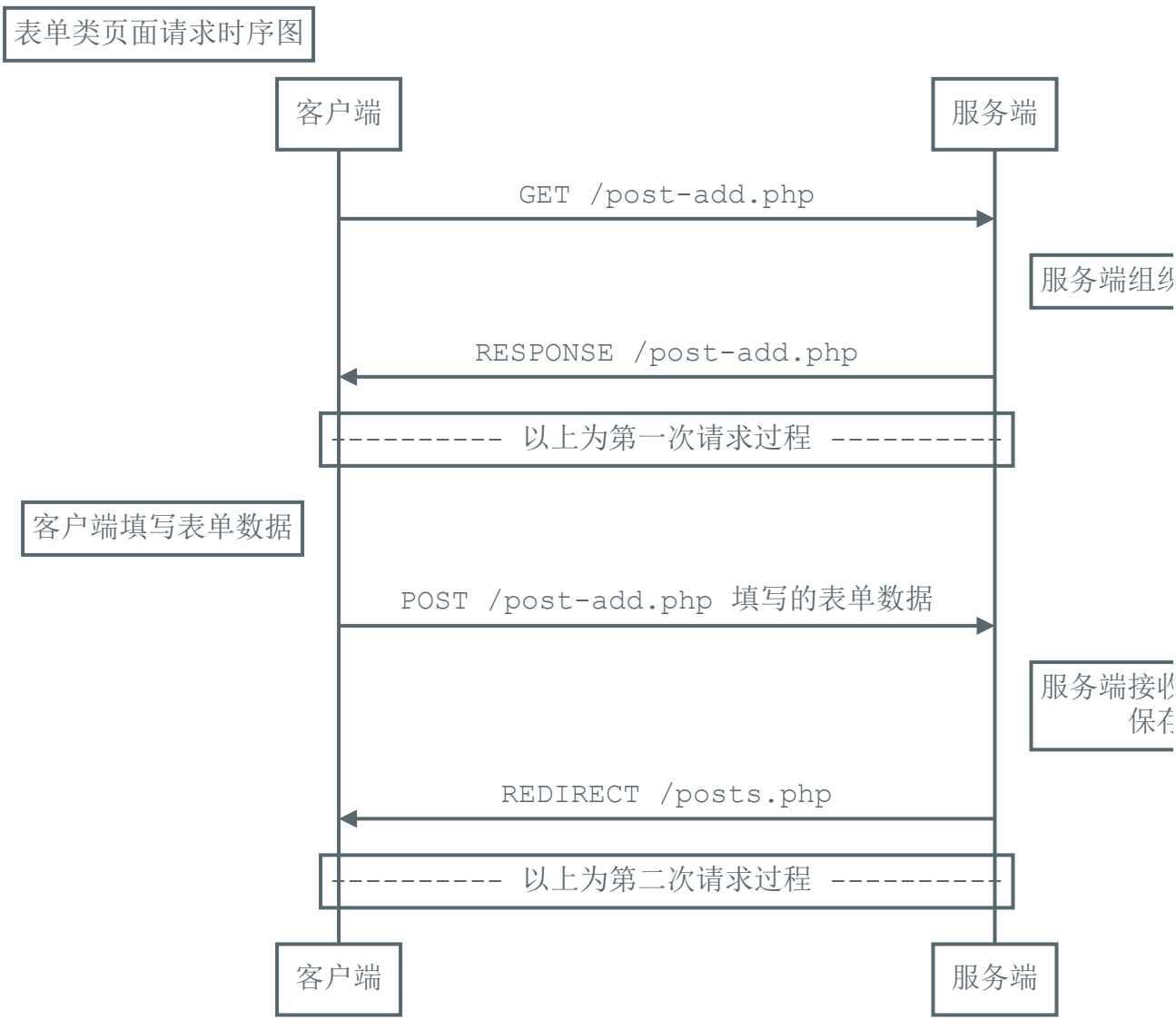
写文章

1. 相关介绍

在动态网站开发过程中，最常见的两种页面情况：

- 单纯的对数据进行展示；
- 表单类页面的两次请求；

以下为一个非常经典的表单类请求时序图，可以清晰看到两次请求过程：



MAKE IT BETTER

2. 页面调整

类似于之前的操作，我们还是需要调整页面上细节：

- 表单属性
- 表单元素属性
- 等等

2.1. 为 `<form>` 添加必要属性

1. `action="/admin/post-add.php"`：提交给页面自身。也可以在单独创建一个 php 文件，让 action 提交到这个单独的文件。
2. `method="post"`：1. 提交的数据篇幅比较大；2. 主观意义上也是给服务端发数据。
3. `enctype=""`：暂时不提，待会会用到，再说。

