文档阅读人群：对ReactNative 有所了解。

一 大致介绍

- **react native 优势：**
    - 1 react native 赋予了移动端好于网页的打开速度和体验。
    - 2 react native 相比于移动端来说可以不通过发包就可以更新业务，所以 react native 其中最重要的能力就是热更新。
    - 3 代码复用，不论是android平台还是ios平台都可以运行同一套代码。

为何react native 能够热更新，如果可以，该怎么做?

答案： react native是基于js，然后通过第三方库 翻译成js文件，最后再通过执行js文件，对不同平台翻译成平台对应的组件然后运行，所以最重要的就是最终翻译成的js文件，他决定最终界面长什么样有什么对应逻辑，所以react native 更新问题就变成了如何更新jsbundle文件问题。

- **热更新分类：**
    - 1 全量更新 : 更新整个jsbundle文件。 缺点：体积大，费流量。
    - 2 增量更新 ： 只更新两个版本的包之间差量部分，然后客户端将安装包内base.bundle与差量包进行合并最终生成最新版本的包。 （ps: base.bundle 文件在打包apk时放在assets目录）

  优点：体积小，下载快。

什么是base.bundle,该如何生成?

```
import React, { Component } from 'react';
```

```
import { AppRegistry } from 'react-native';
```

构造一个base.js，里面的内容如上，

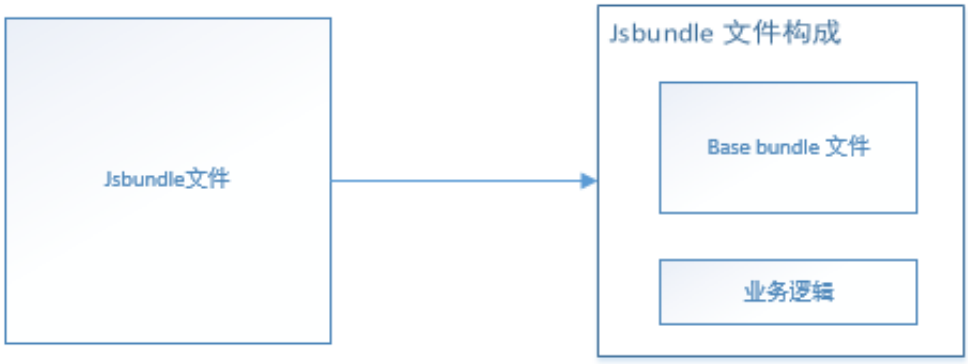
然后通过打包命令生成base.android.bundle.

