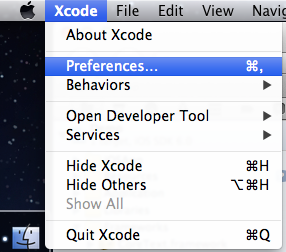
z11应用工程配置方法

文档说明了基于z11跨平台中间件框架进行应用开发时，在各个平台的应用工程配置方法，你可以选择下载z11的源码并编译或者直接下载z11的发布版本包进行应用开发，本文档从原理上说明了z11的库和应用的关系，以及如何配置应用工程包含z11的开发库。

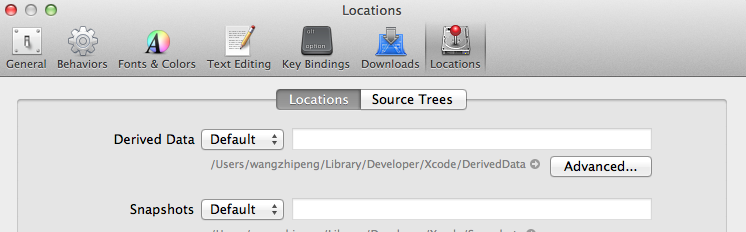
# XCode工程配置

需要在XCode中配置工程引用路径的引用规则，在XCode4中，工程引用的路径是可以修改的，需要设置成基于目标设置的路径。

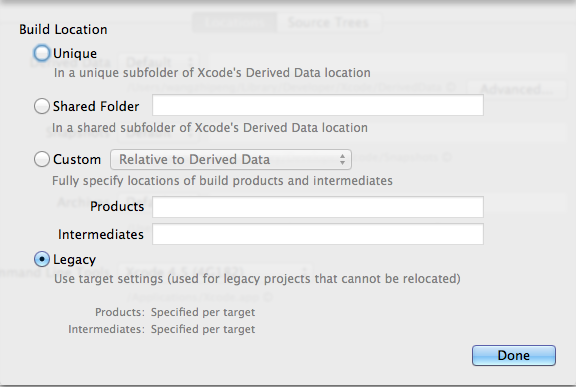
打开XCode设置，如下图：



选择Locations，Derived Data的Advanced，如下图：



选择Build Location为Legacy，如下图：

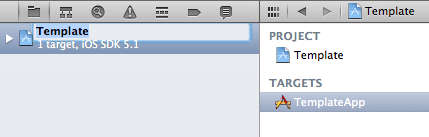


这样才能保证工程的引用路径都是基于工程的Target Settings的。

# iOS应用工程配置

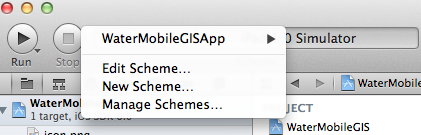
请严格按照你的电脑上的Apps和z11的相对路径配置以下提到的相对路径。

首先从z11的安装目录或者源码目录中找到Workspace/XCode4目录，将该目录下的Template.zip解压后得到应用工程模板。将解压后的目录中的文件放置在需要建立的应用工程目录的proj.iOS目录下。双击Template.xcodeproj文件，启动XCode开发工具，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。

