

## 28-怎么应对各种富文本表现需求？

你好，我是戴铭。今天，我要和你分享的主题是，在iOS开发中，如何展示富文本的内容。

在iOS开发中，富文本的展示是一个非常常见的需求。为了帮助你更好地了解如何展示富文本，我在今天这篇文章中，会结合一个项目来跟你说说面对富文本展示需求时，要怎么考虑和实现。这样，你在自己的项目中，也可以借鉴今天这样的实现思路和方法。

简单来说，富文本就是一段有属性的字符串，可以包含不同字体、不同字号、不同背景、不同颜色、不同字间距的文字，还可以设置段落、图文混排等等属性。

我以前做过一个 [RSS 阅读器](#)，阅读器启动后，需要抓取最新的 RSS 内容进行展示。RSS 里面的文章内容属于富文本，是用HTML标签来描述的，包含了文字样式、链接和图片。

比如，RSS阅读器中的某篇文章内容如下：

```
<item>
<title>涉国资流失嫌疑 东方广益6亿元入股锤子科技被调查</title>
<link>https://www.cnbeta.com/articles/tech/841851.htm</link>
<description>
<![CDATA[
<p><strong>据虎嗅得到的独家消息，成都成华区监察委已立案调查“东方广益6亿元入股锤子科技（北京）股份有限公司”事宜，认为这个项目有
]]>
</description>
<author>ugmbbc</author>
<source>cnBeta.COM</source>
<pubDate>Sat, 27 Apr 2019 09:46:45 GMT</pubDate>
<guid>https://www.cnbeta.com/articles/tech/841851.htm</guid>
</item>
```

文章的 HTML 代码就在上面 RSS 中的 description 标签里。解析出 RSS 中所有文章的 HTML 代码，并将它们保存到本地数据库中。

接下来，如何展示 HTML 内容呢？当时，我的第一反应就是使用 WebView 控件来展示。

### WebView

使用 WebView 显示文章只需要创建一个 UIWebView 对象，进行一些基本滚动相关的设置，然后读取 HTML 字符串就可以了，具体实现代码如下：

```
self.wbView = [[UIWebView alloc] init];
self.wbView.delegate = self;
[self.view addSubview:self.wbView];
[self.wbView mas_makeConstraints:^(MASConstraintMaker *make) {
    make.top.left.right.bottom.equalTo(self.view);
}];
self.wbView.scalesPageToFit = YES; // 确保网页的显示尺寸和屏幕大小相同
self.wbView.scrollView.directionalLockEnabled = YES; // 只在一个方向滚动
self.wbView.scrollView.showsHorizontalScrollIndicator = NO; // 不显示左右滑动
[self.wbView setOpaque:NO]; // 默认是透明的
```

```
// 读取文章 html 字符串进行展示
[self.wbView loadHTMLString:articleString baseURL:nil];
```

和 UIWebView 的 loadRequest 相比，UIWebView 通过 loadHTMLString 直接读取 HTML 代码，省去了网络请求的时间，展示的速度非常快。不过，HTML 里的图片资源还是需要通过网络请求来获取。所以，如果能够在文章展示之前就缓存下图片，那么无需等待，就能够快速完整地展示丰富的文章内容了。

那么，我应该使用什么方案来缓存文章中的图片呢？

在 Cocoa 层使用 NSURLProtocol 可以拦截所有 HTTP 的请求，因此我可以利用 NSURLProtocol 来缓存文章中的图片。

接下来，我再来和你说说，**如何用我写的一个 Web 页面预加载库 [STMURLCache](#)来预缓存 HTML 里的图片**。这个库你也可以应用到自己项目中。

**首先**，我需要从数据库中取出所有未缓存图片的文章内容 HTML。实现代码如下：

```
[[[[[SMDB sharedInstance] selectAllUnCachedFeedItems] subscribeOn:[RACScheduler schedulerWithPriority:RACSch
// 在数据库中获取所有未缓存的文章数据 x
NSMutableArray *urls = [NSMutableArray array];
if (x.count > 0) {
    self.needCacheCount = x.count;
    for (SMFeedItemModel *aModel in x) {
        // 将文章数据中的正文内容都存在 urls 数组中
        [urls addObject:aModel.des];
    }
}
...
}];
```

