**2022晋升述职材料**

1. **个人介绍**
2. 教育经历

河北大学工商学院 自动化专业

1. 过往工作经历

2018.09-2019.09 北京你我他互学数据科技股份有限公司

2017.10-2018.08 北京花韵在线科技有限公司

2015.12-2017.10 北京手工客信息技术有限公司

1. 在公司内成长经历

19年11月28号入职蛋卷，职位为高级iOS开发工程师。

1. **组织架构与工作职责**
2. 组织架构：

研发部 > 大前端 > 基金交易平台 > iOS1组 > 高级iOS开发工程师，直属领导：孙泉。

承接产品部和设计部的需求，与研发部的其它二、三级部门是合作关系。

1. 职责：

基金交易平台主要负责雪球和蛋卷客户端中基金相关业务的开发与维护，保证应用的稳定性，为用户提供良好的使用体验。我主要负责基金投前业务开发。

团队使命：不断提升技术实力，保证质量、提高效率、完善组织，进而帮助雪球业务走向成功。

团队愿景：成为让人信赖、受人尊重、令人向往的技术团队。

1. **业绩贡献**
2. 业务需求
3. 筑巢-自选基金

|  |  |
| --- | --- |
| **背景** | 作为筑巢改版的一部分，重新梳理自选在全局类目下的整体结构。 |
| **目标** | 需求：完成自选股，自选基金及自选组合的整体合并。去除自选的标的属性， 提升自选功能渗透率。  技术：梳理自选相关逻辑和依赖关系，为后面自选模块独立打下基础。 |
| **挑战** | 对雪球自选相关逻辑不熟悉，需要在原有框架基础上做改造。 |
| **行动** | 我负责自选基金部分开发，使用swift重构，移除不必要的依赖。 |
| **结果** | 需求按期完成，基金部分没有线上问题。 |
| **亮点** | 在完成需求开发的同时，考虑到工程架构演进，逐步优化。 |
| **不足** | 添加自选、取消自选动作现在写的比较分散，自选状态同步逻辑也不太完善，跟社区同事讨论优化方案，因其它需求暂时搁置。 |

1. 浮萍-基金货架页(第n次改版)

|  |  |
| --- | --- |
| **背景** | 雪球日活在140w左右，真正访问到个基页的用户日均在5w左右。用户去基金货架页往往带着明确的购买意愿，在刷了有限的标的后就离开了。 |
| **目标** | 通过调整基金货架页面card展示形式，并将货卡融入到首页推荐信息流中，希望用户决策场景由被动「货架展示」切换到「发现消费」的内容电商，吸引用户进行内容发现，并进一步完成点击-交易转化。 |
| **挑战** | 工期比较紧张，iOS客户端由7人协作开发。浮萍货卡相对比较复杂，同时需要融入到首页推荐信息流，未来还会应用在更多场景，需要降低接入成本，逻辑尽量内聚。 |
| **行动** | 我负责浮萍货卡框架开发和对外提供的能力封装。 |
| **结果** | 浮萍需求顺利上线。 |
| **亮点** | 卡片封装比较完善，后续需求接入成本低；首页浮萍货卡转化率各项数据正向，个基导流率 、加自选率、购买点击率均高于旧广告卡片。 |
| **不足** | 基金货架页购买转化率略降 0.26% -> 0.23% |

1. 基金货架页(首次改版)

|  |  |
| --- | --- |
| **背景** | 2020年7月份行情趋势比较好，雪球一级页面增加了基金tab来增加基金内容曝光。紧急上线的页面与雪球整体风格不匹配且不支持夜间模式，这个需求是做的二期优化。 |
| **挑战** | 项目时间比较紧急，由蛋卷和雪球人员联合开发，两方使用的主要开发语言不同，基础组件使用习惯也不同，在协调配合上可能会存在一些问题。最终决定由蛋卷主导开发，需要更多的考虑基础框架设计和沟通协作的问题。 |
| **行动** | 本次需求由5人协作开发，我负责基础框架搭建，网络请求、页面状态处理，顶部入口、Banner位模块开发，私募风险等级通用逻辑封装。采用了早期的Table View Section分层架构来解决模块划分和协作开发冲突问题。 |
| **结果** | 项目按期完成，个人得到了同事的认同，Table View Section分层架构积累了实践经验。 |
| **亮点** | 解决多人协作开发代码冲突的问题，提高了开发效率，其中私募风险等级通用逻辑封装在后面其它需求中也有应用。 |
| **不足** | 对蛋卷和雪球融合的功能整体规划不足，目前代码放在了蛋卷工程中，但这部分代码对蛋卷是无用的。随着战略升级，未来类似的需求和问题可能会越来越多，需要适时调整项目结构。 |

