加餐01｜什么是好的项目？

2021-11-29 大圣

《玩转Vue 3全家桶》 课程介绍 >

讲述：大圣

时长 12:34 大小 11.52M

你好，我是大圣，全家桶实战篇已经快更新完了。今天我特意为你准备了一个加餐，目的是想跟你聊一下什么是好的项目，以及你该如何对项目做优化，从而让它能成为一个足够好的项目。

很多同学面试的时候都会被问到：你做过什么项目？看起来很简单的一个问题，却难住了无数面试者，**因为面试官想听到的并不是你的项目流水账，而是你项目中的亮点。**这个过程其实和相亲很像，你心仪的女生问你有什么爱好，你会绞尽脑汁想突出自己的优点和亮点。然而，如果你想很好地回答项目中的亮点这个问题，需要日常工作中做出很多额外的修炼。

足够好的项目

你不用怀疑的一点是，我们在日常工作中的项目，都是足够好的项目。换句话说，这些项目至少能够完成你现在的老板、产品和用户的需求，能做好系统中的增删改查。

但是对于“足够好的项目”的标准，其实是相对而言的。为什么这么说呢？因为好项目的标准，根据你所在的工作环境和你的岗位级别不同，都会发生变化。下面，我们就来分情况讨论一下。

如果你本身就在大厂里，项目有大量的流量、效率的挑战，那只完成项目的基本功能是远远不够的。如果你还能再考虑提升一下项目运行的效率、发布部署的效率等等，那对你来说，这就是一个足够好的项目，并且已经亮点颇多了。**这也是为什么我建议你有机会一定要去大厂待两年的原因**。

如果你本身就在小公司里，那你可能就会稍微悲催一些，因为按要求去完成项目的基本功能，就算是足够好的项目了。但这样，你就会不停地被老板要求做增删改查，时间长了你就会认为这就是项目的全部。

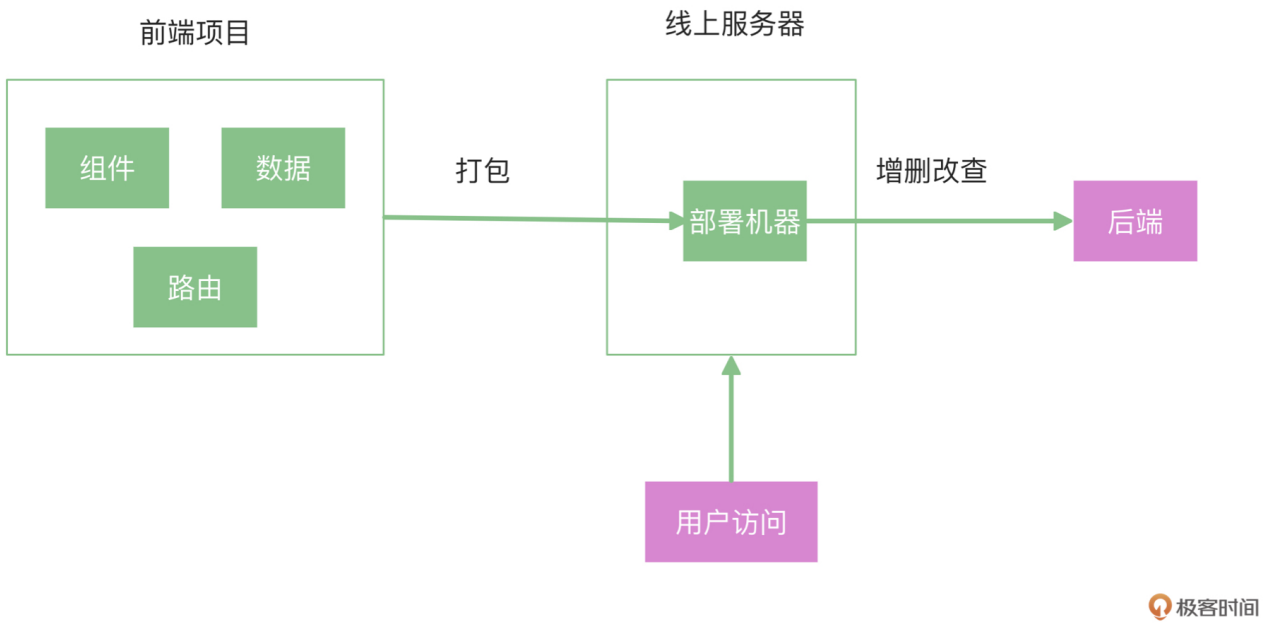
这其实能引申出大部分前端开发者工作久了的一个困惑，也是之前很多人私聊问我的一个问题：“我都工作七年了，做过二十几个 Vue 和 React 的项目，为啥百度面不过呢？”。这就涉及项目中的亮点了，也就是你能否在增删改查之外，做更多的有亮点的工作。下面，我来跟你聊一下到底什么才算是有亮点的项目。

