第 3 章

常用 UI 组件

【学习目标】

- 理解 HarmonyOS 中的 UI 组件及组件容器
- 掌握常用的显示型组件，如 Text、Image、ProgressBar 等
- 掌握常用的交互型组件，如 TextField、Button、RadioButton、Switch 等
- 会综合使用 HarmonyOS 中常有的 UI 组件

9min

3.1 概述

UI 即 User Interface，是应用的用户界面，是用户和应用之间的交互接口。用户和应用之间的接口非常广泛，如：查看屏幕上应用的界面、单击按钮、语音输入、扫一扫、摇一摇等都是用户和应用的接口。应用 UI 组件是 HarmonyOS 为应用开发提供的界面元素，例如文本框、图像、按钮、进度条等。组件是应用界面的基本构成单元，借助组件，开发者可以高效构建自己的应用图形界面。

组件，即 Component，也可以称为控件，它是绘制在设备屏幕上的一个对象，组件是组成应用用户界面的基本单位，组件需要放置到组件容器中。组件容器，即 ComponentContainer，组件容器可以容纳组件，也可以容纳组件容器。可以说应用中绝大多数的用户界面效果是由组件容器和组件对象构成的，二者通过包含和被包含、相互配合形成丰富的用户界面，应用界面中的组件容器和组件在组织上是一个树形结构，如图 3-1 所示。

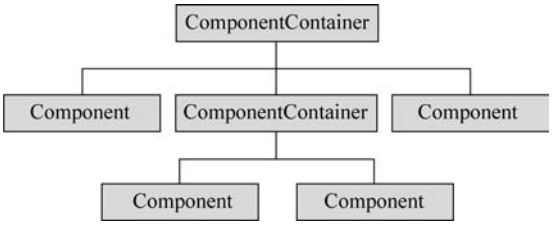

图 3-1　用户界面组件的结构

HarmonyOS 中定义了一些常用的 UI 组件类，如 Text、Button、Switch 等，这些组件类直接或间接继承了 Component 类，因此在管理上，它们都可以作为组件进行统一管理，在实现上它们又各不相同，各自实现了具体的功能。开发者可根据具体的应用场景选择合适的组件。例如，显示内容可以选择 Text、Image 等，如果需要响应用户交互动作，则可以选择 Button、Switch 等组件。

HarmonyOS 中也定义了 ComponentContainer 类，即组件容器类，该类继承了 Component 类，因此，组件容器实际上也是一种特殊的组件，它可以包含其他组件或组件容器。组件容器对应的是界面布局，常用的布局如 DirectionalLayout、DependentLayout 等，它们都继承自 ComponentContainer，关于布局后续章节会进一步阐述。

在 HarmonyOS 应用开发中，UI 实现包括 Java UI 和方舟开发两个框架，本章主要围绕 Java UI 框架阐述，Java UI 框架提供了常用组件的 Java 实现。为了阐述方便，本书中把这些组件从主要功能方面划分为显示型组件和交互型组件，前者主要用于在应用界面中展示信息数据等，如显示文本、图片等，后者多用于和用户进行交互，如按钮单击、开关操作等。下面分别详细介绍一些组件，供开发者参考，需要特别注意的是所有的 UI 组件操作默认都是在界面主线程中执行的。

3.2 显示型组件

显示型组件一般用于在应用界面中显示信息，如显示文字、展示图片等，一般不需要为组件编写事件响应代码，因此显示型组件一般通过 XML 文件定义其显示内容和效果，下面介绍几个常用的显示组件。

11min

3.2.1 Text 组件

Text 组件，即文本组件，也可称为文本或文本显示组件，在应用开发中，当需要显示文本信息时，可以考虑使用 Text 组件。

1. 使用 Text 组件

使用 Text 组件的基本方法是在需要的界面布局中加入< Text >标签。实际上，当创建一个基本的应用工程时，向导创建的默认布局文件 ability_main. xml 中就有一个 Text 组件，在 XML 布局文件中，默认生成的代码如下：

```
<DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    <Text
        ohos:id = " $ + id:text_helloworld"
        ohos:height = "match_content"
```

```
        ohos:width = "match_content"
        ohos:background_element = " $ graphic:background_ability_main"
        ohos:layout_alignment = "horizontal_center"
        ohos:text = " $ string:HelloWorld"
        ohos:text_size = "50vp"
        />
</DirectionalLayout >
```

Text 标签有若干属性，通过设置属性可以改变显示的内容和效果。所有的属性都是以 ohos:开头的，这是 HarmonyOS 规定的属性开头，其含义是 OpenHarmonyOS。

Text 标签的属性值可以是直接值，如上述示例中 Text 的高度 height、宽度 width、布局中的对齐方式 layout_alignment、文本大小 text_size 属性，它们被直接进行了赋值，其中，Text 组件的高度值和宽度值的含义是适应其内容 match_content，组件在布局中的对齐方式为水平居中 horizontal_center，文本大小为 50vp。

Text 标签的属性值也可以是间接值，如上述示例中 Text 的背景元素 background_element 和文本内容 text 属性，它们的值中包含了 $ 符号，$ 符号表示引用别的资源，如 $ graphic:background_ability_main 表示引用 graphic 下的 background_ability_main 资源，被引用的资源要求必须是存在的，因此，本例要求在资源目录 graphic 下必须有一个 background_ability_main.XML 文件资源，该资源文件的内容如下：

```
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape xmlns:ohos = "http://schemas.huawei.com/res/ohos"
       ohos:shape = "rectangle">
       < solid
         ohos:color = " # FFFFFF"/>
</shape >
```

在该资源文件中定义了一个形状 shape，该形状是一个矩形 rectangle，该矩形是实心 solid 的，颜色 color 是白色 # FFFFFF 的，因此，ohos:background_element = " $ graphic:background_ability_main"的含义是说 Text 组件的背景是一个白色矩形。

再如，ohos:text = " $ string:HelloWorld"表示的是文本组件显示的文本内容取自资源 string 中的 HelloWorld，因此，在资源目录 element 下有一个 string.json 文件，其中定义的所需的字符串资源如下：

```
{
    "string": [
        {
            "namie": "HelloWorld",
            "value": "Hello World"
        }
    ]
}
```

在该 string.json 文件中定义了一个字符串，名字为 HelloWorld，值为 Hello World，因此在文本组件中显示的内容为 Hello World。

示例中 Text 组件的 id 属性值为"$+id:text_helloworld"，其值在包含了 $ 符号的同时，还包含了 + 号，这个表示为该 Text 组件建立一个 id，此时会在资源表文件 ResourceTable.java 中自动加入一个静态整型常量 Id_text_helloworld，该常量可以供项目源代码使用，以便能够在程序中引用对应的 Text 组件。资源表文件是开发环境自动建立的帮助项目维护所建立的资源的源代码文件，一般不需要程序员修改，该文件位于项目的 build\generated\souce\r\目录下。

2. 设置 Text 属性

前面已经使用了 Text 组件的一些属性，另外 Text 组件还有一些别的属性，通过设置 Text 组件的属性可以改变它的显示效果，下面介绍几个比较常用的属性。

文本颜色 text_color、字体 text_font、是否倾斜 italic 等属性是 Text 组件属性中关于文字显示效果的，下面的 Text 属性用于将字体设置为蓝色、大小设置为 80vp、采用 HwChinese-medium 字体且倾斜效果。

```
ohos:text_color = "blue"
ohos:text_size = "80vp"
ohos:text_font = "HwChinese - medium"
ohos:italic = "true"
```

