

## 拆包实践

在amap中新建一个空白工程并全量转换，然后在workspace目录下创建base.js，引入App.js，同时引入可能需要使用到的依赖包，如下所示

```
import {AppRegistry} from 'react-native';
import 'react';
import 'react-native';
import React, {Fragment, Component} from 'react';
require('@ant-design/react-native');
require('axios');
require('create-react-class');
require('cssom');
require('lodash');
require('moment');
require('native-echarts');
require('prop-types');
require('rc-form');
require('react-dom');
require('react-native-calendar-select');
require('react-native-collapsible');
require('react-native-image-pan-zoom');
require('react-native-image-zoom-viewer');
require('react-native-keyboard-aware-scrollview');
require('react-native-menu');
require('react-native-refresh-list-view');
require('react-native-svg');
require('react-native-swiper');
require('react-native-webview');
require('react-navigation');
require('react-redux');
require('redux');
require('redux-actions');
require('redux-form');
require('redux-persist');
require('redux-thunk');
require('remote-redux-devtools');
require('rmc-date-picker');
require('rmc-tabs');
require('semver');
require('xmldom');
require('amap-directive');
require('amap-mobile');
require('amap-observe');
import './src/window';
import { WebSDK } from './src/sdk/webSDK';
import ErrorUtils from "ErrorUtils"
```

有了基础包入口文件base.js，然后就可以配置metro.config.js了

```
module.exports = {
  transformer: {
    getTransformOptions: async () => ({
      transform: {
        experimentalImportSupport: false,
        inlineRequires: false,
      },
    }),
  },
};
```

```

serializer:{
  createModuleIdFactory:() => (path) =>'amap\\'+path.split('\\target\\rn\\workspace\\')[1]
}
};

```

createModuleIdFactory方法作用是固定jsbundle中依赖包, 使之具有唯一值, 以便实现从全量包中过滤出业务包

以base.js为基础包入口文件, 通过命令执行打包, 得到base.xxx.jsbundle, 如下命令供参考 (ios是jsbundle, android是bundle)

```

react-native bundle --platform ios --dev false --entry-file base.js --bundle-output ./output/base.ios.jsbundle --assets-dest ./output/
react-native bundle --platform android --dev false --entry-file base.js --bundle-output ./output/base.android.bundle --assets-dest ./output/

```

记下来获取微应用的全量包, 在amap中打开微应用工程并全量转换, 以index.js为入口文件 (有原生操作系统名后缀优先, 如ios平台下index.ios.js) 执行打包命令, 如下命令供参考 (ios是jsbundle, android是bundle)

```

react-native bundle --platform ios --dev false --entry-file index.ios.js --bundle-output ./output/index.ios.jsbundle --assets-dest ./output/
react-native bundle --platform android --dev false --entry-file index.android.js --bundle-output ./output/index.android.bundle --assets-dest ./output/

```

至此得到了微应用的全量包index.xxx.jsbundle

接下来是以基础包base.xxx.jsbundle过滤全量包index.xxx.jsbundle, 得到业务包business.xxx.jsbundle

过滤方法, 通过node实现, diff.js代码参考网上, 如下

```

// diff.js: 找出common和business的不同行
var fs = require('fs');
var readline = require('readline');

```

```

function readFileToArr(fReadName,callback){
  var fRead = fs.createReadStream(fReadName);
  var objReadline = readline.createInterface({
    input:fRead
  });
  var arr = new Array();
  objReadline.on('line',function (line) {
    arr.push(line);
    //console.log('line:'+ line);
  });
  objReadline.on('close',function () {
    // console.log(arr);
    callback(arr);
  });
}

```

```

var argvs = process.argv.splice(2);
var commonFile = argvs[0]; // common.ios.jsbundle

