前端开发

- HTML
  - 历史发展
  - 基本概念
  - 标签
    - 语义话标签
      - h1
      - head
      - section
      - p
    - 块元素
      - p
      - div
      - blockquote
      - form
    - 内联块元素
      - img
      - input
    - 行内元素
      - span
      - a
      - i
    - 特殊元素
      - svg
      - canvas
      - iframe
- CSS
  - 历史发展
  - 概念
    - 层级
    - 定位

- 布局
- 单位
  - px
  - rem
  - em
  - 百分比
- 属性
  - 主要属性
    - float
      - left
      - right
      - none
      - inherit
      - ...
    - display
      - none
      - block
      - inline-block
      - inline
      - flex
      - grid
      - ...
    - position
      - static
      - relative
      - absolute
      - fixed
      - sticky
    - transform
    - z-index
    - opacity
    - background
  - 其他属性

- 盒模型
- 选择器
- 动画
- 媒体查询
- 预处理框架
  - sass
  - less
  - stylus
- 常用布局（https://juejin.im/post/5bbcd7ff5188255c80668028）
  - 圣杯
  - 双飞燕
  - ...
- 性能优化
  - 导致重绘
- 业务使用
  - css文件的缓存处理
    - 浏览器的css缓存特别严重，通过手动修改html的css文件引入链接名，或者通过webpack等
  - 同时兼容pc和mobile
    - 利用rem和媒体查询，设置页面内容最大宽度，pc端显示这个最大宽度，用rem来适应ui图在mobile和pc端的显示
  - 比较常规的时候，移动端页面宽度常用rem，高度考虑比较多，常用rem/百分比/vh
- 其他

- DOM
  - 概念
    - 定义 1
  - 数据类型
    - document
    - element
    - nodeList
    - attribute
    - namedNodeMap
  - node

- 节点类型常量
  - ELEMENT_NODE
    - 介绍
      - Element提供了对元素标签名，子节点和特性的访问，我们常用HTML元素比如div，span，a等标签就是element中的一种；
    - 特性
      - nodeType为1
      - nodeName为元素标签名，tagName也是返回标签名
      - nodeValue为null
      - parentNode可能是Document或Element
      - 子节点可能是Element，Text，Comment，Processing_Instruction，CDATASection或EntityReference
  - TEXT_NODE
    - 介绍
      - Text表示文本节点，它包含的是纯文本内容，不能包含html代码，但可以包含转义后的html代码。
    - 特性
      - nodeType为3
      - nodeName为#text
      - nodeValue为文本内容
      - parentNode是一个Element
      - 没有子节点
  - PROCESSING_INSTRUCTION_NODE
    - 介绍
      - 一个用于XML文档的 ProcessingInstruction ，例如 <?xml-stylesheet ... ?> 声明
  - COMMENT_NODE
    - 介绍
      - Comment表示HTML文档中的注释
    - 特性
      - nodeType为8
      - nodeName为#comment
      - nodeValue为注释的内容
      - parentNode可能是Document或Element

        - 没有子节点
    - DOCUMENT_NODE
        - 介绍
            - Document表示文档，在浏览器中，document对象是HTMLDocument的一个实例，表示整个页面，它同时也是window对象的一个属性。
        - 特性
            - nodeType为9
            - nodeName为#document
            - nodeValue为null
            - parentNode为null
            - 子节点可能是一个DocumentType或Element
    - DOCUMENT_TYPE_NODE
    - DOCUMENT_FRAGMENT_NODE
        - 介绍
            - DocumentFragment是所有节点中唯一一个没有对应标记的类型，它表示一种轻量级的文档，可能当作一个临时的仓库用来保存可能会添加到文档中的节点
        - 特性
            - nodeType为11
            - nodeName为#document-fragment
            - nodeValue为null
            - parentNode为null
    - ...
- 操作DOM
    - 节点创建型API
        - 方法
            - createElement
            - createTextNode
            - cloneNode
            - createDocumentFragment
        - 总结
            - 它们创建的节点只是一个孤立的节点，要通过appendChild添加到文档中
            - cloneNode要注意如果被复制的节点是否包含子节点以及事件绑定等问题
            - 使用createDocumentFragment来解决添加大量节点时的性能问题