项目中的亮点

首先，我建议你今天上班的时候，在坐到工位上后，先环顾一下周围的同事，问自己一句：“我到底做了什么他们做不到的功能？”如果你面对这个问题，没啥思路的话，那恭喜你，这篇文章正是你需要的。

面对这一个问题，你可以从手里负责的项目开始考虑，所以我们可以从项目的结构说起。下面的示意图所展示的，是在开发过程中，我们常见的前端项目的基本结构。在这种结构下，我们对项目上线后的基本功能的要求，都能得到满足。

在开发的代码中，我们使用组件 + 数据 + 路由的方式实现了项目需求，然后项目打包后部署到服务器之上。用户访问到的都是线上的代码，有增删改查的操作就会调用后端的接口实现。对于大部分中小厂来说，项目的开发只是不停地堆积页面，没有太高的复杂度。



对于项目功能的实现来说，这种结构没有问题，但这种结构是没有亮点的。当你开始考虑上图中每一个环节的优化项，当你开始思考左侧的组件如何能在多个项目复用？整体项目的性能如何优化？项目打包上线的过程如何更稳定？如何提前发现项目中的报错等等问题的时候，亮点也就随之诞生了。

我们能看到，对于一个项目来说，有很多值得优化的点。**但是，这并不意味着你需要一个人去承包所有的待优化项，我们可以根据你在项目开发中的角色来分别做讨论。**

给项目普通开发者的优化建议

如果你现在是团队内的开发者之一，那你能做的，主要还是从开发者的角度去思考现在手里负责的需求如何能够更进一步做优化，首先是需求中的数据量比变大之后如何优化，我在这里给你举两个常见的场景，相信会带给你不少启发。