Text 组件除了可以设置显示的文本内容外，还有一些组件显示样式设置，如显示的背景、形状等。如果要将组件背景设置为图形资源，首先需要创建一幅图形资源，创建方法是在 graphic 目录下创建 XML 资源文件。具体操作是右击 resources/base/graphic，在弹出的窗口中选择 New→File，如图 3-2 所示。随后会弹出输入文件名对话框，如图 3-3 所示，在输入框中输入创建的文件名，如输入 text_element.xml，单击 OK 按钮，即可创建图形资源文件。

编辑所建立的资源文件，建立< shape >节点，在其属性中指定形状，在其子节点中通过< solid >设置填充颜色。如形状为椭圆，背景为红色的背景资源代码如下：

```
<?xml version = "1.0" encoding = "utf - 8"?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "oval">
    < solid
        ohos:color = "#FF0000"
    ></solid >
</shape >
```

创建好背景资源后，就可以在 Text 组件的属性中进行引用了，使用 Text 的背景元素属性 background_element 设置 Text 的背景代码如下：

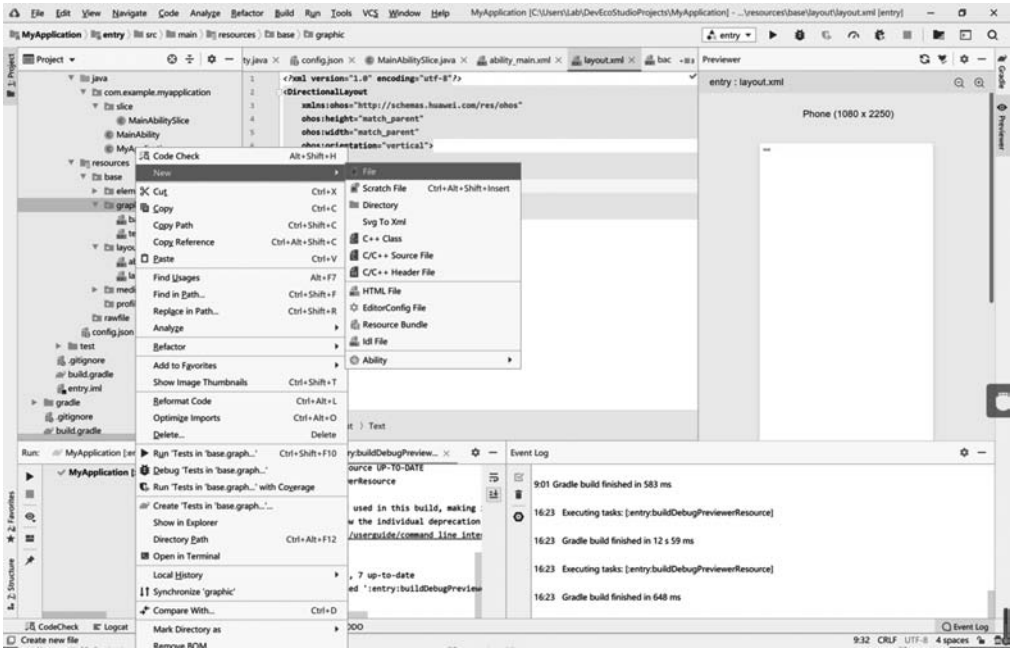

图 3-2　新建图形资源文件

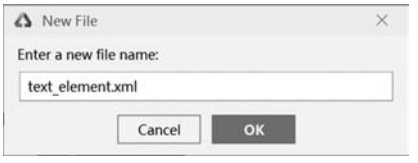

图 3-3　新建文件文本框

```
ohos:background_element = " $ graphic:text_element"
```

Text 组件还有关于边距的属性，组件中的文本内容在显示时可以设置文本距离组件边框的边距，这个边距称为内边距，内边距的相关属性为 padding，组件和外部其他组件的边距称为外边距，外边距的相关属性为 margin。内外边距既可以整体设置，也可以在上、下、左、右四方向上分别设置。

将 Text 的上外边距设置为 10vp，将下外边距设置为 10vp，将左内边距设置为 50vp，将右内边距设置为 50vp，代码如下：

```
ohos:top_margin = "10vp"
ohos:bottom_margin = "10vp"
ohos:left_padding = "50vp"
ohos:right_padding = "50vp"
```

另外，当 Text 组件的大小不足以显示其中的全部文本内容时，可以通过属性 truncation_mode 设置省略方式，该属性的值对应的含义如表 3-1 所示。

表 3-1　truncation_mode 属性值说明

truncation_mode 属性值	说　明
ellipsis_at_start	省略号在开头
ellipsis_at_middle	省略号在中间
ellipsis_at_end	省略号在末尾
auto_scrolling	自动滚动

其中，auto_scrolling 表示自动滚动，可以实现跑马灯的效果。不过要让文本真正动起来，还需要运行代码支持，在代码中首先获得 Text 对象的引用，然后通过 startAutoScrolling()方法启动自动滚动，这样就可以获得文本组件显示跑马灯的效果，核心代码如下：

```
public void onStart(Intent intent) {
    super.onStart(intent);
    //设置显示的布局
    super.setUIContent(ResourceTable.Layout_ability_main);
    //声明并绑定布局中 Text 组件 text_helloworld
    Text text = (Text)findComponentById
                        (ResourceTable.Id_text_helloworld);
    //将 text 设置为开启自动滚动
    text.startAutoScrolling();
}
```

3. Text 使用实例

实例 TextDemo 给出了 5 个不同的 Text 组件显示效果，如图 3-4 所示，它们均被放置在屏幕的水平中间，它们的内容都是“欢迎来到鸿蒙开发世界”，它们的显示样式不同。

图 3-4　Text 使用实例

实例 TextDemo 对应的布局文件 ability_main.xml 的具体的代码如下：

```
//ch03\TextDemo 项目中的 ability_main.xml
<DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    <Text  ohos:id = " $ + id:text_1"
           ohos:height = "match_content"
           ohos:width = "match_content"
           ohos:layout_alignment = "horizontal_center"
           ohos:text = "欢迎来到鸿蒙开发世界"
           ohos:text_size = "28fp"/>
    <Text
        ohos:id = " $ + id:text_2"
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:background_element = " $ graphic:background_ability_main"
        ohos:layout_alignment = "horizontal_center"
        ohos:text = "欢迎来到鸿蒙开发世界"
        ohos:text_color = "blue"
        ohos:text_font = "HwChinese - medium"
        ohos:italic = "true"
        ohos:text_size = "80"
        />
    <Text
        ohos:id = " $ + id:text_3"
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:background_element = " $ graphic:text_element"
        ohos:layout_alignment = "horizontal_center"
        ohos:text = "欢迎来到鸿蒙开发世界"
        ohos:text_color = "blue"
        ohos:text_font = "HwChinese - medium"
        ohos:italic = "true"
        ohos:text_size = "80"
        />
    <Text
        ohos:id = " $ + id:text_4"
        ohos:height = "100vp"
        ohos:width = "260vp"
        ohos:layout_alignment = "horizontal_center"
        ohos:background_element = " $ graphic:text_element"
        ohos:text = "欢迎来到鸿蒙开发世界"
        ohos:text_color = "blue"
```

```
            ohos:text_size = "80"
            ohos:top_margin = "10vp"
            ohos:left_padding = "50vp"
            ohos:truncation_mode = "ellipsis_at_end"
            />
        < Text
            ohos:id = " $ + id:text_5"
            ohos:height = "100vp"
            ohos:width = "260vp"
            ohos:layout_alignment = "horizontal_center"
            ohos:background_element = " $ graphic:text_element"
            ohos:text = "欢迎来到鸿蒙开发世界"
            ohos:text_color = "blue"
            ohos:text_size = "80"
            ohos:top_margin = "10vp"
            ohos:left_padding = "50vp"
            ohos:truncation_mode = "auto_scrolling"
            />
    </DirectionalLayout >
```

