1、h5新特征
2、标签页通信
3、继承
4、构造函数
5、数组去重（优雅的方法）
6、position
7、页面跳转 window.open
8、cookie
9、跨域
10、函数提升
11、布局 两栏、三栏、居中等
12、ajax jquery和原生的实现
13、事件绑定 捕获、目标、冒泡
14、函数柯里化（前端编程练习50题－504行）
15、深度拷贝（JSON、遍历递归）
16、jquery的优点：选择器（sizzle），看一下jquery的结构，jquery和其他框架到区别
17、换行那些知识：
18、transform（变形）、transition（过渡）、animation（动画）、svg（矢量图）、canvas
19、阻止事件冒泡、阻止默认事件
event.preventDefault();//阻止默认事件
event.stopPropagation();//阻止冒泡
return false;//阻止冒泡和默认事件
event.stopImmediatePropagation()//组织事件冒泡和取消当前事件
20、媒体查询@media
21、哪些事件不冒泡
22、事件代理
23、前端题
24、对项目的看法
25、webpack

学习途径：博客园、掘金、简书、stackoverflow、公众号（程序员公读、程序

员大咖）、看书（高程、ES6入门）、MDN

平时自己如何学习
科研研究的是什么
弄点小demo
ES6箭头函数
正则
高性能网站
项目中的图
http具体内容

MVC、MVP和MVVM的区别：
MVC, MVP, MMVM用来解决业务逻辑和视图之间的耦合
MVVM模式中，一个ViewModel和一个View匹配，它没有MVP中的IView接口，而是完全的和View绑定，所有View中的修改变化，都会自动更新到ViewModel中，同时ViewModel的任何变化也会自动同步到View上显示。
在MVP中，V是接口IView, 解决对于界面UI的耦合; 而MVVM干脆直接使用ViewModel和UI无缝结合, ViewModel直接就能代表UI. 但是MVVM做到这点是要依赖具体的平台和技术实现的，比如WPF和knockoutjs, 这也就是为什么ViewModel不需要实现接口的原因，因为对于具体平台和技术的依赖，本质上使用MVVM模式就是不能替换UI的使用平台的.
WPF和html界面中使用Knockout，实现了observable， 所以使用MVVM.(应该说WPF就是为使用MVVM设计的)
在web应用中,由于http是基于请求和响应方式协同工作的, 无法一直保持连接状态，所以无法达到MVP中Presenter之间的消息传递和MVVM中的ViewModel和界面之间的绑定， 所以MVC是最佳的选择。
MVVM 在使用当中，通常还会利用双向绑定技术，使得 Model 变化时，ViewModel 会自动更新，而 ViewModel 变化时，View 也会自动变化。所以，MVVM 模式有些时候又被称作：model-view-binder 模式。
MVC
1、View接受用户的交互请求

2、View将请求转交给Controller

3、Controller操作Model进行数据更新

4、数据更新之后，Model通知View数据变化

5、View显示更新之后的数据

jQuery与Vue的区别：

jQuery：选择器操作DOM，数据和界面是一起的（耦合的）。

Vue：MVVM，通过Vue对象，将数据和view完全分离开来。通过双向数据绑定将View层和Model层连接起来。不会直接修改DOM结构，不会出现类似于$("ul").append("
one
")这样的操作。

react和vue的区别：

1、Vue使用模板系统而不是JSX。React与Vue最大的不同是模板的编写。Vue鼓励你去写近似常规HTML的模板。写起来很接近标准HTML元素，只是多了一些属性。

Vue鼓励你去使用HTML模板去进行渲染，使用相似于Angular风格的方法去输出动态的内容。因此，通过把原有的模板整合成新的Vue模板，Vue很容易提供旧的应用的升级。这也让新来者很容易适应它的语法。另一方面，React推荐你所有的模板通用JavaScript的语法扩展——JSX书写。

2、Vue主要是由一位开发者进行维护的，而不像React一样由如Facebook这类大公司维护。

3、React和Vue都有自己的构建工具，你可以使用它快速搭建开发环境。React可以使用Create React App (CRA)，而Vue对应的则是vue-cli。两个工具都能让你得到一个根据最佳实践设置的项目模板。由于CRA有很多选项，使用起来会稍微麻烦一点。这个工具会逼迫你使用Webpack和Babel。而vue-cli则有模板列表可选，能按需创造不同模板，使用起来更灵活一点。

4、在React中你需要使用setState()方法去更新状态。在Vue中，state对象并不是必须的，数据由data属性在Vue对象中进行管理。而在Vue中，则不需要使用如setState()之类的方法去改变它的状态，在Vue对象中，data参数就是应用中数据的保存者。对于管理大型应用中的状态这一话题而言，Vue.js的作者尤雨溪曾说过，（Vue的）解决方案适用于小型应用，但对于对于大型应用而言不太适合。

相似：

1、Reat与Vue只有框架的骨架，其他的功能如路由、状态管理等是框架分离的组件。

2、vue2.0和react都是virtual DOM。改变真实的DOM状态远比改变一个

JavaScript对象的花销要大得多。Virtual DOM是一个映射真实DOM的JavaScript对象，如果需要改变任何元素的状态，那么是先在Virtual DOM上进行改变，而不是直接改变真实的DOM。当有变化产生时，一个新的Virtual DOM对象会被创建并计算新旧Virtual DOM之间的差别。之后这些差别会应用在真实的DOM上。能够智能地计算出重新渲染组件的最小代价并应用到 DOM 操作上。

