User Manual

AWTK Designer 用户手册

用户手册

V1.0.2 Date:2025/10/31

类别	内容
关键词	AWTK Designer、使用说明、操作、许可
摘要	本文介绍了 AWTK Designer 的主要特性和基本使用流程，并对各个界面的操作进行了详细说明。

修订历史

版本	日期	原因
1.0.2	2022/12/08	• 添加第六章，快速入门指南
1.0.1	2022/10/19	• 删除第五章中的自定义控件相关内容
1.0.0	2022/06/17	• first implement

目 录

文档导读

本文介绍了 AWTK Designer 的主要特性和基本使用流程，并对各个界面的操作进行了详细说明，可以帮助读者了解 AWTK Designer。

1. 简介

AWTK Designer（下面简称 Designer）是专门用来制作 AWTK 应用程序 UI 界面的实用工具。只要通过拖拽和点击就可以完成复杂的界面设计，操作简单；可以随时预览效果，所见即所得，如下图所示。

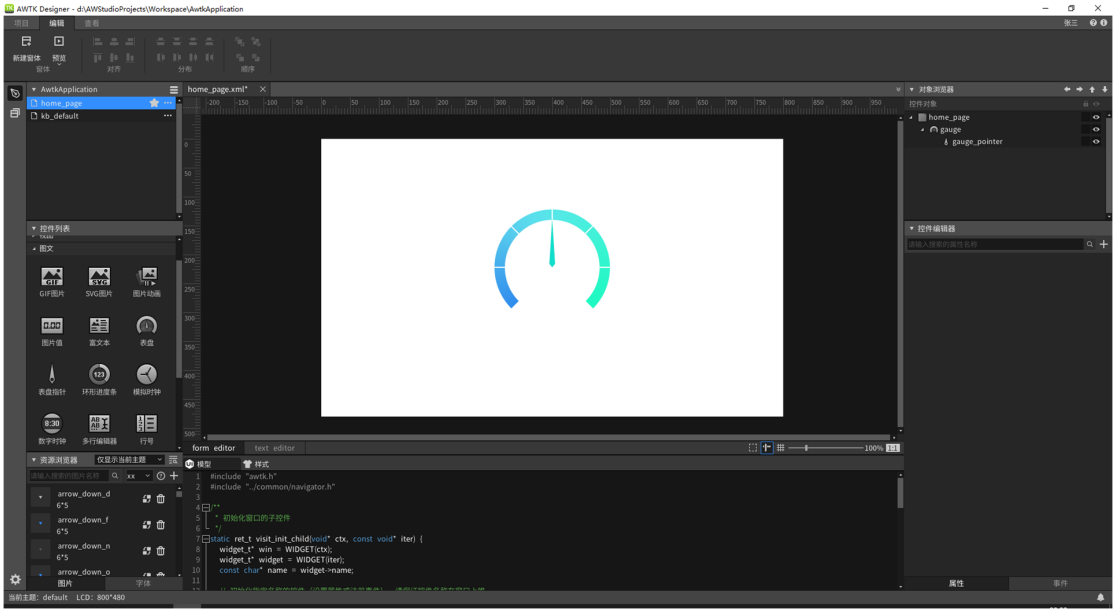


图 1.1　使用 Designer 设计界面

主要特性：

- 项目管理：以项目的形式管理 UI 文件、样式文件等文件，可以设置启动页面、默认国家语言、LCD、资源打包等；
- 图形编辑：以拖拽的方式在窗体上添加控件，改变控件位置、大小、层次等，所见即所得；
- 丰富的内置控件：内置丰富的可直接使用的控件，可以设置控件属性、样式等；
- 动画编辑：可以设置丰富的窗体过渡动画、控件动画；
- 多国语言翻译：可以快速检索需要翻译的文本、编辑翻译文本、预览翻译结果；
- 多主题编辑：可以编辑不同 LCD 分辨率、不同样式的主题，可以切换主题预览显示效果；
- 自定义控件导入：可以导入自定义控件，导入后可以像内置控件一样使用；
- 预览：可以选择运行时的渲染环境，实时预览运行时的效果；
- 打包：打包项目所有资源，转换为目标平台可以直接加载的资源；
- 编译与模拟运行：可以在 PC 上简单编译整个项目，直接模拟运行；
- 弱化的代码编辑：可以对 C 代码文件进行简单的文本编辑。

基本使用流程：

如下图所示，进入项目后可以通过新建、编辑窗体来完成界面设计；编辑的过程中可以

随时预览运行时的效果；之后通过打包资源、编译源代码、模拟运行，就可以启动应用程序。

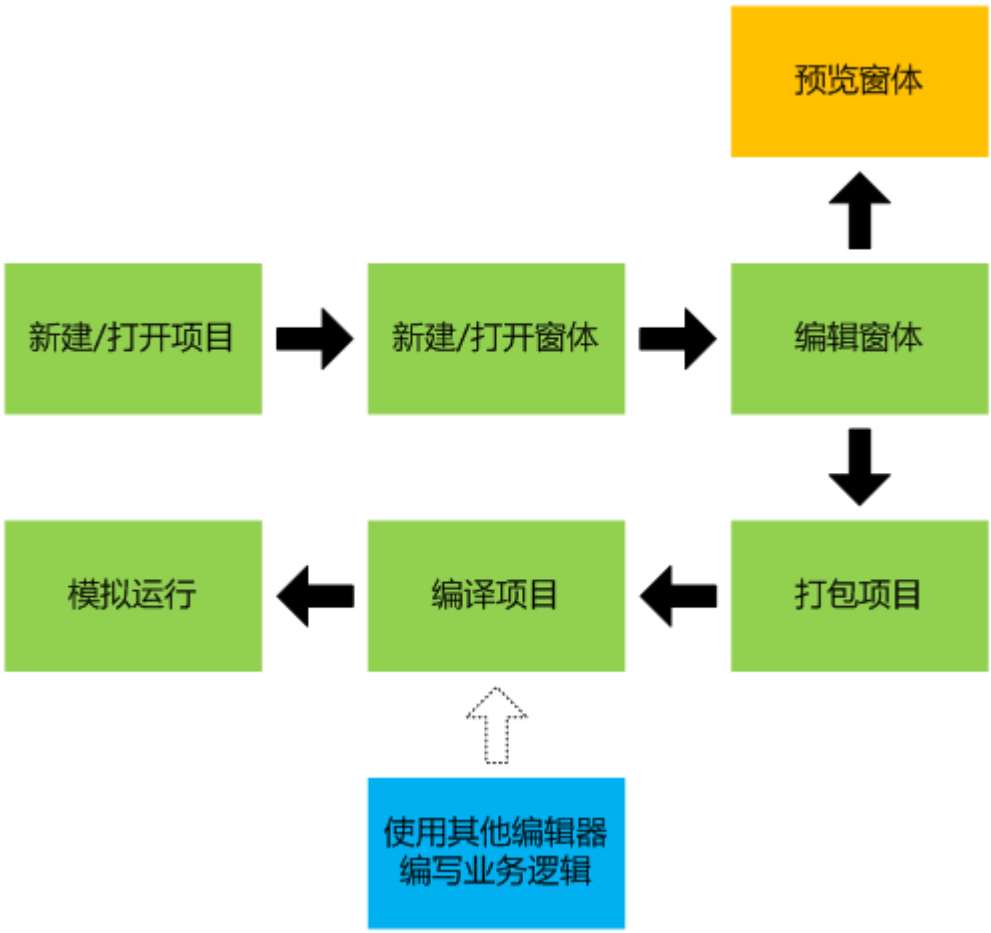


图 1.2　界面设计的基本流程

注：默认情况下，模拟运行仅是显示启动页面；由于目前 Designer 仅支持简单的文本编辑，因此也可使用其他编辑器编写代码，再重新编译、运行。

2. 界面操作与说明

下面详细介绍如何使用 Designer。

注: Designer 需要登录并激活许可后才能正常使用。关于许可的更多信息，请参阅本文第三章。这里假定 Designer 已激活许可。

2.1 新建/打开项目

Designer 以项目的形式来管理各类文件，因此，第一步先新建或打开一个项目。

2.1.1 新建项目

Designer 目前支持 2 种类型的项目，具体如下：

- 常规项目：基于 AWTK 的 GUI 应用程序，可以使用 XML 描述 UI、C 语言实现逻辑，可以通过 AWTK API 直接访问 Widget。
- 自定义控件：基于 AWTK 的自定义控件库，导入 AWTK Designer 后可以像内置控件一样编辑和实时预览显示效果。

1. 新建常规项目

Designer 启动后默认打开”新建项目”对话框，如下图所示。设置好项目的相关参数后，点击”创建”按钮即可创建一个常规项目。

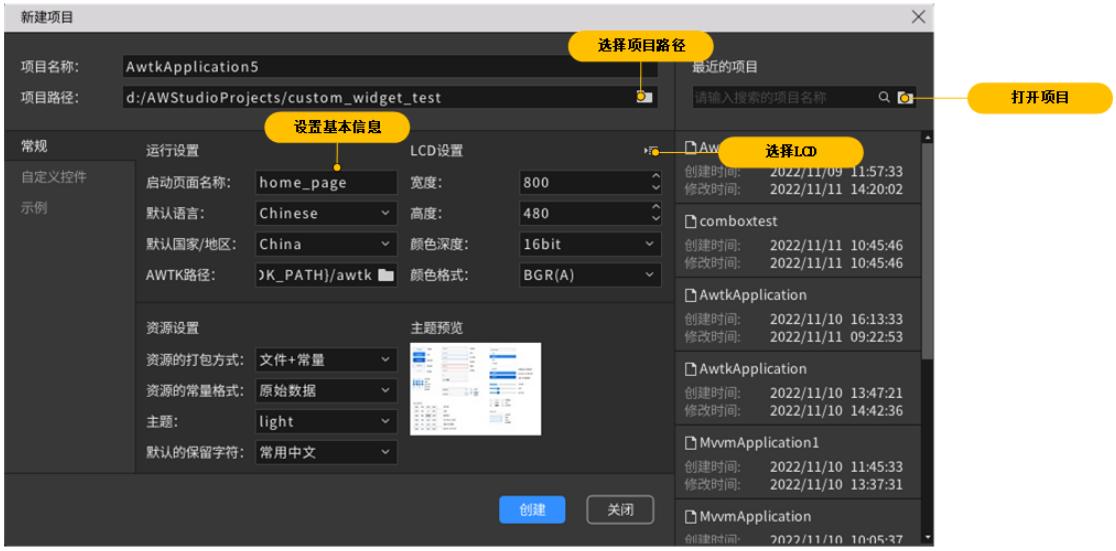


图 2.1 新建常规项目

注: 点击主界面工具栏的”新建项目”按钮，也可打开”新建项目”对话框。

需要设置的参数如下：

(1) 项目设置：包括项目名称和项目路径。

(2) 运行设置：

- 启动页面名称：应用程序启动时显示的第一个页面的名称。

- 默认语言：应用程序默认使用的语言。。
- 默认国家：应用程序默认使用的语言所属的国家或地区。
- AWTK 路径：项目依赖的 AWTK 的路径，默认为”${AWTK_SDK_PATH}/awtk”，表示 Designer 安装目录中的 SDK/awtk 目录。

(3) LCD 设置：

- 宽度：LCD 的宽度。
- 高度：LCD 的高度。
- 颜色深度:LCD 的颜色深度，选项有 1bit、16bit、32bit。
- 颜色格式:LCD 的颜色格式，选项有 MONO、RGB(A)、BGR(A)。
- LCD 的颜色深度和颜色格式决定了 LCD Frame Buffer 的格式：
  - 颜色深度为 1bit 时，颜色格式固定为 MONO，表示 Frame Buffer 为 MONO；
  - 颜色深度为 16bit 且颜色格式为 RGB(A)，表示 Frame Buffer 为 RGB565；
  - 颜色深度为 32bit 且颜色格式为 RGB(A)，表示 Frame Buffer 为 RGB8888；
  - 颜色深度为 16bit 且颜色格式为 BGR(A)，表示 Frame Buffer 为 BGR565；
  - 颜色深度为 32bit 且颜色格式为 BGR(A)，表示 Frame Buffer 为 BGR8888。

(4) 资源设置：

- 资源的打包方式，有 2 个选项：
  - 文件 + 常量：表示同时生成用于文件系统的资源和可直接编译到代码中的常量资源（以常量数组的形式存在）；
  - 仅文件：表示仅生成用于文件系统的资源。
- 资源的常量格式，它影响字体、图片打包生成的常量资源的格式。有 2 种格式：
  - 原始数据：表示数组中缓存的是原始的文件数据（比如，PNG 图片的原始数据为 PNG 数据）；
  - 位图数据：表示数组中缓存的是 Bitmap 数据。

(5) 插件设置：点击”插件设置”下方的预览图，可以进入的”选择插件”对话框，可以设置项目使用的插件。

需要注意的是：LCD Frame Buffer 的格式决定了打包生成的字体位图数据（也就是字模数据）和图片位图数据的格式。

对于字体：

- Frame Buffer 为 MONO 时，字模的 alpha 通道为 1bit；否则为 8bit。

对于图片：

- Frame Buffer 为 MONO 时，无论图片的颜色通道是什么格式，统一转为 1bit；原颜色灰度 >10 时为 1（表示白色），否则为 0（表黑色）。
- Frame Buffer 为 565 格式时，没有 alpha 通道的图片会转为 565 格式的位图数据，但有 alpha 通道的图片则仍会转为 8888 格式的图数据（运行时经过半透混合会转成 565 格式）；
- Frame Buffer 为 8888 格式时，则全部转为 8888 格式的位图数据。

另外，对于控件样式（比如背景颜色、文本颜色等）有一点需要注意：由于运行时计算灰度影响效率，Frame Buffer 为 MONO 时，AWTK 认为颜色的 R 通道为 0 时表示黑色，否则为白色；因此，建议 MONO 模式下，样式里的颜色均显式地设为黑色或白色。

2. 新建自定义控件项目

点击”新建项目”对话框中的”自定义控件”标签，如下图所示。设置好项目的相关参数后，点击”创建”按钮即可创建一个自定义控件项目。

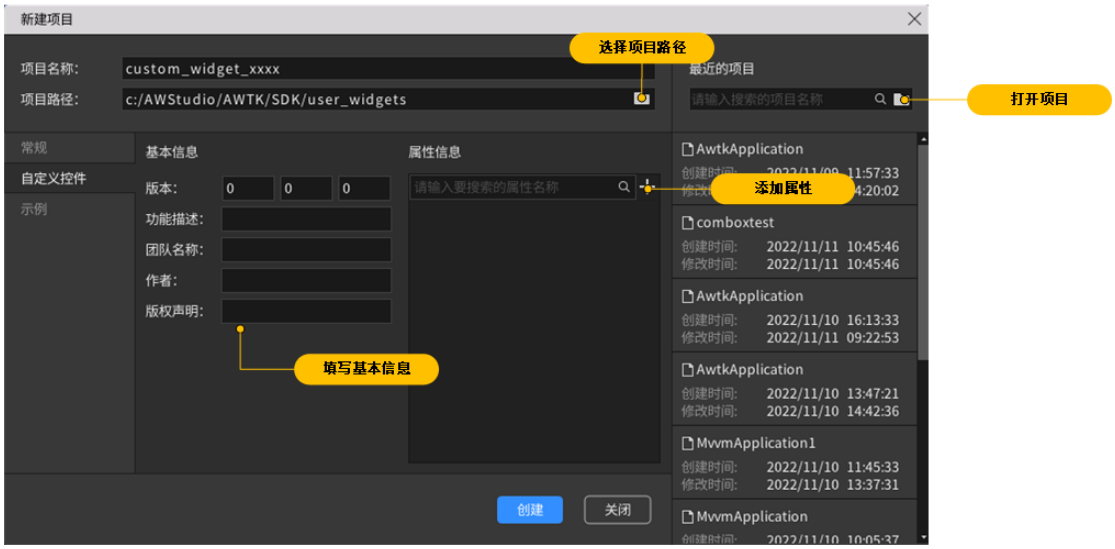


图 2.2 新建自定义控件项目

注: 打开该类型的项目时，Designer 会自动加载该控件库中的控件到控件列表。

需要设置的参数如下：

(1) 项目设置：包括项目名称和项目路径，其中项目名称为控件库的名称，须为全小写英文字母，单词之间用下划线连接。

(2) 基本信息：

- 版本：控件库的版本号。
- 功能描述：控件库功能的概要描述。
- 团队名称：控件库的开发团队的名称。
- 作者：控件库的作者及联系方式。
- 版权声明：控件库的版权声明。
- AWTK 路径：控件库依赖的 AWTK 的路径，默认为”${AWTK_SDK_PATH}/awtk”；AWTK_SDK_PATH 表示 Designer 安装目录中 SDK 目录。

(3) 属性信息：

- 名称：属性名称，须为全小写英文字母，单词之间用下划线连接。
- 类型：属性类型。
- 初始值：属性的初始值。
- 描述：属性的功能描述。

3. 插件设置

插件指的是可以在项目中复用的组件。目前 Designer 支持以下 3 种插件：

- 主题：也就是皮肤，由一个或多个描述控件外观（比如背景颜色、边框颜色、文本颜色、背景图片等）的样式文件组成。默认安装到项目主题目录的 styles 目录、images 目录。
- UI 片段：指已设计好的可在项目中直接使用的 UI，比如软键盘、系统栏等。默认安装到项目主题目录的 ui 目录。
- 自定义控件：指 AWTK 内置控件之外的其他任意控件。默认安装到项目的 3rd/widgets 目录。

注：

1. 对于同一类型的插件，插件的 name 是唯一的。导入 Designer 时，会对插件进行唯一性校验。
2. 关于主题目录的更多信息，请参阅本文 2.1.3 章节。如何使用自定义控件类型，请参考本文 5.1 章节。

图 2.3 插件管理

在 Designer 左侧点击插件管理可以进入插件管理栏

- 安装：点击“安装”按钮，可以将已存在的插件导入到 Designer 中以便复用。

- 更多操作：点击“更多”按钮，可以弹出菜单，比如删除本地插件、在资源管理器显示（即打开本地文件夹）重新加载控件等。
- 应用：对于自定义控件之外的插件有效。点击会弹出项目当前的主题列表，可以选择将插件应用到指定的主题。

需要注意：

- 应用到主题时如果没有指定主题，在插件右上角会显示黄色警告标志，鼠标悬停在警告标志上方时会提示”该插件实际不会添加到项目中”；
- 主题类型的插件会覆盖指定主题目录中的同名文件，因此注意备份 design 目录；
- 软键盘、系统栏插件会替换项目正在使用的软键盘、系统栏，因此注意备份 design 目录；其他类型的 UI 片段插件如果遇到同名文件会自动重命名。

2.1.2 打开项目

在”新建项目”对话框上，有两种方式可以打开已存在的项目：

- 双击“新建项目”对话框中的”最近的项目”列表中的一项，即可打开项目；
- 点击“新建项目”对话框中的”最近的项目”下方的”文件夹”按钮，可以打开下图所示的”打开项目”对话框。选中项目描述文件 project.json 后，点击”打开”按钮即可打开项目。

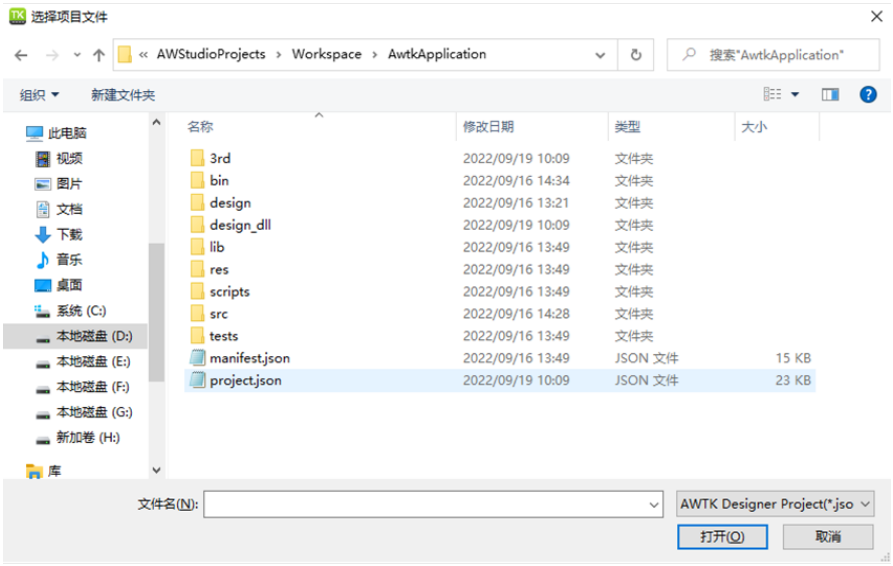


图 2.4 打开项目对话框

注: 点击主界面工具栏的”打开项目”按钮，也可打开”打开项目”对话框。

2.1.3 项目的目录结构

默认情况下，项目的目录结构如下表所示：

项目	说明
bin	存放可执行文件。
demos	存放 Demo 的源代码，仅自定义控件库项目会自动创建。
design	存放项目的原始资源，可在 Designer 上编辑。
res	存放项目运行时需要的资源，由 Designer 打包或者 scripts/update_res.py 脚本生成。
scripts	存放打包资源的脚本。
src	存放项目源码。
project.json	项目的描述文件，包含项目设置等信息。
SConstruct	Scons 编译脚本。

项目创建后，design 目录默认有一个名称为”default”的主题目录。主题的目录结构如下表所示：

项目	说明
fonts	存放该主题使用的字体，内涵子目录详见下文。
images	存放该主题使用的图片，内涵子目录详见下文。
strings	存放该主题使用的语言文件。
styles	存放该主题使用的样式文件。
ui	存放该主题使用的 UI 文件。
data	存放字体、图片、语言、样式、UI、XML、脚本之外的文件。

(1) fonts 目录中包含以下子目录：

- config：临时目录，存放记录字体的保留字符的文件；
- origin：临时目录，存放被裁剪的 TTF 的原始文件。

(2) images 目录中包含以下子目录：

- xx：存放与屏幕密度无关的图片；
- x1：存放普通密度屏幕上使用的图片；
- x2：存放 2 倍密度屏幕上使用的图片；
- x3：存放 3 倍密度屏幕上使用的图片；
- svg：存放 svg 图片。

AWTK 运行时会优先在 x1、x2、x3 目录中的其中一个查找图片（比如 LCD 的 DPR=1，则在 x1 目录），如果找不到，则在 xx 目录中查找。

设备像素比（DPR）= 设备像素（又称物理像素）/设备独立像素（又称密度无关像素或逻辑像素，可以认为是计算机坐标系统中的一个点，代表一个可以由程序使用的虚拟像素），比如 DPR=2，则表示 1 逻辑像素等于 2 物理像素。

项目的 src 目录结构，如下表所示：

项目	说明
SConscript	用于管理 src 目录下的源代码。
main.c	应用程序的 main 文件，包含 main 函数。
application.c	Application 实例的代码文件，可添加应用程序的初始化和退出时的逻辑。
pages	存放窗体对应的 C 代码，与 ui 目录的结构一致。

2.2 主界面

新建/打开项目后，Designer 会进入主界面并默认打开启动页面。主界面可分为如下图所示的若干个区域，下面分别对各个区域进行详细介绍。

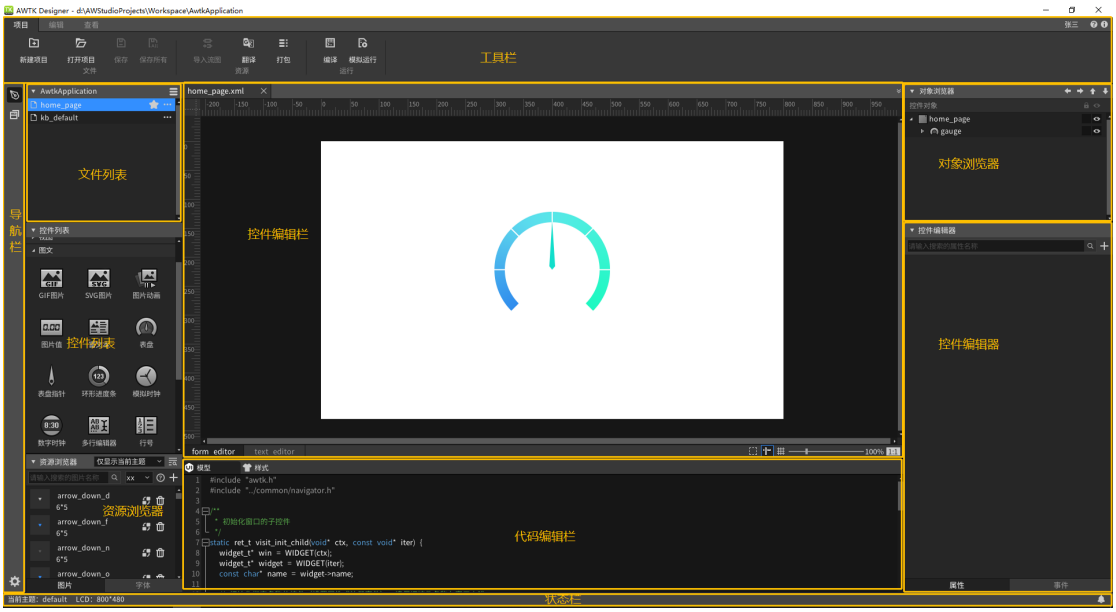


图 2.5 主界面

2.3 工具栏

工具栏由一系列常用操作的按钮组成，可以分为三大类。

2.3.1 项目操作

图 2.6 工具栏上提供的项目操作

工具栏上提供的项目操作如下：

- 新建项目：点击“新建项目”按钮，可以弹出“新建/打开项目”对话框。
- 打开项目：点击“打开项目”按钮，可以弹出“打开项目”对话框。
- 保存：点击“保存”按钮，可以保存当前正在编辑的文件。

- 保存所有：点击“全部保存”按钮，可以保存已打开的全部文件。
- 翻译：点击“翻译”按钮，可以启动多国语言翻译，Designer 将自动检索 UI 文件中需要翻译的文本，并更新到语言文件（strings.xml）。
- 打包：点击“打包”按钮，可以将 design 目录下的原始资源转换为 AWTK 应用程序运行时需要的格式，并输出到指定目录（默认为 res 目录）。
- 编译：点击“编译”按钮，可以打开终端并调用 Scons 命令编译源代码，如果编译成功，则会在 bin 目录生成可执行文件。
- 模拟运行：点击“模拟运行”按钮，可以运行编译生成的可执行文件（bin/demo.exe）。

1. 打包

如果项目未打包或 design 目录下的原始资源发生修改，可以点击”打包”按钮进行打包，设置参数详见 2.4.1 章节。

假设项目有两个主题（default、dark），打包后的结果如下图所示。

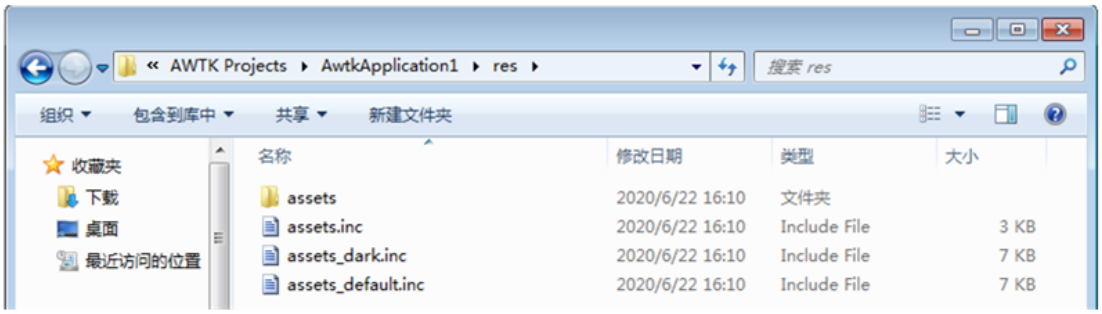


图 2.7 打包结果

Designer 默认以多主题的方式使用生成的资源，也就是在 app_main.c 中默认引用 assets.inc，代码如下。如果需要在运行时切换主题，则只需调用 assets.inc 中提供的 assets_set_global_theme() 即可。

```
// app_main.c
/**************************************************************/
/*** THIS FILE IS GENERATED BY AWTK DESIGNER, DO NOT MODIFY ****/
/**************************************************************/
#include "awtk.h"
#include "../res/assets.inc"

extern ret_t application_init(void);

extern ret_t application_exit(void);

#include "awtk_main.inc"
```

注: 如果目标平台不支持文件系统且内存资源等有限，可以选择仅使用单个主题，比如仅使用 dark 主题，则修改 app_main.c 中的 #include “…/res/assets.inc”为 #include “…/res/assets_dark.inc”即可。

2. 编译

点击”编译”按钮，可以打开终端并调用 Scons 命令编译源代码，如果编译成功，则会在 bin 目录生成可执行文件。

注: 这里假定已经安装好编译环境。如果未安装，请参阅《AWTK 开发实践》中的 1.4 章节。

编译前需要注意，要使编译正常有 2 个前提条件：

- 需要先打包资源

如果编译前资源未打包，会出现两个问题：一个是找不到资源文件会导致编译失败，对于这种情况 Designer 会有如下图所示的提示；另一个是资源没有更新会导致运行时显示结果与实际不符。

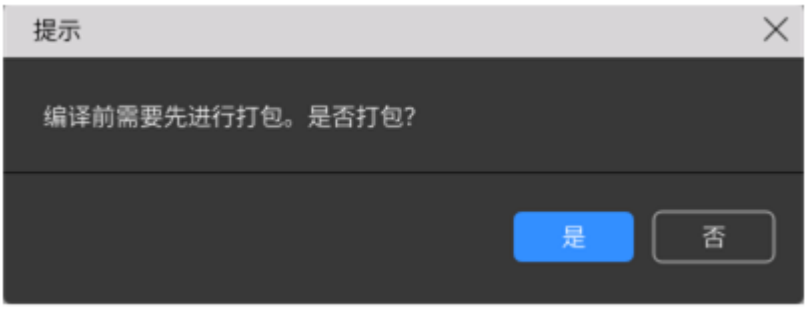


图 2.8 编译时的打包提示

- 依赖的 AWTK 必须存在且已编译

如果检查到 AWTK 未编译，Designer 会有如下图所示的提示。此时，点击”是”按钮，可以打开终端并开始编译 AWTK。编译成功后，再点击”编译按钮”重新项目。