实际上,对于 Text 组件的属性,在对应的 Text 类实现中,一般提供了对应的 set()方法,可以进行属性设置,例如: Text 类提供了一个 setTruncationMode()方法,该方法可以在代码中设置 truncation_mode 的取值,设置文本自动滚动模式的代码如下:

```
//ch03\TextDemo 项目中的 MainAbilitySlice.java
public void onStart(Intent intent) {
    super.onStart(intent);
    super.setUIContent(ResourceTable.Layout_ability_main);
    Text text = (Text)findComponentById(ResourceTable.Id_text_5);
    //自动滚动
    text.setTruncationMode(Text.TruncationMode.AUTO_SCROLLING);
    text.startAutoScrolling();
}
```

通过 text.setTruncationMode(Text.TruncationMode.AUTO_SCROLLING)方法设置文本组件自动滚动模式和在 XML 文件中设置是等价的,HarmonyOS 体系 Java UI 框架提供了 XML 中的属性值和 Text 类中的常量对应关系,如表 3-2 所示。

表 3-2 truncation_mode 设置值的对应关系

XML 中的属性值	Text 类中对应的常量
auto_scrolling	Text.TruncationMode.AUTO_SCROLLING
ellipsis_at_start	Text.TruncationMode.ELLIPSIS_AT_START
ellipsis_at_middle	Text.TruncationMode.ELLIPSIS_AT_MIDDLE
ellipsis_at_end	Text.TruncationMode.ELLIPSIS_AT_END

另外，绝大多数组件的常用属性可以通过对应的组件类的 set()方法进行设置，set()方法的名称和属性的名称基本是对应的。

3.2.2　Image 组件

18min

Image 是用来显示图片的组件，在程序开发中，经常会遇到需要显示图片的情形，这时就可以考虑使用该组件了。使用 Image 需要为其准备其中显示的图片，项目中的图片资源一般放在 entry→src→main→resources→base→media 文件夹下。

这里准备一张图片，文件名为 Harmonyos.png，如图 3-5 所示，将此图片放到 media 目录下，后面将通过 $ media:HarmonyOS 引用该图片资源。

1. 使用 Image 组件

创建 Image 组件和 Text 组件的方式类似，可以在 XML 布局文件中添加 Image 节点，也可以在代码中直接创建 Image 对象。在 XML 布局文件中添加 Image 节点的参考代码如下：

```
//ch03\ImageDemo 项目中的 Ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < Image
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:image_src = " $ media:HarmonyOS"/>
</DirectionalLayout >
```

以上布局运行出的显示效果如图 3-6 所示。

图 3-5　Harmonyos.png 图片

图 3-6　使用 Image 组件

2. Image 常用属性

通过 alpha 属性可以设置 Image 的透明度，alpha 取值范围为 0～1，0 表示完全透明，1 表示完全不透明。

图片缩放也是经常使用的，Image 组件提供了 scale_x、scale_y 两个属性，可以分别设置其中的图片在 X、Y 轴方向缩放的倍数。以下代码是将图片设置为 50%透明，X 轴方向缩小到 0.5 倍，Y 轴方向放大到 2 倍的属性设置代码：

```
ohos:alpha = "0.5"
ohos:scale_y = "2"
ohos:scale_x = "0.5"
```

图片缩放默认为以 Image 组件中心为基准点的，当 Image 组件的宽度和高度与图片本身的宽度和高度比例不一致时，缩放可能会出现图片显示不正常的情况。Image 组件提供了一个 scale_mode 属性，可以让 Image 组件中的图片以一定的方式适应组件。

例如，当将 Image 的宽度和高度值都设置为 200vp 时，按比例放大或缩小并居中显示图片的代码如下：

```
< Image
    ohos:height = "200vp"
    ohos:width = "200vp"
    ohos:background_element = " # FF0000"
    ohos:image_src = " $ media:HarmonyOS"
    ohos:scale_mode = "zoom_center"
    />
```

运行效果如图 3-7 所示，背景展示了 Image 组件的大小，图片以中心为原点进行了自动放大，当宽度和组件大小一样后，适配了组件大小，上下留了一些组件背景。

图 3-7　按照 zoom_center 缩放并居中显示

当 scale_mode 取不同的值时，对应的图片放大或缩小适配组件的方式如表 3-3 所示。

表 3-3 Image 支持的缩放方式

scale_mode 值	缩 放 方 式
zoom_center	按比例将原图扩大或缩小到 Image 的宽或高，居中显示，在宽或高有一个方向上撑满组件。此情况下，图片显示是完整的，组件背景不一定会被完全覆盖
zoom_start	宽和高按比例将原图扩大或缩小到 Image 组件的边界，和居中显示不同的是，此时图片放置在 Image 组件的左上角位置
zoom_end	缩放同上，不同的是图片放置在 Image 组件的右下角位置
stretch	图片拉伸或挤压到 Image 组件的大小，图片撑满整个组件，不一定保持原来的宽高比
center	保持原图片的大小，图片显示在 Image 组件的中心。当图片尺寸小于组件大小时，按图片大小显示；当图片的尺寸大于 Image 组件的尺寸时，对超过部分进行裁剪处理
inside	图片放置到 Image 组件内，当图片大于组件大小时，按比例将原图缩小到 Image 的宽度或高度，将图片的内容完整地居中显示；当图片小于组件大小时，直接居中显示图片
clip_center	按比例将原图扩大或缩小到 Image 组件大小，图片恰好可以覆盖满组件，对多余的部分进行裁剪。此情况下，组件背景会被完全覆盖，图片不一定显示完整

除了缩放，图片裁剪也是比较常用的功能，一般情况下，当所展示的图片大于所在的 Image 组件大小时，就会进行裁剪。裁剪可以选择不同的对齐方式，关于裁剪的对齐方式，读者可以认为 Image 组件就是一个贴板，其中的图片可以认为是贴纸，当贴纸大于贴板时，会进行裁剪，裁剪前需要把贴纸放到贴板的指定位置，也就是和贴板的哪个边或角对齐的问题。Image 组件提供的 clip_alignment 属性可以设置裁剪方式，以下代码是将裁剪对齐方式设置为左侧对齐：

```
ohos:clip_alignment = "left"
```

默认情况下，裁剪的对齐方式是居中的，因此以上代码的功能是将裁剪对齐设置为水平方向左侧对齐、垂直方向居中对齐。clip_alignment 属性的取值和裁剪对齐方式如表 3-4 所示。

表 3-4 Image 裁剪方式

clip_alignment 值	裁剪方式说明
left	图片左侧和 Image 组件对齐，裁剪右侧
right	图片右侧和 Image 组件对齐，裁剪左侧
top	图片上方和 Image 组件对齐，裁剪下侧
bottom	图片底部和 Image 组件对齐，裁剪上侧
center	图片中心和 Image 组件对齐，裁剪四周

在进行裁剪对齐方式设置时，可以多个值同时使用，如设置右侧下侧对齐方式的代码如下：

```
ohos:clip_alignment = "right|bottom"
```

此代码采用的是一种或运算，但实际上表示的是而且关系，也就是同时具有两个值的效果。


8min

3.2.3 DatePicker 组件

DatePicker 是一个日期组件，应用中有时需要选择填写日期，DatePicker 组件是专门用来显示选择日期的。

使用 DatePicker 组件可以在布局中加入< DatePicker >节点，设置宽和高等属性即可创建 DatePicker 组件，下面是使用该组件的示例代码：

```
//ch03\DatePicker 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < DatePicker
        ohos:id = " $ + id:data_picker"
        ohos:height = "match_content"
        ohos:width = "match_parent"
        ohos:date_order = "year - month - day"
        ohos:text_size = "30vp"
        ohos:selected_text_size = "36vp"
        ohos:normal_text_color = "blue"
        ohos:selected_text_color = "red"
        ohos:background_element = " # CCCCCC"
        />
</DirectionalLayout >
```

