■输入网址按下回车后，浏览器做了什么

●第一步：解析URL

输入的URL也叫“统一资源定位符”俗称“网址”

URL只是IP地址的映射

拿到输入的URL后，浏览器会先读取本地浏览器缓存，如果没有对应的IP，就去DNS域名系统去匹配真实的IP地址，并存到本地浏览器缓存，第二次访问就可以直接读取了。

●第二步：建立连接

拿到IP地址后，建立连接 TCP三次握手

●第三步：拿到数据，渲染页面

HTML → Dom树

CSS → CSS结构体

JS → 脚本

通过以上拿到的数据，然后以下步骤

Render Tree 渲染树

计算布局信息

UI引擎渲染

用户所见页面

●第四步：断开连接

TCP四次挥手

■TCP协议UDP协议 三次握手与四次挥手

|  |
| --- |
| 应用层 |
| 传输层 |
| 网络层 |
| 数据链路层 |
| 物理层 |

TCP和UDP协议都工作在传输层，目标都是在程序之间传输数据

TCP和UDP的区别在于一个基于连接，一个基于非连接

UDP基于非连接

TCP基于连接

TCP由于是基于非连接的，因此有信息传输安全性问题，例如：

对方是否收到

内容是否完整

顺序是否正确

等

**为了保证传输的信息安全无误，TCP有三个步骤：**

**三次握手**

**传输确认**

**四次挥手**

**三次握手是建立连接的过程**

**四次挥手是关闭连接的过程**

两者区别

TCP 稳定可靠（适用于传输文件，发送邮件，浏览网页等对于丢包要求敏感的场景）

UDP 性能损耗少，资源占用少，但稳定性弱（适用于对实时性要求较高，对丢包不敏感的场景。比如域名查询，语音通话，视频直播，隧道网络等）

■前端性能优化

●页面加载性能

提高加载速度，提高用户体验

减少http请求（雪碧图/精灵图，文件的合并）

减小文件大小（资源压缩，图片压缩，代码压缩）

CDN（第三方库、大文件、大图等，通过链接来引用）

SSR服务端渲染（预渲染）

懒加载

分包（一个包太大，为了性能优化，分开，让一开始不用加载那么大）

●性能优化（动画与操作性能）

目前浏览器性能都是超过项目需求的。但也可以通过以下，来再次提高浏览器性能：

减少dom操作，避免回流，文档碎片

●内存占用

内存占用过大，浏览器崩掉等

即时通讯等场景。暂时不用太考虑

●电量消耗

游戏方面，暂时不考虑

■

跨域

●浏览器的同源策略

协议、域名、端口，3个都相同就是同源

CORS策略 （Cross Origin Resource Sharing）

注意：图片、视频文件不存在跨域。只有通过ajax请求的数据存在跨域问题

因此，同源策略不会限制<link><img><script><iframe>这些标签加载第三方同源

●解决方法

1、后端修改相应头

“Access-Control-Allow-Origin”

2、JSONP

后端开个后门，前端通过script标签来请求

3、Ngax反向代理

**◎目前vue等主流框架都使用代理来解决跨域问题**

**前端配置代理来中转请求**

**前端请求中转服务器，**

■前端安全问题

●前端安全的类型：

XSS攻击：跨站脚本攻击，主要实用javascript向网站注入代码，以实现盗号、窃取资料等目的。

csrf攻击：跨站伪造请求，冒充用户进行一些操作，甚至完成银行的转账。

sql注入：直接攻击系统的数据库，可以实现数据级别的操作，盗取数据。

文件上传漏洞：上传自定义的php代码并执行。

●XSS的防范

过滤：对提交内容中的非法标签和属性（例如<script> onclick等）进行过滤，从而避免XSS攻击。

转义：通过实用htmlspecialcharts等函数，将提交内容中的字符”<” 和”>”转换为HTML实体。

●CSRF的防范

referer防范：通过判断请求头中的referer头，确定请求的来源，从而避免CSRF攻击。

Token防范：通过在表单请求中添加token，标识表单的合法性，来避免CSRF攻击。

■session 和token和 cookie

●什么是session

Session存在服务器端。服务器为了保存用户状态而创建的一个特殊对象。

当浏览器第一次访问服务器时，服务器创建一个session对象（该对象有一个唯一的id，一般称之为session id），服务器会将session id以cookie的方式发送给浏览器。

当浏览器再次访问服务器时，会将session id发送过来，服务器依据session id 就可以找到对应的session对象。

●什么是token

Token是服务端生成的一串字符串，以作客户端进行请求的一个令牌。当第一次登录后，服务器生成一个token便将此返回给客户端，以后客户端只需带上这个token前来请求数据，即可无需再次带上用户名和密码。

●Token和cookie的区别

Cookie有储存上限，4KB

COOKIE是浏览器默认存储和发送的行为

Token是程序员自己存储和发送的行为,存到localStorage中