前端高薪进阶训练营

开营破局 · 打开晋升通道

- 前端如何一步一步变成大前端
- 大前端时代下的前端开发
- 前端开发者的进阶路径
- 训练营整体内容概述

Part 1 · JavaScript 深度剖析

> *JavaScript 语言本身及周边*

ECMAScript 新特性

- JavaScript vs. ECMAScript
- 块级作用域、模板字符串
- 对象与数组的解构、rest 操作符
- 函数进阶（箭头函数、默认参数）
- 对象和数组的扩展用法
- Proxy、Reflect、Map、Set、Symbol
- for...of、迭代器模式、生成器函数
- ES Modules 模块系统
- ES2016 - ES2020（ES7 - ES11）特性一览
- 新特性编译工具（Babel）的使用
- 新特性的 Polyfill：CoreJS 标准库

JavaScript 异步编程

- JavaScript 的单线程设计
- 同步模式和异步模式的调用差异
- 回调函数的执行原理
- Promise 异步方案的使用进阶与剖析
- 处理异步任务的任务队列和事件循环
- JavaScript 内部的宏任务与微任务
- ES 6 Generator 迭代器的异步应用
- 使用 Async / Await 语法糖编写扁平的异步代码

TypeScript 高级编程

- 编程语言的几种不同类型系统
- JavaScript 自有类型系统的问题
- Flow 静态类型检查方案
- Flow 工具的配置及相关插件的使用
- TypeScript 基本语法
- TypeScript 高级特性（泛型、接口）
- TypeScript 内置对象标准库
- TypeScript 的类型声明

函数式编程范式

- 函数式编程的本质以及应用场景
- 如何以函数式编程风格创建应用程序
- 用简单的代码构建复杂的应用程序
- 纯函数的定义以及为什么使用纯函数
- 为什么消除和控制副作用如此重要
- 柯里化、compose、高阶函数的优点
- 不可变的数据结构
- 常见库（Lodash、Ramda.js）

JavaScript 性能优化

- JavaScript 中的垃圾收集
- JavaScript 内存管理
- V8 垃圾回收机制分类
- 引用计数、标记清除、标记整理和增量标记
- Preformance 工具的使用及注意事项
- 20 个代码层面的优化细节

Part 2 · 前端工程化实战

> *远离刀耕火种的原始时代，全面提升战斗力*

工程化的组成

- 工程化组成和项目中的表现
- 工程化与工具之间的关系

脚手架工具

- 脚手架设计思想与目标
- 脚手架工具的本质作用
- 常用的脚手架工具一览
- Yeoman 的基本使用以及自定义 Generator
- Yeoman Sub Generator 特性
- 基于 Yeoman 创建适合自己的脚手架工具
- Plop 生成器的基本使用
- 使用 Plop 提高项目创建同类文件的效率
- 脚手架工作原理剖析
- 手写一个自己的脚手架工具

自动化构建

- 如何使用自动化构建提高开发效率
- 常用的自动化构建任务执行器
- npm scripts & script hooks
- Grunt 工具的使用及扩展开发
- Gulp 的使用以及任务结构
- 基于 Gulp 创建自动化构建工作流
- 封装独立的自动化构建工具
- FIS 3 的使用以及常用的扩展插件

自动化测试

- 自动化测试的主要分类：单元测试、集成测试、E2E 测试
- 高性能应用开发所必要的性能测试与压力测试
- 常见的自动化测试框架与自动化测试的实现原理
- Mocha、Jest、Enzyme、Cypress、Nightmare、Puppeteer
- 前端项目中自动化测试的最佳实践（基础设施、公共组件的测试）

自动化部署（CI / CD）

- 持续集成与持续部署
- 基于 GitHub / GitLab 的自动化工作流搭建
- 常见的 CI 实践：Jenkins、GitLab CI、Travis CI、Circle CI
- 开源项目的新选择：GitHub Actions
- 基于常用 CI 系统实现静态 / Node 类型的项目自动部署

