- BOM
  - window对象属性
    - 在浏览器中window 小写w, 指向当前浏览器窗口 ，也是当前页面的顶层对象，其实window, 有自己的含义，早期设计js的缺陷，现在已经无法修正
    - window.name属性，表示当前浏览器窗口的名字，浏览器窗口也可以没有名字，这个名称为字符串，如果当前浏览器窗口关闭，那么window.name 也就失效了
    - window.closed，window.opener， window.status没有实际用途，不做了解
    - window.self和window.window 都表示当前窗口本身，只读属性
    - window.frames 返回一个类似数组的对象，成员为页面内所有框架窗口，包括frame元素和iframe元素。window.frames[0]表示页面中第一个框架窗口，window.length返回当前网页包含的框架总数。如果当前网页不包含frame和iframe元素，那么window.length就返回0。window.frames.length === window.lengthwindow.frameElement属性主要用于当前窗口嵌(iframe)在另一个网页的情况,返回当前窗口所在的那个元素节点。如果当前窗口是顶层窗口，或者所嵌入的那个网页不是同源的，该属性返回null。
    - window.top属性指向最顶层窗口，主要用于在子窗口里面获取顶层的父窗口。window.parent属性指向父窗口。如果当前窗口没有父窗口，window.parent指向自身。
    - window.devicePixelRatio ,返回一个数值，表示当前css像素和设备像素的比率，也就是说一个css像素需要多少个物理像素，如果比值越大就表示用户正在使用高清屏幕，因此可以显示较大像素的图片。
    - window.screenX，window.screenY属性，返回浏览器窗口左上角相对于当前屏幕左上角的水平距离和垂直距离（单位像素）。这两个属性只读。
    - window.innerHeight，window.innerWidth属性，返回网页在当前窗口中可见部分的高度和宽度，即“视口”（viewport）的大小（单位像素）。这两个属性只读。
    - window.outerHeight，window.outerWidth属性, 返回浏览器窗口的高度和宽度，包括浏览器菜单和边框（单位像素）。这两个属性只读。
    - window.scrollX，window.scrollY, window.scrollX属性返回页面的水平滚动距离，window.scrollY属性返回页面的垂直滚动距离，单位都为像素。这两个属性只读。注意，这两个属性的返回值不是整数，而是双精度浮点数。如果页面没有滚动，它们的值就是0。
    - window.pageXOffset属性和window.pageYOffset属性，是window.scrollX和window.scrollY别名。
  - window对象的方法
    - window.alert() ,window.prompt() , window.confirm() 提示信息window.alert() 方法只有对话框，只有确定按钮，用于提示用户信息window.prompt() 方法弹出一个对话框有一个输入框，输入的值为window.prompt() 返回的值window.confirm() 方法弹出一个对话框有确定按钮和取消按钮，点击确定按钮的话，window.confirm() 返回true, 否则返回false
    - window.open(url, windowName, windowFeatures) , window.close() , window.stop() , 方法用于新建另一个浏览器窗口，类似于浏览器菜单的新建窗口选项。它会返回新窗口的引用，如果无法新建窗口，则返回null
    - window.stop()方法完全等同于单击浏览器的停止按钮，会停止加载图像、视频等正在或等待加载的对象。
    - window.moveTo() , window.moveBy() , 方法 移动浏览器窗口到指定位置window.moveTo()方法用于移动浏览器窗口到指定位置。它接受两个参数，分别是窗口左上角距离屏幕左上角的水平距离和垂直距离，单位为像素。window.moveBy方法将窗口移动到一个相对位置。它接受两个参数，分布是窗口左上角向右移动的水平距离和向下移动的垂直距离，单位为像素。为了防止有人滥用这两个方法，随意移动用户的窗口，目前只有一种情况，浏览器允许用脚本移动窗口：该窗口是用window.open方法新建的，并且它所在的 Tab 页是当前窗口里面唯一的。除此以外的情况，使用上面两个方法都是无效的。
    - window.resizeTo(outerWidth，outerHeight), window.resizeBy() 方法， 放缩窗口到指定大小
    - window.scoll(), window.scrollTo() ,window.scrollBy() 方法滚动页面到指定位置，window.scroll()方法是window.scrollTo()方法的别名。window.scrollTo(x,y) 或者window.scrollTo(options) // options = {top, left, behavior}
