开课吧 《JavaScript 高级工程师》 - 课程手册

第一阶段 前端高级工程师的重要基石

第一章 走入 JavaScript 的世界

- JavaScript简介
  - 计算机语言划分
    - 汇编语言
    - 脚本语言
    - 机器语言
    - 高级语言
  - JavaScript 介绍
- JavaScript可以做什么？
- Web中哪些时候需要用到
  - 行为交互
  - 数据交互
  - 逻辑交互
- JavaScript组成
  - ECMAScript：JavaScript语法和基本对象
  - DOM：文档对象模型
  - BOM：浏览器对象模型
- JavaScript 注释写法
  - 单行注释
  - 多行注释
- JavaScript 应该放在什么位置
  - 行间 JS `<div onclick="alert("我爱JavaScrip! ")"></div>`
  - 内部 JS `<script> // 这里放置js的代码 </script>`
  - 外部 JS `<script src="main.js"><script>`
- 获取元素
  - document.getElementById(' ID ')
  - document / parent.querySelector('选择器')
- 调试
  - alert()
  - console.log()
- 给元素绑定事件
  - onclick 点击事件
- 操作元素的样式
- 什么是变量
  - 变量声明
  - 变量赋值
- 标识符命名规则
  - 关键字和保留字
  - 大驼峰和小驼峰
- 多变量同时声明

- function 函数
    - 什么是函数
    - 函数声明及调用
    - 匿名函数

第二章 玩转属性操作

- 什么是属性
- JS 属性操作
    - 属性读操作
    - 属性写操作
- JS 中两种操作属性的方法
    - . 点运算符
    - [] 方括号运算符
- 常用 JS 属性
    - id
    - className
    - value
    - style
        - background
        - color
        - width
        - height
        - ……
    - cssText
    - innerHTML
    - href
    - src
    - tagName
- 属性操作时的注意事项及常见问题
    - href 值和 src 值获取到的是绝对路径
    - style 是行间属性
    - cssText 会替换掉当前所有的行间属性
    - class 是保留字，改成 className
    - tagName 获取到的是大写字母
- 案例：揉捏div
- 案例：静态留言板实现

第三章 if 语句和布尔值

- if 语句语法
    - if(){}
    - if(){} else {}
    - if(){} else if() {}
    - if(){} else if() {} else {}
- 判断条件 之 布尔值
    - true - 真

    - false - 假
- 比较运算符
    - 大于 >、小于 <、等于 ==、大于等于 >= 、小于等于 <=
- 逻辑运算符
    - && 与、|| 或、！非
    - 利用逻辑运算符书写多条件判断
- 案例：自定义下拉菜单
- 数组
    - length
    - 数组的下标
- 案例：京东幻灯片初实现

第四章 for 循环和this指向

- 什么情况下需要使用循环
- 组成循环的四大条件
    - 初始值
    - 循环结束条件
    - 循环执行语句
    - 自增和自减
- for 循环语法
    - for 循环的执行顺序
    - 循环变量 数值说明(循环结束之后，循环变量是多少)
    - 死循环是怎么造成的
- 案例：循环生成拼图
- for 套 for 的复杂应用 - 九九乘法表生成
- 函数的 this 指向
    - 循环套事件，事件中的循环变量调用问题
    - 事件函数的 this 指向
    - 非事件函数的 this 指向
- classList
    - add()
    - remove()
    - contains()
    - toggle()
- 案例：自定义单选控件
- 案例：自定义复选控件
- 索引值使用
- 案例：腾讯选项卡实现
- 案例：完整版新浪下拉菜单控件

第五章 ECMAScript数据类型及类型转换

- ECMAScript 数据类型划分
    - 简单类型：number、string、boolean、null、undefined、symbol
    - 复杂类型：object
- 传值和传址

    - JS 常见的对象类型：Array、Object、Element、Elements、Function
- typeof 运算符
    - number、string、boolean、function、undefined、symbol、object
- 强制数据类型转换
    - 强制转换为数字
        - parseInt()、parseFloat()
        - Number() 方法
            - 字符串转 number 规则
            - 布尔值转 number 规则
            - 对象转 number 规则
            - null 和 undefined 转换成 number
        - isNaN 和 NaN
        - MAX_VALUE、MIN_VALUE、infinity
    - 强制转换为布尔值 - Boolean()
        - 数字转换布尔值规则
        - 字符串转换布尔值规则
        - 对象转换布尔值规则
        - null 和 undefined 转换布尔值
    - 强制转换为字符串 - String()
        - 数字、布尔值转换字符串规则
        - null 和 undefined转换字符串规则
        - 对象转换字符串规则
            - 数组的 toString
            - function 的 toString
- 运算符及隐式类型转换
    - 数学运算符的运算规则及相应的隐式类型转换规则
    - 赋值运算符的运算规则及相应的隐式类型转换规则
    - 比较运算符符的运算规则及相应的隐式类型转换规则
    - 逻辑运算符的运算规则及相应的隐式类型转换规则
    - 三元运算符的运算规则
- 案例：数据筛选
- 案例：QQ号码验证

第六章 ECMAScript 流程控制

- 程序中的三种基本流程结构
    - 顺序结构
    - 分支结构
    - 循环结构
- 分支结构相关语句
    - if 语句
    - switch 语句
        - switch 语法
        - case 和 break
        - break 穿透
        - default
- 循环结构相关语句

    - for 循环
    - while 循环
        - while 语句语法
        - while 语句循环流程
    - do while 循环
        - do while 语句语法
        - do while 语句循环流程
    - continue 和 break 跳出及终止循环

第七章 Function 详解

- 函数声明和调用
    - function 声明
    - 函数表达式
    - 函数的事件调用 和 非事件调用
- 函数传参
    - 形参 和 实参
    - 案例：封装选项卡函数
- arguments 不定参
    - arguments 的特性：length 和 下标
    - 案例：参数求和
- return 返回值
    - 函数返回值定义
    - return 使用的注意事项
    - 案例：封装 Byld 方法
- 获取计算后样式
    - 计算后样式和 style 的区别
    - getComputedStyle() 方法
    - 封装 getStyle() 获取计算后样式的兼容方法
- JavaScript 预解析
    - var 的预解析规则
    - function 的预解析规则
- 作用域
    - 全局作用域
    - 函数作用域
- 作用域链
    - 作用域中数据的查找关系
    - 案例：修改文本框的值
- 闭包
    - 闭包的概念
    - 匿名函数自执行
    - 案例：获取当前操作元素的下标
    - 案例：商品累计
- this 指向
    - call 和 apply

第八章 定时器和日期对象

- setTimeout 和 clearTimeout
    - setTimeout 的语法说明
    - clearTimeout 清除定时器
    - 案例：QQ 面板延时菜单
    - 案例：站长之家导航
- setInterval 和 clearInterval
    - 案例：控制 div 移动
    - 定时器管理的两种模式
    - 案例：淘宝自动播放的选项卡
- 案例：自动播放的选项卡嵌套
- 日期对象
    - 获取系统时间
        - Date()、getDate()、getDay()、getMonth()、getFullYear()、getHours()、getMinutes()、getSeconds()、getMilliseconds()
        - getTime()、Data.now()
        - 案例：电子时钟
    - 设置日期对象
        - Date()、setDate()、setMonth()、setFullYear()、setHours()、setMinutes()、setSeconds()、setTime()
        - 案例：倒计时

