


Unity 可试玩广告实践指南





技术基础







请注意，可试玩广告使用到的知识都为 **web** 开发相关知识，若您的开发人员不懂 **web** 开发，那做起来可能需要去查找很多相关的信息，效率比较低；若您的团队有懂 **web** 开发的人，那么制作可试玩广告所涉及的知识对他们来说应该都是比较熟悉的了。

可试玩广告必须放在一个单一的 **html index** 文件。在一个设备要展示广告时，只下载这一个文件。

Name	Size	Kind
 index.html	GOOD 2 MB	HTML

像下边两个图的例子里，除了 **index.html** 还有几个其他文件，这样其他几个文件就都丢失了。

	CSS	
	images	4 Files -> BAD
	game.js	
	index.html	

Name	Size	Kind
	4 KB	Windows icon image
 index.html	BAD 3 KB	HTML document
	403 KB	Folder
	34 KB	BBEdit text document
	6 KB	Folder
	2.7 MB	Folder

index.html 文件必须 minified

Non-inlined:

```
1 
```

Inlined:

```
1 
```

在上述“BAD”图里，**index.html**里包含了链接到其他的文件和文件夹，这样

实际上在展示的时候还需要下载其他的文件，并不是所有的文件都被缓存到了设备里。而且也包含了一些相对路径，比如
/myParentFolder/resources/images/pic- ture2，这样设备里根本找不到任何文件。

Inline或者minified的意思是使用javascript压缩工具，把整个文件中的代码去掉非必要字符（比如空格、回车和缩进等），把整个源文件，此处为index.html变成一个长字符串的技术。对于一般的web开发来说，这一步不是必须的，但对于可试玩广告，这一步是必须的。

Non-minified

```
var webAudioCtx = null;
try {
    // Fix up for prefixing
    var _AudioContext = window.AudioContext || window.webkitAudioContext;
    webAudioCtx = new _AudioContext();
} catch(e) {
    alert('Web Audio API is not supported in this browser');
    webAudioCtx = null;
}
function isSupportWebAudio() {
    return webAudioCtx != null;
}
function getWebAudioContext() {
    return webAudioCtx;
}
```

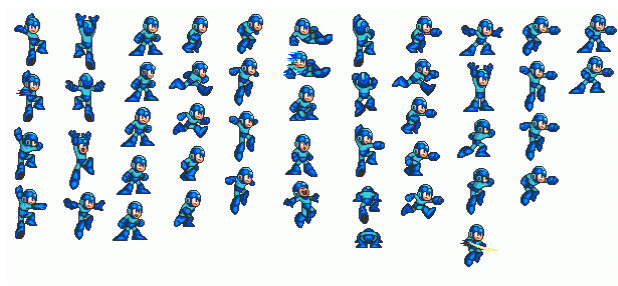
Minified

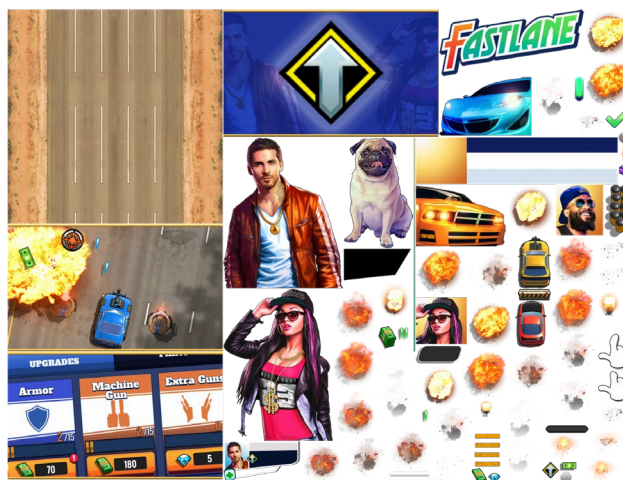
```
var __reflect=(this&&this.__reflect)||function(p,c,t){p.__c
EgretShaderLib.primitive_frag="precision lowp float;invariant vec2
},200));Object.defineProperty(HtmlSoundChannel.prototype,"volume",{
},enumerable:true,configurable:true));WebAudioSound.prototype.load=
video.addEventListener("error",function(){return _this.onVideoError
}this._fullscreen=!!value;if(this.video&&this.video.paused==false){
}if(this.headerObj){for(var key in this.headerObj){this._xhr.setReq
};HTML5StageText.prototype._initElement=function(){var point=this.$
});HTML5StageText.prototype.$removeFromStage=function(){if(this.in
egret.$callAsync(function(){inputElement.style.opacity=1,self});HT
newContext.setTransform(1,0,0,1,0,0);newContext.drawImage(oldSurfac
web.WebTouchHandler=WebTouchHandler;__reflect(WebTouchHandler,proto
}}}}var winURL=window["URL"]|window["webkitURL"];if(!winURL){Html!
egret.sys.canvasRenderer=egret.sys.systemRenderer;egret.sys.customI
var context=canvas.getContext("2d");this.contextFps=context;context
}search=search.slice(1);var searchArr=search.split("&");var length_;
```