图 2.9 检查到 AWTK 未编译时的提示

推荐使用安装 Designer 时附带的 AWTK。但如果需要使用其他版本的 AWTK，可以进入项目设置修改 AWTK 路径。

2.3.2 编辑栏

图 2.10 工具栏上提供的窗体操作

工具栏上提供的窗体操作如下：

- 新建窗体：点击“新建窗体”按钮，可以弹出“新建窗体”对话框，完成设置后可新建一个窗体。
- 预览：点击“预览”按钮，可以预览当前正在编辑的窗体的运行时效果。
- 对齐：选中 2 个以上控件时有效，可实现多个控件的上/下/左/右/水平/垂直居中对齐。
- 分布：选中 3 个以上控件时有效，可实现多个控件的垂直/水平、靠上/下/左/右/居中的均匀分布。
- 顺序：修改控件在 Z 方向上的层次顺序，可移动控件层级，将其置于顶层/底层或者上移/下移一层。

1. 新建窗体

点击工具栏上的”新建窗体”按钮，可以打开”新建窗体”对话框。设置好窗体类型、名称、路径后点击”创建”即可新建窗口。

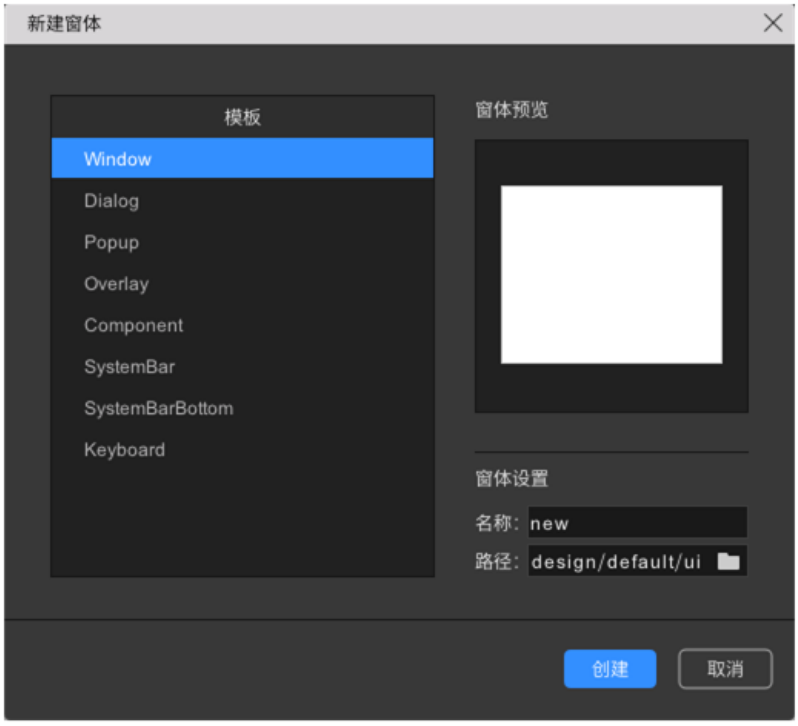


图 2.11 新建窗体对话框

Designer 目前支持的窗体类型如下：

- Window：缺省窗口，占据除 SystemBar 之外的整个区域，大小和位置不可以设置。
- Dialog：对话框，大小和位置可自由设置，可以是模态的，也可以是非模态的。
- Popup：弹出窗口，大小和位置可自由设置，主要用来实现右键菜单和下拉列表。
- Overlay：悬浮窗口，大小和位置可自由设置，同时不阻止窗口外的事件向下传递。
- Component：可复用的 UI 组件，比如作为 tab_button 控件的 load_ui 属性实现动态加载。
- SystemBar：顶部系统栏，独占 LCD 顶部区域，用来显示当前窗口的标题、关闭按钮等；一个应用只有一个，需在打开应用的第一个窗口之前打开。
- SystemBarBottom：底部系统栏，独占 LCD 底部区域，用来显示状态信息等；一个应用只有一个，需在打开应用的第一个窗口之前打开。
- Keyboard：软键盘，由编辑控件通过输入法自动打开和关闭。

如果编辑控件设置了 keyboard 属性，则使用该属性值作为键盘的 UI 文件名。如果未设置，则使用 input_type 属性对应的 UI 文件名。input_type 属性与 UI 文件的对应关系如下：

- phone 对应 kb_phone.xml；
- int 对应 kb_int.xml；
- uint、ipv4、date、time、time_full 对应 kb_uint.xml；
- float 对应 kb_float.xml；
- ufloat 对应 kb_ufloat.xml；
- hex 对应 kb_hex.xml；
- email、password 对应 kb_ascii.xml；
- custom、custom_password 表示使用自定义键盘；
- 其他对应 kb_default.xml。

对于常规项目，新建窗体成功后会自动在 src/pages 目录创建窗体对应的 C 代码文件；

2. 窗体预览

当鼠标悬浮在”预览”按钮上方时会显示当前预览的模式；点击图标下方的”预览”文本，可以选择预览模式；点击预览图标，可以预览正在编辑的窗体，如下图所示。

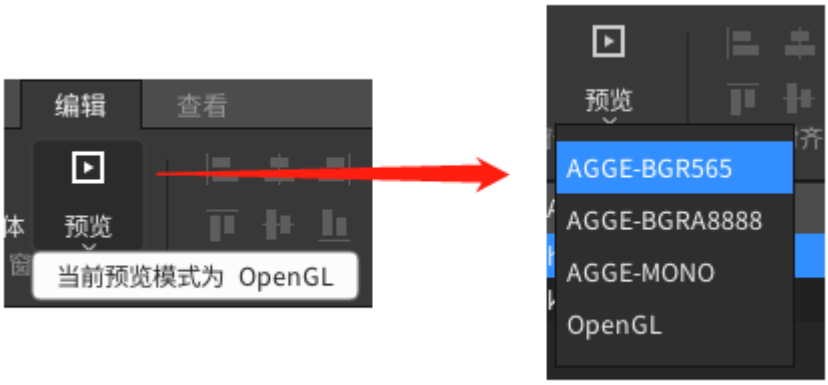


图 2.12　窗体预览

Designer 目前支持的预览模式如下：

- AGGE-BGR565：表示运行时绘图使用 AGGE 引擎，LCD Frame Buffer 的格式为 BGR565。
- AGGE-BGRA8888：表示运行时绘图使用 AGGE 引擎，LCD Frame Buffer 的格式为 BGRA8888。
- AGGE-MONO：表示运行时绘图使用 AGGE 引擎，LCD Frame Buffer 的格式为 MONO。
- OpenGL：表示运行时绘图使用 OpenGL 引擎。

3.　对齐的参考控件

参考控件指的是对齐时的基准控件。如下图所示，参考控件为 edit 控件，点击”左对齐”时则会 edit 控件的左边缘为基准进行对齐。

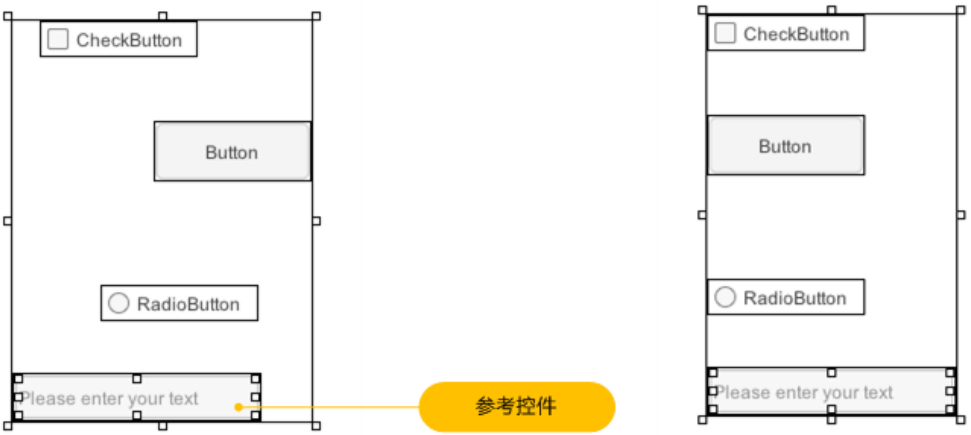


图 2.13　对齐时的参考控件

如果需要修改参考控件，则单击被选中的其他控件即可。

2.3.3 查看栏

图 2.14 查看栏

工具栏上提供的查看操作如下：

- 全局样式：点击“全局样式”按钮，可以查看“全局样式”列表。
- APP 对象：点击“APP 对象”按钮，可以查看 `application.c` 的代码文件。
- 终端：点击“终端”按钮，可以弹出“调试终端”框

1. 全局样式

在全局样式列表可以查看控件的默认样式，也可以在此处进行添加或者修改，保存后会直接修改 `design/default/styles/default.xml` 文件中的对应属性。

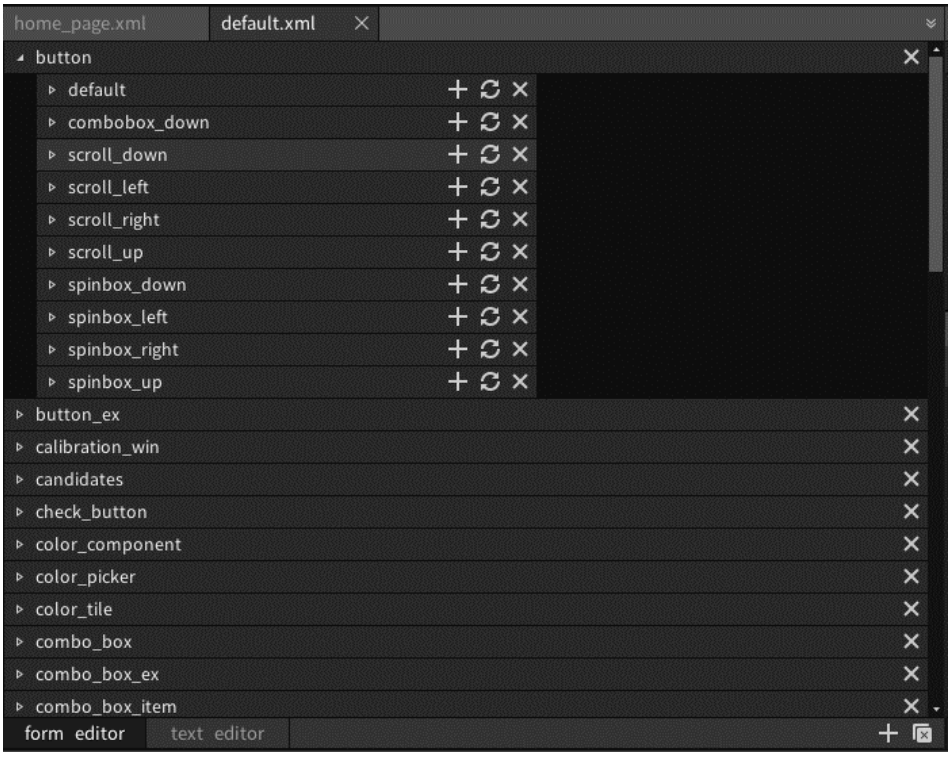


图 2.15 全局样式

2. APP 对象

在 APP 对象中可以查看当前项目的 `application.c` 文件

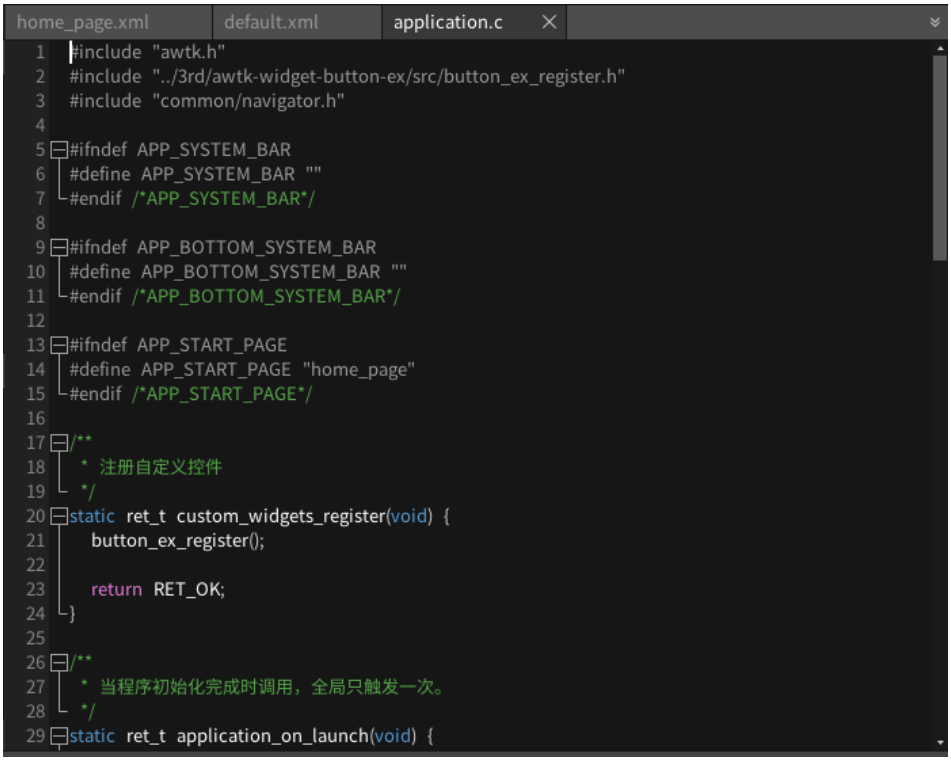


图 2.16 application.c

3. 终端

点击”终端“按钮会弹出可用于调试应用的终端栏，点击”模拟运行“按钮时也会自动弹出。

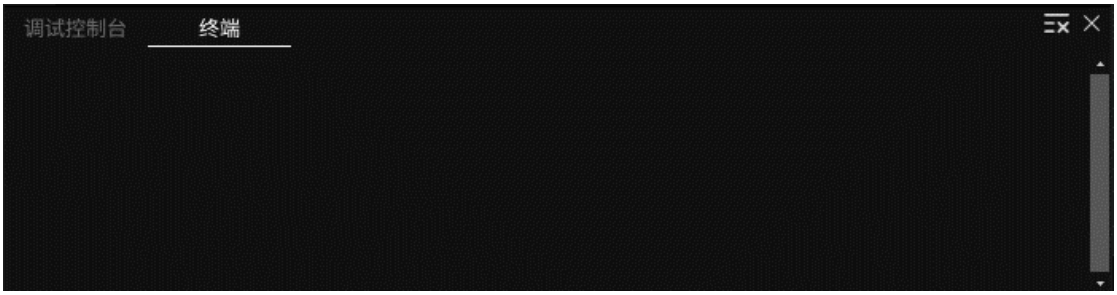


图 2.17 终端

2.4 导航栏

导航栏主要用于窗口导航，目前有三个按钮，功能如下：

- “编辑”按钮：点击切换至编辑功能页面，目前只有编辑功能。
- “插件管理”按钮：点击切换至插件管理页面。
- “设置”按钮：点击可以打开设置菜单，可以选择通用设置、项目设置。

2.4.1 通用设置

点击导航栏的”设置”按钮，之后点击”通用设置”，可以打开”通用设置”对话框。通用设置可分为”环境设置”与”文件设置”。

1. 环境设置

环境设置页面如下图所示。

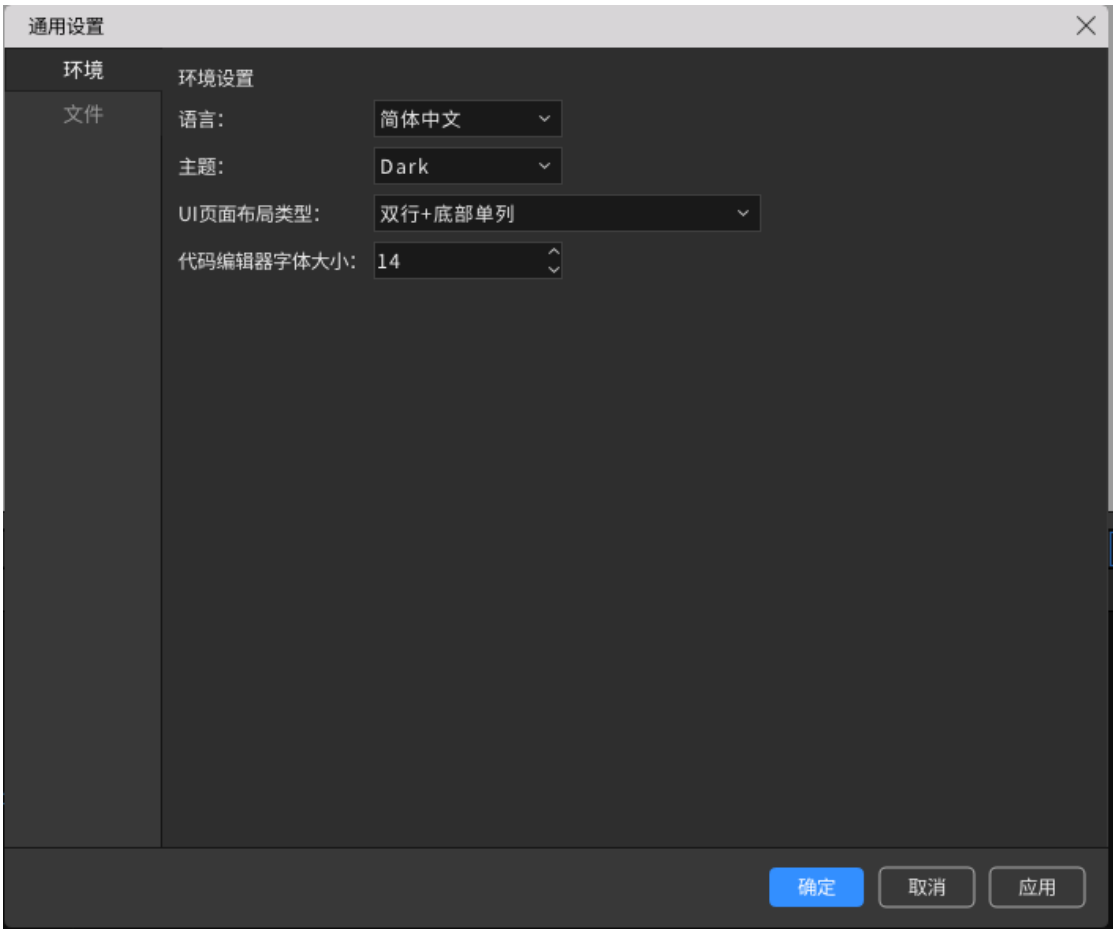


图 2.18 环境设置

环境设置页面可以设置的参数如下：

- 语言：Designer 所显示的语言，有如下选项：简体中文、English。
- 主题：Designer 所显示的主题，有如下选项：Dark。
- UI 页面布局类型：Designer 主页面的 UI 布局。

- 代码编辑器字体大小：Designer 代码编辑器字体的大小。

2. 文件设置

文件设置主要包括使用 Designer 编辑文件过程中的相关设置，如下图所示。

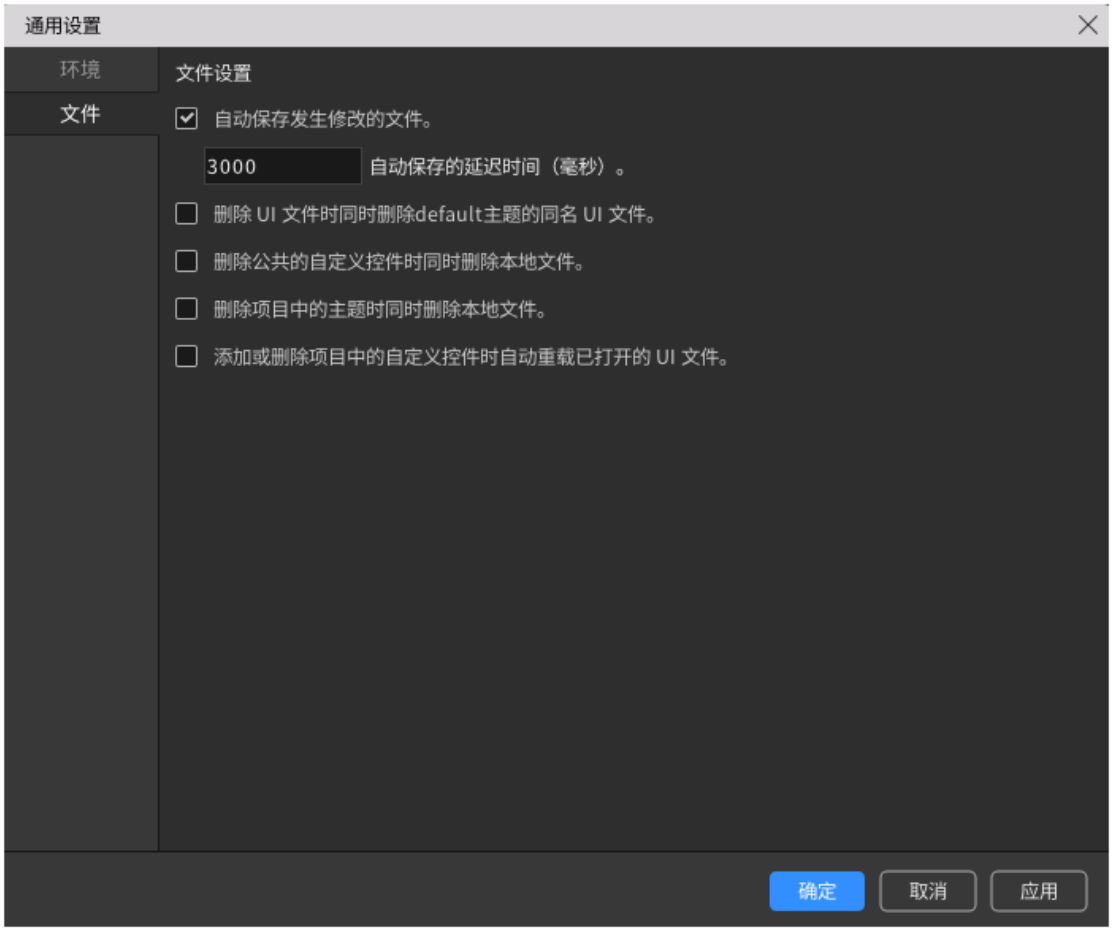


图 2.19 文件设置

文件设置页面可以设置的参数如下：

- 自动保存发生修改的文件：文件发生需改后是否自动保存。自动保存的延迟时间以毫秒为单位。
- 删除 UI 文件时同时删除 default 主题的同名 UI 文件：删除 UI 文件后是否自动删除 default 主题中的同名 UI 文件。
- 删除公共的自定义控件时同时删除本地文件：删除公共的自定义控件时是否同时删除本地文件。
- 删除项目中的主题时同时删除本地文件：删除项目中的主题时是否同时删除本地文件。
- 添加或删除项目中的自定义控件时自动重载已打开的 UI 文件：添加或删除项目中的自定义控件时是否自动重载已打开的 UI 文件。

2.4.2 项目设置

点击导航栏的”设置”按钮，之后点击”项目设置”，可以打开”项目设置”对话框。项目设置可分为”常规设置”和”主题设置”。

1. 常规设置

常规设置主要包括运行设置和资源打包设置，如下图所示。

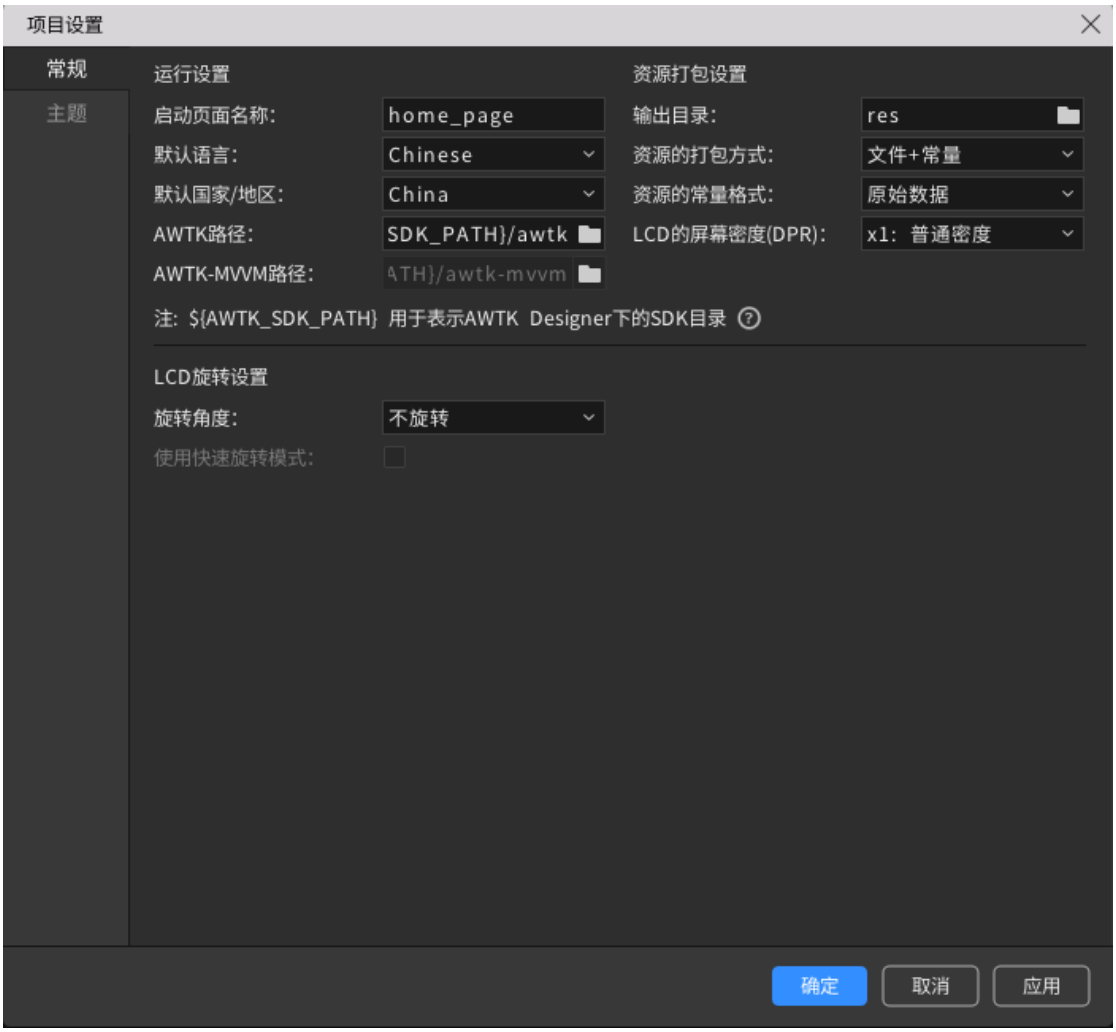


图 2.20 项目的常规设置

常规设置可以设置的参数如下：

(1) 运行设置：

- 启动页面名称：应用程序启动时显示的第一个页面的名称。
- 默认语言：应用程序默认使用的语言。
- 默认国家：应用程序默认使用的语言所属的国家或地区。
- AWTK 路径：项目依赖的 AWTK 的路径。

(2) 资源打包设置：

- 输出目录：打包生成的资源的输出目录，默认为项目的 res 目录。

- 资源的打包方式：
  - 文件 + 常量：表示同时生成用于文件系统的资源和可直接编译到代码中的常量资源（以常量数组的形式存在）；
  - 仅文件：表示仅生成用于文件系统的资源。
- 资源的常量格式：
  - 原始数据：表示数组中缓存的是原始的文件数据（比如，PNG 图片的原始数据为 PNG 数据）；
  - 位图数据：表示数组中缓存的是 Bitmap 数据。
- LCD 的设备像素比（DPR）：影响打包生成常量资源时应包含哪些图片资源。
  - DPR=x1 时，图片资源包含 x1、xx、svg；
  - DPR=x2 时，图片资源包含 x2、xx、svg；
  - DPR=x3 时，图片资源包含 x3、xx、svg；

注：关于 DPR 与图片的关系，请参阅本文 2.1.3 章节。

(3) LCD 旋转设置

- 旋转角度：改变 LCD 显示时的方向，可选项有：逆时针 90°、逆时针 180°、逆时针 270°。
- 使用快速旋转模式：可以选择是否打开快速旋转模式。

2. 主题设置

图 2.21 项目的主题设置

通过”主题”选项页，可以进行以下操作：

- 添加主题：点击主题列表右上角的”+”按钮，可以在弹出的编辑框中设置新主题的名

称；“确定”或“应用”设置后，如果 design 目录存在同名的子目录，则直接使用该目录为主题目录；否则新建主题目录。

- 重命名主题：双击主题名称，可以重命名主题；“确定”或“应用”设置后，会重命名对应的目录。
- 激活主题：”星星”标记的主题为当前使用的主题；“确定”或“应用”设置后，窗体编辑器将按该主题的设定显示窗口的外观。
- 设置打包的主题：”星星”标记右侧的”折叠”标记表示打包时是否生成对应的资源文件。
- 删除主题：点击主题列表右上角的”删除”按钮，可以删除主题；如果勾选同时删除本地文件，“确定”或“应用”设置后，会删除对应的主题目录。
- 搜索主题设置：在主题列表上的搜索框中输入主题的名称，可以搜索主题列表中包含该名称的主题。
- 编辑主题设置：点击选中主题列表中的某个主题，“LCD 设置”、“系统栏设置”、“主题的字体列表”等会更新为该主题的设置，之后直接编辑即可。

(1) 主题的 LCD 设置

LCD 设置相关的参数如下：

- 宽度：LCD 的宽度。
- 高度：LCD 的高度。
- 颜色深度：LCD 的颜色深度，有如下选项：1bit、16bit、32bit。
- 颜色格式：LCD 的颜色格式，有如下选项：MONO、RGB(A)、BGR(A)。

LCD 的颜色深度和颜色格式决定了 LCD Frame Buffer 的格式：

- 颜色深度为 1bit 时，颜色格式固定为 MONO，表示 Frame Buffer 为 MONO；
- 颜色深度为 16bit 且颜色格式为 RGB(A)，表示 Frame Buffer 为 RGB565；
- 颜色深度为 32bit 且颜色格式为 RGB(A)，表示 Frame Buffer 为 RGB8888；
- 颜色深度为 16bit 且颜色格式为 BGR(A)，表示 Frame Buffer 为 BGR565；
- 颜色深度为 32bit 且颜色格式为 BGR(A)，表示 Frame Buffer 为 BGR8888。

