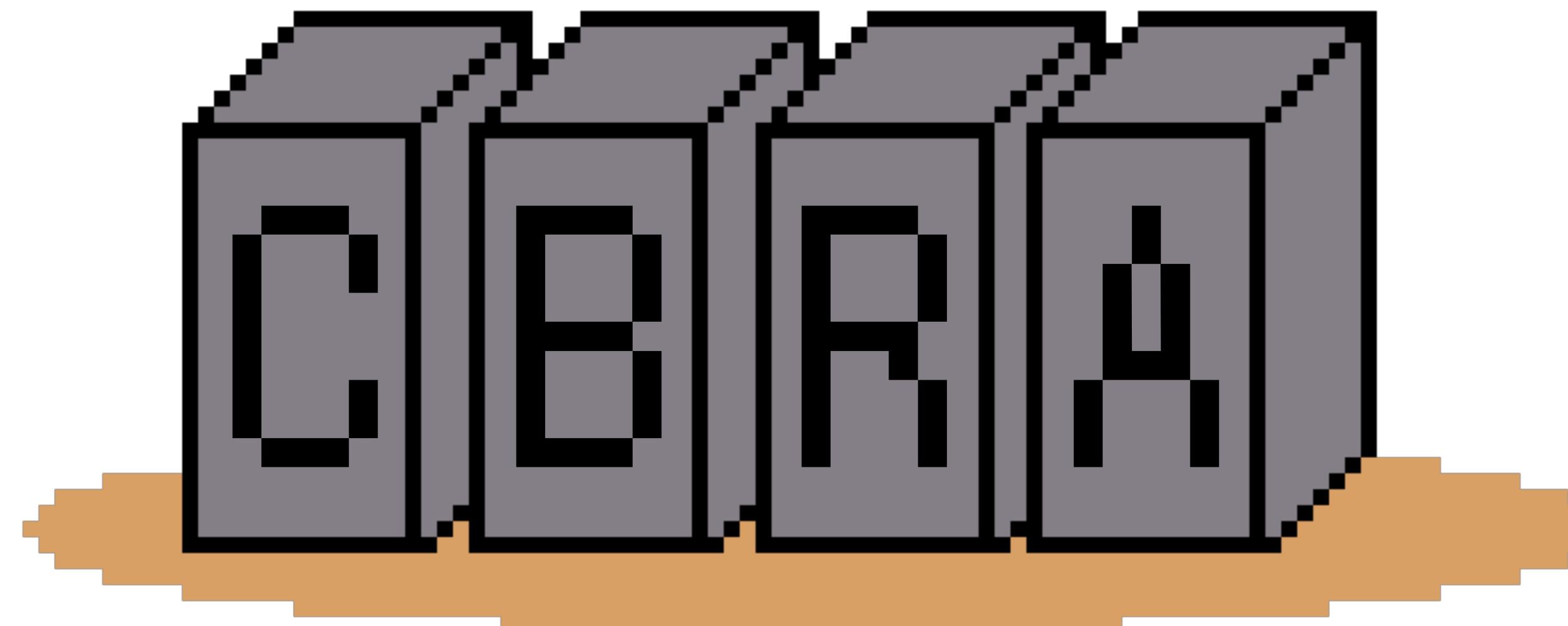


模块化的Rails，微服务以外的另一种选择

@Madao

github.com/Madao-3



自我介绍



@Madao

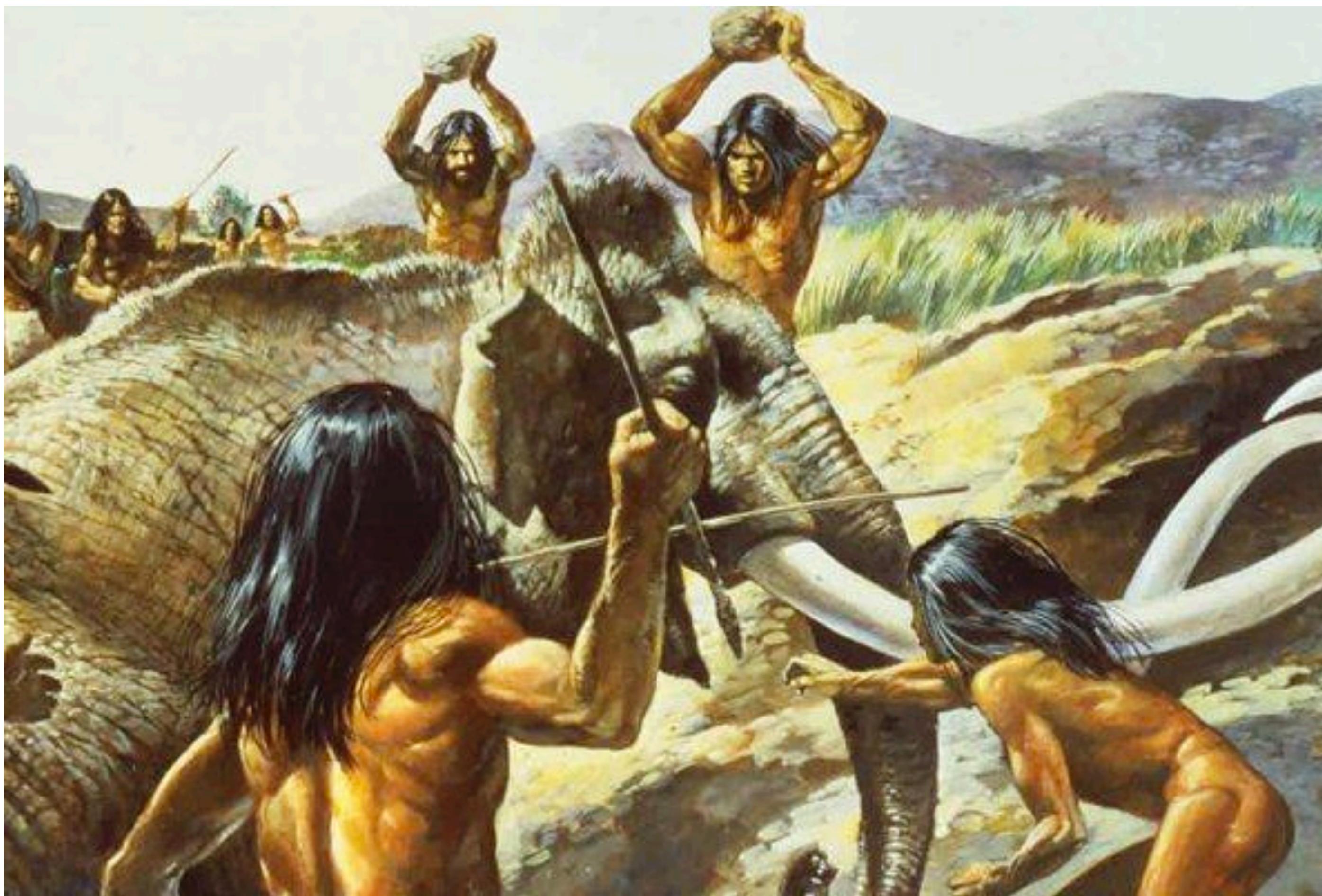
github.com/Madao-3

野生全栈工程师

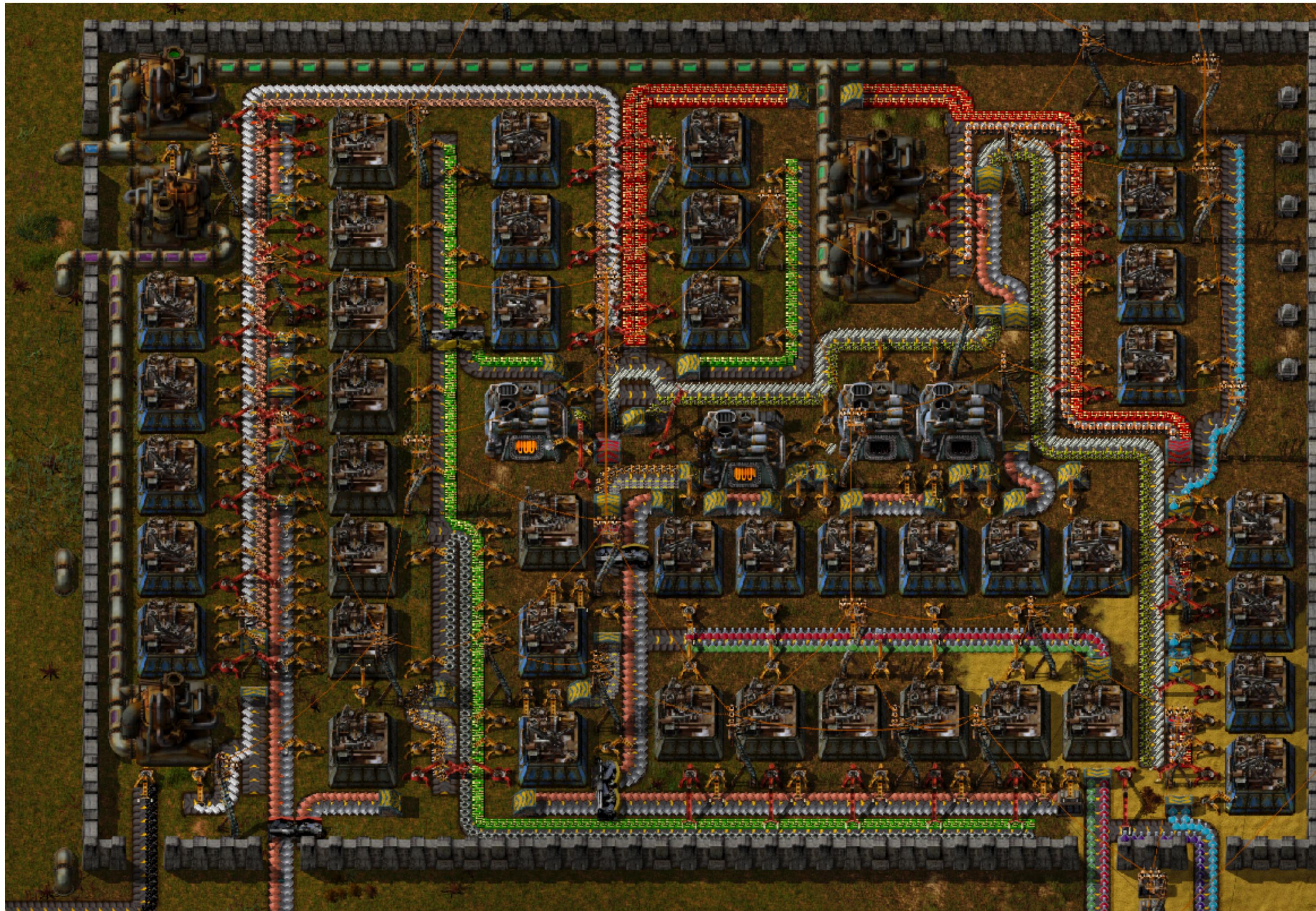
为什么不选择微服务？

单体应用优先策略

最早的时候我们是这样



我们以为项目演进项之后项目会变这样



但其实一般是这样



让我们来假设一个场景

Controller

```
email_notifications_controller.rb
```

```
1 class Admin::EmailNotificationsController < Admin::ApplicationController
2   before_action :set_email_notification, only: [:show, :edit, :update, :destroy, :send_email]
3
4   def index
5     set_seo_meta title: 'Mail Notification'
6     @builder = EmailNotification.all
7     if params[:valid_start_date].present? && params[:valid_end_date].present?
8       @builder = @builder.where("created_at between ? and ?",
9                                 Date.parse(params[:valid_start_date]),
10                                Date.parse(params[:valid_end_date]) + 1)
11    end
12    @builder = @builder.select { |e| e.country_ids.include? params[:country_id] } if params[:country_id].present?
13    @email_notifications = @builder
14  end
15
16  def new
17    set_seo_meta title: 'Create Rules'
18    @email_notification = EmailNotification.new
19  end
20
21  def create
22    @email_notification = EmailNotification.new(notification_params)
23    if @email_notification.save
24      redirect_to admin_email_notifications_path
25    else
26      render :new
27    end
28  end
29
30  def edit
31    set_seo_meta title: 'Edit Rules'
32  end
33
34  def update
35    params[:notification_params]
36    params[:country_ids] ||= []
37    params[:plan_state] ||= []
38
39    if @email_notification.update(params)
40      redirect_to admin_email_notifications_path
41    else
42      render :edit
43    end
44  end
45
46  def send_email
47    plan_params = {
48      country_ids: @email_notification.country_ids,
49      state: @email_notification.plan_state,
50      start_date: @email_notification.out_start_date.to_s,
51      end_date: @email_notification.out_end_date.to_s,
52    }
53
54    recipients = PlanService.get_plans(plan_params, current_staff)
55    .map { |u| [u.planner, u.manager, u.advisor] }
56    .flatten.compact.uniq.map(&:email)
57
58    recipients.each do |recipient|
59      Mailer.send_email_notification(recipient, @email_notification)
60    end
61  end
62
63  private
64
65  def set_email_notification
66    @email_notification = EmailNotification.find(params[:id])
67  end
68
69  def notification_params
70    params.require(:notification).permit(
71      :country_ids, :plan_state, :out_start_date, :out_end_date
72    )
73  end
74end
```

Model

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9end
```

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9end
```

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9end
```

```
form.html.erb
```

```
<div class="row">
<div class="col-md-12">
  <form>
    <div>
      <label>Country</label>
      <div>
        <select name="email_notification[country_id]" type="select2" style="width: 500px;" placeholder="选择国家">
          <option value=""></option>
          <option value="1">中国</option>
          <option value="2">美国</option>
          <option value="3">日本</option>
          <option value="4">韩国</option>
          <option value="5">其他国家</option>
        </select>
      </div>
    </div>
    <div>
      <label>计划状态</label>
      <div>
        <select name="email_notification[plan_state]" type="select2" style="width: 500px;" placeholder="选择计划状态">
          <option value=""></option>
          <option value="1">待审核</option>
          <option value="2">已审核</option>
          <option value="3">已通过</option>
          <option value="4">已驳回</option>
        </select>
      </div>
    </div>
    <div>
      <label>开始日期</label>
      <div>
        <input type="text" class="form-control" name="email_notification[out_start_date]" type="text" placeholder="开始日期" />
      </div>
    </div>
    <div>
      <label>结束日期</label>
      <div>
        <input type="text" class="form-control" name="email_notification[out_end_date]" type="text" placeholder="结束日期" />
      </div>
    </div>
    <div>
      <label>接收人</label>
      <div>
        <input type="text" class="form-control" name="email_notification[recipients]" type="text" placeholder="接收人" />
      </div>
    </div>
  </form>
</div>
</div>
```

```
form.html.erb
```

```
<div class="row">
<div class="col-md-12">
  <form>
    <div>
      <label>Country</label>
      <div>
        <select name="email_notification[country_id]" type="select2" style="width: 500px;" placeholder="选择国家">