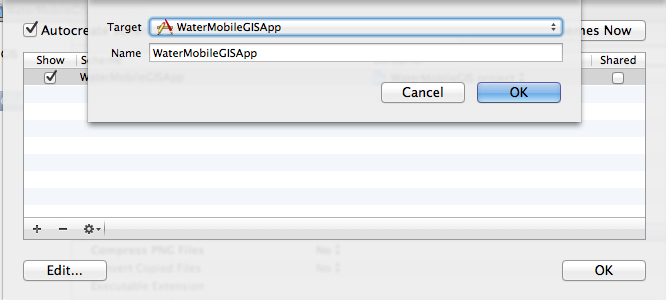


修改之后，会提示关联的修改内容，选择确定完成修改。

XCode左上角选择Manager Schemes，如下图所示：

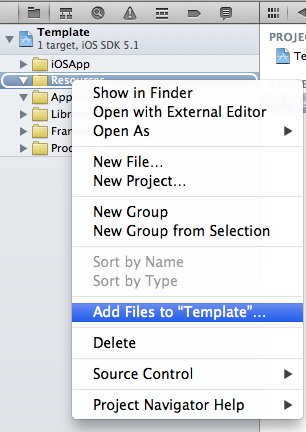


删除原来的Scheme，并选择添加新的Scheme，如下图所示：

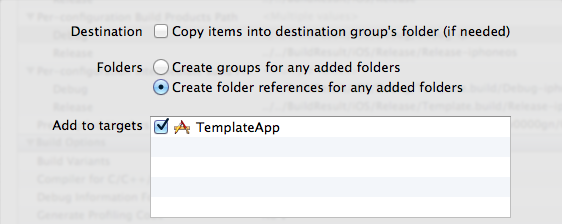


Target和Name是自动填写的，直接点击OK即可完成设置。

添加资源文件，如下图所示：

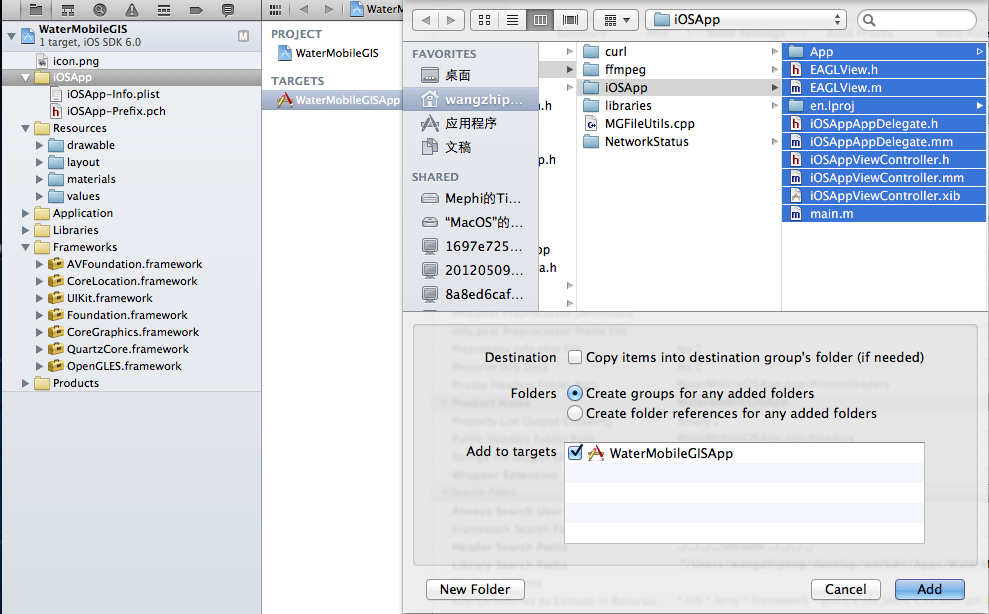


在Resources上右键选择添加文件到工程，将资源文件添加进来，资源文件添加时需要选中Create folder references for any added folders，如下图所示：

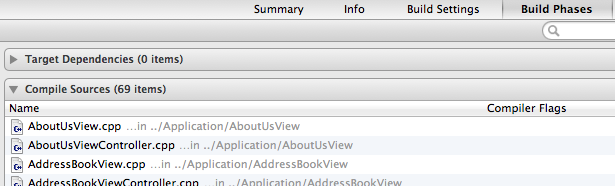


添加应用程序代码文件，右键选择Application选择添加文件到工程，找到对应的code目录下源代码添加进来，源代码添加注意选择Create groups for any added folders.