以上代码的运行效果如图 3-8 所示，默认显示选中的日期是当天。

图 3-8 DatePicker 显示效果

DatePicker组件常用的属性说明如表3-5所示，开发者可以根据需要进行相应的属性设置，以便满足应用使用日期组件的具体需求。

表3-5 DatePicker的常用属性及说明

属性名称	描述	取值及说明	使用举例
date_order	设置显示格式，值实际为0～10枚举值，不同的值组件显示的日期格式不同	day-month-year 显示为日月年 month-day-year 显示为月日年 year-month-day 显示为年月日 year-day-month 显示为年日月 day-month 显示为日月 month-day 显示为月日 year-month 显示为年月 month-year 显示为月年 only-year 仅显示年 only-month 仅显示月 only-day 仅显示日	ohos:date_order="year-month-day"
year_fixed	是否固定年份，如果固定，则年份不能改变	true 或 false true 表示固定，false 表示不固定，默认值为 false	ohos:year_fixed="true" ohos:year_fixed="$boolean:true"
month_fixed	是否固定月份	同上	ohos:month_fixed="true" ohos:month_fixed="$boolean:true"
day_fixed	是否固定日期	同上	ohos:day_fixed="true" ohos:day_fixed="$boolean:true"
text_size	默认文本大小	取值的默认单位为 px，可以是带 px/vp/fp 单位的浮点数值，可以是浮点数值，也可以引用 float 资源	ohos:text_size="30" ohos:text_size="16fp" ohos:text_size="$float:size_value"
normal_text_size	未选中文本的大小	同上	ohos:normal_text_size="30" ohos:normal_text_size="16fp" ohos:normal_text_size="$float:size_value"
selected_text_size	选中的文本的大小	同上	ohos:selected_text_size="30" ohos:selected_text_size="16fp" ohos:selected_text_size="$float:size_value"
normal_text_color	未选中文本的颜色	可以直接设置色值，可以使用常用的颜色名称，也可以引用颜色资源	ohos:normal_text_color="#0000FF" ohos:normal_text_color="#blue" ohos:normal_text_color="$color:blue"

续表

属性名称	描述	取值及说明	使用举例
selected_text_color	选中文本的颜色	同上	ohos:selected_text_color="#FF0000" ohos:selected_text_color="red" ohos:selected_text_color="$color:red"
operated_text_color	操作时文本显示的颜色，当操作选择年、月或日时，文本显示的颜色	同上	ohos:operated_text_color="#00FF00" ohos:operated_text_color="green" ohos:operated_text_color="$color:green"
selector_item_num	组件界面上显示供选择的项数	取值为一个整数，如5表示5个日期显示在界面上	ohos:selector_item_num="5" ohos:selector_item_num="$integer:num"
wheel_mode_enabled	是否为循环模式，如选择月份，向上滑动到12后，是否出现1月	true表示循环模式，false表示不循环	ohos:wheel_mode_enabled="true" ohos:wheel_mode_enabled="$boolean:true"

11min

3.2.4 TimePicker组件

TimePicker是用来选择时间的组件，和DatePicker显示选择的年、月、日不同，TimePicker显示选择的是时、分、秒，当应用中需要显示时间和选择功能时，就可以考虑使用TimePicker组件。

TimePicker的用法与DatePicker比较类似，在需要显示时间的布局中添加<TimePicker>组件标签即可。下面是TimePicker使用的一段示例代码：

```
//ch03\TimePickerShow项目中的ability_main.xml
<?xml version="1.0" encoding="utf-8"?>
<DirectionalLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_parent"
    ohos:width="match_parent"
    ohos:orientation="vertical">
<TimePicker
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:mode_24_hour="false"
        ohos:am_pm_order="left"
        ohos:hour="22"
        ohos:minute="30"
        ohos:second="29"
        ohos:text_pm="下午"
```

```
            ohos:normal_text_size = "30fp"
            ohos:selected_text_size = "50fp"
            ohos:selected_text_color = "#0000FF"
            ohos:shader_color = "#DCFFC2F7"
            ohos:bottom_line_element = "#FF7700FF"/>
    </DirectionalLayout >
```

该例子中设置了 TimePicker 的几个属性，包括高度(height)、宽度(width)、是否为 24h 制(mode_24_hour)、上下午位置(am_pm_order)、默认选中的时分秒(hour、minute、second)、字体大小(normal_text_size)、选中文本大小(normal_text_size)、选中文本颜色(selected_text_color)、组件背景色(shader_color)、下画线(bottom_line_element)等，运行的效果如图 3-9 所示。

图 3-9　TimePicker 示例

TimePicker 组件的常用属性和 DatePicker 的常用属性比较类似，具体如表 3-6 所示，开发者可以根据需要进行相应的属性设置，以便满足具体应用的需求。

表 3-6　TimePicker 组件的常用属性

属性名称	描　述	取值及说明	使用案例
mode_24_hour	设置是否以 24h 制显示时间，不设置时该属性默认为 24h 制	true 或 false，true 表示是，false 表示否	ohos:mode_24_hour="true" ohos:mode_24_hour="$boolean:true"
am_pm_order	上午和下午的位置，当不采用 24h 制时，该属性才生效	left 表示上/下午在组件左侧； right 表示上/下午在组件右侧	ohos:am_pm_order="2"
text_am	上午显示的文本内容	字符串，或直接设置文本字符串，或引用 string 资源	ohos:text_am="上午" ohos:text_am="$string:am"
text_pm	下午显示的文本内容	同上	ohos:text_pm="下午" ohos:text_pm="$string:pm"
hour	默认选中显示的小时值	取值为 0～23，可直接设置整型数值，也可以引用 integer 资源	ohos:hour="22" ohos:hour="$integer:hour"
minute	默认选中显示的分钟值	取值为 0～59，可直接设置整型数值，也可引用 integer 资源	ohos:minute="30" ohos:minute="$integer:minute"

续表

属性名称	描　述	取值及说明	使用案例
second	默认选中显示的秒值	同上	ohos:second="29" ohos:second="$integer:second"
normal_text_size	未选中时间文本的大小	默认单位为 px,可为带 px/vp/fp 单位的浮点值,也可引用 float 资源	ohos:normal_text_size="30" ohos:normal_text_size="16fp" ohos:normal_text_size="$float:size_value"
selected_text_size	选中的时间文本的大小	同上	ohos:selected_text_size="30" ohos:selected_text_size="16fp" ohos:selected_text_size="$float:size_value"
normal_text_color	未选中时间文本的颜色	可直接设置颜色值,可使用常用的颜色名称,也可以引用颜色资源	ohos:normal_text_color="#0000FF" ohos:normal_text_color="#blue" ohos:normal_text_color="$color:blue"
selected_text_color	选中文本时间的颜色	同上	ohos:selected_text_color="#FF0000" ohos:selected_text_color="red" ohos:selected_text_color="$color:red"
operated_text_color	操作时时间文本的颜色,当操作选择年或月或日时,文本显示的颜色	同上	ohos:operated_text_color="#00FF00" ohos:operated_text_color="green" ohos:operated_text_color="$color:green"
shader_color	组件背景颜色	同上	ohos:shader_color="#DCFFC2F7" ohos:shader_color="black" ohos:shader_color="$color:black"
top_line_element	选中时间的上线显示效果	可直接配置色值,可引用 color 资源,也可引用 media/graphic 中的图资源	ohos:top_line_element="#FF7700FF" ohos:top_line_element="red" ohos:top_line_element="$color:red" ohos:top_line_element="$media:media_src" ohos:top_line_element="$graphic:graphic_src"
bottom_line_element	选中时间的下线显示效果	同上	ohos:bottom_line_element="#FF7700FF" ohos:bottom_line_element="$color:red" ohos:bottom_line_element="$media:media_src" ohos:bottom_line_element="$graphic:graphic_src"