如上面代码所示，在数据库中获取到所有未缓存文章的数据后，遍历所有数据，提取文章数据中的正文 HTML 内容保存到一个新的数组 urls 中。

**然后**，使用 STMURLCache 开始依次预下载文章中的图片进行缓存。实现代码如下：

```
[[STMURLCache create:^(STMURLCacheMk *mk) {
    mk.whiteUserAgent(@"gcdfetchfeed").diskCapacity(1000 * 1024 * 1024);
}] preloadByWebViewWithHtmls:[NSArray arrayWithArray:urls]].delegate = self;
```

STMURLCache 使用 preloadByWebViewWithHtmls 方法去预缓存所有图片，在 STMURLCache 初始化时，会设置 UserAgent 白名单，目的是避免额外缓存了其他不相关 UIWebView 的图片。

缓存图片的核心技术还是 NSURLProtocol，STMURLCache 最终也是使用 NSURLProtocol 来缓存图片的。

NSURLProtocol 是一个抽象类，专门用来处理特定协议的 URL 数据加载。你可以使用自定义 URL 处理的方式，来重新定义系统 URL 加载。STMURLCache 缓存图片的具体实现代码，你可以在 [STMURLProtocol](#) 这个类里查看。

STMURLProtocol 会在所有网络请求的入口 canInitWithRequest 方法中加上过滤条件，比如 STMURLCache 在初始化时设置 UserAgent 白名单，过滤代码如下：

```
// User-Agent来过滤
if (sModel.whiteUserAgent.length > 0) {
    // 在 HTTP header 里取出 User Agent
    NSString *uAgent = [request.allHTTPHeaderFields objectForKey:@"User-Agent"];
    if (uAgent) {
        // 不在白名单中返回 NO，不会进行缓存
        if (![uAgent hasSuffix:sModel.whiteUserAgent]) {
            return NO;
        }
    } else {
        return NO;
    }
}
```

UserAgent 白名单过滤会通过 request 的 allHTTPHeaderFields 获取到当前网络请求的 UserAgent，然后和已经设置的 UserAgent 白名单做比较：如果在白名单中就进行缓存；否则，就不会缓存。

STMURLProtocol 还可以根据域名进行过滤，这样可以灵活、精确地控制缓存范围。如果你设置了域名白名单，那么只有在白名单里的域名下的网络请求才会执行缓存，过滤代码如下：

```
//对于域名白名单的过滤
if (sModel.whitelistsHost.count > 0) {
    id isExist = [sModel.whitelistsHost objectForKey:request.URL.host];
    // 如果当前请求的域名不在白名单中也会返回 NO
    if (!isExist) {
        return NO;
    }
}
```

如代码所示，当前网络请求的域名可以通过 request.URL.host 属性获取到，获取到网络请求的域名后，再去看域名白名单里是否有，如果有就缓存，没有就返回 NO，不进行缓存操作。

在 canInitWithRequest 方法中满足缓存条件后，开始缓存的方法是 startLoading。startLoading 方法会判断已缓存和未缓存的情况，如果没有缓存会发起网络请求，将请求到的数据保存在本地。如果有缓存，则会直接从本地读取缓存，实现代码如下：

```
// 从缓存里读取数据
NSData *data = [NSData dataWithContentsOfFile:self.filePath];
NSURLResponse *response = [[NSURLResponse alloc] initWithURL:self.request.URL MIMEType:[otherInfo objectForKey:
```

```
[self.client URLProtocol:self didReceiveResponse:response cacheStoragePolicy:NSURLCacheStorageNotAllowed];

// 使用 NSURLProtocolClient 的 URLProtocol:didLoadData 方法加载本地数据
[self.client URLProtocol:self didLoadData:data];
[self.client URLProtocolDidFinishLoading:self];
```