  - cookie
    - cookie 是服务器保存在浏览器的一段小文本信息，每个cookie 不会超过4 kb大小，浏览器每次向服务器发送请求都会带上这个信息
    - cookie 主要用来保存是否来自同一个浏览器，和保存一些状态信息，比如保存登录状态，和购物车信息，个性化，比如网站颜色主题，追踪用户行为
    - 比如，浏览器访问www.baidu.com ，百度服务器自动向浏览器写入一个cookie, 那么再次访问baidu.com时，浏览器自动带上cookie, 发给服务器，浏览器可以设置不接受cookie, 和设置不发送cookie
    - cookie 属性 expire 过期时间，Domain域名，Path路径，Secure，HttpOnly
    - document.cookie 用于读写当前网页的cookie , document.cookie = "x-auth-token=abcde"document.cookie 可以写入，必须是key=value形式，是写入不是覆盖，删除一个cookie的唯一方法是设置他的过期时间为一个已逝去的时间
  - 同源策略
    - 三同，同协议，同域名，同端口，保护用户信息安全，防止恶意篡改网站信息
    - 三个限制范围是，不能读取非同源的cookie, sessionStorage, localStorage, IndexedDB， 无法接触非同源的DOM， 无法获取非同源的ajax
    - 实现跨域资源共享方法有，window.postMessage(), 是H5提供的跨文档api cross document messaging比如，我在 http://javascript.ruanyifeng.com/bom/same-origin.html 的url 下 实现打开 baidu.comlet openBaidu = window.open('https://www.baidu.com', 'baiduName')window.addEventListener('message', function(e) { console.log('e', e)}, false)openBaidu.postMessage('tell baidu i post message to you', 'https://www.baidu.com')另一方面我们在window.open() 方法打开的baidu.com ，使用window.addEventListener('message', function(e) { console.log('e', e)}, false)监听message 信息事件window.opener.postMessage('tell to javascript.ruanyifeng.com i got it message', 'http://javascript.ruanyifeng.com') 向 javascript.ruanyifeng.com 发送信息 但这也是两个页面的通信， 这样就可以通过相互访问localStorage了
    - 实现ajax 的跨域， jsonp 和 websocket, cors
    - 子主题
      - jsonp = json padding 意思是用json 填充，是服务器与客户端跨源通信的常用方法。最大特点就是简单适用，老式浏览器全部支持，服务端改造非常小。 只要在 页面中家在一段script 脚本就行了<script type="text/javascript" src="https://www.baidu.com/?callback=getJson">function getJson(data) { console.log('data', data)} 服务器收到这个请求以后，会将数据放在回调函数的参数位置返回。getJson({ "ip": "8.8.8.8"});
      - WebSocket 是一种通信协议，使用ws://（非加密）和wss://（加密）作为协议前缀。该协议不实行同源政策，只要服务器支持，就可以通过它进行跨源通信
      - CORS Cross Origin Resource Sharing 跨域资源共享，W3C 标准，属于跨源 AJAX 请求的根本解决方法。相比 JSONP 只能发GET请求，CORS 允许任何类型的请求。所有浏览器都支持该功能， 整个跨域资源共享都是浏览器自动完成的，开发者和普通的ajax 方法一样没什么区别，浏览器一但发现这是一个跨域请求，那么就在请求头上添加一段头信息，有些可能会多出一次请求，cors 主要的还是服务器端，只要服务器支持cors，就可以实现跨域通讯