2.2. 为 `<input>` 添加必要属性

1. `name` 属性，表单元素提交必要属性
2. `label` 关联，每一个表单元素都必须有一个关联的 `label`，不管界面上需要还是不需要。（现在的 Web 开发者很不讲究了，很少有人注意这个问题了）

2.3. 提交按钮 type

表单想要被提交必须要有提交按钮的点击，所以页面上必须有提交按钮

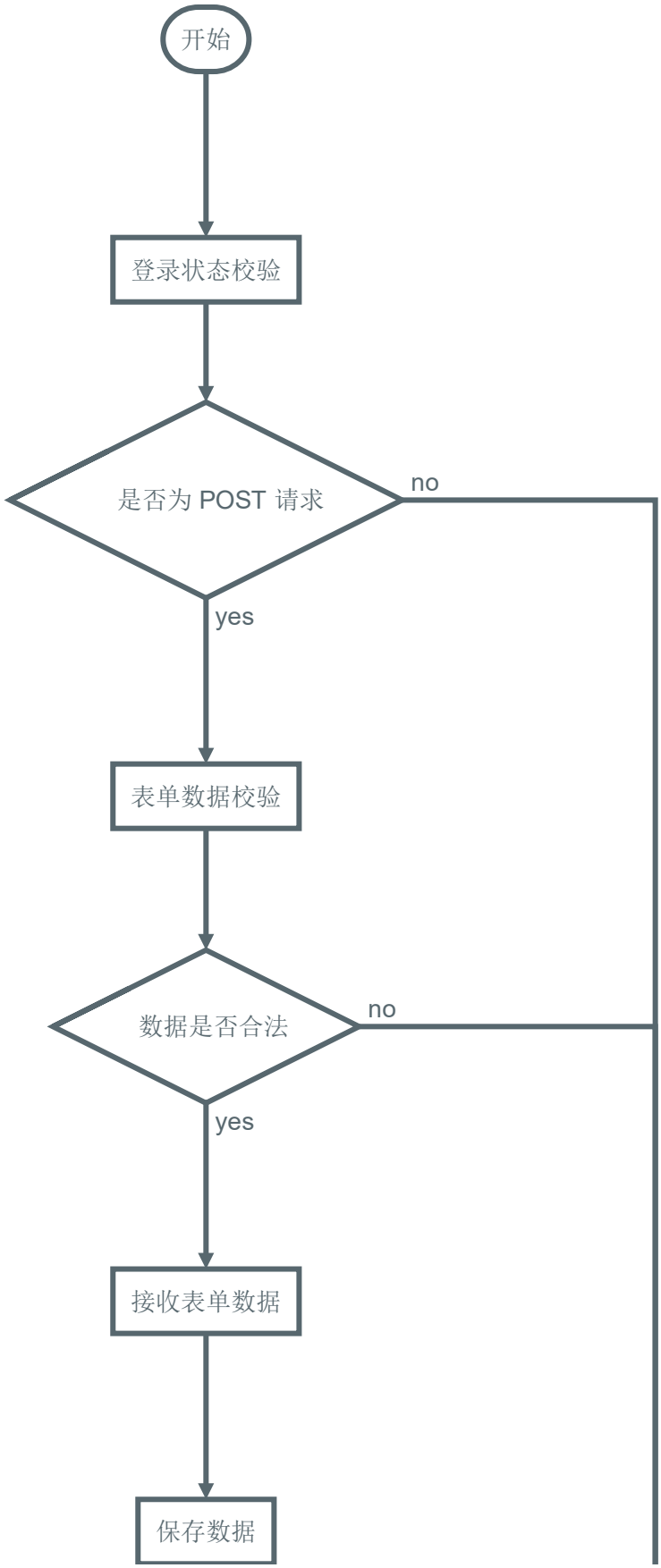
当然，我这里提供的 HTML 绝大多数问题都已经考虑了，所以不需要太多调整。

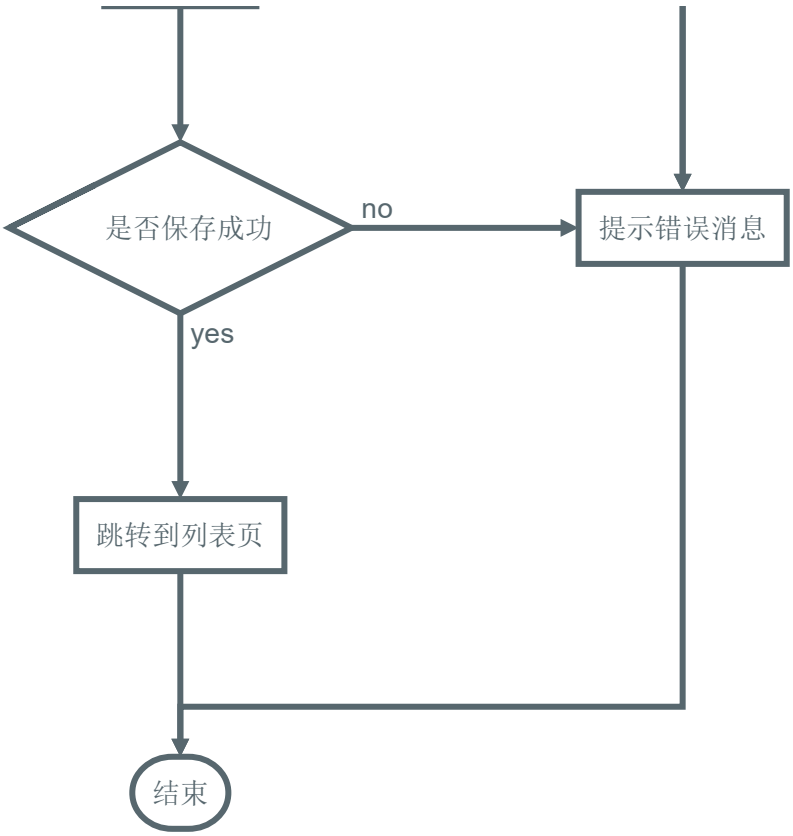
> 源代码: step-38

MAKE IT BETTER

3. 业务核心

在这个页面中，业务的核心自然是：当表单被提交（POST）回来，接收表单上填写的数据，校验，然后保存起来，具体流程如下：





3.1. 数据校验

```php
// 处理提交请求
// ========================================

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
  // 数据校验
  // ----------------------------
  if (empty($_POST['slug'])
    || empty($_POST['title'])
    || empty($_POST['created'])
    || empty($_POST['content'])
    || empty($_POST['status'])
    || empty($_POST['category'])) {
    // 缺少必要数据
    $message = '请完整填写所有内容';
  } else if (xiu_query(sprintf("select count(1) from posts where slug = '%s'", $_POST['slug']))
