新风机Demo工程实现步骤

一、 准备工作(Windows平台)

1.1 必备工具

1. **python2.7**
2. **scons**
3. **Visual Studio C++(版本>=2015)**

备注：安装以上软件时请按上面的次序安装。

备注：可选安装Visual Studio Code（建议安装：C++ Intellisense 插件，该插件主要用于跳转到定义、自动提示等）用于编写xml文件。

1.2 编译awtk

1. 下载awtk: https://github.com/zlgopen/awtk
2. 进入awtk所在目录，编辑SConstruct文件，保留：LCD='SDL'、NANOVG_BACKEND='AGGE'、FRAME_BUFFER_FORMAT='bgr565'等变量赋值，其他的赋值语句用#注释掉，代码如下：

```
LCD='SDL'       #保留这句（新版本的AWTK已经不需要定义该变量）
#LCD='GL'       #注释掉其他的

#NANOVG_BACKEND=xxx
NANOVG_BACKEND='AGGE'       #保留这句
#NANOVG_BACKEND='BGFX'      #注释掉其他的

#FRAME_BUFFER_FORMAT=xxx
FRAME_BUFFER_FORMAT='bgr565'          #保留这句
#FRAME_BUFFER_FORMAT='bgra8888'       #注释掉其他的
```

3. 在cmd下运行下列命令

```
scons
```

二、应用程序开发

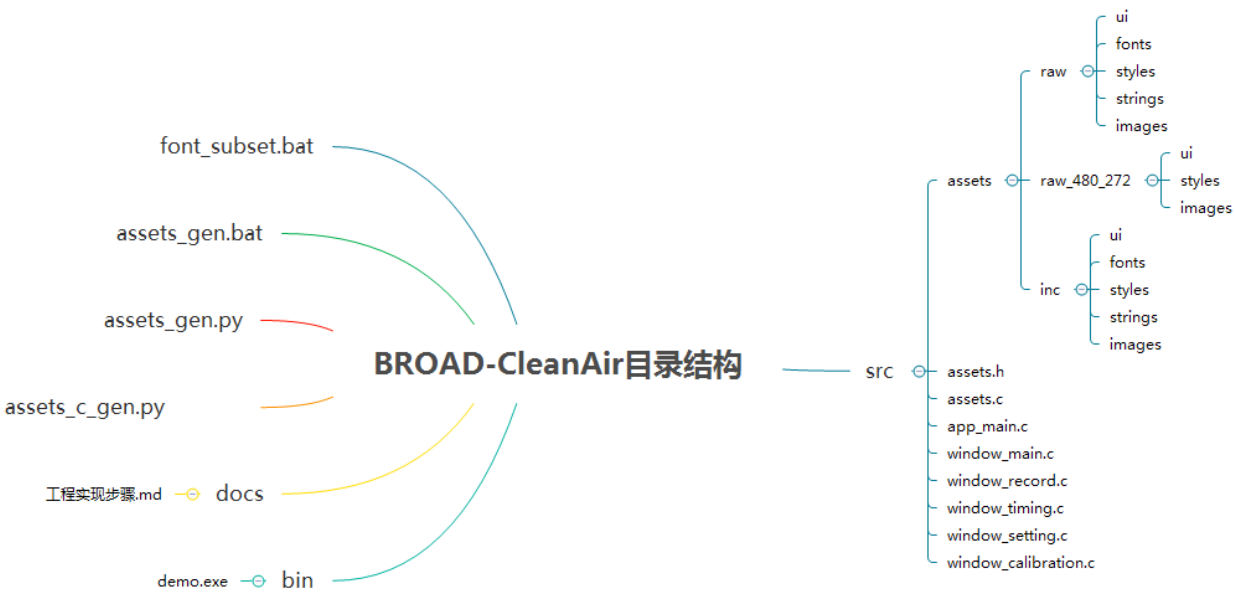
下面以远大洁净新风机Demo为例，介绍如何使用AWTK进行应用开发，该Demo主要有以下功能：

1. 点击“开关”按钮，启动/停止PM2.5、二氧化碳浓度、送风室内外温度 、排风室内外温度等模拟读数，以及模拟报警
2. 点击“自动”按钮，启动/停止背景（淡入淡出）切换
3. 点击“定时”按钮，弹出定时设置对话框，模拟开/关定时功能（即时钟图标的显示/隐藏）
4. 点击“记录”按钮，弹出记录列表（模拟数据）页面
5. 点击“设置”按钮，弹出设置页面，设置控制、报警参数
6. 点击三角形按钮，可以加/减频率、温度、湿度
7. 实时更新系统时间

   运行效果如下：

2.1 目录结构介绍

1. 在介绍具体实现细节之前，我们先看看该Demo的目录结构，如下：



项目	说明
bin	可执行文件目录
docs	说明文档
src	源代码
SConstruct	scons脚本
assets_gen.bat	双击生成资源文件（具体使用说明请打开该文件查阅）
font_subset.bat	双击将default.ttf字体文件切割成只包含text.txt文件里面字符的default.mini.ttf字符文件
assets_gen.py	被assets_gen.bat调用，生成资源文件
assets_c_gen.py	被assets_gen.bat调用，生成assets.c

2. 在AWTK中，约定src\assets\raw的目录结构如下表所示(**注意目录位置不可随意更改)**：

项目	说明
fonts	字体文件目录，AWTK目前支持TTF
images	图片文件目录（包含x1、x2、x3目录），AWTK目前支持的格式有png、jpg、bmp
strings	多国语言文件目录，包含使用xml描述文本信息在不同语言环境下的表述结果的语言文件
styles	样式文件目录，包含使用xml描述界面外观的样式文件，比如下文所介绍的default.xml
ui	UI文件目录，包含使用xml描述界面结构的ui文件，比如下文所介绍的main.xml

