

支付宝前后端分离 思考与实践

花名: @苏千
微博: @Python发烧友
Github: @fengmk2

支付宝体验技术部

致力于 Node.js
在中国的推广和发展

taobao npm, CNode,
cnpmjs.org, koa, urlib,
FaWave, 杭JS, 京JS, 沪JS



我们的痛点



为什么要前端分离

- 提高研发效率
- 更清晰的职责划分
- 更好的工程化: 开发, 单元测试, 自动化集成测试, 自动化UI测试
- 预期的性能提升
- 更多请查看 @赫门 [《淘宝前后端分离实践》](#)

前后端理想分工

按工作职责的前后端分工

后端

前端

服务器

浏览器

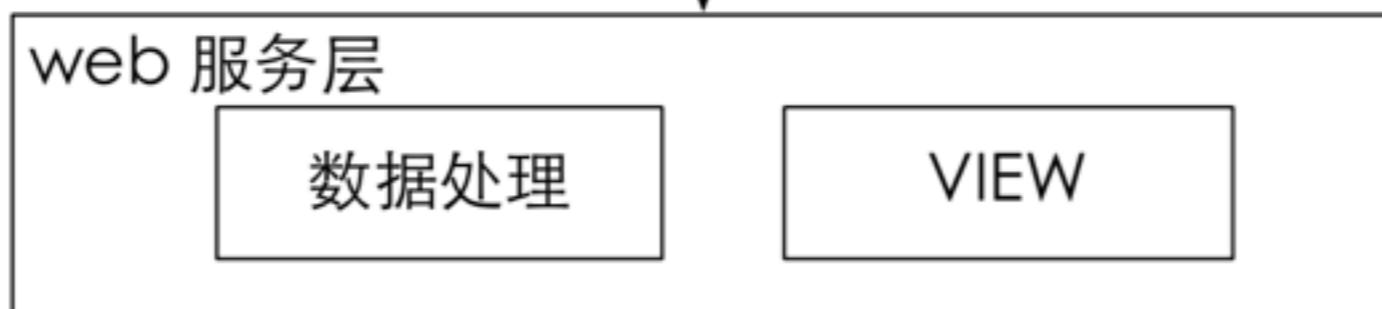
JAVA

NodeJS

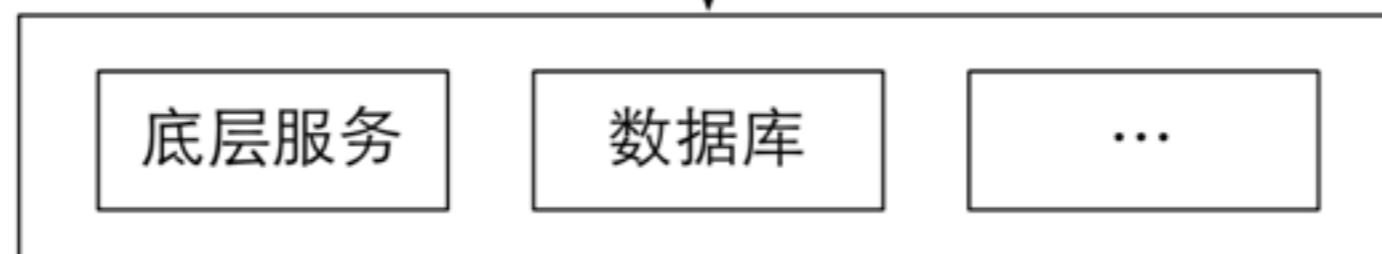
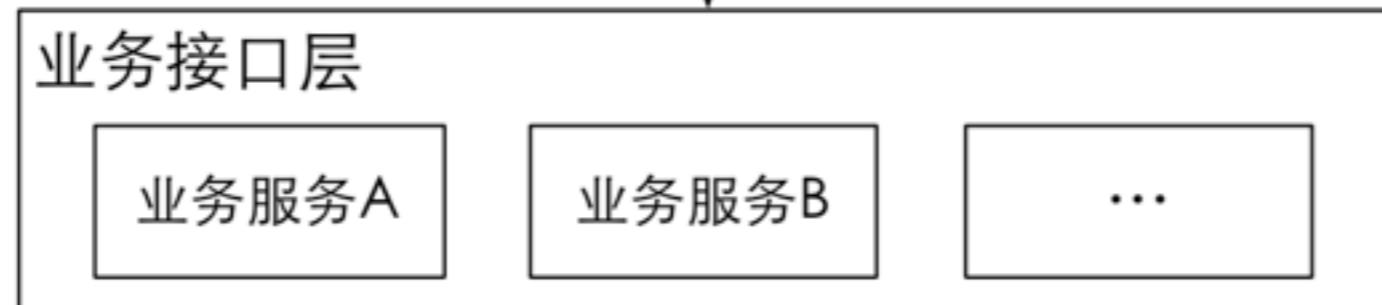
JS + HTML + CSS

- | | | |
|--|--|---|
| <ul style="list-style-type: none">• 服务层• 提供数据接口• 维持数据稳定• 封装业务逻辑 | <ul style="list-style-type: none">• 跑在服務器上的JS• 转发数据，串接服务• 路由设计，控制逻辑• 渲染页面，体验优化• 更多的可能 | <ul style="list-style-type: none">• 跑在浏览器上的JS• CSS、JS加载与运行• DOM操作• 任何的前端框架与工具• 共用模版、路由 |
|--|--|---|

