**通过babel 转译es6语法**

结论：babel支持大部分的es6转译，但通过transform-runtime实现的Promise等语法转译对IE8有兼容问题。

出现兼容问题的原因：

对方法的调用中使用了 default作为属性名，如

function \_interopRequireDefault(obj) { return obj && obj.\_\_esModule ? obj : { default: obj }; }

在IE8中，default可能作为关键字，出现在属性名中会报错

修复：需要对default属性使用字符串形式。如：{ ‘default’: obj }

exports[‘default’] = …

1. 基础版

通过gulp，babel preset进行转译

缺陷：不支持class、promise、set、map等语法

原因：babel是通过调用插件的方式，引用node-module中解析e6语法的相应的js

而preset只是对某一类插件包装在一个preset，（如es2015中，不包含对promise等语法解析的js）

npm install babel-preset-stage-0 babel-preset-stage-1 –save-dev

用于实现对部分es7 语法的支持。如class中实例属性和静态属性的定义

class Cls(){

state1 = {}  
 static state2 = {}

}

gulp配置：

gulp.src( file )

.pipe( babel( {

presets: ['es2015', 'react', ‘stage-1’]

} ).on('error', (e) => {

console.error('error', e.message);

}) )

.pipe(gulp.dest( filePath ));

1. 通过gulp配置特殊语法支持

注：网上很多地方会要求再安装 babel-runtime ，但在新版babel中，babel-runtime和babel-core似乎已经合并到gulp-babel中。故不需要另外安装，也无法安装

缺陷：使用transform-runtime 是通过 commonjs中的require语法来引用node-module中的解析文件来解析es6。而gulp本身是没有模块管理功能的，所以需要引用其他工具来实现模块管理功能。常用的有webpack和browserify

npm install babel-plugin-transform-runtime –save-dev

gulp配置：

gulp.src( file )

.pipe( babel( {

presets: ['es2015', 'react'],

plugins: [‘transform-runtime’]

} ).on('error', (e) => {

console.error('error', e.message);

}) )

.pipe(gulp.dest( filePath ));

1. 通过gulp + webpack来实现模块管理

在gulp中使用webpack需要安装webpack-stream（不安装直接使用webpack 也可以，但webpack-stream更符合gulp的pipe书写方式）

注：vinyl-named 用于实现输出文件与原文件同名。

缺点：将模块管理和es6编译分别使用webpack、gulp来实现，相对繁琐。webpack仅作为模块管理使用，有些大材小用，没有发挥其价值。能实现promise等语法的转译，但对 includes 等es7方法的支持依然没有实现

npm install webpack-stream –save-dev

gulp配置：

var matchRex = /(src.\*)\/.\*\.(\w+)/;

var babel = require('gulp-babel');

var babelTask = (e) => {

var match = e.path.replace(/\\/g, '/').match( matchRex ),file, filePath;

if(match.length) {

file = match[0]; // src/dataRow.jsx

filePath = match[1];

gulp.src( file )

.pipe( babel( {

presets: ['es2015', 'react'],

plugins: ['transform-runtime'] // 'transform-es2015-modules-umd'

} ).on('error', (e) => {

console.error('error', e.message);

}) )

.pipe(gulp.dest( filePath ));

}

};

// webpack

var webpack = require('webpack-stream');

var named = require('vinyl-named');

var webPackTask = (e) => {

gulp.src(['src/\*\*/\*.js', '!src/\*\*/react\*.js', '!src/\*\*/reflux\*.js'])

.pipe(named())

.pipe(webpack())

.pipe(gulp.dest('dist'));

};

// watch

gulp.task('watch', function() {

// babel & jsx

gulp.watch(['src/\*\*/\*.jsx', 'src/\*\*/\*.es6']).on('change', (event) => {

console.log('File ' + event.path + ' was ' + event.type + ', running tasks...[babel]');

babelTask(event);

console.log('File ' + event.path + ' was ' + event.type + ', running tasks...[webpack]');

webPackTask(event);

});

});

1. gulp实现对 webpack 的集成，babel的转译工作在webpack中完成

通过webpack.config.js做相应配置，需要使用babel-loader，

缺点：对 includes 等es7方法的支持依然没有实现

npm install babel-loader –save-dev

gulp配置：

// webpack

var webpack = require('webpack-stream');

var named = require('vinyl-named');

var webPackTask = (e) => {

gulp.src(['src/\*\*/\*.js', '!src/\*\*/react\*.js', '!src/\*\*/reflux\*.js'])

.pipe(named())

.pipe(webpack(require(‘./webpack.config.js’)))

.pipe(gulp.dest('dist'));

};

gulp.task('watch', function() {

// babel & jsx

gulp.watch(['src/\*\*/\*.jsx', 'src/\*\*/\*.es6']).on('change', (event) => {

console.log('File ' + event.path + ' was ' + event.type + ', running tasks...[webpack]');

webPackTask(event);

});

});

webpack 配置：

module.exports = {

watch: true,

devtool: "source-map",

resolve: {

extensions: ['', '.js', '.jsx', '.es6']

},

module: {

loaders: [{

test: /\.jsx$/,

loader: 'babel',

exclude: /(node\_modules)/,

query: {

presets: ['es2015', 'react', ‘stage-1’]

}

}, {

test: /\.es6$/,

loader: 'babel',

exclude: /(node\_modules)/,

query: {

presets: ['es2015']

}

}]

}

};

1. 最终版

使用babel-polyfill 转译项目代码，在项目文件的最上面添加 require("babel-polyfill");

babel-polyfill支持所有es6语法的转译，包括Set、Map、Array.includes等。不过很多方法在项目中不一定需要，会大大增加加载文件的大小。所以建议在SPA项目中使用，其他项目中不适合

替代方案：

1. 添加 plugin:[‘transform-runtime’]
2. 对项目中使用到的Array.includes/Array.from/Object.assign等方法，在文件开头添加

require('core-js/fn/array/includes');

即用到什么es6对象方法，手动添加对应的引用