# 安装手册

跨平台智能终端中间件z11目前支持Win32、iOS和Android系统，跨平台智能终端应用开发中间件（以下简称中间件）支持不同智能操作系统中的应用开发，你可以通过创建不同平台的工程文件，来编译同一套源代码文件，而实现在不同平台运行统一用户体验的应用程序。

中间件的发布包除了包含有中间件引用库目录外，还直接含有各个平台的工程文件模板，创建新的应用工程可以通过修改工程文件模板来实现。

中间件发布包包含以下内容：

* Readme.txt 工程说明，至少包含中文应用名称
* code 应用的源代码目录
* BuildResult 应用的生成目录
* BuildResult/Resources 应用程序的资源文件目录
* Workspace 工程配置文件目录
* z11 跨平台应用开发中间件引用库目录

因为中间件实现了在不同平台的代码统一，因此可以实现在Win32桌面平台来进行应用开发，而需要在终端部署运行时再去配置iOS和Android的工程，以下章节主要说明在应用源代码和资源文件都有的情况下，如何配置不同平台的工程文件，以下是各个系统平台的应用工程配置说明。

## Win32工程配置

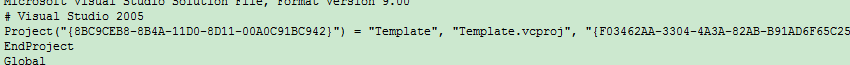
因为中间件使用了OpenGLES作为核心绘制引擎，因此在Win32系统平台上需要安装支持OpenGLES的模块，安装步骤如下：

* 双击ExternalLib目录下的OpenGL\_ES\_Emulator\_1.3.0.7490\_Win32 .msi文件进行安装。
* 安装完毕后找到其安装目录X:\Program Files\ARM\Mali Developer Tools\OpenGL ES Emulator v1.3.0 (X为安装时选择的盘符，默认为C:)。
* 把lib目录下的文件拷贝到C:\Program Files\Microsoft Visual Studio 8\VC\lib目录下。
* 把include目录下的文件拷贝到C:\Program Files\Microsoft Visual Studio 8\VC\include目录下
* 把bin目录下的文件拷贝到C:\WINDOWS\system32目录下。
* 其中C:\Program Files\Microsoft Visual Studio 8\VC为目标系统中VS2005的安装目录。
* 至此安装OpenGLES桌面模拟器完毕。

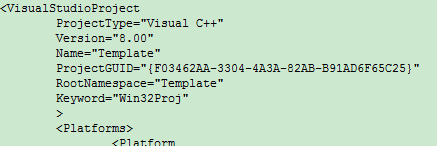
工程配置说明如下：

在z11/Distribute中拷贝某一版本至Apps/Demo（即自己项目文件夹下）

然后进入Apps/Demo/Workspace/Win32目录，把Template.sln与Template.vcproj更名为自己的工程名，接着用编辑器打开Template.sln，把下图中的Template改成工程名。



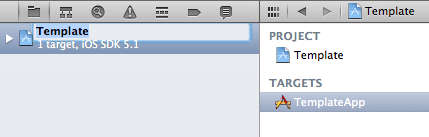
把Template.vcproj中的Template也改成工程名，双击打开该sln文件即可。



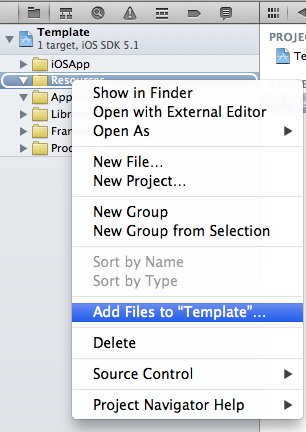
然后在工程导入源文件，并在BuildResult\Resources加入应用所需的资源文件。即可开始运行自己的应用程序。

## iOS工程配置

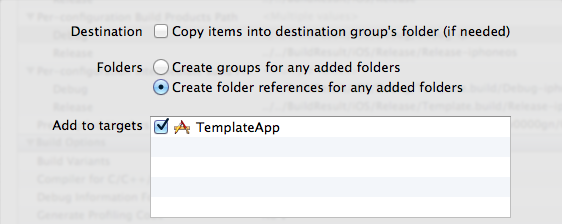
在Mac电脑上安装XCode4后，双击打开Workspace/iOS/Template.xcodeproj工程模板文件，在左上角工程名称位置双击后，修改工程名称为需要的应用名称。



修改之后，会提示关联的修改内容，选择确定完成修改。



在Resources上右键选择添加文件到工程，将资源文件添加进来，资源文件添加时需要选中Create folder references for any added folders.