续表

属性名称	描　述	取值及说明	使用案例
selector_item_num	组件界面上显示供选择的项数	整数，如 5 表示 5 个时间显示在界面上	ohos:selector_item_num="5" ohos:selector_item_num=" $integer:num"
wheel_mode_enabled	是否为循环模式，如选择小时，向上滑动到 23 时，是否出现 0 月	true 表示循环模式，false 表示不循环，默认模式为不循环	ohos:wheel_mode_enabled="true" ohos:wheel_mode_enabled=" $boolean:true"

3.2.5 ProgressBar 组件

ProgressBar 为进度条组件，在开发过程中，经常会遇到进度的显示，如下载文件进度、资源加载进度等，ProgressBar 组件可以在应用界面中显示某种进度。

ProgressBar 进度条对应的标签为< ProgressBar >，使用方法和其他组件类似，在需要的布局中添加< ProgressBar >节点，配置需要的属性，下面是一个配置进度条的示例代码：

```
//ch03\ProgressBar 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < ProgressBar
        ohos:height = "60vp"
        ohos:width = "300vp"
        ohos:progress_width = "10vp"
        ohos:orientation = "horizontal"
        ohos:background_element = "＃888888"
        ohos:background_instruct_element = "＃FF0000"
        ohos:progress_color = "＃00FF00"
        ohos:min = "0"
        ohos:max = "100"
        ohos:progress = "30"
        ohos:progress_hint_text = "30％"
        ohos:layout_alignment = "center"/>
</DirectionalLayout >
```

以上代码运行的效果如图 3-10 所示，该示例展示了 ProgressBar 常用的一些属性效果。

ProgressBar 常用属性如表 3-7 所示，开发者可以根据需要选择合适的属性进行设置。

图 3-10　ProgressBar 效果图

表 3-7　ProgressBar 组件的常用属性

属性名称	描　　述	取值及说明	使用举例
height	组件的高度	单位为 px 的整数值，或单位为 px/vp/fp 的浮点数值，也可以引用资源	ohos:height="20" ohos:height="60vp" ohos:height="$float:size_value"
width	组件的宽度	同上	ohos:width="100" ohos:width="300vp" ohos:width="$float:size_value"
progress_width	进度条的宽度	同上	ohos:progress_width="10vp"
progress_color	组件中完成进度部分的颜色	可直接设置颜色值，也可引用 color 资源	ohos:progress_color="#00FF00" ohos:progress_color="$color:green"
progress_element	组件中完成进度部分的元素	或为配置颜色值，或引用 color 资源，或引用 media/graphic 下的图片资源	ohos:progress_element="#00FF00" ohos:progress_element="$color:green" ohos:progress_element="$media:media_src" ohos:progress_element="$graphic:graphic_src"
background_instruct_element	组件中进度部分的元素	同上	类似 progress_element 取值
background_element	整个组件的背景元素	同上	类似 progress_element 取值
progress_hint_text	进度条上提示的文本信息	字符串，可以设置文本字串，也可以引用 string 资源	ohos:progress_hint_text="30%" ohos:progress_hint_text="$string:test_str"

续表

属性名称	描　述	取值及说明	使用举例
progress_hint_text_color	进度条上提示文本信息的颜色	可以直接设置颜色值，也可以引用color资源	ohos: progress_hint_text_color = "#000000" ohos: progress_hint_text_color = "$color:black"
progress_hint_text_size	进度条上提示文本信息的大小	float类型，其默认单位为px；可以是带px/vp/fp单位的浮点数值；也可以引用float资源	ohos:progress_hint_text_size="100" ohos:progress_hint_text_size="20fp" ohos:progress_hint_text_size="$float:size_value"
progress_hint_text_alignment	进度条上提示文本信息的对齐方式	left表示左对齐，right表示右对齐，top表示顶部对齐，bottom表示底部对齐，horizontal_center表示水平居中，vertical_center表示垂直居中，center表示居中	可以设置单个值，也可以使用“|”进行多值组合。 ohos: progress_hint_text_alignment = "top" ohos: progress_hint_text_alignment = "top|left"
max	最大值，进度条进度结束时的值	整数，可以直接设置整型数值，也可以引用integer资源	ohos:max="100" ohos:max="$integer:i100"
min	最小值，进度条进度开始时的值	同上	ohos:min="0" ohos:min="$integer:i0"
progress	进度条当前的进度值	整数，介于最大值和最小值之间	ohos:progress="30" ohos:progress="$integer:i30"
step	进度的步长	整数，默认值为1。若将step设置为10，进度值则为10的倍数	ohos:step="10" ohos:step="$integer:ten"
orientation	组件的放置方向	水平或垂直，horizontal表示水平，vertical表示垂直，默认为水平放置	ohos:orientation="horizontal" ohos:orientation="vertical"

续表

属性名称	描　述	取值及说明	使用举例
divider_lines_enabled	进度条进度是否分段显示	true 表示分段显示，false 表示不分段，默认值是 false	ohos:divider_lines_enabled="true" ohos:divider_lines_enabled="$boolean:true"
divider_lines_number	进度条进度分段显示时的分段数	整数，介于最大值和最小值之间	ohos:progress="5" ohos:progress="$integer:i5"

3.3 交互型组件

交互型组件一般在用于界面显示信息的同时还需要用户与之进行交互，如输入文字、单击、滑动等。交互型组件的外观一般通过属性进行定义，这一点和前面介绍的显示型组件类似，组件交互过程一般需要编写事件响应代码，关于响应代码将在事件处理部分详细介绍。下面主要介绍几个常用的交互型组件的基本用法。

12min

3.3.1 TextField 组件

TextField 组件，即文本框组件，也可称为文本域或输入框组件。文本框主要用于用户输入文本内容，如输入用户名、密码等。在 Java UI 框架中的 TextField 类继承了 Text 类，因此 Textfield 组件拥有 Text 组件所拥有的属性，文本框特有的主要功能是文本的输入。TextField 类似于 Android 中的 EditText 或 iOS 中的 UITextField，下面具体介绍一下 TextField 组件。

1. 使用 TextField 组件

使用文本框组件可以在相应布局文件中添加< TextFiled >节点，通过设置属性修饰组件，如组件大小、背景颜色、提示信息等。下面是一个通过 XML 布局文件使用文本框的例子代码：

```
//ch03\TextField项目中的ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < TextField
        ohos:height = "40vp"
        ohos:width = "260vp"
        ohos:left_padding = "10vp"
```

```
        ohos:text_alignment = "center"
        ohos:layout_alignment = "center"
        ohos:hint = "输入电话号码"
        ohos:text_size = "30fp"
        ohos:background_element = " $ graphic:background_textfield"
        />
</DirectionalLayout >
```

以上代码在设置文本框背景元素(background_element)时引用了图形资源,所引用的资源文件 background_textfield.xml 的代码如下:

```
//ch03\TextField 项目中的 background_textfield.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "rectangle">
    < corners
        ohos:radius = "40" />
    < solid
        ohos:color = " # 999999" />
</shape >
```

以上示例代码的运行效果如图 3-11、图 3-12 所示,单击文本输入框可以输入文本信息,输入信息时提示的信息"输入电话号码"会自动消失,同时自动弹出输入虚拟键盘。