3、组件化，React与Vue都鼓励组件化应用

4、Props，在上面两个例子中，我们可以看到React和Vue都有'props'的概念，这是properties的简写。props在组件中是一个特殊的属性，允许父组件往子组件传送数据。·

浏览器攻击：

1、CSRF（Cross-Site Request Forgery）：

跨站点请求伪造，对于未被授权的系统有权访问某个资源的情况。（可以通过XHR访问的任何URL也可以通过浏览器或服务器来访问）

CSRF攻击者在用户已经登录目标网站之后，诱使用户访问一个攻击页面，利用目标网站对用户的信任，以用户身份在攻击页面对目标网站发起伪造用户操作的请求，达到攻击目的。

防御措施：验证发送者是否有权访问相应的资源。

1）要求以SSL连接来访问哭用通过XHR访问的资源；

2）要求每一次请求都要附带经过相应算法计算得到的验证码。但是出于用户体验考虑，网站不能给所有的操作都加上验证码。因此验证码只能作为一种辅助手段，不能作为主要解决方案。

3）Referer信息检查。通过检查referer信息是否合法来判断用户是否被CSRF攻击，仅仅是满足防御的充分条件，Referer Check的缺陷在于服务器并非什么时候都收到Referer，并且Referer信息可以伪造

4）Token需要足够随机，必须使用足够安全的随机数生成算法

Token可以放在用户的Session中或Cookie中，在提交请求时，服务器只需要验证表单中Token与用户Session（或Cookie）中的Token是否一致，一致则认为合法

在使用Token时尽量把Token放在表单中，使用POST提交，以避免Token泄露

如果该网站还存在XSS漏洞，那么使用Token方法防御CSRF攻击也就无效了（XSRF攻击）

token 可以在 QueryString、POST body 甚至是 Custom Header 里，但千万不

能在 Cookies 里。

问题就在你刚才访问过的网页。假设你的博客id=8, b网页内容大致如下：

```
<html>
 ...
 <img src='http://www.a.com/resource/delete/8'/>

 ...
<html>
```

网页中img src正是删除你的博客链接，或许你会说，后台不是有身份认证么？是的，后台的确有身份认证，但此时访问b，你并没有退出登录，而此时b中浏览器又发起了 http://www.a.com/resource/delete/8 请求（同时会发送该域下的cookie），这样一来，后台用户认证会通过，所以删除会成功。ps:是不是以后可以用这招去删帖了。。。

2、XSS攻击Cross Site Script：

跨站脚本攻击。XSS攻击通常指黑客通过“HTML注入”篡改了网页，插入了恶意脚本，从而在用户浏览网页时，控制用户浏览器的一种攻击。

XSS根据效果的不同可以分为如下几类：

- 反射性 XSS

发出请求时，XSS代码出现在URL中，作为输入提交到服务器端，服务器端解析后响应，XSS代码随响应内容一起传回给浏览器，最后浏览器解析执行XSS代码，这个过程像一次反射，因此叫做反射型XSS

- 存储型XSS

存储型XSS会把用户输入的数据存储到服务器，这种攻击具有很强的稳定性,也叫“持久型XSS”

- DOM Based XSS

通过修改页面的DOM节点形成的XSS

XSS之所以会发生， 是因为用户输入的数据变成了代码。 所以我们需要对用户输入的数据进行HTML Encode处理。 将其中的"中括号"， “单引号”，“引号”之类的特殊字符进行编码。

防御措施

1）后端在接收请求数据时，需要做输入检查，过滤特殊符号和标签

2）前端在显示后端数据时，需要做输出检查，不仅是标签内容需要过滤、转义，就连属性值和样式也都可能需要。 例如： 年龄的textbox中，只允许用户输入数字。 而数字之外的字符都过滤掉。

3）在处理富文本时可以设置标签白名单

4）设置HttpOnlly防止cookie劫持。将重要的cookie标记为http only， 这样的话Javascript 中的document.cookie语句就不能获取到cookie了.

5）对数据进行Html Encode 处理；过滤JavaScript 事件的标签。例如 "onclick=", "onfocus" 等等。

前端路由和服务端路由的区别

- 服务端路由：每跳转到不同的URL，都是重新访问服务端，然后服务端返回页面，页面也可以是服务端获取数据，然后和模板组合，返回HTML，也可以是直接返回模板HTML，然后由前端js再去请求数据，使用前端模板和数据进行组合，生成想要的HTML。
- 前端路由：每跳转到不同的URL都是使用前端的锚点路由，实际上只是JS根据URL来操作DOM元素，根据每个页面需要的去服务端请求数据，返回数据后和模板进行组合，当然模板有可能是请求服务端返回的，这就是 SPA 单页程序。

Web 前后端分离的意义大吗?

1、该网站前端变化远比后端变化频繁，则意义大。
2、该网站尚处于原始开发模式，数据逻辑与表现逻辑混杂不清，则意义大。
3、该网站前端团队和后端团队分属两个领导班子，技能点差异很大，则意义大。
4、该网站前端效果绚丽/跨设备兼容要求高，则意义大。