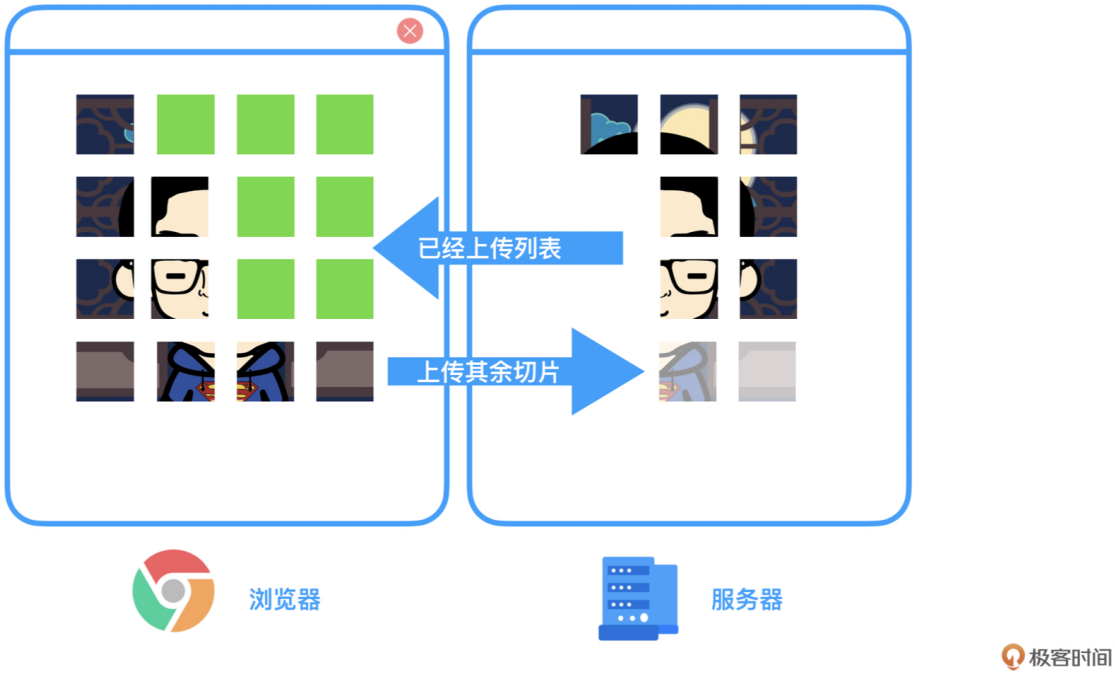
文件上传的场景

文件上传这个场景，是我很喜欢举的例子。像日常项目开发中的头像上传、用户简历上传、视频上传等等，都属于这类需求。我们直接使用 axios.post 就可以实现这个需求了，文件的体积就是这个场景下的数据量，那么文件变得很大之后，该如何处理呢？

比如，在我们上传一个 2GB 大小的视频文件的时候，如果直接使用 axios.post 上传，那么中途一旦出现网络卡顿，就需要重新上传这个视频文件。这就会对用户的体验造成不好

的影响，所以在这种数据量极大的场景下，我们需要采用断点续传的解决方案。

对照下面断点续传的示意图，你能看到，我们可以把文件切成数据块依次上传，如果上传的过程中，出现了网络错误，那么再次上传的时候，就会把已经存在的切片列表过滤掉，只上传其他的切片。



完成上述断点续传的功能之后，我们就完成了项目的初步优化，在文件上传之前，我们需要在前端计算出一个文件的 Hash 值作为唯一标识，用来向后端询问切片的列表。但是对于一个 2GB 大小的文件来说，即使是使用 MD5 算法来计算 Hash 值，也会造成浏览器的卡顿。那怎么解决计算 Hash 值时，浏览器的卡顿的问题呢？

对于卡顿问题，我们可以通过 web-workder 去解决，这有点像孙悟空可以用猴毛变出一个分身，我们这里的 hash.js，就相当于浏览器主进程的分身，用分身就可以去计算 Hash 值，不耽误主进程的任务。在下面的代码中，我们使用 new Worker 加载一个 hash.js 去计算文件的 hash 值。

复制代码

```
async calculateHashWorker(chunks) {
  return new Promise(resolve => {
    // web-worker 防止卡顿主线程
    this.worker = new Worker("/hash.js")
    this.worker.postMessage({ chunks })
    this.worker.onmessage = e => {
```

```
          const { progress, hash } = e.data
          this.hashProgress = Number(progress.toFixed(2))
          if (hash) {
            resolve(hash)
          }
        };
      });
    },
```

通过对性能瓶颈的分析，我们能看到，现在这个卡顿主要是由计算量过大导致的。在前端发展史那一讲中，我们有讲到：在 React 框架下，当项目庞大之后，如果 Diff 的计算量过大，那么也会导致卡顿。所以我们可以借鉴 React 的 Fiber 解决方案，使用浏览器的空闲时间去计算 Hash。