          <option value=""></option>
          <option value="1">中国</option>
          <option value="2">美国</option>
          <option value="3">日本</option>
          <option value="4">韩国</option>
          <option value="5">其他国家</option>
        </select>
      </div>
    </div>
    <div>
      <label>计划状态</label>
      <div>
        <select name="email_notification[plan_state]" type="select2" style="width: 500px;" placeholder="选择计划状态">
          <option value=""></option>
          <option value="1">待审核</option>
          <option value="2">已审核</option>
          <option value="3">已通过</option>
          <option value="4">已驳回</option>
        </select>
      </div>
    </div>
    <div>
      <label>开始日期</label>
      <div>
        <input type="text" class="form-control" name="email_notification[out_start_date]" type="text" placeholder="开始日期" />
      </div>
    </div>
    <div>
      <label>结束日期</label>
      <div>
        <input type="text" class="form-control" name="email_notification[out_end_date]" type="text" placeholder="结束日期" />
      </div>
    </div>
    <div>
      <label>接收人</label>
      <div>
        <input type="text" class="form-control" name="email_notification[recipients]" type="text" placeholder="接收人" />
      </div>
    </div>
  </form>
</div>
</div>
```

Views

```
new.html.erb
```

```
1 <h3>Add New Rule
2 <button type="button" class="btn btn-info check-tips">How ?</button>
3
4 <%= render 'form' %>
```

回忆一下常规单体开发的历程

Controller

```
email_notifications_controller.rb
```

```
1 class Admin::EmailNotificationsController < Admin::ApplicationController
2   before_action :set_email_notification, only: [:show, :edit, :update, :destroy, :send_email]
3
4   def index
5     set_seo_meta title: 'Mail Notification'
6     @builder = EmailNotification.all
7     if params[:valid_start_date].present? && params[:valid_end_date].present?
8       @builder.where("created_at between ? and ?", Date.parse(params[:valid_start_date]), Date.parse(params[:valid_end_date]) + 1)
9     end
10    @builder = @builder.select { |e| e.country_ids.include? params[:country_id] } if params[:country_id].present?
11    @email_notifications = @builder
12  end
13
14  def new
15    set_seo_meta title: 'Create Rules'
16    @email_notification = EmailNotification.new
17  end
18
19  def create
20    @email_notification = EmailNotification.new(notification_params)
21    if @email_notification.save
22      redirect_to admin_email_notifications_path
23    else
24      render :new
25    end
26  end
27
28  def edit
29    set_seo_meta title: 'Edit Rules'
30  end
31
32  def update
33    params[:notification_params]
34    params[:country_ids] ||= []
35    params[:plan_state] ||= []
36
37    if @email_notification.update(params)
38      redirect_to admin_email_notifications_path
39    else
40      render :edit
41    end
42  end
43
44  def send_email
45    plan_params = {
46      country_ids: @email_notification.country_ids,
47      state: @email_notification.plan_state,
48      start_date: @email_notification.out_start_date.to_s,
49      end_date: @email_notification.out_end_date.to_s,
50    }
51
52    recipients = PlanService.get_plans(plan_params, current_staff)
53    .map { |u| [u.planner, u.manager, u.advisor] }
54    .flatten.compact.uniq.map(&:email)
55
56    recipients.each do |recipient|
57      Mailer.send_email_notification(recipient, @email_notification)
58    end
59  end
60
61  private
62
63  def notification_params
64    params.require(:notification).permit(
65      :country_ids,
66      :plan_state,
67      :out_start_date,
68      :out_end_date,
69      :recipients
70    )
71  end
72
73  def set_email_notification
74    @email_notification = EmailNotification.find(params[:id])
75  end
76
77  def destroy
78    @email_notification.destroy
79  end
80
81  def set_seo_meta
82    seo_meta_for_email_notification(@email_notification)
83  end
84
85  def seo_meta_for_email_notification(email_notification)
86    seo_meta_for_email_notification_email_notification(email_notification)
87  end
88
89  def seo_meta_for_email_notification_email_notification(email_notification)
90    seo_meta_for_email_notification_country_ids(email_notification)
91  end
92
93  def seo_meta_for_email_notification_country_ids(email_notification)
94    seo_meta_for_email_notification_plan_state(email_notification)
95  end
96
97  def seo_meta_for_email_notification_plan_state(email_notification)
98    seo_meta_for_email_notification_out_start_date(email_notification)
99  end
100 end
```

Model

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9 end
```

