大家都在用vue，芒果用的是angularjs，所以我这边就不分享技术了。来讲讲平时工作中的一些小技巧吧。

分享提要：

1.小技巧

2.vscode

踩坑：

团队开发，老项目，在没有规范的情况下，经历过许多程序员之手后，该何去何从？

新项目有编码规范，但没有统一的开发习惯又该如何？

在此列举一些情况（希望大家积极补充）：

一、不断新增的css样式

有时候为了实现模块化解耦，我们尽量保持功能的代码独立性，会独立在功能模块添加新的样式。

对同一个人来说，也很难保证样式不重用，不冗余。有时候我们还会为样式取什么名称而烦恼。

1、尽量使用框架样式

能使用框架样式的，尽量使用框架样式。

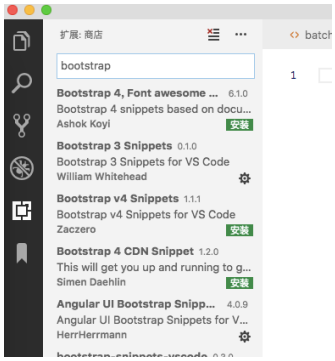
一般情况下选择了某个框架后，我们就可以像搭积木一样开发页面，很少需要定义新的样式。

如果在开发中还在经常写样式，那最好要停下脚步，好好分析一下原因了。

在使用框架样式时，可以参考官方demo

http://www.jq22.com/demo/ace-master20161123/

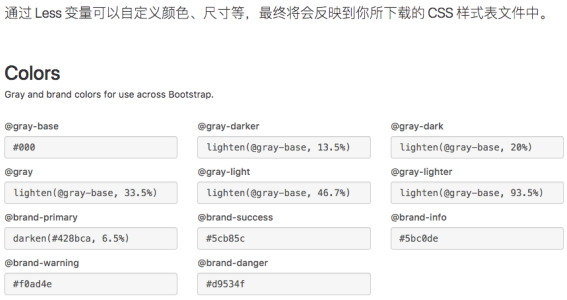
或者在编辑器安装补全插件，比如bootstrap



2、对框架样式重写

对框架样式重写的目的不能只为了满足某个局部的功能点，考虑后续全局使用，像bootstrap在官网可以定制默认样式

https://v3.bootcss.com/customize/



3、对框架样式扩展

也就是我们定义的全局样式。我们可以考虑把样式进行拆解。

比如：常用的颜色，边框，边距，填充距，宽，度等。

```
// 这样的样式似乎不太合适写在全局样式中，如果越来越多的类似样式，我们无法记忆，更别提让新员工快速上手
.panel1 {padding: 5px 10px 10px 5px;}

// 拆解样式如下，有规律可行，自由组合，易于记忆。Q2:有人会提出疑问，css少了，html会不会更多了?
.mar-t-5 {padding-top:5px!important;}
.pad-t-10 {pading-top:10px!important;}

```

4、美工遵循UI框架及历史页面

前端应该第一时间把我们的UI框架告诉美工。

在开发类似功能的页面的时候，美工应参考历史页面，不宜重新设计新的风格的页面。

当前端需要写很多样式来实现效果图的时候，这里可以停下来好好思考一下。这个功能是不是一定要这样设计。

A2:这里我们就能回答Q2的问题了，在设计效果图遵循规范的情况下，我们很少会使用UI框架或全局样式以外的样式。

二、越来越多的代码风格

当团队中存在着不同的编码风格，又没有定义好编码标准时，经常出现的一种情况就是一个人会重写其他人的代码，觉得自己的代码风格是这会扰乱团队成员之间的关系，降低大家对工作的满意度，并且将大量宝贵的时间浪费在没什么生产力的事情上面。

因此，建议大家不要这么做，如果怀疑或是发现有人这么做时要及时提出来。

1、不要编辑现有代码

不要因为现有代码有多“烂”，就去重写。只会产生更多的风格。也许有一天，当看到自己当时改进的代码，又觉得需要改进。

2、一改全改

如果非要去修改前面的代码的话，和团队沟通后，统一改掉。

3、最新风格形成代码片段

在实现类似功能的时候，我们真正要做的通常只是改改里面的参数。

A同事自己手写了一份，非常满足（用时1小时）。

B同事七零八落地在现有的代码中找了出来，修改了参数，快速完成，进入下一个任务（用时20分钟）。

C同事类似B同事，但在找代码的时候对老代码进行了类比，选择比较好的那一份，完成后，对功能代码进行拆解，写在代码片段。（用时4

但在后期开发中，同样的功能，A和B可能只能在代码熟悉度提高效率，

而C同事则会成倍提高效率。因为拆解后的代码片段不仅仅只用于类似功能，而是可以自由组合，用于很多场景。

4、前后端统一

在前后端配合的过程中，后台先出接口，前端开发后联调。

当类似功能后台没有按以前的写法写的时候，前端就没法复用或者要花点时间转化数据格式上。

这里我们应该遵循第1条（不要编辑现有代码）原则，建议后台也参考之前的代码，不要重新写。

这个深有体会：

http://www.mangoerp.com:8080/erp/#/erp/stock/wanyitong

芒果后台最近开发了一个新的海外仓，后台负责人，参考原有的wish海外仓，进行开发，接口格式相同的情况下，前端开发这个页面，复制只用了5分钟。开发完成后放到测试机上，后台自己去联调。

