

Sining-Liu's Blog

September 27, 2016

怎样更能保护注意力，在没钱的情况下——一个小故事

上次李笑来在全栈班上演讲说他总是打车上下班来保护自己在路上的注意力。我觉得这种方法属于拿钱来保护注意力。但是，对于很多人来说，或者，对于很多人的目前阶段来说，钱是有限的：拿钱保护了这方面的注意力，就没钱保护那方面的注意力了。

哪种方法最能保护注意力？

事情是这样的：我之前的 Mac 总是闪屏，非常影响注意力。本来想等新 mac 上市之后再更新自己的设备，但是这一个月（甚至多个月）怎么办？

我可以买一台新的 mac，但是就得借钱。但是，总想着还债太影响注意力。或许，有办法能让我不想着还债。但是这样做的结果通常是，到期了我却还不上贷款，因为我真的没让还债这件事占用我注意力，也就没有花心思想着怎么攒钱（说白了还是赚的太少，所以需要想着怎么攒钱，哈哈）。总之，就在我准备买下一台新 mac 的时候，我停住了，未来好几个月的还款计划真的需要占用我的注意力。

也许事情的走向能像老师买 Aeron 一样，买了之后生产力提高 N 倍，收入随之提升。自己之前有这样试过，买生产力最高的设备，但是很少有成功过，因为我还没有提高收入到一定程度，还款计划就开始剥夺我更多的注意力。或许，我的方法不对，没有想好快速提高收入的方法。也没准，是我对很多提高收入的技能还不熟练，没有把它们变成肌肉记忆，就带着很多包袱和一个还款计划上路去提高收入。

问题应该不是出在购买更好的设备上。用更好的设备提高生产率，从长远来看是正确的，因为自己的生活不能永远在跟低效的设备干吧。问题可能是我借的数额对于我当前的收入能力来说太大了点，导致注意力被抢走太多；或者是，最能提升我赚钱能力的不是买一台设备，而是锻炼一下技能之类的软实力等，也就是说，好设备的加权还比较低，需要配合其他方面的成长一起应用才好。

但我之前一直在想一个叫做折现率的事情。未来的收入 $\div (1 + \text{折现率}) = \text{未来收入在当前的价值}$ （假设投资期数是 1）。设这个价值为 N， $N/24$ 就是未来在现在的时薪。今天耗费在低效设备上的时间是在花费未来的钱嘛？！但是，现在来看，这个公式或许可以修改为 $N/\text{注意力时间}$ ，来表达折现后单位注意力的收入。所以，还是哪种最能保护注意力，哪种浪费未来的钱会比较少。（如果考虑睡后收入，公式可能会有变化，但是睡后收入可以看成是依靠注意力采取一定的主动行动后得到的收入，没有这份主动行动，就没有那份睡后收入。所以这里也算作主动收入的一种。）

最后，家里还有一台旧 Air，用 Time Machine recover 过去，和原来的 Pro 基本相同。速度虽然是慢了点，但一般情况下还好，而且节省了一些借钱上的注意力。所以，我现在在用 Air 写文章。

总结

1. 保护注意力是降低损失的更好方式。具体来说，如果借钱买好设备，和用差一点的设备但是少花一点钱中选一个，更能保护注意力的那个更好。

其次：

1. 购买好设备前考虑一下好设备对于目标贡献的加权。说人话就是，想想有没有更重的事情需要用到注意力，不要光图买设备一时爽。
2. 培养购买设备之上的软实力是很重要的。

[jQuery 实现动画效果](#)

September 19, 2016

jQuery 实现动画效果

以前的互动只是两页纸的动画，只有开始和结束，但是 jQuery 可以用 animate 实现更加连续的动画。

比如这句

```
$(this).animate({'top':'-10px'});
```

这个动作会让 this 这个物体往上跳 10 个 px，连续的。而要想让物体回到原来的位置，就把 -10px 设置为 0。

另外，我们还可以把动画效果调快。

```
$(this).animate({'top': '-10px'}, 'fast');
```

或者直接写上动画持续事件也可以比如写上 400，就是 400ms。

我们也可以把 animate 要动的效果写在 css 里面，

```
.vacation {  
    transition: top 0.2s;  
}
```

```
.highlighted {  
top: -10px; }
```

比如像这样。

但是有一个问题，只有当浏览器支持 transition 的时候，才能用。

如果不支持，我们需要多写上几种，比如：

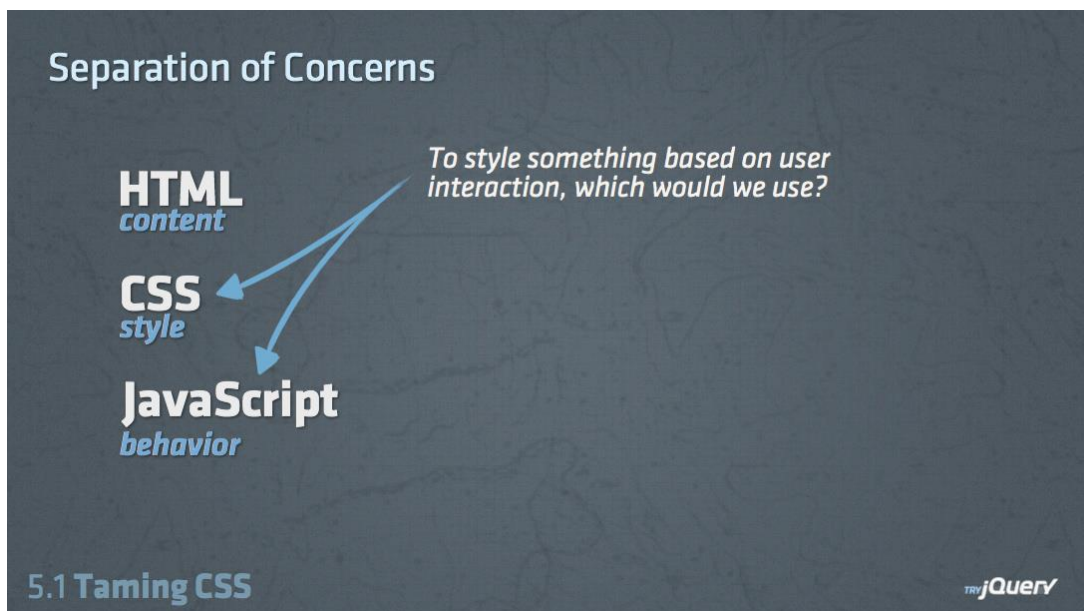
```
.vacation {  
    -moz-transition: top 0.2s;  
    -o-transition: top 0.2s;  
    -webkit-transition: top 0.2s;  
    transition: top 0.2s;  
}  
.highlighted {  
    top: -10px;  
}
```

第二行代码是告诉火狐浏览器的，后面也是类似。

然后我们就可以用这句话来实现 animate

```
$(this).toggleClass('highlighted');
```

最后送上一张图，让我们分清楚 html, css, jquery 的分工。



[jQuery 中更多 on 事件](#)

September 19, 2016

[jQuery 中更多 on 事件](#)

我们之前见过 click，但其实还有更多

1. click
2. dblclick

这两个算点击事件

1. focusin
2. focusout

这两个算焦点事件

1. mousedown
2. mousemove
3. mouseover
4. mouseenter
5. mouseup
6. mouseout
7. mouse leave

这 7 个算鼠标事件

1. keypress
2. keydown
3. keyup

这 3 个算键盘事件

1. blur
2. select
3. change
4. focus
5. submit

这 5 个算表单事件

更多事件请参考 <http://api.jquery.com/category/events/>

而且, on 事件最后一个填写 function 的地方, 可以是在那里自建的 function, 也可以是在别处定义的 function

[jQuery 互动时如何防止浏览器乱跳](#)

September 19, 2016

jQuery 互动时如何防止浏览器页面乱跳

有些显示功能, 点开之后浏览器的 bubble 会往上跳, 导致用户很疑惑。这时候我们需要用一些方法阻止乱跳。

第一步, 创建事件参数

```
$(document).ready(function() {  
    $('.see-photos').on('click', function(event) {  
  
    });  
});
```

这里的 event 参数就是事件参数, 每次用到 on 事件里的这个 function 的时候, event 就产生了, 我们就知道 jQuery 有可能要乱跳了。

第二步, 用事件调用阻止乱跳的方法

如果不想乱跳 (有时候我们也想让它跳), 就用.stopPropagation()这样的方法。

```
$(document).ready(function() {  
    $('.see-photos').on('click', function(event) {  
        event.stopPropagation();  
    });  
});
```

```

    event.preventDefault();
    $(this).closest('.tour').find('.photos').slideToggle();
  });
  $('.tour').on('click', function() {
    alert('This event handler should not be called.');
```

[用 jQuery 做点互动效果](#)

September 18, 2016

用 jQuery 来点互动效果

点击事件

我们平时写 ready 都是这样的

```

$(document).ready(function(){
});
```

同理，如果告诉你 on 是事件，on('click')是点击事件，你也可以照着上面的形式写一个 on('click')之后执行的 function

```

$('button').on('click', function(){
});
```

和上面的 ready 是如出一辙，button 相当于之前的 document，都是 dom 下的文件，on('click')相当于之前的 ready，后面的 function 就是要执行的方法

在方法里写上一些东西，比如 remove，append 之类的，就会在点击按钮是发生

更精确的点击事件

对于这句：

```

$('button').on('click', function(){
    $('button').remove();
});
```

如果有两个 button，那么他们都会被 remove 掉，所以，我们要找到我们是在点那个 button。

用 this 可以做到这一点：

```

$('button').on('click', function(){
    $(this).remove();
});
```

这里把刚才的 button 换成了 this，意思是，我们点哪个，哪个就是 this，哪个就要 remove。

如果你想要在点击后插入一些文字之类的，也可以用 this：

```

$("button").on('click', function(){
    var msg = $('<p>Text</p>');
    $(this).after(msg);
});
```

这样，点了 button 后，这个 button 后面就会加上 msg 代表的信息。

当然，这样 after 加上的信息，跟 button 是包在一个 div 里面的，我们可以用.closest()把 msg 加在任意目录下面，让他跟 button 同级

假如 button 被包裹在 ".class"下面，我们可以用.closest('.class')把他放在 class 的直接下级目录里：

```

$("button").on('click', function(){
    var msg = $('<p>Text</p>');
    $(this).closest('.class').append(msg);
});
```

```
});
```

让数据互动起来

刚才我们制作的点击事件是让样式消失，或添加。但是有一些方法感觉并不明智。比如这句话：

```
var msg = $('<p>Text</p>');
```

如果以后我们要加入别的文字怎么办，难道新建一个 msg 么？

这里有一种数据叫做 data，可以让我们操作显示的内容。

首先，让我们看看如何取出数据。

在有一些表现下，会有这样的数据属性 data-price，data 是这个表现的属性，price 是这个数据的名字。要想取出 price 的值，可以用这样一句话：

操作的 html 语句：

```
<li class="vacation onsale" data-price='399.99'></li>
```

使用的 jQuery 语句：

```
$('.vacation').first().data('price'); // 取出名字叫 price 的 data 的值
```

在点击事件中，也可以用 this，取到当前点击区域的 data。假设按钮和数据是 closest 的关系，就可以这样取到 data 的值：

html：

```
<div id="tours">
```

```
  <h1>Guided Tours</h1>
```

```
  <ul>
```

```
    <li class="usa tour" data-discount="299">
```

```
      <h2>New York, New York</h2>
```

```
      <span class="details">$1,899 for 7 nights</span>
```

```
      <button class="book">Book Now</button>
```

```
    </li>
```

```
    <li class="europe tour" data-discount="176">
```

```
      <h2>Paris, France</h2>
```

```
      <span class="details">$2,299 for 7 nights</span>
```

```
      <button class="book">Book Now</button>
```

```
    </li>
```

```
    <li class="asia tour" data-discount="349">
```

```
      <h2>Tokyo, Japan</h2>
```

```
      <span class="details">$3,799 for 7 nights</span>
```

```
      <button class="book">Book Now</button>
```

```
    </li>
```

```
  </ul>
```

```
</div>
```

jQuery 取值：

```
$('.button').on('click', function() {
```

```
  var tour = $(this).closest('tour');
```

```
  var discount = tour.data('discount');
```

```
});
```

上面这段话的意思是对于 button 标签都起作用的，只不过，如果找不到 tour 就打印不出来东西而已。

如果我们想让他只能对 tour 里面的 button 做操作，就限定一下 jQuery 的范围，所以改成这样更精准：

```
$('.tour').on('click', 'button', function() {});
```

要不要和别人比

September 18, 2016

要不要和别人比

不要

原因是：

1. 用双赢或者社交可以有效代替竞争
2. 我们还没有到达有必要竞争的地步

「比」是什么

如果单单给出一个结论，就说不要跟别人比，未免有点太草率，因为我们还没有清楚的定义什么是「比」。实际上，在这句结论中也只有比这个字需要详细说明一下，剩下的每个字的意思都比较确定。当然，如果有时间，我们还是可以讨论一下什么是别人。

什么是比？我今天在听认知学习法，其中谈到有两种心智模式，一种是僵固性人格，一种是成长型人格。或者，按照李笑来的说法，一种是表现型人格，一种是进取型人格。僵固性人格的特点是，认为人的智力等因素是固定的，他们要做事情来表明自己智力很优越。而成长型人格认为一个人的能力是可以改变的，学习之后可以提升。当然，作者提示，通常人们是混合了两种性格特点，在某些场合是前者，在某些场合是后者。而跟别人「比」的作用是，让僵固性人格得到胜利的快感，认为自己的能力强于别人。按照 Xdite 后来的提示，僵固性人格可能认为这一瞬间的胜利就是永恒（因为他们认为人的能力值是固定的），这一瞬间的比别人强，基本能说明自己永远比别人强。而我猜想成长型人格会是那种只和自己比（但并不是为了逃避在和别人的比较中落于下风），并且一直在行动的人。

所以，我们这里说的，**跟别人比是指有一类人（僵固性人格）为了证明自己的优越，和别人比较当下的能力。**

比赢了当下的能力，就以为永恒的超越。当然，这种能力可能是通过竞赛，考试，或者工作体现出来。

按照僵固性人格的特点，这种「比」可能会有两个推论。在比不过别人的时候沮丧难过，选择放弃，因为反正这辈子都比不过别人了。比的过别人的时候耀武扬威，觉得赢了一辈子战争，招人嫉妒或者不思进取。这么一说，感觉僵固性人格就是典型的 loser：碰见怂的就不可一世，碰见硬的就怀疑人生。虽然我们可以给僵固性人格找一些优点，比如热爱竞争，不至于安于现状（虽然也可能是安于现状，然后永远找比自己弱的人竞争）。但我现在觉得，对于普通人，真的没有到那个竞争的份儿上。

用双赢或者社交可以有效代替竞争

跟别人「比」有两种角度，一种是让自己比别人强，另一种是让别人比自己弱。前者或许还能让自己发奋一点，后者却可能干出损人不利己的事情。

但是，无论哪种，用社交代替竞争，实现双赢，或许才是更好的生存指南。

为什么这么说，这世界上有千千万万个人，赢了当前这一个并不算什么，赢了他又代表什么呢？不如互相学习，知识互补，大家都学到更多，提高各自的劳动生产率，如此一来可以面向更广阔的世界做更多的事情。尤其是，对于个人来说，大多数人才市场都不是零和博弈，不是有你没我，不是孤岛生存，你死我活。个人的生活不是像那些大公司一样，全球只有这么多用户，用了你家的就不用我家的了。尤其是在学习的时候，普通人还远远没有达到零和博弈的地步。

但如果，两个人采取竞争的态度，无论是让自己比别人强，还是让别人比自己弱，都很有可能阻断两个人之间的知识分享：让对方不能获得你的精华，你也不能获得别人的精华。这样，在更广阔的人才市场上，你们都只有自己的那一点知识，而不是两者知识的叠加。

或许有些人特别善于在竞争中挖到对方的技术，但是也有些人特别善于捂住自己的技术。如果大家都把心思放在挖别人的墙角和捂住自己的蛋，估计会损失很多原本可以用于学习的时间。

我们还没有到达必须竞争的地步

上面已经提到，竞争很可能让很多人走偏，本来是要进步，却花费了大量时间，让别人无法进步。

其次，我觉的对于个人来说，世界太广阔了。不是搜索引擎 google 一家独大，绝对不会和微软分享技术的这种情况。如果个人在成立这样的世界级公司之前就想着干掉别人，那等于放弃了所有藏在别人脑子里的

知识（因为社交总是相互的，应该不会有人愿意无条件的给你知识，而不想从你这里得到点知识）。或许，这样的人可以看书，那里面也有很多知识。但是，书里的知识是固定的，或者说，是死的。这些知识在不同人的脑子里会有不同的变体，而且这些变体是智能的，可以根据你的问题随即应变，有时候跟那些书上的知识同样给人启发。

总结

对于普通人，也就是还没有真正的成果来证明自己不是普通人的人，应该用社交（或者叫分享）来代替竞争，因为：

1. 世界很大，两个人的合作比竞争更能让个人学到知识，从而让两个人都能面向世界做更多的事情。
2. 普通人还没到达必须竞争的地步，对于个人来说，市场是无穷大的，甚至是随着个人能力的增长越来越大的，不是固定的。所以，更快提高自己的水平，才是实现人生目标更好的途径。而如前所述，社交比竞争更能提高水平。

另外

如果看到别人比自己强，怎么办？

我觉得要压抑住自己的嫉妒心，想想他的技能是不是自己需要的，如果是自己需要的，就跟他 social，赞扬他的优点，问问他是怎么做到的。

如果自己不需要这方面的能力（我想不出有什么能力是不需要的），也不妨先交个朋友，以后可以一起合作嘛。

最后，选择什么样的态度来面对和别人比这件事，跟环境有很大关系。我觉得选择一个乐于分享的环境是更好的选择，而如果在一个竞争的环境中，也要想清楚自己的目标是什么（也就是说，自己的目标不是和某个人竞争）。

[如何创建访客人用户](#)

September 14, 2016

如何创建客人用户

有时候，我们想让客人先进来逛一逛，可以随便看点什么，等到有操作，比如购买或者关注的时候再登录。这时候我们可以创建一个客人用户，方法如下：

创建一个用户，并把他存储在 session 里

以下代码来自 devise 的 github：

```
class ApplicationController < ActionController::Base
  protect_from_forgery with: :exception
  # if user is logged in, return current_user, else return guest_user

  def current_or_guest_user
    if current_user
      if session[:guest_user_id] && session[:guest_user_id] != current_user.id
        logging_in
        # reload guest_user to prevent caching problems before destruction

        guest_user(with_retry = false).reload.try(:destroy)
        session[:guest_user_id] = nil
      end
      current_user
    else
      guest_user
    end
  end
end
```

```
end
```

```
# find guest_user object associated with the current session,
```

```
# creating one as needed
```

```
def guest_user(with_retry = true)
```

```
  # Cache the value the first time it's gotten.
```

```
  @cached_guest_user ||= User.find(session[:guest_user_id] ||= create_guest_user.id)
```

```
rescue ActiveRecord::RecordNotFound # if session[:guest_user_id] invalid
```

```
  session[:guest_user_id] = nil
```

```
  guest_user if with_retry
```

```
end
```

```
private
```

```
# called (once) when the user logs in, insert any code your application needs
```

```
# to hand off from guest_user to current_user.
```

```
def logging_in
```

```
  # For example:
```

```
  # guest_comments = guest_user.comments.all
```

```
  # guest_comments.each do |comment|
```

```
    # comment.user_id = current_user.id
```

```
    # comment.save!
```

```
  # end
```

```
  #也就是说，你可以把 guest 用户产生的一些数据保留下来，比如 guest 的评论内容
```

```
end
```

```
def create_guest_user
```

```
  u = User.create(:name => "guest", :email =>
```

```
    "guest_#{Time.now.to_i}#{rand(100)}@example.com", :is_guest => true)
```

```
  u.save!(validate => false)
```



```

    session[:guest_user_id] = u.id
  u
end

```

end

其中的 create_guest_user 方法就是在创建访客，而调用这个方法的地方好像是在这个方法中：

```

def guest_user(with_retry = true)
  # Cache the value the first time it's gotten.

```

```

  @cached_guest_user ||= User.find(session[:guest_user_id] ||= create_guest_user.id)

```

这里好像就把刚刚产生的 guest 用户装进了 session 里。

实际上，访客是根据 user 的 model 创建的，只不过没有密码，一般 user 的各种属性他也是 nil 的，为了更好的判断一个 user 是不是访客，我们可以给 user model 中加一个栏位，比如 is_guest，当访客被创建的时候赋值为 true

有了 is_guest 这个栏位，我们可以在 user model 中设定访客的判断方法：

```

def guest?

```

```

  is_guest

```

```

end

```

以后我们在调用当前用户的时候，就可以用 current_or_guest_user，而在区别两者的时候 current_user.guest? 这样的方法

最后，我们要修改一下 config，因为 guest 用户要跳过很多校验才能被创建，具体方法如下：

```

# config/initializers/warden_strategies.rb

```

```

Warden::Strategies.add(:guest_user) do

```

```

  def valid?

```

```

    session[:guest_user_id].present?

```

```

  end

```

```

  def authenticate!

```

```

    guest = User.find_by(id: session[:guest_user_id])

```

```

    success!(guest) if guest.present?

```

```

  end

```

```

end

```

```

# config/initializers/devise.rb

```

```

Devise.setup do |config|

```

```

  ...

```

```

  # ==> Warden configuration

```

```

  config.warden do |manager|

```

```

    manager.default_strategies(scope: :user).unshift :guest_user

```

```

  end

```

```

  ...

