## 更新部门接口

### 1.1SysDeptService

*/\*\*  
 \* 更新部门实现  
 \*  
 \** ***@param:*** *[param]  
 \** ***@return:*** *void  
 \** ***@author:*** *nino  
 \** ***@date:*** *2019/5/29 15:21  
 \*/***public void** update(DeptParam param) {  
 BeanValidator.*check*(param);  
 **if**(checkExist(param.getParentId(), param.getName(), param.getId())) {  
 **throw new** ParamException(**"同一层级下存在相同名称的部门"**);  
 }  
 SysDept before = **sysDeptMapper**.selectByPrimaryKey(param.getId());  
 *//使用checkNotNull方法进行校验会抛空指针异常，还可以用自定义异常说明的方法* Preconditions.*checkNotNull*(before, **"待更新的部门不存在"**);  
 **if**(checkExist(param.getParentId(), param.getName(), param.getId())) {  
 **throw new** ParamException(**"同一层级下存在相同名称的部门"**);  
 }  
  
 SysDept after = SysDept.*builder*()  
 .id(param.getId())  
 .name(param.getName())  
 .parentId(param.getParentId())  
 .seq(param.getSeq())  
 .remark(param.getRemark())  
 .build();  
 after.setLevel(LevelUtil.*calculateLevel*(getLevel(param.getParentId()), param.getParentId()));  
 after.setOperator(RequestHolder.*getCurrentUser*().getUsername());  
 after.setOperateIp(IpUtil.*getRemoteIp*(RequestHolder.*getCurrentRequest*()));  
 after.setOperateTime(**new** Date());  
  
 *//更新子部门* updateWithChild(before, after);  
 **sysLogService**.saveDeptLog(before, after);  
}  
  
*/\*\*  
 \* 更新子部门  
 \* 更新当前部门及其子部门，为保证原子性，添加事务注解，要么都成功，要么都失败  
 \*  
 \** ***@param:*** *[before, after]  
 \** ***@return:*** *void  
 \** ***@author:*** *nino  
 \** ***@date:*** *2019/5/29 17:11  
 \*/*@Transactional  
**private void** updateWithChild(SysDept before, SysDept after) {  
 String newLevelPrefix = after.getLevel();  
 String oldLevelPrefix = before.getLevel();  
 *//判断更新后的level与更新前的level是否相等，不相等才需要更新子部门* **if** (!after.getLevel().equals(before.getLevel())) {  
 List<SysDept> deptList = **sysDeptMapper**.getChildDeptListByLevel(before.getLevel());  
 **if** (CollectionUtils.*isNotEmpty*(deptList)) {  
 **for** (SysDept dept : deptList) {  
 String level = dept.getLevel();  
 **if** (level.indexOf(oldLevelPrefix) == 0) {  
 level = newLevelPrefix + level.substring(oldLevelPrefix.length());  
 dept.setLevel(level);  
 }  
 }  
 **sysDeptMapper**.batchUpdateLevel(deptList);  
 }  
 }  
 **sysDeptMapper**.updateByPrimaryKey(after);  
}

### 1.2SysDeptMapper

SysDeptMapper中添加getChildDeptListByLevel、batchUpdateLevel方法

*/\*\*  
 \* 根据指定level的子部门及其子部门  
 \*/*List<SysDept> getChildDeptListByLevel(@Param(**"level"**) String level);  
  
*/\*\*  
 \* 批量更新level  
 \*/***void** batchUpdateLevel(@Param(**"sysDeptList"**) List<SysDept> sysDeptList);

在映射文件中编写相关的SQL语句：

<**select id="getChildDeptListByLevel" resultMap="BaseResultMap"**>  
 select  
 <**include refid="Base\_Column\_List"** />  
 from sys\_dept  
 WHERE level like #{level} || '.%'  
</**select**>  
  
<**update id="batchUpdateLevel" parameterType="map"**>  
 <**foreach collection="sysDeptList" item="sysDept" separator=";"**>  
 UPDATE sys\_dept  
 SET level = #{dept.level}  
 WHERE id = #{dept.id}  
 </**foreach**>  
</**update**>

%为通配符，匹配一个或多个字符

### 1.3SysDeptController

*/\*\*  
 \* 更新部门请求  
 \*/*@RequestMapping(**"/update.json"**)  
@ResponseBody  
**public** JsonData updateDept(DeptParam param) {  
 **sysDeptService**.update(param);  
 **return** JsonData.*success*();  
}