大前端

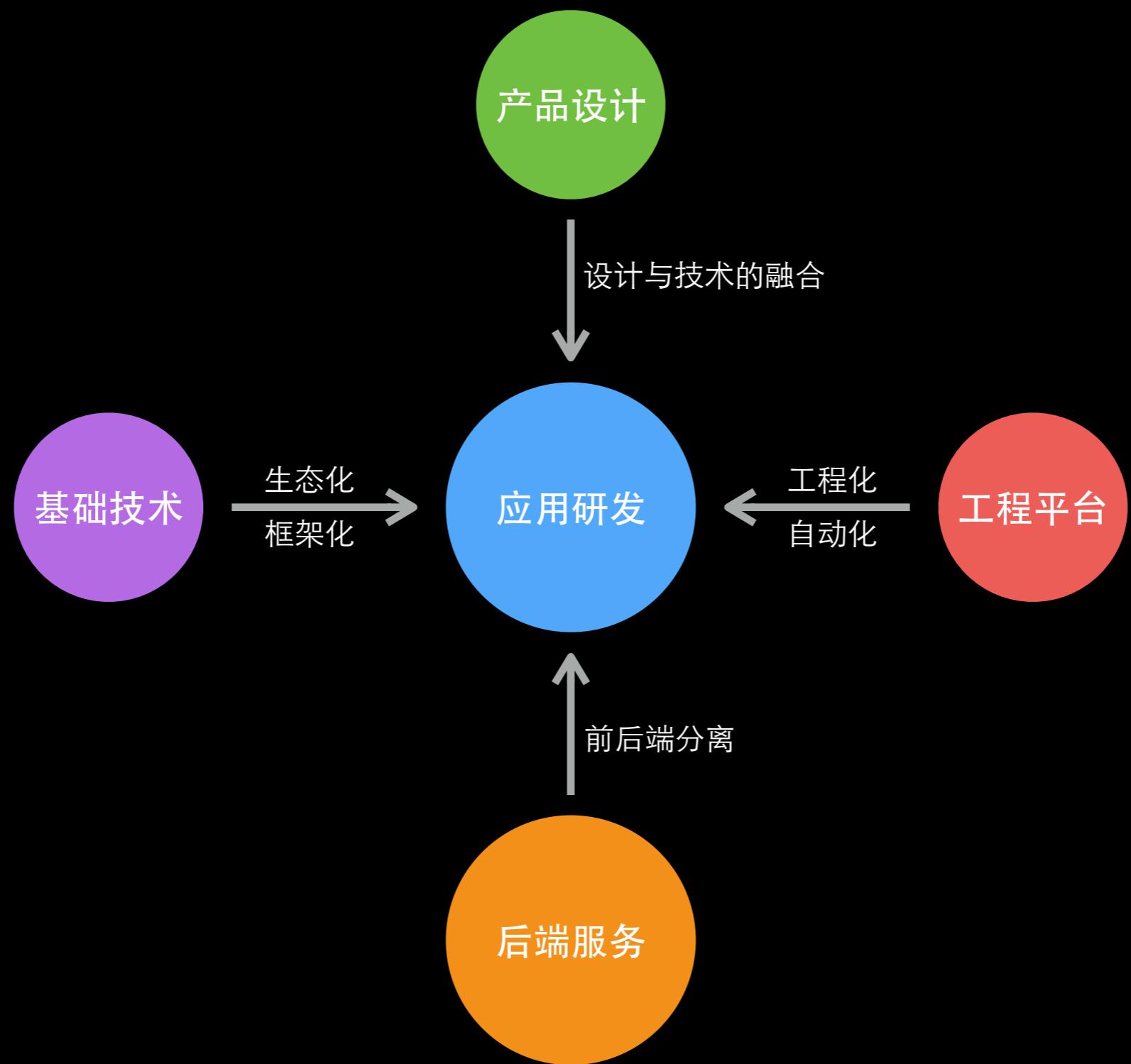


后端



大前端开发模式

我们要做的事



于是我们造了
Chair



web framework for alipay

Base on koa (Node.js)

“为什么要自己造？”

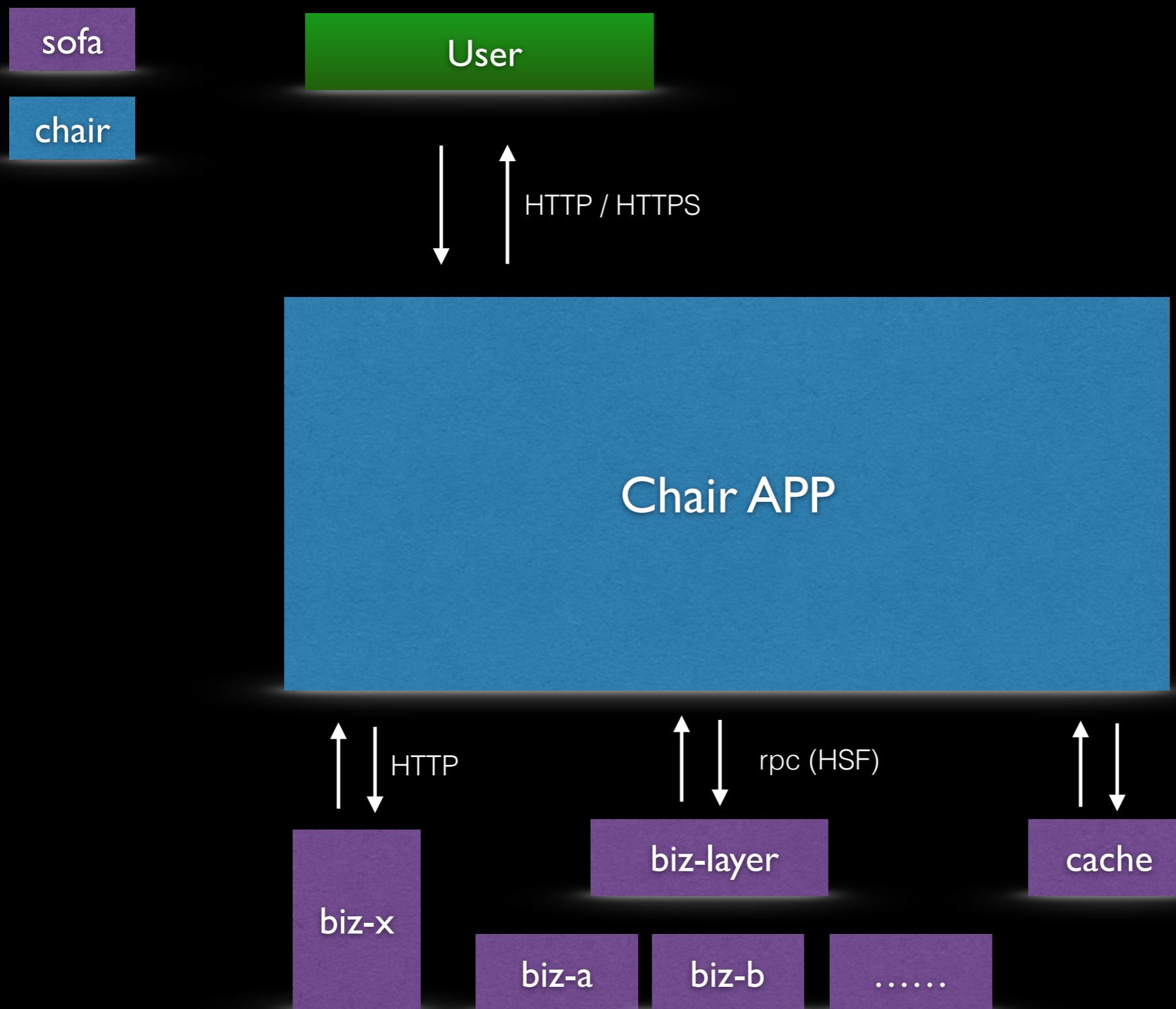
复用

- 让基于 chair 的系统尽可能复用相同逻辑
- 融合现有内部服务系统
- 统一的安全策略
- 一致的开发体验
- 开发人员一次性学习成本
- 业务系统重点在业务逻辑
- 基于 koa

