webpack工程化25道重点面试突击巩固题 (https://github.com/minsion/interview-special)

面试题 1. 简述前端自动化构建工具 ?

推荐指数: ★★★ 试题难度: 初级 试题类型: 原理题

试题回答参考思路:

前端自动化构建工具有:
Bower，Gulp，Grunt，node，yeoman

自动化构建是作为前端工程化中重要的部分，承担着若干需要解决的环节。包括流程管理、版本管理、资源管理、组件化、脚手架工具等。

1：流程管理
完整的开发流程包括本地开发，mock调试，前后端联调，提测，上线等。在每个团队的基础设施当中(如cms系统，静态资源推送系统等)，都会存在一定程度将前端开发流程割裂。如何运用自动化的手段，对开发流程进行改善将可以大幅减低时间成本。

2：版本管理
web应用规模愈加复杂，迭代速度也愈加频繁。而前端界面作为一种远程部署，运行时增量下载的特殊GUI软件，如何使用自动化构建工具，对不同版本的资源文件进行管理，并让用户第一时间感知版本的回撤以及升级(尤其是在浏览器缓存以及cdn广泛使用的今天），将对企业有更好的安全保障，对用户有更佳的使用体验。

3：资源管理
随着每个团队业务复杂程度的加深，对于功能模块封装的粒度必将愈加精细。衍生出来的，将会是资源数量以及依赖关系等的管理问题。以人工的方式考虑单个页面或单个功能的资源优化是片面的，并且效率低下。通过工程化的手段，在前端构建过程中自动化地以最优方式处理资源的合并以及依赖，是提升性能以及解放人力资源的重要途径。

4：组件化
组件化方案，是以页面的小部件为单位进行开发，在系统内可复用。如何以最优化方式实现组件化（js、css、html片段，以就近原则进行文件组织，以数据绑定方式进行代码开发，业务逻辑上相对外部独立，暴露约定的接口）；并且随着组件化的程度加深，如何对组件库进行管理，合并打包以及多人共同维护等，都是无法避免的问题。

5：脚手架工具
我们希望每次研发新产品不是从零开始，不同团队不同项目之间能有【可复用的模块】沉淀下来。对于前端而言，【可复用的模块】除了【组件】，另外就是【脚手架工具】。运用脚手架工具，一键安装，自动化搭建不同类型项目的完整目录结构，工程师将有更多时间专注在业务逻辑代码的编写上。

面试题 2. 简述WebPack的I理解和认识?

推荐指数: ★★★★★ 试题难度: 中级 试题类型: 原理题

试题回答参考思路：

WebPack是一个模块打包工具，可以使用 WebPack管理模块依赖，并编译输出模块所需的静态文件。它能够很好地管理与打包Web开发中所用到的HTML、 JavaScript 、CSS以及各种静态文件（图片、字体等），让开发过程更加高效。对于不同类型的资源，WebPack有对应的模块加载器。Web Pack模块打包器会分析模块间的依赖关系，最后生成优化且合并后的静态资源。
WebPack的两大特色如下。
（1）代码切割（ code splitting）
（2） loader可以处理各种类型的静态文件，并且支持串行操作WebPack以 CommonJS规范来书写代码，但对 AMD/CMD的支持也很全面，方便对项目进行代码迁移。
WebPack具有 require.js和 browserify的功能，但也有很多自己的新特性，
（1）对 CommonJS、AMD、ES6的语法实现了兼容。
（2）对 JavaScript、CSS、图片等资源文件都支持打包
（3）串联式模块加载器和插件机制，让其具有更好的灵活性和扩展性，例如提供对CoffeeScript、 EMAScript 6的支持
（4）有独立的配置文件 webpack.config. js。
（5）可以将代码切割成不同的块，实现按需加载，缩短了初始化时间。
（6）支持 SourceUrls和 SourceMaps，易于调试。
（7）具有强大的 Plugin接口，大多是内部插件，使用起来比较灵活
（8）使用异步I/O，并具有多级缓存，这使得 WebPack速度很快且在增量编译上更加快。

