

LayaPlayer-iosSDK 使用文档

目录

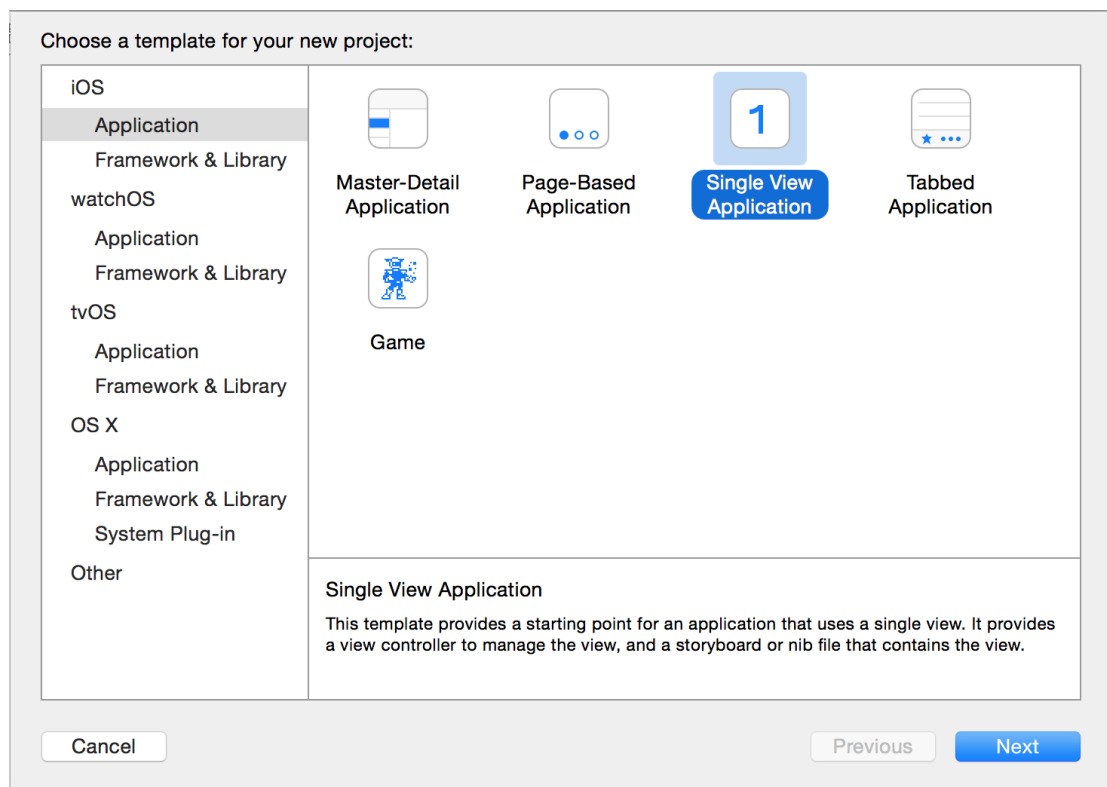
1、适用人群.....	2
2、创建项目.....	2
3、修改最低版本.....	2
4、设置头文件搜索路径.....	3
5、修改 lib 搜索路径.....	4
6、添加依赖库.....	4
7、添加 Framework.....	5
8、修改 BitCode.....	5
9、ViewController 代码.....	6
10、修改 AppDelegate 代码.....	10
11、导入资源.....	10
12、编译并运行项目.....	11
13、启动地址的修改.....	12
14、项目配置，config.ini.....	12
15、平台对接.....	12
16、SDK 目录描述.....	12
17、IPV6.....	12
18、编译项目.....	13
19、Logo 界面.....	14
19.1 进度条控制.....	14
19.2 替换掉开发者的自己的 Logo.....	15
19.3 想要做特别酷炫的进度条怎么办？.....	15
19.4 后续白名单功能.....	15
20、注意事项.....	16

