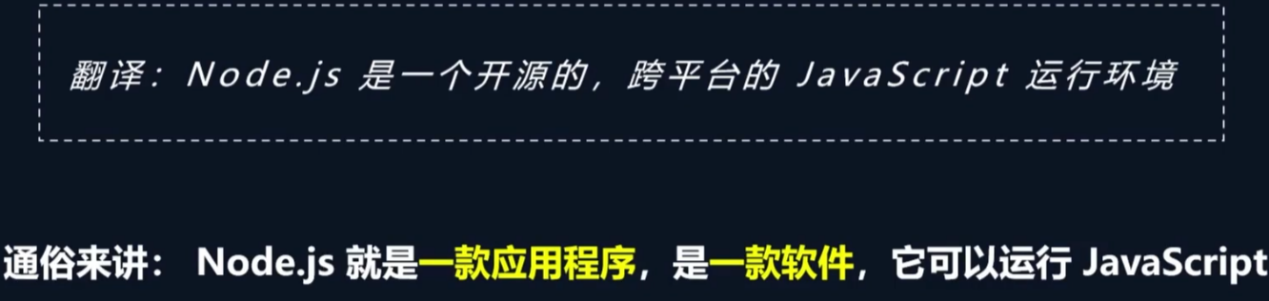
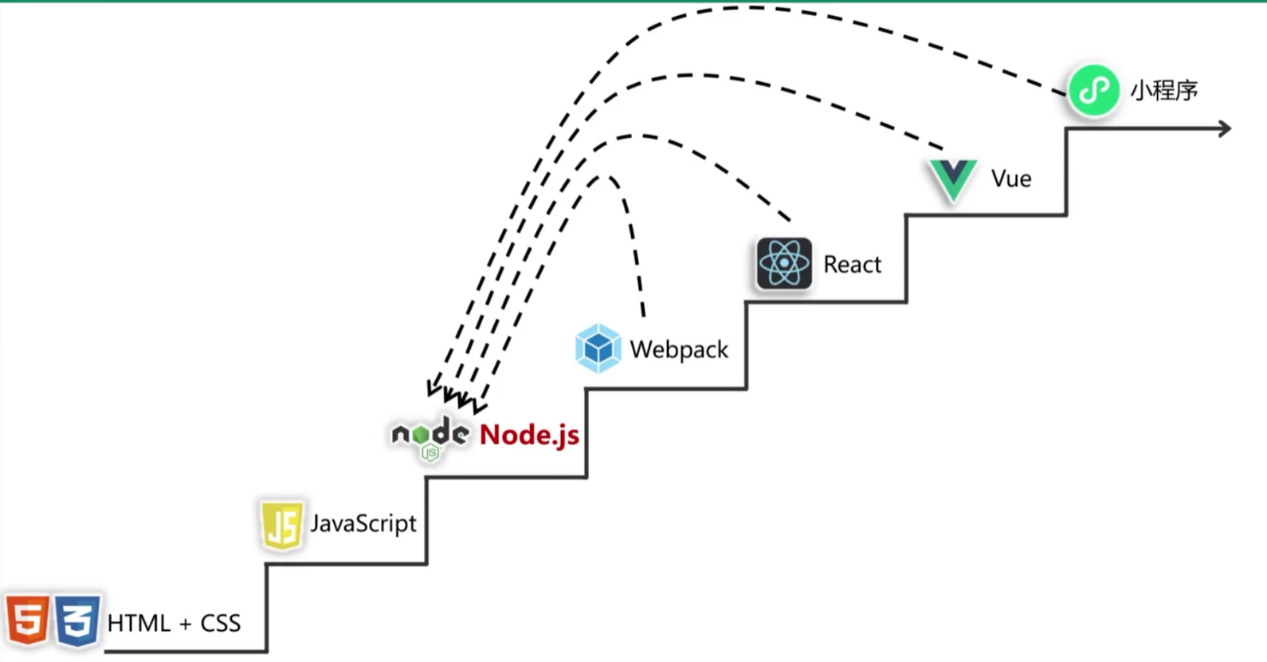
Web开发-JS应用&WebPack构建&打包Mode&映射DevTool&源码泄漏&识别还原