第九章 数组和字符串方法

- 字符串方法
    - 查找类方法: charAt()、indexOf()、lastIndexOf()
    - 截取类方法：slice()、substr()、substring()
    - 其他常用方法：split()、toLowerCase()、toUpperCase()、concat()、charCodeAt()、fromCharCode()、toFixed()
    - 案例：文字搬运工
- 数组方法
    - 数组的length
    - 添加和删除：push()、pop()、unshift()、shift()
    - 案例：伪3D图片切换
    - 数组元素替换：splice()
        - 替换数组元素
        - 数组指定位置删除
        - 数组指定位置添加reverse()
    - 数组排序：
        - sort() 字典序排序
        - 队列排序
        - 自定义规则排序
        - 随机排序 - 随机数
        - 案例： 随机打乱图片
    - 其他常用方法：concat()、join()、reverse()
    - ECMAScript 5.1 数组新增方法
        - forEach()
        - filter()

        - map()
        - reduce()
        - every()
        - some()
- 案例：查找替换文字
- 案例：员工信息排序和筛选
- 案例：
- 对象
    - 对象的语法
    - keys()和values()
    - for in 循环
    - 删除对象中的属性
- json
    - json 是什么
    - parse()、stringify()
- 常用Math方法
    - 取整方法：ceil()、floor()、round()
    - 随机数：Math.random()
    - 最大值最小值：max()、min()
    - 其他：Math.PI、Math.abs()

第十章 排序算法

- 冒泡算法
- 快速排序
- 递归及递归执行顺序
- 数组去重的N种方法
    - 利用对象进行去重
    - filter 去重
    - 循环去重
    - Set 去重
- 案例：无限级菜单生成和交互

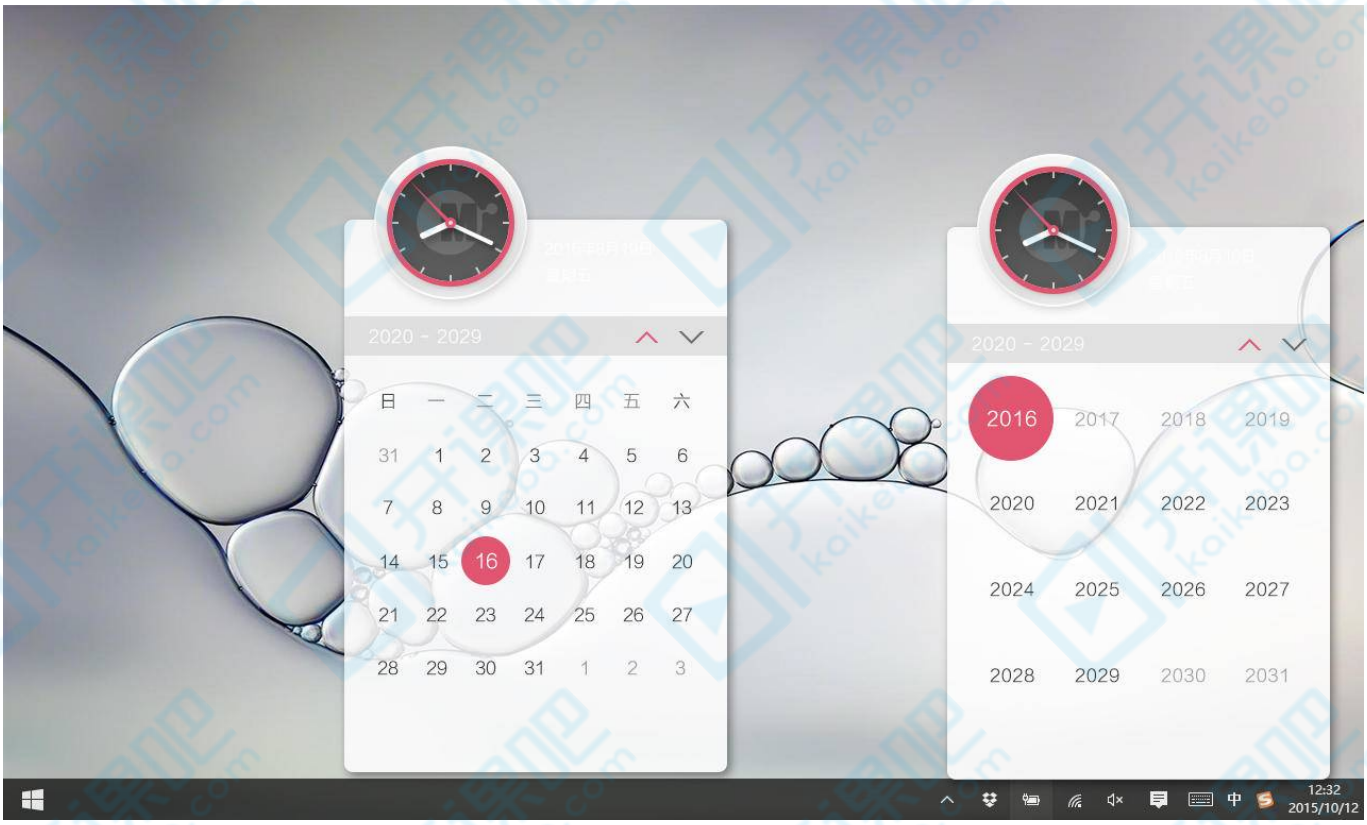
第十一章 动画专题训练

- 利用CSS3 transition 实现动画
    - transition 语法详解
    - transition-delay
    - transition-duration
    - transition-property
    - transition-timing-function
        - 贝塞尔曲线运动
    - transition 针对为渲染元素的问题
    - transitionend 事件
- 案例：京东渐隐渐现幻灯片实现
- 利用CSS3 animation 动画
    - keyframes

    - animation-name
    - animation-duration
    - animation-timing-function,
    - animation-delay
    - animation-iteration-count
    - animation-direction
    - animation-fill-mode
    - animation 相关事件
        - animationstart
        - animationiteration
        - animationend
    - animation 在和 JS 搭配时需要注意的问题
- 案例：水滴按钮
- JS 动画帧
    - requestAnimationFarme
        - 动画帧语法
        - 动画帧和定时器的区别
        - 利用动画帧进行动画
    - cancelAnimationFrame
- 动画帧兼容处理
    - 各浏览器前缀说明
    - 利用定时器兼容低版本浏览器
- Tween 动画公式
    - Tween 参数解析
        - t: current time（当前时间）
        - b: beginning value（初始值）
        - c: change in value（变化量）
        - d: duration（持续时间）
    - Tween 使用详解
- 封装基于 Tween 的动画框架
    - 封装处理样式的 css 函数
    - 封装多样式同时动画
    - 动画管理
    - 参数默认值处理
    - 添加动画播放完的回调函数
    - 动画时间优化
- 回调函数初使用
- 案例：无缝滚动幻灯片
- 案例：韩雪冬官网首页图片切换效果
- 案例：妙味导航实现
- 封装抖动函数
    - 抖动原理解析
    - 封装抖函数
    - 抖动管理
- 案例：倒计时商品下架
- CSS3 transform 详解