```

end

现在，当你调用 `current_or_guest_user` 的时候，如果没有 `current_user`，会有一个 `guest` 用户被创建出来，返回值就是一个 `user`，然后通过 `current_user` 的校验。你也可以把访客的数据打印出来看一下。

总结：

1. 在 `application_controller` 中创建产生访客的方法
2. 更改 `config`，让访客通过校验

参考：

<https://github.com/plataformatec/devise/wiki/How-To:-Create-a-guest-user>

<http://ruby-rails.hatenadiary.com/entry/20141212/1418380288>

Rails 前端加速方法一

September 14, 2016

在 Rails 中后挂 js，给前端加速

js 的想过通常是 dom 已经 ready 之后才会应用，如果过早下载可能会影响 css 在页面上的输出，让用户觉得网站加载好慢，然后拜拜。

为了让页面看起来加载快一点，可以把一开始用不到的 js 放在 css 和 html 都下载之后再下载。

方法如下：

把 layouts 中的 js 文件放到 body 结束之前

```
<%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
```

这样的语句可以放在 body 结束之前 (`</body>`) 加载，而这个时候 html 都已经加载完了。

把各个页面的 js 放在 application.js 之后加载

刚才我们已经把 `application.js` 放到了 body 结束之前加载。而如果有些 view 里面也写了 js 的话，就要放在 `application.js` 之后，body 结束之前加载。

具体方法是：

在 layouts 中，写上 js 文件印出的位置，这个位置要在 `application.js` 之后，body 结束之前。语句如下：

```
<%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
```

```
<%= yield :handwrite_javascript %>
```

```
</body>
```

主要看 `<%= yield :handwrite_javascript %>` 这一句

在具体的页面中，把 js 文件包裹在

```
<%= content_for :handwrite_javascript do %><%end%>
```

中就好了。

如何利用 gem 在 Rails 中做 SEO

September 13, 2016

如何在 Rails 中做 SEO

在 Rails 中可以做到自动设定页面的一些 meta 标签。而 title 和 description 对于 SEO 是有帮助的。

要想自动设定页面的 meta，步骤如下：

安装 seo_helper

```
gem 'seo_helper', '~>1.0.3'
```

然后

```
bundle install
```

更改 config

```
touch config/initializers/seo_helper.rb
```

填入

```
SeoHelper.configure do |config|
```

```

    config.site_name = "My Awesome Web Application"
end
这样你的 title 就变成上面设置的 site_name
当然，你也可以给 description 之类的元素设定默认值
SeoHelper.configure do |config|
    config.skip_blank = false
    config.site_name = "My Awesome Web Application"
    config.default_page_description = "My default description"
    config.default_page_keywords = "My default keywords"
    config.default_page_image = "/example.png"
end

```

但目测有时候 default_page_image 不好用

使用设定好的 title 和 description

我们已经在上面设定好了 default 的 title 等元素，现在我们可以使用它们。

在 head 文件中写上这样几句就可以

```

<%= render_page_title_tag %>
<%= render_page_description_meta_tag %>
<%= render_page_keywords_meta_tag %>
<%= render_page_image_link_tag %>

```

上面几句分别是设定 head 的 title, description, keywords, icon, 打开网页，就能检查到 head 文件中的 title, meta 已经改变。

为每个页面设定不同的 title 等

你可以在 controller 的 action 中给页面设定特别的 title 等，只需要写成这样：

```

def show
  #...

  set_page_title @article.title #设定 title

  set_page_description @article.description #设定 description，下面以此类推

  set_page_keywords @article.tags
  set_page_image @article.excerpt_image_url
end

```

end

另外

如果你的 icon 打印不出来的话，可以把<%= render_page_image_link_tag %> 换成 html 语句：

```
<link rel="shortcut icon" href="/default_image.png">
```

参考

https://github.com/techbang/seo_helper

团队写 Code 的合作方式

September 13, 2016

团队写 code 的合作方式

我在的「约霸」项目采用的是一个人开发完整功能，然后大家把功能合起来的合作方式，每个人的工作包括前端和后端，这样锻炼了一个人全栈的能力。

这种合作方式在初期的时候效果不错，每个人能够具体的想像功能效果，然后把它做出来。但是也有问题，

包括：

1. 前端风格不统一
2. 代码重复量高
3. 动线混乱，用户容易迷路

今天看 Xdite 老师给第一组分配任务，是另外一种分工合作方法。觉得能解决以上问题，具体情况如下：

User Story

建立产品围绕需求，能够把需求转化为开发方向的工具之一就是 User Story。

在接到一个整体需求后，一人把他拆分成各个 User Story。

前端、后端

一人建前端画页面，留出后端接入的位置（这里面可能需要请设计师来帮忙设计页面）

一人建立后端，工作可能包括建表，建 controller，写路径。

对于前后端对接的问题，我们猜到的情况是命名可能不一致，这可能需要前后端多讨论一下，或者后期再把 views 改名。

onboarding, landing page

为了让产品上线就能用，需要从一开始就规划商业上的问题，让产品容易被用户接受。所以，一人来做 onboarding。

我自己想到，on boarding 设计时产生的方法，可以和前端对接，把功能样式展现出来，提醒后端接入。

landing page 相对独立，老师找了一个以前做广告的同学来写这个东西，总之，landing page 的设计可能不只是前端的问题，还有很多非程序的设计在里面。

September 13, 2016

做决策的标准—初步思考

全栈班的同学给我提了个小建议。她说，她观察到我有许多假设中的反向思考。也就是说，假想做什么能够产生最好的结果。

如果仅仅是思考如何最好，也还好，因为这是在思考如何进步。但我知道，我的问题是，拿不准如何更好，所以把各种假想摆在脑子里不知如何是好。

以下是我答同学的信，包括了我自己的思考：

信息很容易缺失，但要尽量收集

决策时会经常存在一个信息缺失的问题，所以需要有很多假设。如果这些假设的条件都摆在眼前，也就没什么好假设的了。

但是，显示生活中，要想让决策信息不缺失，成本是高昂的。或者，就算信息全面了，决策者可能又被信息淹没了，最后只能了解一部分信息，结果还是只能依赖部分信息，而非完整信息。

《原则》这本书对此的建议是：大胆地想，但是要找优秀的人做压力测试，然后根据他们提供的信息做决策。

我想，这样能保证信息尽量全面。值得尝试。

根据「精益创业」和「Growth Hack」，做小规模实验也是很有用的。

过了足够好的标准线，每个选项就都挺好

得到了信息，可能有很多因素要考虑，比如那个信息的权重更大，各有千秋的选项如何决策？

通过你的建议，我想到，我需要定义一个标准叫做“足够好”。过了“足够好”这条标准，就可以成为决定。

你觉得我在反向思考，是因为我在判断这件事是不是最好的。我在找瑕疵，也在找优点。总之，有时候想的有点多，这给我带来了一些好处和一些坏处。

好处是：

我的独立思考能力可能还不错

坏处是：

做决策不够快，犹豫

我现在意识到的问题是，犹豫，或者让别人看出来我的犹豫是不好的。这在组织团队的时候影响士气。我想到的解决办法包括：

1. 抢夺决策时间，当别人对我的决策造成压力的时候，我要控制时间，让大家的节奏慢下来，给自己争取到更多思考时间，这样思考的更全面。
2. 明确决策的标准，达到标准的就可以成为决定。
3. 听听别人的意见，补充自己的思考。

具体的实践方法呢，我想我可以这么做：

1. 什么时候别人会造成我的决策压力呢？就是当别人向我提建议而我不想让他们失望，别人对我有所请求但我不知道是否应该回绝。如果我想要多争取时间，可能需要把争取时间的方法烧成肌肉记忆，给大脑腾出空间来做决策。我已将想好，我要对着镜子联系这样几句话：“这个很不错，但你是基于什么样的想法呢？”，“这个很不错，但我需要想一想如何安排。”
2. 我需要定义一下什么是足够好。从经验来看，似乎总有更好。比如，如果我有更多资源的话，我可以直接创业，如果有很好的项目的话，我可以不用去找工作。还比如，我们的项目要加什么功能，要做什么改进，停下来还是继续做。有很多因素在这里面都是可以考虑的，比如时间，金钱，人员，未来发展。但对于我来说，每天因素的权重是不一样的，我需要给最看重的因素设定一个比较高的权重。我可能比较看重未来发展。但我依然不清楚继续做项目还是找工作，对未来的发展那个更好。所以，我想到事实是他们都很好，都过了“足够好”的标准，都是可以的。那对于订阅李笑来文章的这种事呢，我想，“足够好”的标准是对我的效率和思考有用就可以买，对于买了这些东西，就会占用的现有的资源，导致我没有办法购买下一个资源，我只能说，任了。但我也会想，当前提升的效率会让我以后更有可能购买以后遇到的资源。因为我不是全知的，所以这两种都有可能发生的情况下，我选择第一种。因为如果是第二种，我怕我会永远想，未来还有更好的，然后没有买，导致无法在当下埋下提高效率的种子。如果仅是对享受有用，就还没有到“足够好”的标准。但是，有时候我们无法特别明确的区分享受和效率的区别，偶尔把账算糊涂了的话，就忍一忍吧。
3. 听听别人的意见，是我在第 1 类的实践方法里就埋下伏笔的。当别人向我提要求，或者提建议的时候，我希望我能用肌肉记忆说出：“你的提议不错，但是，你是基于什么想法呢？”让别人吐出建议的时候，也好让我有实践思考。

总结

1. 决策前，收集更多信息：搜索或者找人做压力测试
2. 用 lean startup 或 Growth Hack 的方法做验证
3. 定义足够好的标准，过了标准线的选项就可以不再纠结
4. 学会控制决策节奏

[git blame 追溯是谁在 update 文件](#)

September 12, 2016

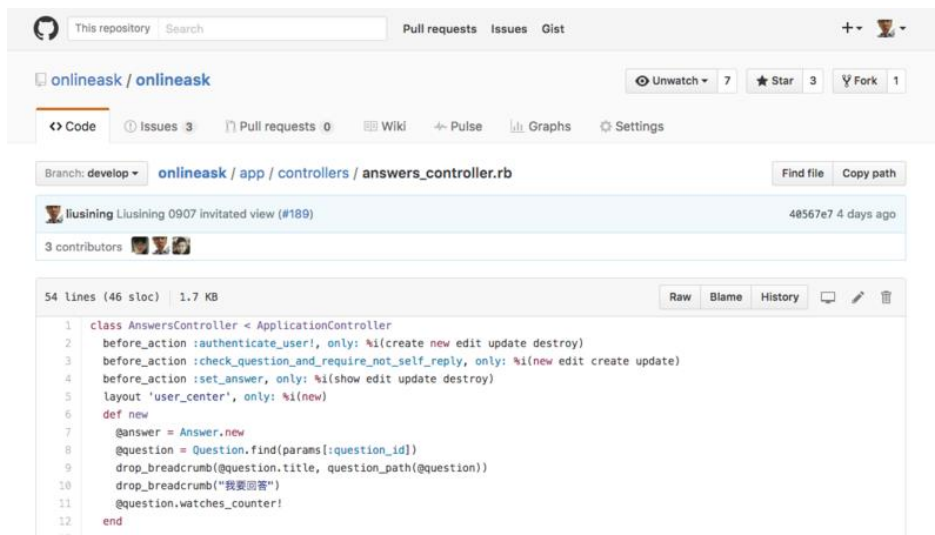
git blame 追溯是谁在 update 文件

前几天在 [angela zhu 的公众号](#)上看到说，她曾经使用 git blame 查看 api 的创建者，我试验了一下 git blame，不能够用因为语法不对，usage 信息是：

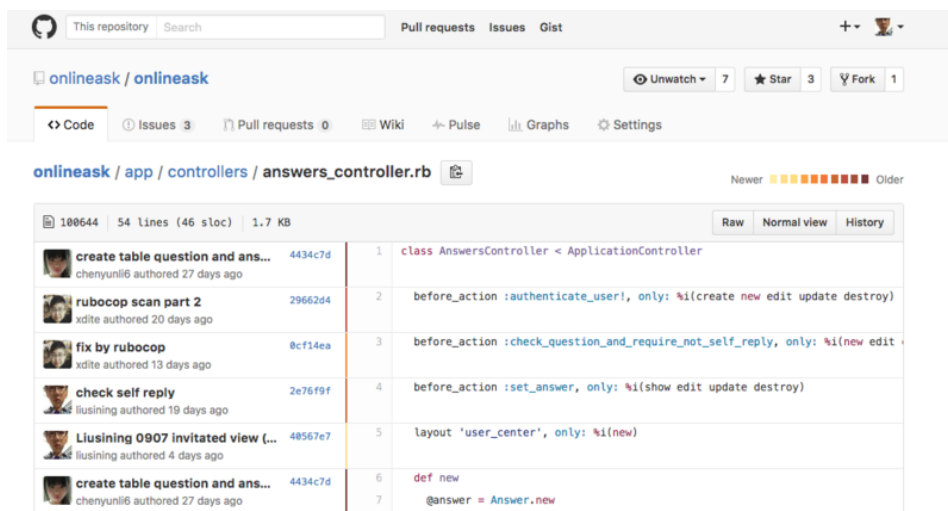
```
usage: git blame [<options>] [<rev>] [--] <file>
```

那么我们要怎么看到这个文件的各个修改历程呢？也就是这个文件的 timeline。

用网页版 blame



点击右侧中间部分的 blame 按钮，你就会看到有谁修改过文件。



指令版 git

在终端输入：

git blame foo

可以查看 foo 这个文件下的修改历程（foo 是任意文件名，我看有别的程序员这么用）

git blame --since=3.weeks -- foo

查看 foo 这个文件 3 周内的更新

参考及更多

更多 git blame 请看 <https://git-scm.com/docs/git-blame>

什么叫做升级

September 11, 2016

什么叫做升级

我现在所处的生活环境中，很多人看李笑来的文章，用李笑来发明的词汇来说话，就比如“升级”这个词。但我一直排斥使用这个词，因为我觉得计算机系统的升级方法给人脑是有差别的，我不确定这样比喻是否真的能行。今天又看到 [xdite 老师发文](#)谈人脑升级和电脑升级。所以又勾起了我的思考，什么是升级，人脑升级真的可以用电脑升级的模型来研究么？如果不行的话，哪里不行，如果行的话，又该如何理解“升级”这个词？

升级是改变

我想，电脑和人脑升级的共同特点是改变吧。计算机有了新的算法或者特性，更安全，操作更人性化等等，而人脑呢，有了新的思维方式，有了新的熟练思考的技能，有了新的思考领域，这些都是改变。如果我观察过的人还算多的话，计算机的改变是更容易看出来的，有数据可以量化，有内在数据，包括运算速度，bug 修复，功能补充，外在数据有使用者人数，更新率，用户报错数量等等。但是人的改变首先可能是没那么容易看出来，可能有时候自己都看不出来，或者以为自己看出来但是并没有升级。其次可能只是改变，而不是升级，也就是说，跑到另外一个地方原地踏步。我们来依序讨论这两种。或许还有其他类型的改变，但是我现在的思考可能没那么全面（也就是没那么多），所以有些内容还留待以后补充。

人的升级真的能看出来么？

我在回想这样一些情景，跟有些人对话的时候真的能产生更多的思考，他们说的话所反映出的思考的点就是和我不一样。但是，当我们出生的时候，大家好像都只会哭，或者说，就算有些人哭的声音大，有的声音小，有的基因好，有的基因差，大家看起来的差别并不大。也就是说，在某个领域有些人的状态会比另外一些人不一样，比如编程领域，甚至思考人生这个领域。

如果人跟人能够不一样，当我从我这个点走向他那个点的时候，我不就是改变了么？而且这种改变就跟原先我和他的差距一样大。所以改变是能够看出来的，如果你知道自己原先什么样（记录，分享，拍照，录视频都是帮助自己记住原来样子的方法）。

我想，我们现在是在谈论是不是能够看出来自己在升级，所以这个比喻还算恰当。总结一下，就是：人和人的差别就会很大，如果一个人从自己的思考习惯（这或许就是大家说的思维模式）变成另外一个人的思考习惯，这期间此人的改变就同样大，大到可以看出来。

升级是好的改变

我们上面说了，就是个体的改变如果不容易发现，那不同个体的对比或许能说明人和人的差别可以很大，但因为两个人都是人，所以两个人的思考方式都在人类的可能性之中，但是我们如何评判两种可能性，那种更好呢。又或者回到刚才的情境中，如果一个人想要模仿另外一个人的思考方式，如何判断这种改变就是升级，就是好呢？

首先，我想问，什么是好？标准是什么？

我还不知道宇宙间有什么绝对标准说这就是好，但是在一个文化背景下，一个商业模式下，一个社会关系下，应该会有相对的好，就比如在思考人生这个领域，李笑来就比我好，知道的词就比我多，价值观就更明确，虽然他的价值观可能不适用于美国人，但是身处在北京这块地方，他的思考至少让我看到这样一个可能性，就是他可以创造很多词汇，有更准确的人生概念，有更清晰的价值观，他可以流量更大，收入更多，有更多资源干事情。所以说，对于完成特定目标，甚至人生目标，我们能知道什么样的指标更好。

我想，这就是名人效应的由来吧。我们总是引用名人说的话，觉得那就是对的。有时候，这确实有一点问题，因为名人所处的环境跟我们并不相同，我们也不知道他们说这话的背景，不是他们说什么都是真理，但是他们确实让我们看到这样的可能性，就是他们过得更好。

那为什么他们过的更好呢，或者说，为什么他们已经完成了升级呢？我也不知道，总觉得得有点原因吧（如果不是随机事件的话，但是有绝对的随机事件和不随机事件么？），所以就想模仿看看。应该说，这就是我们想要模仿，甚至主动模仿，别人的思考方式，然后达到跟那些自己看好的人一样的水平。如果实现了，这在一定的环境下，应该算是升级。

那如果没实现呢？也就是说，改变了之后发现不好，那算什么？也就是说，升级是个马后炮概念（或者叫事后诸葛亮概念，英文应该叫 rationalization，后期合理化）么？改变对了，就是升级，改变错了就不是升级对么？

想到这里，我觉得我们需要讨论一下，如何在事前就知道一个改变是不是真正的升级？

有不好的改变么

李笑来讲了很多关于知识变现的故事，大致的轮廓是，有个人很好学，别人不理解他为什么花那么多时间学习，因为很多人相信学习无用论。但是这些学习的人出了国，进入了另外一个世界，用自己的知识在国外年

入百万。而且用更多的钱学更多知识，财富继续增长。

透过这个例子，我想要分辨的是，如果一个改变被定义为不好的改变，那他是当前状态下的不好，还是永远的不好，是相对的不好的，还是只是没有那么好？

对于学习知识这一块老师，我觉得比较好的思考工具是李笑来的运气概率数轴。



按照这张图，我们或许可以认为，就追求财富这条路上，不学习的人（可能没有完全不学习的人）也能有运气获得自己想要的财富，学习的人也有可能获得不了什么财富，但是两者的概率是有很大的差别的。如果让一个人理解了这张图，让他选自己通往财富自由的道路，为什么不选学习呢，概率高啊。

可能很多人认为小孩学习是痛苦的，所以让他们从积累知识的路上获得财富自由，会觉得这样的财富也太艰辛了吧。但是，我觉得离开中国（我见到的一些老师都在告诉学生学习是痛苦的，感觉好可怕，好怕这些老师在我上学的时候给我的思想写入了什么东西），离开一部分特定的学习环境，学习就可以很快乐，比如一个班的同侪都很乐于分享，和乐于进步的人一起学习，会让我很快乐。

所以，学习知识，作为一种改变，变成不好的改变的可能性是比较小的。

那有什么进步是可以不归结到学习上来的？我的意思是说，好像很多进步都是可以归结到学习知识这个类别里边来的。体育竞技是需要学习锻炼肌肉和使用肌肉的技巧，投资理财需要学习投资的方法，我感觉很多事情都是靠学习和了解知识可以提高成功的概率，虽然他们可能没有下棋的概率那么高，但是通过学习，了解一些规律，还是能帮助这个行业的人提高胜算的。这种学习，就是升级吧。

当然，学习什么可能就要靠个人运气（比如你出生的环境恰好适合学这个），个人天赋和个人选择了。我想到的例子是山西煤老板。如果说煤老板读过的书确实没那么多，还比大学教授有钱的话，那真是个人运气的问题，而不是学习的问题。因为地处山西，煤老板更可能学到搞煤矿的知识，在城市里土都挖不了的小孩确实没有学习这个的条件。另外就是，我一个叔叔是学农机的，据说当时国营农机厂特别多，但是后来农机厂都不行了，这个叔叔就失业了。也就是说，一门知识被抄底了，不再被需要了，原本学的东西可能就没有那么有用了。

学习什么知识更有用，我觉得要看个人目标吧，对于个人目标八竿子打不着的东西学了或许有用，或许能启发新思考，但是概率是不是有点小呢，有启发概率更大的知识在那里为什么不去学呢（除非启发足够大的知识都已经被你搞定了，只能靠跨界混搭来激发新思考，但是，这种概率如何评价呢？这是创新者的窘境么？）？

所以，由学习知识带来的改变至少是好的改变，可能没有那么好，但至少是好。如果想要学习足够好的知识，提高自己升级的概率，[以我的经验来看，找老司机要可以模仿的 example 会是一条路](#)。如果找不到老司机

（但是，现在交易越来越方便，更多老司机出来卖知识了，我们应该庆幸一下我们更容易找到好的知识），如果学习了没那么有用的知识（就比如学校里），你可能会觉得学习这些知识跟学习另外一些知识比，可能就是在浪费时间，会不会有这样的感觉，如果跟他人的进步对比，你会不会觉得自己学这些知识，而没有用这些时间学习另外一些知识是一个错误的选择，你的改变会相对来说是个不好的改变（或者说，质量不够好

的改变)。这是有可能的。从社会角度来说，个体的多样性就是这么来的，不能大家都学习一样的知识，而且，学了一样的知识也没有用，因为下一波技术革命是什么，可能没有人，没有公司能够准确预测，不然那些有钱的公司干嘛不扑上去。所以说，学了就目前社会来说没那么重要的知识也没关系，就当丰富社会的多样性了，哈哈。如果你不想被丰富了社会多样性，怎么办？我也不知道，我觉得要学习一些知识，从而才能判断另外一些知识是否重要，所以总得有个学习的起点，而且，如果你知道要学什么了，就快点学吧，怎么快，或许这一篇对你有帮助。又或者，如果你能看到下一个将要成为稀缺的领域是什么，你就上吧，只不过，在那个领域可能还是已经有老司机了，问他们会比较快。

如何升级

我们上文说道，学习，在总体方向上会是一个好的改变。我觉得这就是一个升级的方向。升级=学习，更好的升级=把握更重要的学习。

但是有些人升级快，有些人升级慢，或许是因为连升级都需要学习，要搞一套行之有效的升级方法才行。就好像李笑来公众号的名称，学习学习再学习，学会了学习的方法，再去学习或许更有效。

对于如何学习，我感觉学习硬知识，就是手艺活，各个具体的领域，用 xdite 老师的方法是我目前见过的最快的。又或许，李笑来提到的最小必要知识是我们应该最先学习的，而不是课本第一页上的知识。对于软知识，就是如何思考，可能需要先发现自己思考的特点，教徒弟，分享，得到反馈，了解自己，然后或许能更好的差缺不漏；看看认知心理学的书，应该也会有帮助，而且可能是更直接的帮助，因为他们已经做了一些实验。

总结

1. 学习知识至少是不退步的，只是学习什么知识才能让你对进步的速度满意
2. 学习好的学习方法很重要，让你接下来的学习更有效
3. 升级的概念应该是指符合特定目标的思考方式的改变，模仿被证明有效的思考方式是一条可选之路

如何检验自己的学习效果

September 10, 2016

如何检验自己的学习方法

教徒弟并看他们做了什么事，导致了什么学习结果

好的学习策略可以拿来为我所用，错误的事情可以避免。

昨天问 Xdite 老师为什么想用认知心理学来检验自己的学习方法，她提到自己在教徒的过程中看自己的徒弟如何学习，加上自己的阅读，思考习惯，总结出一套很有用的学习方法。但没能够解释他的原因，现在她想用认知心理学的实验结果来看看自己的学习方法中哪些是真正有效的。

我们自己也要检验自己的学习方法

老师通过认知心理学实验检验自己的学习理论，那我们呢？

Xdite 老师在自己的[微信公众号](#)中写了一些她观察到的不好的学习方法：

- 看很多理论但是不动手
- 买有难度的书，按照自己的步调学（我理解为，一点一点扣大师的书，每个小细节都扣的那种，而不是理解大师在讲的真正的重点）
- 还没学会就急着飞
- 问一些他现在还不能理解的问题
- 连问题都无法清楚表达

在全栈工程师班上了 7 周的课之后，我确实觉得以前的学习方法太慢了，而且，超出我的意志力范围。我之前想也想过解决办法，包括，提高自己的意志力，降低学习对自己的意志力要求，补偿自己的意志力，但比较少研究如何让自己的学习更快速，从而在意志力减退之前冲出一小点成就感，可能是我之前比较相信勤奋是进步最重要的元素，从而忽略了方法。

回头看 xdite 老师的方法，就我自身的学习效果来看，确实比我原来的方法有效，方法如下：

1. 模仿，照着做一遍，让自己认识一下路（对于我来说，这能很好的克服学习新知识的恐惧感，有点

一块吃吃饭下次好说话的感觉)

2. 再模仿, 2 遍以上, 有了之前的基础, 再走一遍的时候自己会看到更多细节
3. 修改自己模仿的 example, 看看输入和输出的结果又什么关联, 相当于拿自己的 example 做实验
4. 了解为什么。应该说, 这个时候我们有能力了解为什么 (我在 Rails 黑箱这篇文章中说过 Rails 这种人工系统是能够被人们比较清楚的了解的, 因为这个系统是人造的, 应该会有人类用语言记录下了他们, 不像自然系统, 很多原理没有记录, 或者说没有用人类的语言记录)
5. 分享, 或许别人有补充, 或者说教给别人后, 别人就能在自己的 example 上做实验, 如果做出了跟你不一样的结果, 也能给你自己的实验结果提个醒。

精炼一下, 就是:

1. 模仿
2. 再模仿
3. 修改
4. 了解为什么
5. 分享

但是, 这是在老师的辅导下 (老师设计了教材和教学过程), 我能够按照这种学习方法来学习 Rails, 但是, 如果是其他领域呢, 如果没有 xdite 老师来设计教材呢?