图 3-11　TextField 运行效果

图 3-12　文本框输入时的效果

2. TextField 常用属性

对于 TextField 组件,Text 组件所拥有的属性 TextField 都有,因此一些基本的属性这里不再赘述。文本框主要用于输入文本,当光标点移入文本框时,界面上会出现一个气泡,

如果开发者不想使用默认的气泡样式，可以通过 element_cursor_bubble 属性进行重新设置，该属性可以采用一幅图形资源作为值，也可以自由定义所需要的样式，如下代码为设置气泡的例子：

```
…
    <TextField
        …
        ohos:element_cursor_bubble = "$graphic:cursor_bubble_textfield"
        />
…
```

其中，引用了图形资源 cursor_bubble_textfield，该资源对应的资源文件 cursor_bubble_textfield.xml 的代码如下：

```
//ch03\TextField 项目中的 cursor_bubble_textfield.xml
<?xml version = "1.0" encoding = "utf - 8"?>
<shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "oval">
    <solid
        ohos:color = "#0000FF"/>
    <stroke
        ohos:width = "5vp"
        ohos:color = "#FF0000"/>
</shape>
```

图 3-13　修改气泡后的效果图

这里定义的气泡形状是一个圆形 oval，填充颜色为蓝色 #0000FF，边线宽度是 5 个虚拟像素 5vp，颜色是红色 #FF0000，运行效果如图 3-13 所示。

multiple_lines 属性可以设定是否多行显示，若该属性值为 true 则表示多行，若该属性值为 false，则表示单行显示。多行显示效果与 TextFiled 组件的高度有关系，当高度值不够大时可能无法显示多行效果。

basement 属性可以为文本框设置一个基线，也就是可以在文本框下面加一条带颜色的线，该属性的值为基线的颜色。

在实际应用中，我们还会经常碰到输入密码、输入手机号等具有一定限制的文本框输入。如输入密码时往往不希望以明文的形式显示，而是希望以掩码的方式显示，即用符号“*”遮挡所输入的密码内容。在输入手机号时，希望直接弹出的是数字键盘，而非字母键盘，免得进行键盘切换。为了提供更好的输入体验，文本框组件提供了 text_input_type 属

性，该属性值及说明如表 3-8 所示。

表 3-8　文本输入类型的取值及说明

text_input_type 属性值	说　明
ohos:text_input_type="pattern_null"	无约束限制
ohos:text_input_type="pattern_text"	普通文本
ohos:text_input_type="pattern_number"	数字模式，显示数字键盘
ohos:text_input_type="pattern_password"	密码模式，输入内容以 * 显示

3.3.2　Button 组件


7min

Button，即按钮，是最常用的组件之一。按钮可由文本组成，也可由图标组成，还可由图标和文本共同组成。

1. 使用 Button 组件

使用 Button 组件时可以在布局中添加< Button >节点，也可以在代码中实例化 Button 按钮。下面以在布局文件中添加 Button 节点为例，来说明按钮组件的使用。

如在所建的项目 src→main→resources→layout 布局目录下，在布局文件 ability_main.xml 中添加按钮组件的示例代码如下：

```
//ch03\Button 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < Button
        ohos:id = " $ + id:button01"
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:layout_alignment = "center"
        ohos:text_size = "50fp"
        ohos:text = "按钮 01"
        ohos:element_left = " $ media:icon"
        ohos:background_element = " $ graphic:background_button01"
        />
</DirectionalLayout >
```

上面代码在按钮的左侧加入了一张图标，引用的是 $ media:icon 图标资源，这张图片资源是创建项目时自带的一个默认图标，用户可以根据需要添加并使用新的图片资源。

按钮的形状也是通过引用的资源定义的，background_element 属性引用的 $ graphic:background_button01 资源在 graphic 目录下，对应的 XML 文件名为 background_

button01.xml,其中定义了一个基本的矩形形状,其代码如下:

```
//ch03\Button 项目中的 background_button01.xml
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape xmlns:ohos = "http://schemas.huawei.com/res.ohos"
        ohos:shape = "rectangle">
        < solid
          ohos:color = "#FF007DFF"/>
</shape >
```

以上示例代码运行的效果如图 3-14 所示。

2. 多种形状 Button

按钮可以根据需要制作成不同的形状,如矩形按钮、胶囊按钮、椭圆按钮和圆形按钮等以适用于不同的应用场景,其效果如图 3-15 所示。

图 3-14 Button 运行效果图

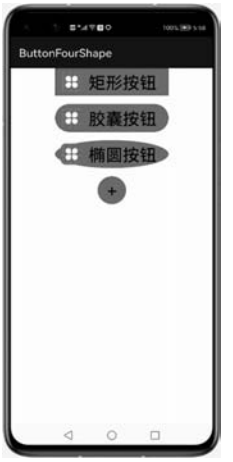

图 3-15 不同形状的按钮

默认情况下使用的按钮为矩形按钮,可以改变按钮中显示的文本和背景的颜色,当然也可以通过背景元素 ohos:background_element 属性设置指定的形状。

胶囊按钮可以通过将背景元素设置为圆角矩形形状实现,椭圆按钮都可以通过设置椭圆形状实现,当椭圆按钮的高和宽一样时,便成为圆形按钮。

图 3-15 显示的是不同形状的按钮,对应的实现代码如下:

```
//ch03\ButtonFourShape 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
```

```
    ohos:orientation = "vertical">
    < Button
        ohos:id = " $ _ + id:button01"
        ohos:height = "50vp"
        ohos:width = "200vp"
        ohos:background_element = " $ graphic:element_button_rectangle"
        ohos:element_left = " $ media:icon"
        ohos:layout_alignment = "center"
        ohos:bottom_margin = "15vp"
        ohos:left_padding = "10vp"
        ohos:right_padding = "10vp"
        ohos:text = "矩形按钮"
        ohos:text_size = "30fp"
        />
    < Button
        ohos:id = " $ _ + id:button02"
        ohos:height = "50vp"
        ohos:width = "200vp"
        ohos:background_element = " $ graphic:element_button_capsule"
        ohos:element_left = " $ media:icon"
        ohos:layout_alignment = "center"
        ohos:bottom_margin = "15vp"
        ohos:left_padding = "10vp"
        ohos:right_padding = "10vp"
        ohos:text = "胶囊按钮"
        ohos:text_size = "30fp"/>
    < Button
        ohos:id = " $ _ + id:button03"
        ohos:height = "50vp"
        ohos:width = "200vp"
        ohos:background_element = " $ graphic:element_button_oval"
        ohos:element_left = " $ media:icon"
        ohos:layout_alignment = "center"
        ohos:bottom_margin = "15vp"
        ohos:left_padding = "10vp"
        ohos:right_padding = "10vp"
        ohos:text = "椭圆按钮"
        ohos:text_size = "30fp"
        />
    < Button
        ohos:id = " $ _ + id:button04"
        ohos:height = "50vp"
        ohos:width = "50vp"
        ohos:background_element = " $ graphic:element_button_oval"
        ohos:layout_alignment = "center"
```

```
        ohos:bottom_margin = "15vp"
        ohos:left_padding = "10vp"
        ohos:right_padding = "10vp"
        ohos:text = " + "
        ohos:text_size = "30fp"
        />
</DirectionalLayout >
```

其中,矩形按钮对应的资源文件 element_button_rectangle. xml 的代码如下:

```
//ch03\ButtonFourShape 项目中的 element_button_rectangle.xml
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res.ohos"
    ohos:shape = "rectangle">
    < solid
        ohos:color = "#007CFD"/>
</shape >
```