    - transform 2D
        - 旋转: rotate()
        - 缩放: scale()、scaleX()、scaleY()
        - 倾斜: skew()、skewX()、skewY()
        - 平移：translate()、translateX()、translateY()
        - transform 多函数书写时的执行顺序
        - transform-origin 源点设置
        - translate 和 源点关系
    - transform 3D
        - 3D旋转: rotateX()、rotateY()、rotateZ()
        - 3D位移：translateZ()
        - transform-style
        - perspective
        - perspective-origin
        - 搭建立方体
- 在动画框架中封装 transform 方法
    - JS 获取 transform 时的问题
    - 利用对象存储 transform 设置
    - 封装 transform 获取及设置方法
    - 利用动画框架对transform进行动画
- 案例：时钟
- 案例：3D 图片轮播效果

第十二章 第一阶段实战《日历》



日历，我想大家经常会用到，那么它是如何实现的呢？是不是很难？做了才知道！

- 一个日历，最基础的就是要显示当前（指定）月份的日期，有多少天，每天对应的是星期几

- 可以方便的查看上个月、下个月、还选择指定的年月
- 除了日历本身之外这个设计中，还有一个钟表的效果也是需要我们实现的

第二阶段 JavaScript 核心 API 探秘

第十三章 ECMAScript 6 基础

- let 和 const
- let 和 var 的差异
- const 常量
- 块级作用域
- 解构赋值
    - 对象的解构赋值
    - 数组的解构赋值
    - 字符串的解构赋值
- 展开运算符
    - 对象展开
    - 数组展开
- Map 对象
    - Map 对象的数据结构
    - Map 相关属性与方法
        - size 属性
        - clear()、delete()、get()、has()、set()
- Set 对象
    - Set 对象的数据结构
    - Set 相关属性与方法
        - size 属性
        - clear()、delete()、get()、has()、add()
- 函数新增扩展
    - 箭头函数
        - 箭头函数的各种写法
        - 箭头函数的 this 问题
        - 箭头函数的不定参问题
    - rest 参数设置
    - 参数默认值设置
- 新增数组扩展
    - Array.form()、Array.of()
    - find()、findIndex()、includes()
    - flat()、flatMap()
- 新增字符串扩展
    - includes(), startsWith(), endsWith()
    - repeat()
    - 模版字符串
- 新增对象扩展
    - 属性简洁表示法
    - 属性名表达式
    - 方法简写

- 案例：员工信息列表操作

第十四章 异步处理专题

- ES6 Promise 对象
  - 同步和异步
  - 异步操作的回调地狱
  - resolve 和 reject
  - then 方法 和 catch
  - finally
  - all
- Async 函数 和 await
- Iterator（遍历器）和 for of 循环
- Generator
  - yield 表达式
  - next()
  - throw() 错误捕获
- 案例：基于 Promise 封装的链式动画 - 简版
- 案例：基于 Promise 封装的链式动画 - mTween完善

第十五章 DOM

- 什么是DOM
  - 文档操作方法
  - DOM 标准
- DOM 节点类型
  - nodeType 属性
    - 元素 1
    - 属性 2
    - 文本 3
    - 注释 8
    - 文档 9
  - nodeName
    - nodeName 是只读的
    - 元素节点的 nodeName 与标签名相同
    - 属性节点的 nodeName 与属性名相同
    - 文本节点的 nodeName 始终是 #text
    - 文档节点的 nodeName 始终是 #document
- DOM 树及DOM关系
  - DOM 树
  - DOM 关系
    - parentNode、offsetParent
    - childNodes、children、firstChild、firstElementChild、lastChild、lastElementChild
    - nextSibling、nextElementSibling、previousSibling、previousElementSibling
- DOM 中的属性操作
  - getAttribute()
  - setAttribute()

    - hasAttribute()
- DOM 节点操作
    - createElement()
    - appendChild()
    - insertBefore()
    - cloneNode()
    - replaceNode()
    - hasChildNodes()
    - createdocumentfragment()
    - removeChild()
    - remove()
- DOM 元素尺寸及位置获取
    - offsetLeft、offsetTop、offsetWidth、offsetHeight
    - clientLeft、clientTop、clientWidth、clientHeight
    - scrollLeft、scrollTop、scrollWidth、scrollHeight
    - getBoundingClientRect()
        - top、left、right、bottom、width、height
- 案例：学生管理系统
- 案例：动态文件夹创建
- 案例：上移下移

第十六章 BOM

- window
    - innerWidth/innerHeight
    - close()
    - open()
    - scroll 事件
    - resize 事件
    - 操作滚动条位置
        - window.scrollX、window.scrollY、window.scrollTo()
        - document.documentElment.scrollTop、document.documentElment.scrollLeft
- location 对象相关操作
    - href
    - hash
        - hashChange
        - 案例：简易 hash 视图切换
    - search
    - reload()
- history 对象相关操作
    - back()、
    - forward()
    - go()
- navigator 对象相关操作
    - userAgent
    - appName
    - appVersion

- screen 对象相关操作：
  - width 、height
- 案例：淘宝搜索框吸顶效果
- 案例：动画回到顶部效果
- 案例：通过 hash 切换的招聘面板

第十七章 Event 事件

- 事件监听器
  - 事件监听和事件绑定的区别
  - addEventListener(type, listener[, options|useCapture])
    - 事件流
      - 事件冒泡
      - 事件捕获
    - 事件监听相关配置
      - capture
      - once
      - passive
  - removeEventListener 取消事件监听
- Event 事件对象
  - Event.target、Event.currentTarget 事件源
  - 事件委托(事件代理)
    - 事件委托的优缺点
  - mousenter、mouseleave 事件
  - Event.stopPropagation()、Event.cancelBubble 取消冒泡
  - Event.clientX、Event.clientY、Event.pageX、Event.pageY 鼠标位置获取
    - 案例：鼠标跟随
- 案例：放大镜效果

- contextmenu 事件
  - return false 和 Event.preventDefault() 阻止默认事件
  - 案例：自定义右键菜单

- 键盘事件
  - keydown、keyup
  - Event.keyCode、Event.keyName
  - Event.altKey、Event.ctrlKey、shiftKey
  - 制作组合键
- 案例：键盘移动操作图片轮播
- 案例：只能输入数字的输入框
- 拖拽思路详解
  - mousedown、mousemove、mouseup
  - 拖拽公式：元素当前位置 = (鼠标当前位置 - 鼠标初始位置) + 元素初始位置
  - 拖拽问题修复
  - 限制范围拖拽
- 案例：妙味经典官网弹窗实现
- 鼠标滚动事件
  - mousewheel 和 DOMMouseScroll 事件

    - Event.detail 和 Event.wheelDelta 滚轮方向获取
- 案例：自定义滚动条控件
- 碰撞检测
    - 矩形碰撞检测
    - 圆形碰撞检测
        - 勾股定理
    - 案例：框选
        - 框选修复 event.button
- 案例：邮箱的批量操作及拖拽删除
- 其他常用事件：
    - dblclick
    - blur、focus、Change、input、submit、reset
    - 表单其他方法：blur()、focus()、select()