打包命令长这个样子：

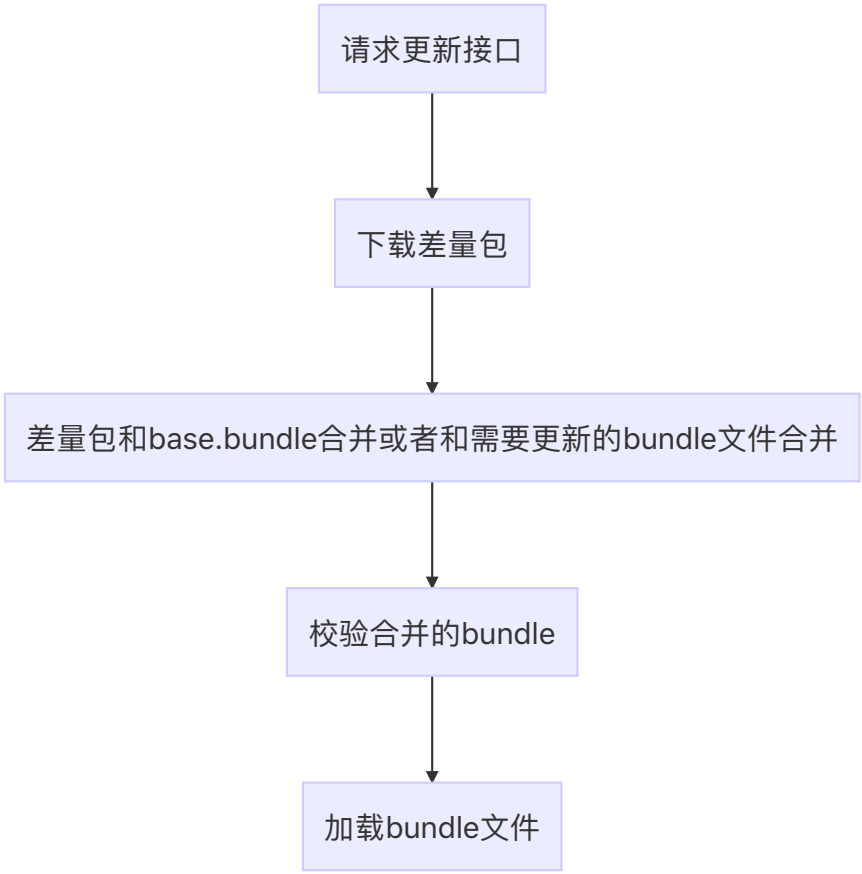
```
react-native bundle --platform android --dev false --entry-file
toutiao.android.js --bundle-output ./toutiao.android.bundle --
assets-dest ./app/res/
```

而我们平时打包出来的是一个包含了base.bundle 部分的一个完整bundle文件。而我们完全可以通过bsdiff算法将业务部分通过差分算法计算出来。

正因为上面的拆分，因此我们采取的是增量更新，因为打包出来的一个bundle文件，其中的base 文件一般情况是一样的。所以没必要每次更新带上base部分的bundle文件，而base.bundle文件的大小是500kb左右，而差量包的大小则只有几十kb大小（视具体业务量而定）。



热更新的整个流程如下：



二 细节介绍

第一步

请求更新接口，上传本地的业务包版本。

后台会去查询当前业务包的最新业务包，然后将客户端版本的业务包与最新版本的差量包下载链接下发下来，然后客户端进行下载。

请求中的参数如下：

```
{
    "pkg_name": "toutiao",
    "pkg_version": 0,
    "pkg_base": 2
}
```

请求参数的含义：

pkg_name：相当于那一个模块。

pkg_version ： 客户端包的版本。

pkg_base ： 客户端**base**的版本信息。


```
{
  "update_type": 2,
  "force": true,
  "max_version": 25,
  "download_url": "http://rn.xmiles.cn/reactNative/toutiao_0_android_25_base_2_1510019037937.patch",
  "md5": "1be279a2f6435f9e743feala3721b9b2",
  "buz_md5": "393ce2a64b95627f25415fcdb1fbf349",
  "result": {
    "status": 1
  },
  "costTime": 191
}
```


