# 自身技能

1. 正则表达式
2. Cookie
3. 表单新旧值对比  
   <input type="text" data-old="bbbb" value="aaaa">

遍历比较data-old 与 value 返回flag

1. 面向对象编程（类，对象）
2. 数组

# 技巧

1. 将模板放入<script type=’text/template’></script>便于管理（不用字符串拼接）
2. 事件绑定方法放入init 函数统一调用，具体实现放到对应方法
3. 对于大量不同操作类型操作同一数据，可将数据放入统一父级元素方便数据获取（多个td 操作同一数据，可以将数据放置tr）
4. 用id取节点提高效率
5. For循环时，将变量放置var i 后统一定义，防止for循环多次定义变量，增加内存消耗

for (var i = 0, len = data.length; i < len; i++) {

}

1. 将文案放置对应操作标签data属性，方便管理（重构）

<input type=’buttn’ data-tips=’正在提交，请稍等’>

1. 比较新旧数值，先通过该条数据的唯一标识（id）获取旧值data{ key: val}对象，在将其与新值对象进行对比（for( var key in data){ }），返回flag，判断新旧值是否一致
2. 对于不同对象（资产平台domain/ip），但有多个key一致时，先将一致的key抽出，组成新的对象，common\_data{ key: val},再通过common\_data.private\_key,赋值私有属性
3. serialize() 方法通过序列化表单值，创建 URL 编码文本字符串。(表单name属性)  
   b=2&c=3&d=4&e=5
4. 单选/多选按钮未选中，则不传递该字段。
5. 方法传参时（可缺省），应将参数作为对象传入，方便维护

# // 从url获取参数

# getUrlParam: function (name) {

# var reg = new RegExp("(^|&)" + name + "=([^&]\*)(&|$)");

# var r = window.location.search.substr(1).match(reg);

# if (r != null) return unescape(r[2]); return null;

# },