我觉得很多领域都可以应用这种学习方法, 让自己进步的更快点。但是如果不能把这种学习方法很好的移植过去, 离开了这个班岂不就废了。而且, 我对此还真有点恐慌, 毕竟, 我前好几年都是按照比较慢的学习方法来的 (我在想, 如果大学四年都用老师的方法来学 Rails, 那岂不是……)。我觉得我这种恐惧是有原因的, 因为我想到学习一门新领域之后, 还是会想要去亚马逊上买教材, 而且还打算懂第一页开始看起。

我在看 codeschool 里的 jQuery 课程的时候, 就感觉到自己的学习过程还是有以前学习的影子, 在小细节上用掉的时间很多 (但我觉得我敢不看 js, 直接跳到 jQuery 就已经是对新学习方法的一种应用), 学到 level3 想要开始在自己的项目中实作, 但是又有点恐慌。怎么办?

恰好我前一天晚上看到老师在拆书, 或许有一些有用的想法。

模仿什么

老师买了很多认知心理学的书, 但是她没有拆开一本看一本, 而是把他们都拆了, 而且拆的很彻底, 把整本书的书脊都用裁纸刀砍掉了。听到老师说这是从日本传过来的方法, 我又想到台湾李敖说他也是裁书, 裁到只留下自己觉得有用的部分。以前觉得这没什么, 但是看着老师裁书真是一刀一刀切下去, 感觉还是很震撼的, 以前没有见过这个架势。但是, 如果真能把一本书中最重要的部分拼在一起, 再次使用精华才方便。而且, 这份精华或许就可以流传出来, 而不是只能教别人从头读起。

所以, 我在想, 我在 codeschool 上学习的时候, 是不是也要把他的教材拆出来呢。或许我可以这样做:

1. 先认识一遍路
2. 再认识一边路, 把精华复制下来, 因为我肯定会忘, 把精华浓缩到一起就更方便
3. 再认识一边路, 看看自己的精华, 把不是精华的东西就去掉吧
4. 找地方修改点 example, 观察效果
5. 了解为什么
6. 分享: 如果有朋友问我 jQuery, 我就分享给他们, 如果没有人问我, 我就主动分享给他们。

我要在 jQuery 的学习上主动试一试新方法。今天下午还学习了一下五笔。既然已经想到这里, 我会想要在自己学习的时候刻意应用新的学习方法。

但是, 跟同学聊关于这个班的时候, 也会觉得, 购买课程或许是一个不错的选择。因为老司机们已经有了精华, 而且提供课程的人很可能已经整理过自己的精华, 模仿这些或许更有价值。

这里给 Xdite 老师打个广告, 她整理的教材让我能够先跳过各种 Ruby 的基础知识, 直接进入更精华的 Rails 世界。如果是我自己学 Rails, 我可能会去啃 Ruby, 然后嫌 Ruby 的知识太庞大, 看不到进入 Rails 的希望, 最后放弃。

我甚至在想，以后想要学习一个新的领域，也要购买老司机的精华，模仿这些精华，这样才能更有效的进入这个领域，而不是长久的在边缘徘徊，徘徊到自己都没有信心进去了。

总结

先模仿再修改，而且尽量模仿精华。

而不是在知识的底层（就是实际中用的少的层，或者已经被别人封装好的层）徘徊。

[用 jQuery 改变 html 元素](#)

September 10, 2016

用 jQuery 改变 Html 元素

创建 html 元素

创建 html 元素需要给一个 js 变量赋值成带 html 标签的 js 方法

```
var price = $('<p>Text</p>');
```

这个变量写在 document 方法中就可以被以后的方法引用，所以这样写

```
$(document).ready(function(){
```

```
    // create a <p> node
```

```
    var price = $('<p>Text</p>');
```

```
});
```

插入 html 元素

插入方法有四种

1. .append()
2. .prepend()
3. .after()
4. .before()

分别是插在内部的最后，内部的最前，后面，前面

我们选定一个位置后，我们可以写成这样：

```
$(document).ready(function(){
```

```
    var price = $('<p>Text</p>');
```

```
    $('<div>vacation</div>').after(price);
```

```
});
```

删除 html 元素

删除就用 remove

```
$(document).ready(function(){
```

```
    var price = $('<p>Text</p>');
```

```
    $('<div>vacation</div>').append(price);
```

```
    $('<div>button</div>').remove();
```

```
});
```

上面这段代码达到的效果是把 price 这个 p 加进去，并且把 button 删掉

另外，我们知道插入元素还有其他语法就是.appendTo()之类的，总过 4 个：

1. .appendTo()
2. .prependTo()
3. .insertAfter()
4. .insertBefore()

可以这么用

```
price.appendTo($('<div>vacation</div>'));
```

[用 jQuery 找 html，更精确版](#)

September 10, 2016

用 jQuery 找 html，更精确版

Descendant selector 后代选择器（次级选择器）

```
$("#id li");
```

这里的 li 就是后代选择器，能够帮助你找到 #id 下面的 li，而不是其他，这样更精准

找到直系后代

```
$("#id > li");
```

一个大于号就解决问题了；

找到两个标签

用，号连接两个要找的标签，放在一个双引号里“如下：

```
$(".class1, #id1");
```

按序号找标签

```
$("#id li:first")
```

```
$("#id li:last")
```

```
$("#id li:odd")
```

```
$("#id li:even")
```

以上 4 个都是更序号有关的查找方法，注意序号从 0 开始

实际上，这一类叫做用伪类

合用

```
$("#id > li:first")
```

直系儿子的第一个

traversing(穿过 html 标签去找内容，于千百 tag 中去一个的首级)

Walking the DOM by traversing it.应该就是 traversing 的由来。

用 traversing 可能会多一些代码，但是据说 js 的效率会更高。有几句：

```
$("#id").find("li"); //用 find
```

```
$("#li").first(); //用序号
```

```
$("#li").last();
```

```
$("#li").first().next(); //用序号的序号
```

```
$("#li").first().next().prev();
```

```
$("#li").first().parent(); //用辈分
```

```
$("#id").children("li"); //(直系后代)
```

语义还是比较明显的

[jQuery 第一句话](#)

September 10, 2016

jQuery 第一句

找到标签

```
$("#h1")
```

\$符就是 jQuery 的意思，里面的 h1 就是你知道的 html 标签。当然，你也可以找到 class，用代表类选择器的".class"或者 id 选择器#id 都可以找到具体的标签，当然，找 id 是最精准的。

找到标签里的内容

```
$("#h1").text()
```

\$("#h1")会先找到 h1, .text()会取出 h1 里面的内容

改变 h1 里面的内容

```
$("#h1").text("new text")
```

在 text 里面直接输入新内容就好了

jQuery 第二句

等待 dom 加载完成之后再执行

要想等网页加载完成再执行 js, 只需要加上一句话, 我才这句话可能会跟固定公式一样:

```
jQuery(document).ready(function(){  
});
```

这样, 放在里面的内容就可以等待页面 ready 有再执行 js。

如果是像上文一样处理 h1, 那就写成

```
jQuery(document).ready(function(){  
    $("#h1").text("new text")  
});
```

其实用\$代替其中的 jQuery 也是一样的

jQuery 下载到自己的页面

在 Rails 里好像不需要自己做这些事情

但是, 如果你想要下载就用这句吧

```
<script src="jquery.min.js"></script>
```

你也可以直接把 js 语句写到 script 标签下面

[用 Google Authenticator 保护你的 AWS](#)

September 10, 2016

用 google authenticator 保护你的 AWS 账户

让根账户多一层校验

AWS 可以做到, 在任何人使用根账户 (就是权限最大的账户) 时, 需要输入 google authenticator 的校验码。而 google authenticator 的校验码又是绑定在你手机上的, 网上的 hack 很难知道这个校验码, 所以会更加安全。

操作方法如下:

第一步, 进入 AWS 的 [IAM](#) 服务



在 AWS 的 console 中可以找到 IAM, 在里面又可以找到 MFA, 就是图中展开的这个, 点击管理 MFA, 不断的下一步到出现二维码为止。二维码的下面会出现两个输入框。

第二步, 下载 google authenticator

在 Apple Store, 或者安卓市场里面有, 打开之后用摄像头扫码, 把生成的第一个 6 位数字和第二个 6 位数

字填入二维码下方的输入框即可。

效果

根据 AWS 的说法，用根账户访问的时候，AWS 会校验 google authenticator 上的校验码，给你的账户多一层保护。

[如何自定义 validate](#)

September 10, 2016

如何自定义 validate

validate 是什么？

validate 是 Rails 中 model 的校验，校验通过的数据才可以存入数据库。Rails 已经内建了一些 validates，比如 validates_presence_of。但仅靠内建的 validates 有时候无法满足需求，这个时候可以自建 validate，只需要一下两步：

第一步，告诉 model 校验方法的名字

比如，我在一个 model 里面要 validate 一种叫 check_self 的方法，就写上：

```
class FollowRelationship < ApplicationRecord
  validate :check_self
end
```

第二步，定义校验的内容

在同一个 model 里面写上 check_self 这样校验的具体内容，形式是这样的：

```
def check_self
  if user_id == follower_id
    errors.add(:user_id, "不能关注自己")
  end
end
```

校验的内容就是 user_id == follower_id，如果校验不通过，就让 model 增加一个错误信息 "不能关注自己"，这里的 user_id 是 FollowRelationship 的一个栏位，报错会指向这个栏位。

这样，你就增加了自己的校验

补充：打印错误信息

你可以在 view 里面打印这段报错内容，用 @follow_relationship.errors.full_messages 可以打印出错误信息(可能需要现在 controller 里面先 render 原来的 action)

更多例子：

在 xdite 的 repo (66kjobs) 中有同样的校验方法，代码如下：

```
class Job < ActiveRecord::Base
  validate :check_salary, fields: [:lower_bound, :higher_bound]
  def check_salary
    if lower_bound.blank?
      errors.add(:lower_bound, "最低薪水不能為空")
    end

    if higher_bound.blank?
      errors.add(:lower_bound, "最高薪水不能為空")
    end

    if lower_bound.to_i < 30000
      errors.add(:lower_bound, "最低薪不能低於 30000")
    end
  end
end
```



```
end

if higher_bound.to_i < 60000
  errors.add(:lower_bound, "最高薪要超過 66000")
end

if lower_bound.to_i > higher_bound.to_i
  errors.add(:lower_bound, "最高薪要能超過最低薪")
end

end
```

end

[同学们的 debug 姿勢](#)

September 8, 2016

同学们的 debug 姿勢

Lily

有时候，她会设置一些输出，比如在页面上打印一些中间结果，比如一个本地变量的值是什么，来检查这个中间值是不是自己想要的结果，如果是说明这个中间值后面的代码中出现了错误，再检查后半段代码的一些中间值，循环几次，就能找到错在哪里了？

后来，pry 在班里传播开来，她就更方便的设置断点，在 pry 中检查中间变量是不是 nil 了之类的，程序进了 if 还 else 之类的。

后来我发现了 better errors，可以检查报错点前后的值了，可以把跟 pry 结合一下。

国峰

我请国峰帮我结一个前端的 bug，他打开浏览器 console 开始用 jQuery 找 html 标签，我之前也见过 lily 用 console 这招，看了国峰解一遍更清楚这是在干什么。

老师

老师帮我解过一个表单的 bug，错误原因是我 form 套 form，导致 form 提交数据的格式有问题。这是一个低级的错误，但老师解我这个 bug 的思考过程可能是有章可循的。

老师先改了我的拼写格式（当时我在 simple form 下制作了一个表单），发现用了他熟悉的拼写格式也不能正常输出结果，然后就去我提交到的 controller。因为我的代码是抄老师的，所以也算是标准答案了，这步检查就过了。

然后老师设置了用于 test 的输出结果，发现了 simple form 并没有按照预想的输出，这就发现了问题，然后有目的的打开了检查器，看到了 form 的 html 标签，发现了问题。

我猜，拼写错误是新手最可能的错误吧，先找这个 bug 找到的可能性更大，然后再找逻辑的问题（用测试值找），然后是其他问题。

总结

感觉 debug 的思考过程是哪里错了，设置点来夹击，看看逻辑从哪里开始出错，找到第一个出错的地方，改。如果有第二个，再改。

而为什么错，多是打错字，没有考虑到一些可能情况（还是会有黑天鹅吧）。

September 7, 2016

在 heroku 部署后实现发邮件

我们用 ActionMail 来实现发邮件，其中用到的邮件服务器是 SendGrid。

前提

已经在 Rails 中建立了发邮件的 mailer 和 views

第一步，配置 Rails 环境

在 production.rb 这个文件中贴上如下代码：

```
config.action_mailer.default_url_options = { :host => 'online-ask.herokuapp.com' }  
# ActionMailer Config  
# Setup for production - deliveries, no errors raised  
config.action_mailer.delivery_method = :smtp  
config.action_mailer.perform_deliveries = true  
config.action_mailer.raise_delivery_errors = false  
config.action_mailer.default :charset => "utf-8"  
  
config.action_mailer.smtp_settings = {  
  address: "smtp.sendgrid.net",  
  port: 25,  
  domain: "heroku.com",  
  authentication: "plain",  
  enable_starttls_auto: true,  
  user_name: ENV["SENDGRID_USERNAME"],  
  password: ENV["SENDGRID_PASSWORD"]  
}
```

第二步，用 heroku 生成 SendGrid 账号和密码

在 Heroku app 里安装 SendGrid 插件，生成用户名和密码。

在终端执行这几句话

```
heroku addons:add sendgrid:starter  
heroku config:get SENDGRID_USERNAME  
heroku config:get SENDGRID_PASSWORD  
就可以发邮件了
```

其他，如果是别人的 heroku app

你也可以用刚刚生成的账号给别人用，打开别人的部署的 heroku app，里面有个 tab 叫 setting，下面又有一个 config，点开，输入你的 SendGrid 账号和密码。

如果你之前已经建立了触发邮件的机制，就去 heroku 的线上环境测试吧。

另外

你可以在 application_mailer 里面吧邮件发送源改成自己的邮箱了。sendgrid 就会把发送人改成你。

[de 什么 bug](#)

September 5, 2016

de 什么 bug 也是有选择的

今天跟 micheal 聊 bug 的事情，在遇到奇怪 bug 的时候我们怎么 de。

我自己的结论是，用我会 de 的方法来 debug，不要用我不会的，至少不要一开始就考虑自己不会的方法。

今天这个 bug 对我来说就很新鲜，找不到路径。

我熟悉的 bug 是什么，是打错字，最有可能的错误是什么，是打错字。

如果不是打错字，我能做什么，我什么都不能做。如果是环境错误了，我能做什么，我不了解环境，我几乎什么都不能做。当然，google 一下还是可以的。

但是环境很容易错误么？飞机掉下来确实很坑，环境错误确实很坑，但是这样的错误很经常发生么？不是吧。飞机掉下来的统计学比率是不是能预测飞机掉下来的概率？不清楚，妈的。概率没学好。我的中心意思是，有一些 bug 我好像不会 de，我可以学习，但是想要一下子 de 出来，可能概率不大。有一些 bug 的可能性更大，新手最常遇到的问题是什么？是打错字，用错语法，找错路径之类的，

[9/4](#)

September 5, 2016

9/4

css 中的 .desc .desc-last 和 .desc.desc-last 是不一样的

前者有空格，当第二个东西在第一个包裹下时，第二个能起作用。

后者中间没有空格，当两个东西写在一个 class 中的时候起作用

类选择器下还可以套标签选择器

这样，在类选择器包裹下的标签都可以用这个标签的规定样式

hash 的调用. 和 [:something]是一样的

都是调用，可以按习惯来，如果你已经有了具体的实例

不熟悉的语句

1. Tweet.find(4,5,6)
2. t = Tweet.new(status: "open") t.save
3. Tweet.order(:zombie) 可以不用双引号
4. t.attributes = {hash} t.save 这样也可以给 t 的栏位赋值，集体赋值
5. t.update(hash)最简洁的 update 方法
6. Query 是查询的意思
7. 数据库保存时回传的错误信息是 array，我们要用 t.errors[:status0] 来取出信息，不是因为 errors 是 array，而是因为 status 是 array
8. validates*presense*of :topic, message: "新的 msg"，这是一种写法，另一种已熟悉
9. t = Tweet.create(status: "XX", zombie: ash)就可以直接把这个 t 和一个 zombie 连起来，zombie*id* 自动填充。
10. 如果不是用 id 这个主键找，那用 find by 试试
11. erb 是嵌入 ruby 的意思
12. link to 是个 helper method link_to tweet.zombie.name, zombie_path(tweet.zombie) = link_to tweet.zombie.name, tweet.zombie #这也是直接引导到 show 页面的（是不是只有 show 页面可以这么做呢？yes）
13. confirm 可以直接加提示内容"Are you sure?"
14. Tweet 是 class，Tweet.all 是 array，tweet 是单个实例
15. 有一些 url 需要一个 tweet，这就是我们的 member
16. 直接写出 path，对于单层，干什么放在前面，edit, new 啊，之类的。然后是 resource 的名字，再加上 path，最后括号里可能需要有特定的实例。
17. web 上，是先进 controller 然后再进 view，有什么参数之类的先导到 controller，controller 再告诉 view
18. variable scope 的意思是变量范围
19. render action: 'status'
20. /tweets?status=Im dead, @tweet = Tweet.create(status: params[:status])这样就接了参数
21. /tweets?tweet[status]=Im dead, @tweet.create(status: params[:tweet][:status])这样可以找到两层参数下的参数，参数是个 hash，如果要生成双层参数，需要 user[status]，外层是 user，里层是 status。这一句可以写成@tweet = Tweet.create(params[:tweet])但是后者很危险，后者不如前者安全。因为

没有定义 tweet 能写什么参数，所以要规定一下

22. 要想让 controller 里的 action 和 son 产生 respond，需要见一个同名的.json 然后写上如下代码

23. respond_to do |format|

24. format.html #show.html.erb

25. format.json {render json: @tweet}

26. end

如果是要跟 xml 互动，需要建一个同名的.xml，然后写上如下代码

respond_to do |format|

 format.html #show.html.erb

 format.xml {render xml: @tweet}

end

当然，两者也可以合并起来

27. check_auth 这个名字好

28. 我直接在有了@zombie 的 controller 里面写了 redirect_to @zombie 了，然后也能跳到 show 页。另外，我明白了，new 页就是为了让别人写参数，然后好传参数到 create

29. only: :show 也可以，或许只有一个，或许有多个

[8/31](#)

September 5, 2016

8/31

except 跟 only 相反，在限制 layout 时

有时比 only 更简便

苛责自己的团队成员是否有用

想想他哪里有问题，引导他解决问题或许更有用。

新功能交给 lily 开发

交给最优技术的人来探索，小白可能都不不知道自己在探索什么，或许吧。

这是不是说，找高手做一些无法量化的东西

记录 article 浏览量，好像可以用 impressionist

<https://github.com/charlotte-ruby/impressionist>

修改 devise 可以看 devise 自家的介绍

<https://github.com/plataformatec/devise>

做 online ask 时，如果遇到金钱的问题怎么办？

我们的价值观应该是，思考长远，建立信誉。不要贪图用户的订金，该退就退。

对于用户的钱，我们的较直观是要对的起，人家想退就退吧。

可以把这种没有什么规矩的日记写成某日十条，20 条之类的

就像金鹏远一样

render template

可以引用整个页面，但是参数还是要给

理解@answer = Answer.find(params[:id])

show 里面没有@answer，那他是怎么调用@answer 的，原来是有 before action

因为页面上有路径的标识，用这个标识中的 id，可以摘下来标记@answer

<https://github.com/xingrowth/fullstack-course/issues/42> 这里

如果不写路径，simpleform 会自动给我传到 update 里

因为他 flash 了

原来 render 这么扯淡，渲染，连 url 都不变

在 render 的 url 中，数据都要重新写，因为他已经不是原来的 url 了

[8/30](#)

September 5, 2016

8/30

浏览器后退，但是不会刷新数据库

要想看最新的数据库，需要单独刷新

另外，如果要看判断的结果对不对，就在上面打印一些，哪怕打印 true 或 false 也行的。

unless 在不需要 else 的时候用的多

如果有 else，用 if 更容易理解

可以用 content tag 替换 html 标签

```
content_tag(:p, "Hello world!")
```

```
# => <p>Hello world!</p>
```

```
content_tag(:div, content_tag(:p, "Hello world!"), class: "strong")
```

```
# => <div class="strong"><p>Hello world!</p></div>
```

```
content_tag(:div, "Hello world!", class: ["strong", "highlight"])
```

```
# => <div class="strong highlight">Hello world!</div>
```

```
content_tag("select", options, multiple: true)
```

```
# => <select multiple="multiple">...options...</select>
```

```
<%= content_tag :div, class: "strong" do -%>
```

```
  Hello world!
```

```
<% end -%>
```

```
# => <div class="strong">Hello world!</div>
```

先写 div 等，再写内容，在写 class 或者其他属性，也可以在内容里面写上其他的 tag，这样就能够实现少用 html 标签，或者在不能用 html 标签的地方实际输出 html 标签

我今天看到了 lily 打开了一个 js 的 function

原本人家是关闭的，在人家 ui 的 js 说明中说了这些属性 allowAdditions，我们可以把这个是

tag 直接上，也有类似 content tag 的功能

tag ("input"……) 就会转化为 input 的 html 标签

程序员不适合工业化培养

要师带徒

[一个 pry 的代替方案](#)

September 4, 2016

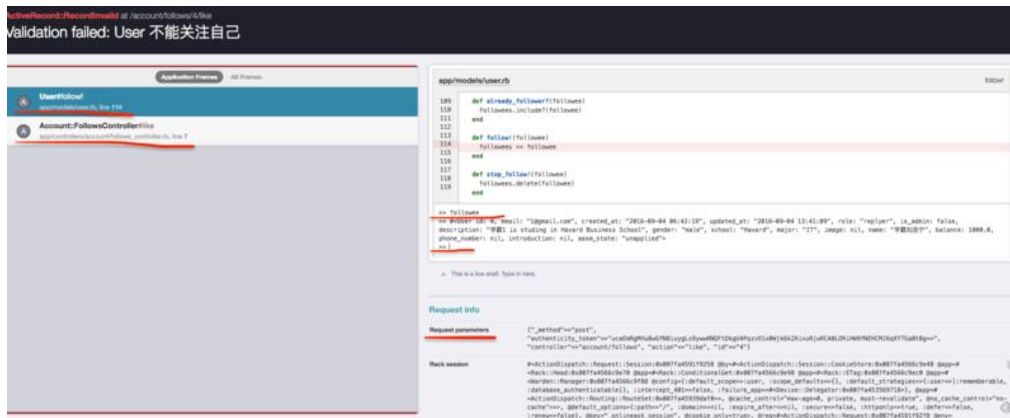
一个 pry 的代替方案

今天翻老师的 repo，发现了一个 gem 组合

```
gem "better_errors"
```

```
gem "binding_of_caller"
```