但如果有些功能原来就有的，后台开发的时候改变了接口格式，那就没这么容易了。沟通成本增加了，出错的可能性也提高了。

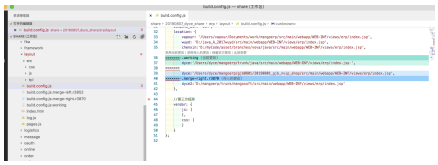
三、vscode相关

好用的功能：

代码片段

代码片段实战演示

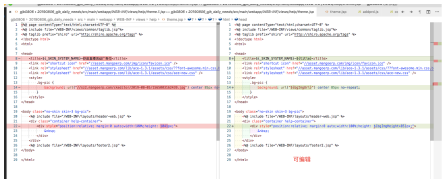
svn冲突解决



比较变更

切换到并排视图

文件对比



jsdoc注释功能

```
/**
 * @param a number
 * @param b
 */
function func(a, b) {

}
```

工作区配置与切换

```
{
    "folders": [
        {
            "path": "."
        },
        {
            "path": "./../server"
        }
    ],
     "settings": {}
}
```

快捷键：

常用：vscode 快捷键.md

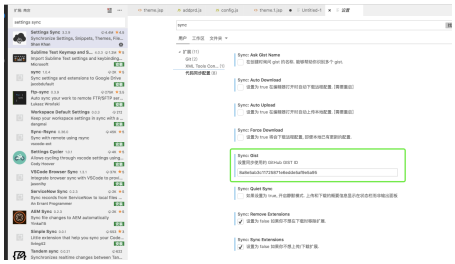
网络：http://elickzhao.github.io/2017/03/VSCode%20%E5%B8%B8%E7%94%A8%E5%BF%AB%E6%8D%B7%E9%94%AE/

好用的插件：

网络：https://zhuanlan.zhihu.com/p/40417719

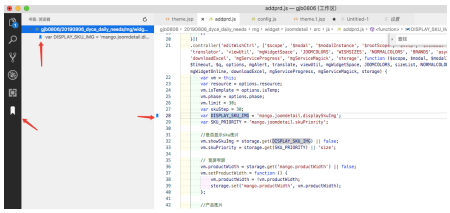
网络：https://juejin.im/post/5b123ace6fb9a01e6f560a4b

配置同步 Settings Sync



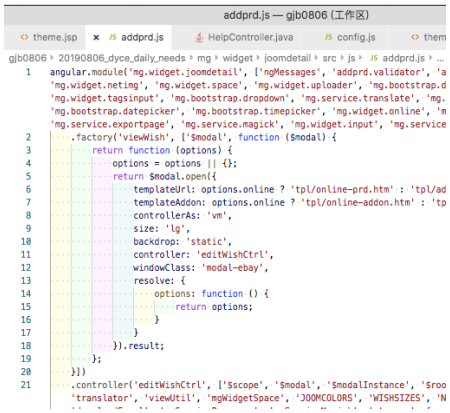
https://gist.github.com/dyce0523/d5ddc26d0f1807eaa9bef075e3aedc6c

bookmark

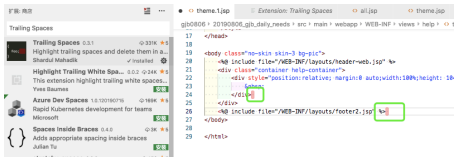


cmd+alt+k 添加或删除书签

Indent-Rainbow

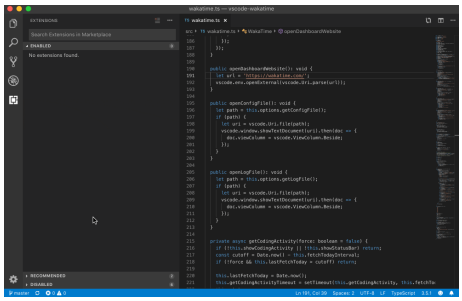


Trailing Spaces



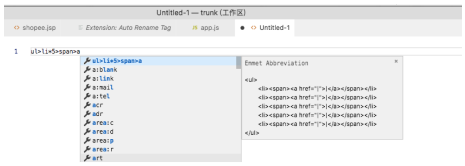
Atuo Rename Tag

WakaTime



Emmet

网络：https://www.cnblogs.com/summit7ca/p/6944215.html



auto close tag

Vetur

SVG Viewer

Auto Import

Todo Tree

感触：

熟悉框架中常用组件使用

能复用就复用，减少新风格

一个趁手好用的开发工具

常用功能拆解成代码片段

美工、前端、后端强制统一

众说纷纭.....