1、适用人群

该文档适用会 ios 开发的程序员，因为还是需要一些 ios 开发的基本知识。

2、创建项目

创建一个 ios 的 Application，选择 Single View Application。



3、修改最低版本

LayaPlayer 最低支持 ios7.0 版本，然后修改一下最低版本为，7.0

▼ Identity

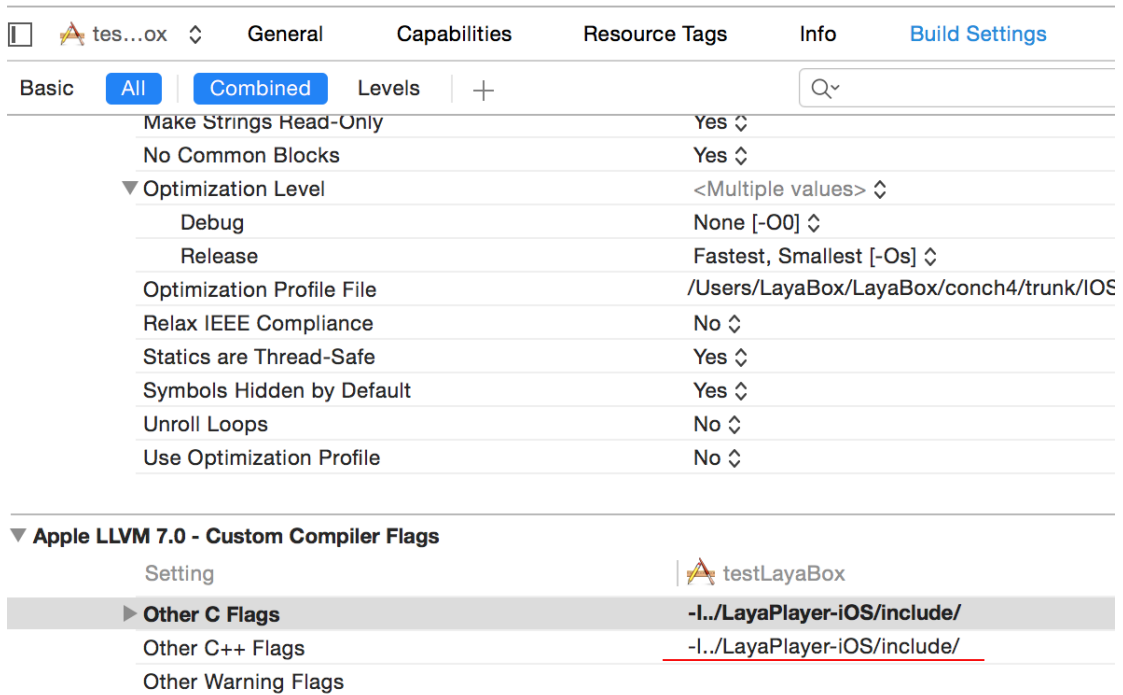
Bundle Identifier	<input type="text" value="comd.laya.testLayaBox"/>
Version	<input type="text" value="1.0"/>
Build	<input type="text" value="1"/>
Team	<input type="text" value="None"/>

▼ Deployment Info

Deployment Target	<input type="text" value="7.0"/>
Devices	<input type="text" value="Universal"/>
Main Interface	<input type="text" value="Main"/>
Device Orientation	<div><input checked="" type="checkbox"/> Portrait <input type="checkbox"/> Upside Down <input checked="" type="checkbox"/> Landscape Left <input checked="" type="checkbox"/> Landscape Right</div>
Status Bar Style	<input type="text" value="Default"/>
	<div><input type="checkbox"/> Hide status bar <input type="checkbox"/> Requires full screen</div>