第十八章 拖放操作和FileReader

- 元素拖拽
    - draggable 设置元素为可拖放
    - dataTransfer.setData() 设置拖拽时传递信息
    - dataTransfer.getData() 设置拖拽时传递信息
    - 拖放相关事件
        - ondragstart 拖动开始时触发
        - ondrag 拖动过程中触发
        - ondragend 在拖动结束时触发
        - ondragenter 拖动的元素进入放置目标时触发。
        - ondragleave 当拖动的元素在放置目标中移出时触发。
        - ondragover 当元素或文本选择在放置目标中移动时触发。
        - ondrop 在有效放置目标上放置元素或文本选择时触发
- 系统文件的拖放操作
    - dataTransfer.items、dataTransfer.files
    - webkitGetAsEntry 和 isDirectory
    - createReader() 和 readEntries 读取文件夹 目录
- FileReader 读取文件信息
    - 事件处理
        - FileReader.onabort
        - FileReader.onerror
        - FileReader.onload
        - FileReader.onloadstart
        - FileReader.onloadend
        - FileReader.onprogress
    - 相应读取方法
        - FileReader.abort()
        - FileReader.readAsArrayBuffer()
        - FileReader.readAsDataURL()
        - FileReader.readAsText()
- 案例：系统文件拖拽显示

第十九章 音频、视频操作

- 标签
  - audio 、video
  - source

- 支持格式说明
  - 音频：MP3、OggV
  - 视频：Ogg、MP4
- 媒体属性设置
  - controls: 显示或隐藏用户控制界面
  - autoplay: 媒体是否自动播放
  - loop: 媒体是否循环播放
  - currentTime: 开始到播放现在所用的时间
  - duration: 媒体总时间(只读)
  - volume: 0.0-1.0的音量相对值
  - muted: 是否静音
  - autobuffer: 开始的时候是否缓冲加载，autoplay的时候，忽略此属性
  - paused: 媒体是否暂停(只读)
  - ended: 媒体是否播放完毕(只读)
  - error: 媒体发生错误的时候，返回错误代码 (只读)
  - currentSrc: 以字符串的形式返回媒体地址(只读)
- Video 额外属性
  - poster: 视频播放前的预览图片
  - width、height: 设置视频的尺寸
  - videoWidth、 videoHeight: 视频的实际尺寸(只读)
- 媒体方法
  - play(): 媒体播放
  - pause(): 媒体暂停
  - load(): 重新加载媒体
- 全屏显示
  - elem.requestFullscreen()
  - elem.webkitRequestFullScreen()
  - elem.mozRequestFullScreen ()
- 相关事件
  - canplay 当浏览器可以播放音频/视频时
  - canplaythrough 当浏览器可在不因缓冲而停顿的情况下进行播放时
  - durationchange 当音频/视频的时长已更改时
  - error 当在音频/视频加载期间发生错误时
  - loadeddata 当浏览器已加载音频/视频的当前帧时
  - loadedmetadata 当浏览器已加载音频/视频的元数据时
  - loadstart 当浏览器开始查找音频/视频时
  - pause 当音频/视频已暂停时
  - play 当音频/视频已开始或不再暂停时
  - playing 当音频/视频在已因缓冲而暂停或停止后已就绪时
  - progress 当浏览器正在下载音频/视频时
  - ratechange 当音频/视频的播放速度已更改时

    - seeked 当用户已移动/跳跃到音频/视频中的新位置时
    - seeking 当用户开始移动/跳跃到音频/视频中的新位置时
    - stalled 当浏览器尝试获取媒体数据，但数据不可用时
    - suspend 当浏览器刻意不获取媒体数据时
    - timeupdate 当目前的播放位置已更改时
    - volumechange 当音量已更改时
    - waiting 当视频由于需要缓冲下一帧而停止

- 案例：自定义视频播放器

第二十章 第二阶段实战《百度云盘》

云盘也是我们经常会用到的一个应用了，实在是很方便，同时它也是一个前端一个极具代表性的应用了，它几乎能够使用到我们前端开发中经常用到的技术知识：数据操作、dom、事件.....

- 数据驱动 - 通过数据来管理来展示试图和功能，数据结构，递归的使用，无限级树、面包屑导航
- 文件夹创建、文件夹展开、文件夹队列命名
- 自定义右键菜单、重命名、命名检测、删除、移动
- 文件夹全选操作、批量移动、批量删除、碰撞检测、框选

第三阶段 跨平台开发，从前端走向全栈

第二十一章 面向对象&继承&组件开发

- 面向过程与面向对象
    - 面向过程的选项卡
    - 面向对象的选项卡
- 面向对象编程的特点
    - 对象的组成
    - 对象的属性：状态特征的描述
    - 对象的方法：功能的实现
- 对象的创建
    - new Object()
    - 对象字面量：{}
    - 属性和方法的添加
    - 对象创建过程的封装
- 构造函数
    - 运算符new的执行过程和原理分析
    - this的使用
    - 构造函数 - 创建并初始化对象的函数
    - 构造函数书写规范
- prototype原型
    - 什么是prototype
    - prototype和__proto__
    - 原型与原型链
    - 通过prototype实现公有属性和方法的复用和继承
    - prototype使用注意事项
- 混合构造函数/原型方法

- 面向对象的选项卡分析及实现过程
    - 特征分析
    - 属性和方法的接口实现
    - 接口编写原则
- 面向对象编程(OOP)总结
    - 三大特性之一：抽象
- 案例：
    - 面向对象封装选项卡
- 包装对象
    - 什么是包装对象
    - 包装对象的作用
    - String、Number、Boolean
    - 字符串、数字、布尔与字符串对象、数字对象、布尔对象的区别及使用注意事项
- 对象常用操作
    - toString()
    - toString()的重写(overWrite)和实现过程
    - hasOwnProperty()方法实现自有属性判断
- for...in/for...of的使用及特点
- constructor属性的使用
- instanceof运算符
- 继承
    - 继承的特点
    - 继承的实现
        - 构造函数继承
        - 原型继承
        - 拷贝继承
            - 浅拷贝
            - 深拷贝
- ES6 中的面向对象
    - class 关键字
    - extends 关键字 - 继承
    - super 关键字
- 案例：
    - 封装拖拽类
- 组件开发
    - 组件介绍
        - 什么是组件?
        - 组件的特点
            - 方法、配置、事件
    - 组件方法的作用和实现
    - 组件配置的作用和实现
        - 配置的作用
        - 配置的实现
        - 实例配置和默认配置 - extend()
    - 组件事件的作用和实现
        - 事件的作用

        - 基于属性的事件的实现
        - 基于属性的事件的弊端
        - addEventListener事件机制的实现
        - 事件监听器addEventListener
        - 事件触发器trigger
        - 事件容器
    - 事件继承
- 编写自己的工具库
    - 闭包、自执行
    - 对象成员与类成员
    - 基于类的工具封装
        - 判断浏览器
        - 判断类型
    - 基于对象的工具封装
        - 元素结果集包装
        - 常用 DOM 操作封装
        - 链式调用实现
    - 扩展插件实现
        - 对象扩展
        - 类扩展