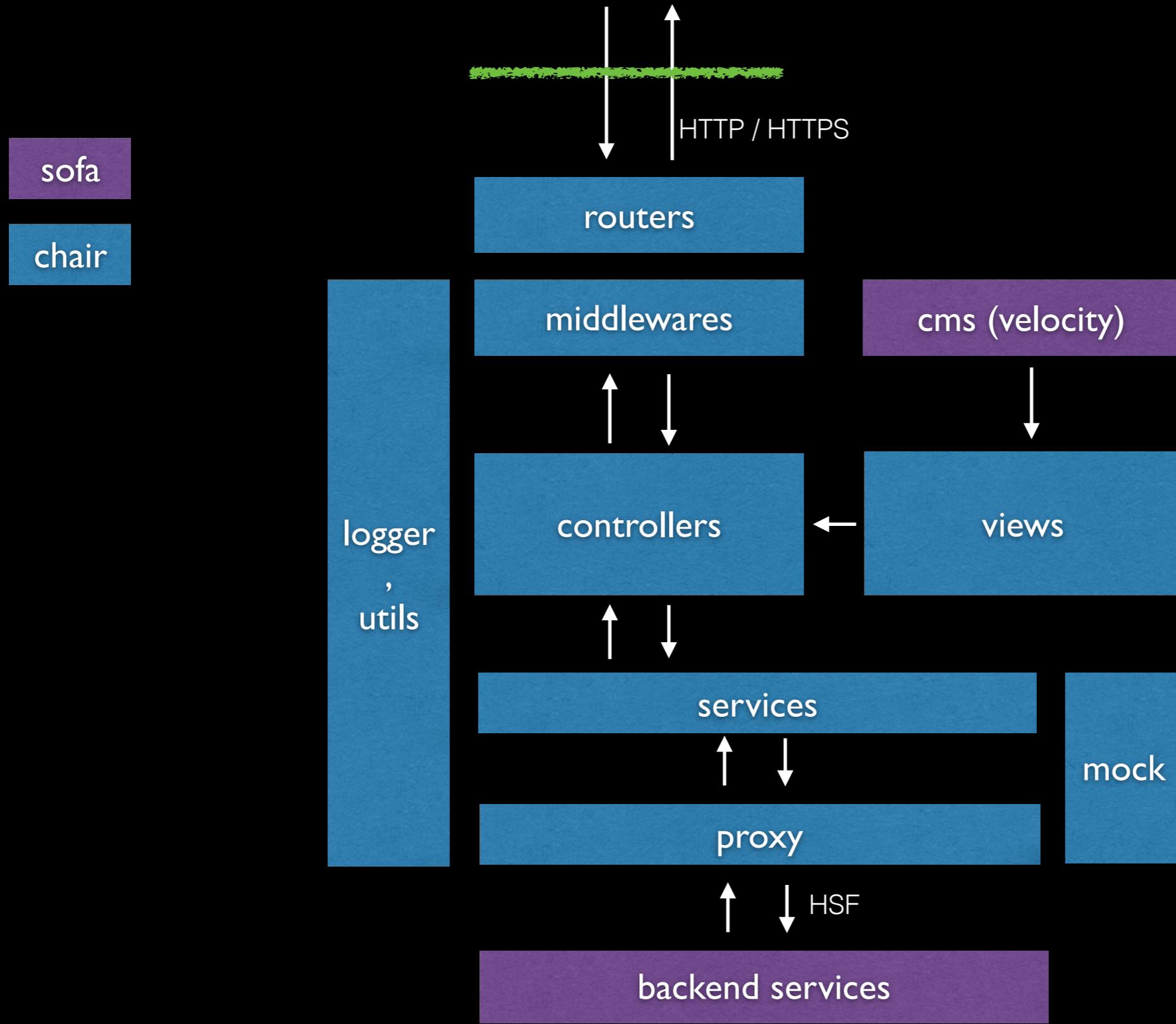
Chair 参照物

- Sofa MVC (是支付宝的Web统一编程模型，是提供给开发人员的统一Web编程界面)
- Webx (是一套基于Java Servlet API的通用Web框架)
- 其他 Java web
- HSF(High-Speed Service Framework) 一个远程调用(RPC)框架
- koa (基于 es6 generator 的 nodejs web 框架)