模块化开发与 Webpack

- 模块化标准与规范
- ES Modules 标准的支持情况
- Webpack 打包工具的基本使用

- Webpack 的配置详解
- Webpack 打包过程和打包结果分析
- Webpack 中资源模块的加载（Loader）
- 如何开发一个 Webpack 的 Loader
- Webpack 的插件机制
- 开发一个 Webpack 插件
- Webpack 周边生态（Dev Server、HMR、Proxy）
- Webpack 高级特性（Tree-shaking、sideEffects)
- Webpack 打包过程及打包结果的优化
- 深度剖析 Webpack 实现原理（AST 语法树）
- 其他常见的打包工具（Rollup、Parcel）

规范化标准

- 常见的代码 Lint 工具（ESLint、Stylelint）
- 创建项目或者团队专属的 Lint 规则 / 风格
- 通用型代码格式化工具 Prettier
- 结合自动化工具或者 Webpack 的使用
- 配合 Git Hook 确保源代码仓库中代码的质量
- 结合脚手架、自动化、模块化、规范化搭现代化前端工程

Part 3 · 核心框架原理与进阶

> *前端三大核心框架的原理深入以及相关高阶技能*

Vue.js 原理深度剖析

- Vue.js 框架基础回顾
- Vue CLI 基础设施深度解剖
- 数据响应式实现原理分析
- 虚拟 DOM 和 Diff 算法的实现
- 模板编译模块的实现原理
- Vue Router 源码剖析

Vue.js 高级与进阶

- 封装自己的 Vue 组件库
- Vue 项目性能优化
- Vuex 数据流管理方案
- 使用 TypeScript 开发 Vue.js 应用
- 原生服务端渲染（SSR）的实现、同构开发
- Nuxt.js 集成式 SSR 框架

- 静态站点生成（SSG）方案及 Gridsome
- Vue.js 3.0 设计和用法的变化以及优势
- Vue.js 3.0 Composition APIs
- Vue.js + Vue Router + Vuex + TypeScript 实战项目开发

React 设计原理解密

- React 框架基础回顾、JSX 语法
- 分析 Virtual-DOM 目的及实现原理
- React 核心算法：Fiber
- React 框架的设计哲学
- React 框架核心源码解读

React 进阶与实战

- 封装 React 自定义组件库
- React 组件的性能优化
- 受控和非受控组件的选用标准
- React 组件的自动化测试
- React 16.8 Hooks 特性的使用以及实现原理分析
- CSS-in-JS 方案以及 emotion 库
- 现代化 React 应用 UI 框架 Chakra-UI
- 使用 TypeScript 开发 React 应用
- React 数据流方案：Redux、Mobx
- Redux 常用中间件以及中间件的开发
- 原生服务端渲染（SSR）的实现、同构开发
- Next.js 集成式 SSR 框架
- 静态站点生成（SSG）方案及 Gatsby 框架
- React + React Router + Redux + Ant Design + TypeScript 实战

Angular 企业实战开发

- Angular 9 基础
- Angular 数据绑定及实现原理
- Angular 组件封装及父子组件通信
- Angular 服务模块及服务注入
- Angular 路由模块
- RxJS 响应式编程的库
- NgRx 状态管理工具

Part 4 · Node.js 全栈开发

> *打通前后端边界，做炙手可热的“无所不能”*

Node.js 高级编程

- 非堵塞 IO、EventLoop、事件队列
- CommonJS 原理解析
- 核心模块、自定义模块、第三方模块
- 文件系统、Buffer 对象、字符编码
- 压缩和解压缩、加密和签名算法
- 网络编程、TCP/IP、HTTP 服务
- cookie 和 session 原理
- 多进程和集群搭建
- 搭建反向代理服务器

NoSQL 数据库

- NoSQL 数据库特性及优势介绍
- MongoDB 的安装、连接、操作
- mongoose 模块以及常用的操作 API
- Redis 快速上手以及它所适合的场景
- 使用 Node.js 操作 Redis