第二十二章 ES6 高阶使用

- ES6 数据处理
- Symbol
- freeze()、isFrozen()
- getOwnPropertyDescriptor
    - writable
    - enumerable
    - configurable
    - set、get
- Object.defineProperty()
- Object.defineProperties()
- assgin() 合并对象
- Object.is()
- Proxy
- 模块化编程
    - CMD 规范 和 AMD
    - export 和 import

第二十三章 Node.js

- Node.js介绍
- 环境搭建
- 模块化
    - CommonJS 规范
    - 模块加载

        - require 方法 - 导入
        - module 对象
        - exports 对象 - 导出
    - 模块分类
        - 文件模块
        - 文件夹模块
        - 核心模块
        - 第三模块（node_modules）
        - 模块加载机制
            - 相对模块（文件模块、文件夹模块）
            - 绝对模块（第三方模块、核心模块）
- NPM 包管理工具
    - 常用命令
        - init：初始化
        - search：查找
        - install：安装
        - update：更新
        - remove：删除
    - 创建 package 模块
    - package.json 文件
        - name：包名
        - version：版本
        - main：入口程序
        - scripts：执行脚本
        - dependencies：运行依赖
        - devDependencies：开发依赖
    - 注册与发布
        - 注册账号：https://www.npmjs.com/
        - 发布包
            - publish 命令
- Koa 介绍
- 项目创建
- .0
- Koa 安装
- Koa 使用
    - 实例化
        - Application 对象
        - Context 对象
    - 请求
        - Request 对象
    - 响应
        - Response 对象
    - HTTP 协议
        - 头信息
            - 简介
            - Content-Type

        - 状态码
            - 200
            - 404
            - 301、302 - location 头
        - 中间件
            - use 方法
            - next 方法
            - 异步 async
- 实例：
    - 文章信息展示
    - node.js - url 模块
    - node.js - fs 模块
    - node.js - queryString 模块
- 第三方中间件
    - 路由中间件 - Koa-Router
        - use方法
        - get、post
        - redirect方法：重定向
    - 正文解析中间件 - Koa-bodyparser
        - 解析类型
            - form
            - json
            - text
        - 评论留言

第二十四章 客户端信息存储

- cookie 简介
    - 响应头信息
        - set-cookie
    - 请求头信息
        - cookies
    - cookie 属性
        - key：名称
        - value：值
        - expires / max-age：保存时间
        - http-only：安全保护
    - 客户端浏览器 cookie 接口
        - document.cookie
    - 实例：
        - 用户注册登陆
- storage - 本地存储
    - localStorage
        - setItem方法
        - getItem方法
        - removeItem方法
        - clear方法

    - sessionStorage
        - setItem方法
        - getItem方法
        - removeItem方法
        - clear方法
    - storage 事件
    - localStorage 与 sessionStorage 差异
    - 实例：
        - 共享购物车
- Application Cache
    - 简介
    - manifest 属性
    - 缓存清单
        - text/cache-manifest 头信息
        - CACHE字段
        - NETWORK字段
        - FALLBACK字段
    - 缓存状态
        - UNCACHED(未缓存)
        - IDLE(空闲)
        - CHECKING(检查)
        - DOWNLOADING(下载中)
        - UPDATEREADY(更新就绪)
        - OBSOLETE(废弃)
    - 事件
        - cached
        - checking
        - downloading
        - noupdate
        - obsolete
        - updateready
- cookie 与 storage 的差异

第二十五章 前后端交互

- XMLHttpRequest 对象
    - open方法
        - 请求类型
        - url
        - 同步与异步
    - send方法
        - 发送请求
        - get请求与post请求
            - querystring
                - 编码与缓存
            - 请求正文
            - setRequestHeader方法

        - application/x-www-form-urlencoded
    - 事件
        - onload
        - onreadystatechange
    - 属性
        - status：状态码
        - responseText：响应文本
        - responseXML：XML类型
    - ajax 封装
        - 请求封装
        - 请求数据封装
        - 响应数据解析数据封装
- 实例：
    - ajax 注册与登陆
- Ajax 上传实现
- Ajax子集upload使用
- FormData对象
    - append方法
    - content-type 头信息：multipart/form-data
- upload 事件
    - onloadstart
    - onprogress
    - onabort
    - onerror
    - onload
    - ontimeout
    - onloadend
- 实例：
    - 无刷新上传，进度监控、速度计算
- 跨域请求
- 同源策略
- 跨域的问题和常用解决方式
- JSONP
    - JSONP 的概念
    - JSONP 的原理
    - JSONP 的实际应用
- CORS
    - 跨域资源共享标准 - cross-origin sharing standard
    - 简单请求
        - 请求方法求头信息
        - Content-Type 字段值
    - CORS 预检请求
        - 请求方法
        - 头信息
        - Content-Type 字段值
    - HTTP 请求首部字段

        - Origin
        - Access-Control-Request-Method
        - Access-Control-Request-Headers
    - HTTP 响应首部字段
        - Access-Control-Allow-Origin
        - Access-Control-Expose-Headers
        - Access-Control-Max-Age
        - Access-Control-Allow-Credentials
        - Access-Control-Allow-Methods
        - Access-Control-Allow-Headers
- 后端代理
    - 基于node.js的proxy
    - 请求转发

- Fetch 基本用法 Request 对象 Headers 对象 Response对象
    - text方法
    - json方法
- Fetch 与 XMLHttpRequest 的差异
- axios

第二十六章 搞定移动端

- 移动端touch事件
    - touchstart
    - touchmove
    - touchend
    - touch 事件 和 mouse 事件的区别
    - 事件点透
        - mouse 事件的延迟问题
    - 阻止默认事件
        - 阻止 touchstart 事件带来的影响
        - 阻止 touchmove 事件带来的影响
- TouchEvent 对象详解
    - touches
    - targetTouches
    - changedTouches
- 案例：移动端滑屏切换的幻灯片
    - 上下滑屏误触问题
    - 固定条件下阻止默认事件
- 移动端的多指操作
    - gesturestart
    - gesturechange
    - gestureend
    - 自定义多指事件兼容安卓
        - Math.atan2() 方位角使用
        - 角度和弧度的转换
        - 手指缩放距离获取

        - 勾股定理
- 移动端的陀螺仪操作
    - orientationchange 横竖屏切换
        - orientation 判断手机横竖屏
    - devicemotion
        - acceleration 手机加速检测
        - accelerationIncludingGravity 重力加速检测
    - deviceorientation
        - alpha、gamma、beta
- 案例：720度家居装修图
- 案例：摇一摇功能实现