基于 Chair 的前后端分离架构

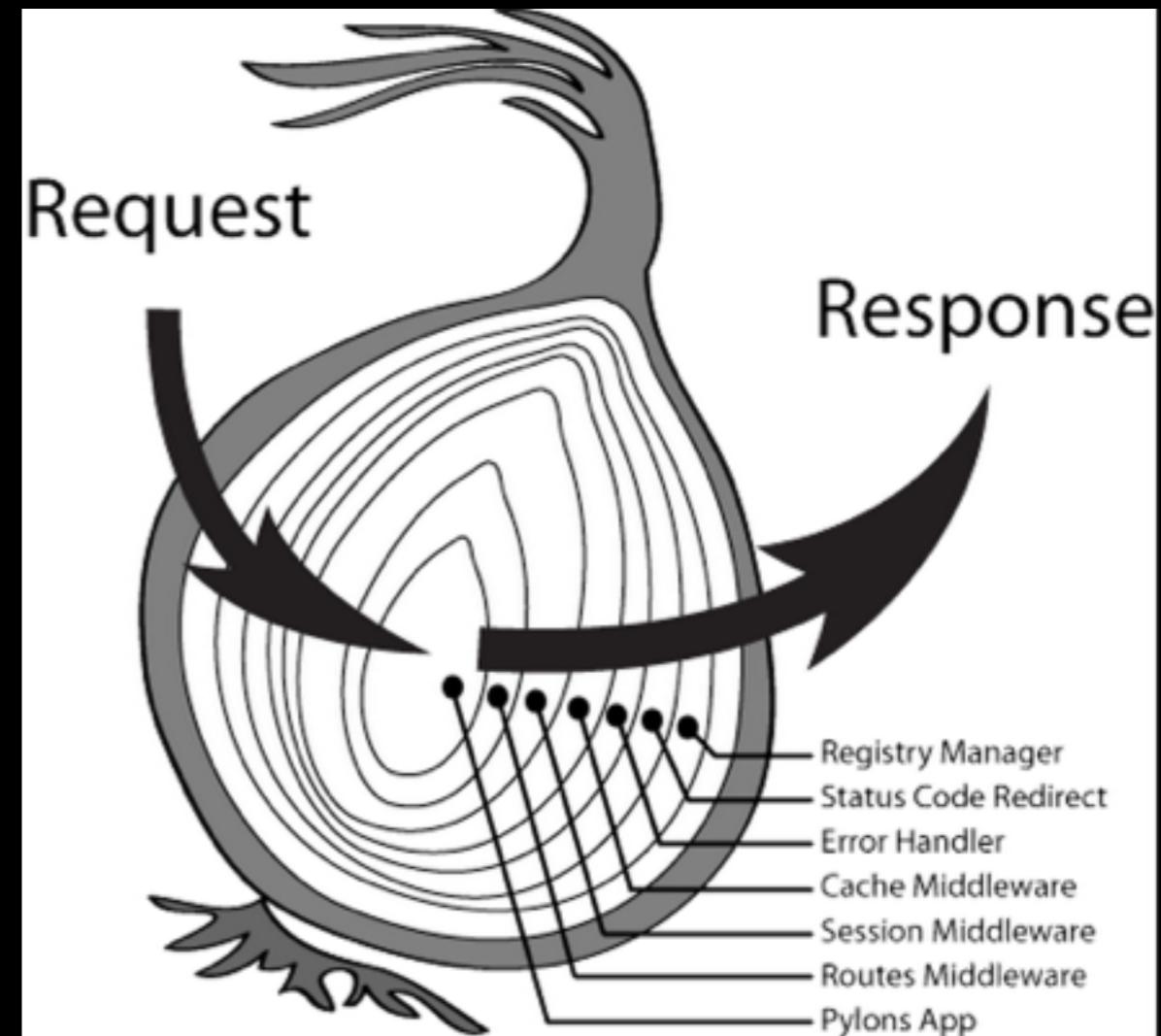


Chair 内部结构



chair 实现介绍

- Chair 基于 [koa](#) 开发，洋葱式的中间件设计，非常便于扩展。

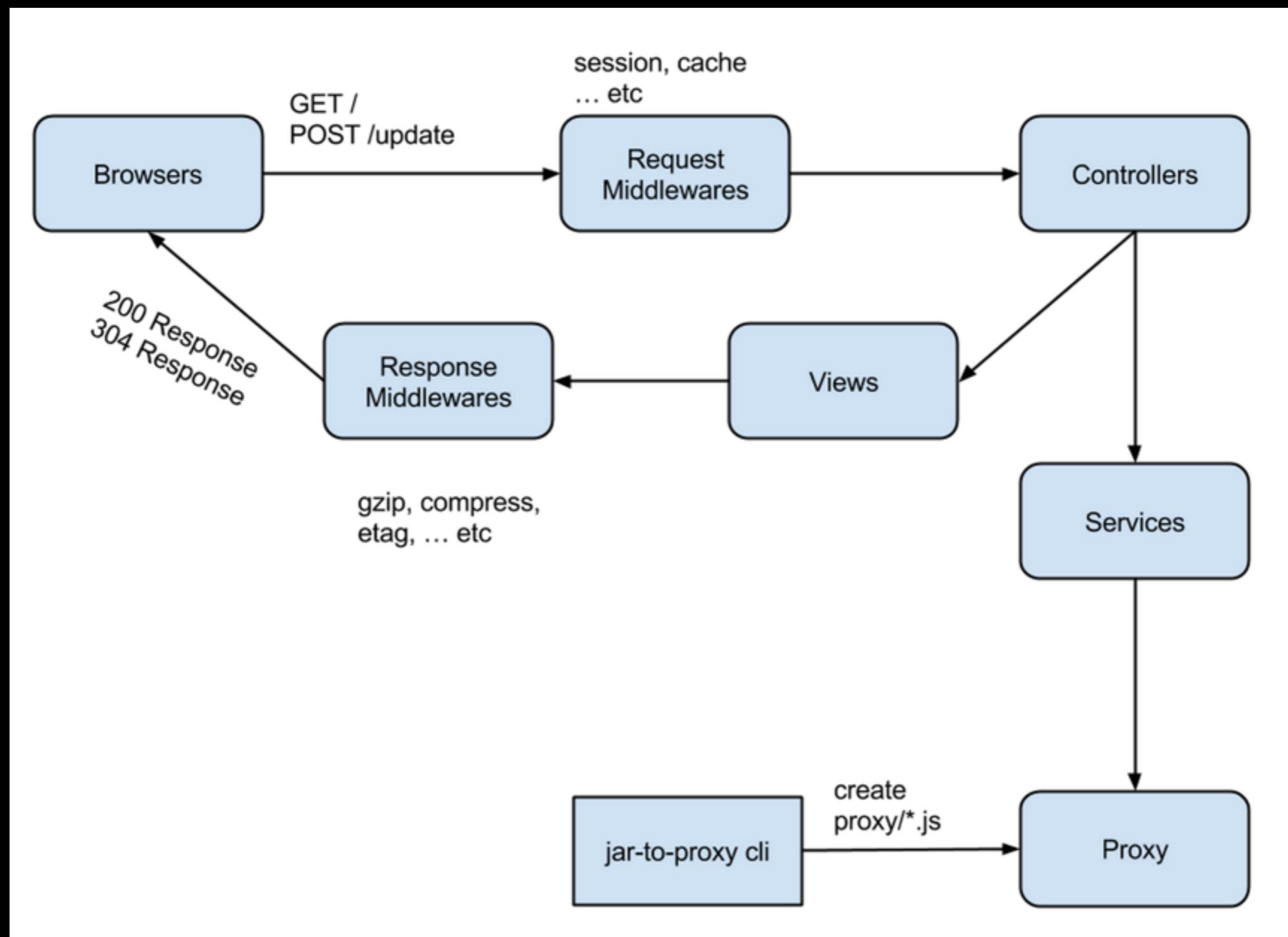


一张图展示 koa 的工作过程

```
1 var koa = require('koa');
2 var app = koa();
3
4 app.use(function* responseTime(next) {
5   var start = new Date;
6   yield next;
7   var ms = new Date - start;
8   this.set('X-Response-Time', ms + 'ms');
9 });
10
11 app.use(function* logger(next) {
12   var start = new Date;
13   yield next; ←————— 4
14   var used = new Date - start;
15   console.log('%s %s %s %sms',
16     this.method,
17     this.originalUrl,
18     this.status, used);
19 });
20
21 app.use(function* contentLength(next) {
22   yield next;
23   if (!this.body) return;
24   this.set('Content-Length', this.body.length);
25 });
26
27 app.use(function* body(next) {
28   yield next;
29   if (this.path !== '/') return;
30   this.body = 'Hello World';
31 });
32
33 app.listen(3000);
```