1. 思考与总结

我们的愿景是做中国人首选的在线财富管理平台，业务也正向这个方向发展，雪球和蛋卷的壁垒越来越薄弱，iOS客户端的基础建设和工程架构面临着挑战，需要不断完善去适应和助力平台化发展。

基金货架页大改版应该已经有3、4次了，目前我们正在进行新的一次改版，每次改版都需要花费大量的资源，而且灰度测试方案也不太灵活，受限于当时的资源分配需要进行妥协。期望从前端技术优化的角度，为产品提供一套更新迭代策略，基于section模块快速迭代、快速试错，灵活进行灰度测试。

1. 技术需求 - 高可用建设
2. 基金业绩曲线重构优化

|  |  |
| --- | --- |
| **背景** | 基金业绩曲线的修改一直是我们开发中的一个痛点，在设计看来很小的改动，我们没有办法快速实现，最终考虑时间成本不断做出妥协。具体原因如下：   1. 扩展性差：早期场景比较单一，没有考虑个性化定制需求。 2. 逻辑耦合严重：数据解析、UI绘制、状态维护逻辑耦合在一起，不好修改。 3. 继承关系和视图层级复杂：在众多版本中代码不断堆砌，已经很难维护。 4. 性能较差：底层使用K线图组件，有很多冗余的特性用不到，业务层封装不太合理，导致定投型组合商品页三张图切换出现卡顿。 |
| **目标** | 从以下4个方面进行优化，降低迭代开发成本：   1. 扩展：提供5个基础Renderer完成基础图表绘制，分别暴露可配置项，可以满足大部分需求场景。支持子类化或新增Renderer完成个性化定制。 2. 性能：独立的数据解析逻辑，减少数据遍历次数，引入缓存机制减少重绘次数。同时可以灵活支持不同数据结构。 3. 细节：从设计上解决快速切换数据错乱问题；解决最后一个点距离图表边缘有异常间距问题；解决长按时刷新数据选中指示器不准确问题。 4. 封装：提供业务层基础封装，满足大部分需求，同时提供定制化Demo。 |
| **挑战** | 需要平衡与业务需求开发之间的关系，没有单独申请排期。 |
| **行动** | 梳理曲线图相关文档，说明重构原因和计划，与小组成员讨论。制定重构计划：   1. 基础功能设计与开发； 2. 实现对业务层的基础封装，完善前期实现不足； 3. 逐步替换现有曲线图表。 |
| **结果** | 目前基金详情业绩曲线、净值估算、业绩球解，浮萍卡片，基金搜索结果等地方使用了新组件。 |
| **亮点** | 可以满足基金业绩曲线这一类需求的开发，个性化定制简单，提高了开发效率。 |
| **不足** | 整个重构及推广工作持续时间比较长。第二阶段跟业务需求一起进行，导致了业绩球解一个UI显示的问题，在灰度阶段发现并修复。 |

1. 异步更新RN Bundle

|  |  |
| --- | --- |
| **背景** | 雪球当前的热更新策略是仅在应用冷启动阶段进行一次热更新，且需要在第二次启动时候才生效。无法保证页面级别的热更新，如果出现一些较严重的 bug，会出现更新不及时的情况，影响用户体验甚至转化率等。这个优化作为大前端重点需求RN容器加载速度优化方案的一部分，iOS端由我负责。 |
| **目标** | 满足类似页面级别的热更新，不影响页面加载时长。 |
| **挑战** | 之前对这部分逻辑不太熟悉，bundle更新逻辑修改影响范围比较大。需要考虑未来拆包粒度更细，可以快速扩展单bundle级别的更新。 |
| **行动** | 验证优化方案可行性，梳理现有逻辑，整理技术文档并进行讲解评估，进行开发。 |
| **结果** | 目前优化方案评估已经通过，还在开发中，预计跟归巢一起上线。 |