第二十七章 better-scroll

- better-scroll 基础使用方法
    - new BScroll(wrap)
    - better-scroll 原理说明
    - 使用 better-scroll 时注意问题
    - 利用 better-scroll 实现各种滑屏
    - better-scroll 基础使用方法
        - startX、startY
        - scrollX、scrollY、freeScroll
        - eventPassthrough
        - bounce、bounceTime
        - momentum
        - refresh() 方法
- 案例：自定义滚动条
    - scrollbar 高级组件
- 案例：手机淘宝幻灯片实现
    - snap 高级组件
    - probeType 和 scroll 事件
    - scrollEnd 事件
    - getCurrentPage()
- 案例：仿 iso 经典选择器控件
    - wheel 高级组件
    - getSelectedIndex()
- 案例：上拉加载和下拉刷新功能实现、
    - pullDownRefresh 和 pullUpLoad 高级组件
    - finishPullDown()、openPullDown(config)、closePullDown()
    - finishPullUp()、openPullUp(config)、closePullUp()
    - pullingDown、pullingUp 事件
- 案例：自定义索引列表

第二十八章 地理信息获取和百度地图API

navigator.geolocation

- 单次定位请求 ：getCurrentPosition(请求成功，请求失败，数据收集方式)
    - 请求成功函数
        - 经度: coords.longitude
        - 纬度: coords.latitude
        - 准确度: coords.accuracy
        - 海拔: coords.altitude
        - 海拔准确度: coords.altitudeAcuracy
        - 行进方向: coords.heading
        - 地面速度: coords.speed
        - 时间戳 : new Date(position.timestamp)
    - 请求失败函数
        - 失败编号 ：code
            - 0: 不包括其他错误编号中的错误
            - 1: 用户拒绝浏览器获取位置信息
            - 2: 尝试获取用户信息，但失败了
            - 3: 设置了timeout值，获取位置超时了
    - 数据收集:
        - enableHighAcuracy: 更精确的查找，默认false
        - timeout: 获取位置允许最长时间，默认infinity
        - maximumAge: 位置可以缓存的最大时间，默认0
    - 多次定位请求 : watchPosition(像setInterval)
        - 移动设备有用，位置改变才会触发
        - 配置参数：frequency 更新的频率
    - 关闭更新请求 : clearWatch(像clearInterval)

- 百度地图API 基本使用
    - 密钥申请
    - 百度坐标转换
    - 创建地图
    - 添加标注
    - 标记信息
    - 本地存储信息持久化
- 案例：行走日记

第二十九章 探秘《淘宝造物节》类VR场景核心原理

绚丽的3D空间、震撼的视觉表现、好玩的用户操控..... 类VR场景，给了我们一种新的玩交互的方式，当然，类VR场景，对我们的移动端的Touch交互，陀螺仪交互，也是一个综合的训练，除此之外也可以极大提升我们对于CSS3 3D 特性的理解

- 图片预加载、animation
- 3D 爆炸效果实现
- 三角函数及多边形计算
- 3D 移动端适配
- 3D 场景搭建

第三十章 正则表达式

- 字符串检索、匹配
    - 基于字符的操作
        - 字符串操作方法
            - charAt
            - indexOf
            - substring
            - includes
            - ……
        - 实例
            - 单个数字字符检索
            - 连续数字字符检索
        - 基于正则的操作
            - 是什么？做什么？怎么做？
                - js中的正则
                    - new RegExp('正则表达式')
                    - 字面量：/正则表达式/
                - 组成
                    - 普通字符
                    - 特殊字符（元字符）
                    - 模式（修饰）
- 哪些地方可以使用正则？
- 在字符串一些方法中使用正则
    - match
    - search
    - replace
    - split
- 通过正则对象进行调用
    - test()
    - exec()
- 元字符概念
- 元字符分类
    - 字符类别（Character Classes）
        - .
            - 匹配行结束符（\n \r \u2028 或 \u2029）以外的任意单个字符
            - 在 字符集合（Character Sets） 中，. 将失去其特殊含义，表示的是原始值
        - \
            - 转义符，它有两层含义
                - 表示下一个具有特殊含义的字符为字面值
                - 表示下一个字符具有特殊含义（转义后的结果是元字符内约定的）
        - \d 匹配任意一个阿拉伯数字的字符
        - \D 匹配任意一个非阿拉伯数字的字符

        - \w 匹配任意一个（字母、数字、下划线）的字符
        - \W 匹配任意一个非（字母、数字、下划线）的字符
        - \s 匹配一个空白符，包括空格、制表符、换页符、换行符和其他 Unicode 空格
        - \S 匹配一个非空白符
        - \t 匹配一个水平制表符（tab）
        - \r 匹配一个回车符（carriage return）
        - \n 匹配一个换行符（linefeed）
        - \v 匹配一个垂直制表符（vertical tab）
        - \f 匹配一个换页符（form-feed）
    - 字符集合（Character Sets）
        - [xyz]
            - 一个字符集合，也叫字符组。匹配集合中的任意一个字符。你可以使用连字符'-'指定一个范围
            - [xyz] 是一个反义或补充字符集，也叫反义字符组。也就是说，它匹配任意不在括号内的字符。你也可以通过使用连字符 '-' 指定一个范围内的字符
    - 边界（Boundaries）
        - ^
            - 匹配输入开始。如果多行（multiline）标志被设为 true，该字符也会匹配一个断行（line break）符后的开始处
        - $
            - 匹配输入结尾。如果多行（multiline）标志被设为 true，该字符也会匹配一个断行（line break）符的前的结尾处
        - \b
            - 匹配一个零宽单词边界（zero-width word boundary）
        - \B
            - 匹配一个非零宽单词边界（zero-width word boundary）
    - 分组（grouping）
        - (子项)
            - 可以使用 () 对表达式进行分组，类似数学中分组，也称为子项
            - 索引分组
            - 命名分组
                - (?...)
                - groups属性
            - 捕获匹配
                - 具有捕获（capturing）特性，即会把匹配结果保存到（子项结果）中
                - (x)
            - 非捕获匹配
                - 不具有捕获（capturing）特性，即不会把匹配结果保存到（子项结果）中
                - (?:x)
            - 零宽断言/预查（Assertions）
                - 用于指定查找在某些内容(但并不包括这些内容)之前或之后的内容
                - 正向零宽断言/预查
                    - 肯定
                        - (?=pattern)
                    - 否定
                        - (?!pattern)

        - 负向零宽断言/预查（注意：ES2018新增）
            - 肯定
                - (?<=pattern)
            - 否定
                - (?<!patten)
        - 捕获与零宽断言的区别
            - 捕获：匹配的内容出现在结果中但不出现在子项结果中
            - 零宽断言：完全不会出现在结果
    - 反向引用（back references）
        - \n
            - 这里的 n 表示的是一个变量，值为一个数字，指向正则表达式中第 n 个括号（从左开始数）中匹配的子字符串
    - 数量词（Quantifiers）
        - x{n}
            - n 是一个正整数。前面的模式 x 连续出现 n 次时匹配
        - x{n,m}
            - n 和 m 为正整数。前面的模式 x 连续出现至少 n 次，至多 m 次时匹配
        - x{n,}
            - n 是一个正整数。前面的模式 x 连续出现至少 n 次时匹配
        - x*
            - 匹配前面的模式 x 0 或多次
        - x+
            - 匹配前面的模式 x 1 或多次。等价于 {1,}
        - x?
            - 匹配前面的模式 x 0 或 1 次
        - x|y
            - 匹配 x 或 y