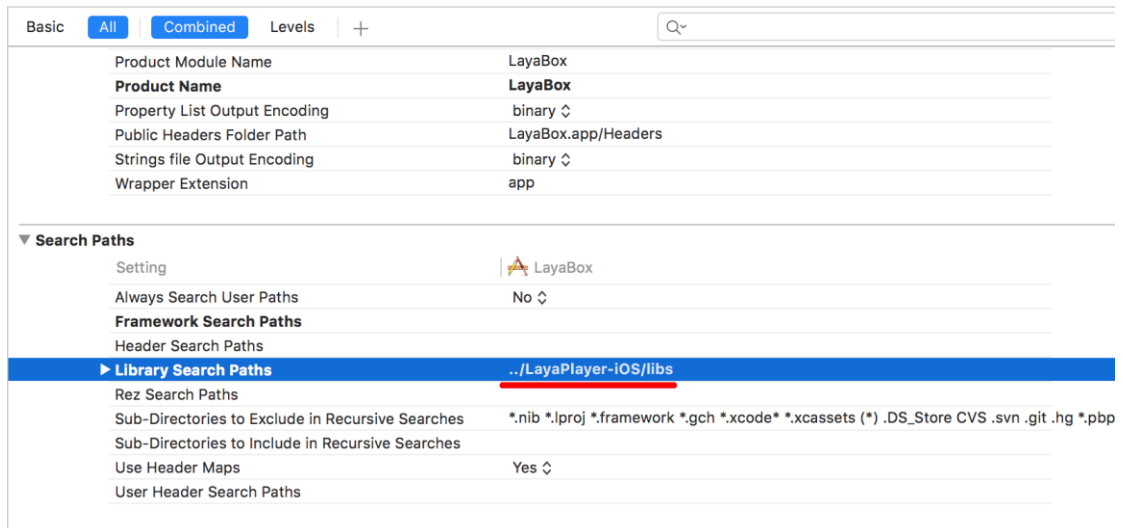
4、设置头文件搜索路径

选择 Build Setting 选项在 Custom CompilerFlags 选项中，增加头文件搜索路径，“-I../LayaPlayer-iOS/include/”，如下图所示：



5、修改 lib 搜索路径

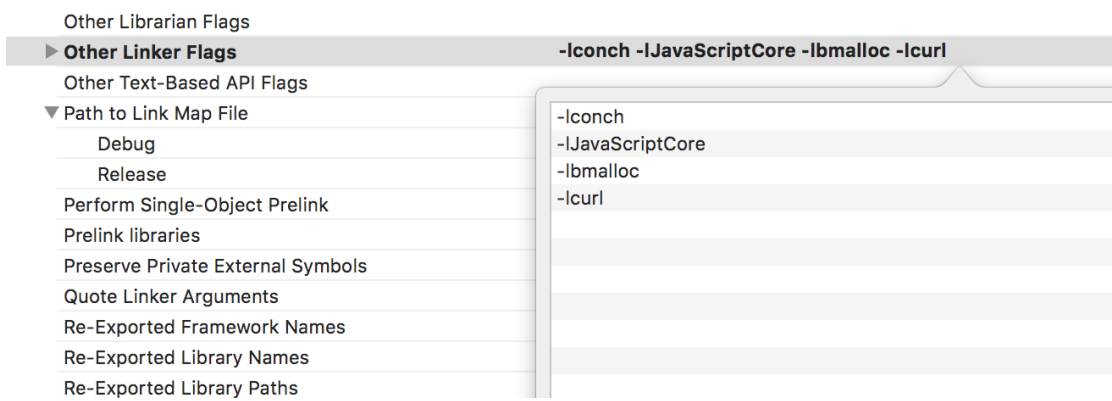
选择 Build Setting-->Search Paths-->Library Search Paths 目录下，添加库文件的搜索路径 "../LayaPlayer-iOS/libs"，如下图所示：



6、添加依赖库

选择 Build Setting-->Linking-->other Linker Flags 目录下，填入依赖的库，"-lconch"、












“-lJavaScriptCore”、“-lbmalloc”，如下图所示：



注意：如果想支持 https，可以 link “-lcurl”。

7、添加 Framework

LayaPlayer 需要一些系统默认的 Framework，需要在 General-->Link Frameworks and Libraries 中添加，如下图所示：