- 页面修改型API
  - 方法
    - appendChild
    - insertBefore
    - removeChild
    - replaceChild
  - 总结
    - 不管是新增还是替换节点，如果新增或替换的节点是原本存在页面上的，则其原来位置的节点将被移除，也就是说同一个节点不能存在于页面的多个位置
    - 节点本身绑定的事件会不会消失，会一直保留着
- 节点查询型API
  - 方法
    - document.getElementById
    - document.getElementsByTagName
    - document.getElementsByName
    - document.getElementsByClassName
    - document.querySelector
    - document.querySelectorAll
  - 总结
- 节点关系型API
  - 父关系型API
    - parentNode
    - parentElement
  - 子关系型API
    - childNodes
    - children
    - firstChild
    - lastChild
    - hasChildNodes
  - 兄弟关系型API
    - previousSibling
    - previousElementSibling
    - nextSibling

        - nextElementSibling
    - 元素属性型API
        - setAttribute
        - getAttribute
        - removeAttribute
    - 元素样式型API
        - window.getComputedStyle
        - getBoundingClientRect
        - 直接修改元素的样式
        - 动态添加样式规则
- 参考
    - https://juejin.im/post/5af43bd5f265da0b8336c6f7
- BOM
    - 概念
        - 定义
            - BOM浏览器对象模型（Browser Object Model）BOM对象是在Web中使用JavaScript的核心，该对象提供了与浏览器交互相关对象结构。BOM由多个子对象组成，其核心为window对象，它是BOM的顶层对象，表示在浏览器环境中的一个全局的顶级对象，所有在浏览器环境中使用的对象都是window对象的子对象。
        - ...
    - 历史发展
    - BOM中的对象
        - 概念
            - 在BOM对象中，window对象是最顶层对象，在浏览器环境中它是一个Global全局对象，其它对象（如：DOM对象）对是这个对象的属性（子对象）。BOM对象是与内容无关，主要用于管理浏览器窗口及窗口之间的通讯。
        - 对象
            - window对象
                - 描述
                    - window对象对象表示一个浏览器窗口或一个frame框架，它处于对象层次的最顶端，它提供了处理浏览器窗口的方法和属性。
                    - window对象是浏览器对象中的默认对象，所以可以隐式地引用window对象的属性和方法。在浏览器环境中，添加到window对象中的方法、属性等，其作用域都是全局的。JavaScript中的标准内置对象，在浏览器环境中也是做为window的方法和属性出现的。
                - 概念

- 属性
- 方法

- window对象子属性
  - DOM（document）相关对象
    - 描述
      - DOM可以认为是BOM的一个子集，DOM中文档操作相关对象，如：Node、Document、Element等DOM节点类型对象，都是做为window对象的子属性出现的。
      - document是window对象的子属性，它是一个Document对象实例，表示当前窗口中文档对象。通过该对象，可以对文档和文档中元素、节点等进行操作
    - 属性
    - 方法
  - frames对象
    - 描述
      - frames对象是一个集合，表示当前页面中使用的子框架。如果页面中使用了框架，将产生一个框架集合frames，在集合中可以用数字下标（从0开始）或名字索引框架。集全中的每一个对象，包含了框架的页面布局信息，以及每一个框架所对应的window对象。
    - 属性
    - 方法
  - navigator对象
    - 描述
      - navigator是指浏览器对象，该对象提供了当前正在使用的浏览器的信息。navigator对象中的属性是只读的，在W3C在HTML5标准中，对该对象进行了规范。由于浏览器的同，该对象的具体值可能有所区别。
    - 属性
    - 方法
  - history对象
    - 描述
      - history对象来保存浏览器历史记录信息，也就是用户访问的页面。浏览器的前进与后退功能本质上就是history的操作。history对象记录了用户浏览过的页面，通过该对象提供的API可以实现与浏览器前进/后退类似的导航功能。
    - 属性
    - 方法

- location对象
  - 描述
    - location是一个静态对象，该对象是对当前窗口URL地址的解析。该对象提供了可以访问URL中不同部分的信息属性，通过location对象也可以实现页面或锚点跳转等功能。
  - 属性
  - 方法
- screen对象
  - 描述
    - screen对象中包含了用户显示器屏幕相关信息。通过该对象，可以访问用户显示器屏幕宽、高、色深等信息。
  - 属性
  - 方法

- 参考
  - https://itbilu.com/javascript/js/4k9JcnZRl.html

- javascript
  - 历史发展
  - 概念
    - 细节概念
      - 堆
      - 栈
      - 作用域
      - 作用域链
      - 闭包
    - 整体概念
      - 根据...分类
        - 解释型语言
        - 编译型语言
        - 弱语言
      - 根据...分类
        - 基于对象
        - 面向对象
  - 语法
    - 概念