- 匹配模式
    - g
        - global，全局模式：找到所有匹配，而不是在第一个匹配后停止
    - i
        - ignore，忽略大小写模式：匹配不区分大小写
    - m
        - multiple，多行模式：将开始和结束字符（^和$）视为在多行上工作，而不只是匹配整个输入字符串的最开始和最末尾处
    - s
        - dotAll / singleline模式：. 可以匹配换行符
    - u
        - unicode，unicode模式：匹配unicode字符集
    - y
        - sticky，粘性模式：匹配正则中lastIndex属性指定位置的字符，并且如果没有匹配也不尝试从任何后续的索引中进行匹配
    - 新增特性
        - https://github.com/tc39/ecma262

- 案例：QQ验证
- 案例：邮箱验证
- 案例：标签过滤
- 案例：敏感词过滤

第三十一章 第三阶段实战《移动端企业网站制作》

这是一个完整的移动端项目，几乎包含了我们在移动端开发时，所需要的各种功能，通过该项目的学习，可以帮助我们快速掌握移动开发及前后端数据传输时，所需的各项技巧

- 跨域的登录和注册
- 滑屏幻灯片
- 自定义滚动条 和 上滑加载
- 留言 与 点赞功能
- 移动端弹窗，及弹窗问题解决

第四阶段 高级前端项目工程化开发实践

第三十二章 React.js 全家桶

- ReactDOM.render 方法
- JSX
  - 什么是 JSX
  - {} 语法
  - 表达式
    - JS 表达式
      - Number、String、Array
      - Boolean、Null、Undefined 不输出
      - Object
    - JSX 表达式
  - 属性
    - 一般属性
    - style属性：{ {} }
    - className属性
  - 子元素
  - JSX 防止注入攻击
    - XSS（cross-site-scripting, 跨站脚本）
  - 虚拟 DOM 对象
  - 更新渲染
  - 条件渲染
    - && 运算符
    - ? : 三元运算符
    - 阻止渲染
  - 列表渲染
    - Array.prototype.map 的使用
    - Keys

        - 唯一性
- create-react-app
    - 安装
    - 常用命令
        - create-react-app <项目名称>
        - npm start：启动当前应用
        - npm run build：构建打包应用
- 组件基础
    - 组件类型
        - 函数组件
        - class组件
            - React.Component 类
    - 组件的使用
    - 组件复用
    - 组件的 `props`
        - 只读性
    - 组件的 `state`
        - 有状态组件与无状态组件
        - class 组件特性
        - setState 方法
        - 异步更新
        - 更新合并
    - prop 与 state 的区别
- 数据流
    - 从上至下（从外至里）
    - 子组件：children
- 事件
    - 函数 `props`
    - 事件处理程序传递参数
    - 从下至上（从里至外）传递数据
    - this 绑定
    - 事件对象
- 表单
    - 受控组件
    - 非受控组件
        - defaultValues
- 生命周期
    - 组件执行流程
    - Mounting（挂载阶段）
        - constructor
        - static getDerivedStateFromProps
        - render
        - componentDidMount
    - Updating（更新阶段）
        - static getDerivedStateFromProps
        - shouldComponentUpdate

        - render
        - getSnapshotBeforeUpdate
        - componentDidUpdate
    - Unmounting（卸载阶段）
        - componentWillUnmount
- 路由介绍
- 安装
    - npm install react-router-dom
- 引入使用
    - import {...} from 'react-router-dom'
- Router 组件
    - 容器组件
    - 低阶路由组件
- BrowserRouter 组件
    - 高阶路由组件
- HashRouter 组件
    - 高阶路由组件
- Route 组件
    - 定义路由
    - path 属性
    - component 属性
    - exact 属性
    - strict 属性
- 动态路由 / 路由参数 - queryString
- match 对象
    - 获取当前组件对应路由信息
- Link 组件
    - 动态链接
    - to 属性
    - replace 属性
- NavLink 组件
    - 高阶版的 Link 组件
    - isActive 属性
    - activeStyle 属性
    - activeClassName 属性
- Switch 组件
    - 多组选择匹配
- Redirect 路由重定向
    - to 属性
    - from 属性
- withRouter
    - 跨组件数据共享
- Redux
    - 简介
        - 数据状态管理器
    - 安装

        - npm install --save redux
    - store：数据容器
        - 创建数据容器：Redux.createStore 方法
        - 获取容器数据：getState 方法
    - action
        - 动作
        - action type：动作类型
        - action creater：动作实体
    - reudcer
        - 纯函数
        - 修改、更新数据
    - dispatch：任务派遣
    - subscribe：订阅
    - Middleware 中间件
        - 概念
        - redux-logger
        - 安装：npm i --save redux-logger
        - applyMiddleware 方法
        - redux-saga 的按照和使用
- React Redux
    - 安装：npm install react-redux
    - Provider 组件
        - store 属性
    - connect 方法
        - component 与 store 的联结器
        - props 属性
- Redux DevTools 工具
    - 安装
    - 在 Redux 中使用

- Ant Design UI 组件库
    - Ant Design 安装流程
    - 在 create-react-app 中使用
    - Ant Design 组件库 API 说明
- 案例：QQ好友列表
- 案例：ToDoList
- 案例：《CNode 实战》

第三十三章 Webpack

- 模块化回顾
- webpack 安装
- webpack 基础打包功能与结构分析
- 配置
    - entry
    - output
    - loaders

    - webpack 中的 module
        - file-loader：静态文件资源处理模块
        - url-loader：基于 file-loader 的 url 处理模块
        - 样式处理：
            - css-loader：css 样式处理模块
            - style-loader：style 样式注入模块
            - sass-loader：css 预处理器模块
            - postcss-loader：css 样式前缀处理模块
        - plugins
            - HtmlWebpackPlugin：html 文件构建
            - clean-webpack-plugin：打包文件清理
            - mini-css-extract-plugin：css 文件提取
    - sourceMap
    - WebpackDevServer
    - Hot Module Replacement (HMR:热模块替换)
        - js 模块处理
    - babel
        - 基础
        - @babel/polyfill
        - @babel/plugin-transform-runtime
        - 配置：.babelrc
    - 配置React打包环境
    - tree Shaking
    - development vs Production模式区分打包
    - code Splitting：代码分割

第三十四章 TypeScript

- TypScript 介绍
- TypeScript 环境配置
- TypeScript 编译命令
- TypeScript 编译配置
    - tsconfig.json
- 类型系统
    - 类型标注
    - 类型检测
    - 基础类型
- 接口
    - 基础语法
    - 可选属性
    - 只读属性
    - 任意属性
    - 索引类型
        - 索引签名
            - 字符串索引签名
            - 数字索引签名
- 类型深入

    - 类型别名
    - 联合类型
    - 交叉类型
    - 字面量类型
    - 类型断言
    - 类型推导
    - 类型保护
- 函数
    - 标注语法
    - 函数类型接口
    - 参数标注
    - 剩余参数
    - 函数中的 this
    - 函数重载
- 类
    - 基础语法
    - 成员属性与成员方法
    - 继承
        - extends
        - super
    - 修饰符
        - public
        - protected
        - private
        - readonly
    - 寄存器
        - getter
        - setter
    - 静态特性
        - static
    - 抽象类
        - abstract
    - 类与接口
        - implements
        - 接口继承
    - 对象类型与类类型
    - 构造函数参数属性
- 泛型
    - 泛型基础概念于使用场景
    - 泛型函数
    - 泛型参数
    - 多泛型
    - 泛型默认类型
    - 泛型类
    - 泛型接口
    - 泛型约束