Name	Status
 AdSupport.framework	Required ⇅
 libicucore.tbd	Required ⇅
 AVFoundation.framework	Required ⇅
 CoreTelephony.framework	Required ⇅
 libz.tbd	Required ⇅
 UIKit.framework	Optional ⇅
 GLKit.framework	Required ⇅
 OpenGL.framework	Required ⇅
 OpenAL.framework	Required ⇅
 SystemConfiguration.framework	Required ⇅
 CoreGraphics.framework	Required ⇅
+ -	

8、修改 BitCode

现在编译一下项目发现无法编译过，提示：

```
ld: '../LayaPlayer-iOS/libconch.a(path.o)' does not contain bitcode. You must rebuild it with bitcode enabled (Xcode setting ENABLE_BITCODE), obtain an updated library from the vendor,
```

```
or disable bitcode for this target. for architecture armv7
clang: error: linker command failed with exit code 1 (use -
v to see invocation)
```

现在需要修改一下项目配置，Build Settings-->Build Options-->Enable Bitcode 改成 no

9、ViewController 代码

正式添加代码，修改 ViewController，让其继承 GLKViewController，代码如下所示：

```
#import <UIKit/UIKit.h>

#import <GLKit/GLKit.h>

#import <conchRuntime.h>

@interface ViewController : GLKViewController
{
}

@public

    GLKView*                m_pGLKView;
    EAGLContext*            m_pGLContext;
    conchRuntime*            m_pConchRuntime;
}

+(ViewController*)GetIOSViewController;

-(id)init;

@end
```

修改 ViewController 为 mm 格式，并且代码如下：

```
#import "ViewController.h"

@implementation ViewController

static ViewController* g_pIOSMainViewController = nil;

+(ViewController*)GetIOSViewController

{
```

```

    return g_pLOSMainViewController;
}

-(id)init
{
    self = [super init];
    if( self != nil )
    {
        g_pLOSMainViewController = self;
        return self;
    }
    return Nil;
}

- (void)viewDidLoad
{
    [super viewDidLoad];

    //保持屏幕常亮，可以通过脚本设置
    [[UIApplication sharedApplication] setIdleTimerDisabled:YES];

    self->m_pGLContext = [[EAGLContext alloc] initWithAPI:kEAGLRenderingAPIOpenGL
ES2];

    if (!self->m_pGLContext)
    {
        NSLog(@"Failed to create ES context");
    }

    m_pGLKView = (GLKView *)self.view;
    m_pGLKView.context = self->m_pGLContext;
    m_pGLKView.drawableDepthFormat = GLKViewDrawableDepthFormat24;
    [EAGLContext setCurrentContext:self->m_pGLContext];

    self.preferredFramesPerSecond = 10000;

    //conchRuntime 初始化 ConchRuntime 引擎

```

```

    m_pConchRuntime = [[conchRuntime alloc] initWithView:m_pGLKView EAGLContext:
m_pGLContext logoPath:@"logo.jpg"];
}

- (void)dealloc
{
    [self tearDownGL];
    if ( [EAGLContext currentContext] == self->m_pGLContext )
    {
        [EAGLContext setCurrentContext:nil];
    }
}

- (void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];

    //conchRuntime 内存警告的时候的处理
    [m_pConchRuntime didReceiveMemoryWarning];
}

- (void)tearDownGL
{
    [EAGLContext setCurrentContext:self->m_pGLContext];
}

- (void)glkView:(GLKView *)view drawInRect:(CGRect)rect
{
    //conchRuntime renderFrame
    [m_pConchRuntime renderFrame];
}

- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
    //conchRuntime touch

```



```

    [m_pConchRuntime touchesBegan:touches withEvent:event];
}

- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event
{
    //conchRuntime touch
    [m_pConchRuntime touchesMoved:touches withEvent:event];
}

- (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event
{
    //conchRuntime touch
    [m_pConchRuntime touchesEnded:touches withEvent:event];
}

- (void)touchesCancelled:(NSSet *)touches withEvent:(UIEvent *)event
{
    //conchRuntime touch
    [m_pConchRuntime touchesCancelled:touches withEvent:event];
}

-(NSUInteger)supportedInterfaceOrientations
{
    return [conchConfig GetInstance]->m_nOrientationType;
}

- (BOOL)shouldAutorotate
{
    return YES;//支持转屏
}

@end

