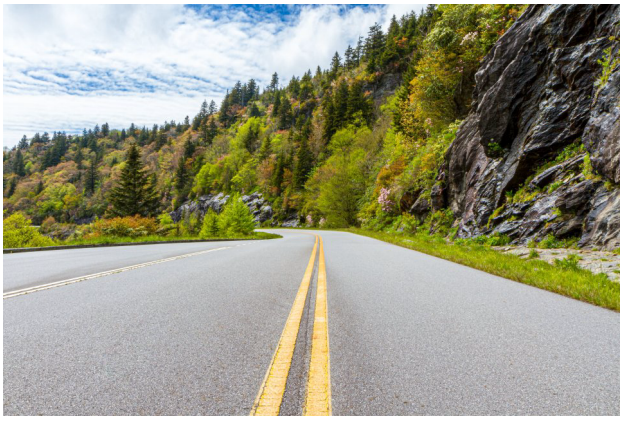


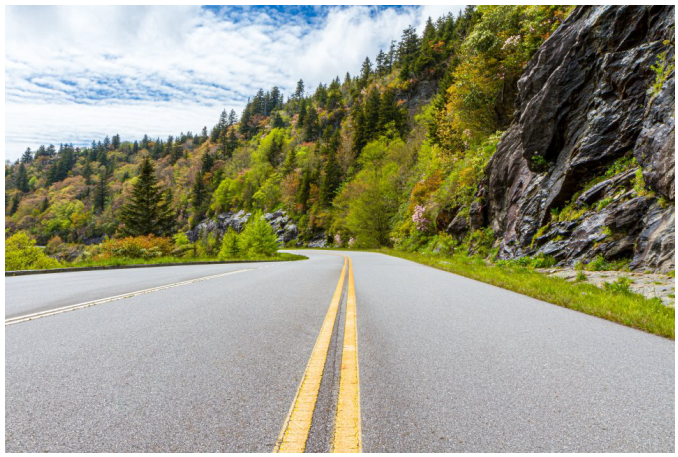
Table of contents ⇑

Margin: top 10, right 5, bottom 20, left 15.

TEMPLATE

```
<image src="path('templates/res/images/sample-image-5.jpg')"
    fit-mode="fill-width" margin="10 5 20 15"></image>
```

RESULT



BORDER RADIUS

The border-radius attribute sets the radius used when rendering border corners. The attribute can be used to set the radius for each of the border corners and is supported by most components which have border support. The border-top-left-radius, border-top-right-radius, border-bottom-left-radius and border-bottom-right-radius attributes can be used for setting the border radius of individual corners.

Syntax

Border radius: top-left 10, top-right 10, bottom-left 10, bottom-right 10.

TEMPLATE

```
<rectangle position="relative" width="250" height="100" border-width="5"
    border-color="#40c4ff" border-radius="10"></rectangle>
```

RESULT

Border radius: top-left 25, top-right 5, bottom-left 5, bottom-right 25.

TEMPLATE

```
<rectangle position="relative" width="250" height="100" border-width="5"
    border-color="#ffb300" border-radius="25 5"></rectangle>
```

RESULT

Border radius: top-left 10, top-right 0, bottom-left 0, bottom-right 25.

TEMPLATE

```
<rectangle position="relative" width="250" height="100" border-width="5"
    border-color="#e53935" border-radius="10 0 25"></rectangle>
```

RESULT

Border radius: top-left 20, top-right 5, bottom-left 30, bottom-right 0.

TEMPLATE

```
<rectangle position="relative" width="250" height="100" border-width="5"
    border-color="#388e3c" border-radius="20 5 30 0"></rectangle>
```

RESULT

GOLANG INTERFACE

Templates can be rendered using the DrawTemplate method of the **creator** package, which invokes the internal template processor. The processing of templates is done in two phases. First, templates are executed as text/template#Template instances. This allows injecting data and executing actions. The second processing phase takes the output of the first one, parses and translates it into **components**.

GO

```
func (c *Creator) DrawTemplate(r io.Reader, data interface{}, options *TemplateOptions) error
```

The method renders the template specified through the specified reader r, using the provided options. The passed in data is available in the rendered template.

Resources like **fonts**, **images**, **colors** and **charts** can be shared with the templates through the resource maps in the **template options**. In addition, **Go functions** can be made available to templates through the **helper function map**. The **subtemplates map** is used to provide additional templates which are automatically available to be accessed both from within the main template and from one another. All the resources added to the maps in the template options can be accessed in templates by their assigned name (map key).

GO

```
type TemplateOptions struct {
    FontMap  map[string]*model.PdfFont
    ImageMap map[string]*model.Image
    ColorMap map[string]Color
    ChartMap map[string]render.ChartRenderable

    HelperFuncMap  template.FuncMap
    SubtemplateMap map[string]io.Reader
}
```

Font map

The **font map** is used to pass fonts to templates. The fonts are accessed in templates using their assigned map key.

TEMPLATE

```
<paragraph>
    <text-chunk color="#4dd0e1" font-size="12"
        font="deja-vu-sans-mono">Sample text</text-chunk>
</paragraph>
```

RESULT

Sample text

GO

```
// Load DejaVuSansMono font.
dejaVuSansMono, err := model.NewPdfFontFromTTFFile("templates/res/fonts/DejaVuSansMono.ttf")
if err != nil {
    log.Fatal(err)
}

// Create template options.
tplOpts := &creator.TemplateOptions{
    FontMap: map[string]*model.PdfFont{
        "deja-vu-sans-mono": dejaVuSansMono,
    },
}

// Draw template.
c := creator.New()
if err := c.DrawTemplate(tpl, nil, tplOpts); err != nil {
    log.Fatal(err)
}
```

Image map

The **image map** is used to pass images to templates. The images are accessed in templates using their assigned map key.