- 执行原理
  - ...
  - Javascript 源码从左往右被扫描并转换成一系列由 token 、控制字符、行终止符、注释和空白字符组成的输入元素。空白字符指的是空格、制表符和换行符等。
  - ...
- 词法
- 分类
  - 原生javascript
  - ES2015
  - typescript
- 声明（3）
  - 分类
    - var
      - 声明一个变量，可选初始化一个值
    - let
      - ES2015添加，声明一个块作用域的局部变量，可选初始化一个值
    - const
      - ES2015添加，声明一个块作用域的只读常量
  - 注意事项
- 数据类型
  - 概念
  - 分类（8）
    - 基础数据类型（7）
      - undefined
      - null
      - number
      - string
      - booleam
      - symbol
        - 概念
          - 定义
            - es6新增

- 方法
  - Symbol()
    - 用来创建 symbol 数据类型实例
- BigInt
  - 概念
    - 在JavaScript中，BigInt是一种数字类型的数据，它可以表示任意精度格式的整数。而在其他编程语言中，可以存在不同的数字类型，例如:整数、浮点数、双精度数或大斐波数。
- 复杂数据类型（1）
  - Object
    - 概念
      - 内存地址
    - 分类（4）
      - Array
      - Function
      - RegExp
      - Date
- 数据类型之间的关系
- 方法
  - 判断类型
    - typeof
    - instanceof
    - constructor
    - Object.prototype.toString
- 字面量
- 设计模式
  - ...
- 其他
  - 缓存
  - 缓冲
- 参考
  - https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Grammar_and_types

- 正则

- 概念
  - 定义
    - 正则表达式是用于匹配字符串中字符组合的模式。在 JavaScript中，正则表达式也是对象。这些模式被用于 RegExp 的 exec 和 test 方法, 以及 String 的 match、matchAll、replace、search 和 split 方法。
- 正则表达式模式
  - 简单模式
    - 简单模式是由你想直接找到的字符构成。比如，/abc/ 这个模式就能且仅能匹配 "abc" 字符按照顺序同时出现的情况。
  - 使用特殊字符
    - ...
- 使用正则表达式
  - 方法
    - exec
      - 一个在字符串中执行查找匹配的RegExp方法，它返回一个数组（未匹配到则返回 null）。
    - test
      - 一个在字符串中测试是否匹配的RegExp方法，它返回 true 或 false。
    - match
      - 一个在字符串中执行查找匹配的String方法，它返回一个数组，在未匹配到时会返回 null。
    - matchAll
      - 一个在字符串中执行查找所有匹配的String方法，它返回一个迭代器（iterator）。
    - search
      - 一个在字符串中测试匹配的String方法，它返回匹配到的位置索引，或者在失败时返回-1。
    - replace
      - 一个在字符串中执行查找匹配的String方法，并且使用替换字符串替换掉匹配到的子字符串。
    - split
      - 一个使用正则表达式或者一个固定字符串分隔一个字符串，并将分隔后的子字符串存储到数组中的 String 方法。
- 通过标志进行高级搜索
  - 接受

- 正则表达式有六个可选参数 (flags) 允许全局和不分大小写搜索等。这些参数既可以单独使用也能以任意顺序一起使用, 并且被包含在正则表达式实例中。
- 标志
  - g
    - 全局搜索。
  - i
    - 不区分大小写搜索。
  - m
    - 多行搜索。
  - s
    - 允许 . 匹配换行符。
  - u
    - 使用unicode码的模式进行匹配。
  - y
    - 执行“粘性(sticky)”搜索,匹配从目标字符串的当前位置开始。
- 注意事项
- 其他
- 使用技巧
  - ...
- 其他

- ui设计
  - 部分概念
  - 简单的设计理念
  - 重要的用户交互思想
  - ui工具使用
    - photoshop
    - 蓝狐
    - 图片压缩
      - 在手机端上展示的图片一般从ui那拿来原图后，还需要压缩下
        - 子主题 1
    - 雪碧图
  - 其他
    - 图片转换为字体

- seo
  - 历史
  - 概念
  - 利于seo的方法
    - ...
- 网络知识
  - 历史
  - 概念
  - 状态码
- 前端安全
  - 历史
  - 概念
  - 安全涉及方面
    - iframe
      - 描述
        - 本站页面不被其他网页当作iframe嵌入
      - 处理方法
    - opener
    - XSS攻击
    - CSRF攻击
    - ClickJacking
    - HSTS
    - CND劫持
- 前端工程化
  - 历史
  - 概念
    - 要解决的问题
  - 版本控制
    - git
    - svn
  - 自动化工程工具
    - webpack
    - gulp