```

conchRuntime 的重点代码解释:

- (1)、initWithView 初始化 conchRuntime 需要传入 GLKView 和 EAGLContext
- (2)、renderFrame 主要的渲染函数

(3)、touchesBegan、touchesMoved、touchesEnded、touchesCancelled 中分别调用 conchRuntime 的函数

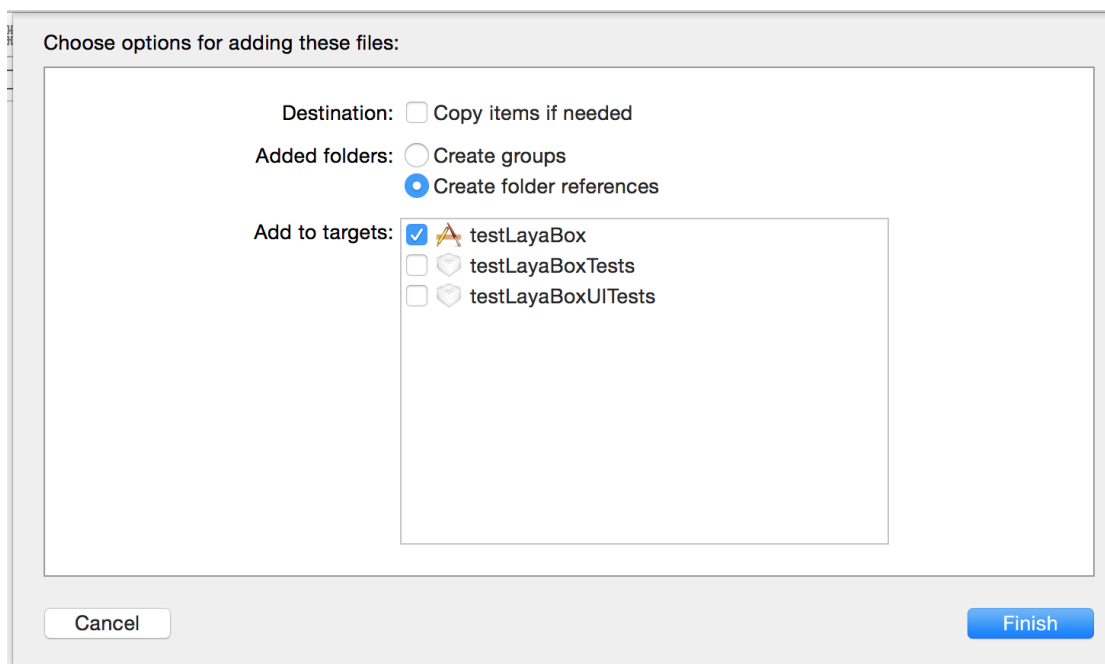
10、修改 AppDelegate 代码

修改 AppDelegate 为 mm 格式在 AppDelegate 的 didFinishLaunchingWithOptions 类中 new ViewController，如下代码所示：

```
self.window=[[UIWindow alloc]initWithFrame:[UIScreen mainScreen] bounds]] ;
ViewController* pViewController = [[ViewController alloc] init];
self.window.rootViewController = pViewController;
[self.window makeKeyAndVisible];
```


















11、导入资源

把 LayaPlayer-iOS/resource 目录拷贝到项目中，然后在 xcode 项目下创建 resource 目录，然后把 resource 所有文件拖到 xcode 的 resource 目录下，选择“Create folder references”如下图所示：



可以在 BuildPhases-->Copy Bundle Resources 目录下是正确的

▼ Copy Bundle Resources (17 items)

	drawImagePS.glsl ...in resource
	LaunchScreen.storyboard
	fillImage_ps.glsl ...in resource
	cache ...in resource
	logo.jpg ...in resource
	defvs.glsl ...in resource
	drawFilterImage_ps.glsl ...in resource
	Assets.xcassets ...in LayaBox
	drawImagePS2.glsl ...in resource
	dbgFillColorPS.glsl ...in resource
	todevVS.glsl ...in resource
	Main.storyboard
	fillColorPS.glsl ...in resource
	todevVS2.glsl ...in resource
	config.ini ...in resource
	defps.glsl ...in resource
	scripts ...in resource