koa work flow

HTTP 在 chair 中的运作过程



页面模板

- 默认模板语言 [nunjucks](#): 丰富可扩展的模板引擎, mozilla 出品
- 兼容渲染 velocity (xxoo.vm): 现有绝大部分模板文件无需修改也能正常渲染
- 不支持在模板文件里面使用黑魔法远程调用服务

```
{% extends "base.html" %}

{% block header %}
<h1>{{ title }}</h1>
{% endblock %}

{% block content %}
<ul>
    {% for name, item in items %}
        <li>{{ name }}: {{ item }}</li>
    {% endfor %}
</ul>
{% endblock %}
```

完善的测试

- 单元测试
- 代码覆盖率
- 自动化集成测试
- 自动化UI测试(尝试中...)

单元测试和覆盖率

```
128 |   --require co-mocha \ lib/middlewares/_ctoken.test.js
129 |
130 ✓ should block when post without ctoken
131   ```
132
133 222 passing (14s) img01.daily.taobaocdn.net/tfscom/TB1Z2iCXXXXXXQXXXXXXXXXX.p
134 2 pending
135   ---
136
137 Writing#coverage object [/Users/mk2/git/chair/coverage/coverage.json]
138 Writing coverage reports at [/Users/mk2/git/chair/coverage]
139 使用istanbulCoverageReport命令（参见istanbulCoverageReport命令）可以一次性将单元测试跟代码覆盖率报告合为一个文件。
140
141          ----- Coverage summary -----
142 Statements : 98.05% ( 1508/1538 ), 1 ignored
143 Branches  : 93.52% ( 592/633 ), 2 ignored
144 Functions  : 97.57% ( 201/206 )
145 Lines     : 98.11% ( 1505/1534 )
146
147   --
148 CODE COVERAGE RESULT OF LINES IS: 1505/1534
149 CODE COVERAGE RESULT OF BRANCHES IS: 592/633
150 CODE COVERAGE RESULT OF FUNCTIONS IS: 201/206
151 CODE COVERAGE RESULT OF STATEMENTS IS: 1508/1538
152 CODE COVERAGE RESULT: 98.11%
153 CODE COVERAGE SUCCEEDED
154 CODE COVERAGE RESULT WAS SAVED TO: http://img01.daily.taobaocdn.net/L1/1/1/alicov/chair/0.0.16/index.html
mk2@~/git/chair (service-sal-digest-log) $
```

NODE_ENV=tes
--ha
--re
--ti
--re
\$(MO
\$(TE
.chair:v1
[chair:vie
.....
222 pass
2 pendir
mk2@~/git/

代码覆盖率

html 报表查看详细情况

Code coverage report for chair/lib/

Statements: **98.23%** (722 / 735) Branches: **95.22%** (279 / 293) Functions: **96.74%** (89 / 92) Lines: **98.23%** (722 / 735) Ignored: none

[All files](#) » chair/lib/

File	Statements	Branches	Functions	Lines
antx_config.js	100.00% (33 / 33)	100.00% (18 / 18)	100.00% (1 / 1)	100.00% (33 / 33)
chair.js	98.25% (56 / 57)	90.00% (9 / 10)	100.00% (6 / 6)	98.25% (56 / 57)
hessian_convertor.js	98.88% (88 / 89)	97.22% (35 / 36)	100.00% (9 / 9)	98.88% (88 / 89)
hsf.js	95.92% (47 / 49)	85.71% (12 / 14)	100.00% (5 / 5)	95.92% (47 / 49)
loader.js	98.35% (119 / 121)	88.89% (32 / 36)	100.00% (13 / 13)	98.35% (119 / 121)
locals.js	100.00% (15 / 15)	100.00% (4 / 4)	100.00% (4 / 4)	100.00% (15 / 15)
logger.js	100.00% (63 / 63)	100.00% (24 / 24)	100.00% (5 / 5)	100.00% (63 / 63)
mock.js	98.90% (90 / 91)	95.56% (43 / 45)	100.00% (14 / 14)	98.90% (90 / 91)
onerror.js	100.00% (5 / 5)	100.00% (2 / 2)	100.00% (1 / 1)	100.00% (5 / 5)
safe_redirect.js	100.00% (23 / 23)	100.00% (18 / 18)	100.00% (2 / 2)	100.00% (23 / 23)
session_store.js	95.83% (23 / 24)	100.00% (4 / 4)	100.00% (4 / 4)	95.83% (23 / 24)
static_server.js	100.00% (77 / 77)	100.00% (37 / 37)	100.00% (8 / 8)	100.00% (77 / 77)
status.js	85.71% (12 / 14)	75.00% (6 / 8)	66.67% (4 / 6)	85.71% (12 / 14)
util.js	96.61% (57 / 59)	95.45% (21 / 22)	91.67% (11 / 12)	96.61% (57 / 59)
watcher.js	93.33% (14 / 15)	93.33% (14 / 15)	100.00% (2 / 2)	93.33% (14 / 15)

自动化集成测试

- 测试要求集成测试必须请求真实的后端服务，不能mock
 - 使用自动化登录手段，通过测试准备好的各种用户账号，实现集成测试
 - 测试用例代码跟单元测试基本一致