#知识点：  
1、安全开发-原生PHP-超级全局变量  
2、安全开发-原生PHP-代码审计案例  
---------------------------------------------  
0、安全开发-原生PHP-数据库通讯  
1、安全开发-原生PHP-身份验证技术  
2、安全开发-Cookie&Session&Token  
3、安全开发-原生PHP-代码审计案例  
---------------------------------------------  
1、安全开发-原生PHP-弱类型脆弱  
2、安全开发-原生PHP-函数&数据类型  
3、安全开发-原生PHP-代码审计案例  
---------------------------------------------  
1、安全开发-原生PHP-开发组件集合  
2、安全开发-原生PHP-模版引擎渲染  
3、安全开发-原生PHP-第三方编辑器  
---------------------------------------------  
1、安全开发-框架技术-ThinkPHP  
2、安全开发-框架安全-版本&写法  
3、安全开发-ThinkPHP-代码审计案例  
---------------------------------------------  
1、安全开发-原生PHP-文件安全操作  
2、安全开发-原生PHP-上传读取删除包含等  
3、安全开发-原生PHP-代码审计文件安全---------------------------------------------  
1、安全开发-原生PHP-RCE安全  
2、安全开发-原生PHP-代码执行&命令注入  
3、安全开发-原生PHP-代码审计案例  
---------------------------------------------  
1、安全开发-原生PHP-PHP.INI安全  
2、安全开发-原生PHP-全局文件&单函数  
3、安全开发-原生PHP-流量检测&AI算法  
---------------------------------------------  
1、安全开发-原生JS-语法模型概念  
2、安全开发-Ajax技术-JQuery&Axios  
3、安全开发-Ajax应用-文件上传&用户登录---------------------------------------------  
1、安全开发-原生JS-DOM树&BOM对象  
2、安全开发-原生JS-DOM安全&安全案例  
---------------------------------------------  
1、安全开发-原生JS-数据加密&代码混淆  
2、安全开发-原生JS-数据解密安全案例  
---------------------------------------------  
1、安全开发-NodeJS-开发环境&功能实现  
2、安全开发-NodeJS-安全漏洞&案例分析  
3、安全开发-NodeJS-特有漏洞&代码审计  
---------------------------------------------  
1、安全开发-WebPack-构建打包器  
2、安全开发-WebPack-源码泄漏还原  
  
#章节点：(待补充)  
#具体点：(待补充)





演示案例：

* WEB开发-原生JS-语法模型概念
* WEB开发-Ajax技术-JQuery&Axios
* WEB开发-Ajax前端验证-上传&登录
* WEB开发-原生JS对象-DOM&BOM
* WEB开发-原生JS对象-DOM-XSS
* WEB开发-原生JS-数据加密&代码混淆
* WEB开发-NodeJS-安装&语法&模块&安全
* WEB开发-NodeJS-原型链污染&代码审计
* WEB开发-WebPack-构建打包&源码泄漏