返回参数的含义

update_type ： 返回更新类型。

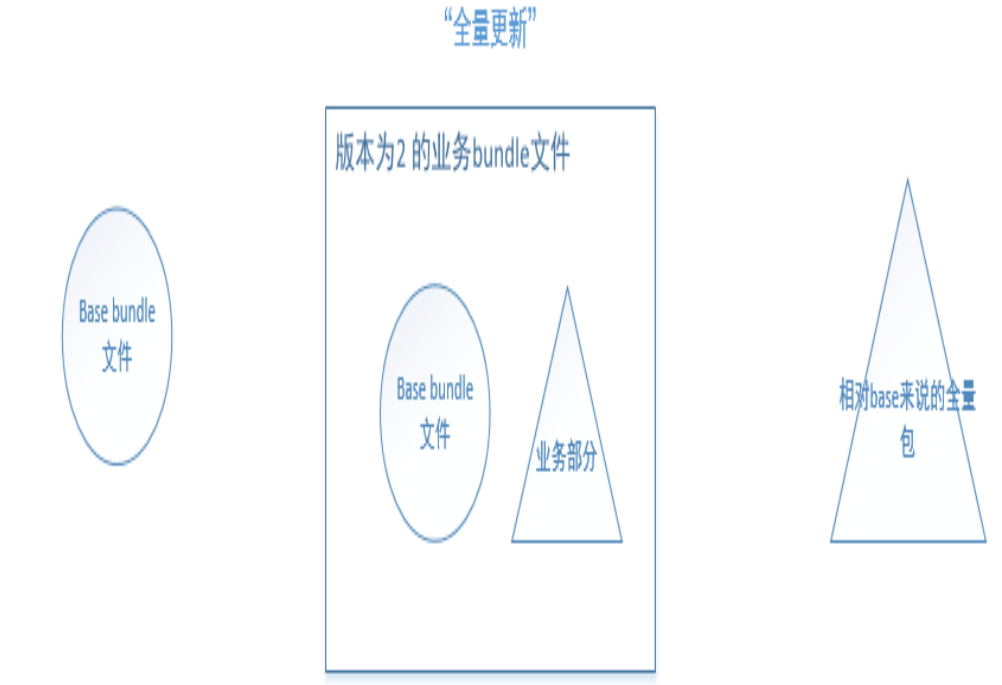
force ： 是否强制更新。

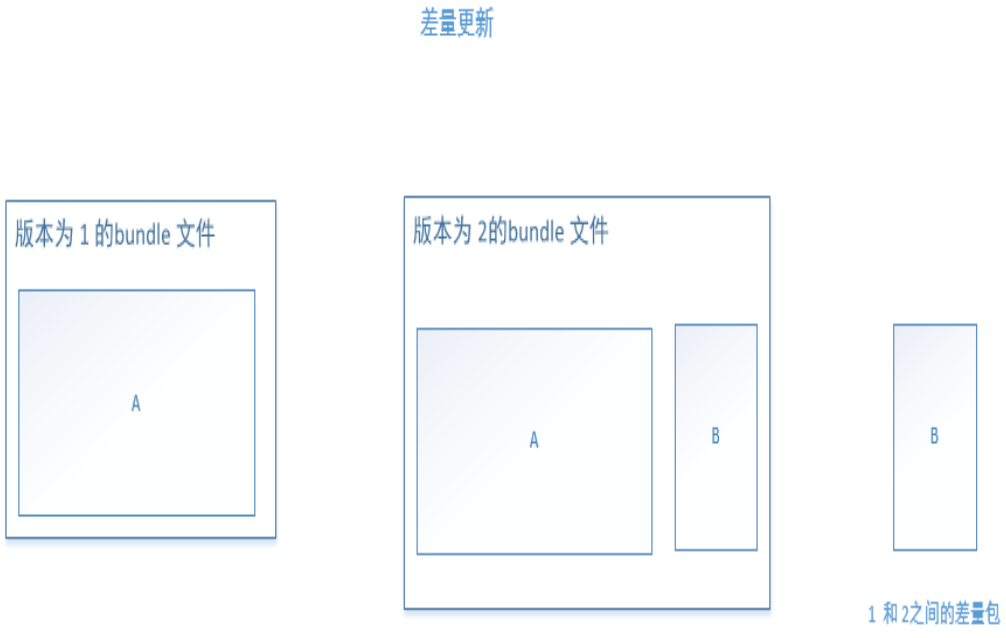
max_version ： 下载包的版本。

download_url ： 下载的**url**。

buz_md5: 版本为 max_version的业务逻辑md5值。（为了校验在合并完成以后是否是一个正确的业务包，如果不是则不进行加载，否则加载一个错误的业务包，会导致react native 造成crash。）

更新类型： 分为全量更新和非“全量更新”。





强制非强制：如果是强制更新，则等到下载完更新以后加载界面，如果是非强制更新则 直接加载界面，加载完界面以后再开始下载。

- 为什么要分强制更行和非强制更新?