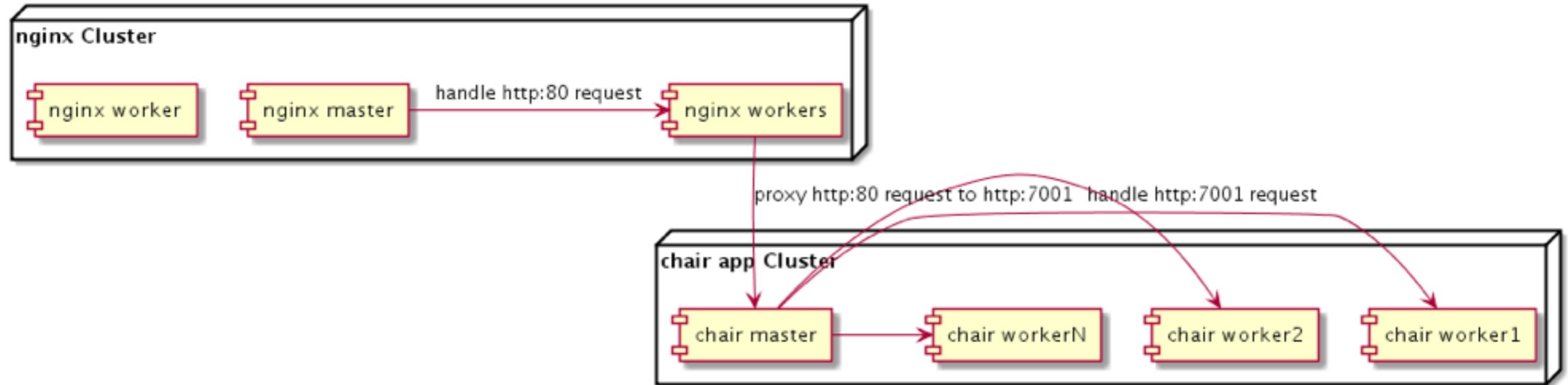
自动化UI测试

- 基于 totoro 实现跨平台多种浏览器的测试方案.
- 目前 UI 测试还在尝试中, 简单的实验性功能已经通过了, 剩下就是如何驱动浏览器行为.

```
mk2@~/git/personalweb-test (master) $ totoro list
150
151 Desktop:
152   chrome 35.0 / windows 7 [1/2]
153   firefox 29.0 / windows 7 [2/3]
154   firefox 29.0 / windows XP [2/2]
155   ie 10.0 / windows 7 [0/1] 请求真实的后端服务, 不能mock
156   ie 11.0 / windows 7 [0/1] 通过测试准备好的各种用户账号,
157   ie 6.0 / windows XP [0/1] 试基本一致
158   ie 7.0 / windows XP [0/1]
159   ie 8.0 / windows 7 [0/1] daily.taobaocdn.net/tfsc
160   ie 9.0 / windows 7 [0/1]
161   phantom 1.9 / linux 2.6 [1/2]
162   phantom 1.9 / windows XP [1/1]
163   safari 5.34 / windows 7 [1/2]
164
165 Node: 基于 [totoro](https://github.com/totorojs)
166   node 0.10 / linux 2.6 [2/2], 简单的实验性功能已经通过
167   node 0.10 / windows7 7 [2/2]
```

部署方案

- 常规 web 应用部署方式一致
- tengine + nodejs cluster

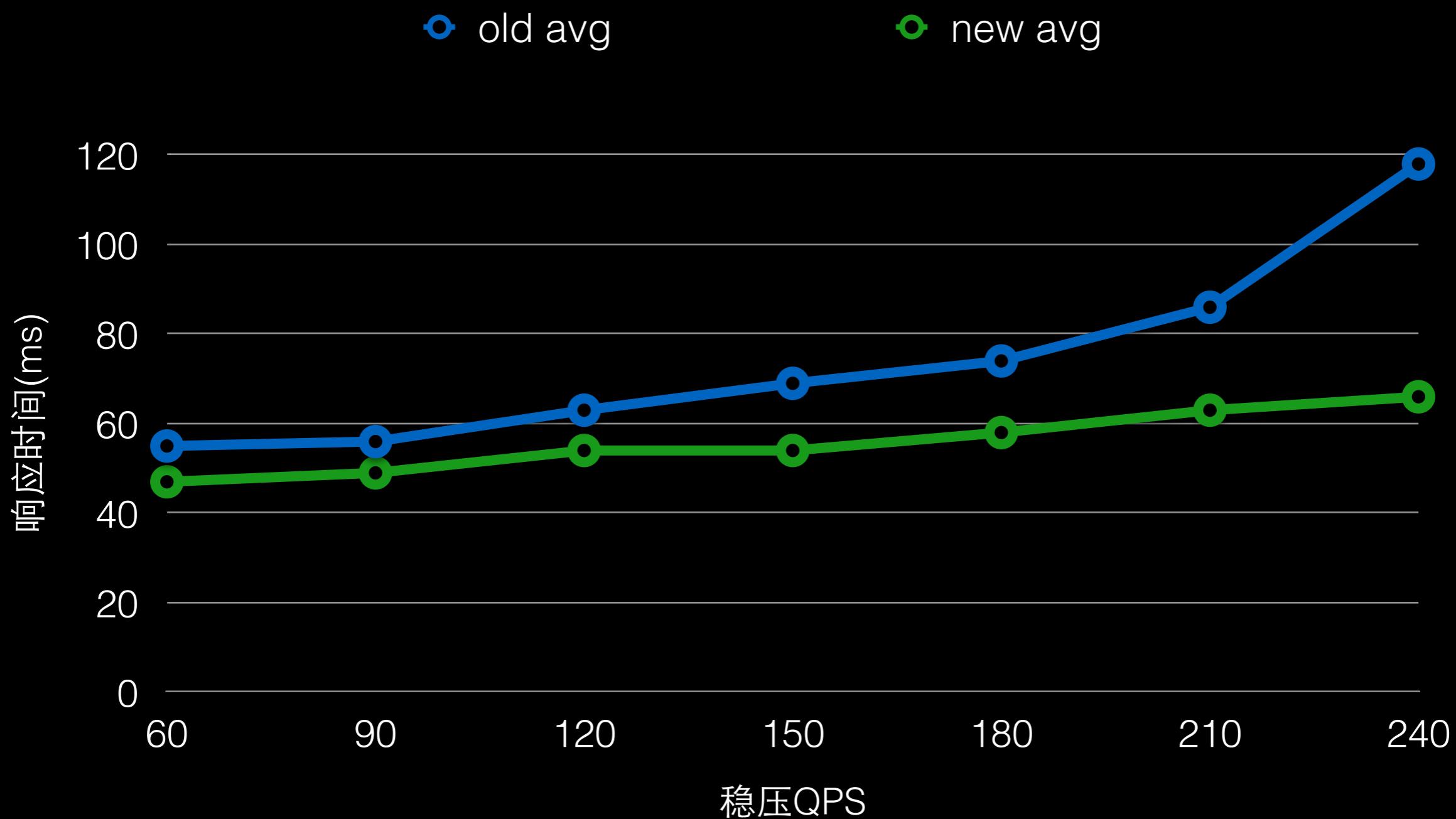


运维

- 日志: 参考 sofa 日志格式, 接入现有监控平台
- 异常报警, 适配现有运维系统
- 尽量让 PE 对 chair 的运维跟普通 Web 应用一致对待, 做到知识经验共用
- TODO: 提供 chair(nodejs) 应用运维手册