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9 end
```

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9 end
```

Views

```
new.html.erb
```

```
1 <h3>Add New Rule
2 <button type="button" class="btn btn-info check-tips">How ?</button>
3
4 <%= render 'form' %>
```

```
email_notification.rb
```

```
1 class EmailNotification < ActiveRecord::Base
2   serialize :country_ids, Array
3   serialize :plan_state, Array
4
5   scope :plan_submitted, -> { where(plan_submit: 1) }
6   scope :guidebook_submitted, -> { where(guidebook_submit: 1) }
7   scope :deposit_contracted, -> { where(deposit_contract: 1) }
8
9 end
```

```
form.html.erb
```

```
<div class="row">
<div class="col-md-12">
  <form>
    <div>
      <label>Country</label>
      <select name="email_notification[country_id]" class="form-control select2" style="width: 100px;" placeholder="选择国家">
        <option value=""></option>
        <option value="1">中国</option>
        <option value="2">美国</option>
        <option value="3">日本</option>
        <option value="4">韩国</option>
        <option value="5">澳大利亚</option>
        <option value="6">新西兰</option>
        <option value="7">其他国家</option>
      </select>
    </div>
    <div>
      <label>计划状态</label>
      <select name="email_notification[plan_state]" class="form-control select2" style="width: 100px;" placeholder="选择状态">
        <option value=""></option>
        <option value="1">未开始</option>
        <option value="2">进行中</option>
        <option value="3">完成</option>
      </select>
    </div>
    <div>
      <label>开始日期</label>
      <input type="text" class="form-control" name="email_notification[out_start_date]" value="2018-01-01" placeholder="选择开始日期" style="width: 100px;" />
    </div>
    <div>
      <label>结束日期</label>
      <input type="text" class="form-control" name="email_notification[out_end_date]" value="2018-01-02" placeholder="选择结束日期" style="width: 100px;" />
    </div>
    <div>
      <label>收件人</label>
      <input type="text" class="form-control" name="email_notification[recipients]" value="1234567890" placeholder="输入收件人" style="width: 100px;" />
    </div>
  </form>
</div>
</div>
```

```
form.html.erb
```

```
<div class="row">
<div class="col-md-12">
  <form>
    <div>
      <label>Country</label>
      <select name="email_notification[country_id]" class="form-control select2" style="width: 100px;" placeholder="选择国家">
        <option value=""></option>
        <option value="1">中国</option>
        <option value="2">美国</option>
        <option value="3">日本</option>
        <option value="4">韩国</option>
        <option value="5">澳大利亚</option>
        <option value="6">新西兰</option>
        <option value="7">其他国家</option>
      </select>
    </div>
    <div>
      <label>计划状态</label>
      <select name="email_notification[plan_state]" class="form-control select2" style="width: 100px;" placeholder="选择状态">
        <option value=""></option>
        <option value="1">未开始</option>
        <option value="2">进行中</option>
        <option value="3">完成</option>
      </select>
    </div>
    <div>
      <label>开始日期</label>
      <input type="text" class="form-control" name="email_notification[out_start_date]" value="2018-01-01" placeholder="选择开始日期" style="width: 100px;" />
    </div>
    <div>
      <label>结束日期</label>
      <input type="text" class="form-control" name="email_notification[out_end_date]" value="2018-01-02" placeholder="选择结束日期" style="width: 100px;" />
    </div>
    <div>
      <label>收件人</label>
      <input type="text" class="form-control" name="email_notification[recipients]" value="1234567890" placeholder="输入收件人" style="width: 100px;" />
    </div>
  </form>
</div>
</div>
```