添加中间件桥接代码，在iOSApp上右键选择添加文件到工程，找到z11目录下的Platform/iOS/iOSApp，将该目录下的文件都添加到iOSApp的组下面，如下图所示：

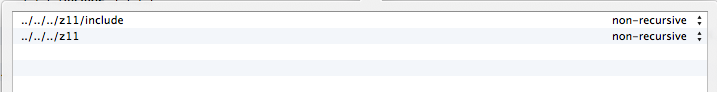


添加完成后，进入工程配置，Build Phases选择项，选中Compile Sources，将里面的AppMainEx.mm删除，AppMainEx.mm是被另外的文件包含的，不需要写在编译列表中，如下图所示：



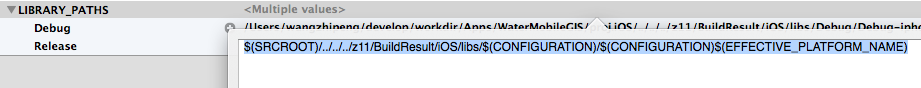
添加依赖库文件，在Libraries组选择添加文件到应用，找到z11的目录，并将Platform/iOS/libraries/libcurl.a文件添加到应用工程中。

修改应用工程配置的Header Search Paths，指向z11的安装目录或者源码目录，如下图所示：



需要包含z11的根目录和z11的include目录，这里的../../../是根据编写文档时的Apps工程和z11工程的目录相对关系确定的，你需要根据自己放置目录的实际情况指定，都需要选择不递归遍历，即non-recursive。

在应用工程配置中增加一项User-Defined，命名为LIBRARY\_PATHS，并将其值根据应用工程目录和z11目录的相对关系进行设置，如下图所示：



$(SRCROOT)/../../../z11/BuildResult/iOS/libs/$(CONFIGURATION)/$(CONFIGURATION)$(EFFECTIVE\_PLATFORM\_NAME)

字符串$(CONFIGURATION)/$(CONFIGURATION)$(EFFECTIVE\_PLATFORM\_NAME)能保证在不同的编译运行条件下（Debug/Release）链接到正确的z11库文件，前面根据相对路径关系设置../../../z11，找到z11的根目录。同时将应用工程配置的Library Search Paths设置为"$(LIBRARY\_PATHS)"，如下图所示：



如果工程是来自Template，配置到此结束，如果工程是之前就存在的工程，现在从z11脱离出来，建议重新通过Template建立工程。

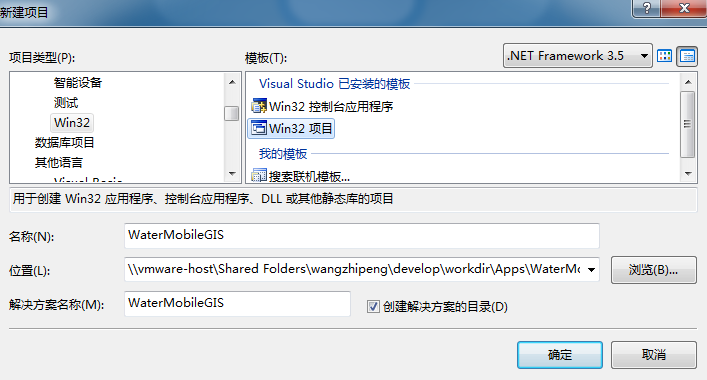
最后注意，编译运行过后，要将工程目录下的BuildResult设置下svn属性为ignore，其实不设置也可以，因为是独立的目录，也不会上传svn，注意不要add就行。如果要设置svn属性为ignore，命令如下：