在胶囊按钮对应的资源文件中,通过< corners >标签将圆角的半径设置为 100,资源文件 element_button_capsule. xml 的代码如下:

```
//ch03\ButtonFourShape 项目中的 element_button_capsule.xml
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res.ohos"
    ohos:shape = "rectangle">
    < corners
        ohos:radius = "100"/>
    < solid
        ohos:color = "#007CFD"/>
</shape >
```

椭圆和圆按钮对应的资源文件 element_button_oval. xml 的代码如下:

```
//ch03\ButtonFourShape 项目中的 element_button_oval.xml
<?xml version = "1.0" encoding = "utf - 8" ?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "oval">
    < solid
        ohos:color = "#007CFD">
    </solid >
</shape >
```

3.3.3　Checkbox 组件

图 3-16　Checkbox 使用示例

6min

Checkbox，即复选框，或称为多选按钮。在一些应用中会碰到多选的情况，如考试系统中的多选题、调查问卷中的爱好选择统计等。Checkbox 是专门为多选提供的组件。

下面通过一个调查兴趣爱好的例子，说明一下复选框的基本用法。该示例运行的效果如图 3-16 所示，可以通过单击选择自己的爱好。

实现该效果，需要在布局文件中添加 4 个< Checkbox >节点，分别代表 4 个爱好选项，设置组件的宽、高等基本属性，这些和按钮、文本等组件类似，Checkbox 提供了两个特有的属性，一个是 ohos:text_color_on，另一个是 ohos:text_color_off，它们可以分别表示当选中和未选中时组件中显示的内容信息，下面是兴趣爱好示例的具体布局代码：

```
//ch03\CheckboxDemo 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < DirectionalLayout
        ohos:height = "match_content"
        ohos:width = "match_parent"
        ohos:orientation = "horizontal">
        < Text
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:text = "你的兴趣爱好有哪些?"
            ohos:text_size = "25fp"/>
        < Text
            ohos:id = " $ + id:text_answer"
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:text = "()"
            ohos:text_size = "25fp"/>
    </DirectionalLayout >
    < Checkbox
        ohos:id = " $ + id:check_box_1"
        ohos:height = "match_content"
        ohos:width = "match_content"
```

```
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "A 读书"
            ohos:text_color_off = "#000000"
            ohos:text_color_on = "#FF3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"/>
        <Checkbox
            ohos:id = "$ + id:check_box_2"
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "B 足球"
            ohos:text_color_off = "#000000"
            ohos:text_color_on = "#FF3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"/>
        <Checkbox
            ohos:id = "$ + id:check_box_3"
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "C 跑步"
            ohos:text_color_off = "#000000"
            ohos:text_color_on = "#FF3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"/>
        <Checkbox
            ohos:id = "$ + id:check_box_4"
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "D 旅游"
            ohos:text_color_off = "black"
            ohos:text_color_on = "#FF3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"/>
    </DirectionalLayout>
```

3.3.4 RadioButton/RadioContainer

RadioButton,即单选按钮,或称为单选框。和复选框不同的是单选框一般应用于多选

一的情况，如性别选择、回答是与否的选择等。对于单选框需要在一个组内实现互斥，为了更好地实现这一点，在使用单选框（RadioButton）时，一般还会使用单选框容器（RadioContainer），位于同一个单选框容器中的单选框自动实现互斥，即同一时刻只能有一个会被选中。

下面是一个通过单选按钮选择性别的示例，该示例的运行效果如图 3-17 所示。

图 3-17　单选按钮示例

该示例对应的布局代码如下：

```
//ch03\RadioButtonDemo 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "vertical">
    < Text
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:text = "请你选择性别"
        ohos:text_size = "25fp"/>
    < RadioContainer
        ohos:id = " $ + id:radio_container"
        ohos:height = "match_content"
        ohos:width = "match_parent"
        ohos:orientation = "vertical">
        < RadioButton
            ohos:id = " $ + id:r_btn_male"
            ohos:height = "match_content"
```

```
            ohos:width = "match_content"
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "男"
            ohos:text_color_off = "#000000"
            ohos:text_color_on = "#ff3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"
            />
        <RadioButton
            ohos:id = "$ + id:r_btn_female"
            ohos:height = "match_content"
            ohos:width = "match_content"
            ohos:background_element = "#CCCCCC"
            ohos:left_margin = "10vp"
            ohos:text = "女"
            ohos:text_color_off = "#000000"
            ohos:text_color_on = "#ff3333"
            ohos:text_size = "25fp"
            ohos:top_margin = "5vp"
            />
    </RadioContainer>
</DirectionalLayout>
```

该示例布局中有两个单选按钮，一个文本显示的内容是“男”，另一个文本显示的内容是“女”，它们位于同一个单选按钮容器中，因此当选择其中一个选项时，另外一个选项会自动取消。

3.3.5 Switch 组件

Switch，即开关组件，在手机设置界面，经常会看到开关按钮，应用开发中使用 Switch 组件可以实现开关功能。

如下是一个模仿常用的设置中打开和关闭个人热点的界面，布局代码如下：

```
//ch03\SwitchDemo 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
<DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:orientation = "horizontal">
    <Text
        ohos:height = "match_content"
        ohos:width = "match_content"
```

```
            ohos:text = "个人热点"
            ohos:top_margin = "15vp"
            ohos:left_margin = "10vp"
            ohos:text_size = "30vp"/>
        < Switch
            ohos:height = "30vp"
            ohos:width = "60vp"
            ohos:top_margin = "10vp"
            ohos:left_margin = "160vp"
            ohos:text_state_off = "关"
            ohos:text_state_on = "开"
            ohos:text_size = "20vp"
            />
    </DirectionalLayout >
```

组件中的 text_state_off 和 text_state_on 属性分别用来设置关闭和打开状态时显示的文本，运行效果如图 3-18(a)和图 3-18(b)所示。

图 3-18　Switch 开关效果图

当然，Switch 开关组件可以通过设置更多属性来修饰其显示效果。

3.4　组件应用示例

14min

本节以一般的应用登录界面为例，综合应用前面所学的组件，本例运行后显示的效果如图 3-19 所示。登录界面的最上方有一张图片 logo，其下是输入用户名和密码框，再往下是记住密码选项，然后是登录和注册功能，最下方是忘记密码功能。

该示例中，各个功能与所使用的组件的对应关系如表 3-9 所示。

图 3-19　登录界面示例效果

表 3-9　文本输入类型的取值及说明

功　　能	使用组件及说明
最上方的图片 logo	Image 组件
请输入用户名	TextField 组件，通过 basement 实现底边线显示效果
请输入密码	TextField 组件，通过 basement 实现底边线显示效果
记住密码	CheckBox 组件
登录	Button 组件，通过背景图形实现圆角效果
注册	Button 组件，通过背景图形实现圆角效果
忘记密码	Text 组件
忘记密码下方线	Text 组件，通过背景线及组件高度控制，显示线效果