在下面的代码中，我们使用 requestIdleCallback 启动空闲时间的计算任务，能很好地解决这个问题。

复制代码

```
let count = 0
const workLoop = async deadline => {
  // 计算，并且当前帧还没结束
  while (count < chunks.length && deadline.timeRemaining() > 1) {
    await appendToSpark(chunks[count].file)
    count++
    // 没有了 计算完毕
    if (count < chunks.length) {
      // 计算中
      this.hashProgress = Number(
        ((100 * count) / chunks.length).toFixed(2)
      )
      // console.log(this.hashProgress)
    } else {
      // 计算完毕
      this.hashProgress = 100
      resolve(spark.end())
    }
  }
  window.requestIdleCallback(workLoop)
}
window.requestIdleCallback(workLoop)
```

这两段代码只是抛砖引玉，你还可以继续深挖这个需求，比如我们上传切片的时候，所有的文件切片一起使用 Promise.all 发起几十个 HTTP 请求，也会导致卡顿，所以我们就需

要手动管理上传任务的并发数量。

由于切片上传速度跟当前网速相关，所以在对上传任务的并发数量进行管理时，我们需要确定切片的大小。那该如何确定切片的大小呢？我们可以借鉴 TCP 协议的慢启动逻辑，去让切片的大小和当前网速匹配，这样，我们就可以通过网速确定切片的大小。

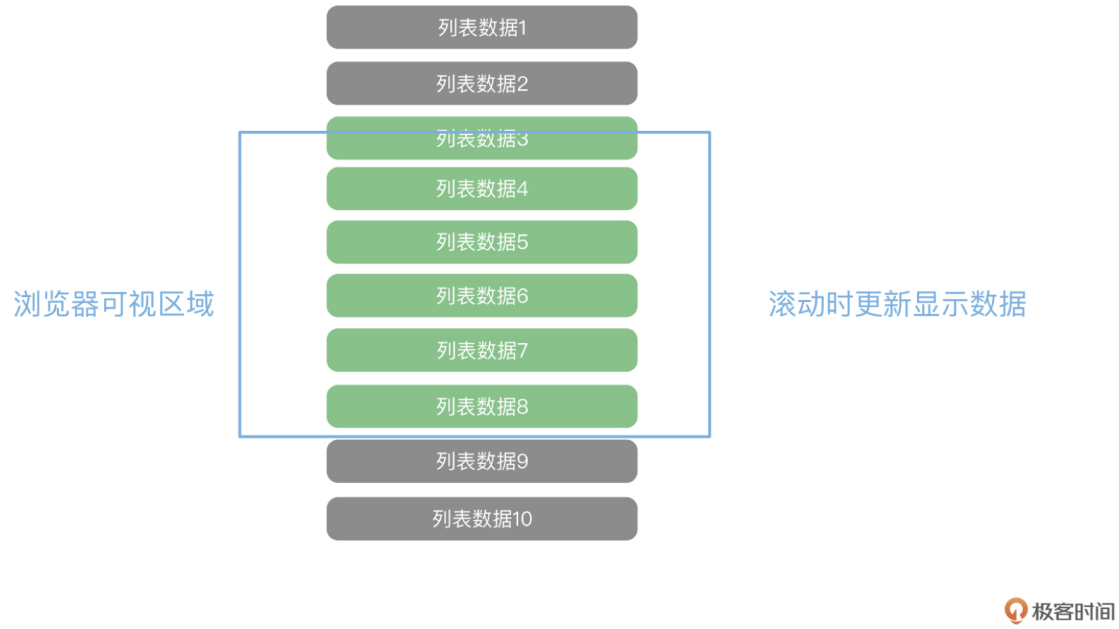
当你顺着这个思路解决了大文件上传的需求之后，这就会成为你项目中的亮点。上述文件上传的演示代码，你可以点击这里的链接，去我的 GitHub 里直接获取。

此外，还有一个典型的场景就是列表渲染，相信现在的你使用 v-for 就可以很快地实现这个需求。但是，你可以设想这样一个场景：列表的数据量不断增多，成千上万个数据的渲染让页面卡顿。按照我们优化需求的思路，需要你做的就是：使用虚拟列表来应对这个场景。

