JavaScript 脚本语言

脚本语言必须有 语言解析器, 所以脚本语言写不了底层逻辑

操作系统, 启动内核, 访问寄存器 脚本语言不擅长

Java 语言面世的时候, 特别受欢迎!

网页最初时候是 没有Javascript! 需要开发一套脚本语言

开发脚本语言这个人呢 , 他就仿照Java 写了一套脚本语言

JavaScript 名字的由来

Javascript 脚本解析器, 不同的浏览器 支持不一样! 特别是微软, 什么东西都喜欢自己搞一套!

浏览器 发展了 几十年, 基本上没啥大进步!

早期写javaScript 还需要 注意 浏览器兼容性! 那个年代就有一个插件专门解决 浏览器兼容问题

Jquery 就专门解决不同浏览器兼容问题的, Jquery 内部自动判断浏览器, 处理不兼容问题!!

$.post $("div")

前端大发展年代的到来

v8 解析引擎面世为分割点, 前端行业迎来了大的跨域进步, 量变到质变的一个过程

推特公司发布了 React 前端框架, Nodejs 的面世 意味着, 前端开始工程化!

代码压缩, css转义 这些操作都需要单独工具操作

webpack 开始一站式 前端工程化操作, 开始有了脚手架概念

之后有了一系列的 前端框架面试

基于 nodejs 的后端框架也蓬勃发展

v8的推出的是 给chrome 做Javascript 解析器

也就是说 为了 推出 chrome浏览器 才做的 V8 引擎! 之前用 火狐浏览器 开发网页

IE6 IE7 IE8 最早的前端 写样式, 写HTML, 有些前端还得会切图, 幻灯片都是后端才能写的

JavaScript 是一个弱类型的语言, 一般不关注数据类型

JavaScript 基础语法

- 数据类型
- 语法声明
- 变量使用
- 正则表达式
- DOM
  - 常用的DOM操作
    - DOM 查询,
    - DOM 取值（取value值）
    - DOM 赋值（innerHTML | innerTEXT）
    - 表单验证
    - DOM 对象监听时间
    - DOM 对象事件处理

- 常用的 BOM对象
  - window
  - document
  - History
  - Screen
  - Location
  - Navigator
- 正则表达式
  - 字面量声明 正则表达式
  - new 对象 正则表达式

```
/**
 * . 匹配所有
 * * 零位或者多为
 * + 一次以上
 * [] 范围匹配
 * () 子匹配
 * ^ 以某某开头
 * $ 以某某结尾
 * ? 可有可没有 懒汉模式
 * {} 匹配次数
 */
```