该示例的布局实现代码如下：

```
//ch03\LoginUIDemo 项目中的 ability_main.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< DirectionalLayout
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:height = "match_parent"
    ohos:width = "match_parent"
    ohos:alignment = "center"
    ohos:orientation = "vertical">

    < Image
        ohos:height = "100vp"
        ohos:width = "100vp"
        ohos:foreground_element = " $ media:icon"
        ohos:margin = "20vp"
```

```
        />

    <TextField
        ohos:height = "match_content"
        ohos:width = "200vp"
        ohos:basement = " $ graphic:bg_line"
        ohos:hint = " $ string:inputName"
        ohos:margin = "10vp"
        ohos:text_alignment = "center"
        ohos:text_size = "27vp"
        />

    <TextField
        ohos:height = "match_content"
        ohos:width = "200vp"
        ohos:basement = " $ graphic:bg_line"
        ohos:hint = " $ string:inputPassword"
        ohos:margin = "10vp"
        ohos:text_alignment = "center"
        ohos:text_input_type = "pattern_password"
        ohos:text_size = "27vp"
        />

    <Checkbox
        ohos:height = "match_content"
        ohos:width = "150vp"
        ohos:bubble_height = "50vp"
        ohos:check_element = " $ graphic:checkbox"
        ohos:margin = "10vp"
        ohos:text = " $ string:checkPassword"
        ohos:text_size = "22vp"
        />

    <DirectionalLayout
        ohos:height = "match_content"
        ohos:width = "match_parent"
        ohos:alignment = "center"
        ohos:orientation = "horizontal">

        <Button
            ohos:height = "match_content"
            ohos:width = "120vp"
            ohos:background_element = " $ graphic:bg_button"
            ohos:margin = "10vp"
            ohos:padding = "3vp"
```

```
            ohos:text = " $ string:login"
            ohos:text_size = "28vp"
            />

        < Button
            ohos:height = "match_content"
            ohos:width = "120vp"
            ohos:background_element = " $ graphic:bg_button"
            ohos:margin = "10vp"
            ohos:padding = "3vp"
            ohos:text = " $ string:register"
            ohos:text_size = "28vp"
            />
    </DirectionalLayout >

    < Text
        ohos:height = "match_content"
        ohos:width = "match_content"
        ohos:layout_alignment = "horizontal_center"
        ohos:text = " $ string:forget"
        ohos:text_color = " # 0000FF"
        ohos:text_size = "20vp"
        ohos:top_margin = "50vp"
        />

    < Text
        ohos:height = "1vp"
        ohos:width = "100vp"
        ohos:background_element = " $ graphic:bg_line"
        ohos:bottom_margin = "50vp"
        ohos:layout_alignment = "horizontal_center"
        />
</DirectionalLayout >
```

布局中引用的资源文件 bg_line.xml 的代码如下：

```
//ch03\LoginUIDemo 项目中的 bg_line.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "line"
>
    < stroke
        ohos:width = "2vp"
        ohos:color = " # 0000FF"
        />
    </shape >
```

布局中引用的资源文件 checkbox.xml 的代码如下：

```
//ch03\LoginUIDemo 项目中的 checkbox.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< state - container xmlns:ohos = "http://schemas.huawei.com/res/ohos">
    < item ohos:state = "component_state_checked"
          ohos:element = " $ graphic:checkbox_checked"/>
    < item ohos:state = "component_state_empty"
          ohos:element = " $ graphic:checkbox_unchecked"/>
</state - container >
```

引用的资源文件 checkbox_checked.xml 的代码如下：

```
//ch03\LoginUIDemo 项目中的 checkbox_checked.xml
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape xmlns:ohos = "http://schemas.huawei.com/res/ohos"
         ohos:shape = "rectangle">
    < solid
         ohos:color = "#FF002CC9"/>
    < stroke
         ohos:color = "#666666"
         ohos:width = "1vp" />
</shape >
```

引用的资源文件 checkbox_unchecked.xml 的代码如下：

```
//ch03\LoginUIDemo 项目中的 checkbox_unchecked.xml
<?xml version = "1.0" encoding = "UTF - 8" ?>
< shape xmlns:ohos = "http://schemas.huawei.com/res/ohos"
         ohos:shape = "rectangle">
    < solid
         ohos:color = "#FFFFFF"/>
    < stroke
         ohos:color = "#666666"
         ohos:width = "1vp" />
</shape >
```

布局中引用的资源文件 bg_button.xml 的代码如下：

```
//ch03\LoginUIDemo 项目中的 bg_button.xml
<?xml version = "1.0" encoding = "utf - 8"?>
< shape
    xmlns:ohos = "http://schemas.huawei.com/res/ohos"
    ohos:shape = "rectangle"
>
    < corners
```

```
            ohos:radius = "30vp" />
        < solid
            ohos:color = "＃FFE2E2FF" />
    </shape >
```

以上示例通过常用的组件实现了一个应用的登录功能界面，当然，开发者可以根据需要设计出更加能够提高用户体验的界面效果。该例子中两次使用 DirectionalLayout 布局，关于布局后续章节还会详细阐述，这里暂时不展开。目前，这个版本的登录功能界面还不能进行登录操作，如果要使登录按钮能够响应单击登录功能，则需要进行组件事件监听等，这些将在接下来的章节中介绍。

除了本章介绍的常用 UI 组件外，HarmonyOS 体系的 UI 框架还提供了一些其他组件，相信读者通过学习已经掌握了组件的使用方法，能够举一反三。更多的组件，开发者可以参阅相关的技术手册，限于篇幅，这里不再赘述。

小结

本章主要讲解了常用的 UI 组件，HarmonyOS 体系中的 Java UI 提供了丰富的界面组件，本章主要介绍一些常用的 UI 组件，主要包括显示型组件和交互型组件，显示型组件主要用于显示界面内容，一般不接受用户操作，如文本框、图片等；交互型组件在显示界面内容效果的同时，一般还会接受用户操作，如输入框、按钮、开关等。组件是构成 HarmonyOS 应用界面的基本元素，组件经过修饰和组合可以得到丰富的界面效果，开发者可以在组件使用的基本方法之上构建丰富多彩的应用界面。

习题

1. 判断题

（1）组件一般直接继承 Component 或它的子类，如 Text、Image 等。（　　）

（2）组件参数中将宽度 ohos:width 的值设置为 match_content 时，表示组件大小与它的内容占据的大小范围相适应。（　　）

（3）组件参数中将高度 ohos:height 的值设置为 match_parent 时，表示组件大小为父组件允许的最大值，它将占据父组件方向上的剩余大小。（　　）

（4）组件设置独立 ID 的目的是为了方便在程序中查找该组件。（　　）

（5）在 Java UI 框架中，Component 和 ComponentContainer 以树状的层级结构进行组织，这样的一个大布局就称为组件树。（　　）

（6）Button 组件既可以显示图片也可以显示文本。（　　）

2. 选择题

（1）HarmonyOS 中，可使用的 UI 框架包括方舟开发框架和（　　）。

A. C++UI 框架　　B. Java UI 框架

C. Swing UI 框架　　D. VUE UI 框架

(2) 鸿蒙应用基于 JS 的 UI 开发采用(　　)和(　　)作为页面布局和页面样式的开发语言,页面业务逻辑则采用(　　)语言。

A. WML　　B. HTML　　C. Java

D. CSS　　E. XML　　F. JavaScript

(3) Text 组件中用于显示内容的属性是(　　)。

A. ohos:text　　B. ohos:id

C. ohos:width　　D. ohos:hint

(4) Image 组件中用于加载显示的图片资源的属性是(　　)。

A. ohos:img　　B. ohos:image_src

C. ohos:src　　D. ohos:pic

(5) Button 组件的 4 种类型主要是通过(　　)属性完成的。

A. ohos:type　　B. ohos:backgroud_element

C. ohos:btn_type　　D. ohos:backgroud

(6) RadioButton 一般和组件(　　)一起使用,以达到单选的效果。

A. RadioGroup　　B. Radio

C. RadioContainer　　D. RadioBox

3. 填空题

(1) 界面中所有组件的基类是_____。

(2) 当 Button 组件的形状 ohso:shape 的值为_____时,其类型为普通按钮和胶囊按钮。

(3) 当 Button 组件的形状 ohso:shape 的值为_____时,其类型为椭圆按钮和圆形按钮。

(4) TextField 组件是_____组件的直接子类。

(5) TextField 组件的提示信息可以通过_____属性设置。

4. 上机题

(1) 仿照 QQ,实现用户的登录界面。

(2) 结合某一业务需求,如餐厅服务质量调查,实现问卷调查界面。