注: 关于 LCD Frame Buffer 的更多信息，请参阅本文 2.1.1 章节。

不同的主题可以有不同的 LCD 设置。比如，default 主题的 LCD 为 800*480，dark 主题的 LCD 为 320*480，则两个主题的显示效果如下图所示。

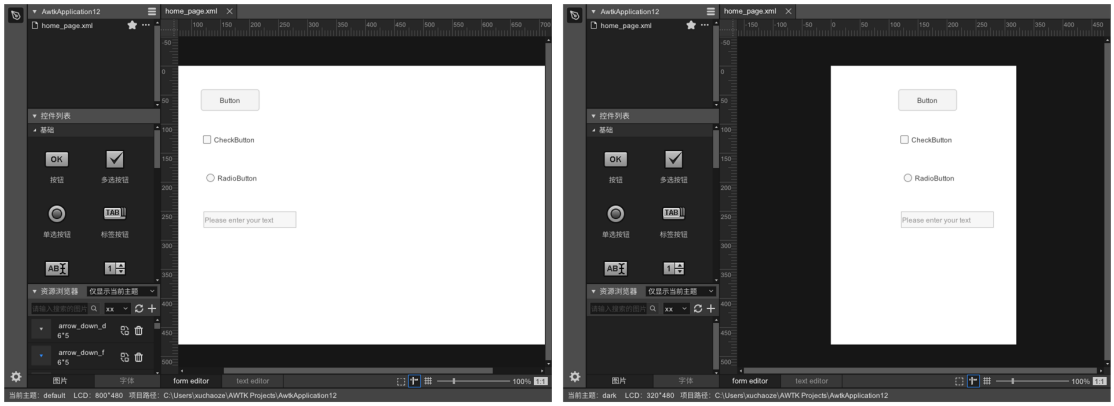


图 2.22　不同 LCD 尺寸的主题

(2) 主题的系统栏设置

系统栏设置的相关参数如下：

- 顶部系统栏：UI 文件相对于 UI 目录的路径。
- 底部系统栏：UI 文件相对于 UI 目录的路径。

注: UI 文件存放在主题目录的 UI 目录下，比如：design/default/ui。

不同的主题可以有不同的系统栏。比如，default 主题下创建一个 systembar（文件名为 system_bar_default.xml）并设置顶部系统栏为 system_bar_default，dark 主题下创建一个 systembar（文件名为 system_bar_dark.xml）并设置顶部系统栏为 system_bar_dark，则两个主题的显示效果如下图所示。

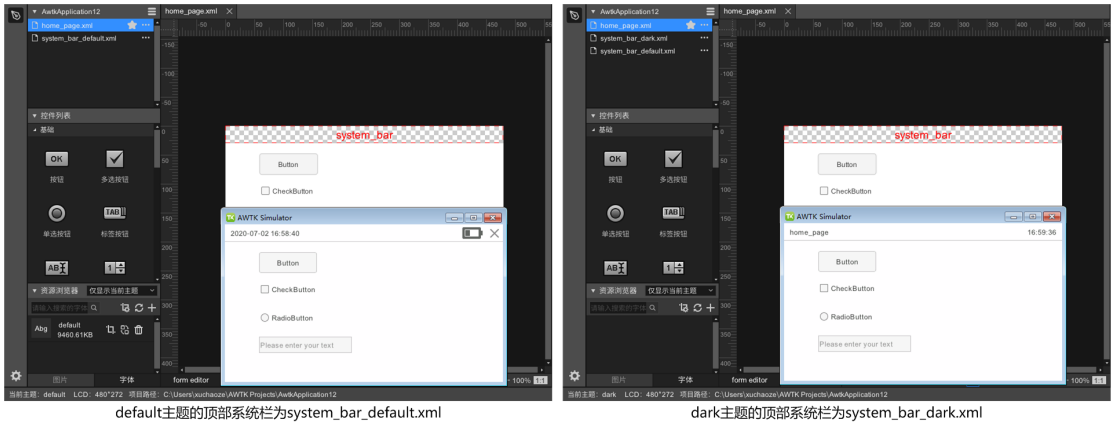


图 2.23　使用不同系统栏的主题

(3) 主题的字体设置

主题列表下方的”主题的字体列表”会显示当前主题使用的所有字体，”字体设置”则是选中的字体的裁剪设置。

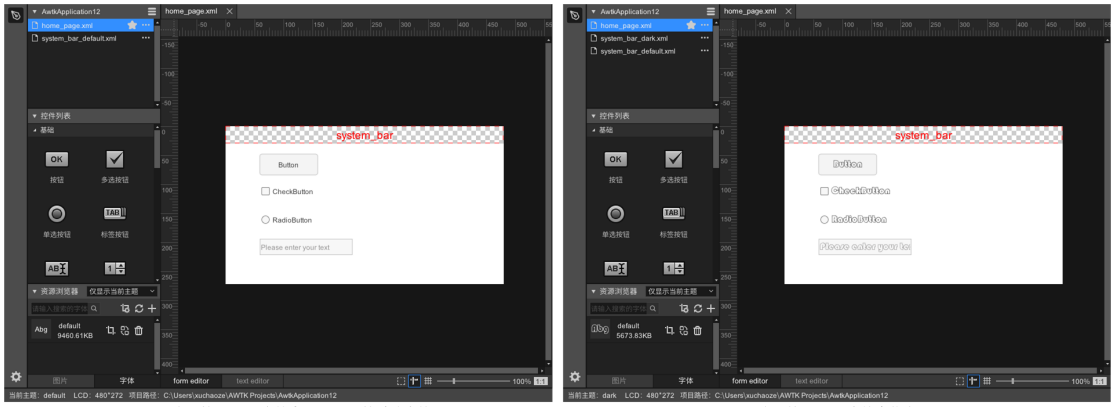
字体设置的相关参数如下：

- 裁剪格式：用来切换裁剪格式，不作为设置的一部分。比如，裁剪格式为 TTF 时，则切换到 TTF 字体的裁剪设置；裁剪格式为 Bitmap 时，则切换到位图字体的裁剪设置。
- 保留的字号：表示字体裁剪后需要保留的字号，裁剪格式为 Bitmap 时有效；点击右侧

的添加按钮可以添加字号，点击字号列表中的删除字体可以删除字号。

- 保留的字符：表示字体裁剪后需要保留的字符；当裁剪格式为 TTF 时，表示 TTF 文件内保留的字符；当裁剪格式为 Bitmap 时，表示对应字号要保留的字符。

不同的主题可以有不同的字体。比如，default 主题的 default 字体为 Designer 创建项目时的默认字体，dark 主题的 default 字体为华文彩云（添加到字体列表中后，在资源浏览器中重命名为 default），则两个主题的显示效果如下图所示。



default主题的default字体为Designer的默认字体　　dark主题的default字体为华文彩云

图 2.24　不同字体设置的主题

2.5　状态栏

图 2.25　状态栏

状态栏用来显示一些信息以及提供一些快捷操作，如下：

- 当前主题：显示当前激活的主题，单击可打开”选择当前主题”对话框，可以选择激活其他主题。
- LCD 尺寸：显示当前激活的主题的 LCD 尺寸，单击可打开”选择 LCD”对话框，可以设置为其他常用的 LCD 尺寸。
- 消息中心：显示账户内未读消息的数量，单击可打开”消息中心”，可以查看消息详情。

2.6　项目管理器

新建或打开项目后，项目管理器窗口会显示项目的文件列表以及提供一些常用操作。

2.6.1 简洁模式

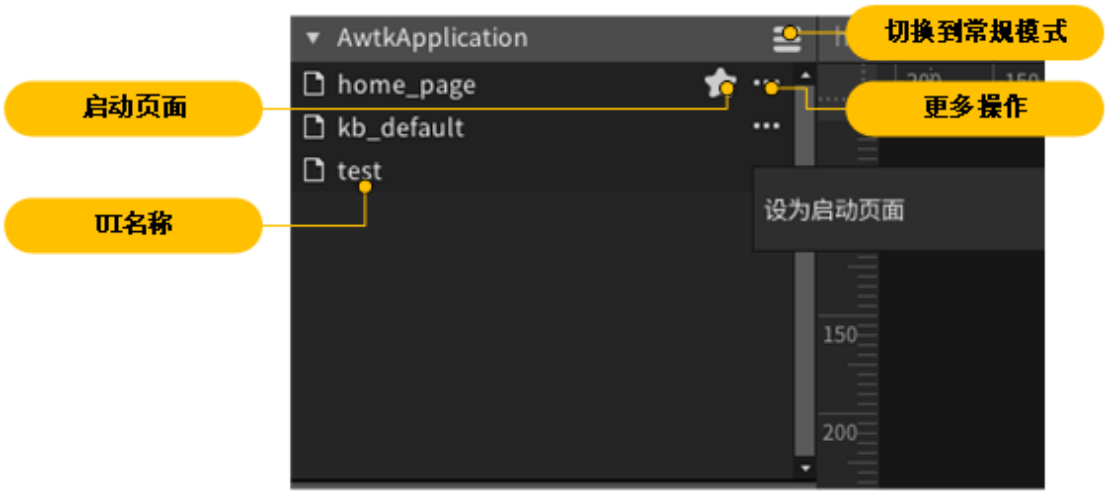


图 2.26 项目管理器的简洁模式

注: AWTK 优先在当前主题中查找 UI 文件，如果找不到则会在 default 主题内查找。因此，如果当前主题中不存在一个同名的 UI 文件，则会显示 default 主题中对应的文件。

比如，项目有 default、dark 两个主题且当前激活的是 dark 主题，如果 dark/ ui/ home_page.xml 存在，则上图中的 home_page 指的是该文件；否则指的是 default/ ui/ home_page.xml。

项目管理器窗口默认为简洁模式，简洁模式仅显示当前主题的所有 UI 文件，可以进行以下操作。

- 打开 UI 文件：双击 UI 文件，可以打开该文件进行编辑，更多信息请参阅 2.11。
- 设置启动页面：点击文件右侧的”更多”按钮，再点击“设为启动页面”，可以将该文件设为项目的启动页面，启动页面的右侧有”星星”标志。
- 切换为常规模式：点击”切换”按钮，可以切换到常规模式。
- 打开右键菜单：右键点击，可以显示右键菜单，更多信息请参阅本文 2.6.3 章节。

2.6.2 常规模式

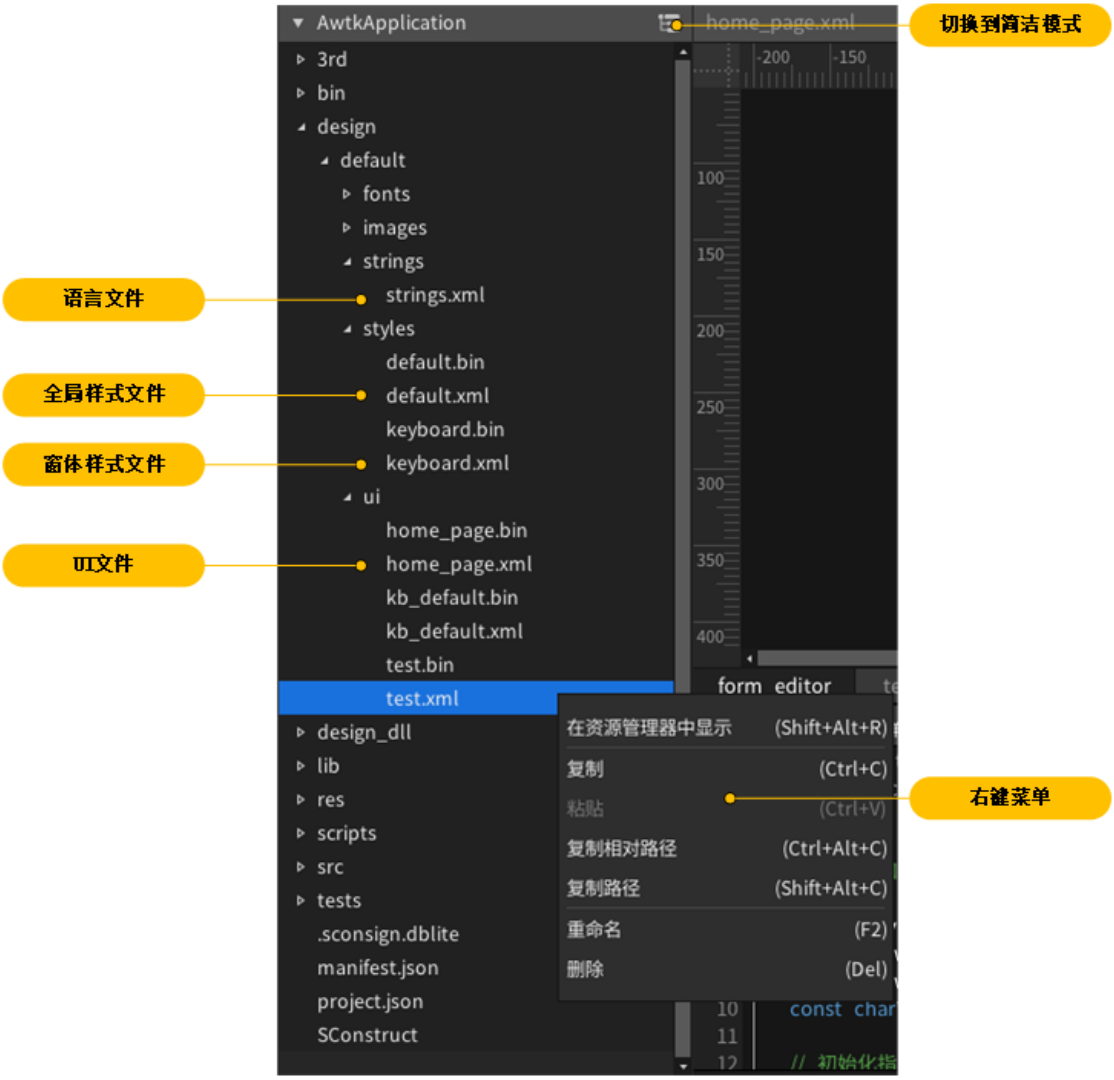


图 2.27 项目管理器的常规模式

项目管理器窗口的常规模式会显示项目在磁盘上的目录结构，可以进行以下操作：

- 打开 UI 文件：双击主题目录中 ui 目录的 UI 文件，可以打开该文件进行编辑，更多信息请参阅 2.12。
- 打开样式文件：双击主题目录中 styles 目录的样式文件，可以打开该文件进行编辑，更多信息请参阅 2.13
- 打开语言文件：双击主题目录中 strings 目录的 strings.xml，可以打开该文件进行编辑，更多信息请参阅 2.14。
- 打开文本文件：双击项目目录中除二进制格式、UI、样式、语言文件之外的其他文件，可以打开文件并编辑。
- 切换为简洁模式：点击”切换”按钮，可以切换到简洁模式。
- 打开右键菜单：右键点击，可以显示右键菜单，更多信息请参阅 2.6.3。

注: 关于主题目录的更多信息，请参阅 2.1.3。

2.6.3 右键菜单

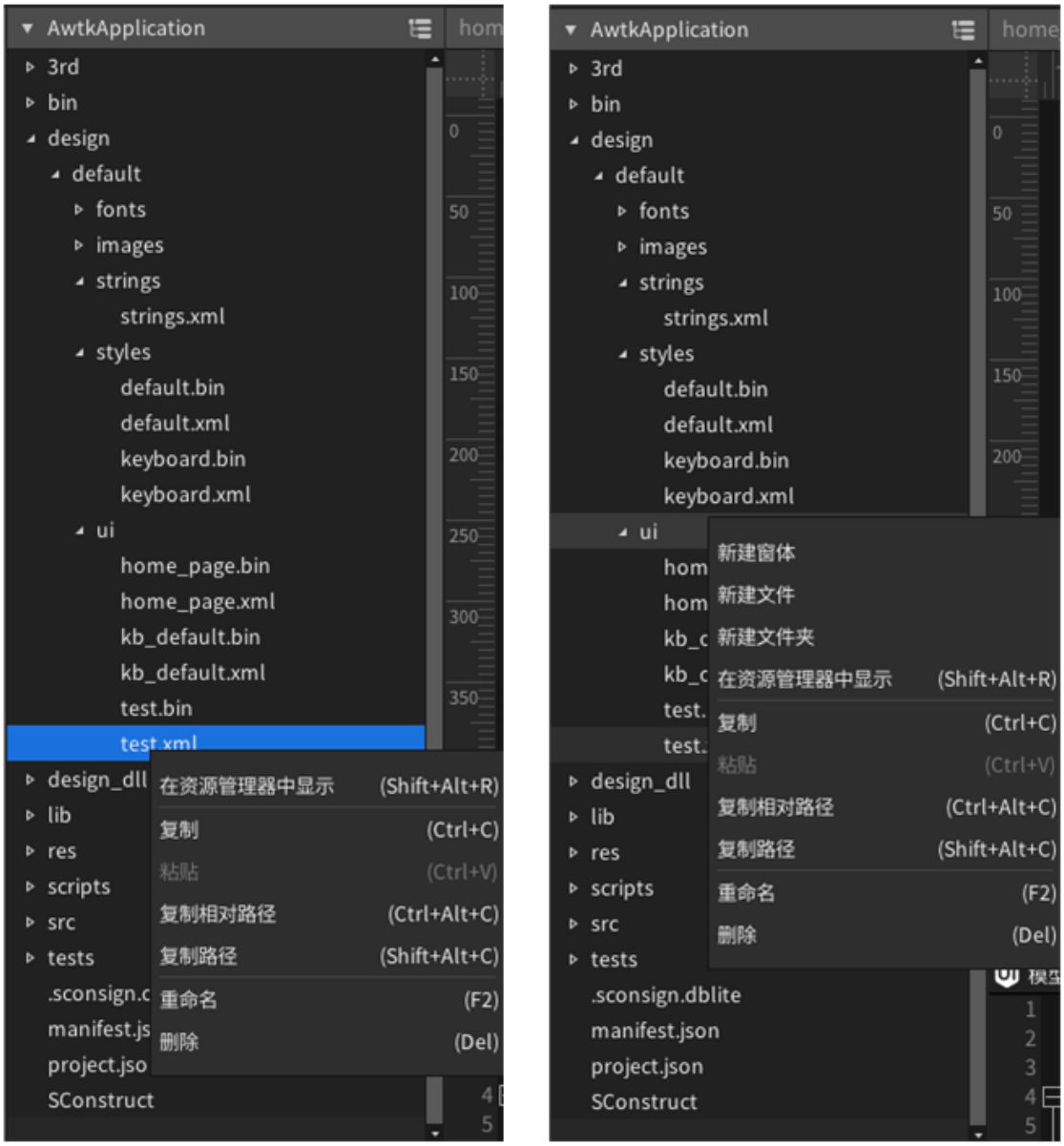


图 2.28 项目管理器窗口的右键菜单

右键点击项目管理器窗口的文件，会打开右键菜单。菜单选项如下表所示。

操作	快捷键	说明
新建窗体	无	可以打开“新建窗体”对话框，当前为 ui 目录时有效。
新建文件	无	新建文件到当前目录。
新建窗体	无	新建文件到当前目录。
在资源管理器中显示	Shift + Alt + R	打开系统的资源管理器，并定位到该文件或文件夹。
复制	Ctrl + C	复制文件或文件夹。
粘贴	Ctrl + V	粘贴文件或文件夹到当前目录，在复制文件或文件夹后有效。
复制相对路径	Ctrl + Alt + C	复制相对于项目目录的路径。
复制路径	Shift + Alt + C	复制绝对路径。

续上表

操作	快捷键	说明
重命名	F2	重命名文件或文件夹。
删除	Delete	删除文件或文件夹。

注: 重命名或者删除主题目录时，会同步修改项目的主题设置；添加、重命名或者删除主题中的字体时，会同步修改项目中对应主题的字体设置。

2.7 控件列表

图 2.29 控件列表窗口

控件列表窗口显示 AWTK 的内置控件、已经安装到项目中的自定义控件。通过拖曳的方式可以把列表中的某个控件添加到窗体编辑器上。

拖曳的方式有 2 种：

- 点击控件（不释放鼠标），拖拽到窗体编辑器后再释放鼠标；
- 单击控件（释放鼠标），移动光标到窗体编辑器上，再次点击但不释放鼠标，拖曳直到适合的控件大小，再释放鼠标。

注: 关于如何导入自定义控件，请参阅 5.1。

2.8 资源浏览器

资源浏览器窗口可以显示项目当前主题正在使用的资源，并提供一些编辑资源的操作。

2.8.1 图片资源

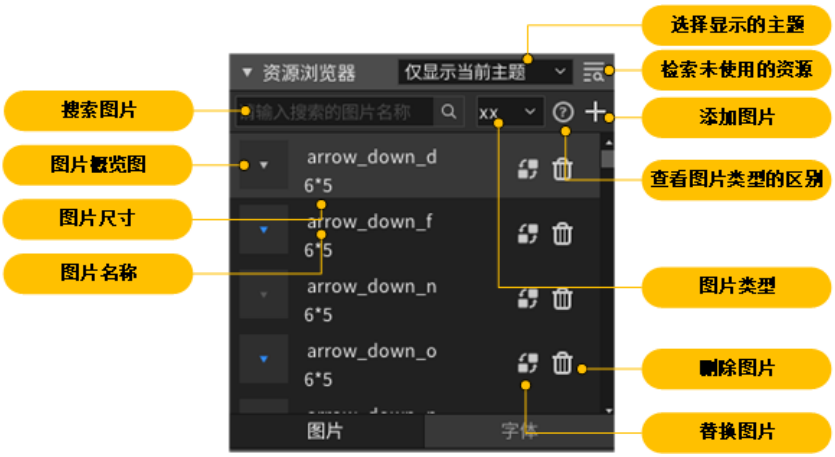


图 2.30 图片资源列表

点击资源浏览器窗口的”图片”按钮，可以切换到图片资源列表。列表上显示图片的概览图、名称、尺寸，通过该列表可以进行以下操作：

(1) 选择显示的主题：

- 仅显示当前主题：显示当前主题目录的图片资源；
- 仅显示 default 主题：显示 default 主题目录的图片资源；
- 显示 default+ 当前主题：显示 default + 当前主题目录的图片资源。

(2) 检索未使用的资源：点击”检索”按钮，可以弹出对话框，显示 UI 文件、样式文件未使用的图片、字体，可以选择删除指定的图片、字体。

(3) 搜索图片：在搜索框中输入要搜索的图片的名称，列表将仅显示匹配项。

(4) 切换图片类型：点击图片类型下拉列表，可以修改要显示的图片类型。有如下选项：

- xx：仅显示与 DPR 无关的图片；
- x1：仅显示 DPR=1 时使用的图片；
- x2：仅显示 DPR=2 时使用的图片；
- x3：仅显示 DPR=3 时使用的图片；
- svg：仅显示 svg 图片。

(5) 查看图片类型的区别：点击”?”按钮，会弹出浏览器界面介绍不同图片类型有什么区别。

(6) 添加图片：点击”+”按钮，可以添加图片到主题的 images 目录中当前图片类型的目录，如果新增图片在项目中不存在，则会拷贝到项目中。

(7) 删除图片：点击”删除”按钮，可以删除图片。

(8) 替换图片：点击”替换”按钮，可以用其他图片替换当前图片，并保持文件名不变。

(9) 创建图片控件：点击图片（不释放鼠标），拖拽到窗体编辑器后释放鼠标，可以创建一个显示该图片的 image 控件。

2.8.2 字体资源

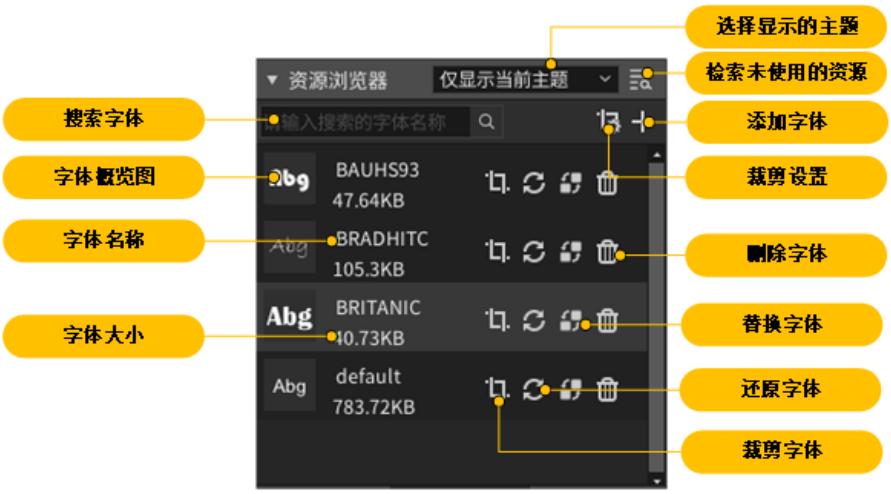


图 2.31 字体浏览器窗口

点击字体 Tab 按钮，可以切换到字体资源列表，列表上显示字体的概览图、名称、文件大小，通过该列表可以进行以下操作。

(1) 选择显示的主题：

- 仅显示当前主题：显示当前主题目录的字体资源；
- 仅显示 default 主题：显示 default 主题目录的字体资源；
- 显示 default+ 当前主题：显示 default + 当前主题目录的字体资源。

(2) 检索未使用的资源：点击”检索”按钮，可以弹出对话框，显示 UI 文件、样式文件未使用的图片、字体，可以选择删除指定的图片、字体。

(3) 搜索字体：在搜索框中输入要搜索的字体的名称，列表将仅显示匹配项。

(4) 修改字体裁剪设置：点击”裁剪设置”按钮，可以打开项目设置页面修改字体的裁剪设置，更多信息请参阅 2.4.2。

(5) 添加字体：点击”+”按钮，可以添加字体到主题的 fonts 目录，如果新增字体在项目中不存在，则会拷贝到项目中。

(6) 删除字体：点击”删除”按钮，可以删除字体。

(7) 裁剪字体的 TTF 文件：点击”裁剪字体”按钮，可以裁剪 TTF 文件，裁剪后文件将仅保留设置的字符，原字体文件将移动到 origin 目录；裁剪后非保留字符将无法正常显示。

(8) 还原字体：点击”还原字体”按钮，可以将已被裁剪的字体文件还原成未被裁剪的文件。

(9) 替换字体：点击”替换”按钮，可以用选定的 TTF 文件进行替换，但不改变名称。

2.9 对象浏览器

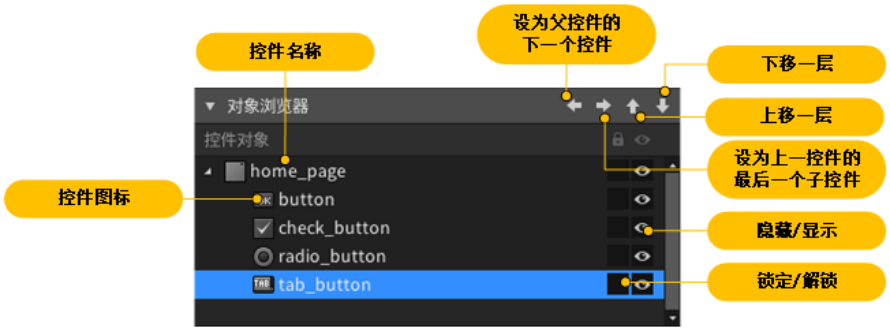


图 2.32 对象浏览器窗口

对象浏览器窗口可以显示正在编辑的 UI 文件的控件树，同时提供以下操作：

- 隐藏/显示控件：点击“隐藏/显示”按钮，可以设置控件是否在编辑器上可见，显示时显示眼睛图标，隐藏后在编辑器上无法选中该控件。
- 锁定/解锁控件：点击“锁定/解锁”按钮，可以锁定或者解锁控件，锁定时显示锁图标，锁定后在编辑器以及对象浏览器上无法选中该控件。
- 放到前面：点击”↑”按钮，可以使选中控件的显示位置上移一层。
- 放到后面：点击”↓”按钮，可以使选中控件的显示位置下移一层。
- 设为父控件的下一个控件：点击”←”按钮，可使选中控件变为父控件的下一个控件。
- 设为上一控件的最后一个子控件：点击”→”按钮，可以使选中控件变为其上一个控件的最后一个子控件。
- 编辑控件名称：双击控件名称，会弹出编辑框，可直接修改控件名称。
- 拖曳修改 Z 方向顺序：点击控件但不释放鼠标，拖拽可直接修改控件的所在层次。

当控件属性设置异常时，控件右侧会显示黄色警告图标。当鼠标悬停在图标上时会显示详细的异常信息，如下图所示。

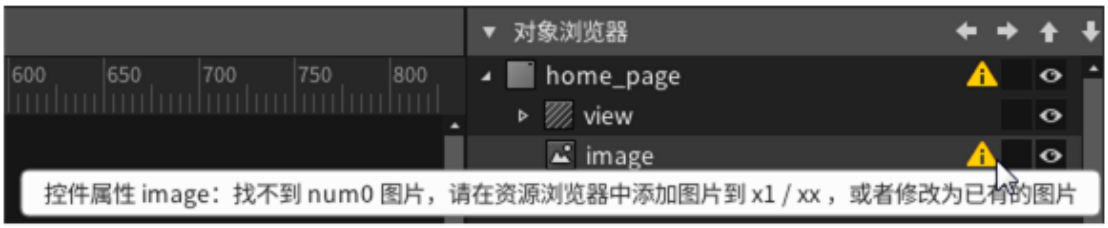


图 2.33 对象浏览器上控件的异常提示

2.10 控件属性编辑器

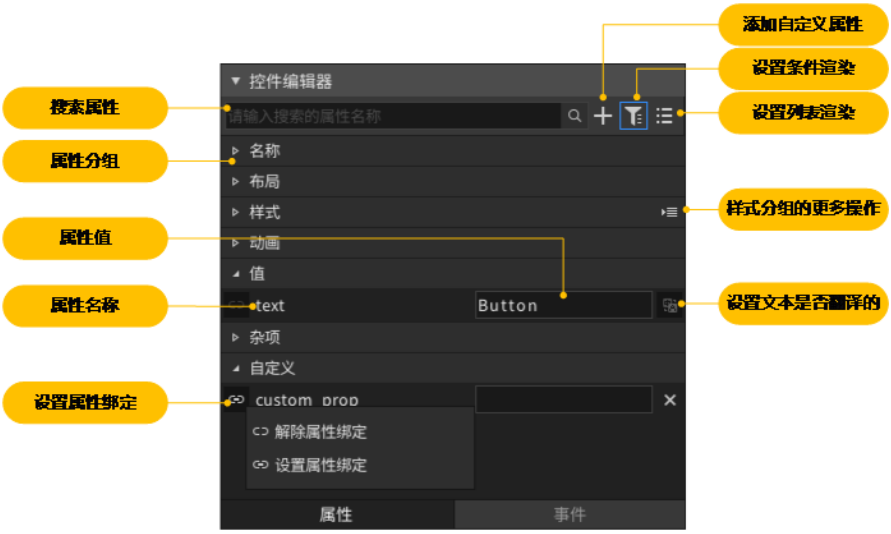


图 2.34 控件属性编辑器窗口

通过控件属性编辑器可以修改被选中的控件的属性值，修改后在窗体编辑器上实时生效。不同的控件有不同的属性。以功能进行划分，可以分为如下表所示的分组。

项目	说明
名称	控件的名称和类型。
布局	控件及其子控件的布局。
样式	控件的外观，比如背景颜色、边框颜色等。
动画	控件的动画，主要指控件进入/退出窗口或强调时的动画。仅非 window 类型的控件有效。
值	控件的显示值，比如 value、text 等。
窗口过渡动画	窗口打开或关闭时的过渡动画，比如平移动画、弹出动画等。仅窗体类型的控件有效。
窗口子控件焦点	窗口内部的子控件的焦点切换方式。仅窗体类型的控件有效。
杂项	其他属性。
自定义	用户自定义的属性。

