|  |  |  |  |
| --- | --- | --- | --- |
| 文档状态 | 草稿 [] 已发布[ ] 修改中[ √ ] | | |
| 文档编号 | 通用进度条和按钮状态锁定使用说明 | | |
| 编 撰 | 万大龙 | 编撰日期 | 2015-8-31 |
| 保密级别 | 机密 | 文档版本 | V1.0.0 |
| 部 门/组 | 技术部 | | |

【**通用进度条和按钮状态锁定使用说明**】

**正方软件股份有限公司**

# 一、通用进度条使用说明

## 通用进度条的存在是为了解决部分功能执行时间较长，因页面长时间无任何反应而引起的误认为假死或者误操作。

## 1、进度条效果



## 2、模拟进度条

模拟进度，即在部分功能因业务逻辑执行时间不是很长，但是又不短，没有比较因此去在后台计算进度值进行实时获取又需要有进度的过程的功能中可采用此方式。

此方式为模拟Ajax请求各个步骤进行进度模拟。

### **（1）、Bootbox弹窗调用**

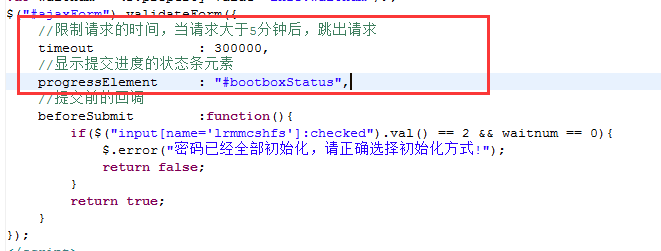
#### 步骤1：打开弹窗时添加启用进度条参数



打开弹窗时添加参数：

|  |
| --- |
| //是否需要进度条  "progress" : **true**, |

#### 步骤2：绑定校验时指定进度条元素



绑定校验时添加参数：

|  |
| --- |
| //限制请求的时间，当请求大于5分钟后，跳出请求  timeout : 300000,  //显示提交进度的状态条元素  progressElement : "#bootboxStatus", |

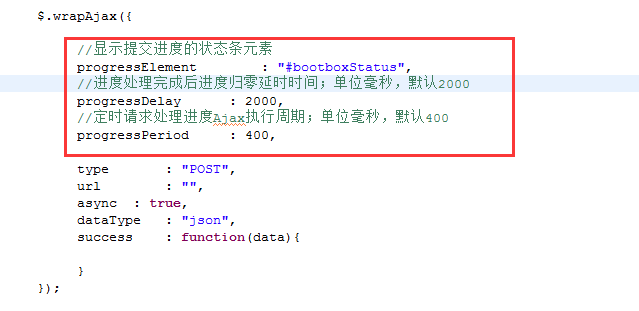
### （2）、自定义Ajax请求调用

#### 步骤1：在页面头部引入js

已经在统一的头部文件引入；参见“indexhead\_v5.ini”，“pagefunc\_v5.ini”

#### 步骤2：绑定校验时指定进度条元素

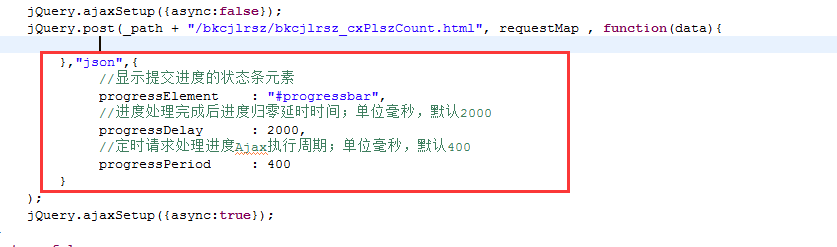
1)、Ajax请求



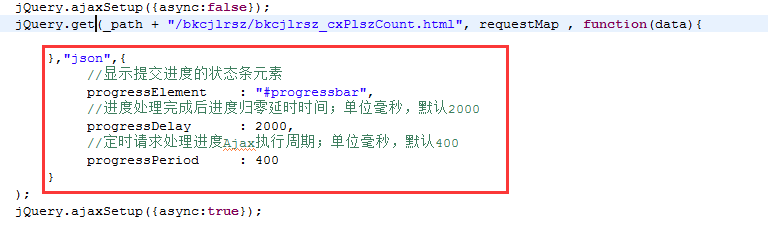
请注意：progressDelay和progressPeriod 参数可以不填

|  |
| --- |
| $.ajax({    //显示提交进度的状态条元素  progressElement : "#bootboxStatus",  //进度处理完成后进度归零延时时间；单位毫秒，默认2000  progressDelay : 2000,  //定时请求处理进度Ajax执行周期；单位毫秒，默认400  progressPeriod : 400,    type : "POST",  url : "",  async : **true**,  dataType : "json",  success : **function**(data){    }  }); |