Web 开发框架

- Express 完成基本的服务端应用开发
- Express 路由、模板引擎、错误处理
- Express 中间件机制的设计思想
- Express 中间件使用以及自定义中间件
- Express 应用程序的进程管理器
- Express 安全与性能的最佳实践
- Express + Handlebars + Mongoose 实战
- Koa 应用与实践、AOP 面向切面编程
- Koa 中间件实现、源码深度剖析
- Koa 的中间件模型与 Express 的差异
- PM2 宿主部署 Node.js 应用

GraphQL API 开发

- 基于 Koa 开发 RESTful API
- 应用层最佳接口实践：GraphQL
- GraphQL 规格标准与设计优势
- GraphQL 快速开发库：Apollo
- API 鉴权标准、jsonwebtoken 模块及其相关 API

- Docker Compose + GitLab CI 自动化部署 Node.js 应用

企业级框架

- Egg.js 项目架构与脚手架工具
- Egg.js 中间件机制、洋葱圈模型
- Egg.js 路由、控制器、服务
- Egg.js 插件机制以及插件开发
- Egg.js 定时任务调度
- Egg.js + Mongoose + Nunjucks + TypeScript 项目实战
- Nest.js 框架的基本概念和内部组成
- 使用 Nest.js 框架构建高效且可伸缩的服务端应用
- Nest.js 面向切面编程、依赖注入的实践
- Adonis.js 框架介绍

Part 5 · 泛客户端开发

> *泛客户端大行其道，高级前端开发者务必掌握多端开发，从容应对不同客户端载体*

小程序与快应用

- 原生小程序 MINA 框架回顾
- 基于 mpvue 框架开发小程序应用
- 基于 mpvue 框架打包快应用和 H5
- 京东 Taro 多端统一开发方案
- uni-app 多端统一开发方案

Hybrid App 开发

- 基于 WebView UI 的基础方案
- Cordova / Ionic 通用型混合 App 开发框架
- Cordova 的实现原理分析以及它的常用插件
- H5 配合原生 WebView 开发混合式 App
- 通过 JSBridge 完成 H5 与 Native 的双向通讯
- 原生 App 开发相关知识

React Native

- React Native 开发环境搭建
- 初始环节搭建以及相关基础配置
- 热更新的开发体验
- 使用 Flexbox 实现界面布局

- 常见界面布局和长列表呈现
- 接入第三方 Native 组件（Objective-C / Swift / Java）
- React Native 架构的实现原理

Flutter 原生 App 开发

- Flutter 概述以及 Windows / macOS 环境搭建
- Dart 语言快速上手、包管理工具
- Flutter 快速上手、开发体验、路由和导航
- UI 开发：内置 Material Design 和 Cupertino（iOS风格）Widget
- 常用 Widget、表单组件、布局方式
- 数据响应：界面状态管理
- 网络编程以及相关第三方包
- Native 功能和 SDK 的调用
- Flutter 在线课堂项目实战案例

Electron 桌面应用开发

- Electron 运行时的基本结构分析
- 快速上手、常用 API、基础案例
- 主进程与渲染进程之间的差异以及相互通信
- 常见桌面应用程序功能的实现
- Electron 结合 React / Vue.js 之类的前端框架
- Electron 应用的调试（主进程与渲染进程）以及相关工具（Spectron / Devtron）
- 集成式打包工具（electron-builder / electron-packager / electron-forge）
- 实战案例：模仿 Microsoft To Do

Part 6 · 商业级技术解决方案

> *通过更多的行业商业级技术解决方案，落地更多优秀技术*

Serverless 无服务端方案

- BaaS / FaaS / PaaS 服务
- Serverless 架构与实现原理
- Serverless 应用场景与局限性
- 国外常见的 Serverless 服务（ZEIT Now、Netlify）
- 国内常见的 Serverless 服务（阿里云、腾讯云）