这两个 gem 的组合效果如下



主要功能是把报错信息页面变得不是一片红，然后在报错的位置可以检查实例变量和本地变量。（就在右边第一个红线的位置）。

有时候比 pry 断点方便一点，不用回到 project 里面加断点。

但缺点是刚开始用有点摸不到头脑。原来的报错页面会说什么 nil 了，现在不知道会在哪里报告。

Prefix 命名规则

September 4, 2016

prefix 命名规则（也就是 path 怎么写）

单层资源

单层资源是指在 routes 里面只写了一个 resources :posts

如果不需要给出特定的实例 id，有下面几种

new_post_path

posts_path

前者是 new，还没有 id

后者是 index，都展示出来，所以不需要特定的 id

不需要 id 的还有 collection，如果我建立一个 collection 叫做 bills 或者 water，那么 path 分别是

bills_posts_path

water_posts_path

也就是说，这里不需要 id 的 path 中，只有 new 的 post 是单数，其他的都是复数。

如果需要实例的 id，有下面几种

edit_post_path(post)

post_path(post)

写法都是把你要做的事情放在最前面，然后 post path

其他的 member 也是需要 id 的，比如你定义了 hide，就把 hide 放在最前面，然后来个 post path，具体来说，就是：

hide_post_path(post)

命名空间下的单层资源

这种形式是指：

namespace :account do

resources :users

end

对于这样的命名形式，请把 account_user 或者 account_users 看成一体，所有的复数、实例 id 都是这个整体的变化，如果有新的 collection do 或者 member do 就在这个整体之前加上，距离来说就是：

new_account_user_path

```
account_users_path
collection_account_users_path #使用 collection 的情况
```

#以上 path 不需要实例 id

```
edit_account_user_path(user)
account_user_path(user)
member_account_user_path(user) #使用 member 的情况
```

#以上 path 需要 user 的实例 id

总之，都是把这个 url 的主要功能放在最前面，然后加上 account_user 这个整体（user 有可能是单数，也可能是复数），最后加上 path。

双层资源

双层资源是指这样的形式：

```
resources :users do
  resources :posts
end
```

这种情况下，**作为外层的 user，永远都只是单数**，里层的 post 有可能是单数，也可能是复数。但是只要外层资源和内层资源一起出现，外层就永远需要实例 id，也就是 user_id。

这看起来稍微复杂一点，但是只要把 user_id 给了外层资源，外层资源就会像 namespace 一样，可以和内层资源拼在一起看了。具体来说就是：

如果不需要内层资源的实例的 id

```
new_user_post_path(user)
user_posts_path(user)
```

这里只要使用内层资源，就必须给出外层资源的实例 id。可以理解为内层资源都是住在外层资源里的，不先找到外层资源，就找不到内层资源。

这时候，如果是内层资源定义了 collection，比如一个叫做 water 的 collection，那么 path 会是 water 在最前，posts 是复数，组合成：

```
water_user_posts_path(user)
```

总之，除了需要给出 user 的实例 id，其他的更 namespace 的命名方法一样

如果需要实例的 id

```
edit_user_post_path(user, post)
user_post_path(user, post)
```

这里就需要把内外两层的实例 id 都给出来。

如果给内层资源定义了 member，path 会是这样

```
member_user_post_path(user, post)
```

其他

1. 对于双层资源，好像也有一些路径可以放在两层资源中间，但是一般都报错，不知道能做什么
2. 有没有不加 member do 或者 collection do 的情况呢，有，在别人写错的时候，不知道具体能做什么
3. 有没有三层资源呢，rails guides 说不要这样做，这样等于把资源藏的太深了，url 可能会找不到资源的。

[如何更改 devise 注册、登录后的跳转](#)

September 1, 2016

如何在 devise 中更改登录、注册之后的跳转

1. 更改注册后的跳转

新建一个 controller

```
rails g controller registrations
```

然后通知 routes 去这个 controller 里面去找 action，而不是原来的 devise 那里。

```
devise_for :users, controllers: { registrations: "registrations" }
```

让这个 controller 继承 devise 的 RegistrationsController，并定义注册后要跳转的路径。

```
class RegistrationsController < Devise::RegistrationsController
  protected
```

```
  def after_sign_up_path_for(resource)
    your_path
  end
end
```

这样，注册后，用户就会跳转到 your_path

2. 更改登录后的跳转

在你的 application controller 里写上登录后的跳转路径，写法如下：

```
def after_sign_in_path_for(resource_or_scope)
  your_path
end
```

这样，登陆后，用户就会跳转到 your_path

3. 参考

https://github.com/plataformatec/devise/wiki/How-To:-redirect-to-a-specific-page-on-successful-sign_in,-sign_out,-and-or-sign_up

[https://github.com/plataformatec/devise/wiki/How-To:-Redirect-to-a-specific-page-on-successful-sign-up-\(registration\)](https://github.com/plataformatec/devise/wiki/How-To:-Redirect-to-a-specific-page-on-successful-sign-up-(registration))

更多 devise 修改方法可以去看 devise 的 github

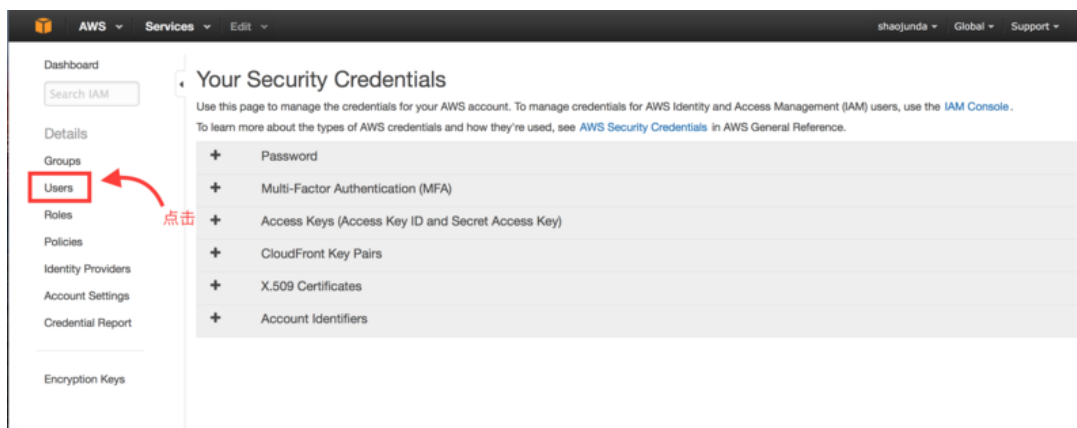
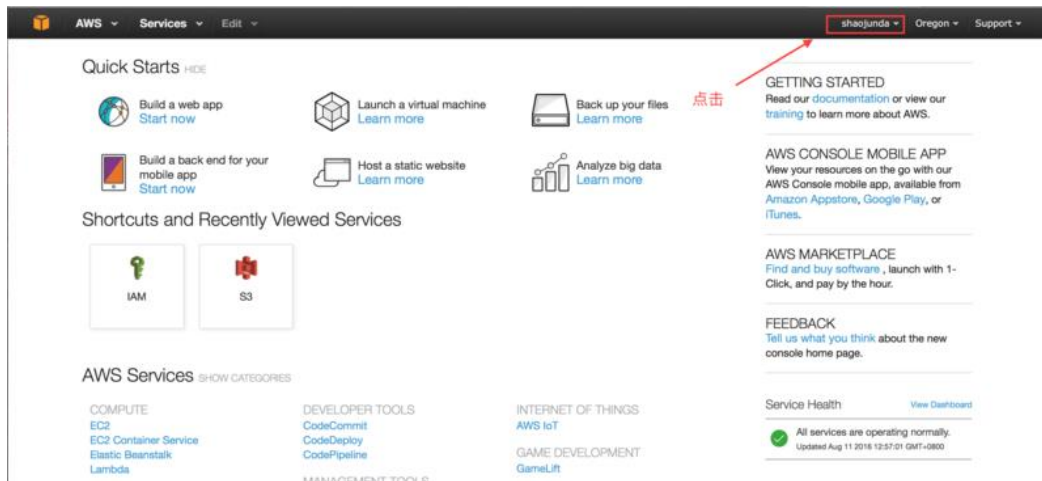
[AWS key with only S3 permission](#)

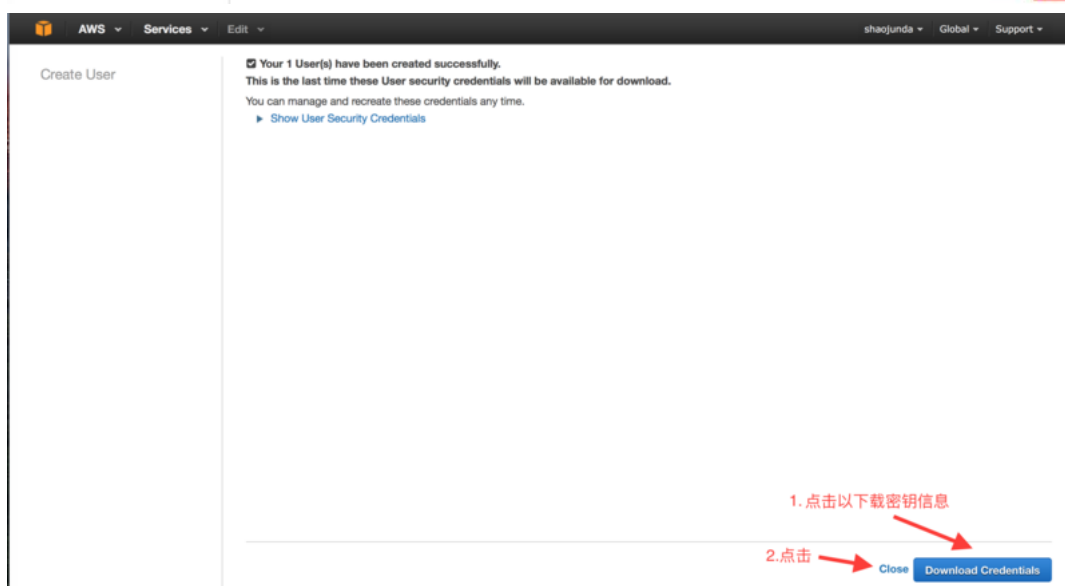
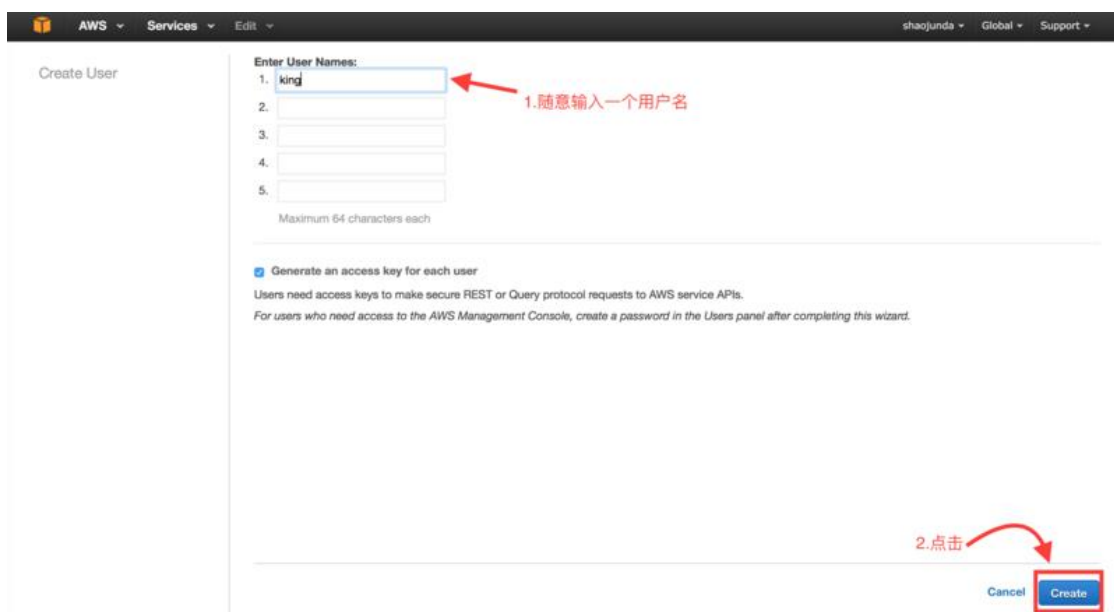
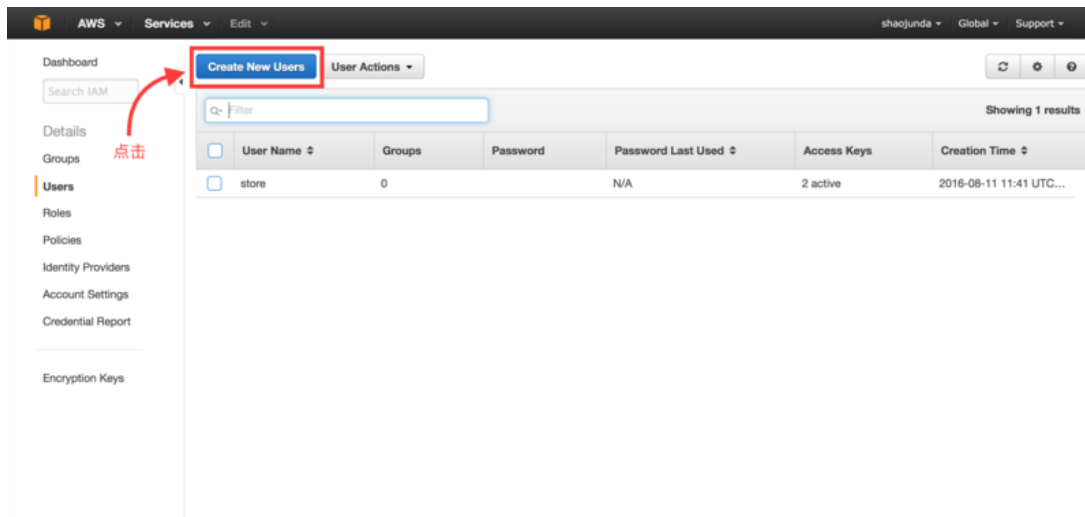
September 1, 2016

To protect your credit cards on AWS, you can use a was key that only permit S3 service access. You can do this through following steps.

First, create keys.

("点击" means "click", "输入" means "input")





You can save AWS_ACCESS_KEY_ID 和 AWS_SECRET_ACCESS_KEY through downloading this credentials.

You can put the new keys in application.yml(ex. in Rails). But remember to put config/application.yml in .gitignore. What's more, to ensure git isn't tracking your application.yml, run the followings:

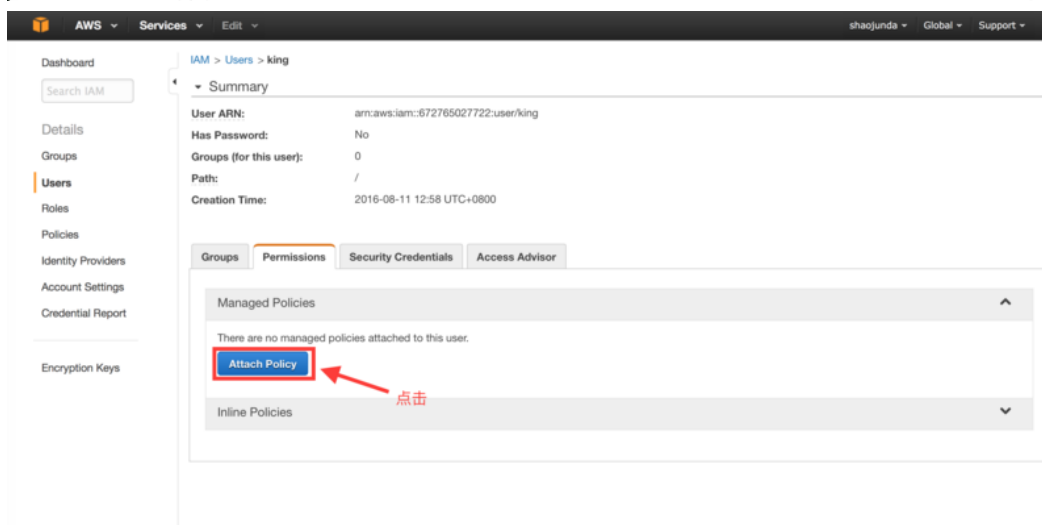
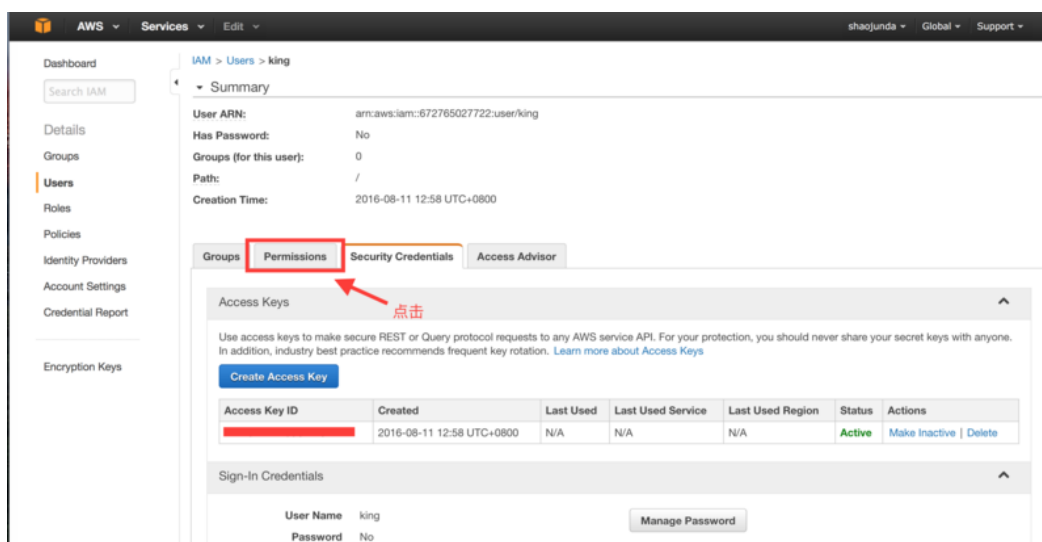
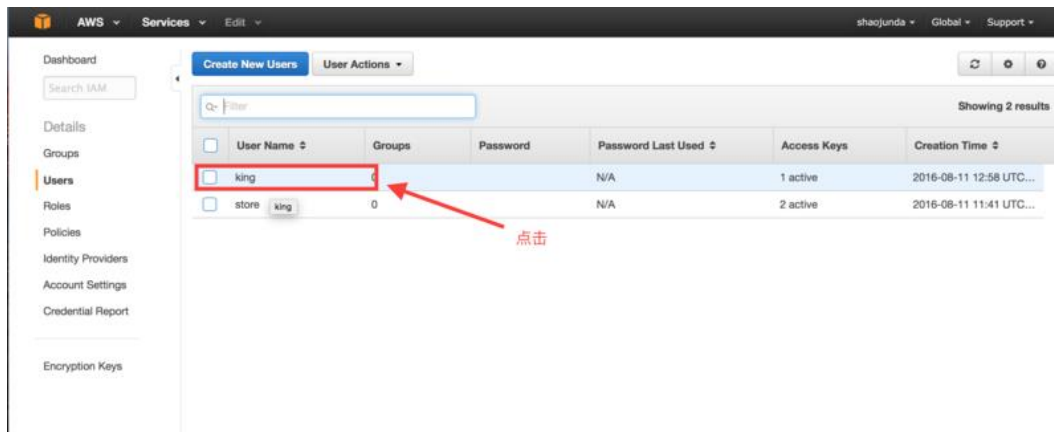
```
git rm -r --cached .
```

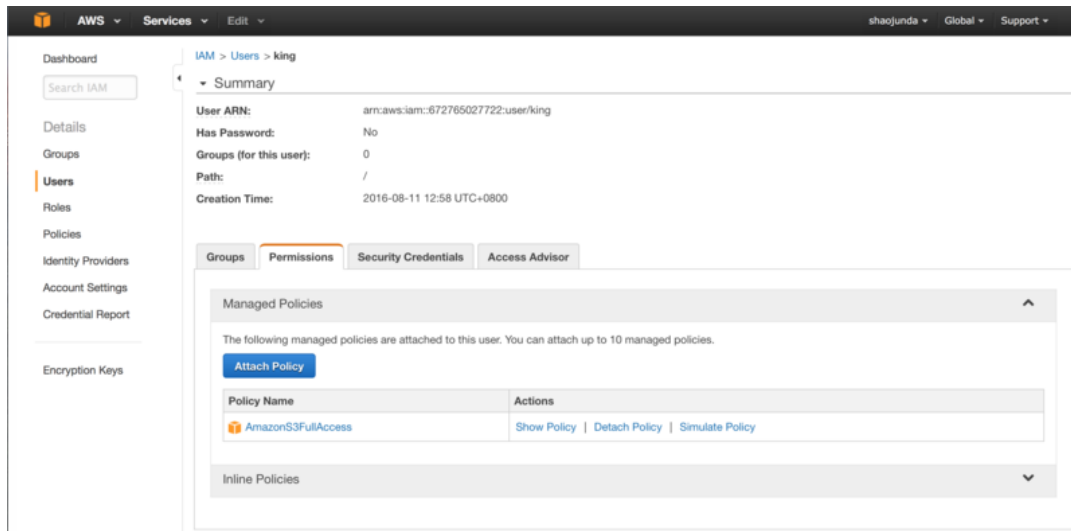
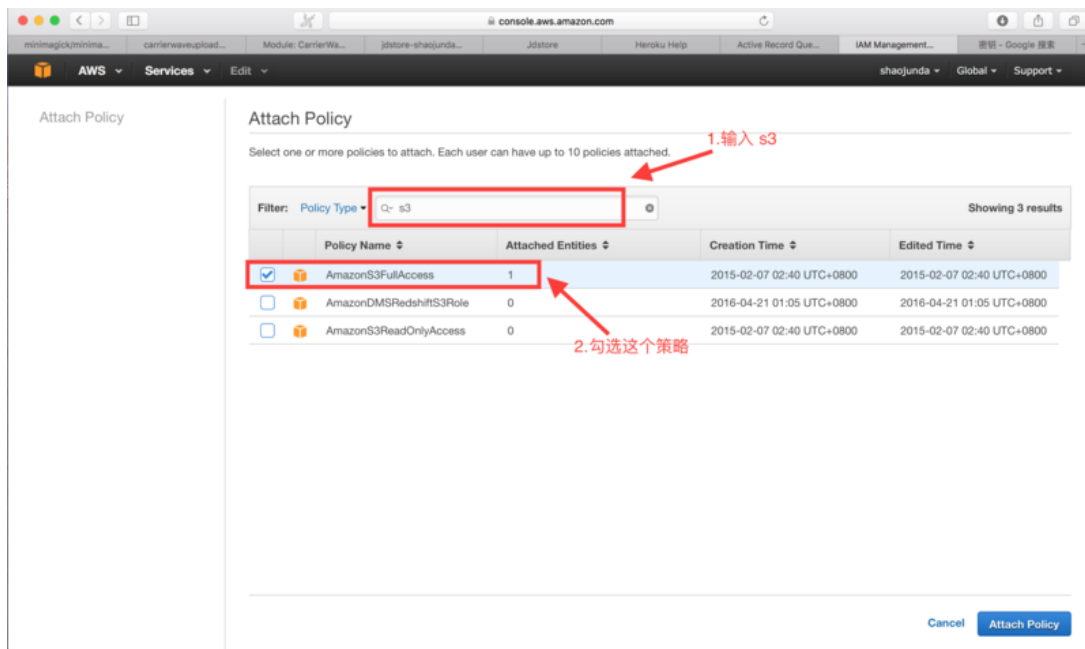
```
git add .
```

```
git commit -m "fixed untracked files"
```

For more details about cleaning git tracking, see the [StackOverflow](#).