3. src\assets\images目录下可以建立的文件夹(**默认建立x1**)如下表所示：

项目	说明
x1	普通LCD的图片
x2	高清LCD的图片
x3	手机等超高清LCD的图片

- 对于嵌入式系统，一般只需要x1的图片。如果开发环境使用高清的PC显示器，为了方便PC上看效果，建议也准备一套x2的图片。
- 需要注意的是，AWTK中使用资源时，除了点阵字体（BITMAP_FONT）之外，资源的名称一般为文件名，且不包含文件后缀。

4. 关于Demo的资源说明

(1) 字体

可以从C:\Windows\Fonts目录下找个 ".ttf"格式的字体，比如STKAITI.TTF，拷贝到assets\raw\fonts目录下，并重命名为"default.ttf"，作为程序的默认字体。

如果字体文件过大，可以使用fonttools中的pyftsubset进行裁剪，步骤如下：

- 使用python的pip命令进行安装，在cmd下运行下列命令

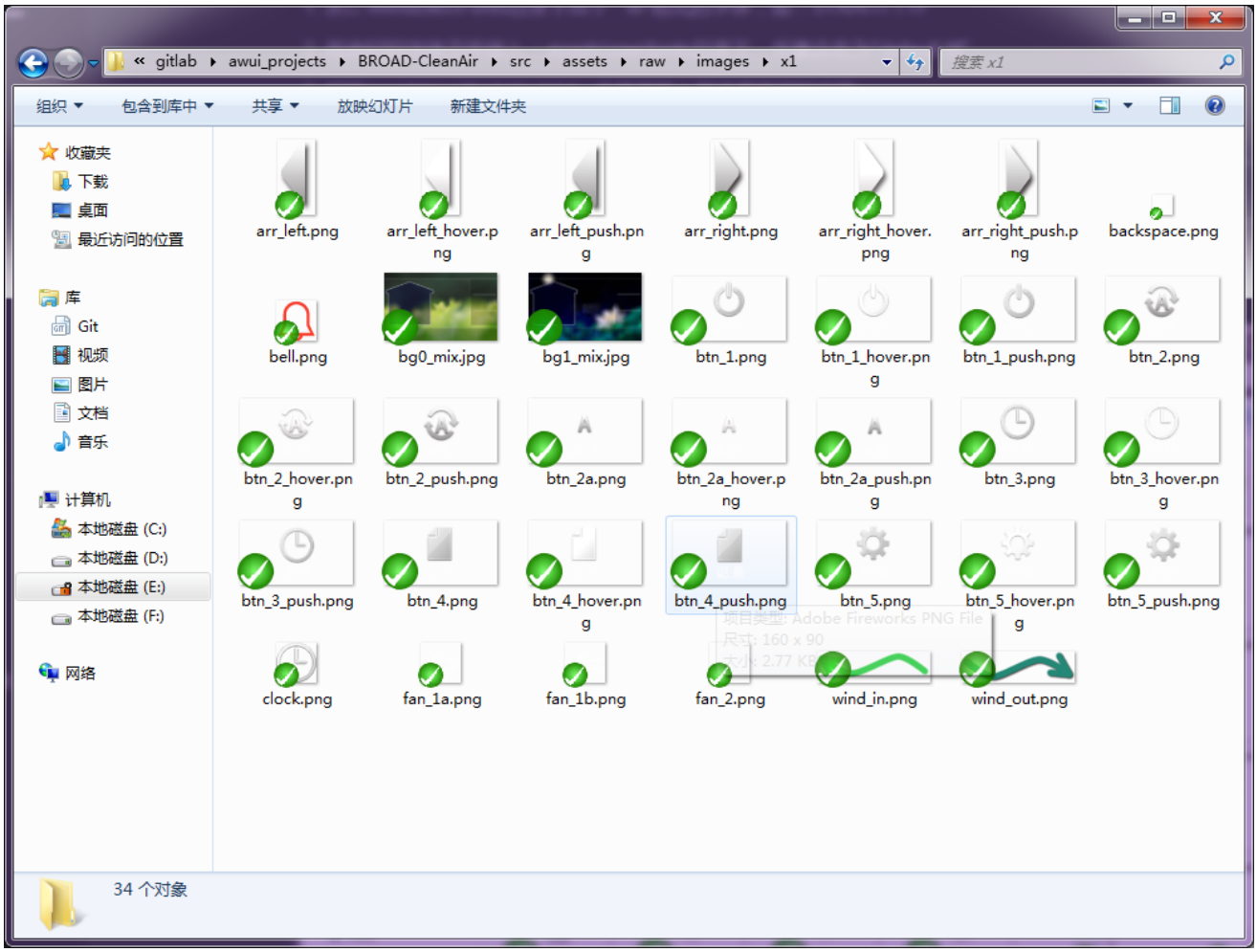
```
pip install fonttools
```

- 打开assets\raw\fonts\text.txt，在里面输入程序要使用的字符
- 使用pyftsubset工具进行裁剪，在cmd下运行下列命令，生成assets\raw\fonts\default.mini.ttf

```
font_subset.bat
```

(2)图片

可以使用Adobe Fireworks等软件切好的图片，放到assets\raw\images\x1目录下，如下图所示：



2.2 编译与执行

2.2.1 生成资源文件

1. 打开assets_gen.bat将 AWTK_BIN_DIR 修改为本地 awtk/bin 所在目录，要修改的内容如下：

```
@set AWTK_BIN_DIR=E:/zlgopen/awtk/bin
```