面试题 3. 简述在使用 WebPack时，常见的应用场景？

推荐指数：★★★ 试题难度：初级 试题类型：原理题

试题回答参考思路：

用来压缩合并CSS和 JavaScript代码，压缩图片，对小图生成base64编码，对大图进行压缩，使用 Babel把 EMAScript 6编译成 EMAScript 5，热重载，局部刷新等。在 output中配置出口文件，在 entry中配置入口文件。
使用各种 loader对各种资源做处理，并解析成浏览器可运行的代码

面试题 4. 简述Gulp都实现了哪些功能?

推荐指数：★★★★ 试题难度：中级 试题类型：原理题

试题回答参考思路：

1.用自动化构建工具增强你的工作流程！
2.gulp 将开发流程中让人痛苦或耗时的任务自动化，从而减少你所浪费的时间、创造更大价值。
3.基于node强大的流(stream)能力，gulp在构建过程中并不把文件立即写入磁盘，从而提高了构建速度。
gulp官网：https://www.npmjs.com/

gulp有哪些功能?

1.文件复制
2.html压缩、css编译压缩、js合并压缩
3.优化图片-压缩图片
4.编译sass
5.es6转换es5

面试题 5. 简述WabPack打包的流程？

推荐指数：★★★★★ 试题难度：中级 试题类型：原理题

试题回答参考思路：

具体流程如下。
（1）通过 entry配置入口文件。
（2）通过 output指定输出的文件。
（3）使用各种 loader处理CSS、 JavaScript、 image等资源，并将它们编译与打包成浏览器可以解析的内容等

面试题 6. 简述WebPack的核心原理？

推荐指数：★★★★★ 试题难度：高难 试题类型：原理题

试题回答参考思路：

（1）一切皆模块。
正如 JavaScript文件可以是一个“模块”（ module）一样，其他的（如CSS、 image或HTML）文件也可视作模块。因此，可以执行 require（'myJSfile js'），亦可以执行require（ 'myCSSfile.css'）。这意味着我们可以将事务（业务）分割成更小的、易于管理的片段，从而达到重复利用的目的。
（2）按需加载。
传统的模块打包工具（ module bundler）最终将所有的模块编译并生成一个庞大的bundle. js文件。但是，在真实的App里， bundle. js文件的大小在10MB到15MB之间，这可能会导致应用一直处于加载状态。因此， WebPack使用许多特性来分割代码，然后生成多个 bundle js文件，而且异步加载部分代码用于实现按需加载

面试题 7. 简述WebPack中 loader的作用？

推荐指数：★★★★ 试题难度：中级 试题类型：原理题

试题回答参考思路：

具体作用如下。
（1）实现对不同格式文件的处理，比如将Scss转换为CSS，或将 TypeScript转化为Javascript。
（2）可以编译文件，从而使其能够添加到依赖关系中。loader是 WebPack最重要的部分之一。通过使用不同的 loader，我们能够调用外部的脚本或者工具，实现对不同格式文件的处理。loader需要在 webpack.config.js里单独用 module进行配置

面试题 8. 叙述工作中几个常用的 loader ？

推荐指数：★★★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

常用的 loader如下：
babel- loader：将下一代的 JavaScript语法规范转换成现代浏览器能够支持的语法规范。因为 babel有些复杂，所以大多数开发者都会新建一个. babelrc进行配置。
css-loader、 style- loader：这两个建议配合使用，用来解析CSS文件依赖。
less- loader：解析less文件。
file- loader：生成的文件名就是文件内容的MD5散列值，并会保留所引用资源的原始扩展名。
url- loader：功能类似于file-loader，但是当文件大小低于指定的限制时，可以返回一个DataURL

面试题 9. 简述plugins和 loader有什么区别?