[0][0] > 0) {
    // slug 重复
    $message = '别名已经存在，请修改别名';
  } else {
    // 数据合法
  }
}
```

3.2. 接收数据

作者 ID 可以从当前登录用户信息中获取

```php
// 接收数据
// ----------------------------

$slug = $_POST['slug'];
$title = $_POST['title'];
$feature = ''; // 图片稍后再考虑
$created = $_POST['created'];
$content = $_POST['content'];
$status = $_POST['status']; // 作者 ID 可以从当前登录用户信息中获取
$user_id = $current_user['id'];
$category_id = $_POST['category'];
```

3.3. 保存数据

```php
// 拼接查询语句
$sql = sprintf(
  "insert into posts values (null, '%s', '%s', '%s', '%s', '%s', 0, 0, '%s', %d, %d)",
  $slug,
  $title,
  $feature,
  $created,
  $content,
  $status,
  $user_id,
  $category_id
);

// 执行 SQL 保存数据
if (xiu_execute($sql) > 0) {
  // 保存成功
} else {
  // 保存失败，请重试
}
```

3.4. 错误消息展示

```php
<?php if (isset($message)) : ?>
<div class="alert alert-danger">
  <strong>错误! </strong><?php echo $message; ?>
</div>
<?php endif; ?>
```

