## Webpack相关：

### loader是什么?有哪些常见的loader?

**loader是模块转换器，用于把模块原内容按照需求转换成新内容。**（**通过使用不同的loader，webpack可以调用外部的脚本或工具，实现对不同格式的文件的处理**，比如说分析转换sass为css，或者把下一代的JS文件（ES6，ES7)转换为现代浏览器兼容的JS文件，对React的开发而言，合适的Loaders可以把React的中用到的JSX文件转换为JS文件。）

babel-loader：把 ES6 转换成 ES5

css-loader：加载 CSS，使用类似@import 和 url(...)的方法实现require()的功能，**支持模块化、压缩、文件导入等特性**。

style-loader：把 CSS 代码注入到 JavaScript 中，通过 DOM 操作去加载 CSS。（**动态创建 style 标签，将 css 插入到 head 中**，将所有的计算后的样式加入页面中）

postcss-loader：配合 autoprefixer 插件，自动生成浏览器兼容性前缀（自动补齐 CSS3 前缀）

file-loader：把图片和字体等本地资源文件输出到一个文件夹中，在代码中通过相对 URL 去引用输出的文件 (处理图片和字体)

url-loader：与 file-loader 类似，区别是用户可以设置一个阈值，大于阈值时返回其 publicPath，小于阈值时将文件 转换为base64 形式编码 (处理图片和字体)

### plugin是什么？有哪些常见的plugin

plugin插件，是用来拓展webpack的功能的，在 Webpack 构建流程中的特定时机会广播出对应的事件，插件可以监听这些事件的发生，在特定时机做对应的处理。

html-webpack-plugin：简化 HTML 文件创建 (依赖于 html-loader)。依据一个简单的html模板，自动生成一个引用你打包后的JS文件的新的html。

clean-webpack-plugin: 清理目录。

copy-webpack-plugin：静态资源拷贝。（将本地已有的js,css文件拷贝至打包的输出目录中）

mini-css-extract-plugin：抽离样式文件，将CSS文件单独打包为独立文件，支持按需加载。

### loader和plugin的区别？（必问）

loader是在打包构建过程中用来处理源文件的（TypeScript,JSX，Sass，Less..），一次处理一个。因为webpack本身只能打包CommonJS规范的js文件，对于其他类型资源，例如css，图片等，是没有办法加载的，这就需要对应的loader充当翻译官，**对这些资源进行转译的预处理工作**。loader只专注于转换源文件，处理单一文件的输入输出

plugin用于扩展webpack的功能，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，在特定的时机执行相应的任务（通过 Webpack 提供的 API ）改变打包的输出结果。**plugin并不直接操作单个文件，而对整个构建过程起作用**。比如mini-css-extract-plugin，可以将所有文件中的css抽离到一个独立的文件中。（这样样式就不会随着组件加载而加载了。）

### Webpack构建流程

Webpack的运行流程是一个**串行**的过程，从启动到结束可以概括为以下三步：

* + **初始化**：**启动构建**，读取与合并配置参数，加载插件，实例化 Compiler
  + **编译**：从入口文件Entry 出发，针对每个模块串行**调用**对应的 Loader翻译文件的内容，再找到该模块依赖的模块，**递归地进行编译处理**。
  + **输出**：将**编译后的模块组合**成 Chunk，再把每个Chunk 转换成单独的文件添加到输出列表中。

在以上过程中，Webpack 会在特定的时间点广播出特定的事件，**插件在监听到感兴趣的事件后会执行特定的逻辑**，并且插件可以调用 Webpack 提供的 API **改变 Webpack 的运行结果**。

具体：