中途岛 / 中间层方案

- BFF 架构的优势及常见方式

- 基于 Node.js 的中间层架构
- 实现更合理的前后端分离架构
- 中间层的目标与职责
- 后端细粒度接口聚合
- 服务端模板渲染
- 前端路由设计

首屏性能提升方案

- 白屏加载和首屏加载时间的区别
- 骨架屏：渲染一些简单元素进行占位
- 使用 PWA 开发可离线化应用
- 客户端缓存策略
- 利用 script 的 async 和 defer 异步加载
- 前端资源的分块 / 按需加载

数据埋点方案

- 数据埋点的原理分析
- 页面访问量统计
- 功能点击量统计
- 埋点系统的实现

长列表无限滚动方案

- 触底加载更多功能的实现
- 长列表渲染卡顿问题的原因
- 高性能长列表渲染的思路：虚拟列表
- 不同框架下长列表无限滚动的实现方法
- 高性能滚动及页面渲染优化

API 接口鉴权方案

- JSON Web Token 方案介绍
- jsonwebtoken 模块及其相关 API
- JWT 创建与签发、解码与验证
- Node.js 鉴权中间件的实现
- Axios 统一鉴权模块
- React / Vue.js 框架下客户端路由鉴权

更多常见方案

- 渐进式加载方案

- RBAC 权限管理解决方案
- 接口 Mock 方案
- OSS 云存储方案
- H5 直播方案
- 多语言化方案
- 防盗链方案
- CDN 加速方案
- 更多商业级技术解决方案请咨询。。。

Part 7 · 高阶技术专题

> *更多高阶技术专题*

微前端架构与实践

- 微前端诞生的背景和解决的问题
- 微前端下的工程化实践
- 如何同时支持 React / Vue.js / Angular 等不同的框架
- 开发一个简单的微前端框架

PWA 渐进式 Web 应用

- PWA 使用场景分析
- 服务端 / 客户端离线缓存技术
- 浏览器多线程环境
- 通过 Service workers 让 PWA 离线工作
- ServiceWorkers 的生命周期
- 基于 PWA 的消息推送、应用更新
- 渐进式加载

数据可视化

- 相关知识储备：Canvas、SVG
- 数据可视化的目标
- 实现数据可视化的常用方式
- 相关库：D3.js、AntV、ECharts.js

现代化 Web 101 架构剖析

- Web 应用主流架构概览
- 域名、DNS、负载均衡等相关概念的普及
- Web 应用服务端、数据库服务器

- 缓存服务、任务队列服务
- 云存储、CDN

Web Components

- Custom Elements
- Shadow DOM
- HTML Templates
- Web Components 案例
- Vue 组件转换成原生组件

更多技术专题

- CSS 预 / 后处理器（Sass、PostCSS）
- CSS 架构（BEM、CSS-in-JS、emotion、styled-components）
- 移动端真机调试
- Web 安全专题（HTTPS、XSS / CSRF、CSP）
- 前端应用性能专题
- Web Assembly
- 更多高新技术专题请咨询。。。

Part 8 · 大厂面试指导

> *临门一脚，如何脱颖而出，立于不败之地？*

Leet Code 精选题

- 常用数据结构介绍
- 常见算法题解析
- 常见数据结构题解析

BATJ 高频面试真题

- BATJ 高频原理题解析
- BATJ 高频应用题解析

面试专项能力突击

- 一面：编程基础能力考察
- 二面：项目经验考察
- 三面：方案设计综合能力考察
- 四面：HR综合软素质考察

面试过程发挥应有水平

- 学会表达：如何把硬实力表达出来
- 扬长避短：如何表现出自己的闪光点
- 洞悉套路：面试考察点与答题套路指导
- 模拟训练：大厂面试官1v1模拟面试
- 模拟面试复盘

打造一份让人无法拒绝的简历

- 高分简历模板分析
- 打造一份高分简历的方法
- 模拟训练：重写简历1v1指导服务

结营精华 · 职业发展规划

- 前端行业前景展望
- 高级前端开发者的晋升通道
- 周边技能的认知