Additional notes: Art assets:

使用 **spritesheets** 来安排美术资源。

下边这种是 **sprite maps**、**sheets**或者叫**databases**的示例：





一般角色的移动、动画之类的都是使用这种技术实现的。把很多图片放在同一个美术文件里，这样只需要使用一次网络请求来读取图片。

浏览器测试:

整个制作好的文件即使不通过MRAID协议投递广告，也应该能够成功在桌面浏览器里运行，点击下载按钮可以跳转到appstore等。

下载按钮功能设计:

请确保下载按钮有按下时的视觉反馈，这样用户才能明确知道自己进行了点击操作。

链接到appstore应该是一个简单的 mraid.open调用:

```
case "Android":mraid.open("https://play.google.com/store/apps/details?id=yourgame"); break;
```

```
case "iOS":mraid.open("https://itunes.apple.com/us/yourgame?mt=8");
```

文件大小限制:

整个广告的文件大小必须在 5 Mb以内

音频（Audio）：

如果您的可试玩广告包含声音，则必须默认优先使用 Web Audio API 若使用不能使用，则fallback到 HTMLAudioElement 或者 HTML DOM Audio 对象. 现代的web游戏开发框架和声音库 (比如Howler.js)都是这样做的，请确保您自己开发时也这样使用。

Web Audio API 是现在iOS上唯一一个可以切换静音和铃声等操作的解决方案。在Android上，当app被置为后台时，请使用 `viewableChangeListener` 来处理播放或静音。

MRAID 基础: Bootstrapping:

```
// Wait for the SDK to become ready
if (mraid.getState() === 'loading') {
    mraid.addEventListener('ready', onSdkReady);
} else {
    onSdkReady();
}

function viewableChangeListener(viewable) {
    // start/pause/resume gameplay, stop/play sounds
    if(viewable) {
        showMyAd();
    } else {
        // pause
    }
}

function onSdkReady() {
    mraid.addEventListener('viewableChange', viewableChangeListener);
    // Wait for the ad to become viewable for the first time
    if (mraid.isViewable()) {
        showMyAd();
    }
}

function showMyAd() {
    ...
}
```

可试玩广告的下载按钮 – 如何打开商店:

```
// Detect platform from user agent
var userAgent = navigator.userAgent || navigator.vendor;
var url = '<replace with iTunes store>';
var android = '<replace with Google Play Store link>';
if (/android/i.test(userAgent)) {
    url = android;
}
mraid.open(url);
```

UnityAds MRAID requirements

全名是Mobile Rich Media Ad Interface Definitions 。

Unity Ads只是MRAID 2.0标准，同时除了以下一些特殊情况：

- 只支持全屏广告, `expand()` and `resize()` 方法不支持。
- 不支持自定义关闭按钮。
- 不支持短信、电话、日历和商店图片
- SDK 2.0.7 之后支持`inlineVideonext`

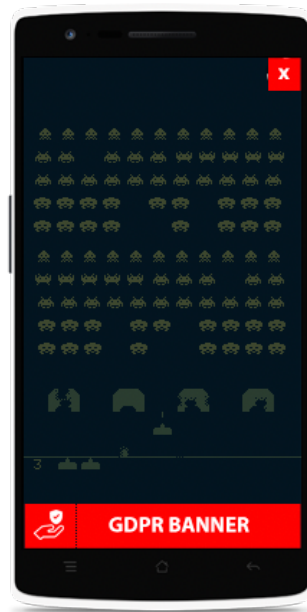
Requirements outside the MRAID specification:

- 必须提供给Unity想要支持的设备和OS版本。
- Unity Ads SDK 支持Android 4.4+ 和 iOS 8.0+
- 是否可跳过是从Unity Ads这边来设置的，广告主这边不能控制。
- 广告需要支持横屏和竖屏
- 广告应该不使用任何XHR请求，但类似对信任的服务供应商回传分析事件是可以的。
- 在设计click-through时，广告的点击应该使用`mraid.open()`直接打开应用商店，任何开始/观看/点击的attribution都从服务器端完成。
- 在广告开始前，Unity Ads SDK会有一个2秒左右的加载界面。这个时间用来加载资源和准备好播放器。
- 在收到 MRAID “ready” 事件后开始初始化。
- 在收到 MRAID “viewableChange”事件后开始展示试玩广告内容。

可试玩广告玩法和布局

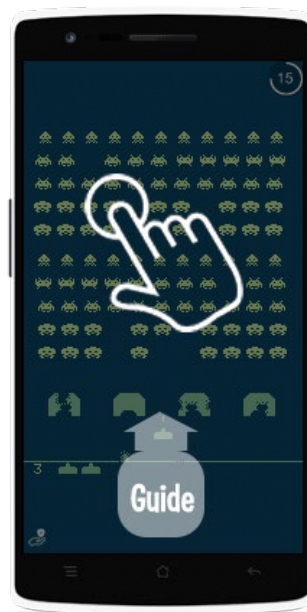
整个广告长度应该在30秒左右。推荐在27秒左右时出现结尾页来提升下载转化。

固定位置的界面元素（FIXED ELEMENTS）



如上图的关闭按钮，右下角的小手，以及可能会出现GDPR banner都是Unity Ads自带的。请不要在这些元素的文职放置容易误遮挡的UI元素。

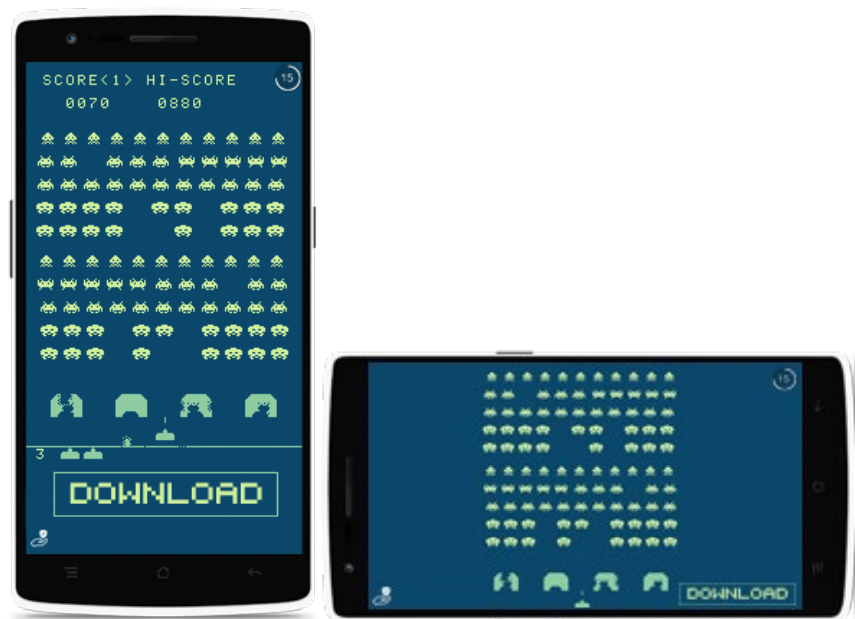
开始页（START SCREEN）



一段简单的指南

- 为了吸引注意力，试玩应该尽快开始。
- 如果超过20秒玩家都没操作，应该开始后边接下来的内容比如结束待操作功能或自动操作等。
- 展示简单的示例操作。

玩法体验（GAMEPLAY）



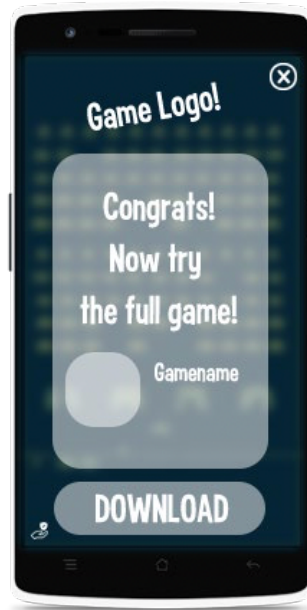
允许玩家完成试玩或试玩失败

- 在合适的位置放置一个下载按钮。
- 确保在横屏和竖屏都能够正常试玩和操作。

可以在空白处放置背景图片，也可以把下载按钮放置在较空旷的位置。

广告可能在横屏或者竖屏时被展示，所以强制玩家转屏是很不好的用户体验。

结尾页



出现试玩游戏失败或者祝贺成功消息。

包含基本的应用信息，游戏icon，名字，logo等。

给下载按钮加一点简单的动画来引起注意。但需注意不要使用闪烁效果，苹果不允许在下载按钮上实现过于炫丽或诱导的效果。

Unity MRAID 测试应用

我们有一个用于测试可试玩广告的应用，这样就不用使用浏览器来测试了。可以通过以下方式下载测试应用：

For iOS:

在iPhone或者iPad里点击下列链接，或者复制后在iOS设备的Safari浏览器里打开：(链接必须包含 itms-services)

```
itms-services://?action=download-manifest&url=https://s3.amazonaws.com/applifier_deployment/ios/enterprise/uads-creative.plist