- 项目规范
  - js规范
    - 命名规范
      - 变量声明规范
      - 常量声明规范
      - 函数声明规范
    - js使用规范
      - 全局变量使用规范
      - 定时器函数使用规范
      - 函数绑定使用规范
  - css规范
    - css框架选用
      - sass
      - less
      - stylus
    - css统一编写规范
      - 命名规范
  - 代码格式化规范
  - 文件结构规范
    - 目录结构规范
    - 文件命名规范
  - 规范检查工具
    - jslint
    - eslint
  - 参考
    - [网易小组规范](http://res.nie.netease.com/comm/doc/professional/javascript%E4%BB%A3%E7%A0%81%E8%A7%84%E8%8C%83.html)
- 代码框架的选用
  - 整体框架
    - 原生
      - 仅用原生javascript、jquery、es6等方式实现功能
    - mvvm框架

- ui框架
  - elementui
  - ant design
  - bootstrap
  - layui
  - mui
- js语言的选择
  - 原生javascript
  - es6
  - typescript
- 基础工具库的构建
  - 弹框
  - 统一请求拦截器
  - log功能封装
  - 仿造jq实现的元素获取
    - id元素选取
    - class元素选取
    - name元素选取
    - tag元素选取
  - 判断元素是否存在后，对简单的元素操作进行封装
    - 创造
    - 删除
    - 出现
    - 消失
    - ...
  - js运行出错，错误采集
    - onerror
    - try catch
  - 图片查看器（放大/缩小/下载）
  - 树状图
  - 轮播图
  - ...
- 单元测试

- 模拟请求
  - mockjs
- MVVM
  - 历史
  - 概念
    - SPA
  - MVVM框架
    - vue
      - 核心
        - vue2.0
        - vue cli
        - vue Router
        - vuex
        - vue 服务端渲染
      - 框架
        - element ui
        - ant-design-vue
    - react
    - angluar
- 工具使用
  - 抓包
    - fiddle
    - wireshark
    - spy-debugger
  - 代码调试
    - pc
      - chrome devtools
        - Source
          - 断点
            - Breakpoints
            - Dom Breakpoints
            - XHR/fetch Breakpoints
          - snippets

- overrides
- Network
  - 资源缓存
    - disk cache
    - memork cache
    - http 请求
  - Disable caceh
  - replay xhr
  - 设置网络速度
- console
  - 命令的使用

- mobile
  - spy-debugger
  - safari->mac
  - eruda

- 模拟请求
  - postman

- 性能优化
  - 优化工具
    - chrome devtools
      - Performance
- 跨平台
  - flutter
  - react native
  - electron
  - nw.js
  - 小程序
    - 微信小程序
    - 支付宝小程序
  - 微信小游戏
- 项目能力
  - 负责
    - 对这个项目负责

- 沟通
  - 多和领导沟通
  - 对需求提出方问题响应及时
  - 前期工作多花时间在需求明确上
- 文档
  - 对需求以文档的方式明确，以邮件等方式让领导、开发、需求提出方对于项目需求达成一致
  - 重要或者复杂的技术实现，最好要有设计文档
  - 后端最好有规范的接口文档，Web端与其他客户端交接时也是
  - 对于自己通过excel管理的工作，建议有类似以下栏目：需求、详细介绍、需求提出时间、需求提出方、前后端归属、开发人员、预计开发时间段、联调时间、实际完成时间、工作进度、测试人员、验收是否完成
  - 对于自己的工作，可以适当写些功能实现、代码结构、方案设计、项目架构的文档，在以后述职、review代码、工作交接的时候都用得着
- 需求
  - 讨论业务需求的时候，对于不合理的需求，或者开发工作量大，难度高的需求，而又有其他业务上能简单替换的方案时，及时提出
  - 接需求的时候，明确需求，尽量不说死自己的完成时间，给自己时间去排期
  - 需求排期之前，要对需求和完成需求所需要的实现方式有一个整体的认知，最好是对设想多完成这个需求，需要哪些步骤，可能会遇到哪些困难一条条列出来，并对每一条列出预期完成时间，这样得到需求的预期完成时间，设想完成步骤的时候，要多多设想，对于时间要给自己宽泛点
  - 需求多时，对需求列出优先级，然后按照优先级对优先级的先后进行排期
- JS引擎
- 浏览器引擎
- nodejs
- 计算机原理
  - 概念
  - 历史发展
  - 计算机组成
    - 输入设备
    - 输出设备
    - cpu
      - 寄存器
    - ...

- 具体领域
  - 算法
  - 可视化
  - 富媒体
    - 音视频开发
  - 实时通讯