+

-

12、编译并运行项目

在 ios7.0 的设备上，会运行不起来直接报错，这个时候就需要，导入的 Framework 的 UIKit.framework 的格式为 Optional，如下图所示：

▼ Linked Frameworks and Libraries

Name	Status
 AVFoundation.framework	Required ⚡
 CoreTelephony.framework	Required ⚡
 libz.tbd	Required ⚡
 UIKit.framework	Optional ⚡
 GLKit.framework	Required ⚡
 OpenGL.framework	Required ⚡
 OpenAL.framework	Required ⚡
 JavaScriptCore.framework	Required ⚡
 SystemConfiguration.framework	Required ⚡
 CoreGraphics.framework	Required ⚡

+

-

13、启动地址的修改

(1)、打开 index.js，最下面修改启动地址

```
loadUrl(conch.presetUrl||"http://10.10.20.200:8899/default.html");
```

14、项目配置，config.ini

```
orientation    这个屏幕的方向按照如下值进行设定
/*
UIInterfaceOrientationMaskPortrait, ===2
UIInterfaceOrientationMaskPortraitUpsideDown, ===4
UIInterfaceOrientationMaskLandscapeLeft, ===8
UIInterfaceOrientationMaskLandscapeRight, ===16
*/
```

15、平台对接

ios-内购和第三方渠道接入（登陆、支付等），需要开发者进行二次开发，LayaPlayer 引擎可以提供了一个 Market 类，这个类可以实现 JS 和 Object-C、Java 之间的互调。

具体对接文档，请参考其他文档。

16、SDK 目录描述

LayaBox 是一个简单的例子，如果觉得特别上面的步骤特别繁琐，您可以直接使用这个例子，进行开发。

LayaPlayer-iOS 是 SDK 目录和文档

17、IPV6

苹果公司在 2016 年 6 月 1 日起，强制执行 ipv6 标准，所以开发者发布项目的时候，http

请求和 `socket` 都必须使用域名的方式，不能使用 `ip` 地址。

如何测试 `ipv6` 网络下是否正常，请参考以下文档

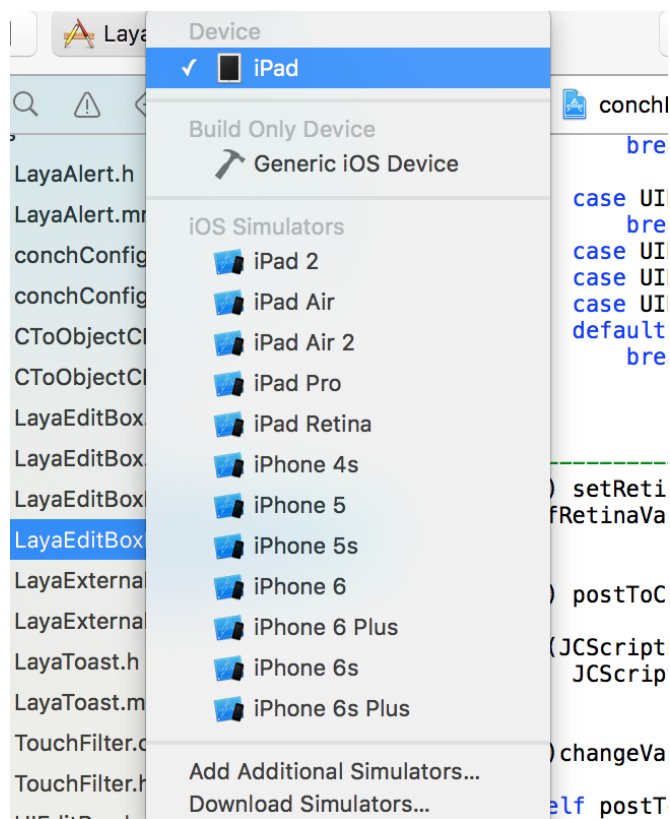
https://developer.apple.com/library/mac/documentation/NetworkingInternetWeb/Conceptual/NetworkingOverview/UnderstandingandPreparingfortheIPv6Transition/UnderstandingandPreparingfortheIPv6Transition.html#//apple_ref/doc/uid/TP40010220-CH213-SW1

http://www.pchou.info/ios/2016/06/05/ios-supporting-ipv6.html?utm_source=tuicool&utm_medium=referral