TEMPLATE

```
<image src="sample"
    width="128" height="72" align="center"></image>
```

RESULT

GO

```
// Load sample image.
f, err := os.Open("templates/res/images/sample-image-2.jpg")
if err != nil {
    log.Fatal(err)
}
defer f.Close()

img, err := model.ImageHandling.Read(f)
if err != nil {
    log.Fatal(err)
}

// Create template options.
tplOpts := &creator.TemplateOptions{
    ImageMap: map[string]*model.Image{
        "sample": img,
    },
}

// Draw template.
c := creator.New()
if err := c.DrawTemplate(tpl, nil, tplOpts); err != nil {
    log.Fatal(err)
}
```

Color map

The **color map** is used to pass colors to templates. The colors are accessed in templates using their assigned map key.

TEMPLATE

```
<rectangle width="6" height="1.5" position="relative"
    fit-mode="fill-width" border-width="3"
    border-color="secondary" fill-color="primary"></rectangle>
```

RESULT

GO

```
// Create template options.
tplOpts := &creator.TemplateOptions{
    ColorMap: map[string]creator.Color{
        "primary":   creator.ColorRGBFromHex("#0772cd"),
        "secondary": creator.ColorRGBFromHex("#f00c27"),
    },
}

// Draw template.
c := creator.New()
if err := c.DrawTemplate(tpl, nil, tplOpts); err != nil {
    log.Fatal(err)
}
```

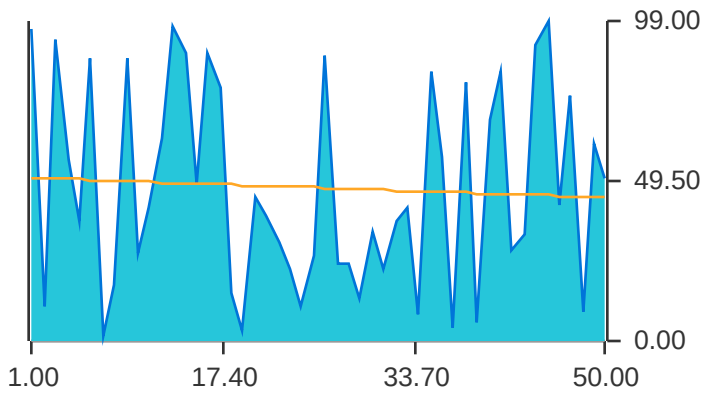
Chart map

The **chart map** is used to pass charts to templates. The charts are accessed in templates using their assigned map key.

TEMPLATE

```
<chart src="sample-chart" height="150"></chart>
```

RESULT



GO

```
// Create sample chart.
mainSeries := dataset.ContinuousSeries{
    XValues: sequence.Wrapper{Sequence: sequence.NewLinearSequence().WithStart(1.0).WithEnd(float64(50)).WithStep(1)}.Values(),
    YValues: sequence.Wrapper{Sequence: sequence.NewRandomSequence().WithLen(50).WithMin(0).WithMax(100)}.Values(),
    Style: render.Style{
        FillColor: color.RGBA{R: 38, G: 198, B: 218},
    },
}

linRegSeries := &dataset.LinearRegressionSeries{
    InnerSeries: mainSeries,
    Style: render.Style{
        StrokeColor: color.RGBA{R: 255, G: 167, B: 38},
    },
}

sampleChart := &unichart.Chart{
    Series: []dataset.Series{
        mainSeries,
        linRegSeries,
    },
}

// Create template options.
tplOpts := &creator.TemplateOptions{
    ChartMap: map[string]render.ChartRenderable{
        "sample-chart": sampleChart,
    },
}

// Draw template.
c := creator.New()
if err := c.DrawTemplate(tpl, nil, tplOpts); err != nil {
    log.Fatal(err)
}
```

Helper function map

The **helper function map** is used to pass functions to templates. The functions are accessed in templates using their assigned map key.

TEMPLATE

```
<paragraph>
    <text-chunk font="helvetica-bold" font-size="12"
        color="#1b5e20">{{strToUpper "sample text"}}</text-chunk>
</paragraph>
```

RESULT

SAMPLE TEXT

TEMPLATE

```
<paragraph>
    <text-chunk font="helvetica-bold" font-size="12"
        color="#c62828">{{now.Format "Jan 02 2006"}}</text-chunk>
</paragraph>
```

RESULT

Mar 06 2023

GO

```
// Create template options.
tplOpts := &creator.TemplateOptions{
    HelperFuncMap: template.FuncMap{
        "now": time.Now,
        "strToUpper": strings.ToUpper,
    },
}

// Draw template.
c := creator.New()
if err := c.DrawTemplate(tpl, nil, tplOpts); err != nil {
    log.Fatal(err)
}
```

For more information, please see text/template#FuncMap.

Subtemplate map

The **subtemplate map** is used to include additional templates to the processing pipeline. The added **subtemplates** can be accessed both by the main template and by the other subtemplates using their assigned map key. Subtemplates defined inside the provided subtemplates are accessible without including their parent templates. All the resources available to the main template are also available to the subtemplates.

Subtemplates are useful both for splitting long content into more manageable pieces and for defining reusable chunks of content. For example, this documentation is split into chapters using subtemplates.

For more information, please see the **text/template** package documentation regarding associated and nested templates.