注: 这里不对控件的属性做详细介绍，当鼠标移动到属性名称上方时可以显示该属性的详细描述。关于控件的更多信息，请参阅《AWTK-API.chm》的 manual 部分。

2.10.1 设置控件样式

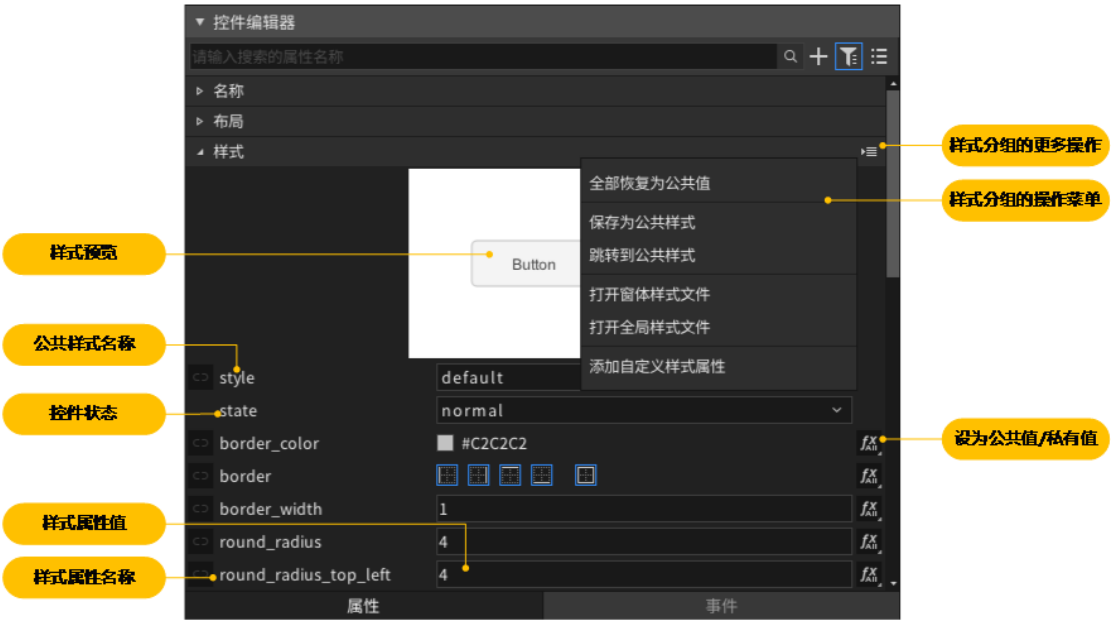


图 2.35 控件属性的样式分组

点击展开控件属性的样式分组，会显示控件 state 状态时的样式预览，以及样式属性列表。通过样式分组，可以进行以下操作：

(1) 设置要使用的公共样式：点击 style 属性右侧的下拉列表，可以设置该控件要使用的公共样式，修改公共样式不会影响控件私有的样式属性。

(2) 切换控件的状态：点击 state 属性右侧的下拉列表，可以切换当前编辑的控件状态。

(3) 设置样式属性：直接设置样式属性的值即可，设置后该属性将变为控件私有的。

(4) 设置样式属性为私有值：可以把当前属性设为控件私有的，此时控件的 XML 上会有该属性的定义。有 3 种方式：

- 点击样式属性值右侧的”fx all”按钮；
- 长按或者右键点击”fx all”按钮，再点击“设为私有值”；
- 长按或者右键点击”fx all”按钮，再点击“设为私有值并应用到全部状态”，应用到所有状态会使该样式属性在所有控件状态下都等于该私有值。

(5) 恢复样式属性为公共值：可以把当前属性恢复为公共值，即默认为公共样式中对应的属性值，同时会删除控件的 XML 上对应的属性定义。有 3 种方式：

- 点击样式属性值右侧的”fx”按钮；
- 长按或者右键点击”fx”按钮，之后点击“恢复为公共值”；
- 长按或者右键点击”fx”按钮，之后点击“恢复为公共值并应用到全部状态”，应用到所有状态会使该样式属性在所有控件状态下都等于公共值。

(6) 恢复全部样式属性为公共值：点击样式分组栏上方的按钮，之后点击“全部恢复为公共值”，可以把控件的所有样式属性恢复为公共的样式属性值。

(7) 保存为公共样式：点击样式分组栏目右侧的按钮，之后点击“保存为公共样式”：在

弹出的对话框中设置好目标样式文件、样式名称后，点击“保存但使用新的样式”即可；如果需要保留控件私有的样式属性，则可以选择点击“保存但保留原样式”。

(8) 跳转到公共样式：点击样式分组栏目右侧的按钮，之后点击“跳转到公共样式”，如果公共样式存在，则可以打开公共样式所属的样式文件，并定位到该样式。

(9) 打开窗体样式文件：点击样式分组栏目右侧的按钮，之后点击“打开窗体样式文件”，如果窗体样式文件存在，则可以打开该文件。

(10) 打开全局样式文件：点击样式分组栏目右侧的按钮，之后点击“打开全局样式文件”，如果全局样式文件存在，则可以打开该文件。

(11) 添加自定义样式属性：点击样式分组栏目右侧的按钮，之后点击“添加自定义样式属性”，在弹出的对话框中设置好属性所属的控件状态、名称后点击“确定”即可。

需要注意的是：

1. AWTK 优先使用控件私有的样式属性绘制控件，其次使用公共的样式属性。
2. 控件私有的样式属性，指控件自身的以”style:”开头的属性，在 UI 文件的 XML 描述中有对应的属性定义，比如 style:normal:bg_color。
3. 公共的样式属性，指在样式文件中定义的样式，通过控件的 style 属性来指定控件当前使用的样式。
4. AWTK 优先使用窗体样式文件，其次为全局样式文件。
5. 窗体样式文件，指以窗体的 theme 属性（如果 theme 属性没有设置则使用窗口的 name 属性）为文件名的样式文件。
6. 全局样式文件，指缺省样式文件，固定命名为 default.xml。

2.10.2 设置控件布局

图 2.36 控件属性的布局分组

点击展开控件属性的布局分组，会显示布局属性列表。通过布局分组可以进行如下操作。

1. 设置控件自身的布局

x、y、w、h 控制控件自身的布局，有 3 种设置方式。

- 像素方式：直接指定像素值，这是缺省方式，也是最不灵活的方式。
- 百分比方式：
  - x、w 的值如果包含“%”，则自动换算成相对于其父控件宽度的百分比；
  - y、h 的值如果包含“%”，则自动换算成相对于其父控件高度的百分比。
- 相对方式：
  - x 的值设置成“c:-10”，可以使控件在水平方向上居中并向左偏移 10 像素；
  - x 的值设置成“r:10”，可以使控件位于其父控件的右侧并向左偏移 10 像素；
  - y 的值设置成“m”，可以使控件在垂直方向上居中；
  - y 的值设置成“b”，可以使控件位于其父控件的底部。

注：像素方式与百分比方式的宽度和高度可以是负数；宽度为负数，其实际值为父控件的宽度 + 该负值；高度为负数，其实际值为父控件的高度 + 该负值。

点击旁边的“切换为百分比”按钮可以将当前数值切换为百分比形式显示，点击“设为属性变化带动画”按钮，可以将该属性变为变化时带动画效果，而不是瞬间变化。

2. 设置子控件的布局

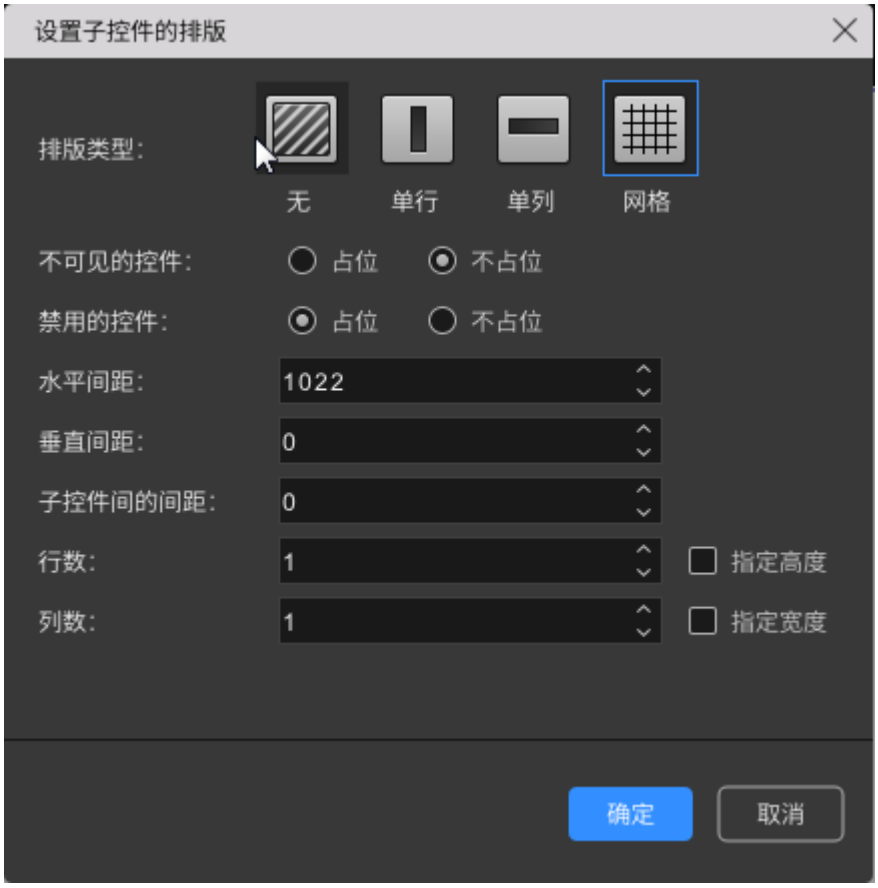


图 2.37 设置子控件的排版对话框

children_layout 属性控制子控件的布局。点击 children_layout 属性右侧的编辑框，可以打开”设置子控件的排版”对话框；选择一种排版类型并设置好参数后，点击”确定”即可看到排版效果。

Designer 目前支持的子控件排版及其参数如下：

- 无：表示不使用子控件布局。
- 单行：表示单行布局，所有子控件在水平方向上排成一行。需要设置的参数有：
  - 不可见的控件：是否给不可见的控件留位置；
  - 禁用的控件：是否给禁用的控件留位置；
  - 水平间距：水平方向上的边距，单位为像素；
  - 垂直间距：垂直方向上的边距，单位为像素；
  - 子控件间的间距：子控件之间的间距，单位为像素。
  - 水平对齐的方式：子控件整体的水平对齐方式。
- 单列：表示单列布局，所有子控件在垂直方向上排成一列。需要设置的参数有：
  - 不可见的控件、禁用的控件、水平间距、垂直间距、子控件间的间距；
- 网格：表示网格布局，所有子控件放在 M*N 的网格中。需要设置的参数有：
  - 不可见的控件、禁用的控件、水平间距、垂直间距、子控件间的间距；
  - 行数/高度：当指定高度时，表示一行的固定高度，单位为像素；否则为行数。
  - 列数/宽度：当指定宽度时，表示一列的固定宽度，单位为像素；否则为行数。
- 列表：表示列表布局，仅对 list_view 控件下的 scroll_view 控件有效。需要设置的参数有：
  - 不可见的控件、禁用的控件、水平间距、垂直间距、子控件间的间距；
  - 默认行高：作用与 list_view 的 default_item_height 属性相同。
  - 行高：作用与 list_view 的 item_height 属性相同，如果行高未指定则使用默认行高。
  - 列数：列表的列数。

2.10.3 设置控件动画

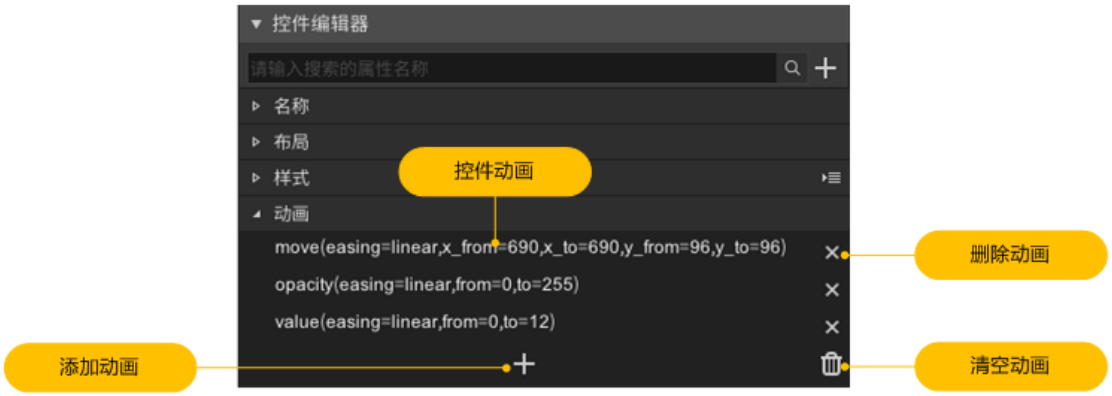


图 2.38 控件属性的动画分组

点击展开控件属性的动画分组，会显示控件动画列表，默认没有动画。通过动画分组，可以进行以下操作：

- 添加动画：点击”+”按钮，可以打开如下图所示的“设置动画”对话框（不同的控件支持的动画类型略有不同），设置动画参数后点击“确定”即可新建一个控件动画。
- 删除动画：点击控件动画右侧的”×”按钮，可以删除该动画。
- 编辑动画：双击控件动画，可以在弹出的“设置动画”对话框中重新设置动画参数。
- 清空动画：点击”清空”按钮，可以删除设置的所有控件动画。

图 2.39　设置动画对话框

Designer 目前支持的控件动画及其参数如下：

- 无：表示无动画。
- 位移：表示位移动画。需要设置的参数有：
  - 名称：动画的名称，在代码中可以通过动画名称控制动画启动、暂停等。

  - 表示位移动画。需要设置的参数有：
  - 名称：动画的名称，在代码中可以通过动画名称控制动画启动、暂停等。
  - 延迟启动时间：动画启动后可以延迟一段时间才真正启动，单位为 ms。
  - 单次动画持续时间：动画启动后一次变化的持续时间，单位为 ms。
  - 时倍率：动画的时间倍率，可以按比例改变真实时间，形成时间变快和变慢的效果。
  - 单次动画趋势：按设定插值算法模拟一次动画的变化过程，形成匀速、加速、减速等变化效果。
  - 往返次数：动画往返的次数（一往一返为 1 次），默认为禁用，为 0 时表示永久播放。
  - 重复次数：动画重复执行的次数，默认为禁用，为 0 时表示永久播放。
  - 动画创建后自动播放：勾选则表示运行时动画对象被创建就立即启动。
  - 动画结束时自动销毁：勾选则表示按设定的逻辑执行完毕后自动销毁动画对象。
  - x 方向：单次动画中控件 x 方向位置的起始值和结束值。
  - y 方向：单次动画中控件 y 方向位置的起始值和结束值。
- 值：表示值动画。需要设置的参数有：
  - 公共参数；
  - 值：单次动画中控件值的起始值和结束值。
- 透明度：表示透明度动画。需要设置的参数有：
  - 公共参数；
  - 透明度：单次动画中控件透明度的起始值和结束值。
- 缩放：表示缩放动画。需要设置的参数有：
  - 公共参数；
  - x 方向：单次动画中控件 x 方向缩放比例的起始值和结束值；
  - y 方向：单次动画中控件 y 方向缩放比例的起始值和结束值。
- 旋转：表示旋转动画。需要设置的参数有：
  - 公共参数；
  - 旋转：单次动画中控件旋转角度的起始值和结束值，单位为弧度。
- 属性：表示属性动画。需要设置的参数有：
  - 公共参数；
  - 属性名称：动画过程中需要改变的控件属性的名称；
  - 属性值：单次动画中控件目标属性的起始值和结束值。

需要注意的是：

1. 公共参数指名称、延迟启动时间、单次动画持续时间、时倍率、单次动画趋势、往返次数、重复次数、动画创建后自动播放、动画结束时自动销毁。
2. 目前仅 image、gif、svg、mutable_image 等图片类型的控件支持缩放动画和旋转动画。

2.10.4 设置窗口过渡动画

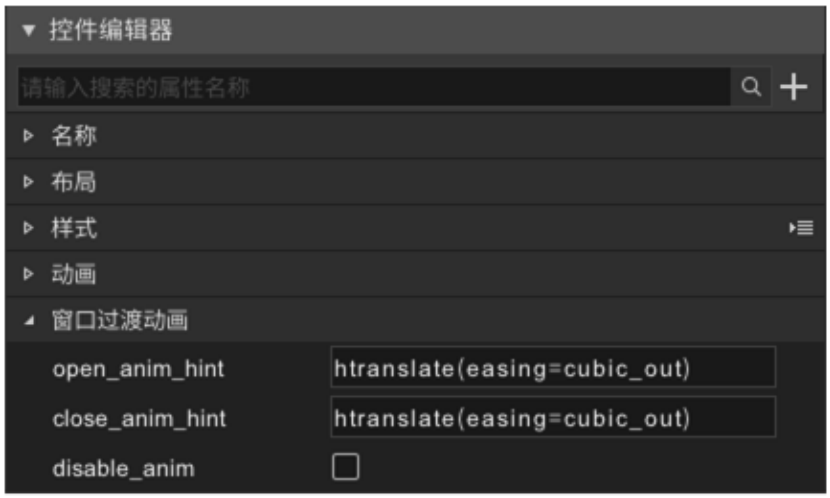


图 2.40 控件属性的窗口过渡动画分组

注: 窗口需要 Ctrl+ 鼠标点击窗口的空白处或者在对象浏览器上点击才能被选中。

点击展开控件属性的窗口过渡动画分组，会显示与过渡动画相关的属性。通过该分组，可以进行以下操作：

- 设置窗口打开动画：点击 open_anim_hint 属性右侧的编辑框，可以打开“设置窗口动画”对话框，设置好动画参数后，点击“确定”即可。
- 设置窗口关闭动画：点击 close_anim_hint 属性右侧的编辑框，可以打开“设置窗口动画”对话框，设置好动画参数后，点击“确定”即可。
- 使能窗口动画：点击 disable_anim 属性右侧的勾选按钮。

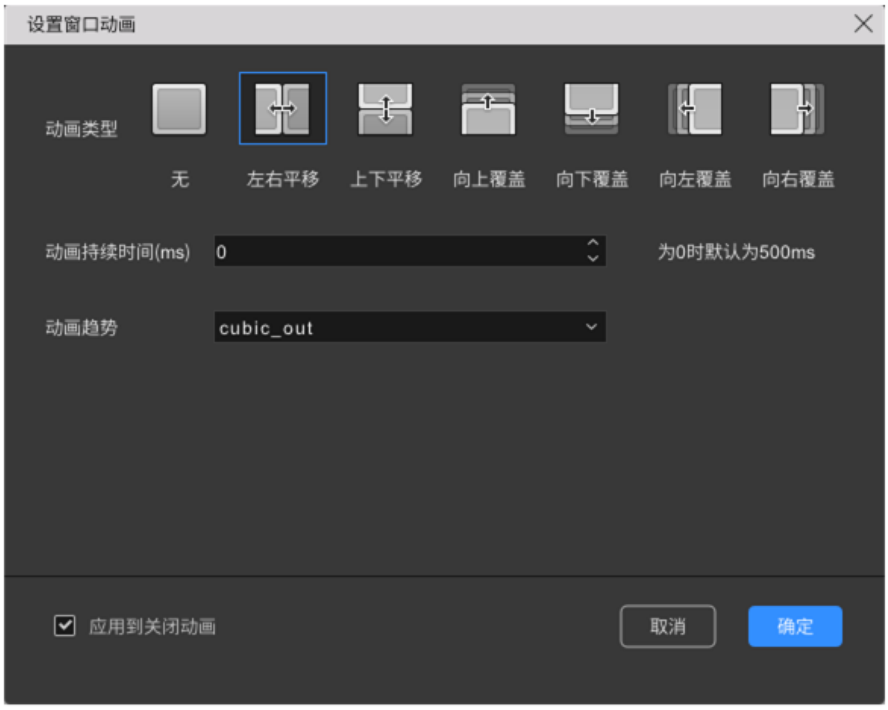


图 2.41 设置窗口动画对话框

Designer 目前支持的窗口动画及其参数如下：

(1) 无：表示无动画。

(2) 平移：表示平移动画，即前后两个窗口同时向一个方向平移，适用于非 dialog 类型的窗口，有 2 种：

- 左右平移：打开窗口时从右往左平移，关闭窗口时从左往右平移；
- 上下平移：打开窗口时从下往上平移，关闭窗口时从下往上平移。

需要设置的参数有：

- 动画持续时间：窗口动画的持续时间，单位为 ms；
- 动画趋势：按设定插值算法模拟动画的变化过程，形成匀速、加速、减速等变化效果。

(3) 覆盖：表示覆盖动画，即后一个窗口覆盖前一个窗口，适用于非 dialog 类型的窗口，有 4 种：

- 向上覆盖：打开窗口时新窗口从下往上覆盖，关闭窗口时当前窗口从上往下退出；
- 向下覆盖：打开窗口时新窗口从上往下覆盖，关闭窗口时当前窗口从下往上退出；
- 向左覆盖：打开窗口时新窗口从右往左覆盖，关闭窗口时当前窗口从左往右退出；
- 向右覆盖：打开窗口时新窗口从左往右覆盖，关闭窗口时当前窗口从右往左退出。

需要设置的参数有：

- 动画持续时间、动画趋势
- 透明度：前一个窗口透明度的起始值和结束值，可以实现背景动态变暗或变亮的效果。

(4) 弹出：表示弹出动画，即新窗口向一个方向弹出，适用于 dialog 类型的窗口，有 2 种：

- 向上弹出：打开窗口时新窗口从下往上弹出，关闭窗口时当前窗口从上往下退出；
- 向下弹出：打开窗口时新窗口从上往下弹出，关闭窗口时当前窗口从下往上退出。

需要设置的参数有：动画持续时间、动画趋势

(5) 淡入淡出：表示淡入淡出动画，适用于 dialog 类型的窗口。需要设置的参数有：动画持续时间、动画趋势。

(6) 缩放：表示缩放动画，适用于 dialog 类型的窗口。需要设置的参数有：动画持续时间、动画趋势。

2.10.5 添加自定义属性

图 2.42 添加自定义属性

点击”控件编辑器”右侧的”+“按钮，可以打开”添加自定义属性”对话框；设置好自定义属性的名称和值之后，点击”确定”，会将新增的属性加入到控件属性编辑器的自定义分组。

需要注意的是：

1. 属性名称不能为控件现有的属性或者”style:”开头的样式属性。
2. 如果需要添加自定义的样式属性，可以通过样式分组的菜单进行添加，更多信息请参阅 2.10.1。

2.10.6 异常提示

如下图所示，当控件某个属性设置有异常时，属性所在分组及该属性的编辑框会用黄色边框高亮显示。当焦点进入编辑框时，会显示详细的异常信息。

图 2.43 控件属性编辑器上属性的异常提示

2.11 控件事件编辑器

图 2.44 控件事件编辑器窗口

如上图所示，选中控件后，通过控件编辑器的”事件”分页，可以设置选中控件的事件绑定，具体的操作如下：

(1) 添加事件绑定：点击搜索框右侧的”+”按钮，可以打开当前控件支持的事件类型；点击指定的事件类型，可以添加一个默认的事件绑定；一个事件类型只能有一个事件绑定。

(2) 删除事件绑定：点击事件绑定右侧的”删除”按钮，可以删除该绑定。

(3) 修改事件绑定：展开事件绑定，可以修改事件绑定编辑，包括：

- 动作：表示事件触发后要执行的动作，不同的动作会有不同的参数；
- 动作结束后是否退出程序；
- 动作结束后是否同步数据到模型；
- 动作结束后是否关闭当前窗口。

事件绑定中支持的动作如下：

(1) 打开窗口：用于打开新窗口。可以设置如下参数：

- 窗口的名称：点击右侧的下拉框，可以选择项目中已有的窗口；
- 切换到已存在的窗口：勾选该参数时，会切换到当前已打开的窗口；
- 切换时关闭当前窗口：勾选该参数时，打开窗口时会关闭当前窗口。

(2) 关闭窗口：用于关闭当前窗口。

(3) 执行回调函数：用来绑定事件回调函数。可以设置如下参数：

- 函数名：事件回调函数的名称。

(4) 执行 Fscript 代码：用于执行一段 fscript 脚本。点击“FScript 代码”右侧的输入框，可以打开“设置绑定规则”对话框，编辑 FScript 脚本。

注: 关于 Fscript 脚本的更多信息，请参阅 SDK/awtk/docs/fscript.md 文档。

2.12 窗体编辑器

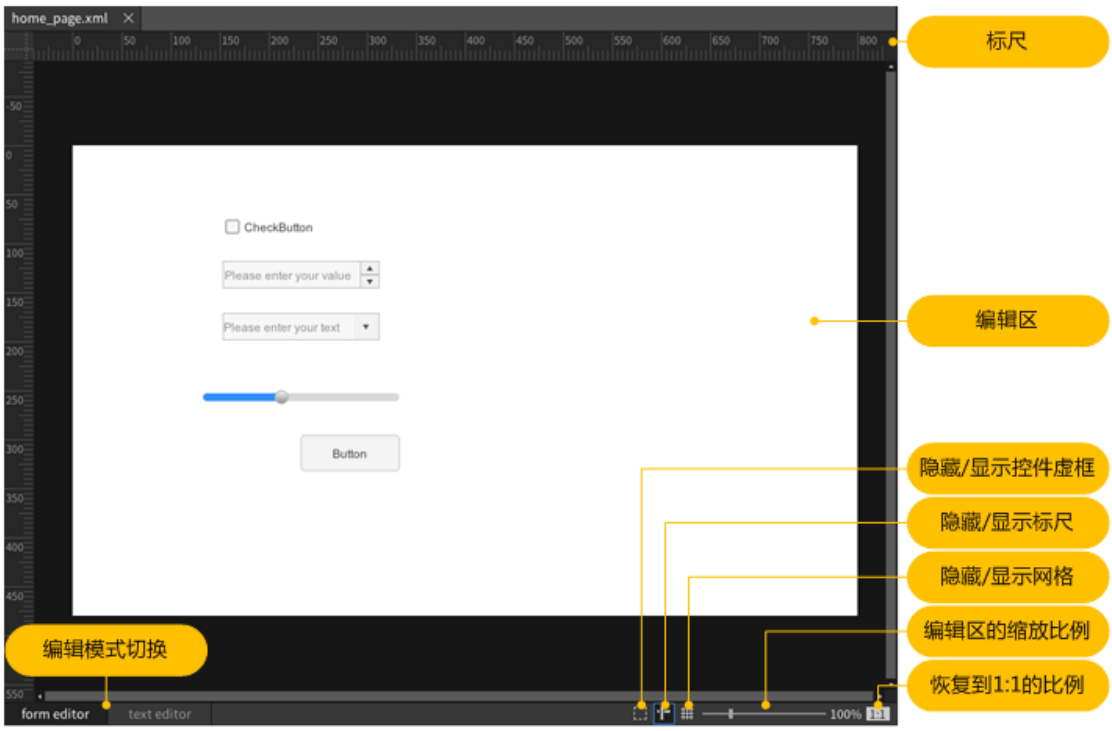


图 2.45 窗体编辑器

Designer 默认使用窗体编辑器打开 UI 文件。窗体编辑器是一个图形编辑器，可以通过拖拽的方式在窗体上添加控件，此外，还可以改变控件位置、大小、层次等，所见即所得。

2.12.1 添加、删除控件

图 2.46 添加一个按钮

在编辑区左侧的控件列表中选中需要的控件并拖拽到窗体上即可添加控件。如果要删除不需要的控件，则选中控件后按 Delete 键即可。

注: 关于控件列表的更多信息，请参阅 2.7。

2.12.2 选中控件

可以通过如下方式来选中编辑区中的控件：

- 点击控件：直接用鼠标点击可以选中控件，但一次仅能选中一个。如果需要多选，则可以用 Ctrl + 鼠标点击的方式。
- 框选：点击编辑区的空白处但不释放鼠标，拖动后会有一个半透框，鼠标释放时被框住的控件均会被选中，如下图所示；如果需要同时选中其他区域的控件，可以按住 Shift 键后再次框选。