列表渲染的场景

使用虚拟列表就意味着，我们只需要渲染视图中可见的 DOM 元素，就可以实现性能优化了。看下面的示意图，我们只渲染窗口中的绿色元素，然后浏览器滚动的过程中我们维护这些 DOM，就可以避免因为页面中 DOM 元素过多，而引起的卡顿问题。

这样，在列表渲染的需求上，你的项目也就有了亮点。虚拟列表我们会在之后实现组件库的那一讲中重点讲解，这里你了解这个优化思路即可。



给项目骨干开发者的优化建议

如果你现在已经是一个小团队的负责人了，那么这个项目对你来说，具体的某个需求优化，可能已经无法成为你这个角色的亮点了。对于你所处的团队负责人这个角色来说，你更需要从项目的整体出发，去思考如何提高项目的研发效率和稳定性。

首先你会发现，一旦团队项目里多个项目之间的配置或者规范不同步，那么每个项目的配置都需要手动修改，而这很浪费时间。所以，你可以发起了一个团队的脚手架项目，把项目中的代码规范、Vite 配置，log 等等都集成在脚手架内部，通过这样的方式，可以提高项目的启动效率，这算是一个亮点。

然后，很多时候，公司多个项目之间会有代码复用和组件复用的需求。这时，你就可以再发起一个基础组件库的项目，做出一个类似 Element3 的基础组件库，并且发布在公司的 npm 服务之上，提供给全公司前端使用。为了让大家用这个组件库的时候能放心，你可以给组件库实现完备的文档系统以及超过 90% 的单测覆盖率，这也能够作为你的亮点。

前端项目的上线需要和后端服务器打交道，为了提高发布和部署的效率，你可以发起了一个 CI/CD 的项目，利用 GitHub 的 action 机制，可以把整个发布过程自动化，并且还可以一键回滚。这样日常开发的需求变更是非常快的，每一个流程的自动化都能够提高团队整体的研发效率。而且这个 CI/CD 系统里还需要能够解决需求频繁变更的问题，以及版本迭代的需求，这些优化项的解决，都能够让整个项目更稳定地交付。

你还可以复盘你现在负责的业务类型，如果你负责营销组，那么面对繁多的营销页面时，你可以搭建一个 Low Code 系统，让运营同学和产品同学自己通过拖拽的方式配置出营销页面。在这个过程中，你需要解决搭建系统时的一系列问题，比如：如何设计物料系统、如何实现跨端搭建系统等等。

然后从项目运行性能和稳定性的角度来看，我们可以制定项目的性能指标，开发项目的性能监控系统，来实时监控客户端的性能，当页面有严重的性能问题或者报错的时候，能够及时通知我们。并且除了常见的性能优化策略之外，我们还可以分析用户访问日志，提前预测用户可能访问的页面，从而做路由级别的预加载等等。

作为项目负责人来说，你要能够在整体上推动项目向前，**提高团队整体的研发效率就是你做的项目最大的亮点**。

无论是做需求还是做项目整体的优化，你都可以在晋升和面试的时候去描述你在项目中做出的亮点。所以在这一讲加餐的最后，我想跟你再聊一下如何用 STRA 原则描述你做过的项目这个问题。

其实面试官问你做过什么项目，目的就是想通过你做的项目，挖掘出你的技术亮点，**所以不要一句“我做过 XXX 项目”一闪而过，我们可以尝试使用 STAR 原则去描述项目**。所谓 STAR 原则，即 Situation（情景）、Task（任务）、Action（行动）和 Result（结果）四个英文单词的首字母组合，也就是你在什么情景下、遇见了什么任务、做了什么动作，拿到了怎样的结果，结果中最好还能带上数字展示，这样你的项目的描述就会很饱满。

相信你在自己的简历和晋升 PPT 里，一定写过和下面例子类似的项目描述，这是现在大部分简历中描述项目的方式。但比较可惜的是，这样的项目描述无法吸引面试官和评委的注意力。