3.5. 跳转

```php
// 保存成功 跳转
header('Location: /admin/posts.php');
exit;
```

> 源代码: step-39

3.6. 加载分类列表

分类数据目前是在 HTML 中写死的，十分不合适，应该是在一开始的时候加载数据库中的分类，然后展示在界面上。

再由用户选择。

```
// 查询数据
// ========================================

// 查询全部分类数据
$categories = xiu_query('select * from categories');

...

<select id="category" class="form-control" name="category">
  <?php foreach ($categories as $item) { ?>
  <option value="<?php echo $item['id']; ?>"><?php echo $item['name']; ?></option>
  <?php } ?>
</select>
```

> 源代码: step-40

3.7. 表单元素状态保持

目前，如果提交表单时发生任何错误，当浏览器再次显示表单时，之前的数据全部会被清空，为了提高使用过程的体验，所以需要保持之前的填写状态，具体方法参考之前登录功能：

在服务端渲染 HTML 时，给每一个表单元素加上 value 属性，值就从表单提交过来的数据中取。

例如文章标题：

```
<input id="title" class="form-control input-lg" name="title" type="text" value="<?php echo isset($_POST['title']) ? $_POST['title'] : '' ?>" placeholder="文章标题">
```

> 源代码: step-41

MAKE IT BETTER

4. 图片上传功能

如果希望表单可以上传文件，必须将表单的 `enctype` 属性设置为 `multipart/form-data` ，

我们不可能在数据库中保存文件，一般情况下，我们都是将文件保存到一个目录中，在数据库中保存访问路径 URL：

4.1. 修改表单的编码类型

> http://www.w3school.com.cn/tags/att_form_enctype.asp

默认情况下表单的编码类型为：`application/x-www-form-urlencoded` ，表示当前表单中的数据是以 `urlencoded` 方式提交的。

但是对于表单中有文件域的情况下，文件是无法用文本表述的，所以必须通过另外一种编码类型：`multipart/form-data`

```
<form class="row" action="/admin/post-add.php" method="post" enctype="multipart/form-data">
  ...
</form>
```

4.2. 修改文件域文件限制

默认文件域允许选择任何文件，可以通过 `accept` 属性限制：

```
<input id="feature" class="form-control" name="feature" type="file" accept="image/*">
```

> http://www.w3school.com.cn/tags/att_input_accept.asp image/* 指的是任意类型图片

4.3. 接收上传文件内容

在表单提交到服务端时，会自动把文件上传到服务器上，PHP（内部）执行时会自动将文件存到一个临时目录，然后将相关信息定义到 `$_FILES` 数组中。

知道了这些，我们接下来要做的就是：

1. 判断是否上传了文件
2. 将上传的文件从临时目录移动到我们希望的目录
3. 将路径保存到数据库中

> http://www.w3school.com.cn/php/php_file_upload.asp

```php
// 接收文件并保存
// ------------------------------

// 如果选择了文件 $_FILES['feature']['error'] => 0
if (empty($_FILES['feature']['error'])) {
  // PHP 在会自动接收客户端上传的文件到一个临时的目录
  $temp_file = $_FILES['feature']['tmp_name'];
  // 我们只需要把文件保存到我们指定上传目录
  $target_file = '../static/uploads/' . $_FILES['feature']['name'];
  if (move_uploaded_file($temp_file, $target_file)) {
    $image_file = '/static/uploads/' . $_FILES['feature']['name'];
    var_dump($image_file);
  }
}
```

这里我们暂时注释掉保存数据的代码，防止保存太多没必要的数据，另外如果保存成功了，页面会跳转，我们看不到调试信息：

```php
// // 执行 SQL 保存数据
// if (xiu_execute($sql) > 0) {
//   // 保存成功 跳转
//   header('Location: /admin/posts.php');
//   exit;
// } else {
//   // 保存失败
//   $message = '保存失败，请重试';
// }
```

经过测试发现，保存文件不会成功，错误信息：

```
Warning:
move_uploaded_file(../static/uploads/photo.jpg): failed to open stream: Permission denied in
/.../baixiu/admin/post-add.php on line 45

Warning:
move_uploaded_file(): Unable to move '/private/var/tmp/phpkIg9CC' to
'../static/uploads/photo.jpg' in /.../baixiu/admin/post-add.php on line 45
```

通过阅读错误信息发现：应该是权限一类的错误。

> 在每一个操作系统中都有文件权限问题：
>
> 1. 系统中每一个工作进程都是由一个用户运行的；
> 2. 如果这个用户没有指定文件或者目录的访问权，那么这个进程就无法操作这个文件或者目录；