Controller

Model



需要解决的问题

- 1.降低单体应用的复杂度
- 2.避免过慢的CI
- 3.便于生成依赖关系图
- 4.减少实现过程中所需的上下文内容
- 5.快速交付

那么我们有什么办法呢？

首先要保证能够持续高效地交付，这时候你会发现继续维持原有的单体架构是不可行的。你需要拆分它，降低它的复杂度，并且确定拆分的边界。

那还不是微服务嘛

如果我们可以把每一个拆分出来的子 Rails 应用当做**Gem** 呢？

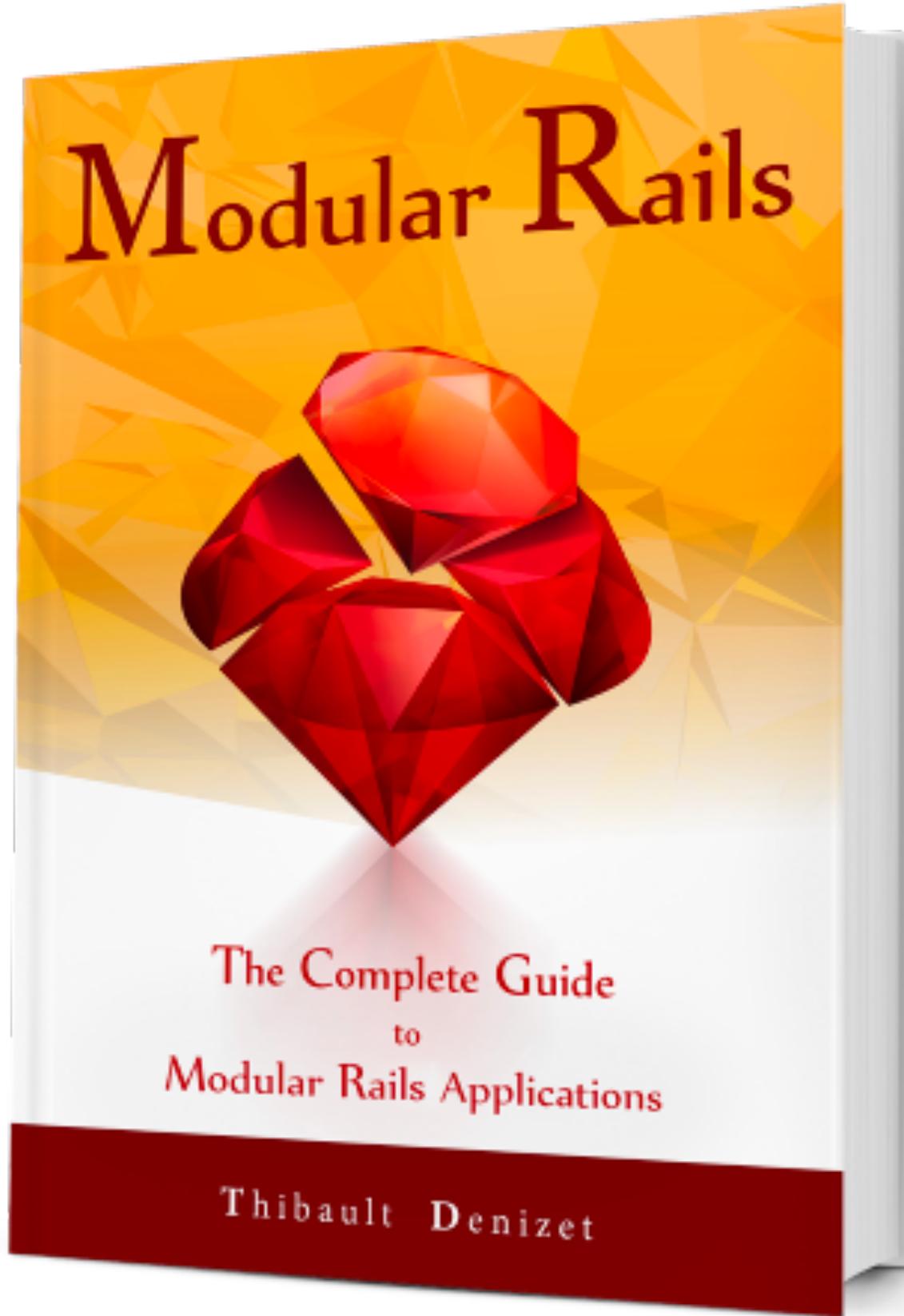


Component-Based Rails Applications



旧版本：<http://shageman.github.io/cbra.info/>

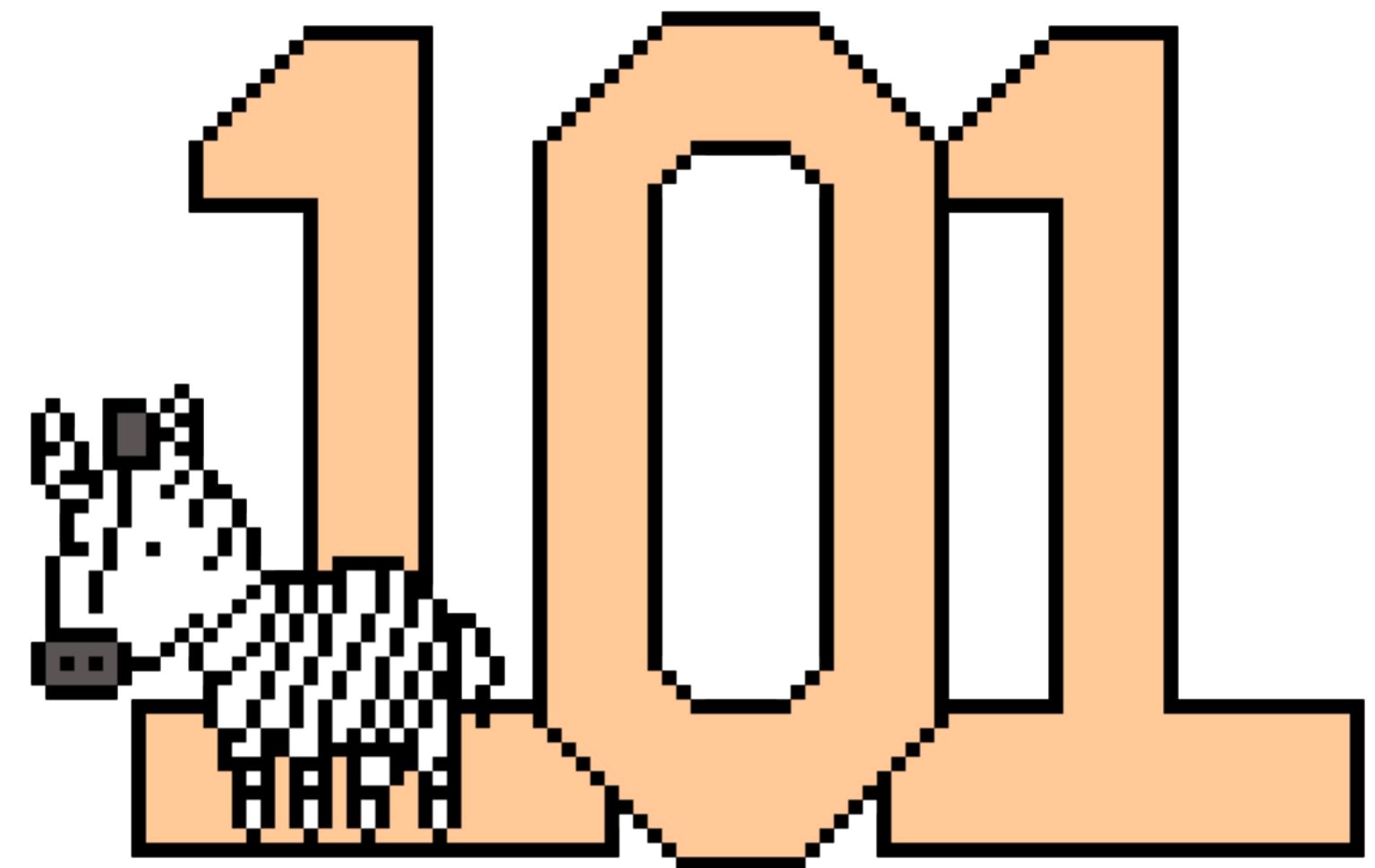
@shageman



<http://modular-rails.samurails.com/>

@T_Dnzt

让我们实践一下



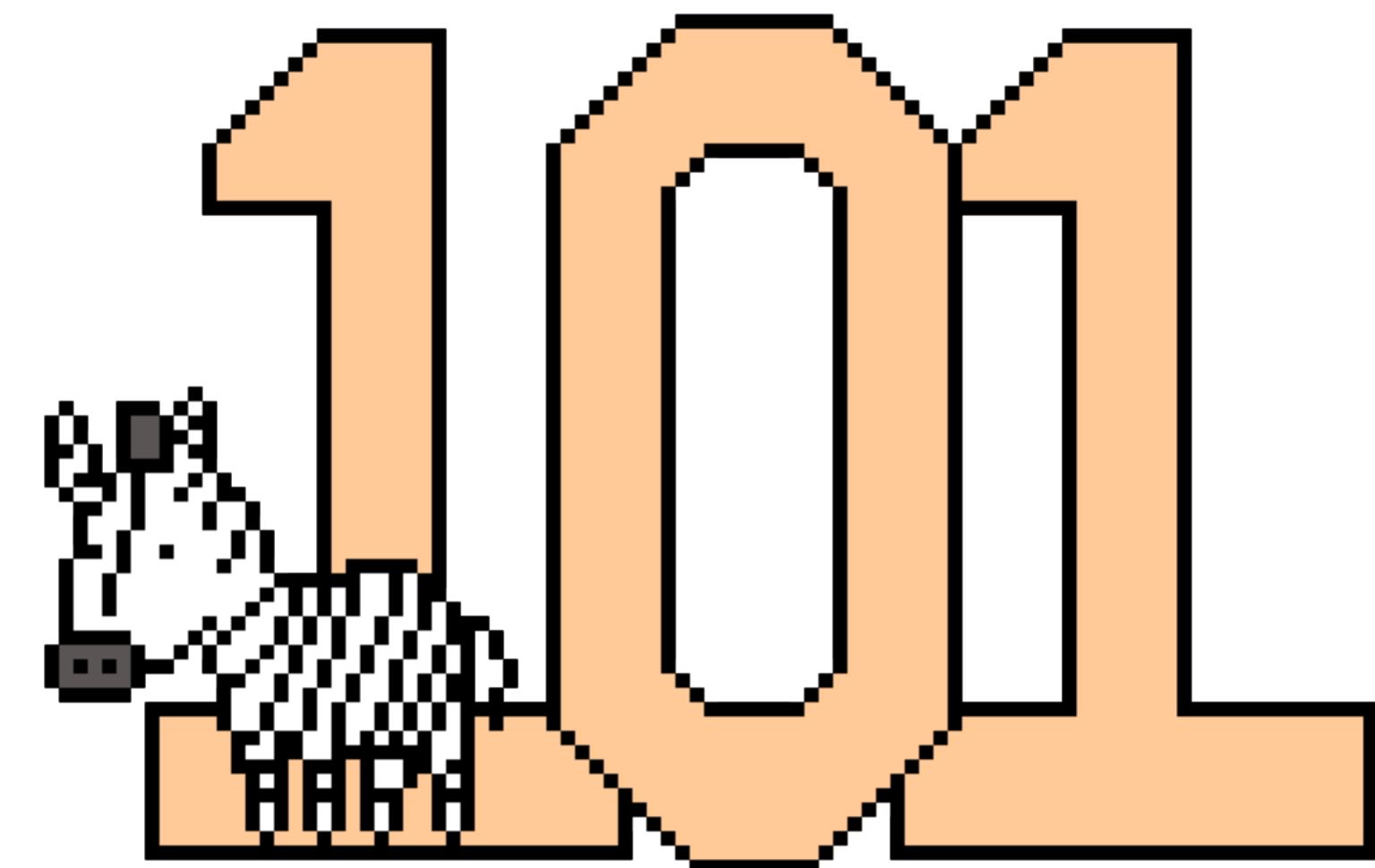
让我们开始我们的第一步：

rails new cbra

然后

cd cbra

rm -rf ./app



收工！



先别急着鼓掌

还有下文

假设我们要实现后台（admin）模块

`rails plugin new engines/admin --mountable`

```
→ cbra git:(master) ✘ rails plugin new engines/admin --full --mountable
  create
  create  README.md
  create  Rakefile
  create  admin.gemspec
  create  MIT-LICENSE
  create  .gitignore
  create  Gemfile
  create  app
  create  app/controllers/admin/application_controller.rb
  create  app/helpers/admin/application_helper.rb
  create  app/jobs/admin/application_job.rb
  create  app/mailers/admin/application_mailer.rb
  create  app/models/admin/application_record.rb
  create  app/views/layouts/admin/application.html.erb
  create  app/assets/images/admin
  create  app/assets/images/admin/.keep
  create  config/routes.rb
  create  lib/admin.rb
  create  lib/tasks/admin_tasks.rake
  create  lib/admin/version.rb
  create  lib/admin/engine.rb
  create  app/assets/config/admin_manifest.js
  create  app/assets/stylesheets/admin/application.css
  create  app/assets/javascripts/admin/application.js
  create  bin/rails
  create  test/test_helper.rb
  create  test/admin_test.rb
  append  Rakefile
  create  test/integration/navigation_test.rb
  vendor_app test/dummy
  append  /Users/madao/dev/cbra/Gemfile
```

Gemfile

```
65  
64 gem 'admin', path: 'engines/admin'
```

然后执行 bundle install

```
→ cbra git:(master) ✘ bundle install  
You have one or more invalid gemspecs that need to be fixed.  
The gemspec at /Users/madao/dev/cbra/engines/admin/admin.gemspect is not valid. Please fix this gemspec.  
The validation error was '"FIXME" or "TODO" is not a description'
```

