PART 05

# 编写规范

# 文件

- 文件名全部小写，以短横杠“—”隔开，例如：my—dear—baby.html
- 文件名应简明、扼要、一目了然，尽量让人一看就能知道这个文件是干嘛的
- 尽量按照类别进行归类

- 以<!DOCTYPE html>首行顶格开始
- 申明文档的编码<meta charset="utf-8"/>
- 根据页面内容和需求填写适当的keywords和description
- Title必须写
- 样式效果尽可能应用css解决
- 不使用行内样式
- 不在元素上使用 style 属性
- 做好注释
- Id唯一
- Class和id单词字母小写，多个单词组成时，用“-”隔开
- 属性的定义，统一使用双引号

- 注释很重要

  备注：顶部文档注释应包含制作人、制作时间、内容描述、涉及模块、修改人和修改时间、备注（或特殊说明）
- 不要使用@import。与<link>相比， @import要慢很多，不光增加额外的请求，还会导致不可预料的问题
- 代码组件化
- @charset：utf-8；
- 多个样式，每条样式独占一行
- 每个规则声明间用空行分隔
- url()、属性选择符、属性值使用双引号。例如： url("//www.google.com/css/maia.css");

  font-family："open sans", arial, sans-serif; .selector[type="text"]{ }
- 将媒体查询放在尽可能相关规则的附近。

- 注释！！！

  1、每个文件最上面写明文件的说明

  2、每个函数必须加注释，标注函数的功能与返回、传入参数的说明

  3、常量、变量标明对应的描述

- 命名

  1、常量：全部大写，多个单词直接下划线“_”连接，eg.

```
var MAX_COUNT = 10;
```

  2、变量：小驼峰命名法(camel case)，eg.

```
var maxCount = 10;
```

  3、函数命名：小驼峰命名法(camel case)，

  eg.

```
function getName() {
    return this.name;
}
```

  4、构造函数：大驼峰命名法(pascal case)，eg.

```
function Student(name) {
    this.name = name;
}
var st = new Student('tom');
```

- 命名

5、类的成员：

（1）公共属性和方法：小驼峰命名法

（2）私有属性和方法：小驼峰命名法，前面加下划线“_”

eg.

```
function Student(name) {
    var _name = name; // 私有成员
    // 公共方法
    this.getName = function () {
        return _name;
    }
    // 公共方式
    this.setName = function (value) {
        _name = value;
    }
}
var st = new Student('tom');
st.setName('jerry');
console.log(st.getName()); // => jerry: 输出_name私有变量的值
```

- 书面整洁、明了
- 在匿名自执行函数中使用严格模式，

eg.

```
(function(){
        'use strict';

        $(function() {
                //  to do ...
          });
}());
```

PART 06

# 联调过程

检查接口

接口的地址是否正确

检查参数

该传的参数传没传，参数的拼接是否正确

检查传参类型

Json，string或是其他格式，header头部的定义

检查报错信息

针对报错信息查找问题