Then, limit the permission of the user.





Now, you have a user with only s3 access.

如何封装 service

August 30, 2016

如何封装 service

第一步，建立 service 文件和类

touch app/services/your_service.rb

其中，your_service 是你的文件名。

然后打开这个文件建立这个 service 的类

```
class YourService
```

```
end
```

类名和你的文件名要相同，只不过用 camel 的写法。

第二步，初始化这个类

```
class YourService
```

```
  def initialize(var1, var2,)
```

```
    @var1 = var1
```

```

        @var2 = var2
    end
end

```

这里的 initialize 就是在初始化，初始化的内容我们一会调用的时候会用到。

第三步，定义你要用到的方法

比如我们在提问押金 service 时定义了一个 perform! 方法

```

class RewardDepositService
  def initialize(user, invitated_users, question)
    @user = user
    @question = question
    @invitated_users = invitated_users
  end

  def perform!
    amount = @question.downpayment
    @user.deposit_money!(amount)
    @user.super_admin_bill!(amount)
  end
end

```

这个 perform ! 方法里写上我们要操作的逻辑。这样就不用把逻辑写在 controller 里面了。(如果你已经写在 controller 里面了，可以把 controller 里面的语句搬过来)

第四步，调用这个 service

在你使用 perform! 方法的位置写上这句话：

```
RewardDepositService.new(@user, @invitated_users, @question).perform!
```

这句话中的 new 就是在往 initialize 传参数，所以两者用到的参数要对应。传好了参数就可以调用 perform!。所以上面这句话会输出 perform! 方法里面写的语句。

参考

为什么要用 initialize 初始化呢，请看《Ruby 基础教程》第 91 页，但是这里先不要看了。

August 29, 2016

实验发送邮件需要在 console 中

我在 term 中直接用会报错，说找不到

找不到资源的报错

1. 可能是你把资源的名称打错了
2. 可能是你把调用资源的对象弄错了
3. validate 之后验证了一个不存在的东西

mailer 里面能用 url 的路径但是不能用 path

用了 path 说找不到，不是 id 的问题

mailer 如果在 mailer.rb 中设立循环找人，发送多封，可能有问题

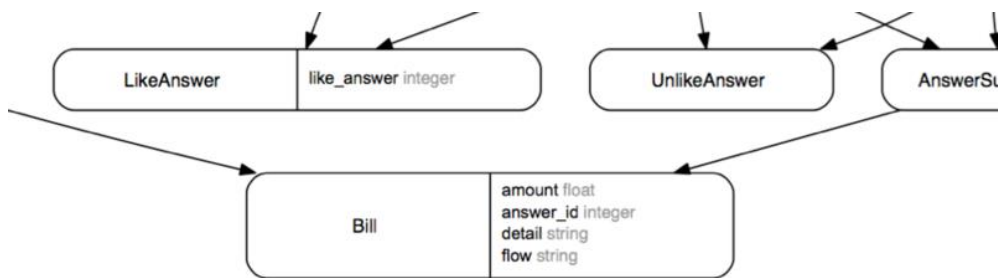
因为好像 mail 只能记录一个，还是说预览的工具只能记录一个

反正在 dev 下，只能看到一个

该用三个 order mailer 的调用后，能够实现 3 个。

或者我们想，只有一个 deliver 命令，所以不能寄出 3 个对么？

如果不用 has many, belongs to 关联 model，引用的其他数据的 id 无法自动找到



如果关联的话，这个栏位会在 erd 里面自动消失，代表 rails 知道这不是一个自己写数的栏位
 如果不关联，要想写 id 栏位，就只能用 id 来取
 另外，我想到，如果用 foreign id，就能让一个表同时存两个 user id 之类的，只不过把另外一个 id 当做 user 这个 class 的 foreign key

[js 检查和 js 从别的标签接受内容](#)

August 29, 2016

js 检查

```

$(function() {
    alert($(".answer-time").attr("class","answer-like-stats").attr("class"));
});
    
```

alert 能弹出消息，显示信息，在 function 下面，就是刷新页面的时候 alert 信息
 最简单的是，打印 1，主要看看进没进如这个 function

```
alert(1)
```

然后可以

```
alert($(".a-class-name").html())
```

这样就是提醒你这个类下面控制的所有 html 内容

然后还可以打印这个类下面的属性

```
alert($(".a-class-name").attr("class"))
```

然后还可以把一个属性换为另一个属性

```
alert($(".a-class-name").attr("class", "another-class-name"))
```

这样，在载入页面的时候，js 会自动把属性换掉，当然，你本地的代码是不会变的。

最后，如果你想查看，被 js 更改了更改了之后的属性，可以用另外一个 attr

```
alert($(".a-class-name").attr("class", "another-class-name").attr("class"))
```

js 允许 html 标签

```

$(function() {
    $('[data-toggle="popover"]').popover({
        html : true,
        content: function() {
            return $('#popover_content_wrapper').html();
        }
    });
});
    
```

.html 这句就是在回传 html 语气，整个 div 都是 html 语句，就回传了

popover 是 bs 的一个 js，他本来是不允许在这个 js 里面使用 js 标签的，你可以用上面的方法把他打开，关键是 html: true 这句话。popover 默认的可能是 false，用这句话就打开了，如果你不想所有的 popover 都打开，你可以设置一个小小的 id 选择器来使得应用这个 id 的 popover 可以接受 html 标签（请注意，如果你

在别处也设置了 popover，不要让两个 popover 冲突)。

方法是：

```
$(function(){
  $('#an-id-selector').popover({
    html : true,
    content: function(){
      return $('#another-id-selector').html();
    }
  });
});
```

这样，写上#an-id-selector 的 popover 就都能接受 html 标签的，而不是把 html 标签当做文本来处理。

最后，注意这个 content 的部分，他更 popover 里面的属性 data-content 似乎是关联的，这句话是在手回传另外一个 id 选择器#another-id-selector 中的内容，当然这种内容是 html 内容。

[初学者的 Rails 黑箱](#)

August 21, 2016

初学者的 Rails 黑箱

什么是黑箱

Rails 对于我来说是一个黑箱，对于其他人也可能是黑箱。

黑箱表现为什么情况呢？

就是我在一定程度上知道输入什么数据就能得出什么结果，但是我并不知道黑箱内部的运作机理。

之前看过一个黑箱的示意图，你以为小球从黑箱上进入，从黑箱下出来，中间是直着走的，但后来无意中在左上角发现了它，所以猜想小球是先去左上角然后再到黑箱底部；再接下来在黑箱的右上角又发现了小球，然后又结实说小球是怎样怎样的。

多于自然系统，人们的理解过程有一些就是这样的（可能很多，但我只知道一些）。比如化学，我学过无机化学中关于电子轨道的理论，能够解释一部分化学键的形成工程，但是有另外一些不适用，所以又有人设想另外一套模型。生物也是这样的，人们以为水是被磷脂分子包裹起来进入细胞的，但是后来看到了水分子通道就有了新的模型。

所以，黑箱就是你可以在一定范围内观察到输入和输出的关系，但是因为你不了解里面的机制，如果你不小心输入跟以前不一样了，输出的结果可能就是你不能理解的了。

正式因为人们还没有完全看透这个黑箱，所以各种让人们看透这个黑箱的方法都存在着，没有统一的答案。宗教说，黑箱是神，求神拜佛吧，这样改变维持或改变输出结果。科学说，探索吧，这样能越来越了解这个世界的运行机制。最近接触到一种新的观点，叫：肌肉联系吧，先掌握这种观察再说，没有观察，连最基本是输入和输出的联系都还没建立，你研究什么黑箱的内部。

Rails 乃人工黑箱

我觉得在一定范围内，人们还是能够了解到黑箱的运作机制的，比如用电子显微镜看到了细胞膜的结构，那就是让人们更加了解以前是黑箱的东西。

但是自然系统和人工系统还有一个区别。人们都不知道自然系统是如何开头的，只能从中途开始找，通过中途观察，在历史的这个时期，在短短的 400 年来（工业革命以来），不断的观察自然系统，能了解多少还是个未知数。因为自然这个黑箱到底有多庞大，人类能不能真正理解都是未知数。

但人工系统就不一样，人工至少是人做的吧，至少有人已经能理解了对吧。那我作为一个人就有可能完全理解，让黑箱变成透明箱。

但是，把 rails 这个黑箱变成透明箱真的重要么？这种可能的代价有多大？虽然我也是人，rails 的构建也绝对在人类是知识边疆之内，但是，我跟那些 rails 的构建者的差别有没有可能就是我跟造物主的差别，rails 的复杂程度有没有可能让我这个初学者看来就是跟自然系统有相近的复杂程度（比如 rails 的复杂指数是 100，

自然系统是 10000，反正都挺复杂)。那我作为一个初学者，我就有了两种方法来学 rails 啊。一种是像人类研究自然一样，观察，观察，连接输入和输出的规律，有了积累之后再去找个理论揣测黑箱的模式；或者，直接了解黑箱，了解完了黑箱，自然能够判断输入和输出的规律。再补充一下，作为一个人工系统，他的模式都不用去揣测，去看说明文档就好了。要想知道一个黑箱的全部之后，再去看输入输出，有个问题是你可能知道了有什么输入之后就有什么输出，但是你可能不知道要输入什么，或者，你不知道到在人世间输入什么比较有价值（这是我猜的，我想到的是我在大学里学的无机化学，学的理论挺多的，但是我还是不知道要干什么，不知道如何把这个知识产生价值，当然也可能是因为我学的还不是特别精通）。

现在 xdite 在带着我们做的事情呢，就是对一个人工系统（复杂的人工系统）做类似于科学观察的黑箱探索，不是把 rails 内部结构全部摸透，而是把 rails101 做 3 遍，观察输入和输出的模式，你就知道自己写这段代码是干嘛用的了。

为什么说这些

因为今天我在用 pry 的断点来 debug 的时候，让程式一句一句的 next，结果 rails server 走到了很多我不熟悉的角落，也就是说，rails 这个黑箱里，我把小球丢进去，小球确实从下面出来了，但是中间的过程我还不不是很理解。

想理解看技术文档，在有了观察结果后，带着目的去看，是不是记得更牢固呢？

Rails 已经难道让我们把他当做黑箱去学习，那更复杂的人工系统呢？

rails 绝对不是一个人开发的，每个人都添砖加瓦，最后 rails 的复杂程度，包括生态圈的规模，不是任何一个单独的人能够理解的了。即使是老司机，也不可能知道 rails 的一切，至少，那么多 gem，不会有人全知吧。

expert 看 rails 可能就像一个透明箱，但他不会是对每个领域都 expert 吧，他在学习不熟悉的人工系统时必然也像黑箱。

世界上有这么多人工系统，而且很多系统都是很多人，经过很多年构建起来的，要想通过一己之力把黑箱看成透明箱，应该会很难吧。用重复三遍来观察系统的方法应该是一种路，他通往的可能不是理解全景，而是，在一个小范围内，让你能用，就像人们对自然系统一样，能够预测一部分，然后利用这部分的预测，进行生产（比如化工）。

[编程语言的历史](#)

August 20, 2016

程序语言的历史

几天前我刚刚知道 mvc 就是 model, views, controller，跟 YY 分享了一下之后，他告诉我 MVC 听起来很高大上，但是现在看起来很简单。但是的但是，这在几十年前又是很新的技术。也就是说，编程语言是在不断改变的，这有点像，今天的汉语中包含很多新词，在 10 年前是没有的，比如 hold 住，比如狗带，或者有一些词有了新的意思，比如老司机。

但是编程语言的变化可能不只是加了几个新词（汉语也不只是加了几个新词吧，也有结构上的变化么？但我不知道），还增添了很多框架，模块，库。其中一个框架就叫做 mvc，在 ruby 上的，这种框架的体现就是 Rails。

这是我想到的，关于编程语言历史的冰山一角。一定还有很多编程语言的历史。我在 amazon 上搜索了一下，编程 + 历史，跳出来的是一个儿童编程学习书，看来还没有相关书籍，这让我系统了解编程历史的难度增加了，但是，话说回来，我可以自己写一本书了！……如果我能够多了解一些编程历史的话。忽然感觉工程量浩大，不如没发现一点，就写一篇小文章，把这样一个工程拆解成无数篇小文章吧。

今天也正好看了一篇文章，是知乎上一个人在讲他看到的编程历史。能让我确认的是，有一个给编程语言排名的机构是 TIOBE，我不知道为什么要给编程语言排名，但说不定这种排名能间接的体现编程语言的市场大小。另外，我还了解到，编程语言谁都可以用，但是他们也是有管家都，他们有协会在“管理”，或者叫做“发展”。另外就是很多语言都产生了模块和库，下载时候可以调用，这里提到了我用过的 BS，还有听说过的 jQuery，分别是 CSS 和 JS 的库和框架。作者还在这里发表了一个观点是：现在，程序员可以通过做一个方

便大家的库或者框架来提高自己的声望，而不只是做一个软件。这和 xdite 在讲如何程序员 social 中讲提到的做 open source contribution 有点像。是一个方法，不过我还没到考虑这种方法的 level。继续学习吧。我想，这大概是在随着互联网发展之后逐渐产生的。图灵时代的计算机没有互联网，不会有下载，做了库也没用。未来，会有更多更好用的库和框架么？

再接下来，这篇文章提到 2010 年之前，程序员们（哪里的程序员，真的么？）并不是很依赖管理工具（我想，具体是指 brew 这样的编程工具的管理工具，管理版本之类的），大家都去官网下载，但是现在有了版本管理工具，输入几个命令，然后管理工具帮你下载，好像有的工具还能帮你更新 project 中用到的库，构架之类的（比如 brew 帮使用者更新 rails 版本）。

最后，这篇文章中还提到测试代码，我在人人都是产品经理中听说过 TDD 代码自动测试这样的事情。但是 Xdite 还没有教，不知道会不会教，我准备问问。

最后之后，

1. 跨平台跨设备也被这篇文章的作者认为是编程语言的发展方向之一。
2. 产生了新的工具帮开发者价差代码，美化代码（排版，去重），比如 atom - beautify 么
3. 版本控制，部署什么的更加自动化了，比如部署到 heroku 么？
4. 版本控制有了新的工具 git，包括 merge 什么的，在大量代码的情况下一定会比人工检查快吧。
5. 程序员会看中编程语言的生态圈，如果这种编程语言没有什么框架和库支持，也就是说生态圈比较小，使用者、贡献者比较少，他们就不会选择这种语言（我觉得使用者也可以看作一种贡献者，毕竟，如果身边有很多人用同一种语言，问问题都方便），这就好比 ruby 的 gems，python 的爬虫对么？
6. API 让 web 服务互相连通，据作者说，这是 twitter 使用 web api 之后认识到 api 的人多了，大家可以互相调用服务，所以要编程语言界多了写 api 的工作。
7. 另外，就是有了比较好的编辑器，能够提示代码之类的
8. 编程语言之间互相借鉴有点，比如没有框架的语言要搞出一个框架来。
9. 编写可维护的容易阅读的代码在工作中很重要，因为方便合作。

总结

本节的话题是编程语言的历史，现在的印象是，编程语言界一开始生出了 java, ruby, python 这样的语言，这样的语言又生出了框架和库，方便 project 使用，然后才有了管理工具，方便使用者更新这些框架和库，又有好的编辑器，版本控制器来帮忙写代码，团队合作之类的。最后还有自动化测试等。

最后附上参考[文章链接](#)

[8/16](#)

August 16, 2016

8/16

删除 dir 内指定文件

```
rm -rf xxx
```

把远端拉下来，要 git pull

拉下来还要 bundle install 和 migrate

我要做什么

我要做的是 admin 的 crud，那么我要做的。那么我要有一个页面，来显示问题，显示答案，然后露 btn 来 hidden 他们，那么我就需要一个 controller，跟这个页面连接着，我可以在 admin 的 namespace 下建两个 controller 分别叫做 q 和 a。

那我在 view 里面怎么呈现呢，我现在只想建立一个 index 页面，这个 index 页面有问题和答案，反正都在 admin 下面，所有就变成了 resources : admins do 还是 admin do。应该是 admin do，但是他能找到这个 controller 么？

如果本地的 master 和远端的远端的 master 有冲突，会显示远端比我靠前一些

这个时候我需要把本地的 git stash 移动到一个缓存中

然后 git status 就太平了

这样就可以把远端的 git pull 下来了

之后呢，我让不太平的 git stash pop，就是把缓存的冲突跳出来，这样就会显示哪里冲突了。改一改。

然后 add，commit。再 push 上去。

不能 merge 的解决办法

如果不能 merge 呢，说明有冲突，方法是

把远端的代码抓下来。git fetch origin

如果本地没有正在 merge 的这个分支，就新建，并 checkout 过去。git checkout -b XXX origin/XXX。其中 XXX 是正在 merge 的这个分支名。这句话能让你进入 origin 中的 XXX 分支。

然后要 merge。git merge master。

应该会报告冲突。但是没关系，看看 git status，就能知道哪些是冲突的，改正。然后把 commit。如果这个时候把所有的冲突都改了都话，git push origin XXX 后，github 会自动更新，在网页上告诉你能 merge 了。

如果你不想改变远端的 XXX 分支。就执行 github 告诉你的方法：

1. 切换到 master。git checkout master （如果本地没有最新的 master，要补一刀 git pull 一下）。
2. 然后 merge。git merge --no-ff XXX。如果还有冲突就接着改一下。改了之后 commit
3. 最后推到远端就好了，git push origin master。github 应该会自动显示 merge 完成。

目前用到的 git 的 merge 方法

August 16, 2016

目前用到的 git 的 merge

如果自己的 master 跟远端的 master 有冲突，怎么办？

如果你修改了 master，远端也有人修改了 master，这个时候，你把远端拉下来的时候就可能有冲突。冲突会这样发生：

git pull origin master 的时候说有冲突。

这个时候，采用以下处理步骤：

1. git stash 这句话让你把自己的 master 藏到一个缓存中。
2. git pull origin master 这样就可以把远端拉下来，而且不冲突。
3. git stash pop 让缓存的 master 跳出来，修改冲突
4. 修改之后 git add . / git commit -m "XXX"就好了。

如果要把 master 合并到你的分支（让你的分支有最新的成果），怎么办？

切换到自己的 master，git checkout master。

然后：

1. git pull origin master 先确保你自己的 master 是最新的，如果遇到冲突，看上一步。
2. 然后回到你的分支 git checkout XXX
3. git merge master 就把 master 上有的成果合并给你了。

如果在 master 上合并大家的分支时，遇到不能 merge，怎么办？（其实 github 页面上会自动提示下面的语句，不用记）

1. 把远端最新的代码抓下来。git fetch origin/XXX。这个命令在本地建一个 XXX 分支的缓存，把远端的 XXX 放到缓存了，如果你用 git branch -a 就能看到了，用 git branch 看不到。
2. 第二步，进入你要 merge 的分支。语句是 git checkout -b XXX origin/XXX。这句让你 checkout 进入刚刚下载的缓存。
3. 然后就 merge 吧。用 git merge master。
4. 应该会报告冲突。但是没关系，把这些冲突改正就好了。然后 commit。如果这个时候把所有的冲突都改了都话，git push origin XXX 后，github 会自动更新，在网页上告诉你能 merge 了。

如果你不想改变远端的 XXX 分支。就执行下面的方法：

1. 切换到 master。git checkout master（如果本地没有最新的 master，要补一刀 git pull 一下）。
2. 然后 merge。git merge —no-ff XXX。这句话是说把分支 merge 进 master（上面有一处是把 master 合并进分支）。
3. 最后推到远端就好了，git push origin master。github 应该会自动显示你已经 merge 完成。

备注：git pull 是 git fetch 和 git merge 的合集。git fetch 只是把远端拉下来，放到缓存里。git merge 只是合并。但是 git pull 就把两步都做了。

[8/15](#)

August 16, 2016

8/16

转眼间竟然 8 月 16 了。

还有一个月就要结训了。

看到一个不加 member do 的后果

如果不加 member do，传参数走的是 order id。如果加上，参数走的是 id，而且，因为这里的 id 还是 token，所以在找参数的时候用的是 find_by_token(params[:id])，就是说参数里面的 key 还是 id，只不过 id 取出来要用 token 找。

拿为什么加上 member do，传的参数会变呢，因为在 resource : order 下面么？如果因为在 resource : order 下面，可以解释为，跟 order 有关的就走 id 来传，没关的就走其他的 id，比如 group id 来传对么。那我上次把 cart item 传到 cart 的 controller 里面，走的是什么 id 呢。答案是，走的就是 id，routes 一定以为这还是 cart 的 id 吧。所以直接给我传了，但是我到了 controller 里面，我却用这个 id 来找 cart item。

[scope 是什麼？怎麼用](#)

August 14, 2016

如何使用 scope

封装半句话

昨天完成，我已经发现了两种用函数封装的方法，一种用 self，调用的时候就让 self 知道你是谁。另外一种就是传参数，把自己当做参数传进去运算。今天还有一种，就是封装半句话，叫做 scope。

可以放方便你以后改条件，也可以用来当做懒人包。

使用方法

带参数的使用方法

scope :test, -> (var) {where('? > 1', var)}, 这样别人就可以调用 test

或者更高级一点

scope :test, -> (var) {where('created_at > ?', var)}, 这样也可以吧

还可以不带参数

scope :test, -> {where(status : 'open')}, 就封装好了。

scope 我见过写在 model 里面，然后用在 controller 里面，帮助 controller 查表。

scope 只是语句替换，可以连着用。

api 中给的例子是

```
class Shirt < ActiveRecord::Base
```

```
scope :red, -> { where(color: 'red') }
```

```
scope :dry_clean_only, -> { joins(:washing_instructions).where('washing_instructions.dry_clean_only = ?', true) }
```

```
end
```

[第三周遇到的最大的坑](#)

August 14, 2016

[delete all 與 destroy all 的區別與用法](#)

August 14, 2016

[8/11](#)

August 13, 2016

8/11

我以前也有删掉数据库的烦恼，现在想想可以遵循这个步骤

1. rake db:rollback 返回上一版
2. rails d migration 防止错误的表再次被 migrate 进去
3. 如果这个时候 rake db : migrate, 是没有表格被 update 的
4. 如果 rails g migration 正确的表之后再 migrate 就有新的了。
5. 如果全删就 rake db : drop, 然后看看 sechma 怎么变

heroku 数据库爆炸，找不到数据怎么办

可以尝试 reset 一下 pg DATABASE。因为前后上传的数据有冲突。

stack overflow 上的同款问题是[这里](#)，反正

heroku pg:reset DATABASE

heroku run rake db:migrate

之后就好了，数据全部新建就正常了。

另外，据说 carrierwave-aws 这个 gem 能帮助我们很快的上传图片，他们的 github 在[这里](#)