2. 在cmd下运行下列命令，生成主题、字符串、字体、图片、UI等资源的二进制文件，同时自动实现assets_init()，默认使用800x480的屏幕资源：

```
assets_gen.bat
```

或运行，使用480x272的屏幕资源：

```
assets_gen.bat -480_272
```

- 执行完上面命令后会生成src\assets\inc目录、assets\raw子目录下的.bin文件、src\assets.c文件
- 注意事项：如果改变（增加、删除，修改等）assets/raw目录下的资源文件，都需要重新执行该命令，并重新编译源代码

2.2.2 编译源代码

1. 打开SConstruct修改 "TK_ROOT = 'E:/zlgopen/awtk'" 为本地awtk所在目录，要修改的内容如下：

```
TK_ROOT = 'E:/zlgopen/awtk'
```

2. 在cmd下运行下列命令，默认使用800x480的屏幕资源：

```
scons
```

或运行，使用480x272的屏幕资源：

```
scons LCD=480_272
```

3. 编译后生成的exe在.\bin目录下

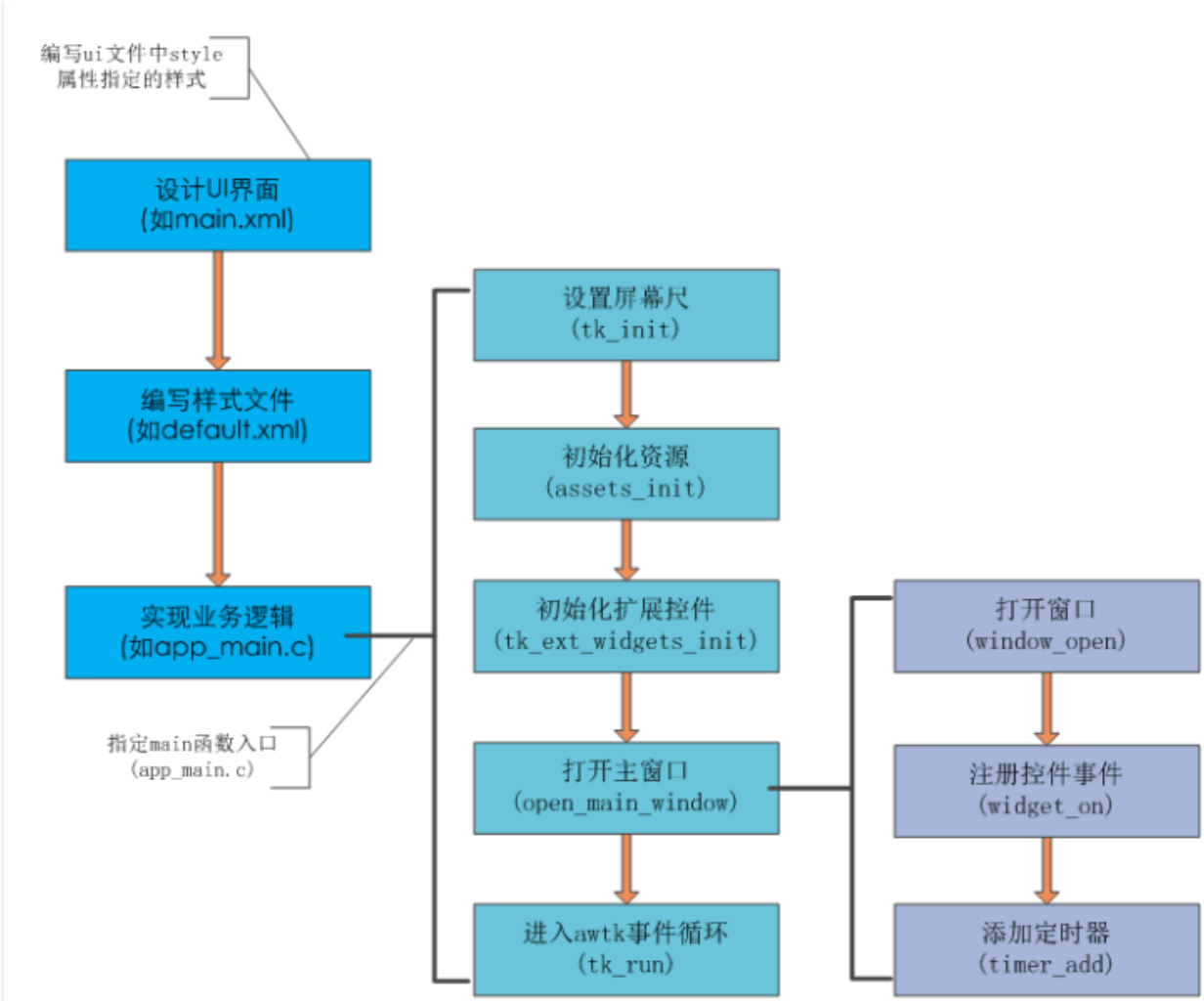
2.2.3 调试

scons编译时并没有生成Visual Studio的工程，如果需要在Visual Studio中调试AWTK应用程序，可按下列步骤进行：

1. 打开Visual Studio
2. 在『文件』菜单中点击『打开』并选中『项目/解决方案』
3. 选择bin\demo.exe (或者其它要调试的可执行文件)
4. 在项目设置中设置调试参数（可选）
5. 将要调试的文件拖到Visual Studio中，在要调试的地方打断点进行调试