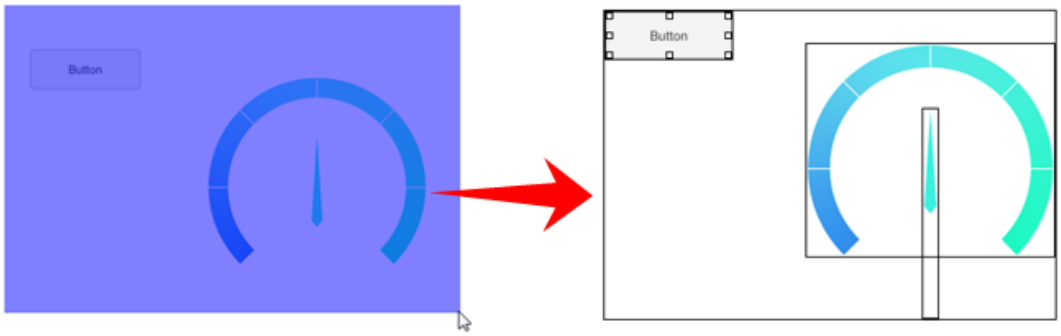


图 2.47 框选控件

- 在对象浏览器上点击控件：关于对象浏览器的更多信息，请参阅 2.9。

注: 对于窗体本身，只能通过 Ctrl+ 鼠标点击或者在对象浏览器上点击的方式选中。

2.12.3 改变控件的大小

选中控件后，可以通过如下方式改变控件的大小：

- 鼠标拖拽

控件被选中后会被有一个带 8 个小正方形的方框框住，将鼠标光标移到小正方形上后按下鼠标并拖拽即可改变控件的大小，如下图所示；如果拖拽的同时按下 Shift 键，则可以锁定纵横比例。

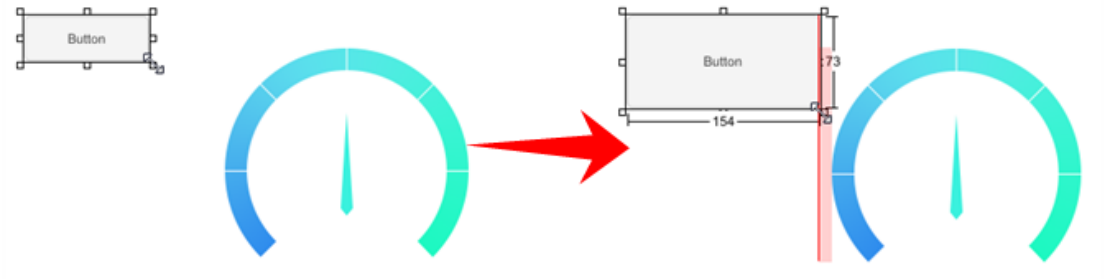


图 2.48 改变控件的大小

- 在控件属性编辑器上修改控件属性

更多信息请参阅 2.10.2 章节。

2.12.4 改变控件的位置

选中控件后，可以通过如下方式改变控件的位置：

- 鼠标拖拽

点击控件后不释放鼠标，直接拖拽即可，这是最简单的方式。如果拖拽的同时按下 Shift 键，则可以将控件限制在水平或垂直方向上。如果拖曳的同时按下 Alt 键，则控件可以按网格一小格一小格地移动。

- 键盘方向键微调

按下键盘方向键，控件可以 1 像素 1 像素地移动。如果同时按下 Alt 键，则控件可以按网格一小格一小格地移动。

- 工具栏上的对齐、分布、顺序等操作

更多信息请参阅 2.3.2 章节。

- 对象浏览器上的控件层次操作

更多信息请参阅 2.9 章节。

- 在控件属性编辑器上修改控件属性

更多信息请参阅 2.10.2 章节。

2.12.5 编辑显示文本

当鼠标停留在部分文本控件上方时，控件右侧会显示一个”编辑”按钮，点击可直接编辑当前显示的文本，如下图所示。

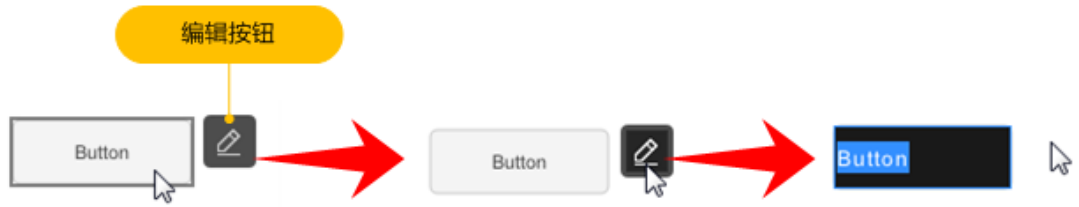


图 2.49 编辑按钮上的文本

注：

1. 双击控件也可以触发上述编辑操作。
2. 此类控件有 button、check_button、radio_button、tab_button、edit、spin_box、label。

2.12.6 编辑容器控件

容器控件内部可以容纳任意控件，但不同类型的容器有不同的特性，因此有不同的编辑方式。

- 通用容器：仅提供简单的收纳功能。

如下图所示，当控件被拖曳到此类容器上方时，可以通过按空格键将其放入容器内部；容器内部的子控件仍可以用鼠标点击选中。

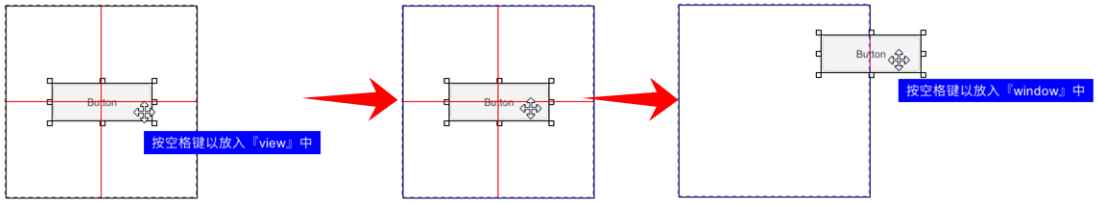


图 2.50 编辑通用容器

注：此类容器有各类窗体、button、edit、combo_box、image、label、view、list_item、list_view、list_view_h、color_picker、tab_control。

- 裁剪容器：该容器会裁剪超出容器范围的子控件，容器的子控件无法直接用鼠标选中。

如下图所示，当鼠标停留在此类容器控件上方时，控件右侧会显示一个”编辑”按钮，点击可进入容器内部编辑其子控件；进入容器内部后，可以通过点击窗体编辑器下方的”←”按钮返回。

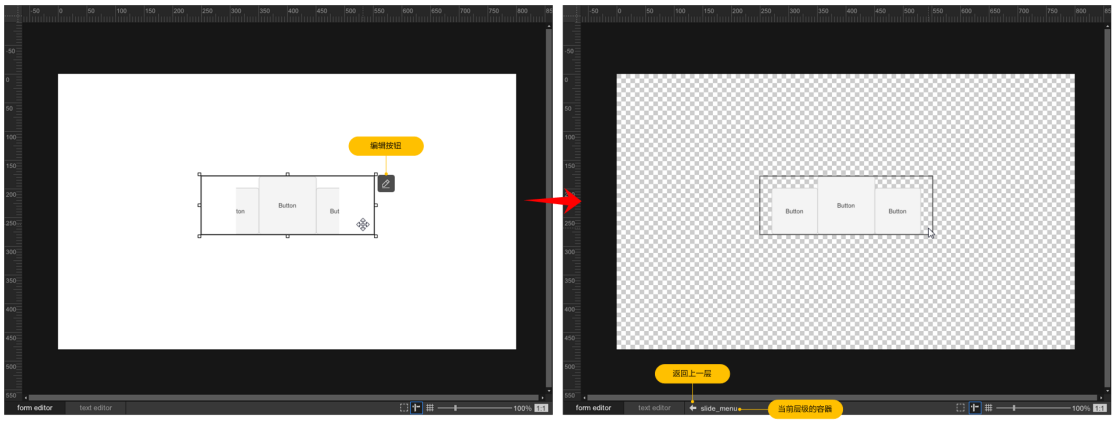


图 2.51 编辑裁剪容器

注：鼠标双击裁剪容器控件，也可以进入容器内部进行编辑。此类容器有 scroll_view、slide_menu、tab_button_group、clip_view。

- 多页容器

多页容器内部可以有多个页面，但同时只能显示一个。

如下图所示，当鼠标停留在此类容器控件上方时，控件右侧会显示一个页面列表。通过该列表，可以进行以下操作：

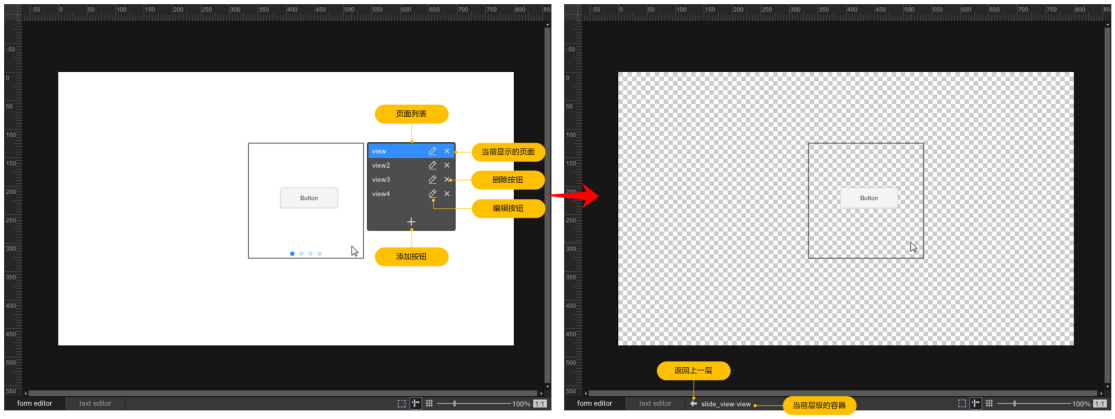


图 2.52　编辑内部有多个页面的容器

- 进入页面内部：点击页面右侧的”编辑”按钮，可以页面内部进行编辑；进入页面内部后，可以通过点击窗体编辑器下方的”←”按钮返回。
- 删除页面：点击页面右侧的”×”按钮，可以删除页面。
- 添加页面：点击”+”按钮，可以添加新的页面。
- 切换当前显示页：双击页面项，可以切换容器当前显示的页面。

注: 鼠标双击容器，可以进入当前页面内部。此类容器有 slide_view、pages。

2.12.7　辅助功能

1.　标尺

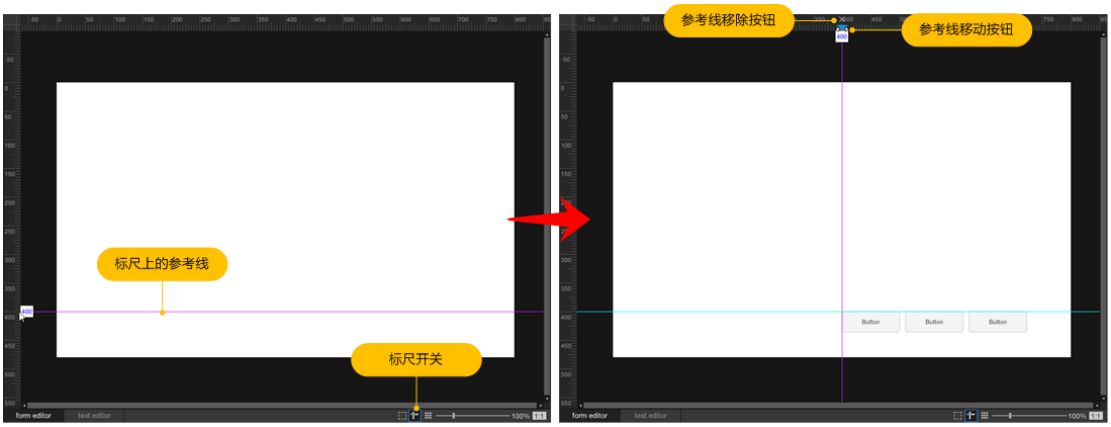


图 2.53　窗体编辑器上的标尺

如上图所示，通过点击编辑器下方的”标尺”按钮，可以启用或禁用标尺。通过标尺，可以进行以下操作：

- 显示鼠标的实时位置：当鼠标在编辑区内移动时，标尺上会显示光标的实时位置。
- 添加参考线：移动鼠标到标尺上，会出现一条跟随鼠标移动并贯穿编辑区的红线，此时点击可以添加一条参考线。
- 改变参考线位置: 移动鼠标到标尺上参考线的位置，会出现一个倒三角，点击并拖拽可以改变参考线的位置。

- 删除参考线：移动鼠标到标尺上参考线的位置，会出现一个”×”按钮，点击可以删除参考线。
- 清空参考线：右键点击标尺，可以弹出右键菜单，再点击”清空对齐线”，可以删除该标尺上添加的的所有参考线。

2. 网格

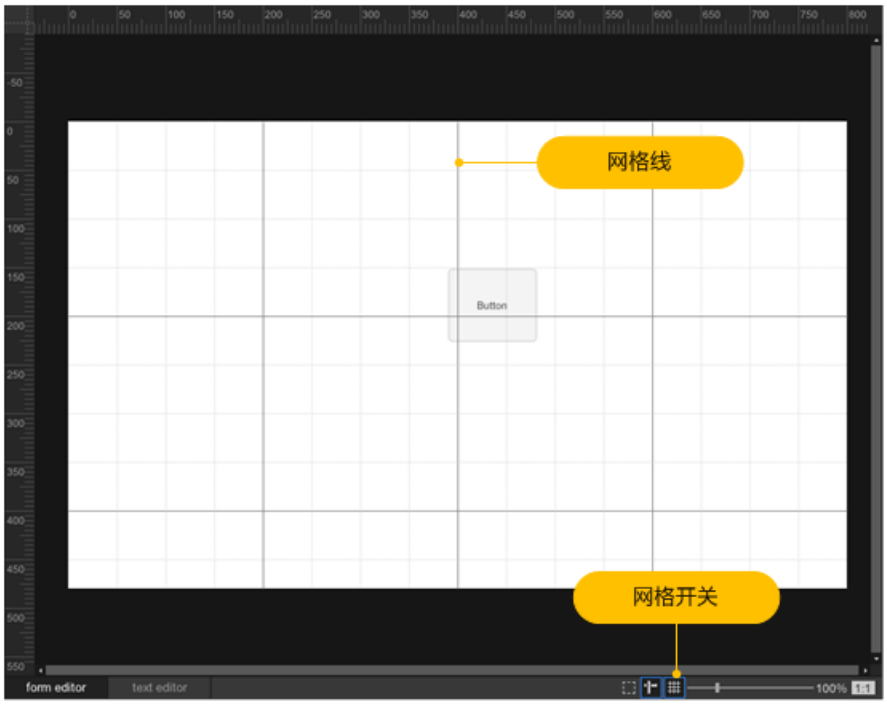


图 2.54 窗体编辑器上的网格

如上图所示，通过点击编辑器下方的”网格”按钮，可以启用或禁用网格。启用网格后，在编辑区上会出现网格。网格线可以用于辅助对齐的参考线。

3. 智能参考线

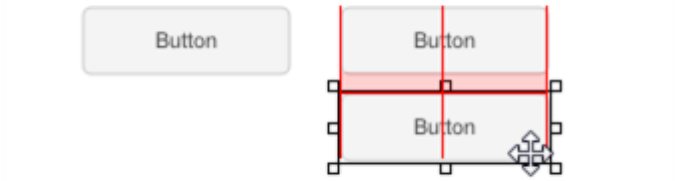


图 2.55 窗体编辑器中的智能参考线

如上图所示，智能参考线是在拖拽或调整控件大小的过程中会出现的红线。当控件与邻近控件的中心点、边缘对齐，或者与窗口的中心点、边缘对齐，或者居于两个控件的中间或相同距离时，会出现一条很直观的对齐辅助线并自动贴齐。

4. 缩放编辑区的显示比例

图 2.56 窗体编辑器中画布的缩放

如上图所示，点击并拖动窗体编辑器下方的缩放比例尺上的滑块，可以缩放编辑区的显示比例；点击”1:1”按钮，可以快速恢复到 100%。

注: 将鼠标光标移动到编辑区上，按住 Alt 键的同时滚动鼠标滚轮，也可以进行缩放。

5. 查看文件的文本内容

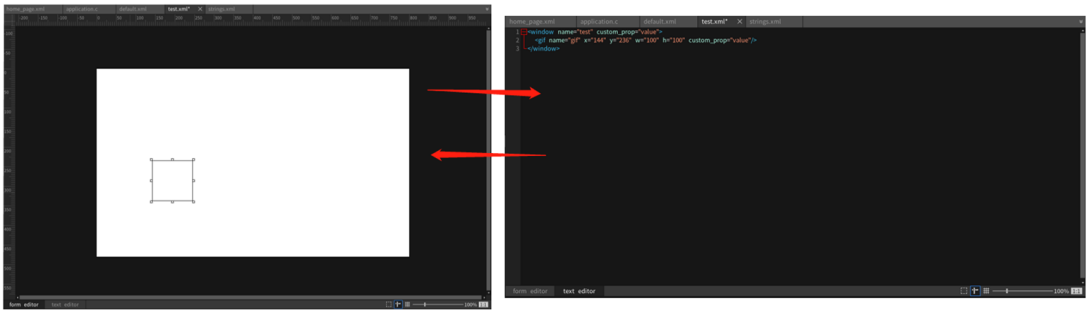


图 2.57 窗体编辑器的编辑模式

如上图所示，点击窗体编辑器的下方的编辑模式按钮可以切换编辑模式，有”form editor”、”text editor”两个选项，详情如下：

- form editor：图形编辑器，以可视化图形方式对文件进行编辑。
- text editor：文本编辑器，以文本方式对文件进行编辑，目前仅为只读。

6. 显示控件的虚线框

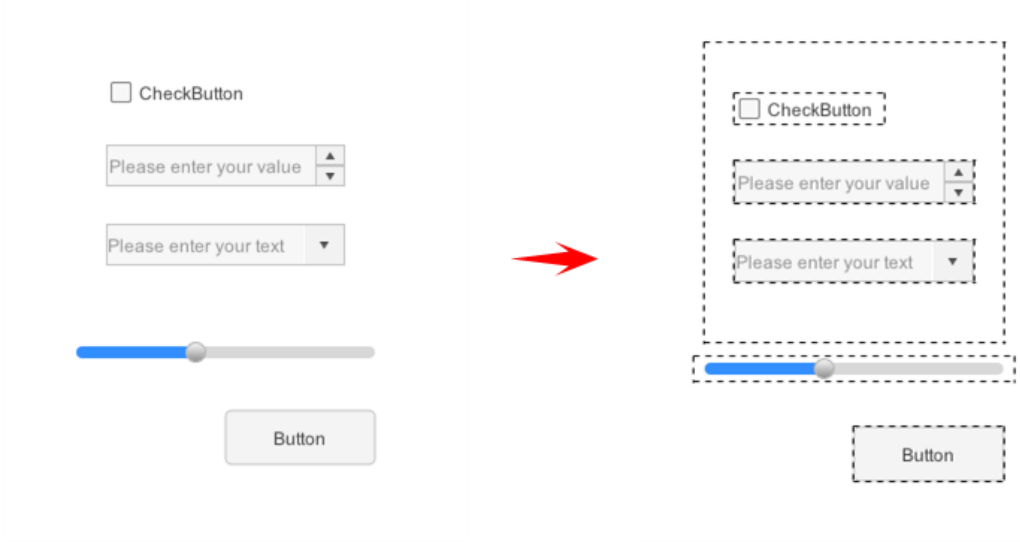


图 2.58 显示控件虚框

如上图所示，点击窗体编辑器下方的”虚线框”按钮，可以显示或隐藏控件框住控件的虚线框。该虚线框表示控件的矩形区域。

2.12.8 键盘快捷键

窗体编辑器的键盘快捷键如下：

- 剪切：选中控件后，按下 Ctrl + X，可以剪切控件。
- 复制：选中控件后，按下 Ctrl + C，可以复制控件；按下 Ctrl 并拖拽控件，可以复制控件。
- 粘贴：按下 Ctrl + V，在鼠标位置粘贴上一次剪切或者复制的控件。
- 撤销：按下 Ctrl + Z，可以撤销上一次的编辑操作。
- 恢复：按下 Ctrl + Y，可以恢复上一次的撤销操作。
- 取消编辑：拖拽控件时，按下 Esc，可以取消当前的拖拽操作，使控件回到原来的位置并恢复大小；按下 Ctrl 并拖拽控件进行复制时，按下 Esc，可以取消复制操作。

注：窗体（window、dialog 等）不支持复制。

2.13 样式编辑器

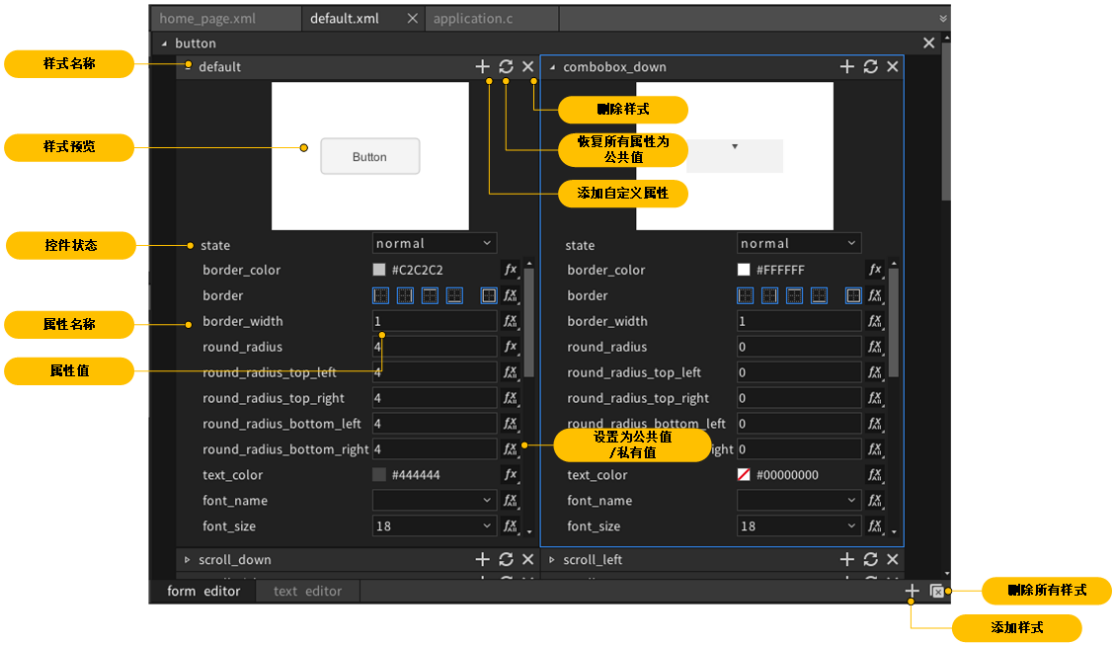


图 2.59 样式编辑器

Designer 默认使用样式编辑器打开样式文件。样式编辑器是一个样式树，可以编辑样式的属性，以及预览样式的显示效果。

注: 样式文件存放在主题目录的 styles 目录。通过项目管理器（请参阅 2.6）或者控件属性编辑器的样式分组（请参阅 2.10.1），可以打开样式文件。

样式编辑器提供以下操作。

(1) 添加样式：点击编辑器右下方的”+”按钮，可以打开“添加样式”对话框，设置好样式类型、名称后点击“确定”即可添加样式。

(2) 删除所有样式：点击编辑器右下方的”×”按钮，会弹出“删除所有样式”弹窗，可以在此删除当前页面展示的所有样式。

(3) 删除样式：展开某个样式，点击右上角的”删除”按钮，可以删除样式。

(4) 编辑样式：展开某个样式，可以直接编辑样式的名称及其包含的所有属性。

(5) 添加自定义样式属性：展开某个样式，点击右上角的”+”按钮，可以添加一个自定义的样式属性。

(6) 恢复全部样式属性为公共值：展开某个样式，点击右上角的”恢复”按钮，可以恢复全部样式属性为缺省值。

(7) 设置样式属性为私有值：可以将当前属性变为该样式的私有属性。有 3 种方式：

- 点击样式属性值右侧的”fx all”;
- 长按或者右键点击”fx all”，再点击“设为私有值”;
- 长按或者右键点击”fx all”，再点击“设为私有值并应用到全部状态”，应用到所有状态会使该样式属性在所有控件状态下都等于该私有值。

(8) 恢复样式属性为公共值可以删除该样式的私有属性，同时把属性值更新为缺省值。

有 3 种方式：

- 点击样式属性值右侧的”fx”；
- 长按或者右键点击”fx”，之后点击“恢复为公共值”；
- 长按或者右键点击”fx”，之后点击“恢复为公共值并应用到全部状态”，应用到所有状态会使该样式属性在所有控件状态下都等于公共值。

(9) 查看文件的文本内容：点击编辑器下方的“form editor”、“text editor”，可以切换编辑模式。

注: 样式的私有属性，指样式自身定义的属性，在样式文件的 XML 描述中有对应的属性定义。

样式编辑器的键盘快捷键如下：

- 剪切：选中样式后，按下 Ctrl + X，可以剪切样式。
- 复制：选中样式后，按下 Ctrl + C，可以复制样式。
- 粘贴：按下 Ctrl + V，可以覆盖被选中的样式，或者新建样式。
- 撤销：按下 Ctrl + Z，可以撤销上一次的编辑操作。
- 恢复：按下 Ctrl + Y，可以恢复上一次的撤销操作。

2.14 多国语言编辑器

图 2.60 多国语言编辑器

Designer 默认使用多国语言编辑器打开语言文件。多国语言编辑器是一个翻译文本列表，可以编辑翻译文本。

注: 语言文件存放在主题目录的 strings 目录。点击工具栏的翻译按钮，可以打开缺省的语言文件。

多国语言编辑器的相关操作如下：

- 添加文本：点击”T+”按钮，可以添加一条翻译文本。
- 添加语言：点击”中 EN+”按钮，可以添加一种语言。

- 清空文本：点击”清空”按钮，可以删除勾选的文本。
- 删除文本：点击文本右侧的”×”按钮，可以删除对应的文本。
- 编辑文本：点击文本编辑框，可以修改文本。
- 检索项目的待翻译文本：点击”检索”按钮，可以检索项目所有 UI 文件中待翻译的文本，并更新到文本列表。
- 编辑模式切换：点击编辑器下方的”form editor”、“text editor”，可以切换编辑模式。

Designer 用下图所示的图标标记控件的文本是否需要翻译。

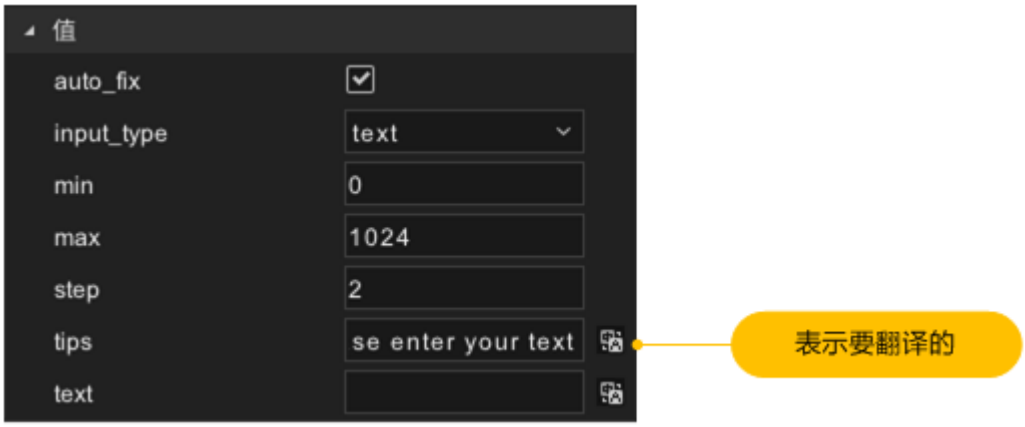


图 2.61　要翻译的文本

注: AWTK 根据原始文本查找当前语言对应的文本，如果找到则使用该文本进行显示，否则使用原始文本。

多国语言编辑器的键盘快捷键如下：

- 撤销：按下 Ctrl+Z，可以撤销上一次的编辑操作。
- 恢复：按下 Ctrl_Y，可以恢复上一次的撤销操作。

3. 许可

3.1 帐号登录

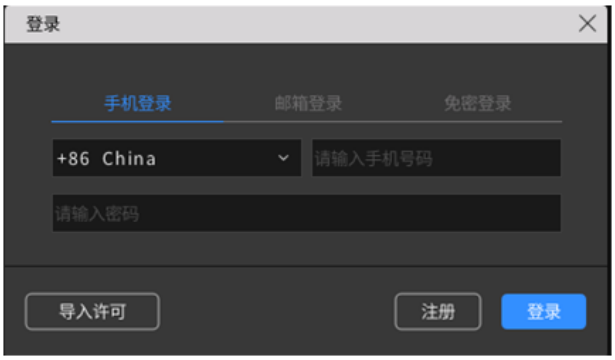


图 3.1 登录对话框

Designer 启动后，需要先登录帐号。如果未登录，会一直停留在登录界面，如图 3.1 所示。如果已有帐号，直接登录即可；如果没有，请点击”注册”按钮，可以自动打开浏览器并跳转到 AWTK 云平台[1]，在网页上完成注册后可以回到 Designer 重新登录。

关于帐号，有一点需要特别说明一下。帐号注册成功后，系统会自动赠送 3 张许可证用于激活 Designer。但是赠送的许可证有效期为 2 个月，为了避免过期影响正常使用，注册时请填写真实有效的个人信息，将有机会获取更多的 Designer 续期卡券。

3.2 用户信息

图 3.2 用户信息窗口

[1] https://awtk.zlg.cn/

如上图所示，登录成功后，工具栏右上角的”登录”按钮会显示为用户名。点击用户名，可以弹出用户信息窗口，显示电话、邮箱、Uid、设备 ID、许可到期时间等基本信息。

通过用户信息窗口，还可以进行以下操作：

- 修改信息：点击”修改信息”按钮，可以自动打开浏览器并跳转到用户信息页面。
- 许可管理：点击”许可管理”按钮，可以打开”许可管理”对话框。更多信息请参阅 3.3。
- 切换帐号：点击”切换帐号”按钮，可以打开如图 3.1 所示的登录对话框，可以选择登录其他帐号。
- 注销帐号：点击”注销登录”按钮，可以注销当前登录。
- 报告问题：点击”报告问题”按钮，将自动打开浏览器并跳转到 issues 页面[2]，可以在上面报告问题。