而且，heroku 的 logs 信息跟 rails 的 log 有点不一样，可以用 cmd + f 来查

如果保护自己的 access key

key 要写在程式里，但是程式会 commit 进 git 里，heroku 部署的时候也要用到 git，这样别人看我的 repo，就能看到我的 key。

要想躲避 git，就用 gitignore，但是会影响上传 heroku，所以要想办法绕过 git，把 key 上传到 heroku，因为 key 是一个 config，所以有办法给 heroku 直接设置 config 的参数来不通过 git 上传 config。heroku 的官方讲解是[这里](#)

上传 git 后，我也第一次查了自己的 repo，没有 access key，放心

原来 bootstrap 可以看更多的 css

我之前看的一直是高级的 css，但其实有基础的 css，老师用的 class 就是。我之前一直很困惑，但是在给谢平讲的时候发现了 navbar 上可以跳转到 css 和 js。

[heroku 上管理密码](#)

August 11, 2016

heroku 上管理密码

heroku 是通过 git 来部署到，如果要想把链接云服务的 access key，比如 aws 的 key 上传到 heroku，是不是一定要经过 git 呢？然后在 push 到 git 上之后，让别人都能看到你的 key，再然后就能用你的 key 来访问你的云服务。并起来是不是很可怕。

所以要想办法不通过 git 来上传密码。要想在管理版本的时候让 git 忽视掉密码，就要用到 ignore。

所以第一步：

把你的 config/application.yml 加入到.gitignore 里面，然后把 key 写在这里面。比如：

AWS_ACCESS_KEY: xxxxxxxx

AWS_SECRET_ACCESS_KEY: ooooooooo

这样，commit 的时候，你的密码就不会被 git 收录。

但是为了告诉你的合作者，你其实是有 application.yml 这个文档的，就需要 copy 一份前者放在 git 里，但是不写机密的 key。

所以在第一步之前，应该有个第 0 步，要：

cp config/application.yml config/application.yml.example

这样就有一份空的 config 文件在那里。

之后呢, 利用正常步骤把专案 push 到 heroku 上就好了。但是这个样子的 heroku 上的 app 不知道你的 access key, 没法访问云服务。这里可以用 figaro 来传 config 的参数到 heroku 上, 这样 heroku 就知道你的配置了。

方法是：

安装 figaro, 可以看她们的 git, 上面有更多的说明 [case](#)

而上传 heroku 只需要先安装 gem 'figaro', 然后：

```
bundle install
```

```
figaro install
```

再让后让 figaro 帮我们上传 config：

```
figaro heroku:set -e production
```

最后再 commit, 再 push 到 heroku (并且 migrate 数据库) 让配置生效。

而怎么样获得 was 的 key 呢, 在 was 的 s3 服务里面找到 bucket (或者叫存储桶), 创建一个, 然后在 security credentials 里面创建 key 就可以了。

总结一下：

1. 注册 aws, 创建 s3 bucket 和 access key
2. cp config/application.yml config/application.yml.example
3. 在 config/application.yml 里面写上你的 access key, 并在想要引用的地方引用
4. 把 config/application.yml 加入到 .gitignore 里面防止被 commit
5. 安装 gem figaro
6. 创建 heroku, 用 figaro 上传 config, 然后 push heroku

最后的最后, 如果要管理 access key 的权限, [请看](#)

[8/10](#)

August 11, 2016

[8/10](#)

把 footer 放在 container-fluid 标签外面, 要不然会重叠

而且, 加上 style : margin-top : 100px 会有空格, 省的敲 br。

在 link to 里面, 如果不指明方法是 post, 他会自动来 get

这跟 simple form 还是有不一样的, simple form 会自动认为动作是 post

erd 会自动找出一个 model 的栏位, 但是会省略掉 user id 之类的信息

它是想告诉我, user id 跟别的栏位有不一样的地方么? user id 是别人跟我的联系, 并不是单纯来让我存储数据用的。

我可以建立两个 path 来分别管理 alipay 和 we chat, 也可以用一个 path 来传两个参数, 一个是 order, 一个是 payment method

这样能让我理解传参数。

@order = Order.find(params[:id]) 我好像更理解这句话了。controller 本来就是指挥官, routes 是地图, 地图上没标记的地方, 根本就没有路, 所以不能走。有路的地方需要指挥官告诉进来的人怎么走。而 views 就是跟用户的交互, 交互的过程中产生参数, 参数从 views 传到 controller, controller 看了之后判断一下, 决定让这个用户去哪里。那这句话是什么意思呢? :id 好像是一个键, 这个键存在于参数包里面, 其它的键也有, 但是我让指挥官先看这个 id 键, 通过这个 id 键, controller 能找到对应的 order, 好对现在的这个 order 进行操作。这里就是根据特征参数来找到对应的 model 来操作。如果没有特征参数, 谁也找不到对应的 model。我还可以试验一下, 瞎传参数行不行, log 里面应该也会打印, 并且不会报错。

而且, 我给梦琪讲明白了。

现在反思, 为什么能用 key 来找到 params 里面的东西呢? 因为 params 类似于一个 hash。

另外，过程中我还背闻到了@controller 是怎么来的，我一想，这是在 show 页，show 页的@order 已经被 controller 里面的 show 的 action 给定义了，所以这个@order 是确定的。

Xdite 对于@order 的理解是

controller 捞出 @變數,在 view 使用

但也可以这么想，show 这个页面之所以有 show 使用，就是因为进入这个页面的时候，controller 就已经把收到的参数转化为了@order，所以这整个页面都是知道@order 是谁的。之前不知道，因为参数是啥都不知道。但是有了 link to order_path，程式就会给我传参数到哪个 controller 的 show 方法，还会给打开哪个页面。

再想，hash 怎么找 value，就是 hash[:key]，所以 params 的形式也是 params[:payment_method]。一模一样。再想，

回头看 git push

git push heroku master, git push origin master

对比一下更明白，origin 和 heroic 都是 remote 端点。

做 RESTful 的 API

怎么设计，怎么接 api，手机和 web 用到。

reload!是在 reloads 环境

这让我碰巧看到了 api

File railties/lib/rails/console/app.rb, line 30

```
def reload!(print=true)
```

```
  puts "Reloading..." if print
```

```
  Rails.application.reloader.reload!
```

```
  true
```

```
end
```

就是看看别人是怎么定义 reload ! 这个 api 的，我输入之后，是怎么调用的。最后的 true 是返回值。

在别处有解释 reload 的效果，就是重新载入代码，但是不会让我新增 object。object 是什么？

调用找不到方法，还可能是 validates 上的调用有问题

今天在潘同学那里遇到一个 quantity 的 no_name_error，我想，可能是找错了调用者，调用者存在但不是我要的。但是这种计算机真的有这么傻，不直接报这个错么？以前好像遇到过。

报错信息更直接的指向 quantity 不存在，如果我能想到是 cart 无法调用 quantity 因为 cart 没有 quantity 属性（或者叫栏位），那我更可能去查 cart 在哪里调用了 quantity。

结果是在 validates 的时候调用了。跟平常用到的 cart.id 还不一样。

今天依然忘记加关闭标签，而且是在 mailer 里，不好检查

今天要从 mailer 里面发送邮件，报错，是 views 的错误，不好查。但是错误是比较低级的，以后要下定决心同时写开始和关闭标签啊。

类比 heroku run rake db:migrate

写 heroku run rails c 能打开 heroku 上的 console。或许还能用 heroku run 来跑更多 ruby 指令

aws 的预览密钥策略是什么鬼

能大致看懂 kms 有几个 action，貌似可以调用这几个 action

[8/8](#)

August 10, 2016

8/8

lily 帮我做功能，我学到了什么

lily 要做一个 js，所以用 script 标签来定义 js 的 function，然后在 form 里面引用这个 id。

当然最后用 on change 中的 this.form.submit()实现了提交，当然，要在 simple form 中使用 input 的 html，

要放在 `input_html: {}` 里面。

最后用 js 的 debugger 能暂定到 js 中的一个点。

另外，可以用浏览器检查 html 页面上的信息，并且浏览器上有 console，可以发送一些参数吧。

我还看到 xdite 建了一个全白页面，来检查参数传递。

@variable 可以在 view 中使用，不加@不能使用

@variable 是在 view 里面输出，在 controller 里面也是承载一个值。

不加@只能在 controller 里面自己用，view 不知道这个参数。

当然，controller 里面的变量更不能被 model 中，除非用 save，update 这样的方法写数据库。

find_by 要指明查询方式 find_by(id: params[:id])

后面那句话跟 `find(params[:id])` 效果相同。

但是，这个 bug 中更重要的内容是如果你的数据传错了，要不然就是 view 里面穿错了，要不然就是 controller 里面算错了存错了取错了，要不然就是 model 里面的存的问题。

不过，find 和 find by 的区别是什么？

belongs 和 has many

belongs 呢，就是能让 ci belongs to product 的时候，用 ci.product 就能调去唯一的 product。如果是 has many，也可以调取，但是只能调取很多个，比如 cart has many ci，所以 current cart 一调取，就是 current_cart.cart_items 是复数的，有很多个单项。都能来回来取的取，只要你关联他们。就能用谁作为主动对象（第一个）去调用后面的数据。

为什么刷新之后会被理解成 show 页面

因为方法改变了么？

是因为没有路径说明，所以会回到 show 吧。接着改一改，看看能不能补上，实在不行的话，把 show 关掉。

是 member 和 collection 的关系么？

删掉数据表单

rails console

```
ActiveRecord::Migration.drop_table(:table_name)
```

这样能删除整个表单

lily 带我读数据库

我在 console 里面打印出来数据库 c.products

```
Product Load (0.2ms) SELECT "products".* FROM "products" INNER JOIN "cart_items" ON "products"."id" = "cart_items"."product_id" WHERE "cart_items"."cart_id" = ? [{"cart_id", 1}]
```

最好能读懂这句话。

从 products 这里面查一些 *，这些 products 参加了 cart items，p 的 id 和 ci 的 p 的 id 是一样的，这里的 ci 的 cart 的 id 是 1

后来我在 count 的时候

```
SELECT COUNT(*) FROM "products" INNER JOIN "cart_items" ON "products"."id" = "cart_items"."product_id" WHERE "cart_items"."cart_id" = ? [{"cart_id", 1}]
```

看到了 count *，还是差不多的单词。

注释掉 erb

views 里面要想注释掉 erb 预计，需要在 % 前面加上 !，但是会出现大红色。为了好看，可以再加一层 html 的注释吧。

在 console 中可以试验很多功能

用 ruby 的语法，

find 只能查 id，find—by 可以查其它的 key

这是用 guide 查的，可以用 site:guides.rubyonrails.org，这种方法可以代替在 console 中试验。也可以查 dash，

但是可能会看不懂，没关系，看就好了。

另外，find(params[:id])说的是查传进来的参数 (params) 中的 id。直接查，不通过参数的话就直接 find (id : *)。

以前的 slack team 上的小技巧总结

August 9, 2016

小技巧

补 css

要補 CSS 各種坑的可以玩這裡的遊戲 <https://www.codeschool.com/learn/html-css>

数据库爆炸处理

<https://github.com/xingrowth/fullstack-course/wiki/%E6%92%B0%E5%AF%AB-seed.rb-%E6%AA%94%E8%87%AA%E5%8B%95%E7%94%A2%E7%94%9F%E8%B3%87%E6%96%99%E5%BA%AB%E6%95%B8%E6%93%9A>(

我寫了一個 seed.rb 可以在資料庫炸掉的時候自動產生數據，方便大家做練習

补 rails

<https://www.codeschool.com/courses/rails-for-zombies-redux>

补 bootstrap

<http://www.w3schools.com/bootstrap/default.asp>

html 切版

跟设计师交接 UI 设计需求时可能会用到

链接 simpleform 和 BS 之后的提示

用法 simple_form_for(@user, html: { class: 'form-horizontal' }) do |form|

如何判断一个 rails 程序员是否厉害

看他的文章，说话是否清楚，行动力是否强。

参加 hackathon 等于半年经验

投影片好第一，code 好

不错的产品，适合主办单位发新闻稿

在限定时间内做出能懂的产品

时间管理能力

1. 定义赢
2. 盘点资源：时间资源（扣掉满满的休息时间），人力资源，要做的事情（一个功能）
3. 找出最有风险的事情：早做最有风险的事情，然后做出能看到的東西然后再补后台（让要求的人安心），再然后做次要功能，最后改善流程找真的使用者来测。另外，例如工期三个月，最后一个月完整的不懂，用来上线（最有风险的事情）。把有风险的事情提前做。

提高自己的写作能力

<http://smalltalk.xdite.net/posts/773327-how-to-improve-your-writing-ability>

成甲的学习之道

https://www.evernote.com/l/ABE6OgYVfS1AC6M_Ydvj7Qm_6RCtZJtbTXQ

Xdite 关于新技能

<https://gist.github.com/xdite/568668a163af1a9b920390645ee62218>

<https://gist.github.com/xdite/89158f8b2bf0487645011e3a04d2d089>

练习 git 的网站

<https://try.github.io/>

学习别人的网站样式搭配

<http://bootsnipp.com/>

params 防弹衣

<http://blog.xdite.net/posts/2012/08/12/strong-parameter-mass-assignment-solution>

混乱是学习的常态

文章推荐

http://www.lallapure.com/2016/07/blog-post_26.html

[8/6](#)

August 8, 2016

8/6

挂 bootstrap 没挂好

我打开 localhost 看页面，navbar 的样式很乱，我才是 scss 没有改好。一看，发现了两个问题，一个是 application.scss 的名称拼写错误，一个是 bootstrap 后面没有留下分号。;

解决办法就是 rename，然后 enter

每建一个功能的话不如都新建 controller 开始

这样省的乱掉，而且报错会不知所云

我忘记了用 formfor 之后会需要加入 sendflash 才能出现提示信息。

用 simpleform 会更简单

如果用 form，我不太清楚为什么要写上 @group.errors.any? 是 group 在报错么？好想是的，是 model 返回了错误信息。

我应该记得用 devise 生成 user 的时候，devise 已经自己改了 routes

如果要摘到 user，就用 rails d devise user，但是不会摘掉 routes 里面的路径，需要自己手动摘掉

另外，对于 authenticate_user! 练的不熟

我在登出的时候忘记了给方法，方法是 delete

我只是告诉 log out 路径，但是打开路径只是读取，没有这个页面的，所以我直接规定 method 是 delete 就好了。不然会告诉我 Routing Error。

在首页显示 user 的 email 的时候并不需要 userid 这个栏位，不信我们就下回试试。

要显示 user 的 email，只需要 current user 就好，并不需要 userid。哪 userid 是干嘛用的呢？难道 userid 是已经写好的么？

应该是 rails 自己知道 userid 是什么东西，不然不会换一个 user 就产生和一个新的 userid，不过这个 user 对于 group 来说是唯一的，他应该会直接去找 user，然后正好有栏位叫做 userid，让他写进去。

我觉得 userid 和 groupie 是同一类的东西，我在开始做 post 之前检查了，全 atom 都没有 groupid 自然没有 groupie 的定义，我们来看看 rails 知不知道 groupie 是什么意思。

userid 是到后来的 post 时才用到的，这个时候要在查询时用到 userid。

我看到了，在我把 posts 放在 routes 中的 groups 下面的时候，rails 就已经在用 groupie 来集合 post 了。这说明要不然就是 rails 自己知道了自己要让 groups 用 id 来集合 posts，要不然就是 model 中的 harmony 起了作用。反正，这样 rails 让 groupid 自动有了用场。

当我在 console 中打印 user 的表单的时候，返现了 user 自身就是有 userid 的只不过没有中间的下划线。但他们还是有对应关系的对么。

如果在生成 post 时不写明 @group，那么 new 的时候，会不知道 groupid，找不到路径

错误信息时 nomethoderror，是因为没有单独的 postpath。而且顺序颠倒了也不行，这是为什么？

在编写 post # new 的时候产生了一个错误，是因为没有在 class 等属性的操作之前加，号

报错信息时 syntax 错误，就是语义错误。改语句啊。然后 class : 和后面的值之间如果不空格的话，也会产生语义错误，但实际上是我检验错了，不会报错。。

把 post 的 belongsto 写成了 users 和 groups，但实际上我只让他 belongs 一个 id

如果 belongsto 好几个 groups 的话，如果找 post 的路径就是 groups 的 id，但是我只有 group 的 id。而且，我定义这句话 @post.group 就会被说成是 no method，因为 @post 只有 groups，没有 group。

这跟我周五讨论的能不能不靠中间体就形成多对多点关系好像有联系，而这个例子倾向于不支持我之前的假设。

这有利于我理解他们的关系，尤其是理解了为什么能 @group.posts 这么调用。答案是因为我在 model 里面自己声明了。

我经常把 html 样式搞烂，不过试试就好了

我的 html 知识确实不多，认识几个标签，但是目前阶段这不是最重要的，再漂亮也实现不了我要的业务逻辑。我目前认为底层是更重要的，因为我可能想出不一样的底层，然后实现出来。

把程式语法包装成功能描述

這裡的 order("created_at DESC") 是「程式語法」，但不是「功能敘述」。我們作為旁觀者，只能猜測這是要按照「最近的時序排列」。

这是老师在教材中写的一句话，也是一个设计原则，方便自己和他人阅读代码。如果我自己设计代码的时候，可以不用回删，直接想好什么是程式语法就好了，直接封装，能这样么？

使用 scope 的时候少写了一个逗号

在 scope 要封装的名字后面，应该加个，号。

belongsto 貌似只能是单数

在引入 group relationships 这一块，所有的 belongs 后面跟的都是单数

[8/4](#)

August 5, 2016

8/4 问题

要想使用动态效果，都要在 js 里面挂上新的文件

用 flashes 的时候挂上 alert，用下来的时候挂上 dropdown，这两个 js 都是 bs 的样式，所以格式都是 require bootstrap/...

在 src 中赋值，用了 erb，需要用 = 号印出

今天我要让 src 引用一个 uploader 声称的地址，就用了 <% %>，但实际上要用

今天发现我很容易低血糖，怎么办

现在觉得我很容易累是因为我很容易低血糖，我要补充葡萄糖才行啊，方法可以是吃东西，也可以是喝蜂蜜，很浓的蜂蜜。但是吃零食可能不太行，因为据说这样不利于意志力。而且，喝完蜂蜜之后好像没有那么快就能补充血糖，我需要提前喝，才能让血糖保持在一定水平吧。或者，干脆输液好不好。另外，别忘了补充氯化钠。

[8/3](#)

August 4, 2016

8/3 Questions

Ulysses 贴出的格式有问题

如果在 U 的编辑器中直接复制 md 文本，贴到网页编辑器中的话，有一些代码事不对付的，这样就导致格式错乱，最好能快速导出一下，选择文本，然后选择 md 文本就好，再复制（不用保存）。

如果错误的建了 db，但是还没有 migrate 进去，可以把他摘下来

rails d migration migration_name，然后再 migrate，就不会把错误的 db 写进去了

用 devise 产生 views 的时候产生错误

实际上应该用 :views，而不是空格 views，如果用了空格 views，会产生新的 model，而不是 views，就好像 devise :install 一样，是让 devise 本身做一些动作对么。

devise 没有回应

我在 routes 中已经有 devise 这个方法了,是上次遗留下来的,每次有 rails 的动作时都会呼叫到这个 routes,但是找不到 devise 方法,因为我还没有安装 devise。解决办法是把 routes 中的 devise 那句删掉。[解答链接](#)这个错误有更多意义:

1. rails 在安装一些 gem, 数据的时候, 会查看原来的路径, 然后把路径已经指向的一些文件也改变, 就好像 devise 创建的登陆系统中自动应用了 simpleform 样式 (我打开 devise 的 views 之后, 看到了 simpleform)
2. 如果遇到报错, 把这个错误复制到 google 搜索一下找答案, 如果报错太长, 就简化一下, 说出最重要的是什么错了 ## 安装 bootstrap 的动作不完整 我记住的只有安装 gem, 但是后续还要挂 sass, 首先用 mv 改 assets/stylesheets 里的 css 为 scss, 然后往上 import。剩下的就是网上写 layouts 了。但是写 flashes 之前要加上一个 js, require bootstrap/alert

我尝试一下做好几步, 最后不知道哪里出了问题

一步一步来, 刚开始的问题都是小问题, 度过去, 在后面多挑战一下。而且注意程式跟我说的话, 应该看看, 有时候对我有用。尤其是新手状态, 并不知道程式说了什么, 不看的话很不尊重他。哈哈。

把天才密码和肌肉联系结合起来

今天一直在前面犯错误, 所以前面练了有 3 遍, 所以最后反而更熟悉了。不如分小块, 多练习几遍。

[8/1](#)

August 1, 2016

8 月 1 日 Rails 问题

赋值之后需要保存

我在 controller 里面给 topic 新增了一些内容, 就是把 votes 的 count 赋值给 vote 这个栏位, 但是就像 console 中, 如果没有保存, 好像就不会被记录, 需要记录才行

新增一个非原来 CRUD 的页面, 就要新增一个 CRUD

我今天要新增一个 about 页面, 但是原来的 crud 已经被占了, 所以我要新建一个 CRUD, 步骤是改 routes, 用 resources, 然后新建 controller, model, view, 把对应的方法放到新生成的文件夹下。然后在原来的某个页面 link 过去。我现在想, 也可以用 show 上面的一个 id 链接过去, 但是如果我哪天删掉 show, 就会又问题, 所以还是新的 crud 比较安全。而且, CRUD 要放在正确的目录下面。

controller 里面的 layout 不会用

要新建同名的 layout 到 layout 文件夹下面, 然后新建个是, 给出 yield 位置

没有定义要送出的 http verb

我以为定义了 publish 是 post, 点开之后就能执行 post 动作, 但是保存说找不到 get 动作。当然找不到 get, 因为我只说了 post。所以, 我要自己说明一下 method 是 post。我以为老师答案写错了, 但实际上应该说一下 post。如果不说的话, 默认为 get 对么?