      - CORS 分为两种，简单请求，非简单请求，简单请求必须满足两个条件， 一必须是head， get, post 请求之一。二必须是请求头必须是 Accept, Accept-language, Content-language, content-type: application/x-www-form-urlencoded、multipart/form-data、text/plain 之中的任意多个， 那么对于非简单请求，流量器会采用新的方式简单请求，浏览器直接发出请求，自动在请求头中加入Origin，Origin字段用来说明，本次请求来自哪个域（协议 + 域名 + 端口）。服务器根据这个值，决定是否同意这次请求。如果，服务器发现这个Origin 不在白名单上，再返回repsonse时，响应头不会添加Access-Control-Allow-Origin字段，就知道出错了，从而抛出一个错误，被XMLHttpRequest的onerror回调函数捕获。注意，这种错误无法通过状态码识别，因为 HTTP 回应的状态码有可能是200。如果Origin指定的域名在许可范围内，服务器返回的响应，会多出几个头信息字段Access-Control-Allow-Origin: http://api.bob.comAccess-Control-Allow-Credentials: trueAccess-Control-Expose-Headers: FooBarContent-Type: text/html; charset=utf-8
      - CORS 对于非简单请求，一般是对服务器提出特殊要求的请求，比如请求方法是PUT或DELETE，或者Content-Type字段的类型是application/json。一预检请求，非简单请求都一般需要增加一次请求，用于检测，浏览器先询问服务器，当前网页所在的域名是否在服务器的许可名单之中，以及可以使用哪些 HTTP 动词和头信息字段。只有得到肯定答复，浏览器才会发出正式的XMLHttpRequest请求，否则就报错。这是为了防止这些新增的请求，对传统的没有 CORS 支持的服务器形成压力，给服务器一个提前拒绝的机会，这样可以防止服务器大量收到DELETE和PUT请求，这些传统的表单不可能跨域发出的请求。 “预检”请求用的请求方法是OPTIONS，表示这个请求是用来询问的。头信息里面，关键字段是Origin，表示请求来自哪个源。 预检请求的回应，关键的是Access-Control-Allow-Origin字段，表示http://api.bob.com可以请求数据。该字段也可以设为星号，表示同意任意跨源请求。如果服务器否定了“预检”请求，会返回一个正常的 HTTP 回应，但是没有任何 CORS 相关的头信息字段，或者明确表示请求不符合条件。 浏览器的正常请求和回应。一旦服务器通过了“预检”请求，以后每次浏览器正常的 CORS 请求，就都跟简单请求一样，会有一个Origin头信息字段。服务器的回应，也都会有一个Access-Control-Allow-Origin头信息字段。CORS 与 JSONP 的使用目的相同，但是比 JSONP 更强大。JSONP 只支持GET请求，CORS 支持所有类型的 HTTP 请求。JSONP 的优势在于支持老式浏览器，以及可以向不支持 CORS 的网站请求数据。
  - websocket
  - 全局对象属性指向一些浏览器原生的全局对象。
    - window.document ：指向document对象
    - window.screen：指向Screen对象，表示设备屏幕信息
      - screen.height/width浏览器窗口所在的屏幕的高度（宽度）（单位像素）。除非调整显示器的分辨率，否则这个值可以看作常量，不会发生变化。显示器的分辨率与浏览器设置无关，缩放网页并不会改变分辨率。
      - screen.availWidth/availHeihgt浏览器窗口可用的屏幕高度（宽度）（单位像素）。因为部分空间可能不可用，比如系统的任务栏或者 Mac 系统屏幕底部的 Dock 区，这个属性等于height减去那些被系统组件的高度
      - screen.orientation返回一个对象，表示屏幕的方向。该对象的type属性是一个字符串，表示屏幕的具体方向，landscape-primary表示横放，landscape-secondary表示颠倒的横放，portrait-primary表示竖放，portrait-secondary。
    - window.navigator：指向Navigator对象，用于获取环境信息
      - navigator.userAgent, 返回当前浏览器的厂商和版本信息
      - navigator.platform 返回用户的操作系统信息，比如MacIntel、Win32、Linux x86_64等
      - navigator.onLine 返回一个布尔值，表示浏览器是否在线
      - navigator.geolocation 返回一个geolocation属性返回一个 Geolocation 对象，包含用户地理位置的信息。注意，该 API 只有在 HTTPS 协议下可用，否则调用下面方法时会报错, Geolocation 对象提供下面三个方法.Geolocation.getCurrentPosition()：得到用户的当前位置Geolocation.watchPosition()：监听用户位置变化Geolocation.clearWatch()：取消watchPosition()方法指定的监听函数
    - window.location：指向Location对象, 用于获取当前窗口的 URL 信息
      - Location对象是浏览器提供的原生对象，提供 URL 相关的信息和操作方法。通过window.location和document.location属性，可以拿到这个对象
    - window.history：指向History对象，表示浏览器的浏览历史
      - 保存了当前窗口访问过的所有页面网址。