2.3 应用实现

应用实现流程如下图所示：



应用实现后，就可以按2.2章节介绍的步骤编译调试。相关代码可以在BROAD-CleanAir\src目录下找到，另外各个控件的具体使用说明请看附录中对应的章节。下面介绍Demo中使用到的awtk相关功能点。

2.3.1 如何打开窗口

1. 在AWTK中，可以使用一个xml文件来描述一个窗口的界面结构。比如描述记录列表窗口的record.xml如下：

```
<!-- src\assets\raw\ui\record.xml -->
<window anim_hint="htranslate">
  <button name="close" x="4" y="4" w="60" h="30" text="返回"/>
  <view x="4" y="36" w="100%" h="40" layout="r:1 c:5">
      <label text="状态" style="header"/>
      <label text="室内温度" style="header"/>
      <label text="室外温度" style="header"/>
      <label text="室内湿度" style="header"/>
      <label text="室外湿度" style="header"/>
  </view>
  <list_view x="4"  y="76" w="-8" h="400" item_height="40">
    <scroll_view name="view" x="0"  y="0" w="100%" h="100%">
      <list_item style="odd_clickable" layout="r1 c0">
        <image draw_type="icon" w="20%" image="bell"/>
        <label w="20%" text="24.4℃"/>
        <label w="20%" text="26.4℃"/>
```

```
        <label w="20%" text="20%"/>
        <label w="20%" text="30%"/>
      </list_item>
      ......
    </scroll_view>
    <scroll_bar_m name="bar" x="right" y="0" w="6" h="100%" value="0"/>
  </list_view>
</window>
```

- 在xml中：

  window： 表示窗口元素

  htranslate：表示水平平移

2. 然后在代码中使用window_open( ) 即可打开该窗口，代码如下：

```
//src\window_record.c
widget_t* win = window_open("record");
```

其中，“record”为xml的文件名，运行后的效果如下：

3. 如果需要手动关闭窗口使用window_close()即可，代码如下：

```
//src\window_record.c
window_close(win);
```

2.3.2 如何打开对话框

1. 比如创建一个定时按钮窗口，代码如下：

```
<!-- src\assets\raw\ui\timing.xml -->
<dialog anim_hint="center_scale" h="200" w="580" x="c" y="m">
    <dialog_title h="35" text="定时设置" w="100%" x="0" y="0"/>
    <dialog_client h="-35" w="100%" x="0" y="bottom">
        <row h="30" layout="row:1 col:3" w="200" x="30" y="10">
            <label style="timing" text="时"/>
            <label style="timing" text="分"/>
            <label style="timing" text="秒"/>
        </row>
        <row h="120" layout="row:1 col:3" w="200" x="30" y="40">
            <text_selector options="1-24" text="15" visible_nr="3"/>
            <text_selector options="1-60" text="10" visible_nr="3"/>
            <text_selector options="1-60" text="16" visible_nr="3"/>
        </row>
        <button h="30" name="ok" style="timing_switch_btn" text="开" w="150"
x="right:30" y="33"/>
        <button h="30" name="cancle" style="timing_switch_btn" text="关" w="150"
x="right:30" y="96"/>
    </dialog_client>
</dialog>
```

- 在xml中：

  h：表示高度

  w：表示宽度

  x：值c表示水平居中

  y：值m表示垂直居中

  dialog：表示对话框元素

  dialog_title：表示标题栏

  dialog_client：表示对话框主界面

  center_scale：表示中心缩放

2. 然后在代码中使用window_open( ) 即可打开该窗口，代码如下：

```
//src\window_timing.c
widget_t* win = window_open("timing");
```

其中，“timing”为xml的文件名，运行后的效果如下：

3. 如果需要手动关闭窗口使用dialog_quit()即可，代码如下：

```
//src\window_timing.c
dialog_quit(dialog, RET_OK);
```

2.3.3 如何查找控件

AWTK中可通过控件的name属性查找该控件，代码如下：

```
<!-- src/assets/raw/ui/main.xml -->
<image name="timing_status" image="clock" draw_type="icon"/>
```

```
// src/window_main.c
widget_t* timing = widget_lookup(win, "timing_status", TRUE);
```

- 在xml中，定义定时图标的name为“timing_status”
- 在C代码中，把该图标的name作为widget_lookup()参数，获取定时图标的句柄

2.3.4 如何响应界面事件

比如在主界面上设置开关按钮的点击事件，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<button name="switch" x="0" w="147" h="100%" style="switch_btn"/>
```

```
//src\window_main.c
...
widget_t* win = widget_get_window(widget);
widget_on(widget, EVT_CLICK, on_switch, win);
...
```

- 在xml中，设置开关按钮控件，其中

  button：表示按钮控件元素

- 在C代码中，查找到按钮句柄后，使用widget_on为"EVT_CLICK"点击事件注册一个回调函数（比如on_switch）即可

2.3.5 如何设置控件是否可见

AWTK中可通过控件的visible属性控制该控件是否可见。比如控制定时图标是否可见，代码如下：

```
// src/window_main.c
widget_t* timing = widget_lookup(win, "timing_status", TRUE);
if (open_timing_window(&t) == RET_OK) {
  widget_set_visible(timing, TRUE, FALSE);
} else {
  widget_set_visible(timing, FALSE, FALSE);
}
```

- 在C代码中，使用widget_lookup()查找到定时图标后，再使用widget_set_visible()修改visible属性即可

2.3.6 如何设置窗口顶部容器

比如在主界面上显示"远大洁净新风机"，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<app_bar x="0" y="0" w="100%" h="40">
        <label name="dev_name" x="0" y="0" w="200" h="100%" text="远大洁净新风机"
style="title_left"/>
        <label name="sys_time" x="200" y="0" w="-200" h="100%" style="title_right"/>
 </app_bar>
```