1. 思考与总结

平衡技术优化和需求开发的关系，在需求开发的同时需要持续不断的更新优化，避免代码堆砌。重大优化需要慎重，影响范围梳理尽量全面，单独申请排期及测试资源。

1. 技术需求 - 架构演进
2. Table View Section分层架构

|  |  |
| --- | --- |
| **背景** | UI和业务逻辑复杂的页面代码量通常会比较多，在不断迭代开发中代码复杂度也会不断提升，逻辑耦合严重不易于迭代维护，多人协作困难模块职责划分不清楚，代码容易出现冲突，互相依赖难以单独测试。 |
| **目标** | 采用分层解耦的方式解决以下三个问题，提高开发效率，降低维护成本。   1. 降低代码复杂度，增加代码的可维护性 2. 提高代码聚合度，增加代码的可复用性 3. 通过组合的方式，增加代码的可扩展性，提高多人协作效率 |
| **挑战** | 从最初提出想法，到逐步完善、推广，需要不断投入精力。 |
| **行动** | 最早在指数选基详情尝试使用Section分层的架构，在后续需求开发中不断实践完善，形成了一套轻量级的架构。整理相关文档在团队内部分享。 |
| **结果** | 目前在合适的页面进行使用，iOS同事基本都理解了section分层的概念，作为一种基础开发常识，比较明显的降低了同一类型需求开发成本。 |
| **亮点** | 学习成本低，理解基本的分层概念后很容易上手；轻量、灵活，针对不同场景衍生出不同解决方案，并提炼出一些通用解决方案，比如锚点定位等。 |
| **不足** | 目前仅作为iOS客户端的代码组织方式在使用，没有把这种组合的思想扩散到其它部门，期望通过制定页面框架标准数据结构，建立section模块复用池，在APP全域内通过配置组合的方式快速搭建页面，实现多样化配置，快速迭代、快速试错，寻找出最优组合。影响产品迭代决策，避免出现类似基金货架页多次大改版的情况。 |

1. SNBRouter分离业务模块Mediator

|  |  |
| --- | --- |
| **背景** | SNBRouter目前内置在雪球工程中(历史原因)，有修改时需要在雪球提交，同时同步到SNBRouter远端仓库供蛋卷使用。业务模块对外提供的方法声明都写在SNBRouter库内，新增功能时需要频繁修改并同步。大家使用意愿不强，导致后期业务拆分和重用是非常困难。 |
| **目标** | 降低业务模块对外提供方法的成本，不用修改SNBRouter基础库，业务库自己管理方法声明和实现，版本同步管理。鼓励和规范使用Mediator，达到业务组件及时插拔的目的。 |
| **挑战** | 改变组件依赖关系，有一定的风险。 |
| **行动** | 调研实现方案，梳理依赖关系，巧妙的运用组件间依赖关系形成可行方案。 |
| **结果** | 蛋卷业务库分离出DJAPPMediator，减少了SNBRouter修改和同步的次数。 |
| **亮点** | 为中间件演进提供了一个比较完备的方案。 |
| **不足** | 由于目前雪球组件化进程停滞，现在方案也基本可以满足业务开发需要，只有蛋卷做了适配，没有推广开。 |

1. 思考与总结

客户端的架构演进在雪球平台化发展的背景下显得非常重要，也比较迫切，会影响到业务需求开发的效率和质量，以及用户体验。

1. 技术需求 - 工具化建设
2. Cocoapods-sled二进制化方案

|  |  |
| --- | --- |
| **背景** | 雪球APP工程编译时间长，断点调试等待时间长，容易引起电脑卡顿，严重影响开发效率。本地全量编译时长14分钟左右，随着使用时间增加电脑卡顿频率越来越高，编译时间也会增加。打包机平均打包时长18.5分钟左右（编译时间15分钟左右）。 |
| **目标** | 减少编译时间，降低电脑卡顿频率。  低成本，对现有开发流程无影响。 |
| **挑战** | 业内有很多二进制化方案分享，对现有开发流程影响都比较大，而且也没有成熟的开源方案（可以直接拿来用）。自己对ruby和cocoapods了解程度不深，需要边学习变开发。 |
| **行动** | 确定二进制缓存复用方案，阅读包管理工具源码，开发插件实现二进制复用。 |
| **结果** | 本地编译提升62% - 76%，电脑基本不会卡顿了。打包机提升37% - 49%。 |
| **亮点** | 轻量级、低成本，对现有开发流程无影响，源码和二进制切换方便，提交代码不会互相影响。 |
| **不足** | 插件代码细节还需要完善，还有部分遗留问题没有解决，还在不断完善中。 |