> 2020-2021 在极客时间负责官网开发和后台管理系统。对此，我们可以用 STAR 原则来对你的项目描述加以优化，对比之下不难发现，下面的描述明显比上面的描述更能突出你的技术特点和个人能力。
>
> 2020-2021 在极客时间带领 3 个同事开发和维护极客时间官网的前端项目，作为核心开发者，参与了组件库的设计，XX 个组件测试覆盖率达到 80%，性能优化了 XX%。

> 2021 年 6 月至今，在极客时间负责开发极客时间后台管理系统，作为团队负责人，负责代码开发和 5 人团队的搭建，项目由 XX 和 XX 核心模块构成，通过引入 XX，提高了 XX% 的性能。

总结

今天这一讲的内容就结束了，我们来总结一下今天学习的内容。这次加餐主要想让你了解一下什么是有亮点的项目，简单来说大部分项目都只是能满足当前业务需求，当你开始思考项目的数据量、研发效率的时候，亮点就会在解决这些问题的同时诞生。

如果你还是普通开发者，那你可以从正在负责的具体需求入手，构造出一个数据量很大的场景，为了解决这个大数据量的场景，你就需要提出一些新的解决方案，比如文件上传的断点续传，列表渲染的虚拟列表等等。

如果你已经是一个项目的骨干力量，可以推动整个项目，你就可以从项目整体的运行效率和研发效率入手，我们可以推动和研发团队的脚手架、组件库、搭建系统、CI/CD 等等项目，去整体提升项目的质量，这也是非常有亮点的项目。在这一讲的最后，我们聊了如何使用 STAR 原则去描述项目，什么情景、什么任务、什么动作、什么结果，这四个维度帮助我们更立体地展现项目的亮点。

思考题

最后，再给你留一个思考题吧，你可以说说你现在负责的项目有什么亮点可以做呢？

欢迎你在留言区进行分享，也期待你在留言区立下一个做出亮点的 flag，我们下一讲再见！

上一篇 18 | 实战痛点4：Vue 3 项目中的性能优化

下一篇 19 | 实战痛点5：如何打包发布你的Vue 3应用？

精选留言 (13)

写留言

技术直男晨晨
2021-11-29

能按时开发完就谢天谢地了。。。。。

展开

9

晴空万里
2021-11-29

类似项目描述怎么写得一笔带过了 还像参考标准答案呢 哈哈 明明知道没有

共 1 条评论 >

3

费城的二鹏
2021-11-29

目前的项目，能按时开发完就谢天谢地了😭

展开

2

[让我看看]

2021-11-29

希望大圣老师能多举例场景,在小公司遇到的不多.

展开

1

润培

2021-11-29

了解了简历的写法，但是该如何量化提升了多少？

展开

1

require

2021-11-29

我建议： 某些场景可以多举例几个，比如上面的大文件上传、 大列表渲染，不用展开，只用提一下就好，小公司没遇到过这些场景， 想知道些，又没有啥门路，希望大圣老师能够多举点例子

1

ll

2021-11-29

很好的文章。有时候只关注自己做了什么，没有关注到别人"要"什么，所谓"埋头苦干"有时候也不是"优势"；对我来说"你做过什么项目？"，确实是"有点痛"，不过这篇文章后，起码有了"祛痛"的思路，手动点赞。

展开

1

阳阳

2021-11-29

如果当前的项目用其他技术栈重新写了一遍是否能作为 一个项目和技术点在简历上说呢

1

赵春艳

2021-12-03

嵌入app的H5项目：4个模块8-9个页面，基本都有：1.下拉刷新，上拉加载。2.沉浸式，导

航返回。3登录热区，跳转app登录，监听登录返回后处理各自热区业务。

木咀
2021-12-01

钱不加，工期不加，林志玲的预期……能用就行啦，哈哈——来自同事

joker
2021-12-01

是不是优化要靠平时技术积累。到时候才能有思路做出来。天天做功能也没思路做优化。没有老师的思路来做优化咋办～

展开

醉月
2021-11-29

起步有问题需要太多时间找补了。

展开

轻度
2021-11-29

干货满满，期望这样的干货越来越多

展开