[全栈工程师班的 peer education](#)

July 31, 2016

全栈工程师班的 peer education

一次, 在郑老师的 conversation 中, 我听到她要把这个班设计成 workshop 的形式, 用 peer education 来教学。我之前一直是比较单打独斗, 跟别人合作比较少, 有一些问题宁可不问, 也不会问同学, 因为我觉得这些问题可能不需要现在解决, 而且跟同学学习的效率不如跟老师或者 google 学习的效率高, 更进一步, 这是因为我觉得同学都是普通人, 甚至是比我还普通的人, 跟他们没什么可以学习的, 所以我很少跟同学讨论问题。高中的时候有这样做过, 但更多的是为了刷存在感, 而不是为了真正的学习一个知识。

但是, 我也不敢肯定跟 google 或者老师学习的效率就一定比 peer education 的效率高。不妨试试。既然来了这里, 有这样的 peer education 氛围, 而且, 老师也不会系统的讲授知识, 很多时候问同学其实是很方便的, 我应该试试, 既然以前没有试过, 这次没准能发现新大陆。

目前来看，我的问题是，我觉得自己是北理工的，不屑于问那些好几本大学毕业的同学问题。但是，我或许可以想想，最后亏的是我自己，如果不问的话，而且，最终，我们都将从这个班结业，我认为多学一点，对我来说比较重要。如果不问的话，我是不是在浪费我超越他们的机会？或者说，提升自己的机会。我认为这种机会更重要。

第一周周记

JULY 29, 2016

学习方法

这周比较重要的是学习方法，如果能把 xdite 的学习方法用好，应该会终生受益吧。至少 xdite 用她自己的这种学习方法把 ruby 学的很好啊。所以，这种方法至少是可行的。

那么这种学习方法是什么呢？

xdite 管这叫肌肉记忆，我觉得这是一种总分总的关系。就像写议论文，上来写给人家一个观点，让别人知道你在讨论什么，但是别人有可能不同意你的观点，或者不明白你为什么会有这样的观点，所以你要分开来解释，当你解释明白了，最后再总结成熟一下观点。别人最后就能记住你的观点。

我觉得郑老师教给我们的学习方法跟这有点像。学校里面的学习方法比较偏向于先分后总。就是把细部的知识先学好，再来创造有整体目的的事情。其实，至少我学校里的老师，在第一届课的时候也会跟我们聊一些这个专业整体的情况，但是可能没有郑老师这样逼着我们把整体先做出来，没有先做一个试试，所以就算听了老师讲的东西，也很有可能不了解这个专业在干嘛。

我想要多了解一种学习方法。因为知识的基础不是那么容易打造的吧，人一直是在不断进步的，什么时候也不完美，如果只有从学校里面学好了，才能出来做产品，那可能就出不来了。所以，还是要培养自学能力。我想，我在这里要更郑老师搭建一个学习编程的结构框架，学会怎么自己往里面填东西。两个月能学会所有知识么？我觉得不行，如果真行的话，这个领域也太浅显了。所以，我觉得学一个最小可行性的方法，然后自己到外面去添东西。把这项技能丰满起来。

最近在看 lean startup，里面也提到最小可行性产品，然后发布到外面去积累数据，判断产品方向，不要学很多没用的知识。

我觉得有道理。我在大学里都不知道我未来的职业方向是什么，不知道我对什么感兴趣，就学了很多环境工程的知识，然后现在基本上没用，浪费自己和学校的资源。

学习框架，然后再添细节。

肌肉记忆，先做出来，我要的是能用，能有一个开始的起点。

第一周学到了一些很棒的概念和工具

JULY 29, 2016

几个小工具

1. 忽然听到旁边的几个同学在讨论 markdown 工具，我之前在简书上写东西上时候接触过 markdown，新生大学里面也支持 markdown。logdown 上面的 markdown 不好用，光标经常错位。我现在用的 ulysses 还不错，听一个叫做阿婵的人推荐过。他在微信上写的文章还是有很多人看的，排版也很漂亮，所以我觉得在这方面它比我厉害，所以我也用了 ulysses，支持 markdown，而且能提示样式。之前在知乎上看到锤子科技的池建强推荐了 dayone，但是看效果图感觉太花哨了。ulysses 的界面能让我比较专心，它的光标能吸引一直盯着写作位置，我喜欢。它们自己觉得自己买点就是抓住注意力的界面，我觉得它们做到了。这是 markdown 工具。
2. 接下来看是中英文切换工具。中国人打代码或许经常遇到这种中文符号在程序里报错的问题吧，而且，写完中文就要按住 ctrl 和 space 来切换，容易打断思路吧。后来跟隔壁同学讨论的过程中发现了，用大写键打开，可以把中文输入变成英文输入，但是不会影响英文输入的大小写。也就是说，打开大写之后，输入的是英文小写。这样，一键切换中英文，比两个键方便一些。
3. 我还看到有同学用 mac 的程序查字典，但是如果用 chrome 自带的插件查字典会更好吧。我用的是

翰林，可以屏幕取词。##这一周把 CRUD 练习的还算熟练 能够不看教材写出 CRUD，但是因为不懂 CSS，所以写出来的页面很丑。在 CRUD 里面几个点，或许可以推广使用，在这里总结一下：

4. 循环的语法是`<% @aaas.each do |xxx| %>`，xxx 是一个参数，它会在循环结构中代表一个 aaa，这样 controller 里面给出的 `Aaa.all` 传到`@aaas`之后，就会一个一个的执行循环里面的内容。循环里面写 `xxx.title` 之类的，这样能取到相应的 aaa 的值，就是这样。最后别忘了关闭循环，用 `end`
5. `simple_form` 是一个表单的 helper，用起来比 `form`，因为一个 `input` 就能解决输入问题。语法是`<%= simple_form_for @aaa do |xxx| %>`这里应该只有一个 aaa，给 aaa 传表单用的，xxx 是这里的参数，这个参数的到 `input`，然后传给 aaa 的感觉。内容一般是`<%= xxx.input :title %>`，后面的冒号，说的应该是说这个位置的输入对应的 title 这个栏位。
6. 新增栏位也有一个固定的套路。显示要 rails g migration add_xxx_yyy 这个样子，然后在 migrate 之前要写进 `add column` 就能增加栏位，也可以在 `add`……这个名称之后直接写上 `title :string` 之类的数据结构（原来的 case 好像是来自 stack overflow）。做完这一步就建立了数据的存储位置，然后让 controller 知道我要允许这些数据进来，所以在 `params.require(:aaa).permit(:title)`里面的 `permit` 中加入新的栏位，这样 controller 就能收到数据。根据老师以前的介绍，利用 url 也能传参数，但正常用户不会这么做吧，所以我们要在 new 的时候给用户新建一个 `input` 的地方，怎么 `input` 可以看 `simpleform`。这样就新增了栏位，当然，有些栏位可能不用输入，直接抓去 `user.id` 之类的，但是也要在 db 里面有相应的栏位吧。比如之前遇到的 `isadmin`，就算不能输入，也要开个看不见的栏位，有必要的时候用 console 来设定这个栏位的值。

第一周遇到的大坑

JULY 29, 2016

几个小坑

1. 我没有想到 rails 这么智能。我曾经想要建立一个 model 叫做 company，这样里面可以有一些 jobs 和一些 users，这里的 company 是单数，如果用复数形式，在英语里面是 companies，但我觉得 rails 不会这么聪明，竟然还懂英文语法，所以直接用了 companys 在 rails 中表示复数，但是后来我看到 db 里面给出了 rails 自己设定的关于 company 的复数形式是 companies，我就知道不应该小看 rails 的智商
2. 在 erb 中，什么时候打印，什么时候不打印。也就是说`<%>`什么时候加 `=` 等于号，什么时候不用。我有一次用 `simpleform` 的时候，没有 `=`，然后就导致整个表达都打印不出来。后来我在循环 `each`，`do` 中加了 `=`，结果导致打印出了 sql 样式的那种数据，很不美观。所以，在表单中需要用 `=` 号打印前面的命令，而循环中不要打印。
3. 经常忘了在 era 中用 `end` 来关闭。包括 `simple form`，包括 `each`，`do`。

不会看报错，束手无策的抓瞎

这是一个比较大的不足吧。要想填坑，好想智能多掉一些坑，或者打好数据库基础。初级练习中，有一个按照投票数量排序的作业，我用了 `order("votes DESC")`，但是报错说没有 votes。其实当初我根本就不知道是报的什么错，是我旁边的同学告诉我，这是 sql 在报错，我反应了一会，才想起来 sql 是数据库。但报错信息就写在我面前，同学不告诉我的话我却读不懂。经过这这一次的经历之后，我会比较知道哪里是 在报错。就是网页上那行大红花下面的小黑自，会告诉我是具体哪里丢了东西。

不过，幸好我英文还不错。能读懂报错的英文。

而且，我不知道怎么确认错误，但是我旁边的同学就能告诉我打开 console 看看数据里有没有栏位。哇，这是解决问题的一个方法。如果缺了哪里就补哪里，不像我之前一直在乱试验。而且，乱试验也不敢大试验，因为变量太多的话，就很难控制变量，我还是不知道具体是哪里出了问题。所以，学会这种分析报错的方法很重要吧。

然后，接下来还是数据库，我不知道怎么往数据库里面加栏位，就直接打开了 db，然后写语句，没用的。应该用 rails g migrate add……

[CRUD 错误 2](#)

JULY 29, 2016

制作 admin 的 CRUD 时遇到的问题

新建 model 的时候自带数据库，但 controller 不带

不知道会有什么问题，想不起来标准答案是什么了。而且建立 model 的时候，rails 把我的 jobs 自动变成了 admin / job.rb。感觉他们认识数字呀。

我建立了 admin 的 routes，但是首页找不到 job

(img)

会是什么原因？

我建立了 controller，但是没有建立 view，所以报了 api 的错

(img)

这个错误以前见过，是因为没有建立相应的 view，找不到去路。

我建立了首页，但是找不到 Job.all

(img)

教材中的 case 给的是 `current_user.participated_groups`，那我这里的问题就是找不到什么？不知道为什么 `job.all` 找不到。改成 `current_user.jobs`，其中 `jobs` 的 `s` 是报错让我试试的。改了之后，错误不在 controller 了，但是在 `admin/jobs` 的 `index`。报错说没有 `end`，我应该想到是循环没有 `end` 吧。可能因为上一次使用 `each do` 这个循环的时候是抄的代码，所以不用考虑 `end` 的问题。

加上 `end` 之后，报错说找不到 `user_id`，所以，这里跟 `user` 有关的内容就是

又报了没有结尾，因为 `simple_form` 没有结尾

(img)

但我已经知道了是因为没有 `end`。我又想到，如此说来 `simple_form_for` 不应该有印出用的 `=` 等于号

错误的使用了 `simple_form`

`simple_form` 上次看到，应该是用来做填表的表单的，不是用来输出的，输出的用普通的 `each.do` 就行了。

create job

```
Job.create( :title => "", :description => "" )
```

在 link 的时候把 `new_job_path` 错写成了 `new_jobs_path`

可以从两方面理解这个事情，我只是新增一个 `job`，所以是单数，另外看 `rake routes` 的第一列，那么面是 `new job`

`simple_form_for` 单独使用时忘了 `end`

如题，不光是 `each` 这个循环需要 `end`，`simple_form`

`simple_form` 的 input 不能使用“XXX”来标记名称

会报错，删掉之后就能进去了

`simple_form_for` 前面要加 `=` 等号印出

不然结果就是这里的任何一项都不会印出

link 到 `delete` 的方法看来真的是 `job` 的 `path`，和 `show` 是一样的，而不是 `jobs` 的 `path`

而且，`path` 的 `id` 需要来自 `||` 里面，比如 `fuck`

`create` 方法打错了，`create` 应该是 `new` 一个 `job` 的 `params`

而我写成了

(img)

实际上，应该是 `@job = Job.new(job_params)`

`atom` 不用老存档，也能更新到 `server`

如题

练习了 `crud`

把循环印出会产生多余的字符，而且很怪，像是数据库里面的东西

each do 不用印出

CRUD 中的一次错误 (step 1)

JULY 28, 2016

- 不会往栏位里写内容

我想要创建一条记录，一位能用 rails c 里面的 j.title 来创建，然后 j.save 一下，毕竟，我的 j.is_admin 就是这么做的，但是这次不知道为什么没有试验成功，可能是因为当时已经有了记录，我只是加一个数据，而今天我的 j 整个都是空的，所以需要 create 吧，最后的解决办法是用 Job.create (title : "", description : "") 来解决的

- index 列表抓不到数据

你看抓的都是神马？

教材上给的 case 是用 group.description 来取到数据的。后来我试验了用 job.description 但还是不行，你看：再看一眼，才能看出来我是忘了用 erb 的语法，没有用 <%= %>。可能这就是眼瞎吧。而且，光有 erb 还不行，因为 erb 只是取出数据，表格的格式是在 html 标签里啊。所以，正确答案应该是 <%= job.description %>。就算你没有 html 标签，至少要用 = 号吧 job.description 打印出来吧。

- <%= @jobs.each do |job| %>摆放的位置会影响输出格式

当我把上面这句话放在 tr 里面的时候，以为没有问题，可能因为当时我只有一条记录，但是当我有了第二条记录的时候，第二条记录没有出现在第一条的下面，而跟他并列了，我想，原因应该是因为这新的一组，没有执行 tr，所以不知道自己是单独的一行。所以，要想让新的组知道自己是单独的一行，要让 job.each do 里面有 tr。

- 打不开除了首页之外的其他页面，比如 new

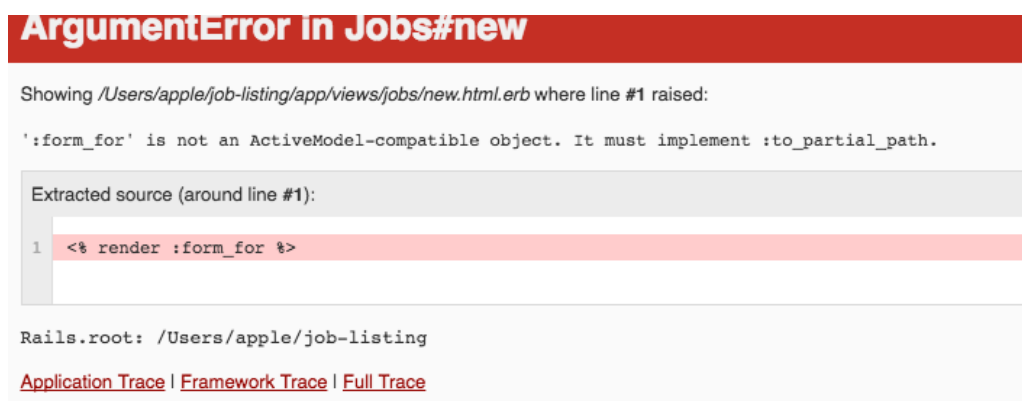
想要大开 new，需要告诉 routes，到 job 里面去找资源，所以需要加上 resources :jobs。而且，这个冒号：为什么要打在 jobs 前面呢，因为 jobs 是个固定的存储位置对么？之所以能打开首页，是因为我给了 root 'jobs#index'，但是其他的不行。除非我给出 # new 之类的吧。

- 在 new 里面不会建地方输入

我记得可以用 rails 的 f.input 来输入（全称应该是 <%= f.input ...%>这个样子，但是我不知道 f 是什么，也不知道 input 是什么，

不会用 _form.html.erb

我想要在 new.html.erb 中 render 一个 partial，但是不会用，不过这种简便方式可以有别的麻烦的方法绕过去，先记下来。



ArgumentError in Jobs#new

Showing /Users/apple/job-listing/app/views/jobs/new.html.erb where line #1 raised:

`:form_for` is not an ActiveSupport-compatible object. It must implement `:to_partial_path`.

Extracted source (around line #1):

```
1 <%= render :form_for %>
```

Rails.root: /Users/apple/job-listing

[Application Trace](#) | [Framework Trace](#) | [Full Trace](#)

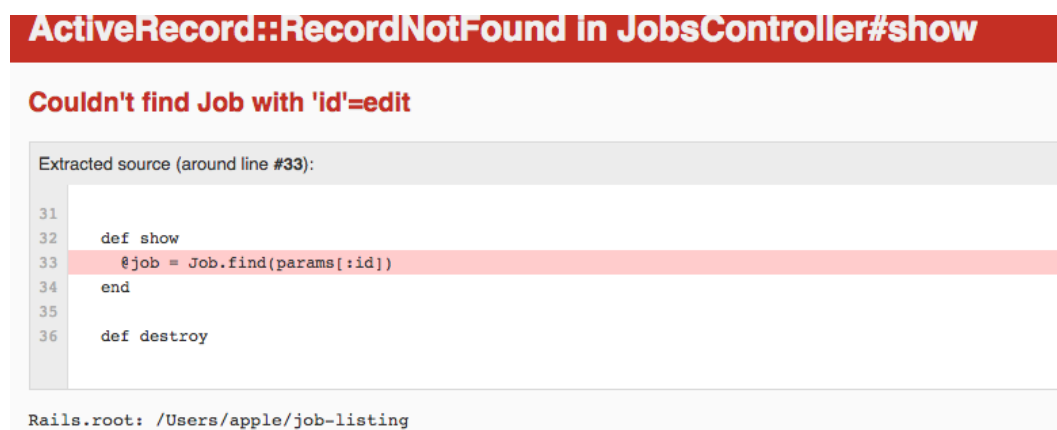
(img)

打不开 show 页，因为 show.html.erb 这个文件被命名成了 show.heml.erb

我一开始以为是我在这个页面写的内容的问题，但是现在知道这是我命名的问题

打不开 edit 页，然后报错事在 show 的 controller

edit 页跟 new 页的内容是一样的，new 就打的开。而且报错说 show 的 controller 有问题，问什么？



(img)

它报错很奇怪，所找不到 id 是 edit 的 job，我看来教材上给出的地址是 groups/1/edit，我没有给 params[:id]，所以活该进不去

我进入了 edit，但是没法 update

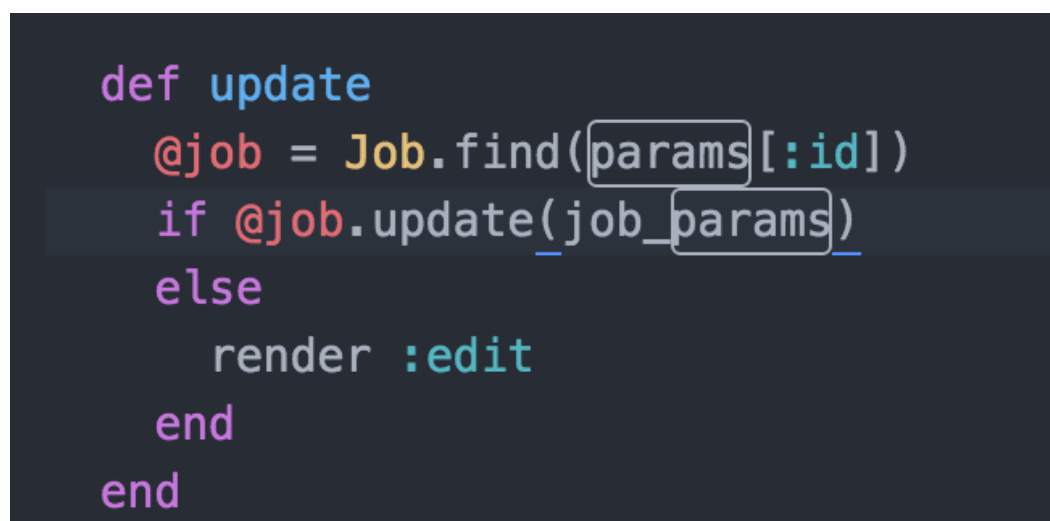
我进去了，但是 update 按钮点不下去，回到首页之后发现信息真的没有更新，是什么原因？

Title:

Description:

(img)

发现原因不是没有更新，而是只更新了一次，然后就定住了，没有跳回首页。如果第一次就改了的话，信息是会更新的



(img)

发现问题在于我没有在 controller 里面加 redirect_to 给 update。

不能通过网址进入 destroy

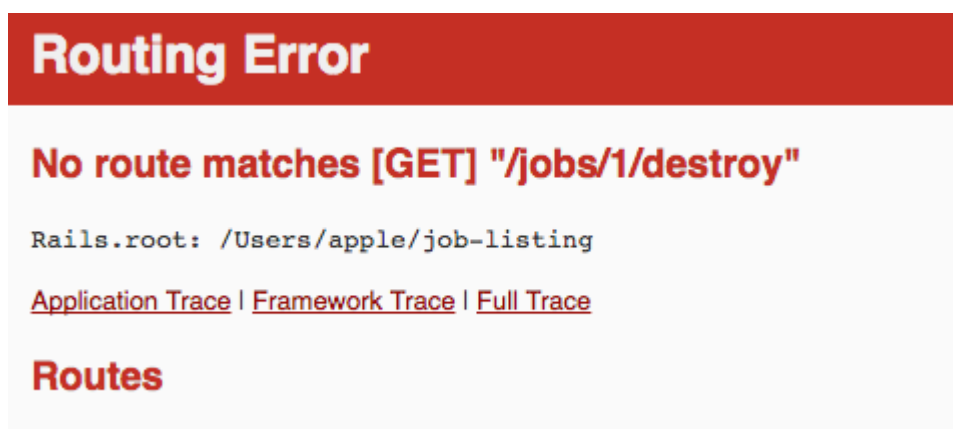
(img)

报错显示没有 routes，底下虽然有 jobs#destroy，但是这一栏的 title 叫做 controller#action，在 helper 这一

栏里 (path/url) 是没有跟 destroy 对应的内容的。而且, 这一页的 routes 真的就是 path/url, http verb 和 controller#action。

(img)

我想在首页上加个 button, 但是不知道语法



(img)

正确的语法是 class: "btn"这样子么? 配合 link_to, 我用了但是报错没有 routes

job_path	GET	/jobs/:id(.:format)	jobs#show
	PATCH	/jobs/:id(.:format)	jobs#update
	PUT	/jobs/:id(.:format)	jobs#update
	DELETE	/jobs/:id(.:format)	jobs#destroy

(img)

我现在已经知道 routes 怎么找了, 就是 rake routes 之后会在第一行有能写在 erb 中的 path, 如果是网址上的报错, 会更直接, 直接把 path 全称都给你打好了。我猜可能是因为我前面已经有 @jobs each do |job| 了, 所以这里不能用 link_to 么?

答案是因为前面已经有 each 了, 所以这里只需要提供 job 就好了, 这样他和 | job | 里面的参数就对上了, 然后就能取到参数了。再用 @job, 电脑不认识吧。

```
<%= link_to("edit", edit_job_path(job), class: "btn btn-primary")
```

那么问题来了, destroy 没有 path 怎么办?