1. SNBQuick命令行工具

|  |  |
| --- | --- |
| **背景** | 雪球和蛋卷是组件化开发，不同组件代码以仓库为级别进行隔离。有时一个feature要操作好几个代码仓库，开发时需要改为本地目录，提测时需要查询组件最新的commit id，然后在壳工程中进行修改，整个过程比较繁琐，在一定程度上影响了开发效率。出于以上原因考虑团队提出了公共业务组件跨应用集成效率提升的重点项目。 |
| **目标** | 尝试通过脚本简化上述操作，并打包成命令行工具，在终端完成更多工作，不用频繁切换上下文。 |
| **挑战** | 没有相关开发经验。 |
| **行动** | 整理归纳重复繁琐的步骤，使用shell脚本代替，提供配置文件支持配置，打包成命令行工具供团队内成员使用。 |
| **结果** | 由于使用起来有一定门槛，整体团队使用度不高，后面随着工程结构变更其中部分功能不可用了。支持的功能如下：  快速切换开发环境、feature快捷提测、手动触发打包、dSYM文件上传、dSYM文件信息解析、Bugly token订阅及统计消息发送 |
| **亮点** | 熟悉后可以提高相应场景工作效率，为大前端工具化建设提供了思路，同时积累了相关经验。 |
| **不足** | 工具设计比较粗糙，推广力度不大，只在小范围内使用。由于需求开发原因没有继续完善，只添加了一些小功能。 |

1. 其它

|  |  |
| --- | --- |
| **背景** | 很多场景手动维护比较麻烦，容易出错，也容易遗忘。 |
| **目标** | 针对特定的场景，采用自动化手段解决问题。 |
| **挑战** | 踏出舒适圈，接触更多新鲜的事物，转变思维方式。 |
| **行动** | 1. dSYM自动上传脚本：每个版本打包提审时自动上传，并发送飞书消息 2. 雪球中蛋卷版本号自动同步脚本 3. Bugly前两日crash率统计，定时发送飞书消息 |
| **结果** | 简化了部分重复性工作。 |
| **亮点** | 带动大家使用自动化手段解决问题。 |
| **不足** | 比较零散，缺乏整体规划。 |

1. 思考与总结

目前做的很多工作都比较零散，缺乏整体规划。过去大多都是从个人视角出发，发现需求并提出解决方案，未来需要更多的去尝试合作，发挥团队的力量。

有时优化方案提出后，在团队内的推广不太理想，需要更积极主动的沟通，收集使用反馈。

1. **专业贡献/管理贡献**
2. 方法论沉淀：
3. 复杂的问题简单化，简单的问题标准化。
4. 每行代码都要明白为什么要写，有什么用。
5. 多用组合，少用继承。
6. 知识传播：
7. cocoapods-sled二进制化方案分享，iOS研发内部分享，23人参加
8. SNBRouter分离业务模块Mediator方案分享，iOS研发内部分享
9. 每日crash率统计实现方案分享，周会上分享，14人参加
10. Table View Sections分层架构分享，周会上分享，14人参加
11. Swift开发特性分享，2次，iOS研发内部分享，18人参加
12. 常用的高阶函数文档整理，iOS研发内部分享
13. 平时解答同事遇到的一些Swift语言特性相关的疑惑，指导新同事进行业务开发
14. 人才培养：