3.3 许可管理

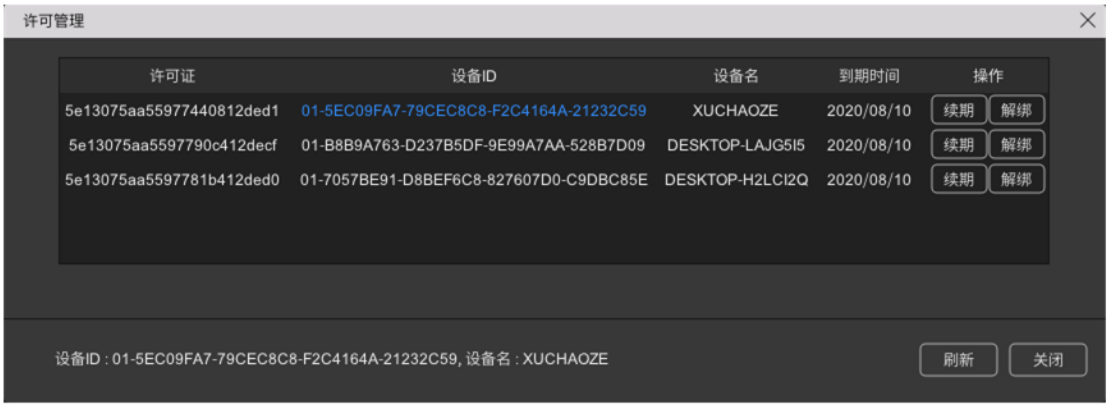


图 3.3 许可管理对话框

如上图所示，登录成功后，如果存在与本机绑定且未过期的许可证，则直接使用该许可证来激活软件。否则，将弹出”许可管理”对话框。

通过”许可管理”对话框，可以进行以下操作：

- 许可激活：点击某个许可证右侧的”激活”按钮，可以使用该许可证激活 Designer。*
- 许可解绑：点击某个许可证右侧的”解绑”按钮，可以解绑该许可证与实际设备的绑定关系。*
- 许可续期：点击某个许可证右侧的”续期”按钮，可以自动打开浏览器并跳转到续期页面进行续期。。*
- 刷新许可列表：点击”刷新”按钮，可以刷新许可证列表。*
- 切换帐号：如果当前没有可用的许可证，则会显示”切换帐号”按钮；点击可以打开登录对话框，可以选择登录其他帐号。

注: 目前，许可证需要使用续期券才能进行续期；登录网页完善个人信息和绑定邮箱，审核通过后将有机会获取续期券。

[2] https://gitee.com/zlgopen/awtk-designer-feedback/issues

4. 帮助与更新

4.1 帮助文档

图 4.1 帮助文档

如上图所示，点击工具栏右上角的”帮助”按钮，可以打开帮助菜单。之后点击”帮助文档”菜单项，可以前往 在线帮助文档[3]。

4.2 检查更新

图 4.2 检查更新

如上图所示，点击工具栏右上角的”帮助”按钮，，可以打开帮助菜单。之后点击”检查更新”菜单项可以检查是否有版本更新。如果有新版本，会弹出更新通知。点击”下载”按钮，会后台下载安装包，下载完毕再弹出通知提示安装。

注: Designer 每次启动时也会自动检查是否有版本更新。

[3] https://awtk.zlg.cn/docs

5. 专题

本章将结合实际的应用场景，来阐述使用 Designer 开发应用程序时的常用操作，帮助开发者快速入门。本章主要内容包括：

- 如何实现多国语言。

5.1 如何实现多国语言

AWTK 支持多国语言的存储和显示，提供文本翻译、图片翻译等功能，并可以在运行时实时切换语言环境。

5.1.1 如何实现文本翻译

1. 编辑 strings.xml 实现文本翻译

AWTK 采用 XML 文件（UTF-8）保存文本的各个语言的对应关系，方便程序员和翻译人员进行编辑，实现多国语言互译需要编写 strings.xml 文件，该文件的具体内容请参阅：awtk/docs/locale.md，这里不详细介绍。

Designer 中通过多国语言编辑器可以快速编辑 strings.xml 文件，添加语言的步骤如下：

- 点击工具栏的翻译按钮，在多国语言编辑器中打开 strings.xml 文件，如图 5.12 所示，若该文件不存在，Designer 会自动创建。
- 点击多国语言翻译器右下角的”中 EN+”按钮，弹出添加语言对话框；
- 在对话框的语言下拉框选择”Chinese”，国家/地区下拉框选择”China”，点击确定按钮，中国中文（zh_CN）添加完成；
- 再以相同的方式添加美国英文（en_US），在添加语言对话框的语言下拉框选择”English”，国家/地区下拉框选择”United States”。

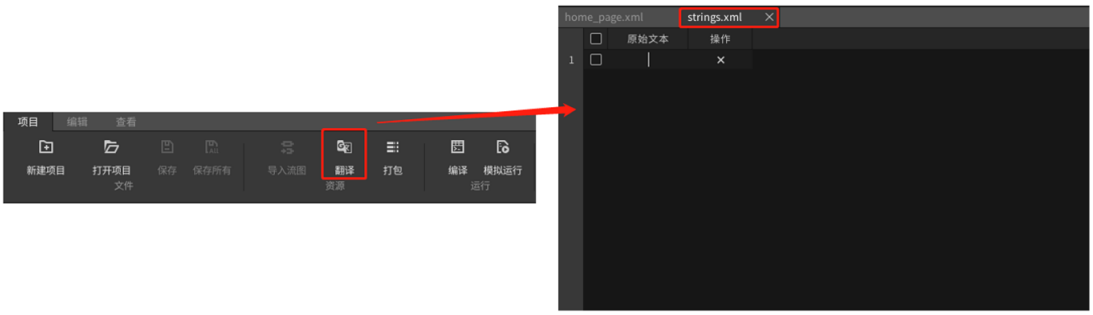


图 5.1 打开多国语言编辑器

两种语言添加完成后，多国语言编译器列表上方新增了两列语言标题，分别为中文”Chinese_China”和英文”English_United States”。

项目的语言种类设置完成后，可为每种语言添加相应的翻译文本，步骤如下：

- 点击多国语言编辑器右下角的”T+”按钮，添加一条翻译文本；
- 设置原始文本和翻译文本，此处设置原始文本为”awtk”；设置中文（zh_CN）为”这

是 AWTK。“；设置英文（en_US）为”This is AWTK.”，如下图所示。

- 保存 strings.xml 文件并打包资源。

保存 strings.xml 文件后，Designer 会在项目主题目录的 strings 文件夹中生成多国语言的二进制文件，此处为 zh_CN.bin 文件和 en_US.bin 文件，分别对应中国中文和美国英文。

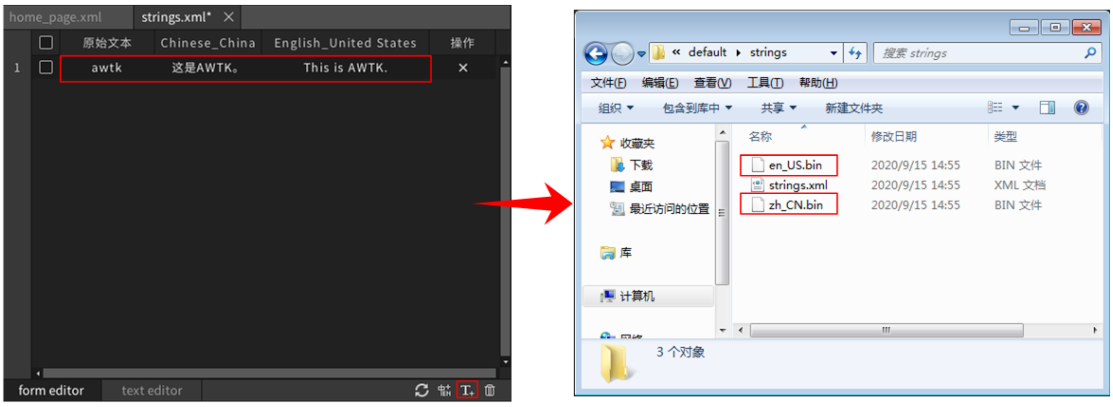


图 5.2　添加一条翻译文本

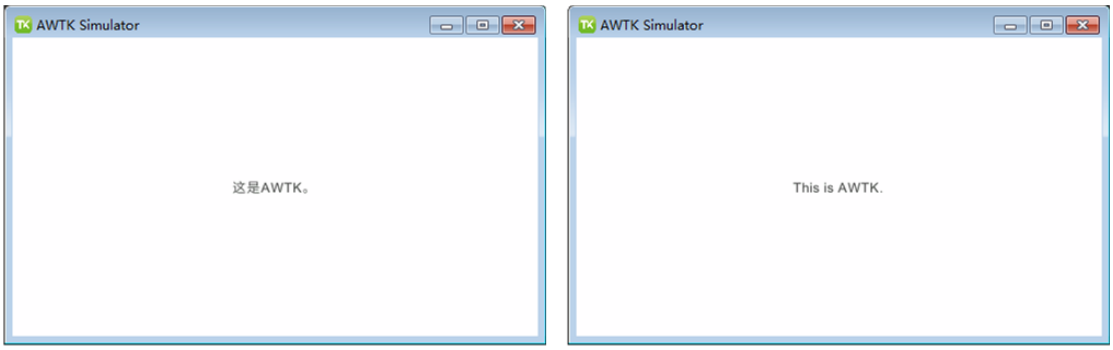
注: AWTK 使用二进制格式保存多国语言文件，为方便计算机处理，每种语言的字符串放在独立的文件中，字符串按升序储存，查询时使用二分法查找。

编辑好 strings.xml 文件后，以 label 控件为例，实现字符串翻译的步骤如下：

- 通过控件列表向 UI 文件添加 label 控件 → 选中控件 → 控件属性编辑器 → 值；
- 点击 text 属性右侧的是否翻译按钮，当按钮显示翻译图标时表示该控件文本是翻译的；
- 设置 text 属性的值为原始文本”awtk”；
- 保存 UI 文件并打包资源。

图 5.3　将 label 控件文本设置为翻译的

编译并运行项目，当项目的语言为中文时，label 控件显示中文文本”这是 AWTK。“，当项目的语言为英文时，label 控件显示中文文本”This is AWTK.”，如下图所示。



中文语言　　英文语言

图 5.4　根据项目语言显示对应字符串

2.　动态文本翻译

若需要翻译的文本是在运行时动态生成的，例如用一个模板根据当前的数据动态生成，可通过多国语言编辑器添加模板翻译文本，然后先调用 locale_info_tr() 函数获取当前语言的模板，生成真正要显示的字符串，再调用 widget_set_text_utf8() 函数设置到控件中，代码如下：

```
//Designer安装目录/SDK/awtk/demos/demo_tr_app.c
static ret_t set_locale_value(widget_t* widget, int32_t value) {
  char str[64];

  /* 获取当前语言模板，生成要显示的字符串 */
  const char* format = locale_info_tr(locale_info(), "value is %d");

  /* 将可变参数格式化至字符串中 */
  tk_snprintf(str, sizeof(str), format, value);

  /* 将字符串设置到label控件中*/
  widget_set_text_utf8(widget, str);

  return RET_OK;
}

/* EVT_LOCALE_CHANGED事件回调函数 */
static ret_t on_locale_changed(void* ctx, event_t* e) {
  widget_t* win = WIDGET(ctx);
  widget_t* value = widget_lookup(win, "value", TRUE);

  set_locale_value(value, 100); /* 设置翻译文本 */
  ...
  return RET_OK;
}

ret_t application_init() {
  ...
  widget_t* win = window_create(NULL, 0, 0, 0, 0);
```

```
  /* 绑定事件回调函数 */
  widget_on(win, EVT_LOCALE_CHANGED, on_locale_changed, win);
  ...
  return RET_OK;
}
```

需要注意的是，对于动态翻译的文本，若切换语言时需立即刷新，则须添加 EVT_LOCALE_CHANGED 事件的处理。

5.1.2 如何实现图片翻译

在一些应用程序中，有些文字是直接绘制在图片上的，所以在切换到不同的语言时，需要加载不同的图片，这时只要在图片名称的最后包含”$locale$“即可，AWTK 会自动将”$locale$”替换成当前的语言。以 image 控件为例，实现图片翻译的如下：

- 通过资源浏览器向 xx 文件夹中添加不同语言环境下的图片资源，要求两张图片名称除后缀外完全相同，中文图片名称后缀为”_zh_CN”，英文图片名称后缀为”_en_US”，此处添加图片”language_zh_CN”和”language_en_US”，如下图左侧所示；
- 通过 Designer 中的控件列表向 home_page 文件添加 image 控件；
- 设置 image 控件的 image 属性为 language_$locale$，如下图右侧所示；
- 保存窗体 UI 文件并在 Designer 的工具栏中打包资源。

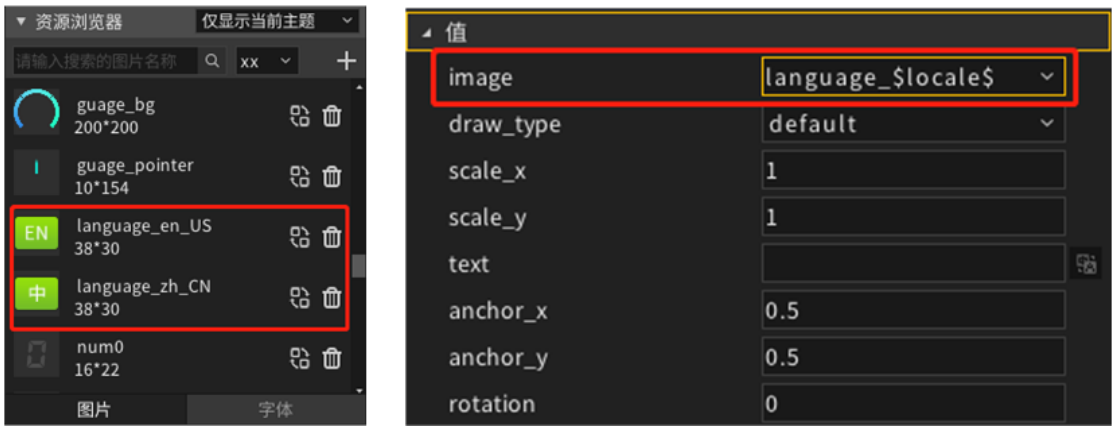
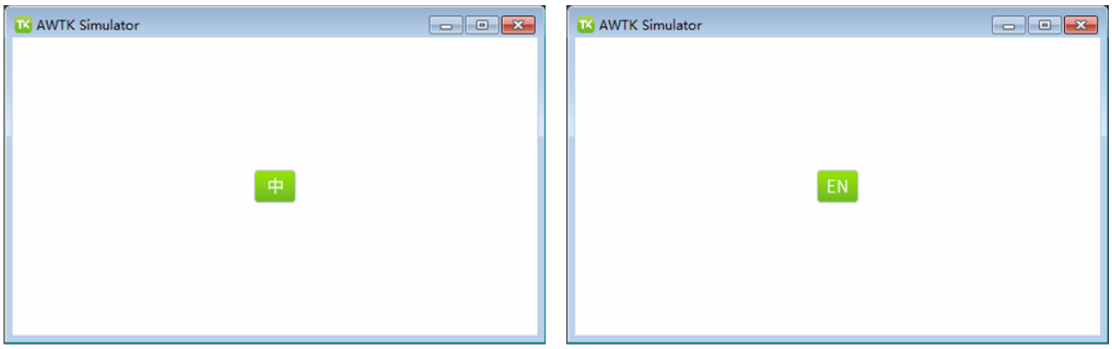


图 5.5 添加不同语言环境下的图片并设置 image 控件

编译并运行项目，当项目的语言为中文时，image 控件显示图片”language_zh_CN”，当项目的语言为英文时，image 控件显示图片”language_en_US”，如图 5.17 所示。



中文语言　　英文语言

图 5.6　根据项目语言显示对应图片

注: 图片翻译所用的图片名称为”language_$locale$”，若当前语言为 en_US，加载图片时会按下列顺序查找：language_en_US、language_en、language_，其它资源类似。

5.1.3　如何设置不同语言环境下的默认字体

AWTK 中默认的缺省字体是 default.ttf，若要设置不同语言环境的默认字体，首先需要在 Designer 中通过资源浏览器的字体页面添加不同语言的字体文件，步骤如下：

- 点击资源浏览器字体页面右上角的”+”按钮；
- 分别添加 ttf 格式的中文字体文件和英文字体文件，添加完成后点击确定按钮，此处以华文彩云、华文琥珀为例；
- 在资源浏览器中分别将华文彩云和华文琥珀重命名为”default_zh.ttf”、“default_en.ttf”；

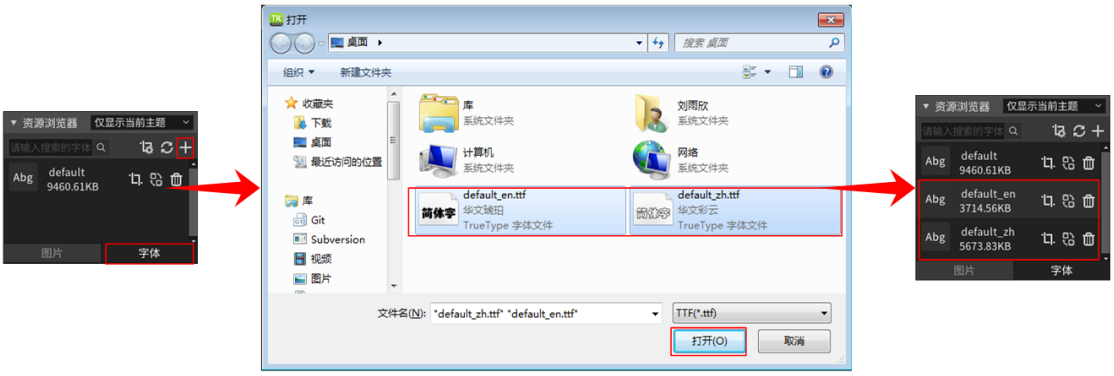


图 5.7　添加不同语言的字体文件

添加完字体文件并打包资源后，可以通过以下两种方式设置不同语言环境的默认字体。

1.　切换语言时重新设置默认字体

切换语言时调用 system_info_set_default_font() 函数设置默认字体，代码如下：

```
/* 设置默认字体为default_zh */
system_info_set_default_font(system_info(), "default_zh");
locale_info_change(locale_info(), language, country); /* 切换语言 */
```

2. 通过 $locale$ 自动加载默认字体

若不想在每次切换语言时都调用 system_info_set_default_font() 函数设置默认字体，与图片翻译类似，只需在字体资源文件名称的最后包含”$locale$”即可。

在程序的初始化函数 application_init() 里调用一次 system_info_set_default_font() 函数，设置默认字体包含”$locale$”，代码如下，效果如下图所示。

```
ret_t application_init(void) {
  /* 设置默认字体包含"$locale$" */
  system_info_set_default_font(system_info(), "default_$locale$");
  widget_t* win = window_open("home_page");
  ...
  return RET_OK;
}
```

中文语言（华文彩云） 英文语言（华文琥珀）

图 5.8 根据项目语言加载默认字体

5.1.4 如何在运行时切换语言

在 C 源文件中，可以调用 locale_info_change() 函数设置当前的国家和语言，点击中英互译按钮实时切换项目语言，代码如下：

```
//Designer安装目录/SDK/awtk-examples.v2/CleanAir.v2-Demo/src/window_main.c
static ret_t change_locale(const char* str) {
  char country[3];
  char language[3];
  strncpy(language, str, 2);
  strncpy(country, str + 3, 2);
  locale_info_change(locale_info(), language, country); /* 切换语言 */
  return RET_OK;
}

/* 中英互译按钮点击事件的处理函数 */
static ret_t on_language(void* ctx, event_t* e) {
  widget_t* language_btn = (widget_t*)e->target;
```

```
  const char* language = locale_info()->language; /* 获取项目当前语言 */

  if (tk_str_eq(language, "en")) {
    change_locale("zh_CN");
    widget_use_style(language_btn, "language_zh");
  } else {
    change_locale("en_US");
    widget_use_style(language_btn, "language_en");
  }

  return RET_OK;
}
```

调用 locale_info_change() 函数会触发 EVT_LOCALE_CHANGED 事件，在该事件的处理函数中可以更新运行时动态翻译的字符串或者添加其他处理，详细请参见 5.2.1 章节。

6. 快速入门

6.1 开始使用 AWTK Designer

AWTK Designer（下文简称 Designer）是专门用来制作 AWTK 应用程序 UI 界面的实用工具。只要通过拖拽和点击就可以完成复杂的界面设计。同时也提供代码编辑器，可以完成逻辑代码的编写。

本文 awtk-demo-hello-designer[4] 为例（又称：HelloDesigner-Demo），介绍如何使用 Designer 开发 AWTK 应用程序。在开发 AWTK 应用程序前，首先需要了解 Designer 的基本用法，比如：如何使用 Designer 新建项目。

6.1.1 新建项目

首先启动 Designer，会弹出”新建项目”对话框，如下图所示：

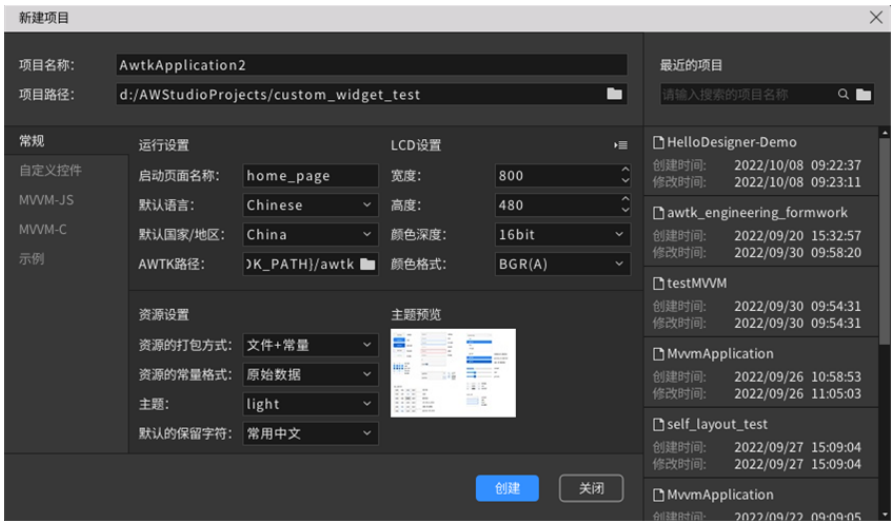


图 6.1 新建项目

选择常规项目，并对项目进行一些设置，其中几个常用的选项如下：

- 项目名称与路径可根据实际情况设置。
- 启动页面名称为程序运行时展示的最初窗体的名称，默认为 home_page。
- 默认语言与国家/地区与翻译相关，在后续介绍翻译功能时讲解，默认为中国中文（zh_CN）。
- AWTK 路径默认为安装 AWStudio 时内置 AWTK SDK，用户也可以按需修改。

其他参数一般采用默认设置，有需要可以更改，最后点击”创建”按钮即可。如果在后续的操作中想对这些项目参数进行修改，可以点击左下角的”设置”按钮，再点击”项目设置”进行修改。

[4] https://jihulab.com/awtk/awtk-demo-hello-designer

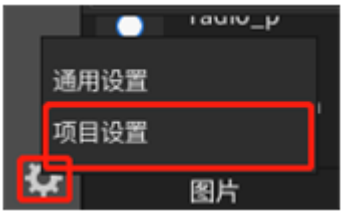


图 6.2 项目设置

6.1.2 新建窗体

此处创建名为 basic 窗体文件，步骤如下：

- 在 Designer 左上角的”编辑”界面中，点击”新建窗体”图标，会弹出”新建窗体”窗口。
- 在窗体类型中选择”Window”，并输入窗体文件名称为”basic”，点击”创建”按钮，即可新建窗体。

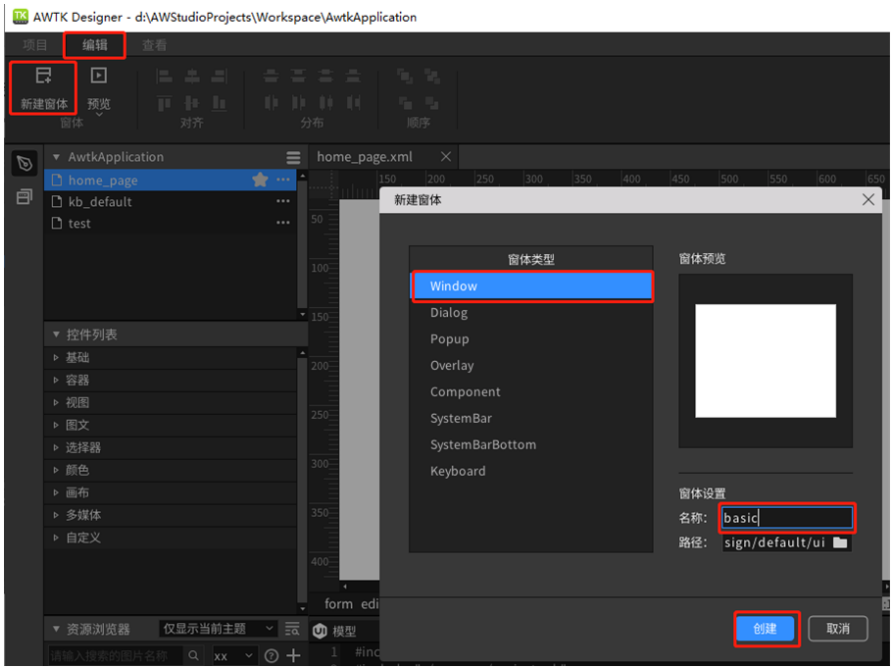


图 6.3 新建窗体

6.1.3 打包资源

使用 Designer 完成界面设计后，在”工具栏”的”项目”页面中点击”打包”按钮，即可打包生成资源。

图 6.4 打包项目资源

6.1.4 编译运行

1. 搭建编译环境

在编译程序之前，需要搭建好 Windows 编译环境，搭建环境的详情步骤请参考 AWTK 在线帮助文档[5] 中的《AWTK 开发实践》。如果已经搭建好编译环境，可以忽略本节内容。

2. 编译和模拟运行应用程序

完成界面设计和代码编辑后，可以用 Designer 编译并模拟运行应用程序，步骤如下：

- 在”工具栏”的”项目”页面中点击”编译”按钮，弹出程序编译的命令行窗口，如果编译成功，会提示”scons:done building targets”，如果编译失败的话也会有错误提醒。
- 编译成功后，可以点击”编译”按钮旁的”模拟运行”按钮进行模拟。

图 6.5 编译和模拟运行

图 6.6 编译成功

[5] http://awtk.zlg.cn/docs

6.2 HelloDesigner-Demo 项目

本章简单介绍 HelloDesigner-Demo 项目的具体功能，并以该项目为例说明 AWTK 应用程序的目录结构，帮助读者更好的理解下文中的界面设计和代码编辑。

6.2.1 功能详解

HelloDesigner-Demo 的运行效果如下图所示，读者可以在 AWStudio 中下载该示例，或者前往仓库下载源码：awtk-demo-hello-designer[6]，下载完成后可以使用 Designer 打开该项目，并编译运行。

图 6.7 HelloDesigner-Demo 运行效果

使用 Designer 打开 HelloDesigner-Demo 的步骤如下：

- 在”工具栏”的”项目”页面中点击”打开项目”按钮；
- 在弹出的”选择项目文件”窗口中，选择项目目录下的”project.json”文件打开。

图 6.8 打开项目

[6] https://jihulab.com/awtk/awtk-demo-hello-designer

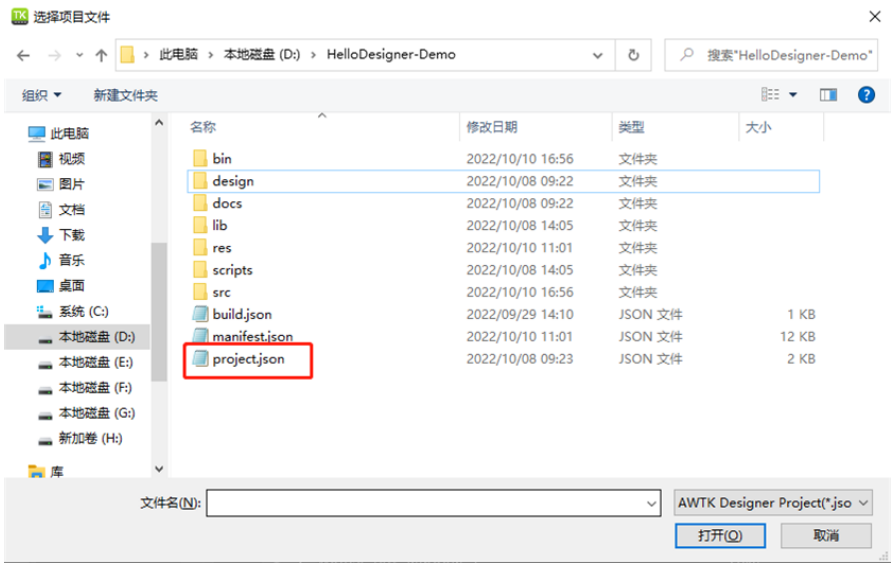


图 6.9　选择 HelloDesigner-Demo 项目打开

HelloDesigner-Demo 中包含的界面和各个界面的实现功能如下所示：

主界面功能：

- 点击右上角的中英互译按钮可实现界面显示文本的中英文切换；
- 点击中间 4 个按钮可打开对应的窗口；

Basic 界面功能：

- 在编辑器中输入文本可同步更新到上方的静态文本控件中；
- 定时增加水平进度条的值；
- 点击 dec/inc 按钮，可减少/增加水平进度条的值；
- 水平进度条的值改变后会同步更新到 dec/inc 按钮之间的静态文本控件上；
- 调整垂直滑动条的值可以改变垂直进度条的值；
- 点击 close 按钮可关闭 Basic 窗口。

Background Change 界面功能：

- 点击中间 6 个背景切换按钮可切换对应的背景颜色或背景图片；
- 点击 close 按钮可关闭 Background Change 窗口。

List View 界面功能：

- 点击包含图标的列表项目时，将列表项的显示文本更新到静态文本控件中；
- 点击 Clone Self 列表项可在列表视图尾部添加一个列表项；
- 点击 Remove Self 列表项可从列表视图中移除该列表项；
- 点击 close 按钮可关闭 List View 窗口。