右键选择Application选择添加文件到工程，找到对应的code目录下源代码添加进来，源代码添加注意选择Create groups for any added folders.

资源和源代码添加完成后，即可选择编译运行。

## Android工程配置

在Windows的Android开发平台，需要使用cygwin和NDK进行开发，分别安装cygwin和NDK。针对每个应用代码，编写对应的makefile文件，然后通过cygwin编译好对应的C++模块，然后在Android工程配置中增加对应编译好的JNI即可完成工程配置。

# 使用说明

利用跨平台智能终端中间件进行应用开发，只需要关注使用的开发语言为C++，利用的所有系统特性都是基于开发中间件提供的开发库进行开发，所以不需要关心各个平台的开发过程，只需要关心应用和中间件的对接方法及开发模式，以下章节是进行应用开发的使用说明。

## 开始开发

想要开始开发基于z11的应用，必须在应用代码中申明并实现一个类GUIRootViewController，且名字不能改动，必须是这个名称，这就是应用与z11中间件关联开发的关键类。以下是这个类的常规写法，必须继承于MIViewController基类。

class GUIRootViewController : public MIViewController{};

在自定义的GUIRootViewController类中实现基类中的两个虚函数方法即可实现应用的初始化和页面加载，两个虚函数如下：

/\*通过MIX文件构建MIViewController，MIX文件是Mobile Interface XML的简写，是符合XML规范的文件，可以用图形化工具生成\*/

virtual MIViewController\* initWithMixNameInBundle(const string& name, const MGBundle\* bundle);

virtual MIViewController\* initManually();

一般情况下，应用开发只需要在initManually()函数的实现中增加应用的实际代码即可。

## 程序结构

应用开发围绕控制器Controller和页面View进行，应用逻辑基于MVC的开发思想，在中间件中控制器的基类是MIViewController，页面的基类是MIView。

### 控制器Controller

控制器一般负责实现特定的功能处理，每一个Controller都有一个View与之对应，Controller可以用于操作和该View绑定的各种数据。

### 页面View

页面是整个应用UI交互和展现的核心，View的组织结构是树形的，GUIRootViewController含有一个view，这个view为root view，即根view，应用中的所有view都需要包含在root view上才能进行显示。每个页面view都可以有子页面即subview，每个subview又能有自己的subview，除了root view外，每个view都有自己的父页面即super view，由此构成了整个view的树形结构。

## 控件说明

所有的控件都继承于MIView，因为要进行显示必须添加到view的树形结构中。MIView有常规的方法可以进行初始化和设置大小和位置。

### MIView

void initWithFrame(MGRect frame);

说明：以指定的大小和位置初始化页面。

void setFrame(const MGRect& frame);

说明：设置大小和位置。

void setCenter(const MGPoint& center);

说明：设置中心点位置。

void addSubview(MIView\* subview);

说明：添加子页面。

void insertSubviewAtIndex(MIView\* subview, uint\_32 index);

说明：在指定位置插入子页面。

void detachFromSuperview();

说明：从本页面的父页面中删除本页面，从父View脱离，不会调用自身的析构。

MIView\* subviewAtIndex(uint\_32 index);

说明：返回指定位置的页面。

MIView\* removeSubview(MIView\* subview);

说明：将指定View从当前View删除，如果指定View自动释放，则返回NULL，否则返回为指定View。

MIView\* detachSubviewAtIndex(uint\_32 index);

说明：将指定位置的View从当前View中脱离，如果指定View为自动释放，则返回NULL，否则返回指定位置的View

MIView\* subviewWithTag(uint\_32 tag);

说明：返回指定Tag标签的页面。

void removeAllSubviews();

说明：删除所有子页面。

### MIButton

按钮。

### MIScreenCapture

屏幕捕捉器。

### MIActionSheet

可操作动作选单。

### MIActivityIndicator

活动指示器。

### MIAlertView

提示警告框。

### MIBarButtonItem

信息栏按钮。

### MIBarItem

信息栏控件。

### MIBaseTextView

文本输入控件基类。

### MICursor

光标。

### MIDatePicker

日期选取器。

### MIEditableTextView

可编辑文本框接口基类。

### MIImageView

图像控件。

### MILabel

标题控件。

### MIMapView

地图控件。

### MINavigationBar

导航栏。

### MINavigationItem

导航栏控件。

### MIProgressView

进度条控件。

### MIRoundedRectButton

圆角按钮。

### MISearchBar

搜索栏。

### MISegmentedControl

分段选择控件。

### MITableView

表格控件。

### MITextArea

文本域控件。

### MITextField

文本框控件。