    - 泛型类类型
- 装饰器
    - 装饰者模式
    - —experimentalDecorators 配置
    - 装饰器语法
        - 装饰器使用限制
    - 装饰器分类
        - 类装饰器
        - 属性装饰器
        - 方法装饰器
        - 参数装饰器
        - 访问装饰器
    - 装饰器实现
        - 装饰器工厂
        - 装饰器求值
    - 装饰器组合
    - 元数据
        - metadata
        - reflect-metadata 库
        - —emitDecoratorMetadata 配置
- 模块系统
    - 模块导出
    - 模块导入
    - ES6 Modules 与 CommonJS&AMD
        - export = 和 import = require()
    - 内部模块
        - 命名空间：namespace
- 高级主题
    - 声明文件 - 造福全人类
        - 作用
        - 结构规范
        - 查找与安装
        - 发布
    - JSX
        - React & TypeScript
    - 配置详解 - tsconfig.json

第三十五章 Vue.js 全家桶

- new Vue()
    - el 选项
    - data 选项
- 模板渲染
    - {{}} 语法

- 指令
  - 输出
    - v-text、v-html……
  - 属性绑定
    - v-bind
    - 样式：class 与 style 绑定
  - 条件渲染
    - v-if、v-else、v-else-if、v-show
  - 列表渲染
    - v-for
  - 事件绑定
    - v-on
      - 参数
        - event 对象
      - 修饰符
  - 表单
    - v-model：双向数据绑定
- 计算属性
  - computed
  - getter
  - setter
- 侦听器
  - watch
- vue-cli
  - 安装：npm install -g @vue/cli
  - 常用命令
    - 通过命令行创建应用：vue create <应用名称>
    - 通过 UI 界面方式创建应用：vue ui
    - 启动本地应用：npm run serve
    - 打包：npm run build
  - 单文件组件
    - template
    - script
    - style
    - lang 属性
- 自定义指令
  - 全局 Vue.directive && 局部 directives 选项
  - 钩子函数
  - 钩子函数参数
  - 函数简写
    - 属性对象字面量

- 组件基础
    - 全局 Vue.component && 局部 components 选项
        - 组件名称约定
        - 选项
            - template：组件模板
                - 顶层节点
            - data：组件私有数据 -（React -> state）， 必须是函数
        - 组件数据通信基础
            - props
            - event
                - $emit
        - 插槽
            - 内置 <slot> 组件
            - slot属性
            - 具名插槽
                - name 属性
- 动画
    - 过渡条件
        - v-if、v-show、组件根节点、动态组件
    - 过渡管理操作
        - css
        - js
    - css 过渡
        - 过渡类名
        - 内置 <transition> 组件
            - name 属性
    - js 过渡
        - 过渡钩子
- props 验证
    - 类型检查
    - 非 props 特性
    - 继承特性
    - 禁用继承特性
- v-model
    - model 选项
        - value
        - event
- .sync 修饰符
- 组件生命周期
    - beforeCreate

    - created
    - beforeMount
    - mouted
    - beforeUpdate
    - updated
    - beforeDestroy
    - destroyed

- 动态组件
    - 内置 <component> 组件
    - 内置 <keep-alive> 组件
    - 生命周期
        - activated
        - deactivated

- 过滤器
    - 全局 Vue.filter && 局部 filters 选项
    - 管道符：|

- 插件
    - Vue.use 方法
    - install 方法

- 路由介绍

- 路由安装
    - npm install vue-router

- 实例化
    - new VueRouter
    - 选项
        - base：基础路径
        - mode：模式
            - history
            - hash
        - routes：路由对象
            - path：路由路径
            - component：路由绑定的组件
            - name：路由名称

- <router-view> 组件

- <router-link> 组件
    - to 属性
    - replace 属性

    - append 属性
    - tag 属性
    - event 属性
    - active-class 属性
    - exact 属性
    - exact-active-class 属性

- 动态路由
    - router 选项
    - $router：全局router对象
        - 跳转路由：push、replace
    - $route：当前路由对象
        - 获取路由数据
            - params 属性，动态路由
            - query 属性，获取 queryString

- 导航守卫
    - 路由导航解析流程
    - 全局守卫
        - beforeEach
            - to 属性
            - from 属性
            - next 函数
        - beforeResolve
        - afterEach
    - 路由独享守卫
        - beforeEnter
    - 组件内守卫
        - beforeRouteEnter
        - beforeRouteUpdate
        - beforeRouteLeave

- 路由元信息
    - meta 属性

- 路由懒加载
    - import 方法
    - 分组
        - webpackChunkName

- Vuex 介绍
- Vuex 安装
    - npm install vuex
- 实例化

    - new Vuex.store
    - state 属性
        - 元数据存储仓库
    - getter 属性
        - 派生数据存储仓库，类似组件计算属性
    - mutation 属性
        - 存储动作，用来修改仓库中的数据
            - state，payload
        - 同步操作
        - 无返回结果
    - action 属性
        - 类似 mutation
        - 异步操作
        - 返回 Promise

第三十六章 git 版本控制工具

- git 简介与安装
    - 版本控制系统
    - window 及 mac安装
- 入门命令
    - 创建仓库 init
    - 查看文件状态 status
    - 追踪文件 add
    - 提交 commit
    - 查看提交状态 log
    - 删除文件 rm
    - 移动文件 mv
    - 查看修改 diff
- 文件的三种状态及三个工作区域
    - 三种状态：已修改、已暂存、已提交
    - 工作区域工作区、暂存区、仓库
- git工作流程
- 中文乱码的处理
- git 的分支管理
    - master
    - 可变指针
    - 创建分支 branch
    - 切换分支 checkout
    - 创建并切换分支 -b
    - 合并分支 merge
    - 合并分支 rebase
        - merge 和 rebase 的区别
    - 快速前移
        - 快速前移产生的问题
        - 禁止快速前移 --no-ff
    - 分支冲突的产生及解决

    - 删除分支 -d / -D
    - 标签操作 tag
- git 版本控制
    - 取消合并 --abort
    - 撤销提交 --amend
    - 取消暂存 reset
    - 撤销文件修改 checkout
- git存储
    - 存储暂存区域内容 stash
    - 查看存储内容 stash list
    - 取出存储内容 stash apply
    - 移除存储 stash drop
- 简化操作
    - 命令别名 alias
    - .gitignore 配置文件
- git 远程仓库同步
    - 提交到远程仓库 push
        - 不同分支的推送
        - 分支的删除
        - 标签的推送
        - 标签的删除
    - 克隆远程仓库至本地 clone
        - 克隆分支
    - 拉取更新 pull
    - SSH 密钥 的生成及使用

第三十七章 总结与面试

- 面试题解析
    - Promise 源码解析
    - 事件轮询(Event loop)、宏任务、微任务
    - 函数式编程、函数柯里化
- JD分析及就业面试准备