运行环境

- 运行环境即浏览器（server 端有 nodejs）
- 下载网页代码，渲染出页面，期间会执行若干 JS
- 要保证代码在浏览器中：稳定且高效

运行环境

- 网页加载过程
- 性能优化
- 安全

题目

- 从输入 url 到渲染出页面的整个过程
- window.onload 和 DOMContentLoaded 的区别

知识点

- 加载资源的形式
- 加载资源的过程
- 渲染页面的过程

资源的形式

- html 代码
- 媒体文件，如图片、视频等
- javascript css

加载过程

- DNS 解析：域名 -> IP 地址
- 浏览器根据 IP 地址向服务器发起 http 请求
- 服务器处理 http 请求，并返回给浏览器

渲染过程 - 1

- 根据 HTML 代码生成 DOM Tree
- 根据 CSS 代码生成 CSSOM
- 将 DOM Tree 和 CSSOM 整合行程 Render Tree

渲染过程 - 2

- 根据 Render Tree 渲染页面
- 遇到 <script>则暂停渲染，优先加载并执行 JS 代码，完成再继续
- 直至把 Render Tree 渲染完成

思考

◆ 为何建议把 css 放在 head 中？

思考

◆ 为何建议把 js 放在 body 最后？

window.onload 和 DOMContentLoaded

```
window.addEventListener('load', function () {
    // 页面的全部资源加载完才会执行，包括图片、视频等
})
document.addEventListener('DOMContentLoaded', function () {
    // DOM 渲染完即可执行，此时图片、视频还可能没有加载完
})
```

从输入 url 到显示出页面的整个过程

◆ 下载资源：各个资源类型，下载过程

◆ 渲染页面：结合 html css javascript 图片等

window.onload 和 DOMContentLoaded 区别

- window.onload 资源全部加载完才能执行，包括图片
- DOMContentLoaded DOM 渲染完成即可，图片可能尚未下载