```

Safari 会问你是否打开iTunes, 选择是 (yes) . iOS会弹出一个窗口问你是否安装 'Unity Creative Test', 选择是 (yes) .

这个应用会出现在桌面上等待开始下载安装。在开始安装前，需要先到设置里去信任企业开发者证书。

打开iOS的设置 (Settings) 应用, 选择通用 (General) , 选择设备和证书

(Profiles & Device Management), 选择Unity Technologies Finland Oy - profile 信任它, 然后点击桌面上待安装的, 处于灰暗状态的应用图标, 它开始下载安装。

For Android:

下载并安装附件里的APK文件。您可能需要先在Android设置里信任并允许安装未知来源的应用。

用法:

架好制作完成的html文件, 并确保可以通过一个链接地址访问它。然后使用一个二维码 (QR Code) 生成器, 把链接生成一个二维码。使用测试应用的二维码扫描功能扫一下生成的二维码以读取链接来测试制作好的试玩广告。

注意: 对于 iOS来说, url必须使用安全链接 (https), http是不行的。

上传制作完成的广告:

请多多测试, 确保找到潜在的问题。请确保html代码都压缩过并在都放置在唯一的index.html里。请确保整个文件小于5MB。

在提交可试玩广告给我们试, 请一并告知支持的凭条和操作系统版本。
(比如 “仅支持 iOS 9+ 和 Android 5.0+”)

FAQs

在服务器端检查CORS header, allow origin必须设成*。

有的制作出的可试玩广告在浏览器里不好用, 所以请使用我们的测试应用进行测试。同时也可以实现在不同的设备里安装测试应用, 来测试不同设备的目的。

有时制作出的可试玩广告在浏览器里好用, 但是并不能很好地在MRAID协议下执行。对于这种情况, 如果广告主很难查出到底是什么问题, 我们的QA可以帮忙测一下。很多时候使用测试应用能事先检查出很多问题。

Unity Ads SDK本身最低支持 iOS 8+ 和 Android 4.4.1+

一般Android 4.4, 5.0需要检查 generally polyfill issues (object.assign, etc.)

需要注意的问题:

- 下载按钮需要链接到正确的商店。
- 时长，从玩家的角度试验从开播到结尾页一共用了多少时间。
- 设备转向，确保播放过程中转屏，可试玩广告仍能正常操作、观看并到达最终页进行下载。

声音问题:

测试声音是否能合理的复合一般用户操作体验。比如如果用户锁屏声音播放应该停止，再次开启应用回到前台声音应该开始播放等。确保设备的声音按键能够正确的控制到可试玩广告的音量。