这样做是为了让用户可以更快的看到界面而不是等到下载完以后才看见界面，还有应对如果出现了严重错误可以实现覆盖的作用。

在下载完以后并且进行和本地的合并完成以后将会校验这个合并的包的md5值如果一致，则进行加载否则进行回滚操作，回滚之前我们是将原有的bundle文件进行了备份的。

- 包文件如何管理?

这些包文件的管理都是通过文件名来进行组织的，例如 toutiao模块的包，名字将会命名为 toutiao.android.bundle

下载的头条包的名字将会是toutiao.download

备份文件的名字为：toutiao.bk.

通过这些特殊的名称组织方式可以方便的管理同一个bundle文件的不同状态。

而bundle 加载则是直接通过包名和固定路径拼接而成，view层只需要知道这个包名即。

三 常见问题解决方法

- 1 后台请求接口出现异常，这个时候如果请求返回异常，更新会立马终止并且会直接抛出异常，然后前端会直接显示服务器错误界面，这个地方可以优化，如果是接口有问题，可以先展示旧的界面，然后展示完成以后再进行重试等操作，当你使用时如果遇见界面空白的问题，不要慌，抓个包看下接口返回。
- 2 如果遇见问题从logcat的日志中可以看见更新走到那个步骤，还有看抛出的异常，sdk更新的所有抛出异常都是经过封装成了一个通用的更新异常类，其中抛出的errorcode会知道是哪里出现了异常。
- 3 可能出现md5校验异常，出现这个异常会有非常多的可能性，此时可以参考以下步骤处理。
    - 1 检查下载的差量包的base包是否与本地相同。

- 2 下载的bundle文件是否正确。如果不正确，则自己检查是否下载的包不完整等。
- 3 如果前面两步没有错，则使用bspatch工具将客户端和base包和差分包进行合并然后看是否与后台的包一致，如果不一致则有可能是后台生成的是错误的差分包，这个时候与后台接口提供者一起解决。

一些不足和改进：

1 出错提示部分不够明显，后期将会修改异常提醒部分，使出错原因更加直观。

2 react 更新部分，因为更新行为很少发生，而我们是在加载每一个react native 界面的时候都会去请求是否更新接口。而这个请求大多数的时候返回的是没有更新状态，这就造成了不必要的请求，和页面打开时间，以后可以考虑将这行行为放在每次打开应用的时候去更新所有有更新bundle文件，而不是在界面即将打开的时候去请求。

四 react native 如何升级

react native 版本在快速迭代中，这个时候不免会有升级操作，我们的这套更新升级分为三部。

- 1 base升级

```
1   删除已下载好的node_modules 模块。

2   更新package.json文件中react native版本（ps：有时候也同时也需要更新react 版本）。

3   再次执行npm install。（出现安装卡住的问题，删掉多试几次，或者切换成淘宝的源）

4   升级gradle中的compile 'com.facebook.react:react-native:*.*.*'。

5   升级RnConfig类中的base_bundle_version。

6   打包base.bundle。

7   升级manifest中的version_code 和version_name。

8   上传打包好base.bundle到后台。
```

- 2 多base打包

  1 有多个版本目录，每个目录下面对应有不同版本的node_modules 依赖。

  2 每个目录下面有相同业务逻辑部分，然后分别执行打包命令，将打包出不同平台的js.bundle文件。

五 消息中心跳转参数介绍

```
{
  "launch": "launch_vc_reactnative",
  "launchParams": {
     "pkgName":"test",                    //包名
     "component":"test",                  //组件名
     "moduleName":"index",                //模块名
     "title":"test",                      //标题
     "clearTop":0,                        //是否清楚栈顶，如果为1清除栈，如果为为0不清除栈
     "showTitle":1,                       //是否显示标题
     "extra":"附加信息"                   //自定义的额外信息
  }
});
```