辅导校招生1位，已具备基础开发能力，目前主要培养分析和定位问题的能力，思考更优解决方案。

1. **自我总结与未来规划**
2. 自我总结：
3. 业务理解：熟悉基金相关业务，对客户端技术优化有一些想法，可指导新同事开发。对整体业务框架认知比较模糊。
4. 产品思维：有待加强，但也可以提出一些自己的建议。比如私募商品基础信息动态配置。
5. 技术问题解决：对常见问题有比较丰富的经验，能够快速定位问题原因，日常开发中经常帮助同事排查问题并提供解决方案。可在框架层面提供通用解决方案。技术栈深度和广度需要进一步提高。
6. 质量意识/技术规范：注重细节，如UI适配、异常逻辑处理。编程质量较高，会主动推动性能优化工作。
7. 架构/代码编程能力：有较为优秀的代码编程能力，日常业务开发中会考虑模块独立性，简单设计便于后期迭代开发。有一定的架构设计能力，给与团队一些帮助。
8. 项目管理：经验较少。
9. 未来规划：
10. 业务需求开发

多整理开发文档，建立良好的开发习惯，给同事提供更多的参考。

在需求开发中发现更多问题，提出通用解决方案，提高开发效率。

推动前端制定一些标准，解决频繁出现的问题，提高跟其它部分同事的协作效率。比如前面提到的Table View Sections分层架构 + 标准的页面框架数据结构。

1. 基础建设&研发效能

推进iOS工程组件化及工程架构演进工作，配合公司业务发展方向，提高开发效率。

推进二进制化工作，通过实践做出业内比较优秀的解决方案，提高外部影响力。

使用自动化手段解决更多问题，形成体系，切实提高开发效率。

平衡与需求开发的关系，碎片时间多思考，努力做出成果，扩大在团队中的影响力。

1. 提高表达能力

平时多做总结，提高自己的概述能力。

开发业务需求时，与各端同事积极沟通，主动推进需求开发进度。

主动推进一些技术改造工作，提高技术的同时锻炼沟通协作的能力。

通过复述练习加强说话的能力。

1. 加强技术栈积累

继续学习前端开发相关的技术，强化基础知识储备，从而提高解决问题的能力。

强化代码设计和架构能力，珍惜目前雪球业务发展带来的实践机会。

学习更多自动化相关技术，通过解决实际问题积累经验。

学习性能优化，编译优化相关知识。

**附录**

1. 分享文档

[Cocoapods-seld 二进制化方案](https://xueqiu.feishu.cn/docx/doxcnIjsYpTpT92oBJCHlCjqKUf)

[SNBRouter分离业务模块Mediator](https://xueqiu.feishu.cn/wiki/wikcnZbNJpFOwqMLYaceAJNcNnf)

[每日crash统计](https://xueqiu.feishu.cn/wiki/wikcnWRvCXkwIOc60KxiJ7u1tOg)

[Table View Sections分层架构](https://docs.snowballfinance.com/display/GFE/Table+View+Sections)

[Swift开发特性：struct](https://docs.snowballfinance.com/pages/viewpage.action?spaceKey=GFE&title=Swift+Struct)

[Swift开发特性：Struct & Class](https://docs.snowballfinance.com/pages/viewpage.action?pageId=111185724)

[常用的高阶函数](https://docs.snowballfinance.com/pages/viewpage.action?pageId=97035138)

[绘制1px线](https://docs.snowballfinance.com/pages/viewpage.action?pageId=69164228)

1. 技术优化文档

[异步更新RN Bundle包](https://xueqiu.feishu.cn/wiki/wikcnEVDYPy8p1iZtqwDdl5UgKf)

[基金详情页优化](https://xueqiu.feishu.cn/wiki/wikcnMxVkFFvUKrOxEQQDnQ6mnd)

[开户流程梳理及优化](https://xueqiu.feishu.cn/wiki/wikcnCUCCm1wMaUfUCRFW2WS0pc)

[蛋卷曲线图表重构](https://xueqiu.feishu.cn/wiki/wikcn6QUg1pxlLzXHbzgVEViq7e)

[路由映射表加载、更新逻辑优化](https://docs.snowballfinance.com/pages/viewpage.action?pageId=77993670)

[FMDB替换WCDB](https://docs.snowballfinance.com/pages/viewpage.action?pageId=69143732)

[开屏广告逻辑优化](https://docs.snowballfinance.com/pages/viewpage.action?pageId=67038323)

1. 开发需求时部分流程图