      - history.length 目前保存的页面网址条数
      - 由于安全的原因，浏览器不提供脚本访问这些地址，只是提供了history.go(), history.back() ,history.forward()的方法在地址之间导航history.go(0) 刷新当前页面
      - history.pushState(state, title, url) 方法用于在历史记录中添加一条记录
      - history.replaceState()方法用来修改 History 对象的当前记录，其他都与pushState()方法一模一样。
      - histroy.popstate事件， 每当同一个文档的浏览历史（即history对象）出现变化时，就会触发popstate事件。仅仅调用pushState()方法或replaceState()方法 ，并不会触发该事件，只有用户点击浏览器倒退按钮和前进按钮，或者使用 JavaScript 调用History.back()、History.forward()、History.go()方法时才会触发。另外，该事件只针对同一个文档，如果浏览历史的切换，导致加载不同的文档，该事件也不会触发。
    - window.localStorage：指向本地储存的 localStorage 数据
      - localStorage.length返回localStorage 保存数据项数
      - localStorage.setItem(key, value)两个参数都是字符串。如果不是字符串，会自动转成字符串，再存入浏览器。当然也可以通过 localStorage.key = value 或 localStorage[key] = value 方式赋值
      - localStorage.getItem(key) // value 也可以用 localStorage.key或 localStorage[key] 获取storage的值
      - localStorage.removeItem(key), 方法用于移除某项storage数据
      - localStorage.clear()清除所有保存的数据。该方法的返回值是undefined。
      - storage事件，可以监听storage事件， window.addEventLinstener('storage', onStorageChange)监听函数接受一个event实例对象作为参数。这个实例对象继承了 StorageEvent 接口，有几个特有的属性，都是只读属性
    - window.sessionStorage：指向本地储存的 sessionStorage 数据
      - sessionStorage.length, sessionStorage.getItem() ,sessionStorage.setItem() , sessionStorage.removeItem(), sessionStorage.clear() 都是和localStorage一样的
    - window.console：指向console对象，用于操作控制台
  - window事件
    - load事件，window.onload属性，load事件发生在文档在浏览器窗口加载完毕时,window.onload属性可以指定这个事件的回调函数window.onload = function () {} 在浏览器窗口加载完毕时，执行function
    - error事件和window.onerror属性，浏览器脚本发生错误时，会触发window对象的error事件。我们可以通过window.onerror属性对该事件指定回调函数
  - window对象的事件监听属性
    - window.onafterprint ：afterprint事件的监听函数。window.onbeforeprint ：beforeprint事件的监听函数。window.onbeforeunload：beforeunload事件的监听函数。window.onhashchange：hashchange事件的监听函数。window.onlanguagechange: languagechange的监听函数。window.onmessage：message事件的监听函数。window.onmessageerror：MessageError事件的监听函数。window.onoffline：offline事件的监听函数，监听浏览器离线事件。window.ononline：online事件的监听函数，监听浏览器在线事件。window.onpagehide：pagehide事件的监听函数，监听当前页面隐藏事件。window.onpageshow：pageshow事件的监听函数，监听当前页面显示事件。window.onpopstate：popstate事件的监听函数。window.onstorage：storage事件的监听函数。window.onunhandledrejection：未处理的 Promise 对象的reject事件的监听函数。window.onunload：unload事件的监听函数。window.onpopstate = function () { console.log("触发 popstate 事件")}
  - IndexedDB浏览器端数据库
  - ajax