- 在xml中：

  style：表示使用title_right的样式

  w：值100%自动换算成相对其父控件宽度的百分比

  app_bar：表示容器元素，只是让xml更具有可读性，并不具有实际性意义

2.3.7 如何显示文本

比如在主界面上显示"PM2.5"对应的值18，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<label name="PM2_5" x="80" w="108" h="100%" text="18" style="reading_pm_co"/>
```

```
//src\window_main.c
...
widget_t* pm2_5 = widget_lookup(win, "PM2_5", TRUE);
...
widget_set_text_utf8(label, tk_itoa(text, sizeof(text), val));
```

- 在xml中：

  label：表示文本元素

  text：表示要显示的文本

  h：值100%自动换算成相对其父控件高度的百分比

- 在C代码中，想要修改主界面上显示"PM2.5"对应的数值，使用widget_lookup获取"PM2_5"label控件的句柄，然后使用widget_set_text_utf8设置对应控件的值

2.3.8 如何显示图片

比如在主界面上显示排风图片(fan_2.png)，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<image name="fan_2" image="fan_2" draw_type="default" y="m"/>
```

- 在xml中：

  image：表示图片元素，image="fan_2"中的fan_2表示src\assets\raw\images\x1\fan_2.png

  draw_type：表示缺省显示，将图片按原大小显示在目标矩形的左上角

2.3.9 如何显示富文本

比如在主界面上显示μg/m^3分两部分处理：显示μg/m和显示上标数字3，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<rich_text x="188" w="-188" h="100%" text="<font color=&quota;white&quota;
align_v=&quota;top&quota; size=&quota;18&quota;>μg/m</font><font
color=&quota;white&quota; align_v=&quota;top&quota; size=&quota;10&quota;>3</font>" />
```

- 在xml中：

  rich_text：表示富文本元素

2.3.10 如何使用布局

比如在主界面上以一行两列的方式显示时钟(clock.png)和报警图标(bell.png)，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<view x="0" y="44" w="88" h="44" layout="r:1 c:2">
       <image name="timing_status" image="clock" draw_type="icon"/>
       <image name="alarm_status"  image="bell"  draw_type="icon"/>
</view>
```

- 在xml中：

  r：值1表示一行

c：值2表示两列

layout：表示布局元素

view：表示视图容器元素

2.3.11 如何实现动画

1. 控件动画API：

   AWTK中，可以通过该控件绑定一个动画对象来实现动画。比如实现左下角风扇图标(fan_2.png)旋转，代码如下：

```
//src\window_main.c
widget_animator_t* widget_animator_get(widget_t* widget, const char* name, uint32_t
duration, bool_t opacity) {
  widget_animator_t* animator = NULL;
  value_t val;
  value_set_pointer(&val, NULL);
  if (widget_get_prop(widget, name, &val) != RET_OK || value_pointer(&val) == NULL) {
    if (opacity) {
      animator = widget_animator_opacity_create(widget, duration, 0, EASING_SIN_OUT);
    } else {
      animator = widget_animator_rotation_create(widget, duration, 0, EASING_LINEAR);
    }
    value_set_pointer(&val, animator);
    widget_set_prop(widget, name, &val);
  } else {
    animator = (widget_animator_t*)value_pointer(&val);
  }
  return animator;
}
...
animator = widget_animator_get(fan_2, "animator", 4000, FALSE);
widget_animator_rotation_set_params(animator, 0, 2*3.14159265);
widget_animator_set_repeat(animator, 0);
widget_animator_start(animator);
```

- 使用widget_animator_rotation_create创建旋转动画对象
- 使用widget_animator_rotation_set_params设置动画旋转角度
- 使用widget_animator_set_repeat设置动画重复次数(0表示无线循环)
- 使用widget_animator_start开始动画

又或者，实现风向箭头的淡入淡出动画 (wind_in.png)，代码如下：

```
//src\window_main.c
animator = widget_animator_get(wind_out, "animator", 1000, TRUE);
widget_animator_opacity_set_params(animator, 50, 255);
widget_animator_set_yoyo(animator, 0);
widget_animator_start(animator);
```

- 使用widget_animator_opacity_create创建透明度动画对象
- 使用widget_animator_set_yoyo设置为yoyo模式(0表示无线循环)，yoyo的次数，往返视为两次

2. image_animation组件

   在AWTK中，可以使用image_animation组件来实现一组图片顺序播放的动画。比如设置主界面上fan_1a.png和fan_1b.png图片动画

```
<!-- src/assets/raw/ui/main.xml -->
<image_animation name="fan_1" image="fan_1" sequence="ab" auto_play="false"
interval="500" delay="100" y="m"/>
```

```
//src\window_main.c
image_animation_play(fan_1);
```

- 在xml中，定义交替显示fan_1a.png和fan_1b.png，其中

  auto_play：表示是否自动播放

  image：表示要显示的图片名称的前缀

  interval：表示每张图片播放的时间(毫秒)

  delay：表示自动播放时延迟播放的时间(毫秒)

  image_animation：表示图片动画控件元素

  sequence：表示播放的序列，字符可选值为:0-9,a-z,A-Z，如：fan_1a.png，fan_1b.png
- 在C代码中，使用image_animation_play播放动画