填充下必要的内容

```
dummy.gemspec — dummy

$:.push File.expand_path("lib", __dir__)

# Maintain your gem's version:
require "admin/version"

# Describe your gem and declare its dependencies:
Gem::Specification.new do |s|
  s.name          = "admin"
  s.version       = Admin::VERSION
  s.authors       = ["Madao"]
  s.email         = ["madao@madao.me"]
  s.homepage      = "admin.yourdomain.com"
  s.summary        = "This is a Admin Module."
  s.description    = "This is a Admin Module For A CBRA Project."
  s.license        = "MIT"

  s.files = Dir["{app,config,db,lib}/**/*", "MIT-LICENSE", "Rakefile",
  "README.md"]

  s.add_dependency "rails", "~> 5.2.1"
  s.add_development_dependency "sqlite3"
end
```

首先

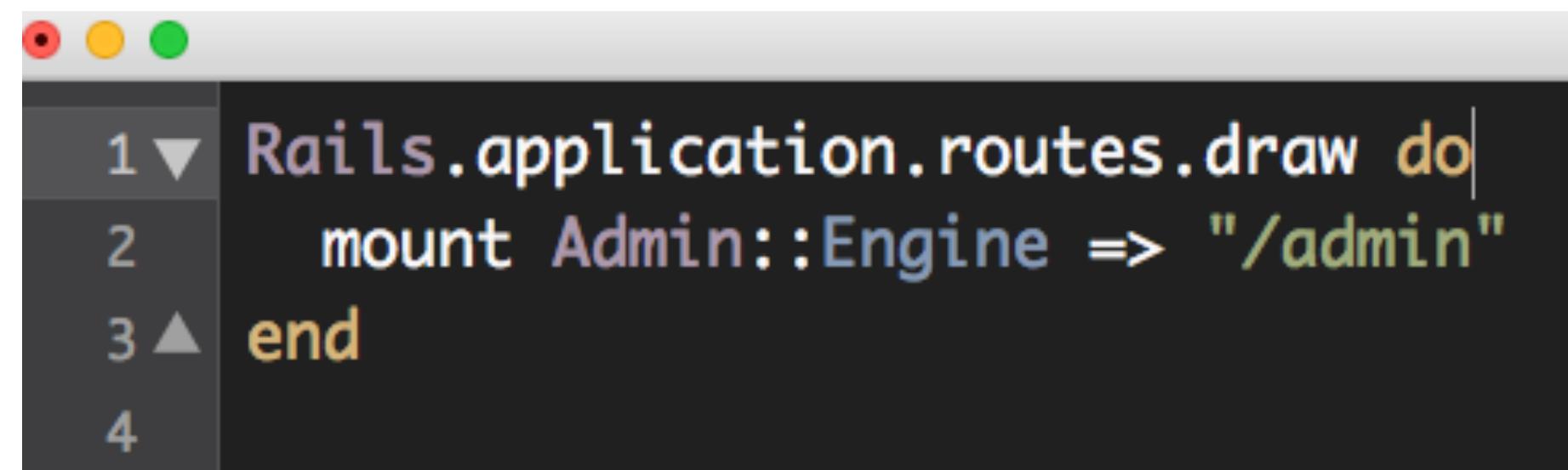
`cd engines/admin`

`rails g controller welcome`

```
→ admin git:(master) ✘ rails g controller welcome
  create  app/controllers/admin/welcome_controller.rb
  invoke  erb
  create    app/views/admin/welcome
  invoke  test_unit
  create    test/controllers/admin/welcome_controller_test.rb
  invoke  helper
  create    app/helpers/admin/welcome_helper.rb
  invoke  test_unit
  invoke  assets
  invoke    js
  create    app/assets/javascripts/admin/welcome.js
  invoke    css
  create    app/assets/stylesheets/admin/welcome.css
```

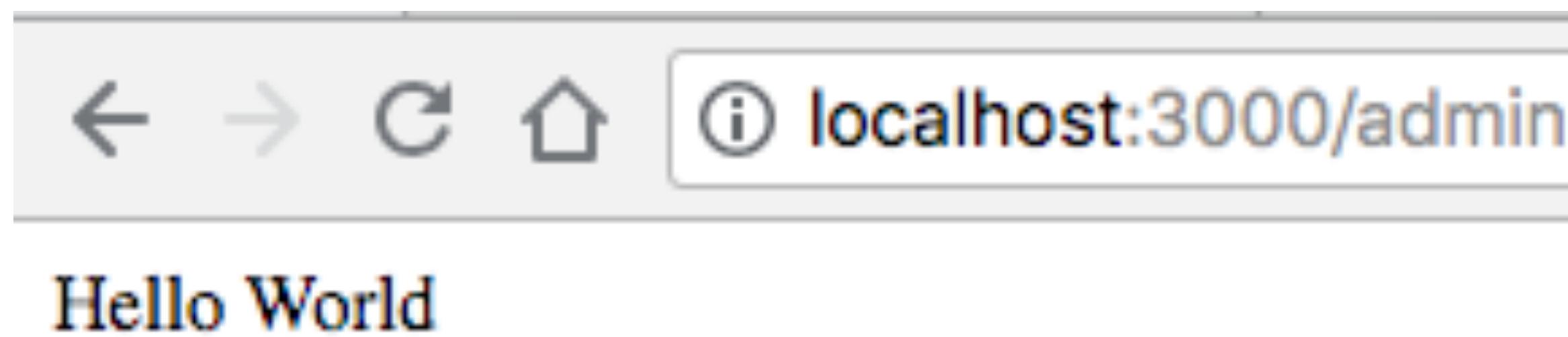
创建好Welcome#index 这个action 所需的内容

然后编辑路由



```
1 Rails.application.routes.draw do
2   mount Admin::Engine => "/admin"
3 end
4
```

然后打开你的 /admin 这个路径，Tada



为什么是Engines 目录下的?

我们用组件是否为一个Rails 实体来决定它所应该在的组件目录:

/engines 目录

需要作为一个Rails 实体服务存在的组件。
或者可以直接理解为，这是一个可直接拆分为微服务的。

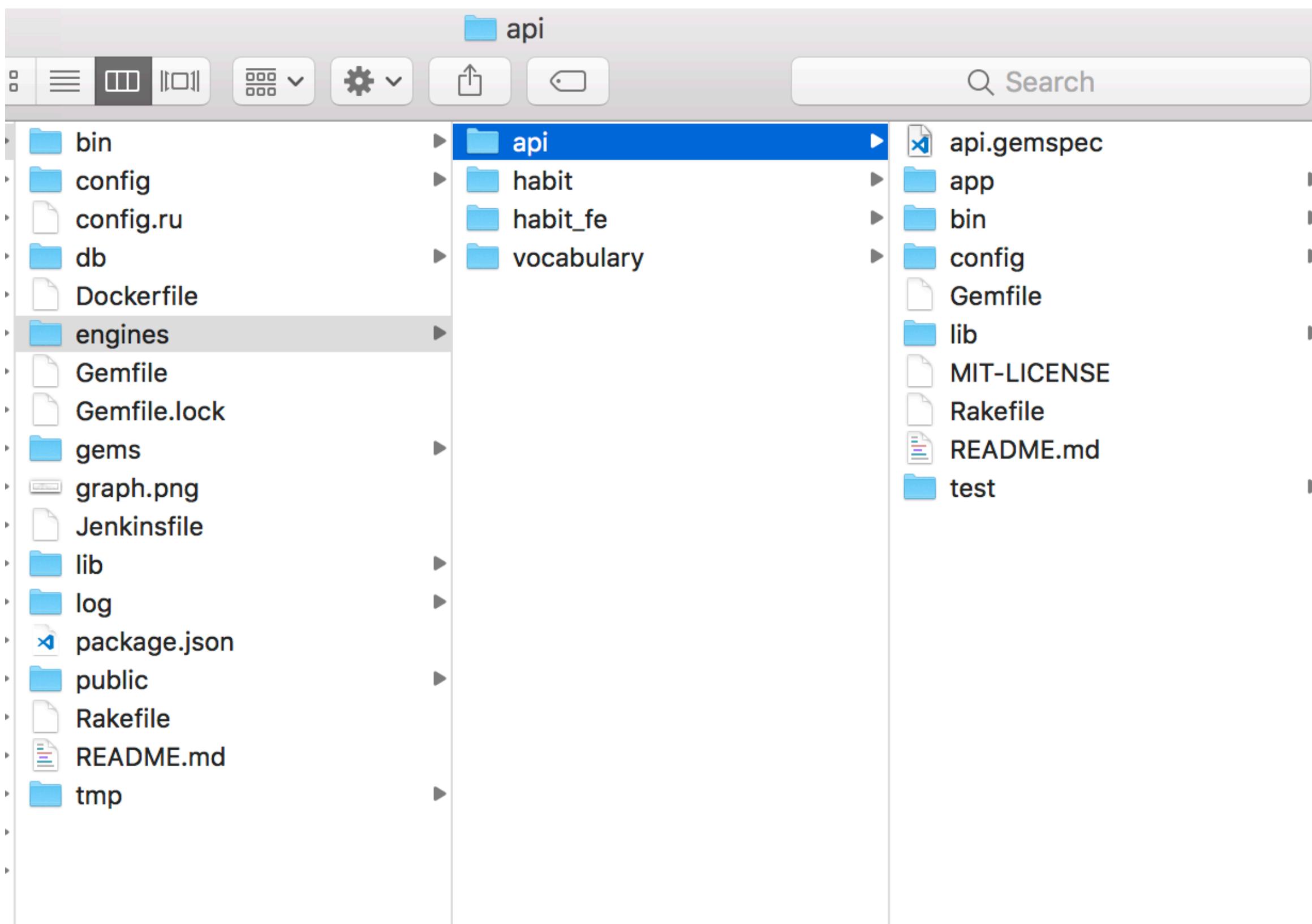
/gems 目录

「无状态」，不依赖Rails 的，或者仅负责数据的流动，或者一些特定的计算。

所以项目会长这样



所以项目会长这样

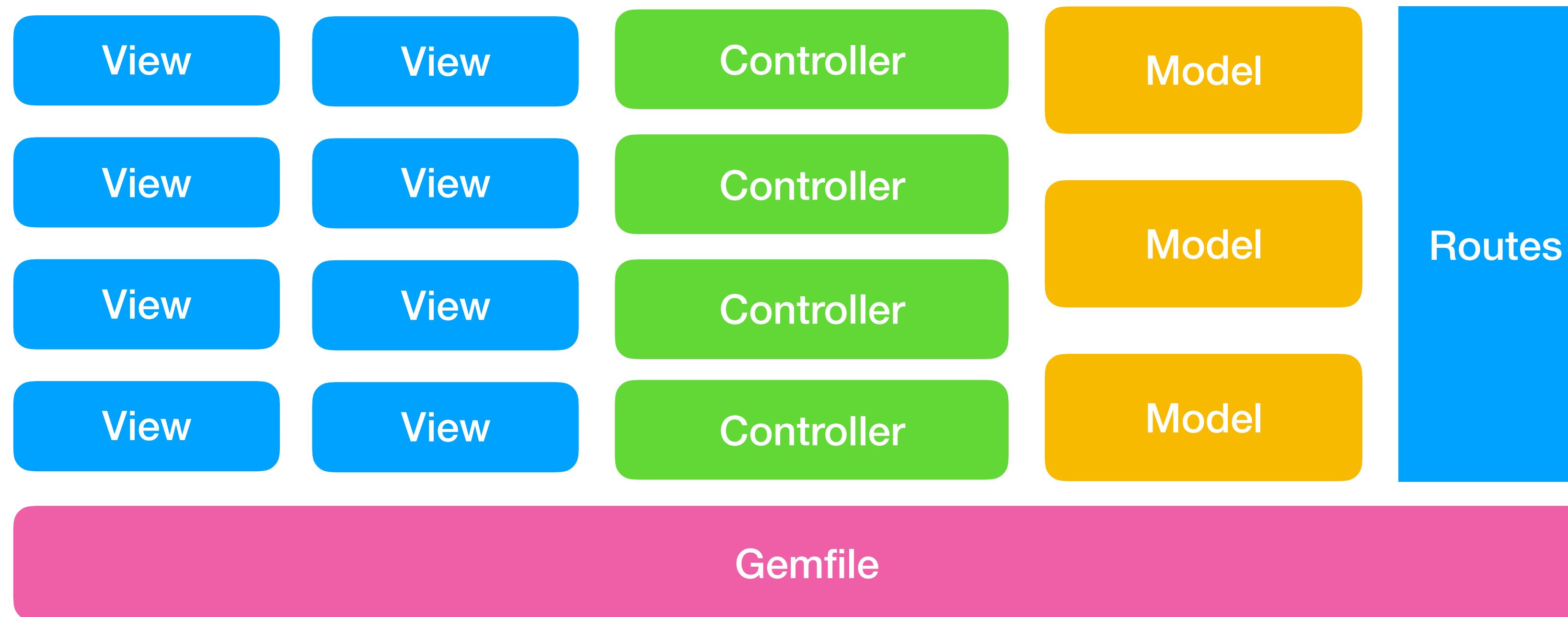


前置内容完毕，我们来尝试思考如何拆分已有的巨大单体

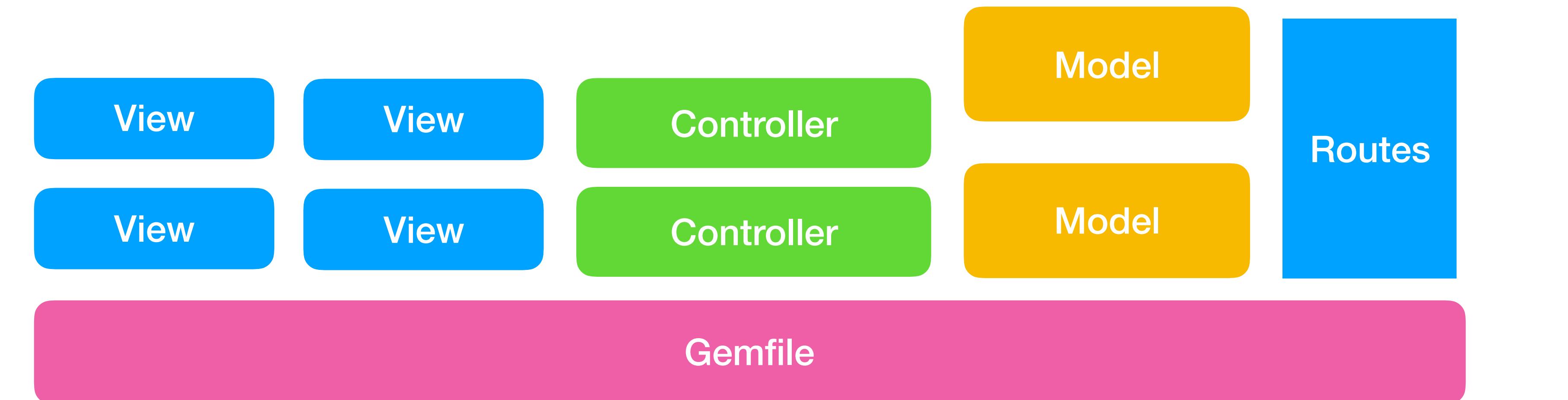
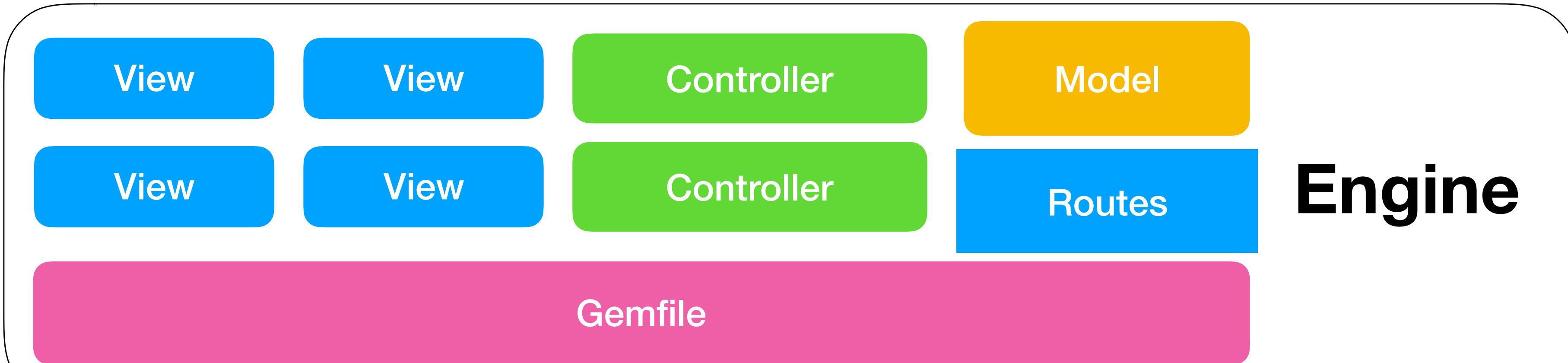
符合以下规则， 我们拆成新的模块