Animate Widget 界面功能：

- 各种控件动画；
- 点击 close 按钮可关闭 Animate Widget 窗口。

6.2.2 项目目录

HelloDesigner-Demo 项目各目录功能如下所示，其中大部分在使用 Designer 新建项目时就会自动生成：

文件名	作用
bin	存放编译后的可执行文件与动态库
design	存放 Designer 工程的原始资源文件
lib	存放编译的静态库 (使用静态编译)
res	存放运行时的资源文件 (Designer 打包资源输出目录)
scripts	存放工具脚本
src	存放 C 源代码
project.json	Designer 工程配置文件
SConstruct	SCons 编译脚本文件

6.3 HelloDesigner-Demo 界面设计

开发 AWTK 应用程序的第一个步骤通常是进行界面设计，本章介绍如何使用 Designer 来制作 HelloDesigner-Demo 项目中的五个界面：主界面、Basic 界面、Background Change 界面、List View 界面和 Animation Widget 界面。

6.3.1 主界面设计

HelloDesigner-Demo 程序运行时的主界面如下图所示，界面功能详见上文第二节，下面介绍如何使用 Designer 设计该界面。

图 6.10 主界面

1. 创建按钮控件

以控件按钮为例，在 Designer 左侧的”控件列表”中，选中”按钮”，将其拖拽进窗体中，便能创建按钮控件，如下图所示，此处创建 5 个按钮控件。

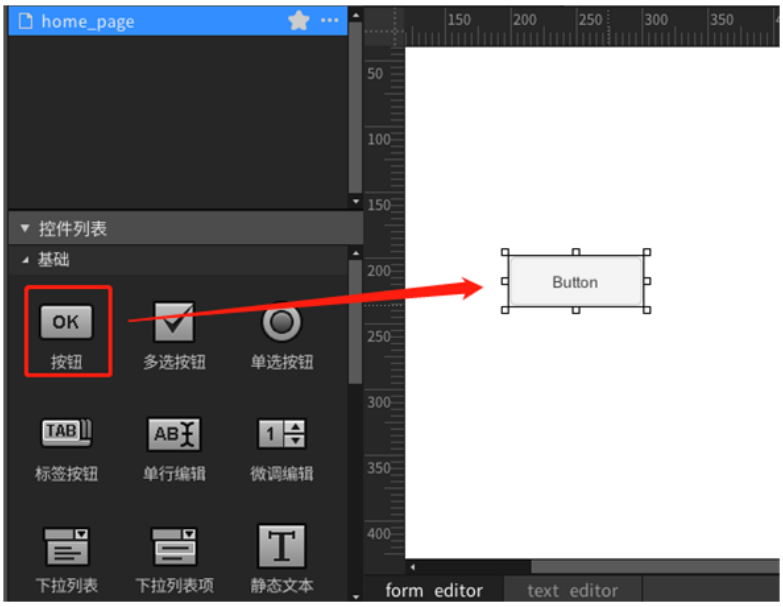


图 6.11　创建按钮控件

2. 修改控件名称

创建好控件后，可以给控件修改合适的名称，以便后续写逻辑代码时，可以通过名称找到对应的控件，步骤如下：

- 在”窗体编辑器”中选中要修改名称的按钮控件；
- 在 Designer 右侧”控件编辑器”的”名称”列表中修改”name”属性，或者双击对象浏览器的对应控件修改名称；
- 此处分别修改 5 个按钮控件的名称为”basic_btn”、“background_btn”、“listview_btn”、“animator_btn”和”language_btn”，如图所示。

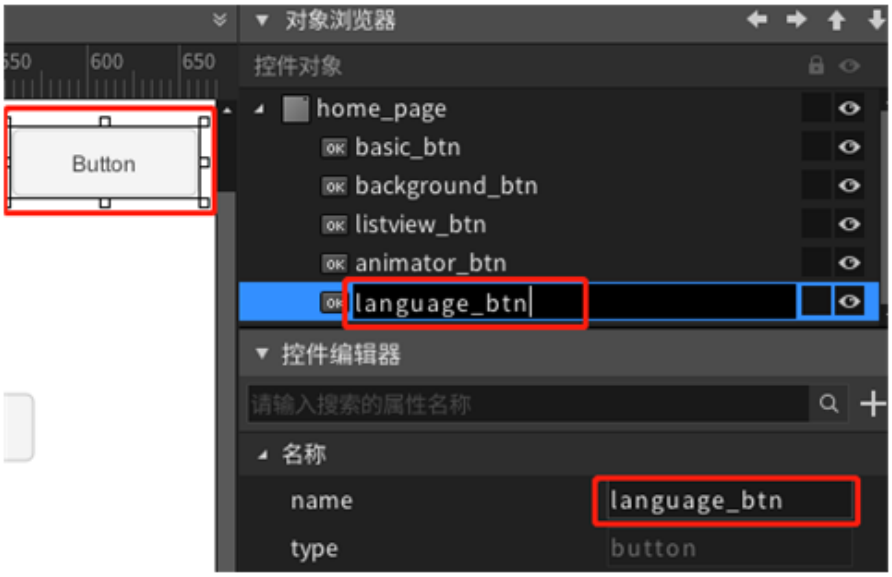


图 6.12　修改控件名称

3. 调整控件布局

以“Basic”按钮为例，可拖拽和拉伸控件来调整控件的布局，也可以选中控件后，在“控件编辑器”中的“布局”列表中设置 x、y 坐标和 w、h 宽高来调整布局。此处采用百分比来设置控件布局，最终布局效果如图所示。

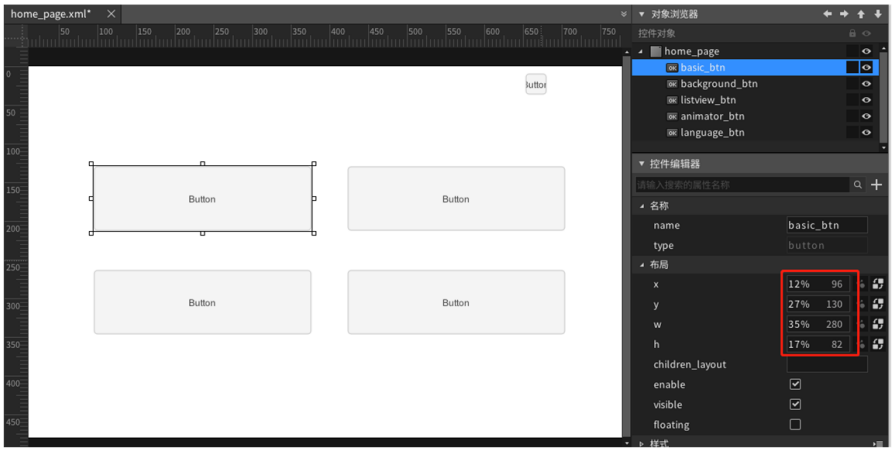


图 6.13　调整控件布局

4. 设置按钮的显示文字

为了更加直观地表示按钮的功能，可在按钮中添加文字显示，步骤如下：

- 在”窗体编辑器”中选中要修改显示文本的按钮控件；
- 在”控件编辑器”的值列表中，修改”text”属性，即可设置按钮的显示文字，或者直接双击按钮控件进行修改；
- 此处将项目右上角的中英互译按钮（language_btn）的显示文本删除，其余 4 个按钮控件的显示文本分别设置为“Basic”、“Background Change”、“List View”和“Animate Widget”，如图所示。

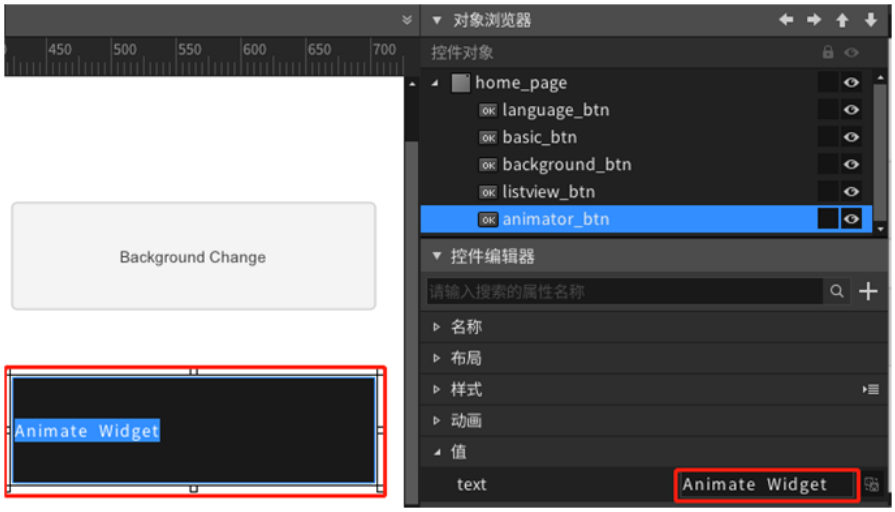


图 6.14　设置显示文字

5. 修改显示字体的大小

创建好控件后，Designer 会提供控件默认样式，如需修改样式，可在“控件编辑器”的“样式”列表中修改相应的样式属性。以“Basic”按钮的字体样式为例，修改“font_size”属

性值为 22（默认值为 18），可增大显示字体的大小，如图所示，其它按钮控件同理。

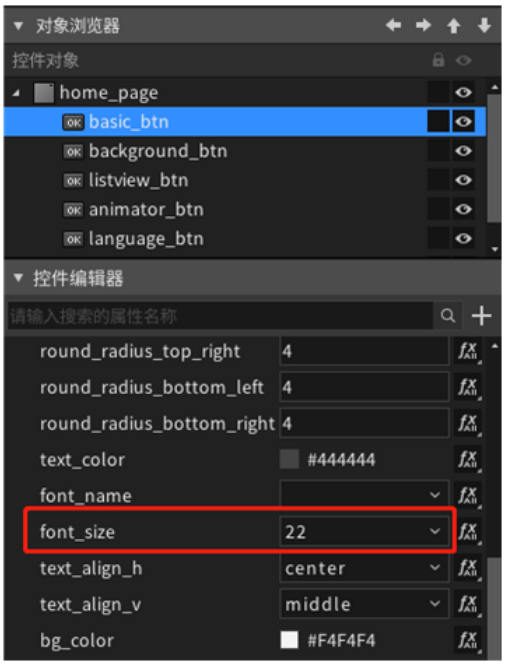


图 6.15　修改控件显示字体大小

6. 设置按钮在其他状态下的样式

控件一般都会有多种状态，前面字体大小设置我们只设置了“normal”状态下的，为避免不同状态下字体大小不一致，因此其他状态下也要进行修改。如图所示，展开“state”属性的下拉框，可选择不同的控件状态。此处将“over”和“pressed”状态下的字体大小也设置 22。

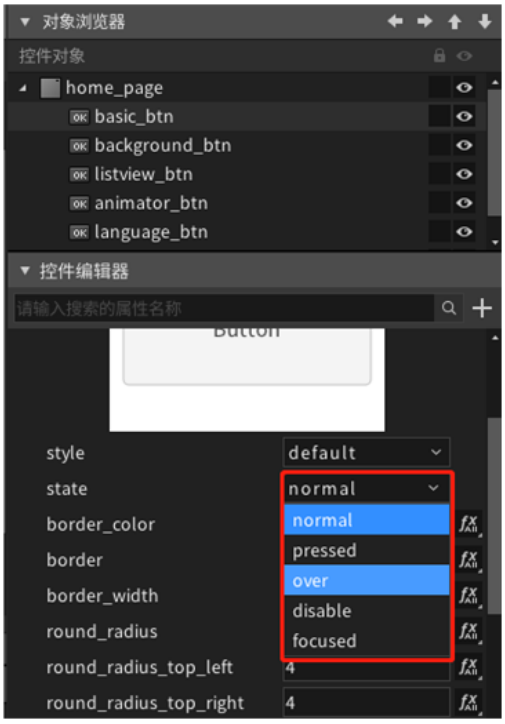


图 6.16　切换控件状态

7. 设置按钮的翻译文本

主界面需实现中英互译功能，因此需要将按钮的显示文本设置为可翻译的，步骤如下：

- 在”窗体编辑器”中选中要设置的按钮控件；
- 在”控件编辑器”的”值”列表中，点击”text”属性右侧的”设为翻译的”按钮即可，此处以主界面上的”Basic”按钮为例，如下图所示，其余按钮同理。

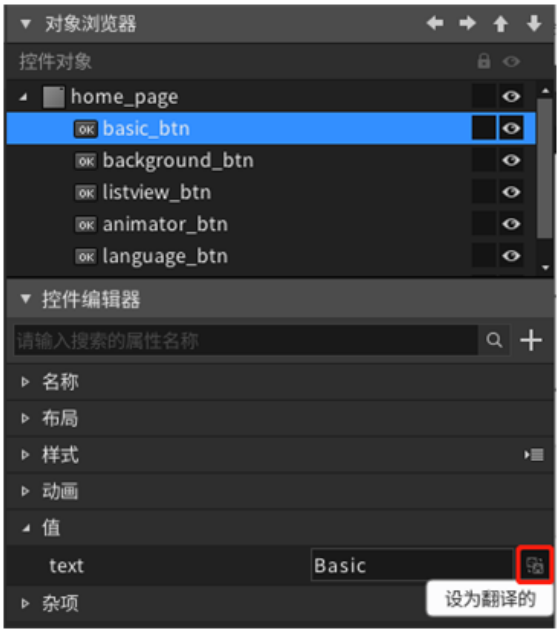


图 6.17　设置按钮显示文本为可翻译的

8. 添加中英互译文本

将按钮控件的显示文本设置为翻译的后，还需要添加中英文翻译文本，添加步骤如下：

- 点击“工具栏”的“项目”页面中的“翻译”按钮，打开“多国语言编辑器”；
- 点击“多国语言编辑器”右下角的“添加一种语言”按钮，弹出“添加语言”对话框，选择“语言”为“Chinese”，“国家/地区”为“China”，点击确定按钮；
- 添加成功后在“多国语言编辑器”中可看见新增列“Chinese_China”，在该列中添加原始文本对应的中文，如图所示，英文同理，选择”English”与”United States”即可。

图 6.18　添加翻译文本

在第一节新建项目时提到的默认地区与语言会在程序初始化时根据翻译文本对有需要翻译的文本进行翻译。

9. 裁剪字库

Designer 支持对字库进行裁剪，去掉应用程序中不需要的字符，特别是在嵌入式平台，可以减少资源占用。本项目使用到中文字符有：“基础背景切换列表控件动画”，在裁剪字库时需保留这些字符，步骤如下：

- 在 Designer 左下角的“资源浏览器”中点击“字体”按钮，切换至字体资源界面；
- 点击字体资源界面右上角的“裁剪设置”按钮，打开“项目设置”对话框；
- 在“保留的字符”编辑框中添加需要保留的中文字符，点击确定按钮；
- 回到字体资源界面，点击 default 字体右侧的“裁剪字体”按钮，即可裁剪字体，可见字体资源大小从原本的 991.96KB，减小为 28.68KB，如图所示。

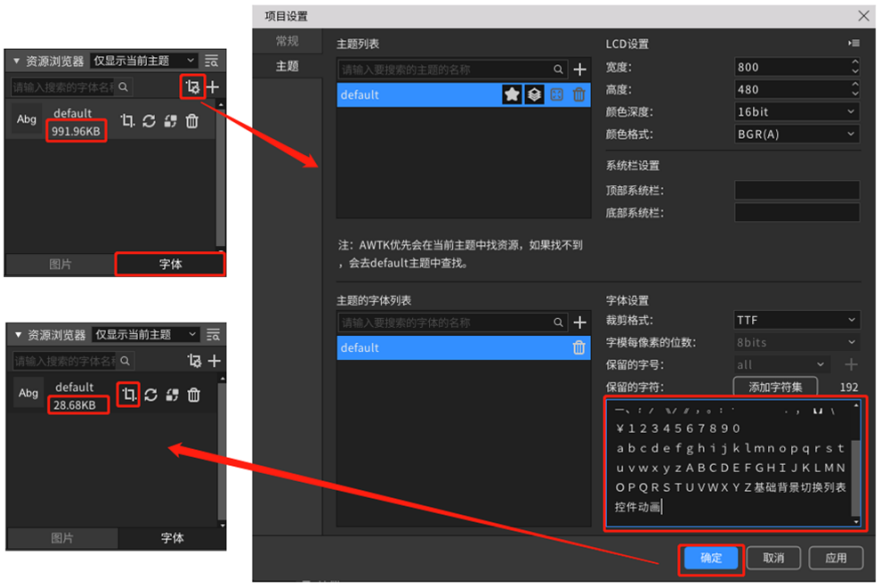


图 6.19 裁剪字体资源

10. 添加图片资源

点击 Designer 左侧“资源浏览器”中的“＋”按钮，在弹出的对话框中选中所需图片，点击“打开”，即可添加图片资源，如图所示。

此处添加图片“awtk_logo.png”、“language_en.png”和“language_zh.png”，这三张图片可在示例程序 awtk-demo-hello-designer-master/design/default/image/xx 目录下找到。

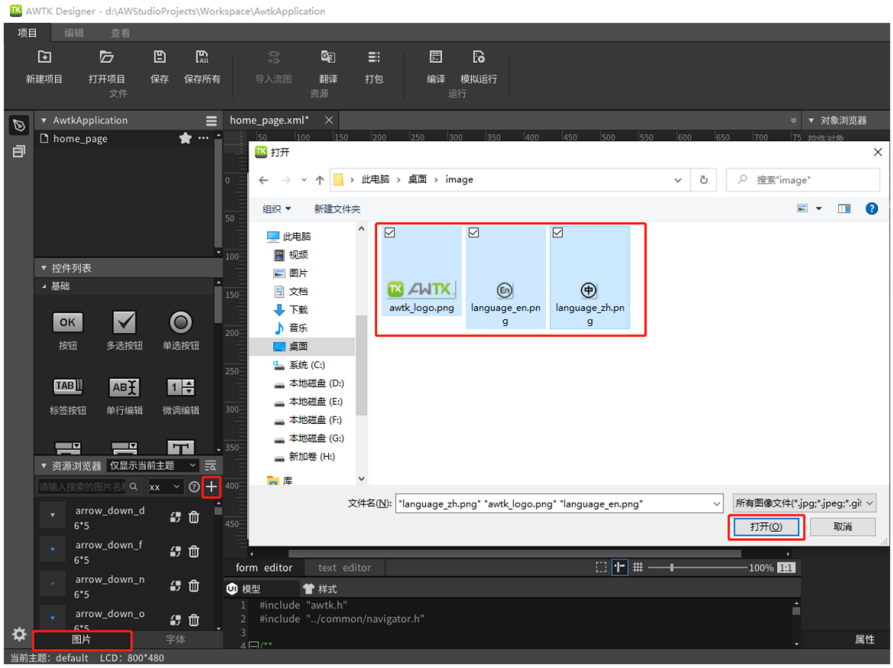


图 6.20　添加图片资源

11. 添加图片控件

将图片添加进工程后，创建图片控件，在“控件编辑器”的“值”列表中，点击“image”属性下拉框，选中上一小节中添加的 awtk_logo.png 图片，即可将图片显示到界面中，如图所示。

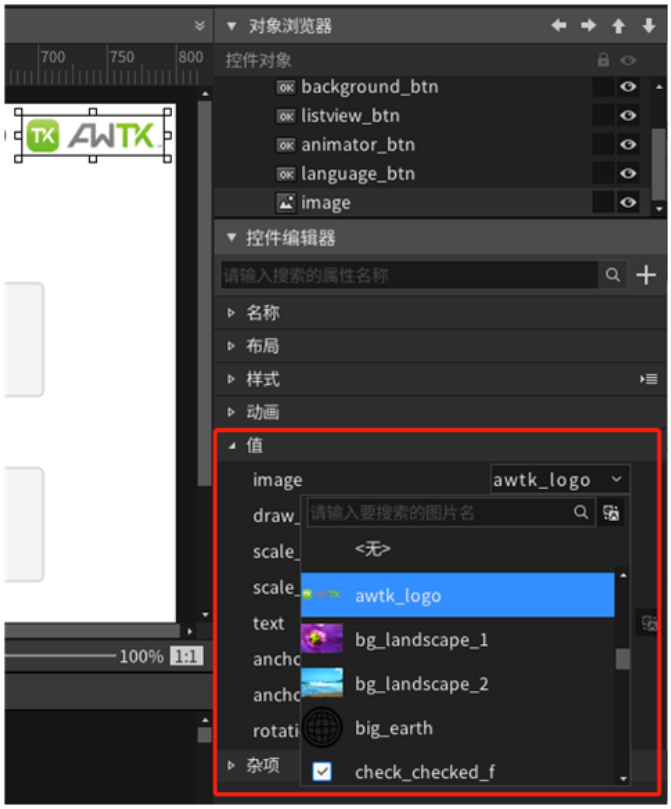


图 6.21　创建图片控件显示 logo

12. 创建按钮控件的样式

中英互译按钮（language_btn）在不同的语言环境下需要显示对应的图标，此处为按钮控件创建样式“en”和“zh”，分别在中文和英文语言环境下使用，步骤如下：

- 点击”窗体编辑器”下方的”样式”按钮；
- 打开”样式”窗口后点击右下方的”+”弹出”新建样式”窗口；
- 设置控件类型为”button”，样式名称为”en”，点击确定按钮，创建样式；
- 将“en”样式中“state”属性切换到“normal”，点击“icon”属性下拉框，选择添加的 language_en.png 图片，如图所示；
- 按照上述步骤再添加按钮的“zh”样式，“icon”属性选择 language_zh.png 图片。

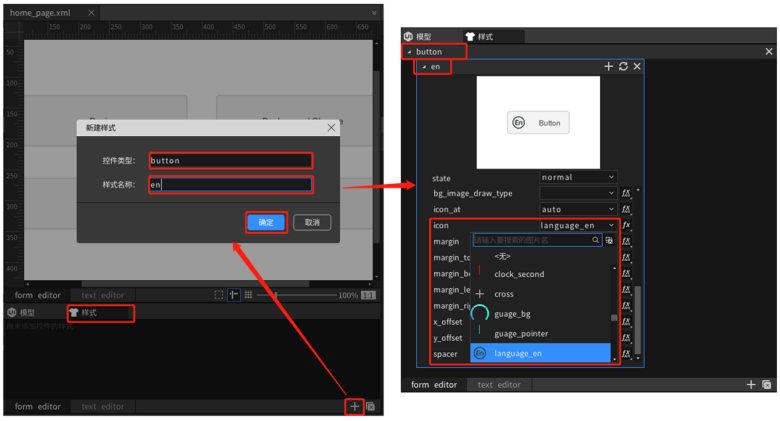


图 6.22　创建按钮控件”en”样式

注: 此处添加的样式默认添加在 design/default/style 文件夹中，并且样式文件与窗口同名。

6.3.2　Basic 界面设计

点击主界面的”Basic”按钮，可以进入 Basic 界面，如下图所示，界面功能详见第二节，可以参考第一节的内容新建 Basic 界面的窗体，下面介绍如何使用 Designer 设计该界面。

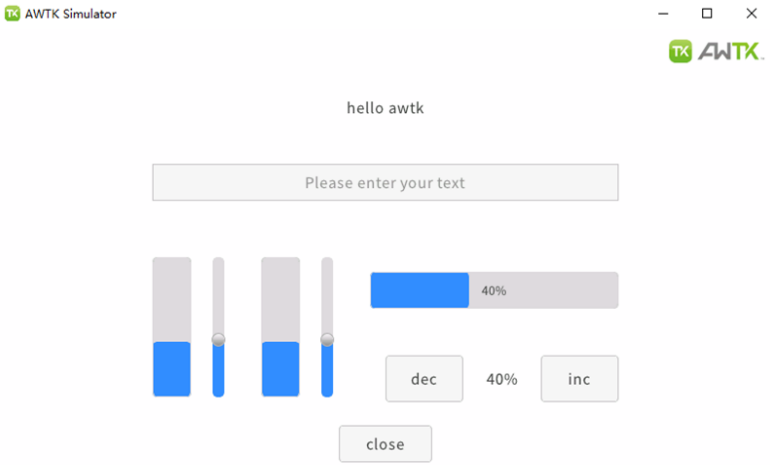


图 6.23　Basic 界面

1. 添加窗口动画

在窗口打开或关闭时，需要播放一些动画过度效果，例如 Basic 窗口打开时向左平移，关闭时向右平移，在 Designer 中可设置窗口动画实现，步骤如下：

- 在“对象浏览器”中选中“basic”窗口；
- 在“控件编辑器”的“窗口过渡动画”列表中，点击“open_anim_hint”右侧的编辑框，弹出“设置窗口动画”对话框；
- “设置窗口动画”对话框中有多种动画类型可以选择，此处选择“左右平移”，设置动画持续时间设置为 500 ms，动画趋势使用默认的“cubic_out”，勾选“应用到关闭动画”，点击确定按钮，如图所示。

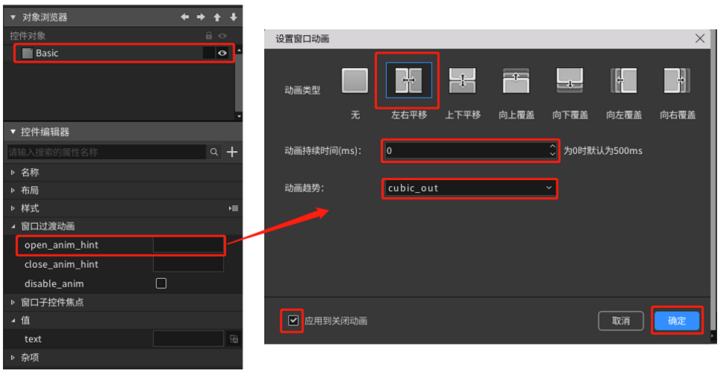


图 6.24　添加窗口过渡动画

2. 添加软键盘

编辑框控件在输入状态下，默认会弹出软键盘，如需使用软键盘，需要先添加软键盘资源，步骤如下：

- 在“工具栏”中切换至“编辑”页面，点击“新建窗体”按钮；
- 在弹出的“新建窗体”对话框的“窗体类型”中选择“Keyboard”，再点击创建按钮即可，如图所示。

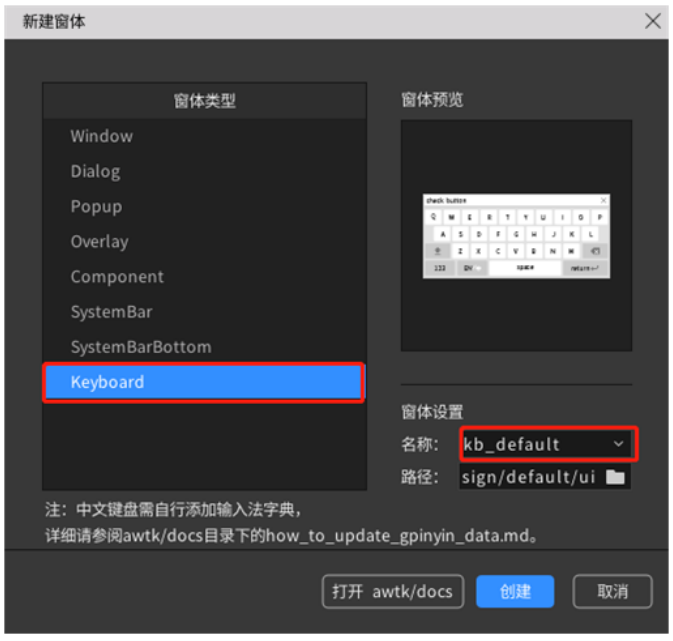


图 6.25　添加软键盘

3. 设置进度条控件垂直显示

进度条控件和滑动条控件默认为水平显示，以进度条控件为例，在“窗体编辑器”中选中进度条控件，然后在“控件编辑器”的“杂项”列表中，勾选“vertical”属性，再对控件布局进行调整，即可将进度条控件修改为垂直显示，如图所示，滑动条同理。

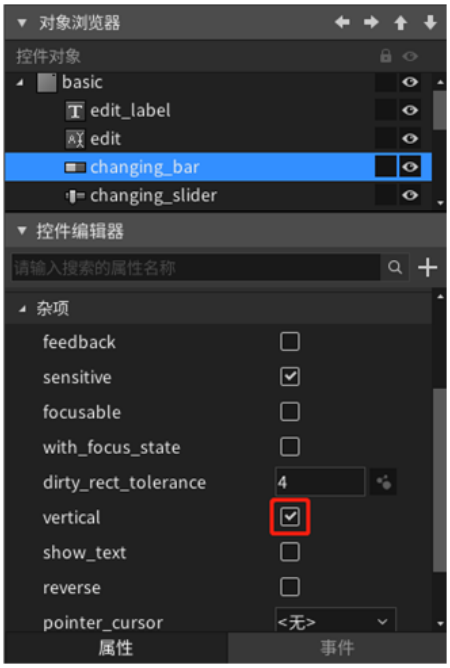


图 6.26　设置进度条垂直显示

4. 设置进度条显示文本进度

进度条可设置显示文本，在“窗体编辑器”中选中对应的进度条控件，然后在杂项列表中勾选“show_text”属性，可让进度条以文本形式显示进度，如图所示。

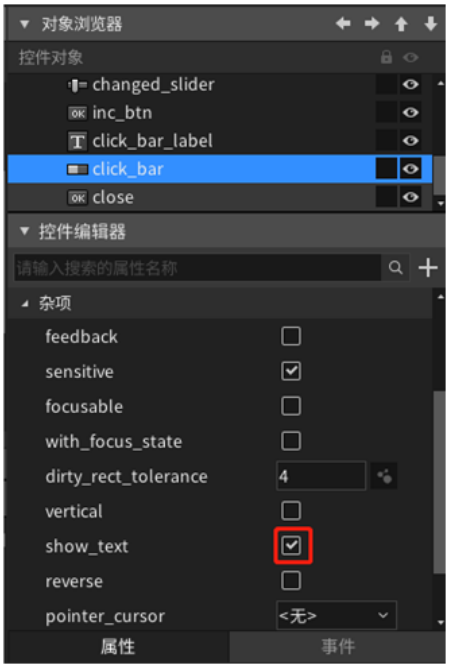


图 6.27　设置进度条显示文本进度

本界面其余控件按上一节进行设置或者使用默认设置即可，所有用到的控件有：图片、按钮、单行编辑、静态文本、滑动条和进度条。

6.3.3 Background Change 界面设计

点击主界面的“Background Change”按钮，可进入 Background Change 界面，如图所示，界面功能详见第二节，下面介绍如何使用 Designer 设计该界面。