```

```

var businessFile = argvs[1]; // business.ios.jsbundle
var diffOut = argvs[2]; // diff.ios.jsbundle
readFileToArr(commonFile, function (c_data) {
  var diff = [];
  var commonArRs = c_data;
  readFileToArr(businessFile, function (b_data) {
    var businessArRs = b_data;
    for (let i = 0; i < businessArRs.length; i++) {
      if (commonArRs.indexOf(businessArRs[i]) === -1) { // business中独有的行
        diff.push(businessArRs[i]);
      }
    }
    // console.log(diff.length);
    var newContent = diff.join('\n');
    fs.writeFileSync(diffOut, newContent); // 生成diff.ios.jsbundle
  });
});

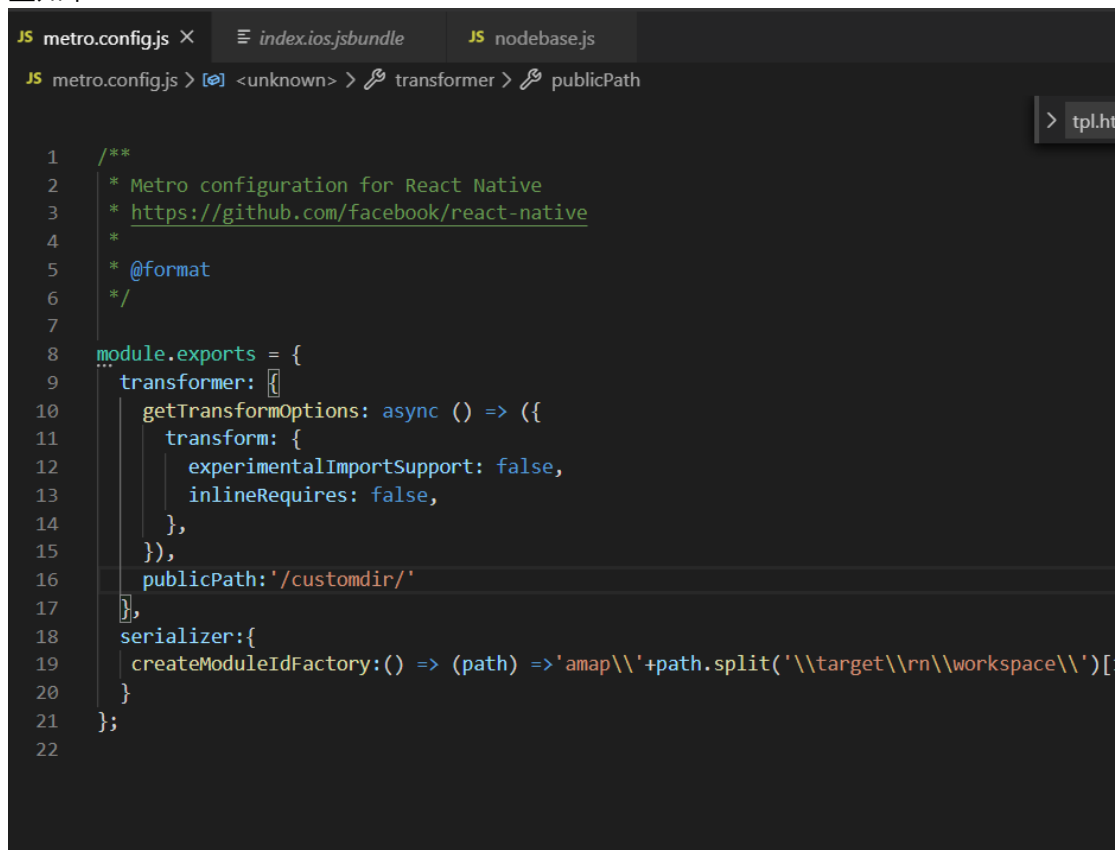
```

过滤业务包方法执行node命令, 以下命令仅供参考

node diff.js ./output/base.xxx.jsbundle ./output/index.xxx.jsbundle ./output/diff.xxx.jsbundle

关于--assets-dest 这块的metro.config配置

查看metro源码, 可以修改输出静态资源存放文件夹的名字, 通过publicPath实现, 具体配置如下



```

1  /**
2   * Metro configuration for React Native
3   * https://github.com/facebook/react-native
4   *
5   * @format
6   */
7
8  module.exports = {
9    transformer: {
10     getTransformOptions: async () => ({
11       transform: {
12         experimentalImportSupport: false,
13         inlineRequires: false,
14       },
15     }),
16     publicPath: '/customdir/'
17   },
18   serializer: {
19     createModuleIdFactory: () => (path) => 'amap\\'+path.split('\\target\\rn\\workspace\\')[
20   ]
21 };
22

```

Github metro 地址

<https://github.com/facebook/metro/tree/master/packages/metro/src>

相关源码截图如下, 便于学习参考

```
const assetData = {
  __packager_asset: true,
  fileSystemLocation: path.dirname(assetPath),
  httpServerLocation: assetUrlPath,
  width: dimensions ? dimensions.width / scale : undefined,
  height: dimensions ? dimensions.height / scale : undefined,
  scales: assetInfo.scales,
  files: assetInfo.files,
  hash: assetInfo.hash,
  name: assetInfo.name,
  type: assetInfo.type,
};

let assetUrlPath = localPath.startsWith('.')
  ? publicPath.replace(/\/$/, '') + '/' + path.dirname(localPath)
  : path.join(publicPath, path.dirname(localPath));

// On Windows, change backslashes to slashes to get proper URL path from file path.
if (path.sep === '\\') {
  assetUrlPath = assetUrlPath.replace(/\\/g, '/');
}
```

```

async function getAssets(
  graph: Graph<>,
  options: Options,
): Promise<$ReadOnlyArray<AssetData>> {
  const promises = [];
  const {processModuleFilter} = options;

  for (const module of graph.dependencies.values()) {
    if (
      isJsModule(module) &&
      processModuleFilter(module) &&
      getJsOutput(module).type === 'js/module/asset' &&
      path.relative(options.projectRoot, module.path) !== 'package.json'
    ) {
      promises.push(
        getAssetData(
          module.path,
          path.relative(options.projectRoot, module.path),
          options.assetPlugins,
          options.platform,
          options.publicPath,
        ),
      );
    }
  }

  return await Promise.all(promises);
}

module.exports = getAssets;

```