svn propset svn:ignore “BuildResult” ./

在应用当前目录下执行这个命令就行。

# Win32应用工程配置

通过Visual Studio创建工程，选择“Win32项目”，设置好应用名称和应用路径，如下图所示：



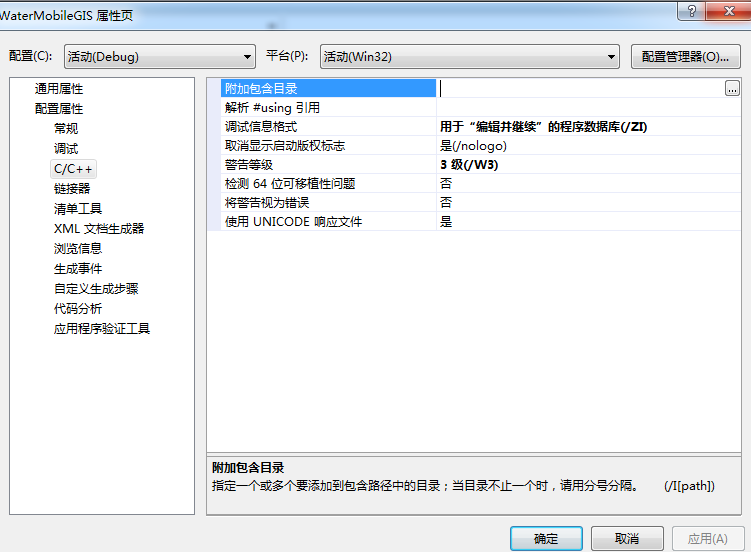
在应用程序设置中选择“Windows应用程序”和“空项目”，如下图所示：



将创建的工程文件目录下的文件拷贝到需要配置的应用工程目录的proj.win32目录下，双击打开.vcproj工程文件进行工程配置。

将z11/Platform/Win32/Application/main.cpp拷贝到应用工程的proj.win32目录下，并通过VS将main.cpp添加到工程中。

打开工程属性设置，选择C/C++，设置附加包含目录，如下图所示：



将包含目录设置为：

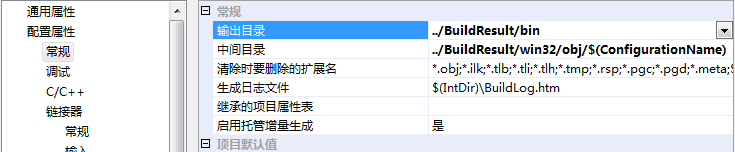
../../../z11/trunk/z11/include

../../../z11/trunk/z11

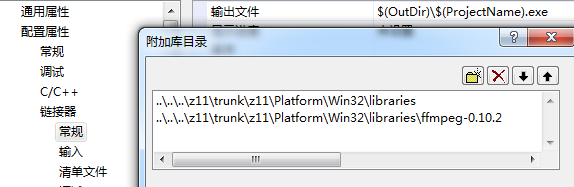
../Application

注意这里的相对路径关系，是指基于应用工程proj.win32目录开始进行相对关系配置。

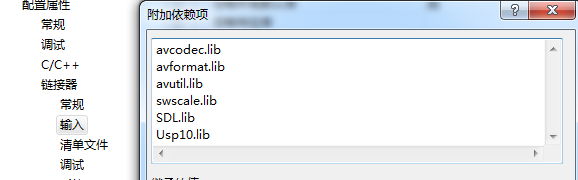
选择常规配置，设置输出目录和中间目录为如下图的内容：



选择链接器常规配置，将附加库目录设置为如下图的内容：



在链接器输入设置中，增加附加依赖项如下图的内容：



应用工程的资源文件和应用执行文件的相对关系是资源文件放在应用工程的Resources目录下，生成的运行文件放置在应用工程的BuildResult/bin/目录下，需要将z11/Dependency/win32/bin/目录下的文件，全部拷贝到应用工程的BuildResult/bin/目录下。

增加z11库的支持，在附加库目录设置中，增加如下内容：

../../../z11/trunk/z11/BuildResult/win32/libs/$(ConfigurationName)