图 6.28 Background Change 界面

1. 设置容器的网格布局

创建通用容器控件（view）后，可设置容器的子控件布局，步骤如下：

- 在”窗体编辑器”中选中要设置的容器；
- 点击“控件编辑器”的“布局”列表中“children_layout”属性的编辑框；
- 在弹出的“设置子控件的排版”对话框中可设置排版类型，此处选择“网格类型”，水平间距、垂直间距和子控件间的间距均设置为 15，行数和列数分别设置为 3 和 2，其余保持默认设置，点击确定按钮，如图所示。

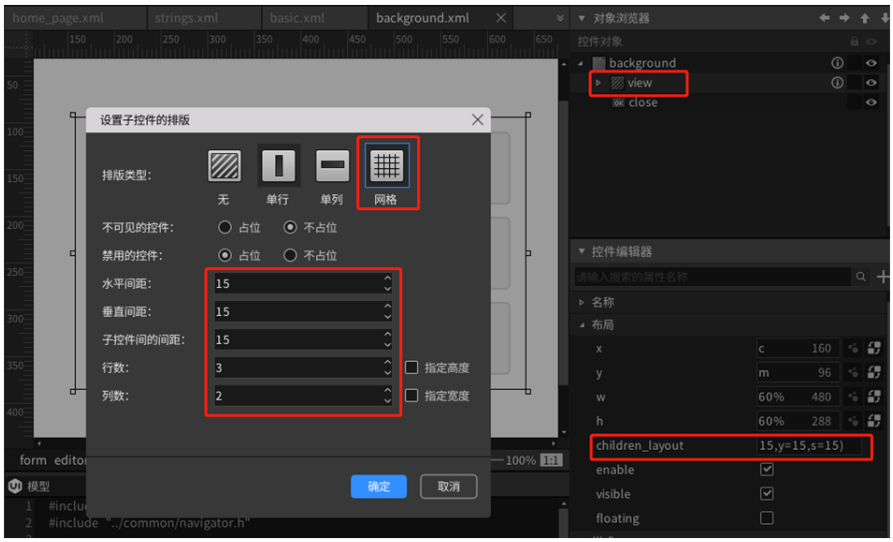


图 6.29 设置容器的子控件布局

2. 向容器中添加子控件

向容器添加子控件有以下两种方式，此处将按钮控件添加到通用容器控件（Veiw）中：

- 在“窗体编辑器”中，将按钮控件拖拽到通用容器控件（Veiw）上方时，通过按空格键将其放入容器内部，如图左侧所示；
- 在“对象浏览器”中，直接将按钮控件拖拽到通用容器控件（Veiw）上方，松开鼠标即可，如图右侧所示；

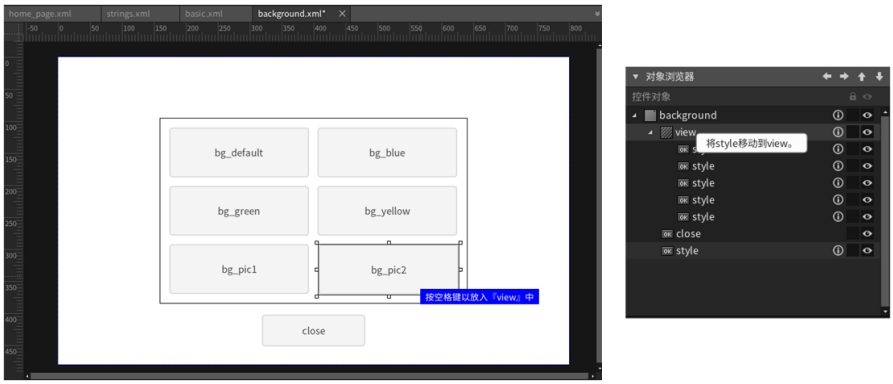


图 6.30　向容器中添加子控件

3. 创建窗口样式设置不同的背景

Background Change 界面要实现背景切换的功能，需要创建窗口样式设置不同的背景，此处以创建蓝色背景的窗口样式为例，步骤如下：

- 参考本节的第一小节，为 window 控件添加新的样式；
- 控件类型为“window”，样式名称为“blue”，点击确定按钮，创建样式；
- 设置“bg_blue”样式中的“bg_color”属性为蓝色，若要设置样式的背景图片，则可以在样式的“bg_image”属性右侧的下拉框中选择对应的图片，如图所示。其余背景颜色和图片的设置方法类似，这里不再累赘。

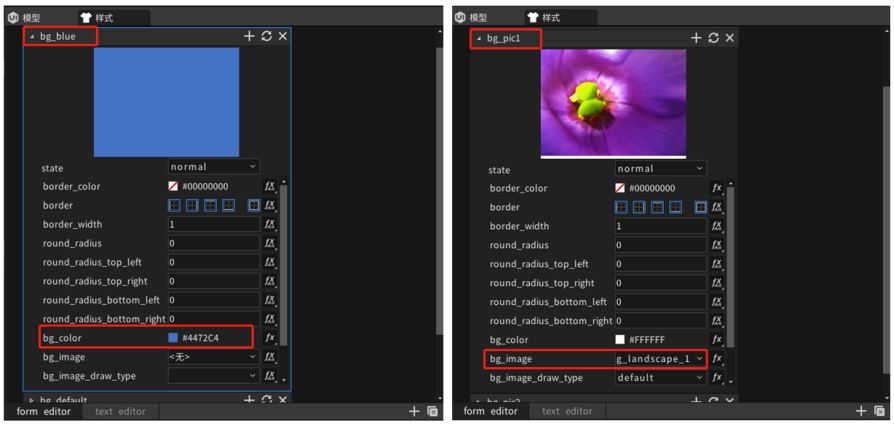


图 6.31　创建窗口样式设置不同的背景

6.3.4 List View 界面设计

点击主界面的“List View”按钮，可进入 List View 界面，如图所示，界面功能详见第二节，下面介绍如何使用 Designer 设计该界面。

1. 创建列表视图

列表视图（list_view）是一个包含滚动视图（scroll_view）和手机版滚动条（scroll_bar_m）的控件，其中的滚动视图可以放置多于屏幕可存放控件容量的容器，在运行时可通过滑动查看不在屏幕中的控件。

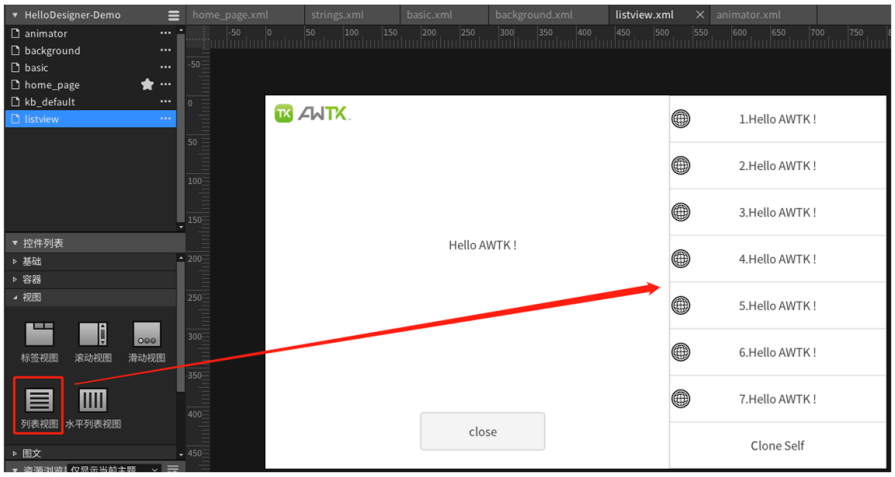


图 6.32 创建列表视图

2. 向列表视图中添加列表项

双击滚动视图控件可以进入滚动视图编辑界面，在该界面放置的控件全部为滚动视图的子控件，往滚动视图中添加多个列表项后，再设置列表项的名称和显示文本，如图所示：

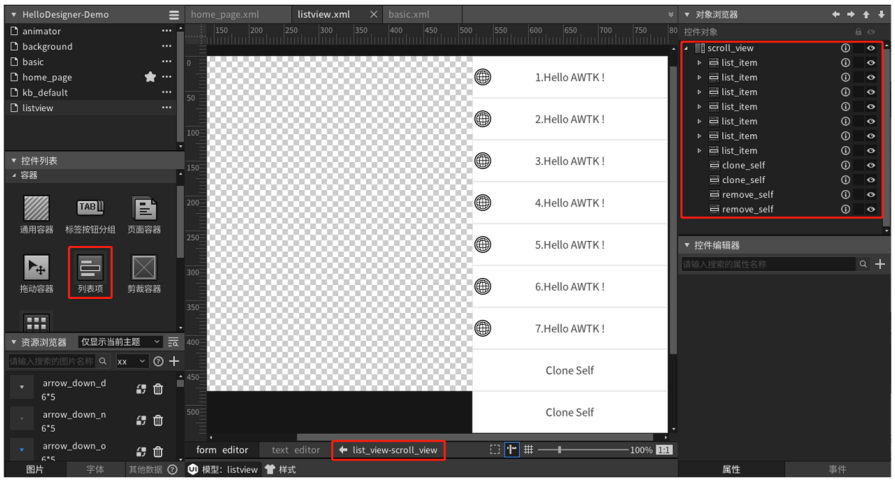


图 6.33 向列表视图添加列表项

3. 向列表项中添加图标

列表项本身也是一个容器，可以参考上一小节的内容向列表项中添加图片控件，然后添加图片资源并显示图标，并且可以将图片的绘制格式修改为图标形式，此处可将“控件编辑器”的“值”列表中的“draw_type”属性设置为“icon”，如图所示。

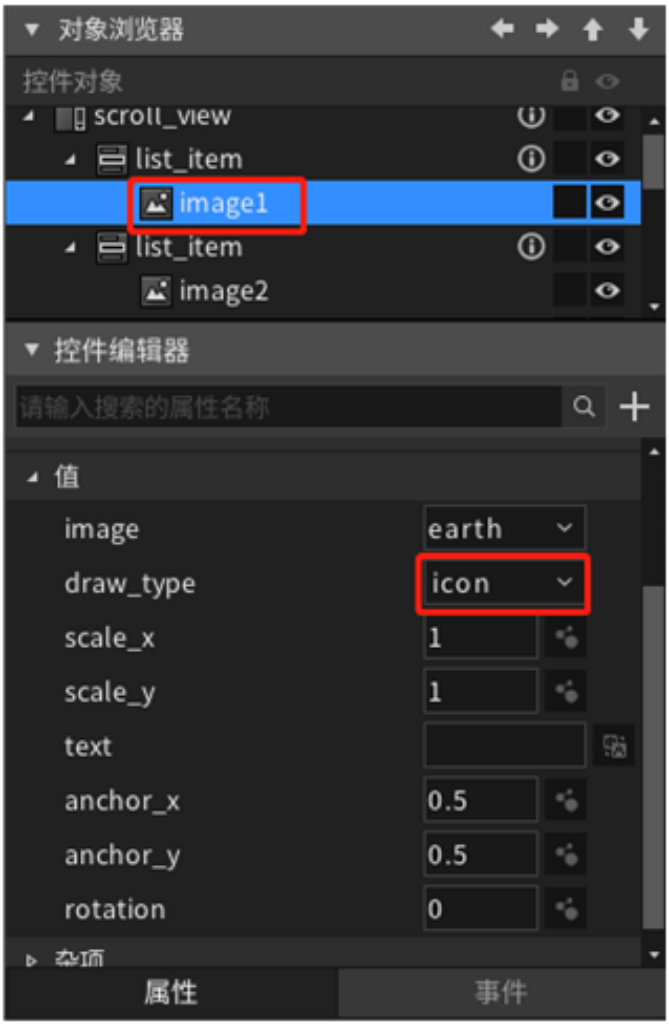


图 6.34　设置图标在图片控件的绘制方式

6.3.5　Animation Widget 界面设计

点击主界面的“Animation Widget”按钮，可进入 Animation Widget 界面，如图所示，界面功能详见第二节，下面介绍如何使用 Designer 设计该界面。

1. 添加控件动画效果

Animation 界面中包含 4 个图片控件，分别展示了位移、缩放、透明和旋转四种不同的控件动画效果，这里以位移动画为例，介绍 Designer 如何添加控件动画效果，步骤如下：

- 在“窗体编辑器”中选中图片控件；
- 在“控件编辑器”的“动画”属性里点击“＋”按钮，弹出“设置动画”对话框；
- 此处选择动画类型为“位移”，设置单词动画持续时间为 1000 ms，并勾选往返播放，往返次数设置为 0，即永久播放，同时设置图片控件在 x 方向上不变，在 y 方向上从 100 移动到 200，如图所示，其他动画效果添加步骤与此类似，这里不再赘述。（旋转动画中角度的单位为弧度）

注: 要注意区分窗口动画和控件动画，窗口动画用于窗口打开关闭时的过度效果，控件动画用于窗口内控件的动作，两者不能混用。

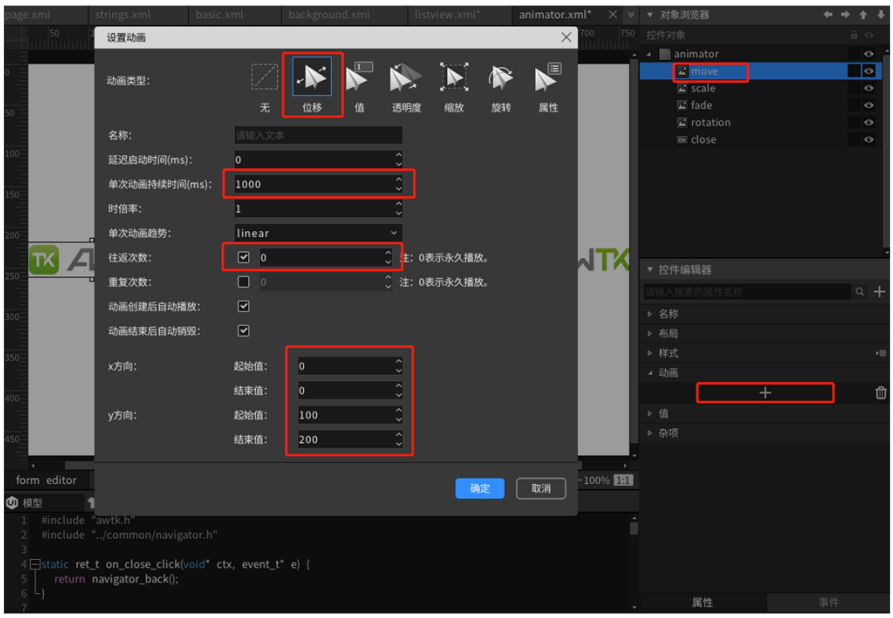


图 6.35 添加控件动画

在完成界面设计后，可以点击”工具栏”的”编辑”页面，点击”预览”查看当前的界面设计。

图 6.36 预览

注: 预览只展示界面设计，不涉及代码相关的预览，每次预览只会展示当前页面的预览结果。

6.4 HelloDesigner-Demo 代码编辑

界面设计完成后，需要编写相应的逻辑代码，本章主要讲解如何源代码中使用 AWTK 实现 HelloDesigner-Demo 项目的功能。如果想要了解更多 AWTK 编程方面的知识，请参阅 AWTK 开发实践[7]。

6.4.1 项目的源代码文件

HelloDesigner-Demo 的源代码文件在项目的 src 目录下，具体说明见下表

源文件	说明
common	存放导航器源代码，由 Desinger 自动生成
pages	存放页面源代码，使用 Designer 新建窗体时自动生成，文件名与页面名相同
application.c	存放当前项目的初始化源代码，由 Designer 自动生成
main.c	存放当前项目的应用程序入口，由 Designer 自动生成

[7] http://awtk.zlg.cn/docs/awtk_docs/AWTK_Guide/

续上表

源文件	说明
SConscript	Scons 编译脚本，由 Designer 自动生成

注: 在 main.c 包含了 awtk/src/awtk_main.inc 文件，该文件中包含了 main 函数以及与 AWTK 框架相关的初始化、事件循环等函数。

代码的编写可以在 Designer 的代码编辑器中完成，也可以使用另外的代码编辑器，例如 vscode 等，但需要注意，在 Designer 保存之后代码才会同步到另外的代码编辑器。

6.4.2 应用程序初始化

AWTK 应用程序的初始化函数为 application_init，该函数在 main.c 中声明，并在 application.c 中实现。Designer 创建项目时会自动生成，用户可在该函数中实现项目初始化，并且可以在 application_exit 函数中实现项目的反初始化，比如释放内存等。

6.4.3 主界面功能

1. 初始化函数

当用户使用 Designer 新建一个界面时，Designer 会在 pages 文件夹下生成对应的 c 代码文件，并为该文件初始化以下代码内容：

```
#include "awtk.h"
#include "../common/navigator.h"

/* 初始化窗口的子控件 */
static ret_t visit_init_child(void* ctx, const void* iter) {
  widget_t* win = WIDGET(ctx);
  widget_t* widget = WIDGET(iter);
  const char* name = widget->name;

  /* 初始化指定名称的控件（设置属性或注册事件），请保证控件名称在窗口上唯一 */
  if (name != NULL && *name != '\0') {
  }

  return RET_OK;
}

/* 初始化窗口 */
ret_t home_page_init(widget_t* win, void* ctx) {
  (void)ctx;
  return_value_if_fail(win != NULL, RET_BAD_PARAMS);
  /* 遍历窗口中子控件，visit_init_child
↪是遍历时的回调函数，每遍历一个控件就会执行一次 */
  widget_foreach(win, visit_init_child, win);

  return RET_OK;
}
```

2. 为按钮添加事件

在主界面菜单中，只有几个按钮需要绑定事件，它们的作用分别是：打开窗口和切换中英文显示，对于打开窗口关闭窗口这种简单的操作，使用 Designer 就可以完成，此处以”basic_btn”按钮为例进行演示如何添加事件，步骤如下：

- 在”窗体编辑器”点击”basic_btn”控件；
- 切换”控件编辑器”为”事件”界面；
- 点击右上角的”+”按钮，并选择想要添加的事件类型，此处为”click”点击事件；
- 选择”动作”一栏为”打开窗口”，并选择打开窗口的名称为”basic”。

图 6.37　设置按钮点击打开指定窗口

设置完成后，Designer 会自动在源代码中添加对应的代码，如下所示，其他几个打开窗口的操作与之类似，就不再赘述。

```
/* 在此处添加了一个回调函数，内容为打开窗口 */
static ret_t on_basic_btn_click(void* ctx, event_t* e) {
  return navigator_to("basic");
}

static ret_t visit_init_child(void* ctx, const void* iter) {
  widget_t* win = WIDGET(ctx);
  widget_t* widget = WIDGET(iter);
  const char* name = widget->name;

  /* 初始化指定名称的控件（设置属性或注册事件），请保证控件名称在窗口上唯一 */
  if (name != NULL && *name != '\0') {
    /* 在此处添加了查找对应名称控件并进行事件注册的操作 */
```

```
    if (tk_str_eq(name, "basic_btn")) {
      /* 为控件注册事件的回调函数 */
      widget_on(widget, EVT_CLICK, on_basic_btn_click, win);
    }
  }

  return RET_OK;
}
```

接下来为”language_btn”按钮添加回调函数用来切换按钮主题与系统语言，步骤如下：

- 在”窗口编辑器”选择”language_btn”按钮；
- 切换”控件编辑器”为”事件”页面；
- 点击右上角的”+“按钮，选择想要添加的事件类型，此处选择”click”；
- 选择”动作”一栏为”执行回调函数”。

完成操作后，Designer 会在源代码文件添加注册回调函数的代码，我们只需要在回调函数中实现切换语言的功能即可：

```
/* 中英文互译 */
static ret_t change_locale(const char* str) {
  char country[3];
  char language[3];
  strncpy(language, str, 2);
  strncpy(country, str + 3, 2);
  /* 切换语言为language，地区为country */
  locale_info_change(locale_info(), language, country);
  return RET_OK;
}

/* 中英互译按钮的事件回调函数 */
static ret_t on_language_btn_click(void* ctx, event_t* e) {
  // TODO: 在此添加控件事件处理程序代码
  widget_t* language_btn = WIDGET(e->target); /* 获取触发事件的按钮对象 */
  const char* language = locale_info()->language;  /* 获取当前语言环境 */
  if (tk_str_eq(language, "en")) {           /* 切换语言并改变按钮样式 */
    change_locale("zh_CN");
    widget_use_style(language_btn, "en");
  } else {
    change_locale("en_US");
    widget_use_style(language_btn, "zh");
  }
  return RET_OK;
}

static ret_t visit_init_child(void* ctx, const void* iter) {
  widget_t* win = WIDGET(ctx);
  widget_t* widget = WIDGET(iter);
  const char* name = widget->name;
```

```
  /* 初始化指定名称的控件（设置属性或注册事件），请保证控件名称在窗口上唯一 */
  if (name != NULL && *name != '\0') {
    if (tk_str_eq(name, "basic_btn")) {
      widget_on(widget, EVT_CLICK, on_basic_btn_click, win);
    } else if (tk_str_eq(name, "language_btn")) {
      /* 此处将窗口对象 win 作为上下文 ctx 传给事件回调函数 */
      widget_on(widget, EVT_CLICK, on_language_btn_click, win);
    }
  }

  return RET_OK;
}
```

注: AWTK API 函数具体的参数与作用，可以查阅：AWTK-API 手册[8]。

6.4.4 Basic 界面功能

1. 点击按钮关闭当前窗口

该操作与上一节的打开窗口类似，故只展示 Designer 设置完成的情况，如下图所示，其他窗口的”close”按钮同理。

图 6.38 设置按钮按下时关闭窗口

2. 点击按钮设置进度条的值

在 Basic 界面中，实现点击 dec/inc 按钮，可减少/增加水平进度条的值，步骤如下：

- 为 dec/inc 按钮添加”click”事件；
- 实现两个按钮的回调函数，代码如下：

[8] https://awtk.zlg.cn/api/awtk/

```
/* 以动画的形式改变进度条的值 */
static ret_t progress_bar_animate_delta(widget_t* win, const char* name, int32_t␣
↪delta) {
  widget_t* progress_bar = widget_lookup(win, name, TRUE); /* 获取进度条控件 */
  /* 获取进度条的值，加上差值，并限制值的范围 */
  int32_t value = (PROGRESS_BAR(progress_bar)->value + delta);
  value = tk_min(100, value);
  value = tk_max(0, value);
  /* 设置控件的值，并以动画的形式变化到指定的值 */
  widget_animate_value_to(progress_bar, value, 500);

  return RET_OK;
}

/* 递减按钮事件回调函数 */
static ret_t on_dec_btn_click(void* ctx, event_t* e) {
  widget_t* win = WIDGET(ctx);
  progress_bar_animate_delta(win, "click_bar", -10);
  return RET_OK;
}

/* 递增按钮事件回调函数 */
static ret_t on_inc_btn_click(void* ctx, event_t* e) {
  widget_t* win = WIDGET(ctx);
  progress_bar_animate_delta(win, "click_bar", 10);
  return RET_OK;
}
```

3. 响应编辑器的值正在改变事件

在 Basic 界面中，在编辑器（edit）中输入文本可同步显示到上方的静态文本中，实现步骤如下：

- 为编辑器（edit）添加”value_changing”事件；
- 实现回调函数，代码如下：

```
/* 编辑器的值变化事件回调函数 */
static ret_t on_edit_value_changing(void* ctx, event_t* e) {
  /* 获取几个需要用到的控件 */
  widget_t* target = WIDGET(e->target);
  widget_t* win = WIDGET(ctx);
  widget_t* label = widget_lookup(win, "edit_label", TRUE);
  /* 将 edit 中的文本设置为 label 控件显示的文本 */
  widget_set_text(label, target->text.str);

  return RET_OK;
}
```

注：在 Basic 界面中拖动滑动条中的滑块，可以立刻改变垂直进度条的值，该功能也是通过响应滑动条的值正在改变事件实现，读者可以举一反三，代码位于 src/pages/basic.c。

4. 响应进度条的值改变事件

Basic 界面中，水平进度条的值改变后会同步显示到 dec/inc 按钮之间的静态文本控件上，实现步骤如下：

- 为水平进度条添加”value_changed”事件；
- 实现回调函数，代码如下：

```
/* 进度条值改变事件 */
static ret_t on_click_bar_value_changed(void* ctx, event_t* e) {
  widget_t* win = WIDGET(ctx);
  widget_t* label = widget_lookup(win, "click_bar_label", TRUE);
  widget_t* bar = WIDGET(e->target);
  char text[32];
  value_t v;

  /* 获取进度条的值，格式化后将其设置为 label 控件显示的文本 */
  value_set_int32(&v, widget_get_value(bar));
  tk_snprintf(text, sizeof(text), "%d%%", value_int32(&v));
  widget_set_text_utf8(label, text);

  return RET_OK;
}
```

注: 在 Basic 界面中拖动滑动条中的滑块，在松开鼠标左键后才会改变垂直进度条的值。该功能也是通过响应滑动条的值改变事件实现，读者可以举一反三，源代码位于 src/pages/basic.c。

5. 使用定时器改变进度条的值

在 Basic 界面中，水平进度条的值会随着时间不断变化，该功能可通过 AWTK 中的定时器来实现，步骤如下：

- 在窗口初始化函数中添加定时器，并设置回调函数为 on_add_bar_value，间隔时间为 1500 ms，代码如下：

```
ret_t basic_init(widget_t* win, void* ctx) {
  (void)ctx;
  return_value_if_fail(win != NULL, RET_BAD_PARAMS);
  widget_foreach(win, visit_init_child, win);

  /* 添加定时器 */
  widget_add_timer(win, on_add_bar_value, 1500);

  return RET_OK;
}
```

- 实现定时器的回调函数 on_add_bar_value，代码如下：

```
/* 定时器回调函数（定时增加progress_bar的值）*/
static ret_t on_add_bar_value(const timer_info_t* timer) {
  widget_t* win = WIDGET(timer->ctx);
  widget_t* bar = widget_lookup(win, "click_bar", TRUE);
  value_t v;
```

```
  int32_t val;

  value_set_int32(&v, widget_get_value(bar));
  val = value_int32(&v);
  if (val >= 100) { /* 限制值范围 */
    progress_bar_set_value(bar, 0);
  }
  /* 调用上文中实现的进度条动画函数 */
  progress_bar_animate_delta(win, "click_bar", 10);

  return RET_REPEAT;
}
```

6.4.5 Background Change 界面功能

1. 点击按钮切换背景颜色或图片

在 Background Change 界面中，实现点击按钮切换对应的背景颜色或图片，在该界面中，用来切换背景或图片的按钮名称都为”style”，所以实现一个按钮的回调函数相当于实现所有按钮的回调函数，步骤如下：

- 为”style”按钮添加”click”事件；
- 实现回调函数，代码如下：

```
/**
 * 改变背景按钮事件
 */
static ret_t on_style_click(void* ctx, event_t* e) {
  // TODO: 在此添加控件事件处理程序代码
  char text[64] = "";
  widget_t* win = WIDGET(ctx);
  /* 获取触发事件的按钮 */
  widget_t* target = (widget_t*)e->target;
  /* 获取按钮的文本 */
  widget_get_text_utf8(target, text, ARRAY_SIZE(text));
  /* 根据按钮文本改变窗口样式 */
  widget_use_style(win, text);

  return RET_OK;
}
```

注: 在本界面中，由于几个按钮的逻辑功能类似，我们可以给几个按钮设置相同的 name，在遍历窗口控件时初始化这些按钮（注册事件回调），在事件回调函数中通过事件对象中的 target 属性获取当前触发事件的按钮控件，并根据控件的特征实现对应的功能。

6.4.6 List View 界面功能

1. 点击列表项设置显示文本

与上一节的按钮相似，几个列表项的功能类似，在此处也将几个列表项的名字设为相同，功能为点击该列表项时，设置左侧静态文本中的显示文本，步骤如下：

- 为”list_item”控件添加”click”事件；
- 实现回调函数，代码如下：

```
/* 普通列表项点击事件回调函数 */
static ret_t on_list_item_click(void* ctx, event_t* e) {
  // TODO: 在此添加控件事件处理程序代码
  widget_t* win = WIDGET(ctx);
  /* 获取触发事件的列表项 */
  widget_t* list_item = WIDGET(e->target);
  /* 获取 label 控件 */
  widget_t* label = widget_lookup(win, "list_view_label", TRUE);
  value_t v;
  /* 获取当前列表项的文本并赋给 label 控件 */
  value_set_wstr(&v, widget_get_text(list_item));
  widget_set_text(label, value_wstr(&v));

  return RET_OK;
}
```

2. 点击 Clone Self 添加列表项

在 List View 界面中，点击 Clone Self 列表项，可以在列表视图尾部添加一个相同的列表项，实现步骤如下：

- 为”clone_self”控件添加”click”事件；
- 实现回调函数，代码如下：

```
/* 克隆列表项点击事件回调 */
static ret_t on_clone_self_click(void* ctx, event_t* e) {
  // TODO: 在此添加控件事件处理程序代码
  widget_t* widget = WIDGET(e->target);
  /* 在列表项的父控件中克隆自己 */
  widget_t* clone = widget_clone(widget, widget->parent);
  /* 克隆控件只克隆属性，还需要在此处手动注册"click"事件 */
  widget_on(clone, EVT_CLICK, on_clone_self_click, clone);

  return RET_OK;
}
```

3. 点击 Remove Self 移除列表项

在 List View 界面中，点击 Remove Self 列表项，可以从列表视图移除该列表项，实现步骤如下：

- 为”remove_self”控件添加”click”事件；
- 实现回调函数，代码如下：

```
/**
 * 列表项移除事件
 */
static ret_t on_remove_self_click(void* ctx, event_t* e) {
  // TODO: 在此添加控件事件处理程序代码
  widget_t* widget = WIDGET(e->target);
  /* 从父控件中移除该列表项 */
  widget_remove_child(widget->parent, widget);
  /* 销毁该列表项 */
  widget_destroy(widget);

  return RET_OK;
}
```

6.4.7 编译运行 HelloDesigner-Demo

完成以上的代码编写后，可以参考第一节的步骤编译并运行 HelloDesigner-Demo 项目。

图 6.39 运行 HelloDesigner

诚信共赢，持续学习，客户为先，专业专注，只做第一

广州致远电子股份有限公司

更多详情请访问 www.zlg.cn

欢迎拨打全国服务热线 400-888-4005