

一、科室列表（接口）



1、添加service接口和实现

(1) 在DepartmentService定义方法

```
1 //根据医院编号，查询医院所有科室列表
2 List<DepartmentVo> findDeptTree(String hoscode);
```

(2) 在DepartmentServiceImpl实现方法

```
1 //根据医院编号，查询医院所有科室列表
2 @Override
3 public List<DepartmentVo> findDeptTree(String hoscode) {
4     //创建list集合，用于最终数据封装
5     List<DepartmentVo> result = new ArrayList<>();
6
7     //根据医院编号，查询医院所有科室信息
8     Department departmentQuery = new Department();
9     departmentQuery.setHoscode(hoscode);
```

```

10 Example example = Example.of(departmentQuery);
11 //所有科室列表 departmentList
12 List<Department> departmentList = departmentRepository.findAll(example);
13
14 //根据大科室编号 bigcode 分组，获取每个大科室里面下级子科室
15 Map<String, List<Department>> departmentMap =
16     departmentList.stream().collect(Collectors.groupingBy(Department::getBigc
17 //遍历map集合 departmentMap
18 for(Map.Entry<String,List<Department>> entry : departmentMap.entrySet()) {
19     //大科室编号
20     String bigcode = entry.getKey();
21     //大科室编号对应的全局数据
22     List<Department> deparment1List = entry.getValue();
23     //封装大科室
24     DepartmentVo departmentVo1 = new DepartmentVo();
25     departmentVo1.setDepcode(bigcode);
26     departmentVo1.setDepname(deparment1List.get(0).getBigname());
27
28     //封装小科室
29     List<DepartmentVo> children = new ArrayList<>();
30     for(Department department: deparment1List) {
31         DepartmentVo departmentVo2 = new DepartmentVo();
32         departmentVo2.setDepcode(department.getDepcode());
33         departmentVo2.setDepname(department.getDepname());
34         //封装到list集合
35         children.add(departmentVo2);
36     }
37     //把小科室list集合放到大科室children里面
38     departmentVo1.setChildren(children);
39     //放到最终result里面
40     result.add(departmentVo1);
41 }
42 //返回
43 return result;
44 }

```

2、添加DepartmentController方法

```

1 @RestController
2 @RequestMapping("/admin/hosp/department")

```

```

3 public class DepartmentController {
4
5     @Autowired
6     private DepartmentService departmentService;
7
8     //根据医院编号，查询医院所有科室列表
9     @ApiOperation(value = "查询医院所有科室列表")
10    @GetMapping("getDeptList/{hoscode}")
11    public R getDeptList(@PathVariable String hoscode) {
12        List<DepartmentVo> list = departmentService.findDeptTree(hoscode);
13        return R.ok().data("list",list);
14    }
15 }

```

二、科室列表（前端）

1、添加隐藏路由

```

1 {
2   path: 'hospital/schedule/:hoscode',
3   name: '排班',
4   component: () => import('@/views/hosp/schedule'),
5   meta: { title: '排班', noCache: true },
6   hidden: true
7 }

```

2、封装api方法

```

1 //查看医院科室
2 getDeptByHoscode(hoscode) {
3     return request ({
4         url: `/admin/hosp/department/getDeptList/${hoscode}`,
5         method: 'get'
6     })
7 },

```

3、添加/views/hosp/schedule.vue组件

```
1 <template>
2   <div class="app-container">
3     <div style="margin-bottom: 10px;font-size: 10px;">选择: </div>
4     <el-container style="height: 100%">
5       <el-aside width="200px" style="border: 1px silver solid">
6         <!-- 部门 -->
7         <el-tree
8           :data="data"
9           :props="defaultProps"
10          :default-expand-all="true"
11          @node-click="handleNodeClick">
12         </el-tree>
13       </el-aside>
14       <el-main style="padding: 0 0 0 20px;">
15         <el-row style="width: 100%">
16           <!-- 排班日期 分页 -->
17         </el-row>
18         <el-row style="margin-top: 20px;">
19           <!-- 排班日期对应的排班医生 -->
20         </el-row>
21       </el-main>
22     </el-container>
23   </div>
24 </template>
25 <script>
26 import hospApi from '@api/yygh/hosp'
27 export default {
28   data() {
29     return {
30       data: [],
31       defaultProps: {
32         children: 'children',
33         label: 'depname'
34       },
35       hoscode: null
36     }
37   },
38   created(){
39     this.hoscode = this.$route.params.hoscode
```

```
40     this.fetchData()
41   },
42   methods:{
43     fetchData() {
44       hospApi.getDeptByHoscode(this.hoscode)
45         .then(response => {
46           this.data = response.data.list
47         })
48     }
49   }
50 }
51 </script>
52 <style>
53   .el-tree-node.is-current > .el-tree-node__content {
54     background-color: #409EFF !important;
55     color: white;
56   }
57
58   .el-checkbox__input.is-checked+.el-checkbox__label {
59     color: black;
60   }
61 </style>
```

说明：底部style标签是为了控制树形展示数据选中效果的