2.3.12 如何定时刷新界面数据

比如实时更新主界面右上角的系统时间，代码如下：

```
//src\window_main.c
timer_add(on_systime_update, win, 1000);
```

- 在C代码中使用timer_add向系统添加定时器，其中

  on_systime_update：表示回调函数

  win：表示回调函数的上下文

  1000：表示时间间隔(毫秒)

2.3.13 如何使用时分秒控件

比如点击主界面中的"定时"按钮弹出"定时设置"界面显示时分秒控件，代码如下：

```
<!-- src\assets\raw\ui\timing.xml -->
<row h="120" layout="row:1 col:3" w="200" x="30" y="40">
    <text_selector options="1-24" text="15" visible_nr="3"/>
    <text_selector options="1-60" text="10" visible_nr="3"/>
    <text_selector options="1-60" text="16" visible_nr="3"/>
</row>
```

- 在xml中：

  text_selector：表示文本选择元素

row：表示水平布局，row:1 col:3表示一行三列

2.3.14 如何显示列表

比如点击主界面中的"记录"按钮弹出的列表视图，代码如下：

```
<!-- src\assets\raw\ui\record.xml -->
<list_view x="4"  y="76" w="-8" h="400" item_height="40">
    <scroll_view name="view" x="0"  y="0" w="100%" h="100%">
      <list_item style="odd_clickable" layout="r1 c0">
        <image draw_type="icon" w="20%" image="bell"/>
        <label w="20%" text="24.4℃"/>
        <label w="20%" text="26.4℃"/>
        <label w="20%" text="20%"/>
        <label w="20%" text="30%"/>
      </list_item>
     ...
    </scroll_view>
    <scroll_bar_m name="bar" x="right" y="0" w="6" h="100%" value="0"/>
  </list_view>
```

- 在xml中：

  list_view：表示列表视图元素

  scroll_view：表示滚动视图

  list_item ：表示列表中的一个子项

  scroll_bar_m：表示手机版的滚动条元素

2.3.15 如何实现分页

1. 比如点击主界面中的"设置"按钮弹出分页视图，代码如下：

```
<!-- src\assets\raw\ui\setting.xml -->
<window anim_hint="htranslate" name="setting_page">
  <button h="30" name="close" text="返回" w="60" x="4" y="4"/>
  <pages h="440" w="80%" x="right" y="40">
    <view h="60%" layout="r:4 c:3" w="100%">
      <label style="setting_page" text="室内温度"/>
      <edit auto_fix="true" input_type="ufloat" max="150" min="22.0" right_margin="16"
step="0.1" text="22.0"/>
     ...
  </pages>
  <list_view auto_hide_scroll_bar="true" h="440" item_height="40" w="20%" x="left"
y="40">
    <scroll_view h="100%" name="view" w="-12" x="0" y="0">
      <tab_button text="控制参数设定" value="true"/>
      <tab_button text="报警设置"/>
    </scroll_view>
    <scroll_bar_d h="100%" name="bar" value="0" w="12" x="right" y="0"/>
  </list_view>
</window>
```

- 在xml中：

  pages ：表示右边的分页，里面包含的view代表一个分页

  tab_button：表示左边的按钮

  scroll_bar_d：表示桌面版的滚动条元素

2. 运行效果如下：

2.3.16 如何设置控件样式

比如设置主界面中的"开关"按钮的样式，代码如下：

```
<!-- src\assets\raw\ui\main.xml -->
<button name="switch" x="0" w="147" h="100%" style="switch_btn"/>
```

```
<!-- src\assets\raw\styles\default.xml -->
<button>
    ...
    <style name="switch_btn">
        <normal icon="btn_1"/>
        <pressed icon="btn_1_push"/>
        <over icon="btn_1_hover"/>
      </style>
    ...
</button>
```

```
//src\window_main.c
widget_t* btn = widget_lookup(win, "auto", TRUE);
widget_use_style(btn, "auto_btn_2a");
```

- 在ui文件main.xml中设置开关按钮，使用style指定要加载想要的样式，运行时awtk会去src\assets\raw\styles\default.xml中寻找button下的switch_btn样式，具体的style加载顺序请看附录中的 主题样式 章节
- 在样式文件default.xml中为button添加一个新的样式switch_btn，其中