推荐指数：★★★★★ 试题难度： 高难 试题类型： 原理题

试题回答参考思路：

不同的作用：

Loader直译为"加载器"。Webpack将一切文件视为模块，但是webpack原生是只能解析js文件，如果想将其他文件也打包的话，就会用到loader。 所以Loader的作用是让webpack拥有了加载和解析非JavaScript文件的能力。

Plugin直译为"插件"。Plugin可以扩展webpack的功能，让webpack具有更多的灵活性。在 Webpack 运行的生命周期中会广播出许多事件，Plugin 可以监听这些事件，在合适的时机通过 Webpack 提供的 API 改变输出结果

loader比较单一就是用来加载文件

不同的用法：

Loader在module.rules中配置，也就是说他作为模块的解析规则而存在。 类型为数组，每一项都是一个Object，里面描述了对于什么类型的文件（test），使用什么加载(loader)和

使用的参数（options）

Plugin在plugins中单独配置。 类型为数组，每一项是一个plugin的实例，参数都通过构造函数传入

常见的loader和plugin插件

Loader:

样式：style-loader、css-loader、less-loader、sass-loader等
文件：raw-loader、file-loader 、url-loader等
file-loader、url-loader等可以处理资源

file-loader可以复制和放置资源位置，并可以指定文件名模板，用hash命名更好利用缓存。
url-loader可以将小于配置limit大小的文件转换成内敛Data Url的方式，减少请求。
raw-loader可以将文件已字符串的形式返回
编译：babel-loader(把 ES6 转换成 ES5)、coffee-loader 、ts-loader等
vue-loader、coffee-loader、babel-loader等可以将特定文件格式转成js模块、将其他语言转化为js语言和编译下一代js语言

校验测试：mocha-loader、jshint-loader 、eslint-loader等
imports-loader、exports-loader等可以向模块注入变量或者提供导出模块功能
Plugin:

webpack内置UglifyJsPlugin，压缩和混淆代码,通过UglifyES压缩ES6代码。
webpack内置CommonsChunkPlugin，提取公共代码,提高打包效率，将第三方库和业务代码分开打包
ProvidePlugin：自动加载模块，代替require和import
new webpack.ProvidePlugin({
$: 'jquery',
jQuery: 'jquery'
})
html-webpack-plugin可以根据模板自动生成html代码，并自动引用css和js文件
extract-text-webpack-plugin 将js文件中引用的样式单独抽离成css文件
DefinePlugin编译时配置全局变量，这对开发模式和发布模式的构建允许不同的行为非常有用。
HotModuleReplacementPlugin 热更新
添加HotModuleReplacementPlugin
entry中添加 "webpack-dev-server/client?http://localhost:8080/",
entry中添加 "webpack/hot/dev-server"
(热更新还可以直接用webpack_dev_server --hot --inline,原理也是在entry中添加了上述代码)

面试题 10. 简述HtmlWebpackPlugin插件的作用 ？

推荐指数： ★★★★ 试题难度： 中级 试题类型： 原理题 ▸

试题回答参考思路：

作用一：
为html文件中引入的外部资源如script、link动态添加每次compile后的hash，防止引用缓存的外部文件问题

作用二：
可以生成创建html入口文件，比如单页面可以生成一个html文件入口，配置N个html-webpack-plugin可以生成N个页面入口

面试题 11. 简述WebPack支持的脚本模块规范?

推荐指数：★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

不同项目在定义脚本模块时使用的规范不同。有的项目会使用 Commonjs规范（参考Node. js）；有的项目会使用 EMAScript 6模块规范；有的还会使用AMD规范（参考Require. js）。WebPack支持这3种规范，还支持混合使用

面试题 12. 如何为项目创建 package. json文件?