1. 项目中明显内聚的领域， 拆分成模块。
2. 粒度较大的服务， 尝试拆分。
3. 用业务域将业务逻辑分解为多个有界上下文， 每个上下文是一个模块

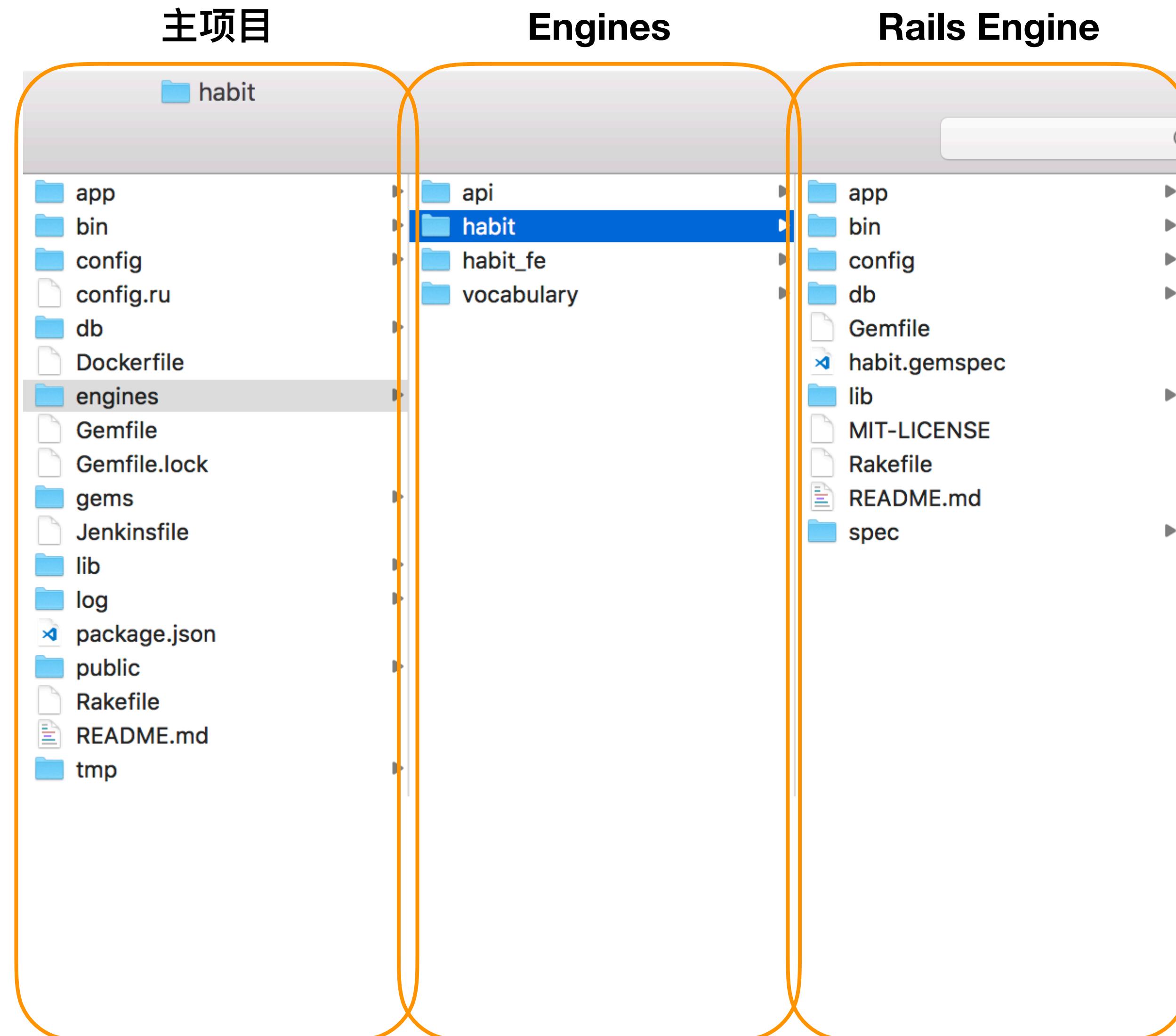
单体应用



单体应用

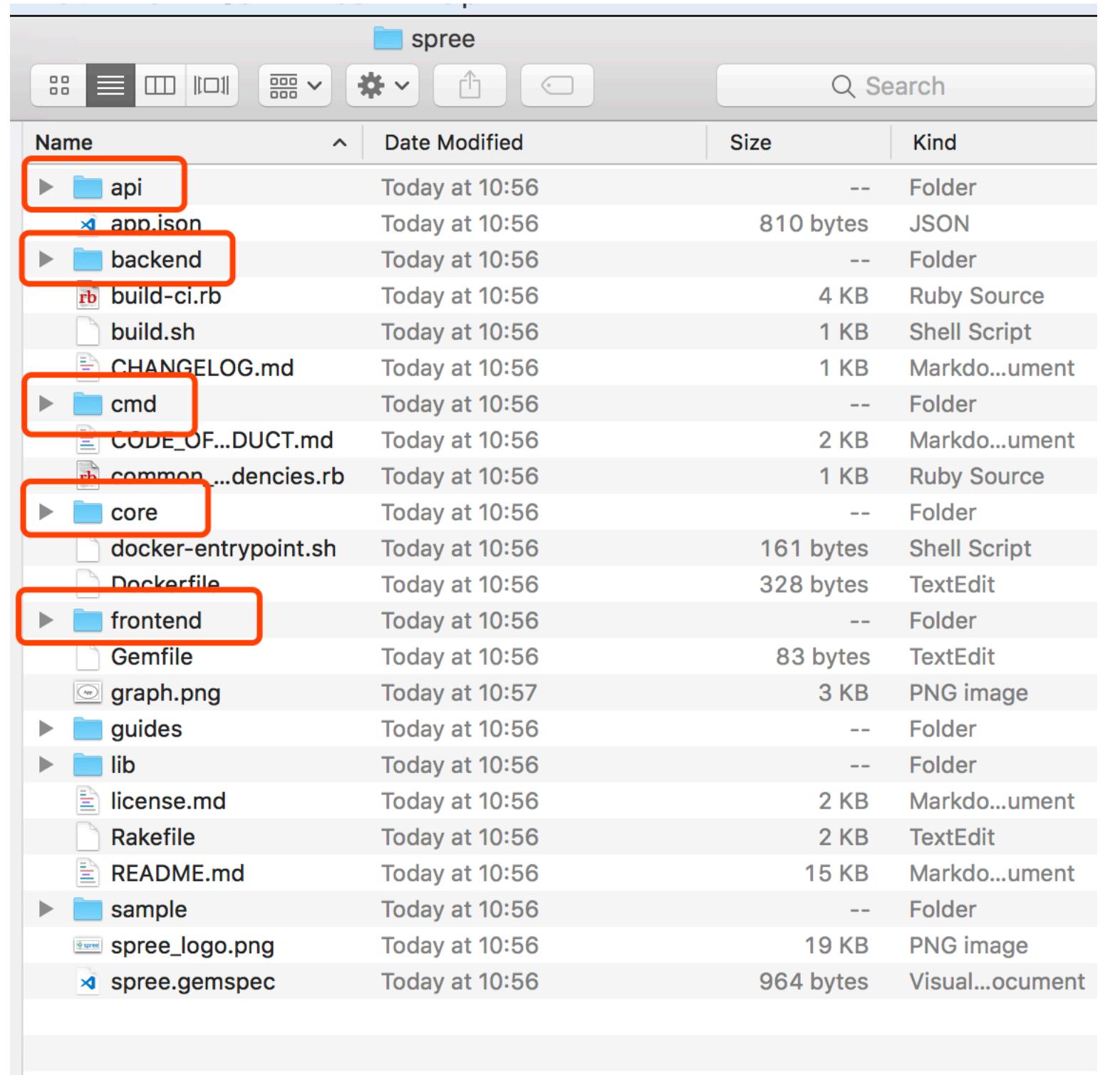


Engines



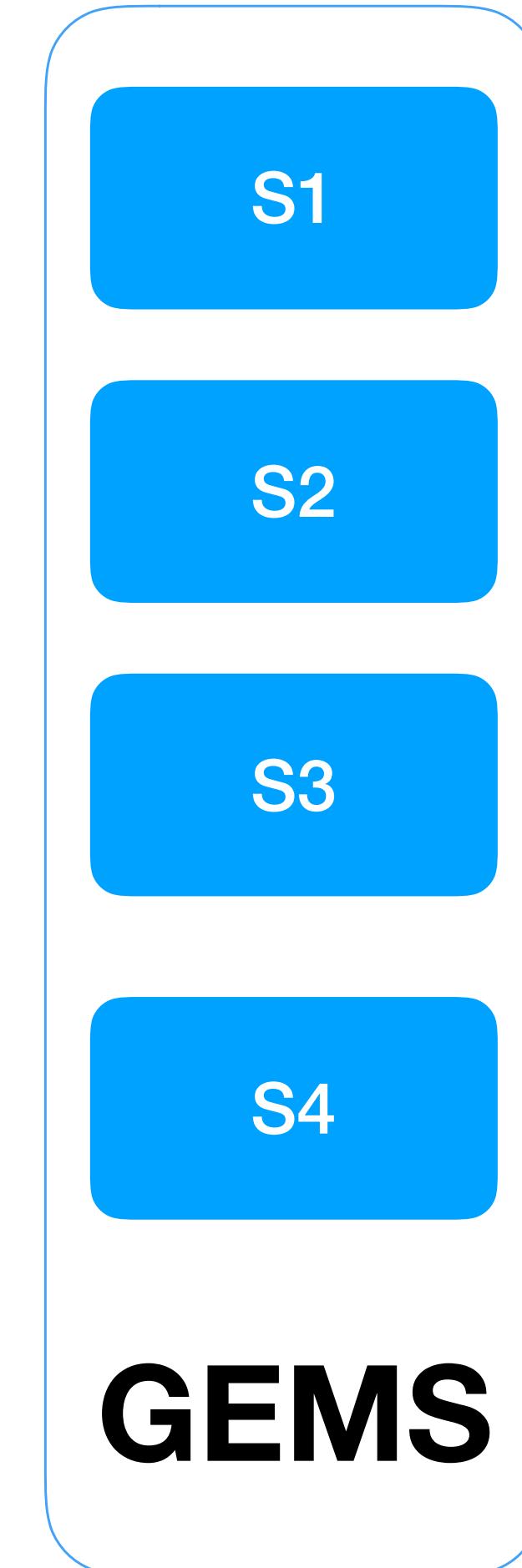
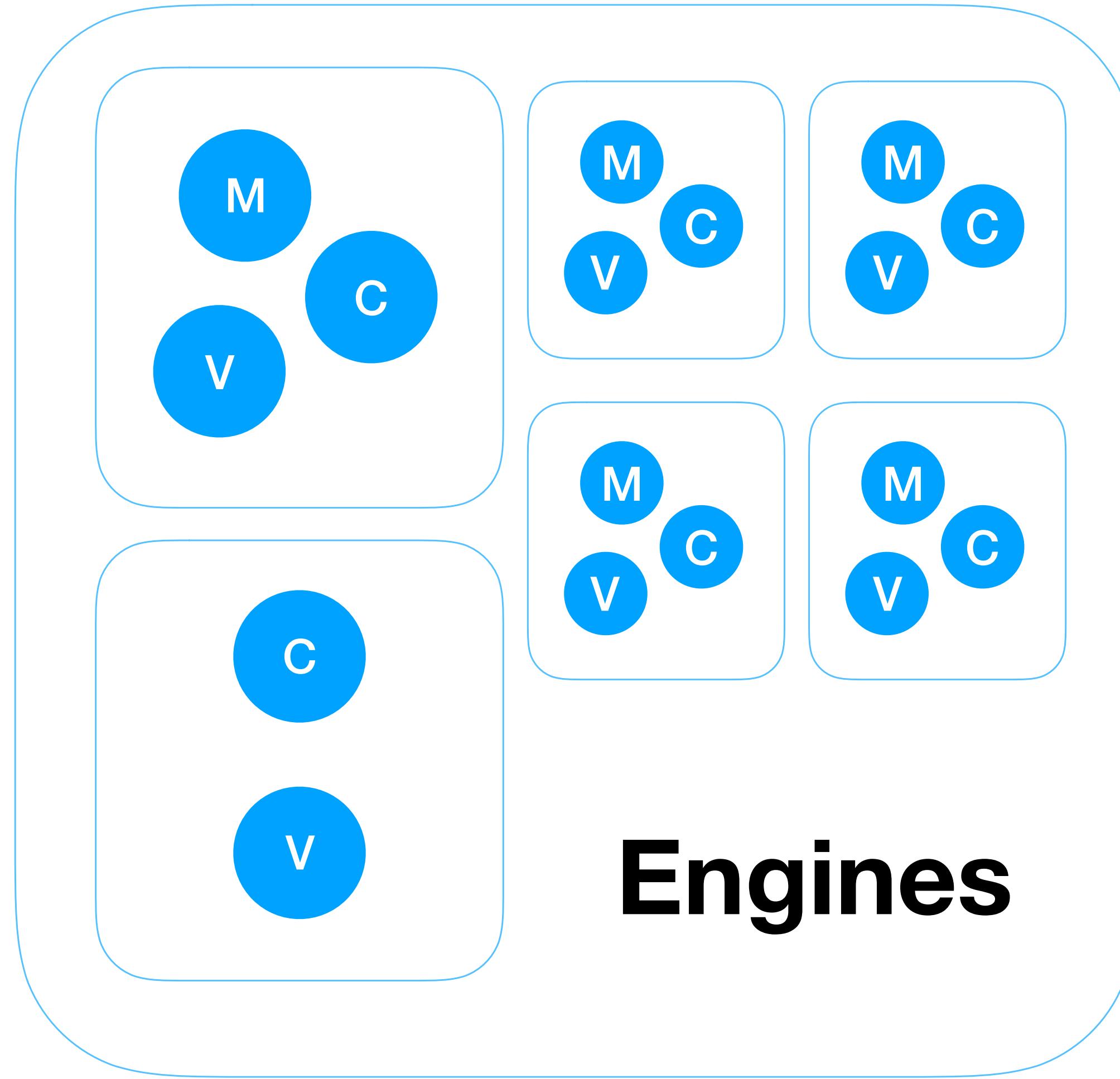
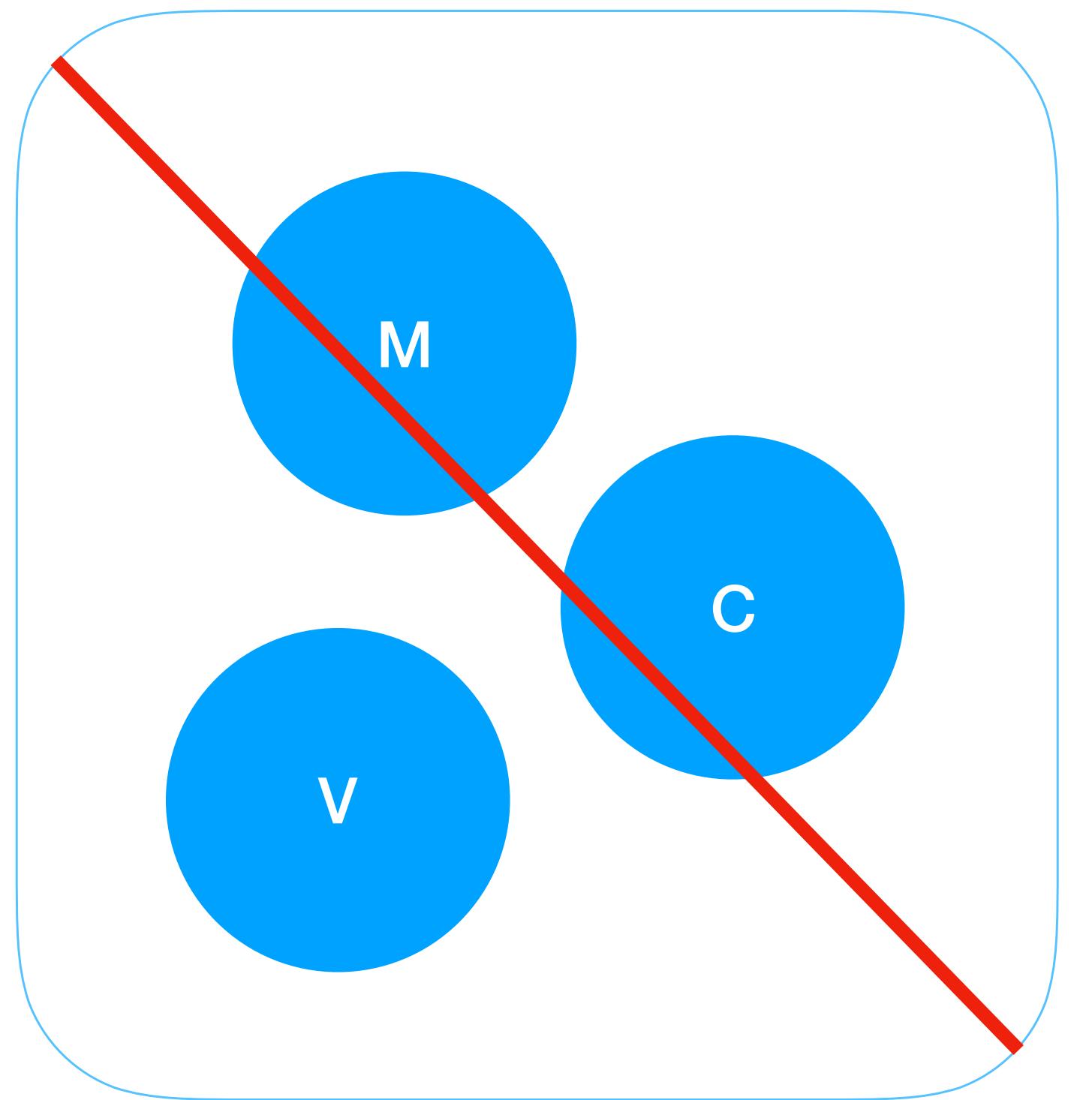
用 Spree 举例





```
s.add_dependency 'spree_core', s.version
s.add_dependency 'spree_api', s.version
s.add_dependency 'spree_backend', s.version
s.add_dependency 'spree_frontend', s.version
s.add_dependency 'spree_sample', s.version
s.add_dependency 'spree_cmd', s.version
```





系统过于庞大是否会带来持续集成的负担？

**事实上只要保证代码分割是正确的，
那就只需要跑对应component 下的CI 即可。**

真的有这样的做的公司/项目吗？

ROOT

<https://www.joinroot.com/>

65,000多行Ruby / Rails应用程序代码

135,000多行测试代码

37 个 Engine

22 个Gem

30+ 工程师



<https://powerhrg.com>

491910 行代码

超过30个 Component



<https://github.com/spree/spree>

优点

更好的代码组织

模块内的代码需要的
上下文信息更少

随时可迁移成微服务

可以选择性的跑集成内容

劣势

同步/学习成本

测试复杂度增加

数据库迁移(**Migration**)依旧会很慢

考虑是否模块化也需要思考成本

部署效率低下的问题仍未消失

适合的场景

1. 体量较大臃肿不堪的单体应用
2. CI 时间过长的单体应用
3. 频繁出现合并冲突的单体项目
4. 符合合并规则的微服务

不建议的场景

1. 初期项目
2. 过小的团队
3. 已经有良好的代码组织的项目

如果你的项目连模块化架构都无法达成，
为什么你会觉得微服务能解决你的困难呢？

软件工程没有银弹

FAQ