  btn_1：表示src\assets\raw\images\x1目录下的图片文件

  normal：表示按钮正常情况下的样式

  pressed：表示鼠标按下时的样式

  over：表示鼠标经过时的样式
- 在C代码中，使用widget_use_style设置样式

三、如何把应用移植到AWorks

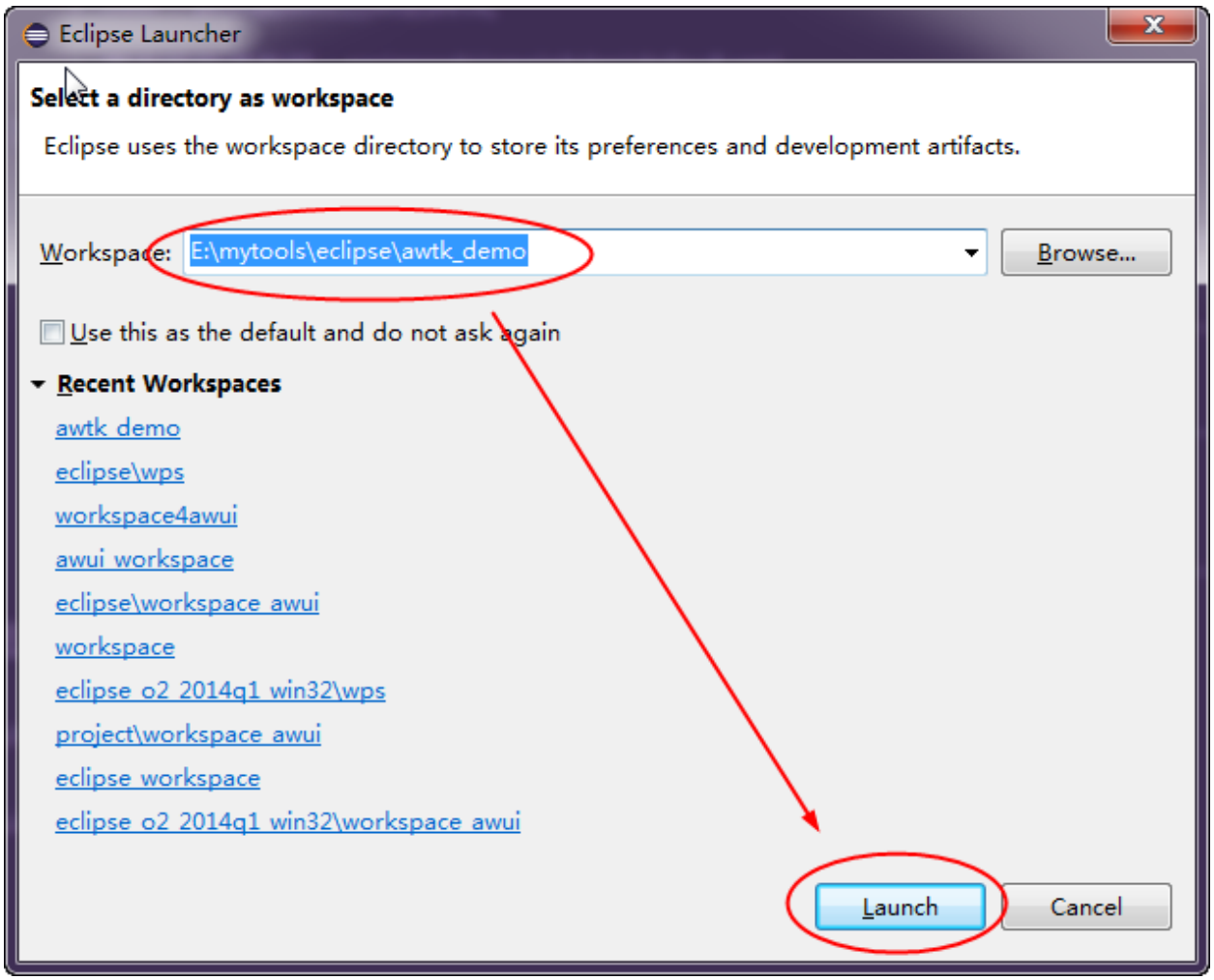
3.1 复制文件

1. 到zlg官网(http://www.zlg.cn/ipc/down/down/id/217.html)下载AWorks AWTK 模板工程，选择"M1052系列光盘资料V1.02"，如下图所示：

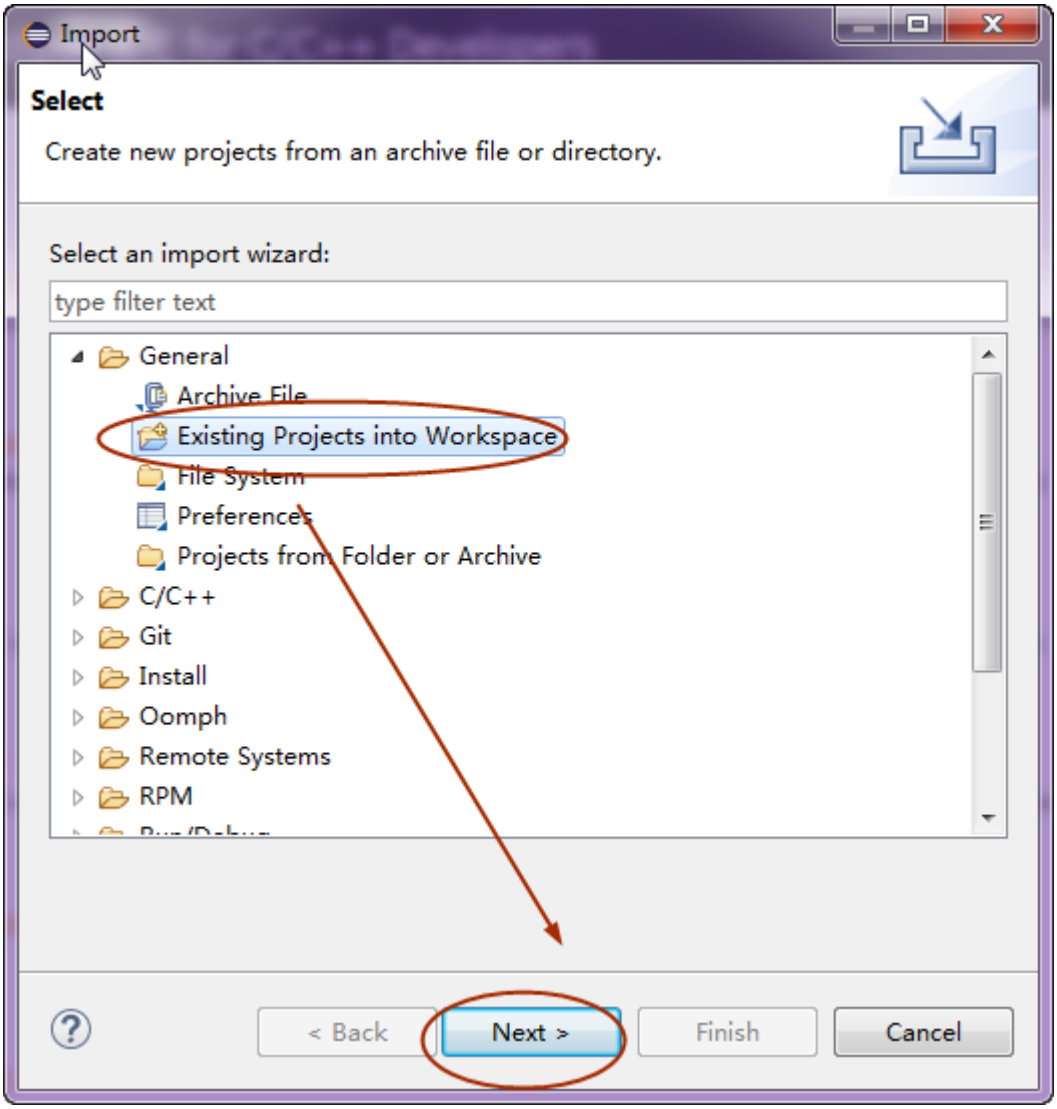
2. 下载完后，找到aworks_m105x_sdk_1.0.2-alpha文件夹并解压
3. 删除works_m105x_sdk_1.0.2-alpha\examples\application\app_awtk_demo\src\awtk-demo目录下的文件
4. 将BROAD-CleanAir\src目录下的assets文件夹、*.h*、.c文件复制到aworks_m105x_sdk_1.0.2-alpha\examples\application\app_awtk_demo\src\awtk-demo目录下