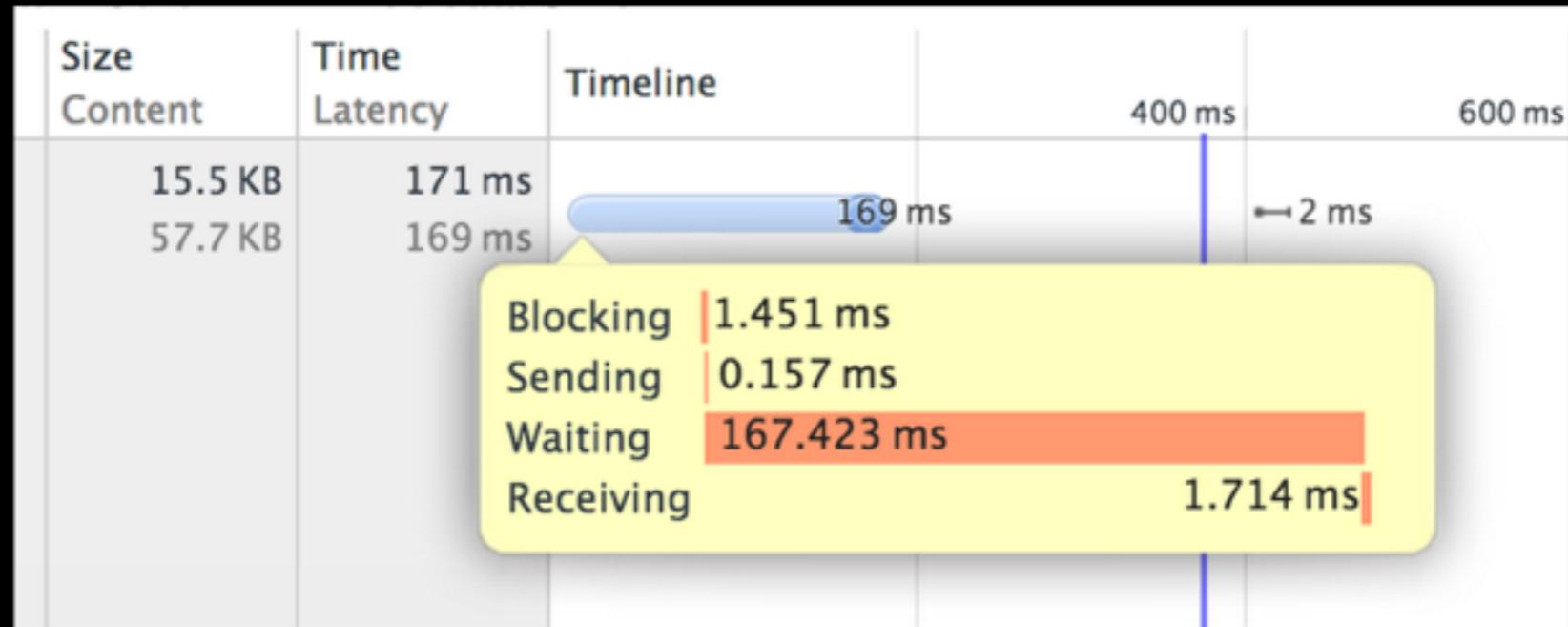
DEMO

新旧版本 rt 对比

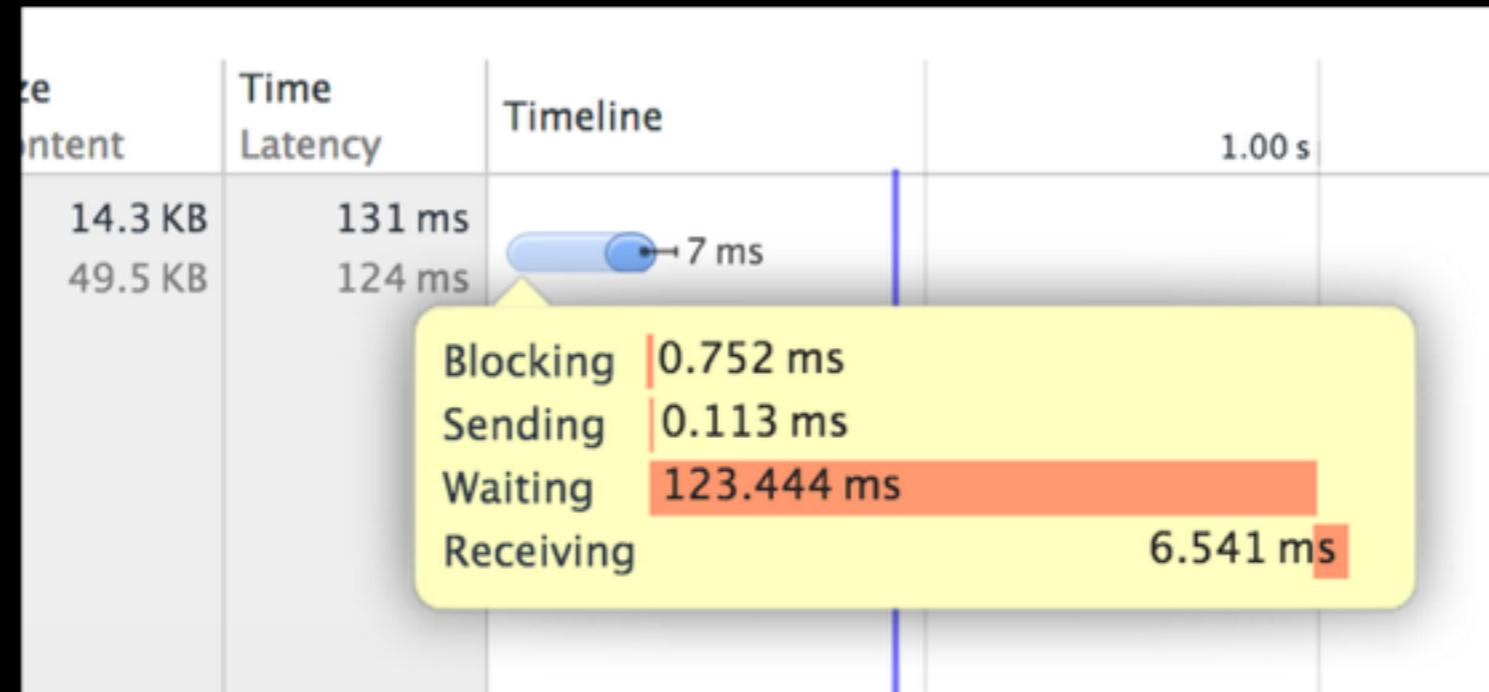


线上首屏加载 rt 对比

- my.alipay.com



- personalweb.alipay.com



性能测试总结

相同压测样本数, 并发数的场景下:

- rt 优化表现: 在 old qps 峰值的时候, new web 依然保证 rt 处于一个正常水平.
- cpu 优化表现: 在 old 单机 cpu 80% 的时候, new cpu 在 45% 以下
- new 单机能接近2倍 old 的 qps
- new + proxy cpu 等于 old cpu
- 未来移除了对 proxy 的依赖, 将能减少一半机器数

C Ȑ https://personalweb.alipay.com/portal/i.htm

ao 淘宝 NPM 镜像 personalweb dev nodejs 台湾 js git hadoop 分享到觅链 冰火鸟元数据系统

支付宝钱包 你好，袁锋 退出 我的支付宝

支付宝

我的支付宝 交易记录

账户资产 账户设置 账户通

8

凌晨好，袁锋 早点睡觉，美梦会陪伴着你

转账付款 转账到银行卡 HOT 信用卡还款 HOT 水电煤缴费 手机充值 担保付款

elements Network Sources Timeline Profiles Resources Audits Console EditThisCookie HTTPS Everywhere

Preserve log

All Documents Stylesheets Images Scripts XHR Fonts WebSockets Other Hide data URLs

Headers Preview Response Cookies Timing

Response Headers view source

Connection: keep-alive
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Date: Thu, 03 Jul 2014 16:49:46 GMT
Expires: Sun Jan 01 1995 08:00:00 GMT+0800 (CST)
Pragma: no-cache
Server: spanner/1.0.6
Set-Cookie: spanner=+R7veFVv6hoeBDJT+j3oIYMkj9fNUdr1;path=/;secure;
Strict-Transport-Security: max-age=31536000
Transfer-Encoding: chunked
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Powered-By: chair

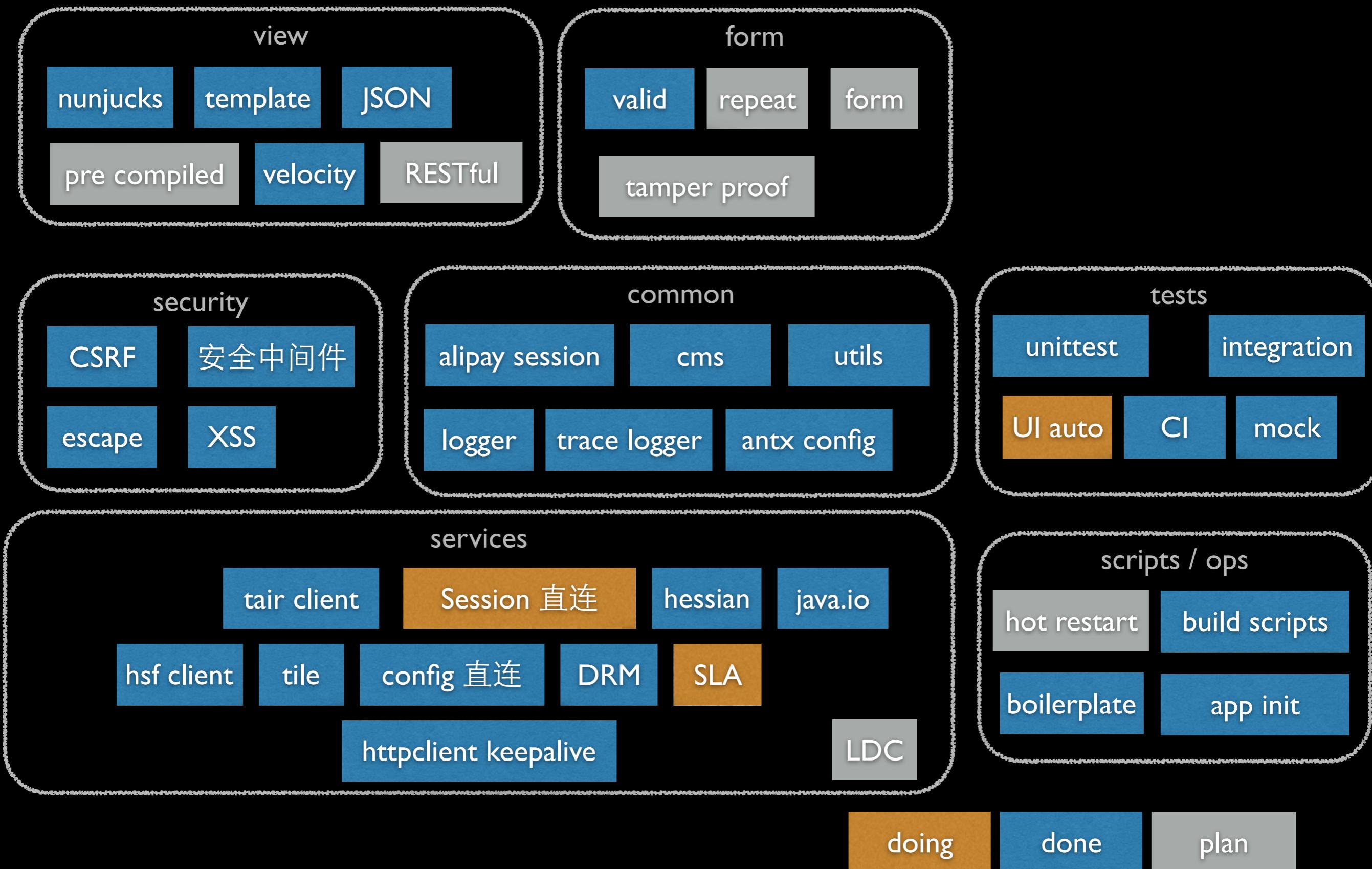
X-Readtime: 107
X-XSS-Protection: 1; mode=block

Requests | 14.3 KB / 592 KB transferred | ...

“240 QPS 也叫性能测试？
微博上都是几千上万的 QPS 啊”

因为那些测试都是
hello world

Chair 功能模块大图



前后端分离的关键点

- * 后端服务化
- 前后端团队的全力协作

yield* QA("你", "我")

Thanks! ^_^\n