在附加依赖项设置中，增加如下内容：

CommonBase.lib

EmbeddedGIS.lib

FreeType.lib

ftgles.lib

GraphicCore.lib

gtest.lib

GUI.lib

json.lib

libcurl.lib

libcurlpp.lib

LibJPEG.lib

LibPNG.lib

Platform.lib

skia.lib

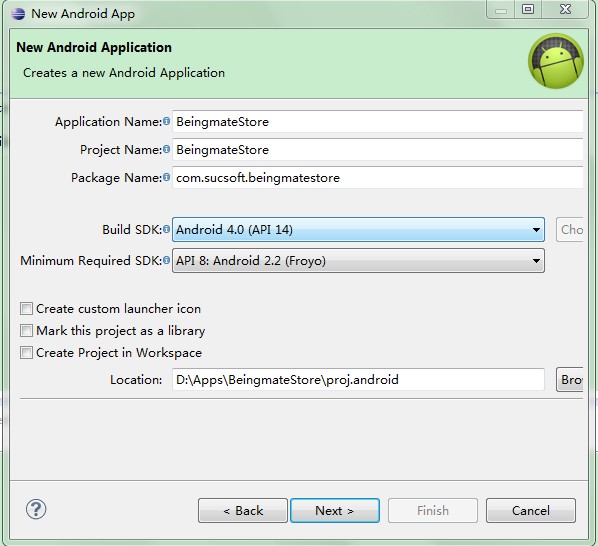
cryptopp.lib

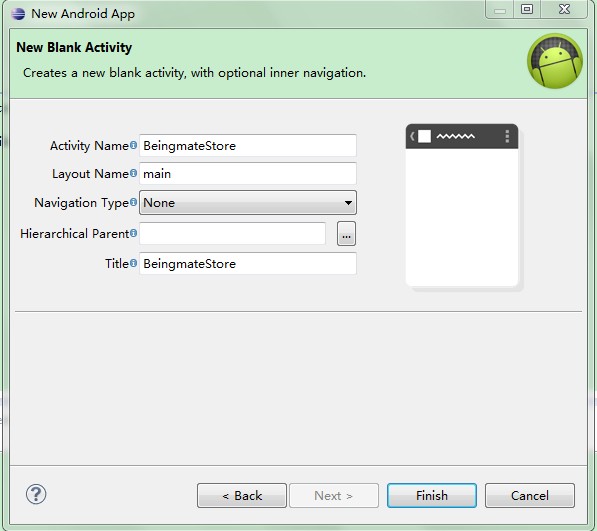
System.lib

zlib.lib

将应用工程Application目录下的文件，加入到工程中进行编译链接即可运行。

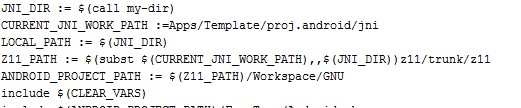
# Android应用工程配置

使用eclipse建立新工程: 



将z11\trunk\z11\Workspace\GNU\Template.zip解压到proj.android。

打开proj.android\jni\Android.mk,将第二行中的Template改成当前工程名:



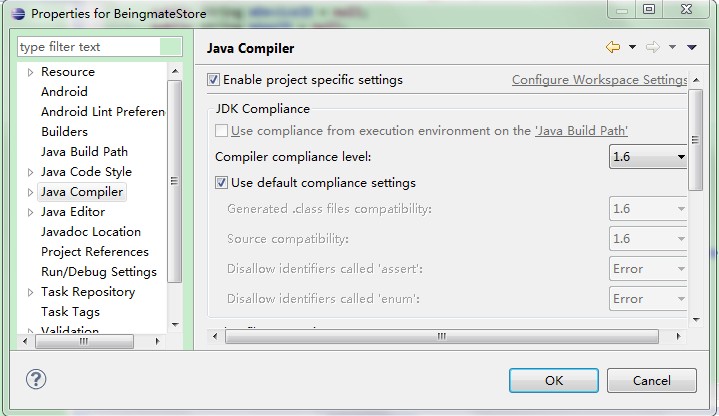
找到“LOCAL\_SRC\_FILES := \”该行，下面添加工程cpp文件：



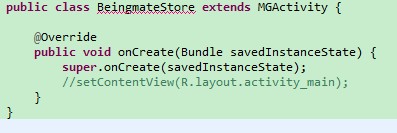
注意“\”后面不要加空格。

打开cygwin，进入proj.android文件夹，输入命令ndk-build编译。

刷新android工程，将Compiler level改成1.6：



修改android工程下主activity文件：



点击运行。