推荐指数：★★★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

将命令行切换至根目录下，运行 npm init，命令行就会一步一步引导你建立package. json文件。手动在根目录下创建一个空文件，并命名为 package. json，在文件中填充JSON格式的常规内容。例如初期只需要name和 version字段。

```
{
"name": "Project",
"version": " 0.0.1"
}
```

面试题 13. 简述WebPack和 gulp/grunt相比有什么特性?

推荐指数：★★★ 试题难度： 初级 试题类型： 原理题

试题回答参考思路：

gulp/ grunt是一种能够优化前端的流程开发工具，而 Web Pack是一种模块化的解决方案，由于 WebPack提供的功能越来越丰富，使得 WebPack可以代替 gulp/grunt类的工具

面试题 14. 描述grunt和gulp的工作方式 ？

推荐指数：★★★★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

Grunt
https://www.gruntjs.net/
是一套前端自动化工具，一个基于nodeJs的命令行工具，一般用于：
① 压缩文件
② 合并文件
③ 简单语法检查

Gulp
https://www.gulpjs.com.cn/
1.介绍：
Gulp 是基于node.js的一个前端自动化构建工具，开发这可以使用它构建自动化工作流程（前端集成开发环境）。
使用gulp你可以简化工作量，让你把重点放在功能的开发上，从而提高你的开发效率和工作质量。
2.特性：

使用方便
通过代码优于配置的策略，Gulp可以让简单的任务简单，复杂的任务更可管理。
构建快速
通过流式操作，减少频繁的 IO 操作，更快地构建项目。
插件高质
有严格的插件指导策略，确保插件能简单高质的工作。
易于学习
少量的API，掌握Gulp可以毫不费力。构建就像流管道一样，轻松加愉快。
Grunt与Gulp的区别：
Gulp 和 Grunt 类似。但相比于 Grunt 的频繁的 IO 操作，Gulp 的流操作，能更快地完成构建。
Gulp 相比于 Grunt 有很多优点，比较直观的：就是学习曲线比较平滑。比Grunt速度更快、配置更少。

面试题 15. 简述WebPack工具中常用到的插件有哪些?

推荐指数：★★★★★ 试题难度：高难 试题类型：原理题

试题回答参考思路：

常用到的插件如下
（1） HtmlWebpackPlugin：依据一个HTML模板，生成HTML文件，并将打包后的资源文件自动引入。
（2） commonsChunkPlugin：抽取公共模块，减小包占用的内存空间，例如vue的源码、 jQuery的源码等。
（3） css-loader：解析CSS文件依赖，在 JavaScript中通过 require方式引入CSS文件。
（4） style- loader.：通过 style标签引入CSS。
（5） extract-text-webpack- plugin：将样式抽取成单独的文件。

（6）url- loader：实现图片文字等资源的打包，limit选项定义大小限制，如果小于该限制，则打包成base64编码格式；如果大于该限制，就使用file- loader去打包成图片。
（7） hostess：实现浏览器兼容。
（8） babel：将 JavaScript未来版本（ EMAScript6、 EMAScript2016等）转换成当前浏览器支持的版本。
（9） hot module replacement：修改代码后，自动刷新、实时预览修改后的效果
（10） ugliifyJsPlugin：压缩 JavaScript代码

面试题 16. 简述如何用 webpack-dev- server监控文件编译?

推荐指数： ★★ 试题难度： 初级 试题类型： 原理题

试题回答参考思路：

打开多个控制台，用 webpack--watch实时监控文件变动，并随时编译

面试题 17. 如何修改 webpack-dev- server的端口?

推荐指数： ★★ 试题难度： 初级 试题类型： 原理题

试题回答参考思路：

用--port修改端口号，如 webpack-dev-server--port888

面试题 18. 简述WebPack中的publicPath ?