答案是, 跳到 job_path(job), 也就是说不在 path 上动脑筋, 但是昨天 xdite 讲过了, method 送出 delete 方法。把这句答案背下来吧。

```
<%= link_to("delete", job_path(job), class: "btn btn-primary", method: :destroy, data: {confirm: "Are you sure?"}) %>
```

而且, 试验了一下, 把 method: :destroy 连在一起是不行的。

```
<%= link_to("delete", job_path(job), class: "btn", method: :destroy, data: { confirm: "are you sure?"}) %>
```

```
<%= link_to("delete", job_path(job), class: "btn", method: :destroy, data: { confirm: "are you sure?"}) %>
```

```
<%= link_to("delete", job_path(job), class: "btn", method: :destroy, data: { confirm: "are you sure"}) %>
```

(img)

但还是报错, 我看了一下, 真的, 是我把方法太多了, http 上的 verb 叫做 delete 对么, 而我打成了 destroy, 好吧。应该是这么解释。

<%= link_to("delete", job_path(job), class: "btn", method: :delete, data: { confirm: "are you sure?"}) %>

然后我用在 title 的链接上用对了。这里对的地方主要在于明白了上面的“”其实是要输出字符，所以我用了 job.title，用这种方法新增 job 的按钮也成功了。

我们如何确切知道宇宙大爆炸真的发生过

JULY 27, 2016

一，我们何而来。

这是人类一直在探索的古老问题。不同的人从哲学、诗歌、数学和物理等各个方面来回答这个问题。

大多数科学家认为我们周围的一切都开始于 140 亿年前，一个叫做“宇宙大爆炸”的时刻。但是，我们要如何才能找到证据来证明那么久以前的事情？

从运动中的星系到古老的宇宙尘埃，我们今天能发现很多宇宙大爆炸的遗迹，这些遗迹都清晰的讲述了宇宙的起源。

二，我们能看见宇宙膨胀

当我们望向夜晚的天空，我们可以看到我们星系范围内的星星。但这其中也有些模糊的斑点。这些星系与我们自己的星系相似，但是它们与星星相比距离我们非常非常远。

几乎所有星系都在远离我们，其中一些以数十万公里每秒的速度远离我们。

如果大多数的星系在远离我们，这就意味着宇宙在膨胀。如果宇宙在膨胀，那么过去的宇宙则一定要比现在小的多。追溯到遥远的过去，有一个瞬间当宇宙所有的物质都被挤成一个点，接着向外扩展。这个瞬间就是宇宙大爆炸。

三，比起星系的本来颜色，为什么他们看起来偏红

<http://a.files.bbc.co.uk/bam/live/content/zcpk6sg/large>

图片内容：

上图标题：就像警车驶向远方时，警笛的音调听起来变低了

左：拉长的声波音调变低 右：缩短的声波音调变高

下图标题：所以等星系远离我们时，它们看起来变得更红

左：拉长的光波颜色变红，缩短的光波颜色更蓝

我们能发现一个星系是否正在远离我们，方法跟我们听到警车飞驰而过时的感觉是一样的。如果警车开着警笛驶向远方，警笛的音调听起来比实际上要低，因为声波给拉长了。光也是由波构成的，所以快速移动的星系也会产生相同的效应。如果星系正在向远方运动，它的光波就会被拉长。这就使得他们看起来变红了。它们远离的速度越快，看起来就越红。

四，发掘宇宙大爆炸的遗迹

我们虽然不能用肉眼看到大爆炸的遗迹，但是望远镜能。肉眼只能看到宇宙光线中的一小部分，除了可见光之外，还有很多其他不可见光，比如 X 射线，红外线，紫外线，无线电波，和微波。它们比可见光的波长更短或者更长。

在宇宙大爆炸之后，宇宙中充满光线，但是随着宇宙的膨胀，这些光线逐渐被拉伸成微波。

一个微波望远镜能够看到来自宇宙初生时期的古老光线。而且，微波望远镜能够全天观测天空，并且发现天空时时刻刻充满着一种辐射，叫做宇宙微波背景辐射。

五，星光和微波有什么区别？

<http://a.files.bbc.co.uk/bam/live/content/z8yjn39/large>

图片内容：

左上：从特定光源发出的光都会随着距离越近而越亮。

右上：宇宙微波背景辐射从任何方向观测都可以看到同样的颜色和亮度。

不同于星星发出的光，宇宙微波背景辐射从任何位置观察都一样，无论你在空间中的认为位置。

六，让我们反观时间的长河

望向太空就如同回顾时间的长河。这是因为光射到距离远的物体比距离近的物体，到达我们这里需要更多

时间。如果一个物体距离我们是一百万光年，我们正看到的则是一百万年前的它。
现代天文望远镜如此强大，以至于它们可以看到距离我们数十亿光年的对象。这个时间接近宇宙大爆炸。
如果宇宙大爆炸真的发生过，我们就有能力去解释一些观点 – 还没有变成恒星和星系的星云。
最近，天文学家发现了这些像是在遥远宇宙中的星云。其中一些大约在一百二十亿或一百三十亿年前。尽管这是个不可思议的距离，我们通过一种被称为光谱分析技术 – 分析光穿过它们从而识别它们。
正如宇宙大爆炸理论的预测，这些古老的星云的组成物质与现在宇宙迥然不同。大多数化学成分是现代宇宙恒星的内部成分。因为星云在星星形成之前，它们几乎由所有的最基础的元素组成 – 氢和氦。

七，那又是什么引发了宇宙大爆炸呢？

这个问题依然困扰着科学家。下面理论有可能是答案么？

黑洞的爆炸（物理学家 Nikodem Poplawski 的理论）

每个黑洞都会吸收物质，也许会有一个相反的白洞在另外一个宇宙中释放物质，形成宇宙大爆炸。

宇宙大爆炸是宇宙的重生（宇宙大循环理论）

宇宙也许会降低它的膨胀速度甚至由于引力而向内塌缩，如此一来会形成一个碰撞，而大碰撞又引发大爆炸。

宇宙大爆炸是来自多元宇宙的气泡爆炸（混沌膨胀理论）

在无边的多元宇宙中，我们的宇宙也许诞生自一个能量膨胀的气泡，气泡爆炸导致宇宙大爆炸并且产生过多气泡。

没有原因（霍金的理论）

宇宙大爆炸就是这样来了，没有为什么。宇宙大爆炸是时间的开端，时间开端之前什么也没有，就像北极点以北并不存在。

[7/27 遇到的错误](#)

JULY 27, 2016

今天发现，只要丢掉教材的拐杖，就能报除很多错误，尤其是 html.erb 里面不会写的太多了。我把错误记在这里吧，给自己的错误笔记增加的素材，也会在看教材之后把正确答案贴出来。主要是不看正确答案，又一些 bug 没法过……

each do 不会用

Showing /Users/apple/job-listing/app/views/jobs/index.html.erb where line #8 raised:

undefined method `model_name' for #<Enumerator: []:each>

Extracted source (around line #8):

```
6   </tr>
7   <tr>
8     <% form_for @jobs.each do |job| %>
9       <td>:title</td>
10      <td>:description</td>
11      <td>:created_at</td>
```

Rails.root: /Users/apple/job-listing

[Application Trace](#) | [Framework Trace](#) | [Full Trace](#)

app/views/jobs/index.html.erb:8:in `app_views_jobs_index_html_erb_439712855214135588_70361891395840'

问题出在不应该有 form _ for，所以问题很有可能是我不知道 form _ for 是啥。先不管，这里的问题是我用 <% @jobs.each do |job| %>, and then, it will be ok. And the transformation between English and Chinese disturb me very much. I decide to write my note in English.

What's more, <% @jobs.each do |job| %> should be added before table row, since every job info will create a row, right?

forget the close tag of

TitleDescriptionTime

我本周的任务就是把这个网站写熟练，但是目前看来，我队 view 里面的东西还停留在只会复制的阶段。我应该全程不看教材，一遍一遍过一过试试。本周剩下的时间就应该都是练习和答疑时间了吧。练吧。这里的问题是没有关闭 talbe，然后 html 就把我的内容放在 footer 下面了。不知道为什么会跑到 footer 下面。我也没有指定 footer 和 navbar 印出的位置，看到问题之后想起来了，尝试写一下印出位置吧。

[各种括号的用法](#)

JULY 27, 2016

() 小括号可以把逻辑表达式和数学表达式括起来，也可以括起来一个 arguments 供函数来调取。比如 $a = 2 * (3 + 4)$

 $b = (x==y) \parallel (m==n)$

Hash.new.send('[]'=:a, :b)

大括号 {} 用于把 Hash 里面的数值或区块包起来，例如， $h = \{1=>2, 2=>3\}$

h.each {|k,v| puts k+v}

中括号用于数列的索引和分隔，也用于从 Hash 里面抓东西。例如：

arr = [1, 2, 3]

two = arr[1]

three = h[2]

[回头一看，竟然发现自己抄都抄错了](#)

JULY 26, 2016

在根据教材练习时，练习题都是有解答的，为了不让自己依赖复制粘贴，我有时候会自己手打。这让我发现了一个问题，就是我手打的时候很难打下完整的一个语句，我是边看教程，遍把这句话打上去的：一个单词一个单词的打，甚至是一个字母一个字母的打。我期望达到的状态是，能看完一整段话，比如 def 一个 method，理解了这段话的意思之后，把语句打上去。而不是简单的敲上去，做一个人型拷贝机。我会这样想，是因为小时候家长告诉我，基于理解的抄猜会更有意义。我觉得有道理，我以前在学校的时候抄课文，只想闭着眼睛抄，根本不想动脑子。或者抄单词的时候，手已经习惯了，却没有在脑子中跟单词的意思形成连接。我不想让我打语句也出现这样的情况。

当然，不是什么语句都是现在能理解的，现在尽量理解，不能理解的就背下来吧。

最后，听着隔壁 xdite 狂打键盘，如果他是在用在这速度写代码，我要疯掉了。我也想着么酷。

[7/26 全栈工程师课程日记](#)

JULY 26, 2016

Objective

今天上午郑老师讲了 Terminal 上的一些基本指令，namely，cd 更改目录，ls 引出目录下的文件，pwd 查看当前问题，rails s (c, g) 被分别是大开 server，console 和创建一些内容。据说新手常常忘记自己在 console 里，然后直接打 Terminal 中用的指令，我就是个例子。irb 和 console 长的差不多，能运行一些 ruby 语句，而且貌似在任何地方都能大开 irb，我在 console 中键入 irb，都打开了。

之后是 git 指令, git 能帮我存档, 目前用的还不是很熟的指令是 `git remote add origin git://github.com/xxx.git`, `git push origin master`, `git push -u origin master`, 因为不知道 origin 和 master 是什么意思。现在知道了, origin 代表 github 上的远端存储点, master 是本地主分支, 这几个语句都是在建立远端和本地的练习。而 `git pull` 从来没用过, 希望快点有机会用到。最后是 heroku, 用来部署, 最熟悉的是 `git push heroku master`。其他的内容包括操作习惯 (3 个窗口), 重开 server 和不用重开的情况有哪些 (改 app 和 db / migrate, 不改环境的时候可以不开, 其他重开), 今天我好像忘了重开, 所以 `admin / jobs` 进不去。最后是 `ruby-toolbox.com` 和 `railscasts.com` 上有很好的教程, 可以试试看。最后看了 html 一些标签的含义, 和 css 的内容。

最后, 开始做招聘网站这个 project, 按照之前做讨论版的教程改了改。

Step 1 :

- 实现 Job 的 CRUD (JobsController)
- 先只开 Job 的 title, description
- JobsController 的 new / create / edit / update / destroy

Step 2 :

- 实现 Job 的 Admin 的 CRUD, 在 `admin/jobs`
- 在 `admin / jobs` 皆需要先过 `before_action :require_is_admin` 才能进入, 否则返回首页
- 定义一个条件 `admin?` 判断是否是 admin

遇到的坑有 :

1. 因为教材上是讨论版, 所以名称叫做 group, 而我要做的是 job, 打错了几次, 生成了 `groupcontroller`, 而不是 `jobcontroller`, 最后导致 `admin / jobs` 找不到 controller, 最后改了 controller 头部的 class, 写成 `admin :: jobscontroller` 就好了, 以前很少改 class 的。
2. ruby 中有单数和复数之分这我知道, 照着教材看也能知道 group 和 groups 应用的地方有点不一样, 但是不知道 ruby 竟然精通以 y 结尾的单词复数形式是 ies, 这让我万万没想到。我建立了一个 model 叫做 company, 以为 ruby 没那么聪明, 所以在需要复数的时候就直接写了 `companys`, 但是报错, 虽然报的不是这个错, 但是我怀疑有影响。恰巧我进了 rails 自己生成的 migration 里面看到了代码中 rails 竟然自己写出了 `companies`, 我就知道 rails 比我以为的知道的更多。
3. 最后, 在做 `require_is_admin(job)` 的时候, (job) 如何变成负数, 我试了 jobs, 但是报错, 所以试了 `@jobs`, 成功了。不太理解 @ 的含义, 但是我猜是指现有的 jobs 吧, 希望以后得到验证。

Reflective

工作持续性有点差, 后来想到了番茄工作法, 情况好一点了。

我想, 以后写日记还是主要写自己新学会的东西, 不熟悉的东西, 已经熟悉的东西就没必要写了, 节省时间。

Interpretive

Decisive

明天是答疑时间, 时间会比较自由, 早点利用番茄工作法。把注意力集中起来。

July 25, 2016

以下试着按照 ORID 来整理思路

Objective

就像我上过的大多数课程一样, 我在全站工程师班首先接触到到内容是很 general 的一些概念——computational thinking。郑老师将这种思维解释为三点: 让计算机来处理问题的思维, 把大问题拆解为小问题的思维, 在实践中找到资源的能力。做到这三点或许能帮助我更好的分析问题, 解决问题。

这种 computational thinking 在程序开发中的具体应用体现在把一个点子转化为软体企划，再把企划一步步实现，运营。为了小试牛刀，课上首先对上班开会的过程进行了 ct 式的策划，然后又对招聘网站的功能进行了设计，把功能的重要性分为 must have, should have, could have, nice to have。然后可以按照重要性来安排开发顺序，这样就算有了企划。

郑老师把招聘网站点功能总结分为四部分，张贴系统，观看系统，应征系统，管理系统，其中观看系统和应征系统是给应征者用的，张贴系统和管理系统是给管理员用的（招聘方和管理员合并为管理员）。为了确定具体开发哪些功能，我们可以应用 user story 点方法，便于敏捷开发，避免需求来回变更（小型项目，写 30 条左右用户故事，大型项目，写 100 条左右用户故事），用户故事的写法可以参照模版：身为 xx 角色，可能用到 xx 系统，以达到 xxx 的目的。

接下来的内容会更细节一些：

1. 如何在 git 上 fork：点击 fork
2. 为什么要 pull request：方便团队合作修改。但是我还有没有团队合作过，所以不理解有多方便
3. 如何 pull request：git push 后，compare 两个 repo，然后 create pull request

我实现了把 xingrowth 的专案 fork 到自己的 git 下面，然后 clone 到本地来修改，修改的内容包括安装 bootstrap, devise, simple_form，之后 git push，然后 pull request

Reflective

今天感觉最紧张的时候是刚到教室到时候，因为作业没有写完，怕跟不上，但是听到有其他同学也没有写完，心里好受一点，觉得没那么孤单，当然，以后还是要抓紧写完作业，点击 submit 的那一刻还是很爽的。今天最高兴的时候就是把 git push origin version-1 master 上传的时候，当时我正在 job-listing 的页面，看到上传结果自动跳了出来，没想到 git 的页面反应这么快，这代表了我写完了作业，也代表了我理解了 git 的一个功能：pull request 之后在 git push，不用再次 pull request。

Interpretive

我觉得今天学到的比较重要的内容是对课程有了了解吧，方便自己安排以后的学习计划。computational thinking 应该是一个很重要的概念，因为这个概念很 general，应该能用到很多方面，而且郑老师分配了很多的时间来讲解这个事情，相对于其他的小知识，能感觉到她对这个概念的看中。而且，这个概念也让我想到新生大学里面的必读书目 principle 中也有关于如何一步步实现 dream 的方法论，其中也涉及到分析问题（更确切的说，应该是分析真正的问题），然后非常 logic 的设计解决办法。

我感觉现在的自己还不能很好的 handle 这种思维，因为我在面对问题的第一个感觉是慌张，经常跳过分析的过程，直接抓一个方法来做到看似解决了问题。

Decisional

今天的工作还算轻松，但我想对未来在项目上可能遇到的挑战提前做好准备。准备的方法是对试验，加一点语句，看看是什么效果，删一些步骤，看看是什么效果，如果不试的话，感觉永远有疑惑。问别人也可以，但是自己试验或许更直接吧。

In addition

今天李笑来的讲话也值得记录一下。

李笑来介绍了一种学习方法：研究一个学科的历史，弄明白这个学科的目的是什么，在解决的核心问题是什么。他认为这是很有效的一种学习方法，这样我想到看看有关编程的历史。我读过图灵传，对编程的理解是解决重复的计算步骤，因为图灵发明图灵机的时候在研究破解密码的方法，而解决的方法就是用计算机来试验所有可能的密码，而且作者还提到，图灵一直在思考如何机械的解决任何问题（图灵的答案是没有这种方法），所以，计算机的发明应该是为了机械的解决问题吧。我需要多看一些有关计算机的历史来验证一下。BTW，我应该会读英文书。

李笑来对于全栈工程师的介绍是，手头用工具能够做出完整的产品，尽管有些工具并不熟练，但是，有些工具能用就行了，用的好不好并不是最重要的。作为全栈工程师，要看到什么是最重要的，积累到最重要的东西，就能开始行动了。而编码这种细节性的工作的重要性可能并不高（尽管在刚开始编码看起来很难）。

他讲的人生三大坑也很有意思：莫名其妙的凑热闹，心急火燎的随大溜，为别人操碎了心肝。第一个坑太浪费注意力资源了，而这种资源如果用在有用的地方，自己会过上更好的生活吧。心急火燎的随大溜说的是什么火就跟什么，不考虑自己有没有积累，没有积累就去跟大溜竞争，可以么？操别人的心可能是一种对时间的浪费，李笑来还从中看出了害怕别人比自己过的好的恶意，我觉得有道理。大家都忙自己的理想，各自安好吧。

最后，教能促进学，他提醒我们在课程期间最好乐于分享。

最后一遍练习中遇到的问题

July 25, 2016

第三遍练习可能是我最后一遍练习 rails101 了，因为今天已经开始上课了，接下来会有更有挑战性的专案让我来练习吧。

在课前我并没有写完 101 的作业，有点紧张，怕被老师说，所以在到教室之前，在教室外面的快餐店坐了一会，把第三遍练习的后半部分花了 1 个多小时做完了，但是没有上传到 heroku，因为我在 create 的时候，被报告说 heroku 满了，当时已经 9:30 了，试了一下 delete 指令，但是在 heroku 上不能执行，所以不知道怎么删，到了教室之后问了同学，才想到去 heroku 网站上登陆后台去删，果然可以。而为什么我的 heroku 满了呢，而有的人就没有满，因为我曾经建了一个空的 heroku app 但是什么都没上传。可能是因为之前手抖吧。

在上传后一切大功告成，但是打开 app 遇到的却是 something went wrong，核对了一下教材，返现是没有在 heroku 上 rake db : migrate。

在练习 rails 的时候遇到了一次好运

July 24, 2016

今天在一家咖啡馆练习 rails 时，被一个女生搭讪了，感觉还不错。她说我在写程式的时候很认真，但我知道我只是因为前面的作业没做完，在 ddl 面前，必须集中精力写程式。这或许就叫狗屎运吧。

但我提醒自己，如果不认真对待接下来 rails 的学习，我踩我很有可能踩狗屎，而不是走狗屎运。就像李笑来分析“惊喜” (serendipity) 的那篇文章，只有增加实力才能避免坏运气，甚至实力足够强，连坏运气都能变成好事情。

有一个犯了 2 遍错误的地方

July 24, 2016

今天在做讨论版的 posts 的时候，犯了一个似曾相识的错误，程式对 each 报错，说我没有定义 each。在我第一遍练习做 posts 的时候，好像也遇到过类似的问题。找到问题的方法还是顺着出错的地方往前找。这样，我找到了问题：我在 groups controller 的 show 方法中，没有加入 @posts = @group.posts，所以程式说找不到 each 是因为我根本没在 show 方法中定义有 post 这么一个东西对么？

NoMethodError in Groups#show

Showing /Users/apple/rails101_2/app/views/groups/show.html.erb where line #21 raised:

undefined method `each' for nil:NilClass

Extracted source (around line #21):

```
19     </thead>
20     <tbody>
21       <% @posts.each do |post| %>
22         <tr>
23           <td><%= post.content %></td>
24           <td><%= post.user.email %>      </td>
```

[一个意外的错误：把 resources: posts 放错了位置](#)

July 24, 2016

今天在做 rails 讨论版练习的第二遍，进度意境落后了，先是遇到一些小问题，比如在 checkout 用户权限的试用用了 if 却没有加 end，原因可能是我得意<%if...这样的格式并不熟悉，总是把 erb 看成是 html 标签，有了%这样的符号却不知道是什么意思，我不知道课前还有没有查查 rails 的书，如果没有就在课上带着问题去听讲吧。感觉自己并没有很好的完成自己答应郑老师的两件事，作业还是拖延了。但无论如何，今天还是要完成作业，把练习都做完。

然后，现在 groups 内有 posts 的修改时，程式对我做的 groups # show 里的 write a post 按钮报错，我检查了这句代码，跟教材没有出入，于是就按照教材的顺序往前查，我发现我错误的把 resources: posts 和 resources: groups 并列了，而不是放在了后者的里面（也就是手没加 do ）。然后再回头想为什么程式要对 groups # show 报错了，可能是因为里面有一句@group，但是 posts 不在 groups 底下，所以取不到 group 的 id 对么？

希望这个问题能在下面的练习中得到验证或纠正。

[同一个错误犯了两遍](#)

July 21, 2016

今天开始第二遍制作讨论群的专案，在 very beginning 的地方就遇到的一个问题。Hello World！欢迎页无法访问，报错显示是在 application.scss 里，就是下面这个部分。其实我在输入代码的时候就感觉不对，因为同样是@import，颜色却不一样，但没有找到产生这种结果的原因，就继续往下走了，等到报错，再来根据教程检查代码，发现少打了分号——“；”。

这个问题我在第一次做 rails101 的时候就犯了，但是当时想着这么小的错误，不知道写一篇文章，因为就放过了。另外，其他事情下改 rails 很少遇到需要加分号来标注语句结尾的情况，没有其他机会提醒自己不要犯这个错误。可能基于这两个原因，这个问题没有记住。

现在，我应该是记住了在改 scss 的时候要用分号标注语句结尾，但是为什么，还是不清楚，希望第三遍做之前能找到答案。

[第一次接触 Rails](#)

July 21, 2016

从开始课前搭建 Rails 开发环境到现在已经有 10 天了，这 10 天遇到了一些困难，现在回头感觉很多困难很小，但是，如果我不是交了钱来上课，不解决这些困难我就对不起自己的钱的话，我可能已经放弃了。既然真的要学习 Rails 了，我还是摆正态度，摆脱“傲慢”。

今天遇到的一个问题是，在做用户后台的时候，第一遍搭建好了，试验成功，但是一会又不行了，期间我并

没有改代码。我以为造成问题的原因是我误碰了键盘，所以想办法从第五章开始重新做了一下（因为问题出在 post 功能上，所以我把有关 post 的内容重新写了一遍），但还是报错。于是我把报错的 title 给删掉了，这样可以打开 account / posts 了。进去一开，有一些 post 没有 group 我这才想起来，我在设计删除 post 功能的时候，试用了一下删除 group 的功能。而现在想明白了，我删除了 group，group 下面的 post 却没有一起删掉，所以有一些 post 找不到 group.title。浴室我在 console 里面 Post.destroy_all 。

NoMethodError in Account::Posts#index

Showing /Users/apple/rails101/app/views/account/posts/index.html.erb where line #19 raised:

undefined method `title' for nil:NilClass

Extracted source (around line #19):

```
17     <tr>
18       <td> <%= post.content %> </td>
19       <td> <%= post.group.title %> </td>
20       <td> <%= post.updated_at %> </td>
21
22     </tr>
```

这个问题的深层次原因是什么呢，是我还不够理解 post 和 group 的关系，所以没想到删掉 group 竟然会残存 post，也没想到没有 group.title，account/posts 就不能进。这些应该都是写在代码里的，代码是我按照教程写的，但是并不真正理解这些代码。应该说，按照 Xdite 的方法——“如果不理解就先记下来”，我已经比之前多明白了很多代码的含义，但 views 里面很多 html 的标记，看书可能更快一点。另外.rb 里面很多语法感觉不到像 C 语言那样明确的规则，可能也需要看 Ruby 语法书来学习。

目前的目标是把作业练熟，然后争取在课前有时间看 html 的书。

[Newer](#) → [Blog Archives](#)

Recent Posts

- [怎样更能保护注意力，在没钱的情况下--一个小故事](#)
- [jQuery 实现动画效果](#)
- [jQuery 中更多 on 事件](#)
- [jQuery 互动时如何防止浏览器乱跳](#)
- [用 jQuery 做点互动效果](#)

Copyright © 2013 - Sining_Liu - Powered by [Logdown](#) and [Octopress](#)