解决方法很简单，修改一下上传目录的读写权限即可，重点要理解为什么！

U系操作系统：

```
$ sudo chmod -R 777 /.../baixiu/static/uploads
```

Windows：

右键菜单找吧 ;)

> 源代码: step-42

4.4. 保存文件路径到数据库

```
// 上述逻辑执行成功就会定义一个 $image_file 变量
$feature = isset($image_file) ? $image_file : '';
```

> 源代码: step-43

MAKE IT BETTER

5. 页面交互

上述过程已经完成了，当前**发表文章**功能的核心业务，但是很多地方的体验不佳，例如：

1. 上传图片没有预览
2. 文章内容用 `<textarea>` 做纯文本编辑并不合理

接下来我们可以利用我们之前的所学之长，解决这些问题。

> 声明：谁要是还在说前端只是写页面，我 TM 弄死谁！前端也是开发者，开发者的核心是**实现功能**服务用户。

5.1. 本地图片预览

本地图片预览的意思就是当选择图片后在界面上显示当前选中图片，显示图片肯定就是用 `<img>` 标签，`<img>` 标签需要工作就必须有一个图片 URL，所以核心就是怎么给选中图片一个 URL。

正常我们在页面上用的 URL 都是类似 `http://www.demo.com/a.jpg` 这样的地址，也就是说是服务器上的一个图片，但是我们目前的情况，图片在选中过后，还没有上传到服务端，所以不可能有这种 URL 地址。

选择图片这个过程中，我们都是在浏览器本地（客户端）单机操作的，没有跟服务交互，所以这个需求应该是用 JavaScript 处理。

> 解决这一类的问题，核心把握：我现在在哪？我手头上有什么（能拿到什么）？我想要什么？

> **我现在在哪？** 客户端浏览器，用户正在操作。 **我手头上有什么（能拿到什么）？** 与这个业务有关的东西，我们只能拿到用户操作的文件域 DOM 对象 -> 选中的文件对象。 **我想要什么？** 我想要这个文件域中选择文件的 URL。

那么接下来就是解决：文件对象 -> URL

在了解了 HTML5 Web API 过后，我们知道 HTML5 提供了两种办法：

1. Object URL
2. FileReader

后续会详细学习到这些，我们这里通过 Object URL 解决目前这个问题，在页面中加入一段 JavaScript 代码：

```
// 当文件域文件选择发生改变过后，本地预览选择的图片
$('#feature').on('change', function () {
  var file = $(this).prop('files')[0]
  // 为这个文件对象创建一个 Object URL
  var url = URL.createObjectURL(file)
  // url => blob:http://zce.me/65a03a19-3e3a-446a-9956-e91cb2b76e1f
  // 不用奇怪 BLOB: binary large object block
  // 将图片元素显示到界面上（预览）
  $(this).siblings('.thumbnail').attr('src', url).fadeIn()
})
```

> 源代码: step-44

> 作业： 尝试使用一些 jQuery 插件，提高图片上传过程的体验

5.2. 文章 URL 预览

没什么好说的，直接看结果

```
// slug 预览
$('#slug').on('input', function () {
  $(this).next().children().text($(this).val())
})
```

> slug 是一个专有名词，用于表示 URL 中辨别标识，就是 ID 的作用，只是比 ID 更友好，更美观 例如：
>
> - https://zce.me/category/1
> - https://zce.me/category/travel

> 源代码: step-45

5.3. 富文本编辑器（Markdown）

1. 传统的富文本编辑器（HTML）
    - TinyMCE
    - CKEditor
    - UEditor
2. Markdown 编辑器
    - SimpleMDE

具体使用方式，参考官方示例即可。

注意：autoDownloadFontAwesome 选项

> 源代码: step-46

> 作业： 自学 Markdown，思考 Markdown 的优势。 说明: 必须学，一个开发者的基本素质，一手格式良好的 Markdown，不解释

5.4. 初始化时间

由于 `<input type="datetime-local">` 标签的 `value` 属性要求是一个 `yyyy-MM-ddThh:mm` 格式的字符串，原生 JavaScript 格式化时间特别麻烦，我们可以通过社区提供的一个开源库 moment.js 解决。

```
// 发布时间初始值
$('#created').val(moment().format('YYYY-MM-DDTHH:mm'))
```

> 源代码: step-47

> 作业： 将表单中发布时间元素改为第三方提供的时间选择器插件。 参考: https://eonasdan.github.io/bootstrap-datetimepicker/