推荐指数： ★★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

在 WebPack自动生成资源路径时，比如由于 WebPack异步加载分包而需要独立出来的块，或者打包CSS时， WebPack自动替换掉的图片、字体文件，又或者使用html-webpack-plugin后 WebPack自动加载的入口文件等，这些 WebPack生成的路径都会参考 publicPath参数。不需要关注CDN，需要关注的是，文件发布出来后，应该部署到哪里。如果文件是与页面放到一起的，那么可以按相对路径来设置，比如'./'之类的；而如果JavaScript、CSS文件用于存放CDN，当然就要填写CDN的域名和路径

面试题 19. 描述WebPack如何切换开发环境和生产环境?

推荐指数： ★★★ 试题难度： 初级 试题类型： 原理题

试题回答参考思路：

生产环境与开发环境的区别无非就是调用的接口地址、资源存放路径、线上的资源是否需要压缩等方面。目前的做法是通过在 package. json中设置node的一个全局变量，然后在webpack. config. js文件里面进行生产环境与开发环境的配置切换

面试题 20. WebPack命令的-- config选项有什么作用?

推荐指数: ★★ 试题难度: 初级 试题类型: 原理题

试题回答参考思路:

-- config用来指定一个配置文件，代替命令行中的选项，从而简化命令。如果直接执行WebPack, WebPack会在当前目录下查找名为 webpack. config. js的文件

面试题 21. 请问Babel通过编译能达到什么目的?

推荐指数: ★★★★ 试题难度: 初级 试题类型: 原理题

试题回答参考思路:

能达到以下目的。

（1）使用下一代的 JavaScript标准（ EMAScript 6、 EMAScript 7）语法，当前的浏览器尚不完全支持这些标准。
（2）使用基于 JavaScript进行拓展的语言，比如 React的jsx语法

面试题 22. 当使用Babel直接打包的 JavaScript文件中含有jsx语法的时候会报错，如何解决这个问题?

推荐指数: ★★★★ 试题难度: 高难 试题类型: 原理题

试题回答参考思路:

修改 package. json并添加 react，如以下代码所示:

```
"babel": {
"presets": [
"es2015",
"react",
"stage-o"
],
"plugins" : [
"add-module-exports"
]
}
```

面试题 23. 简述WebPack与gulp的区别？

推荐指数: ★★★★★ 试题难度: 中级 试题类型: 原理题

试题回答参考思路:

区别如下：

（1）用途不同。gulp是工具链，可以配合各种插件使用，例如对 JavaScript、CSS文件进行压缩，对less进行编译等；而 WebPack能把项目中的各种 JavaScript、CSS文件等打包合并成一个或者多个文件，主要用于模块化开发。

（2）侧重点不同。gulp侧重于整个过程的控制管理（像是流水线），通过配置不同的任务，构建整个前端开发流程，并且gulp的打包功能是通过安装gulp-webpack来实现的；WebPack则侧重于模块打包。

（3） WebPack能够按照模块的依赖关系构建文件组织结构

面试题 24. 简述export、 export default和 module.export的区别是什么?

推荐指数：★★★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

export、 export default都属于 EMAScript 6 模块化开发规范。
export和 export default的区别如下。
在同一个文件里面可以有多个 export，一个文件里面只能有1个 export default。
使用 import引入的方式也有点区别。
在使用 export时，用 import引入的相应模块名字一定要和定义的名字一样；而在使用 export default时，用 import引入的模块名字可以不一样。
module. export属于 CommonJS语法规范

面试题 25. 简述webpack 热更新原理，是如何做到在不刷新 浏览器的前提下更新页面的？

推荐指数：★★ 试题难度： 中级 试题类型： 原理题

试题回答参考思路：

1. 当修改了一个或多个文件；
2. 文件系统接收更改并通知 webpack；
3. webpack 重新编译构建一个或多个模块，并通知 HMR 服务器进行更新；
4. HMR Server 使用 webSocket 通知 HMR runtime 需要更新，HMR 运行时通过 HTTP 请求更新 jsonp；
5. HMR 运行时替换更新中的模块，如果确定这些模块无法更新，则触发整个页面刷新