# CrossApp之HelloWorld详解

使用CrossApp创建好了工程后，在工程的Classes目录下，可以看见两个类，一个是AppDelegate类，这个是所有工程必须包括的入口类，我们开发自己的App，实际编码也是从这个类开始，这里主要解释从这个入口开始，我们的HelloWorld是如何工作的。

在Classes目录下还有一个新建工程默认创建的FirstViewController类，这个类就是我们的第一个HelloWorld。

打开AppDelegate.cpp，找到如下方法：

bool AppDelegate::applicationDidFinishLaunching()

{

// initialize director

CAApplication\* pDirector = CAApplication:: getApplication ();

CCEGLView\* pEGLView = CCEGLView::sharedOpenGLView();

pDirector->setOpenGLView(pEGLView);

// create a scene. it's an autorelease object

m\_pWindow = FirstViewController::createWindow();

// run

pDirector->runWithScene(m\_pWindow);

return true;

}

找到红色标记的那两行，从这里开始就表示运行一个window，也就是第一个HelloWorld程序。在正式讲解HelloWorld之前，我们必须理解CrossApp的基本界面运行显示原理，CrossApp是基于MVC设计模式的，这就意味着一个最简单的HellWorld也必须符合这种模式。HelloWorld的详解在这里主要通过三个类来解说，分别是：CAWindow窗体类、CAViewController视图控制类、CAView视图类。

打开FirstViewController.h

class FirstViewController: public CAViewController

{

public:

static cocos2d::CAWindow\* createWindow();

FirstViewController();

virtual ~FirstViewController();

protected:

void viewDidLoad();

void viewDidUnload();

};

在FirstViewController类中，我们看见FirstViewController是继承自CAViewController视图控制类，它主要用于控制视图的加载，切换等操作，是所有视图的管理器。而我们这里所说的视图就是CAView，我们在App中能看见的一切界面其实都是视图。对于CrossApp的MVC模式我就可以理解为Model层(逻辑算法)+Control层(CAViewController)+View层(CAView)，这样也就构成了CrossApp开发的一个基本框架。

在FirstViewController类中有一个createWindow()工厂方法，用于创建一个窗体，返回值为CAWindow\*类型。这里的window是可以理解为容纳我们的App所有界面的一个载体，它是唯一的，在每个App中唯一存在一个window，在应用启动时创建。打开FirstViewController.cpp，

CAWindow\* FirstViewController::createWindow()

{

CAWindow\* \_window = CAWindow::create();

FirstViewController\* \_viewController = new FirstViewController();

\_viewController->init();

\_window->setRootViewController(\_viewController);

\_viewController->release();

return \_window;

}

从上面代码中可以看出，创建了一个窗体，并为当前窗体添加了一个视图管理器\_viewController，视图管理器默认为我们创建了一个白色背景的view。在FirstViewController.h

中还有两个受保护的函数viewDidLoad()和viewDidUnload()，这是其父类视图管理器的两个纯虚函数，供子类实现的接口，其作用就是在CAViewController的生命周期内，加载视图和销毁视图，而我们使用CrossApp开发App的编码也基本是从viewDidLoad()这个函数开始的。

void FirstViewController::viewDidLoad()

{

// Do any additional setup after loading the view from its nib.

CAImageView\* imageView=CAImageView::createWithTexture(CAImage::create("HelloWorld.png"));

imageView->setFrame(CCRect(200, 300, 330, 214));

this->getView()->addSubView(imageView);

CALabel\* ttf = CALabel::createWithFrame(CCRect(200, 450, 0, 0));

ttf->setColor(CAColor4b\_black);

ttf->setfontName("Arial");

ttf->setfontSize(50);

ttf->setText("Hello World!", "Arial");

this->getView()->addSubView(ttf);

}

以上就是显示我们HelloWorld的代码，首先创建一张背景图，设置其坐标和大小后，作为子节点添加进当前view，通过this->getView()即获取当前的视图。之后在再添加一段文本，设置其颜色、坐标、大小后，在视图上显示出来，这就是我们的第一个HelloWorld。