2）、Post,Get 函数



或



|  |
| --- |
| jQuery.ajaxSetup({async:**false**});  jQuery.post(\_path + "/bkcjlrsz/bkcjlrsz\_cxPlszCount.html", requestMap , **function**(data){    },"json",{  //显示提交进度的状态条元素  progressElement : "#progressbar",  //进度处理完成后进度归零延时时间；单位毫秒，默认2000  progressDelay : 2000,  //定时请求处理进度Ajax执行周期；单位毫秒，默认400  progressPeriod : 400  }  );  jQuery.ajaxSetup({async:**true**}); |

## 3、实时进度条

实时进度，即在部分功能因业务逻辑执行时间过长，需要在后台计算进度值进行实时获取又需要有进度的过程的功能中可采用此方式。

### **（1）、Bootbox弹窗调用**

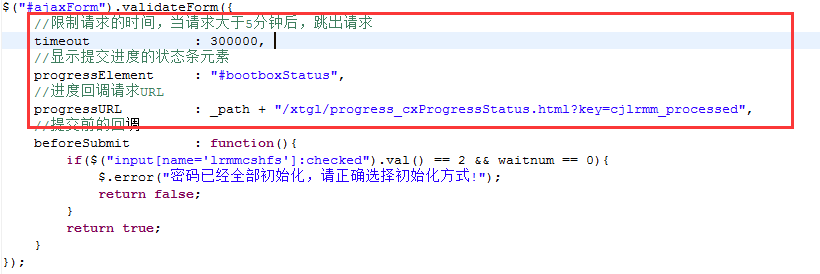
#### 步骤1：打开弹窗时添加启用进度条参数



打开弹窗时添加参数：

|  |
| --- |
| //是否需要进度条  "progress" : **true**, |

#### 步骤2：绑定校验时指定进度条元素



绑定校验时添加参数：

|  |
| --- |
| //限制请求的时间，当请求大于5分钟后，跳出请求  timeout : 300000,  //显示提交进度的状态条元素  progressElement: "#bootboxStatus",  //进度回调请求URL【该路径为通用路径，必须传参数key= 指定的值 】  progressURL: \_path + "/xtgl/progress\_cxProgressStatus.html?key=cjlrmm\_processed", |

#### 步骤3：后台逻辑实现实时计算进度



实现如下结构逻辑代码；期中：CJGLConstant.*BKCJLRMM\_PROCESSED\_KEY*是唯一值常量，该值对应这步骤二请求中的key

|  |
| --- |
| **try** {  //统计数  AtomicInteger completeCount = **new** AtomicInteger(0);  //查询数据；根据自己情况调整  List<BaseMap> infoList = dao.getKeyInfoByScope(model);  **if**(!BlankUtil.*isBlank*(infoList)){  //记录总数  BigDecimal maxCount = **new** BigDecimal(infoList.size());  //循环 记录  **for** (BaseMap infoMap : infoList) {  **if**(infoMap == **null**){  **continue**;  }  //此处写你自己的逻辑代码    //调用计算百分比方法，设置计算结果到session  PercentageUtils.*setPercentage*(completeCount, maxCount, CJGLConstant.*BKCJLRMM\_PROCESSED\_KEY*);  }    }**else**{  //设置标识到session中  SessionFactory.*getSession*().setAttribute(CJGLConstant.*BKCJLRMM\_PROCESSED\_KEY*, 100);  }  } **finally** {  //成绩更新处理完成，移除标识：这里防止页面突然关闭导致的未清除  SessionFactory.*getSession*().removeAttribute(CJGLConstant.*BKCJLRMM\_PROCESSED\_KEY*);  } |

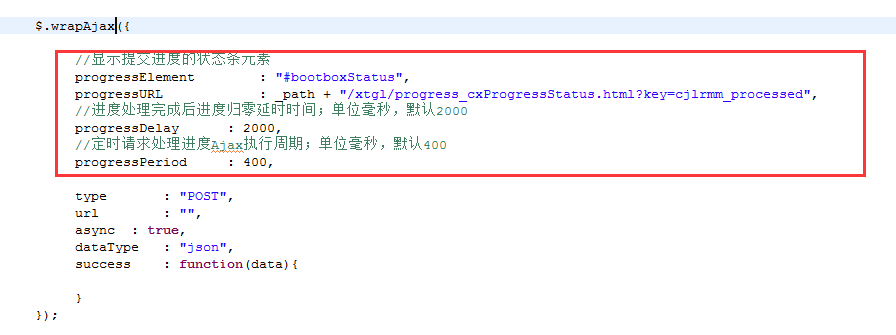
### **（2）**、自定义Ajax请求调用

#### 步骤1：在页面头部引入js

已经在统一的头部文件引入；参见“indexhead\_v5.ini”，“pagefunc\_v5.ini”