Webpack 是一个强大的模块打包工具，它主要用于将 JavaScript 代码和其他资源（如 CSS、图片、字体等）打包成浏览器能够高效加载的文件。下面是使用 Webpack 的一些常见原因和优势：  
1. 模块化支持  
Webpack支持模块化开发，可以将代码分割成多个文件（模块），然后将这些文件按需打包成一个或多个最终的输出文件。这对于管理复杂应用程序的代码非常重要，特别是现代JavaScript应用程序中，大多数代码和资源都已是模块化的。  
例如，你可以将前端代码分为多个模块（如组件、工具函数等），然后让 Webpack 负责打包这些模块，并管理它们之间的依赖关系。  
2. 处理各种资源  
Webpack 不仅仅是处理 JavaScript 文件，还能处理多种类型的资源：  
CSS：你可以使用 Webpack 将 CSS 文件打包到最终的输出中。  
Sass / Less：Webpack 配合相应的加载器（如 sass-loader、less-loader）可以处理 Sass 或 Less 文件。  
图片和字体：通过 file-loader 或 url-loader，Webpack 可以处理图片、字体文件等静态资源。  
HTML 文件：可以使用 html-webpack-plugin 插件自动生成 HTML 文件并插入打包后的资源。  
3. 代码分割（Code Splitting）  
Webpack 支持 代码分割，它可以将大型的 JavaScript 应用程序拆分成多个小的文件（chunks），并按需加载这些文件。这样，初次加载时，浏览器只会加载最小的必需代码，而不是所有的代码，从而提高页面加载速度。  
例如，你可以按页面或功能进行代码分割，只加载用户需要的部分，而不是一次性加载整个应用程序。  
4. 性能优化  
Webpack 提供了多种方式来优化性能：  
压缩和优化：在生产模式下，Webpack会自动压缩JavaScript和CSS文件，减小文件大小，从而加速加载速度。  
Tree Shaking：Webpack 能够识别并删除未使用的代码，这样在最终打包时只会包含必要的代码，减少最终打包文件的体积。  
缓存优化：通过配置 hash 和 chunkhash，Webpack 可以为输出的文件生成唯一的哈希值，这有助于缓存优化。这样，浏览器可以缓存大部分静态资源，而只有在资源内容更改时才会重新下载。  
5. 热模块替换（HMR）  
Webpack 提供了 热模块替换（HMR） 功能，允许开发者在不刷新浏览器页面的情况下更新应用程序的部分模块。这对于开发时快速预览修改、提高开发效率非常有用。  
6. 兼容性和 Polyfill  
Webpack 可以与 Babel 配合使用，将现代 JavaScript 转译成兼容旧浏览器的代码。通过 Babel 和 Webpack 配合，你可以确保代码在不同浏览器中的兼容性。  
7. 自动化任务  
Webpack 配置可以结合其他工具自动化一些常见的任务：  
自动生成 HTML 文件，并将打包的 JS 和 CSS 插入其中。  
自动处理样式表（Sass、Less）。  
自动优化图片文件。  
自动生成文件哈希和版本控制。  
8. 插件系统  
Webpack 提供了强大的插件机制。你可以通过使用插件（如 HtmlWebpackPlugin、CleanWebpackPlugin、MiniCssExtractPlugin 等）来扩展 Webpack 的功能，定制化自己的构建流程。  
9. 支持不同的开发环境  
Webpack 可以根据不同的环境（开发、生产）使用不同的配置。你可以使用 mode 配置来选择开发模式或生产模式，Webpack 会根据模式做出相应的优化。例如，在生产模式下，它会自动启用代码压缩和优化功能。  
10. 生态系统和社区支持  
Webpack 拥有非常活跃的社区，很多流行的前端框架（如 React、Vue）和库都提供了针对 Webpack 的最佳实践和插件，使得集成变得更加简便。  
结论：  
Webpack 是现代前端开发中不可或缺的工具，特别是对于构建复杂的、模块化的应用程序。它的优势在于模块化处理、资源管理、性能优化、代码分割和插件扩展，使得构建和部署变得更加高效和灵活。如果你的应用程序规模较大，或者涉及到多个资源文件，使用Webpack打包将带来显著的性能和开发体验提升。  
  
#WebPack技术：  
1、打包模式差异  
2、打包代码差异  
3、打包文件差异  
参考：  
https://docschina.org/  
https://www.webpackjs.com/  
https://mp.weixin.qq.com/s/J3bpy-SsCnQ1lBov1L98WA  
npm i webpack --dev  
npm i webpack-cli --dev  
例子：打包JS，NodeJS，HTML等  
简单来说，webpack具备以下功能：  
1、支持js模块化  
2、处理css兼容性  
3、将多个html/css/js文件压缩合并  
  
#源码泄漏：  
1、mode配置  
production（生产），development（开发），开发模式下会存在泄漏  
还原：浏览器webpack://  
  
2、devtool配置  
参考：https://mp.weixin.qq.com/s/tLjSb5cinXawMEC7RfJEJQ  
在使用webpack打包部署代码时，如果参数devtool配置不当，将会在部署代码文件中生成对应匹配的soucemap文件（源码映射），如果将参数devtool配置为“source-map”、“cheap-source-map”、“hidden-source-map”、“nosources-source-map”、“cheap-module-source-map”等值时，打包后将生成单独的map文件。  
  
还原项目：shuji，reverse-sourcemap  
npm install --global shuji  
shuji xxx.js.map -o xxxxxxx  
npm install --global reverse-sourcemap  
reverse-sourcemap --output-dir ./ xxx.js.map  
识别项目：https://github.com/SunHuawei/SourceDetector  
  
#案例应用  
源码泄漏真实应用

涉及资源：[资源下载地址](https://docs.qq.com/doc/DQ3Z6RkNpaUtMcEFr)