3.2 建立工程

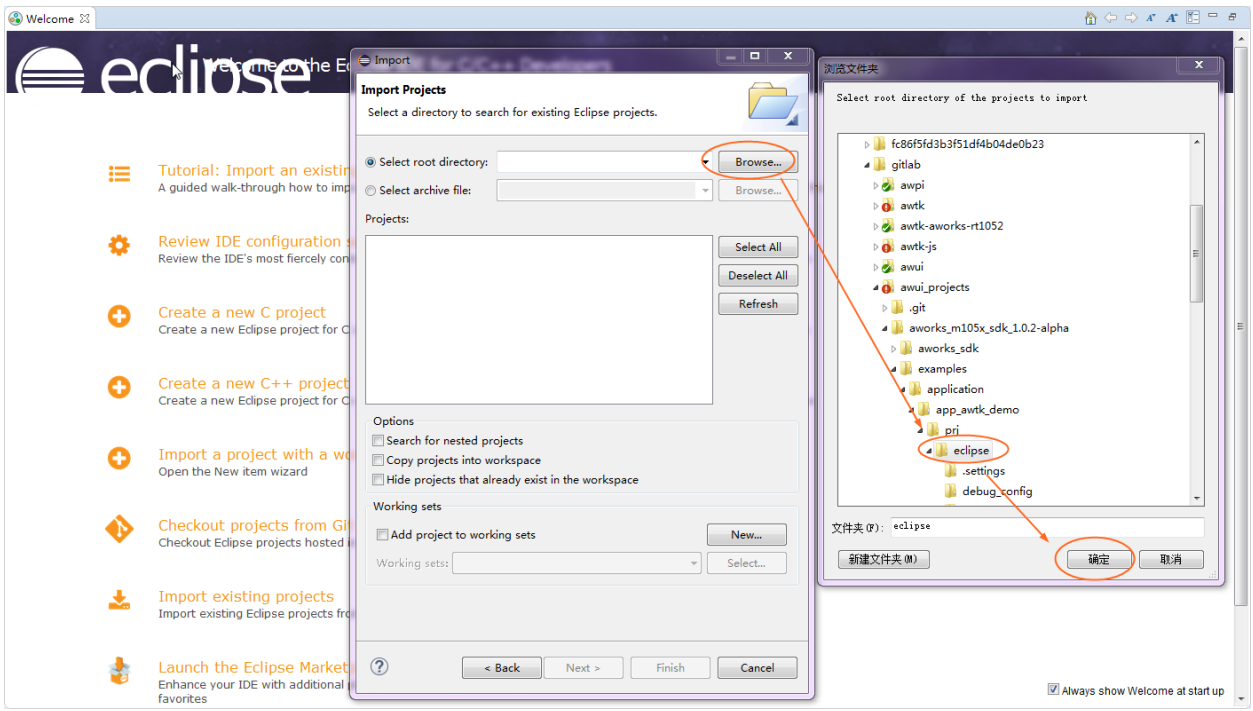
1. 打开eclipse后建立workspace，可以随便设置一个路径，比如：E:\mytools\eclipse\awtk_demo



2. 选择菜单File --> Import后，按下图操作：



3. 选择aworks_m105x_sdk_1.0.2-alpha\examples\application\app_awtk_demo\prj\eclipse所在目录，如下图所示：



4. 选择Projects下的目标工程后，点击"Finish"按钮导入
5. 导入成功后，即可编译，编译完成后烧写到开发板就可以了

备注：如果有兴趣了解AWTK是如何移植到AWorks平台，则请下载awtk-aworks-rt1052: https://github.com/zlgopen/awtk-aworks-rt1052，参阅README.md文档即可。

四、附录

1. 控件布局
2. UI控件
3. 主题样式
4. 事件处理
5. 定时器
6. 控件动画