18、编译项目

`android` 项目，直接导入到 `eclipse` 或者 `android-studio` 即可。

`ios` 项目，打开 `testLayaBox`(或者你自己的项目目录).`xcodeproj` 文件，打开后选择真正的 `ios` 设备进行 `build`。(注意：真正的设备是 `armv7`、`armv7s`、`arm64` 架构，如果是 `ios Simulator` 这些都是 `x86` 架构，目前 `LayaPlayer` 在 `ios` 设备上尚未支持 `x86` 架构，如果选择模拟器编译是无法通过的)。



ios 如果想真机运行调试，需要开发者去苹果购买开发者账号、并设置程序签名等等，这部分请开发者自行学习。

19、Logo 界面

app 启动会先下载 html，然后进行解析，再下载、执行 js，这个过程需要一点时间，所以在 LayaPlayer 中，默认项目显示了一个 LayaBox 的动画。

19.1 进度条控制

Logo 界面默认播放完这个动画，进入游戏，这个 logo 动画中还可以配置加载进度，如下代码所示

```
if(window.loadingView)
```

```
{
```

```
    window.loadingView.loading(100); // 设置 0 - 100 之间的值，当这个值达到 100 的时候 loadingView 消失，显示游戏画面
```

```
}
```

其他设置方法,在 script/config.js 中,开发者可以根据自己需求进行调用相应方法,如下代码所示:

```
loadingView.loadingAutoClose=true;  
loadingView.bgColor("#ffffff");//设置背景颜色  
loadingView.setFontColor("#000000");//设置字体颜色  
loadingView.setTips(["新世界的大门即将打开", "敌军还有 30 秒抵达战场"]);//可以设置一个数组,这个数组中的文字会自动切换。
```

loadingAutoClose 默认这个变量为 true,当 LayaBox 的动画播放完一遍,就会自动把这个 logo 界面隐藏掉了,如果开发者想要精准控制,需要把这个变量设置为 false,通过 window.loadingView.loading(100),进行精确控制。

19.2 替换掉开发者的自己的 Logo

如果开发者想要把 LayaBox 的动画替换成自己的 Logo,需要做好一张图片放到 resource/logo/logo.png (注意必须为 png),这样就不会加载 LayaBox 的动画了,其他设置和原来的一样,设置背景色、tips、fontColor 等。

19.3 想要做特别酷炫的进度条怎么办?

在实际项目中,如果开发者想要做出自己喜欢的酷炫的进度条,LayaPlayer 现有的方案是不够满足的,建议开发者快速加载 LayaAir-JS 引擎和必备的图片,通过 LayaAir 自己实现酷炫的进度条。

19.4 后续白名单功能

后续 LayaBox 会有白名单机制,如果开发者购买了授权或者和 LayaBox 联合运营产品,便可以去掉 LayaBox 的 logo,如果没有则需要强制增加 LayaBox 的 logo。引擎内部会有检测

机制，随机检测，如果检测不通过则无法进入游戏。

20、注意事项

文本格式的文件（例如:ini、xml、html、json、js 等）都必须是 utf8 编码格式，因为 iOS 设备现在尚不支持非 utf8 格式编码的文件。

在 LayaPlayer 中背景音乐目前只支持 mp3 格式，并且不能同时播放多个。音效可同时播放多个，但是为了效率考虑，目前只支持 wav 格式，具体格式为 PCM 格式、单声道、16 位、22050 采样率

mp3 转 wav 示例教程 <http://www.layabox.com/html/documents/71.html>