如代码所示，STMURLProtocol 先通过缓存的路径获取到缓存的数据，再使用 NSURLProtocolClient 的 URLProtocol:didLoadData 方法加载本地缓存数据，以减少网络请求。

显示文章内容时使用 NSURLProtocol，对于那些已经缓存过图片的文章就不用发起图片的网络请求，显示的速度跟本地加载显示速度一样快。

虽然通过 URLProtocol 重新定义系统 URL 加载的方式，来直接读取预缓存提升了加载速度，但在长列表的 Cell 上展示富文本，就需要性能更高、内存占用更小的方法。那么接下来，我们再看看除了 UIWebView 还有没有什么方法可以展示富文本呢？

当然还有了。

在长列表这种场景下，如果不用 HTML 来描述富文本的话，想要使用原生 iOS 代码来描述富文本的话，你还可以使用苹果官方的 [TextKit](#) 和 [YYText](#) 来展示。

其中，YYText 不仅兼容 UILabel 和 UITextView，在异步文字布局和渲染上的性能也非常好。所以接下来，我们就一起看看 YYText 是如何展示富文本的吧。

## YYText

集成 YYText 到你的 App 非常简单，只需要在 Podfile 中添加 pod ‘YYText’ 就可以了。下面代码展示了如何展示图文混排的富文本：

```
NSMutableAttributedString *text = [NSMutableAttributedString new];
UIFont *font = [UIFont systemFontOfSize:16];
NSMutableAttributedString *attachment = nil;

// 嵌入 UIImage
UIImage *image = [UIImage imageNamed:@"dribbble64_imageio"];
attachment = [NSMutableAttributedString yy_attachmentStringWithContent:image contentMode:UIViewContentModeC
[text appendAttributedString: attachment];

// 嵌入 UIView
UISwitch *switcher = [UISwitch new];
[switcher sizeToFit];
attachment = [NSMutableAttributedString yy_attachmentStringWithContent:switcher contentMode:UIViewContentMo
[text appendAttributedString: attachment];

// 嵌入 CALayer
CASharpLayer *layer = [CASharpLayer layer];
layer.path = ...
attachment = [NSMutableAttributedString yy_attachmentStringWithContent:layer contentMode:UIViewContentModeB
[text appendAttributedString: attachment];
```

如代码所示，YYText 对于富文本的图文混排使用的是自定义的 NSMutableAttributedString 分类，自定义分类不光简化了 NSMutableAttributedString，还增加了功能，除了图片外，可以嵌入 UIView 和 CALayer。

通过上面 YYText 描述富文本的代码，你会发现原生代码描述富文本跟 HTML 比，既复杂又啰嗦。HTML 代码更易读、更容易维护，所以除了长列表外，我建议你都使用 HTML 来描述富文本。

对于 UIWebView 内存占用高的问题，你可以考虑使用 HTML 代码转原生代码的思路解决。比如，你可以参考我以前做的将 HTML 代码转原生代码的示例项目 [HTN](#)里的解决思路。

## 小结

今天我跟你介绍了如何通过 UIWebView 和 YYText 来展示富文本。

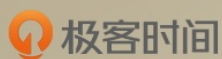
UIWebView 展示的是使用 HTML 描述的富文本。HTML 是描述富文本最简单和最常用的方式，相对于 YYText 或 TextKit 那样描述富文本的方式来说，更加简洁和标准。不过，UIWebView 的缺点也比较明显，同时创建多个 UIWebView 实例，对于内存的占用会非常大。

所以，我对于富文本展示的建议是，如果是列表展示富文本建议使用 TextKit 或者 YYText，其他情况可以选择使用 UIWebView 来展示富文本。

## 课后作业

使用 [STMURLCache](#)预加载你工程中的一个 Web 页面，看看打开速度提升了多少，预加载成功后，在弱网环境和无网络的环境都可以试试。

感谢你的收听，欢迎你在评论区给我留言分享你的观点，也欢迎把它分享给更多的朋友一起阅读。



# iOS 开发高手课

从原理到实战，带你解决 80% 的开发难题

戴铭

前滴滴出行技术专家



新版升级：点击「👤请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。