#### 步骤2：绑定校验时指定进度条元素

1）、Ajax函数



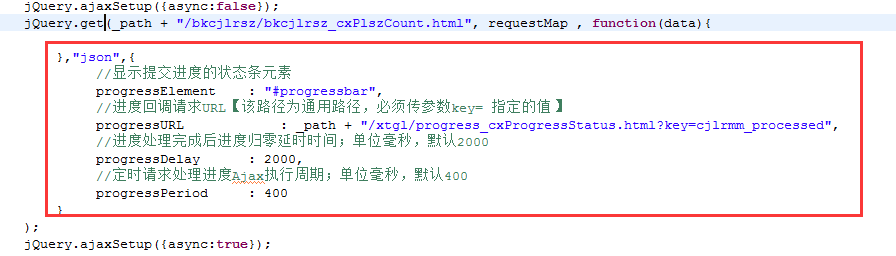
请注意：progressDelay和progressPeriod 参数可以不填

|  |
| --- |
| $.ajax({    //显示提交进度的状态条元素  progressElement : "#bootboxStatus",  //进度回调请求URL【该路径为通用路径，必须传参数key= 指定的值 】  progressURL : \_path + "/xtgl/progress\_cxProgressStatus.html?key=cjlrmm\_processed",  //进度处理完成后进度归零延时时间；单位毫秒，默认2000  progressDelay : 2000,  //定时请求处理进度Ajax执行周期；单位毫秒，默认400  progressPeriod : 400,  //实时进度时，完成一次百分比检查后的回调函数，默认$.noop  progressComplete:**function**(percentage){  },  type : "POST",  url : "",  async : **true**,  dataType : "json",  success : **function**(data){    }  }); |

1. 、Post,Get函数



或



|  |
| --- |
| jQuery.ajaxSetup({async:**false**});  jQuery.post(\_path + "/bkcjlrsz/bkcjlrsz\_cxPlszCount.html", requestMap , **function**(data){    },"json",{  //显示提交进度的状态条元素  progressElement : "#progressbar",  //进度回调请求URL【该路径为通用路径，必须传参数key= 指定的值 】  progressURL : \_path + "/xtgl/progress\_cxProgressStatus.html?key=cjlrmm\_processed",  //进度处理完成后进度归零延时时间；单位毫秒，默认2000  progressDelay : 2000,  //定时请求处理进度Ajax执行周期；单位毫秒，默认400  progressPeriod : 400,  //实时进度时，完成一次百分比检查后的回调函数，默认$.noop  progressComplete:**function**(percentage){  }  });  jQuery.ajaxSetup({async:**true**}); |

# 按钮状态锁定使用说明

### **（1）、Bootbox弹窗调用**

#### 步骤1：打开弹窗时添加启用进度条参数



打开弹窗时添加参数：

|  |
| --- |
| //是否点击按钮时，锁定按钮点击状态  "btnlock" : **true**, |

#### 步骤2-1：在自己的请求成功回调函数中重置按钮状态



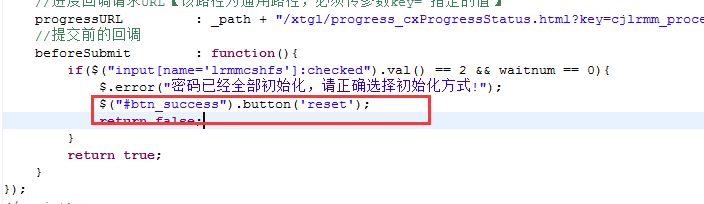
在弹窗按钮callback回调函数中适合位置执行 $this.reset();

|  |
| --- |
| **var** $this = **this**;  //没有通过校验需要重置按钮状态  **if(!$('#ajaxForm').isValid()){**  **$this.reset();**  **}**  submitForm("ajaxForm",**function**(responseText,statusText){  //请求完成需要重置按钮状态  $this.reset();  }); |

#### 步骤2-2：手动重置按钮状态



**因为部分情况下，回调函数不会被调用，需要手动去恢复按钮可点击状态；**



代码如下：

|  |
| --- |
| $("#btn\_success").button('reset'); |

### 、自定义Ajax请求调用

**参考Bootstrap 3帮助文档中 “javascript插件” - 按钮部分**



#### 步骤1：按钮添加属性 data-loading-text="Loading..."

#### 步骤2：在自己的Ajax请求过程中进行状态改变

|  |
| --- |
| $('#loading-example-btn').click(function(){varbtn=$(this)btn.button('loading')$.ajax(...).always(function(){btn.button('reset')});}); |