芋道源码 2022-03-26

分页实现

- 前端：基于 Element UI 分页组件 Pagination
- 后端：基于 MyBatis Plus 分页功能，二次封装

以 [系统管理 -> 租户管理 -> 租户列表] 菜单为例子，讲解它的分页 + 搜索的实现。

1. 前端分页实现

1.1 Vue 界面

界面 `tenant/index.vue` 相关的代码如下：

```
<template>
    <!-- 搜索工作栏 -->
    <el-form :model="queryParams" ref="queryForm" size="small" :inline="true" v-
      <el-form-item label="租户名" prop="name">
        <el-input v-model="queryParams.name" placeholder="请输入租户名" clearable
      </el-form-item>
      <el-form-item label="联系人" prop="contactName">
        <el-input v-model="queryParams.contactName" placeholder="请输入联系人" cl
      </el-form-item>
      <el-form-item label="联系手机" prop="contactMobile">
        <el-input v-model="queryParams.contactMobile" placeholder="请输入联系手机
      </el-form-item>
      <el-form-item label="租户状态" prop="status">
        <el-select v-model="queryParams.status" placeholder="请选择租户状态" clea
          <el-option v-for="dict in this.getDictDatas(DICT_TYPE.COMMON_STATUS)"
                     :key="dict.value" :label="dict.label" :value="dict.value"
        </el-select>
      </el-form-item>
      <el-form-item>
        <el-button type="primary" icon="el-icon-search" @click="handleQuery">搜索
        <el-button icon="el-icon-refresh" @click="resetQuery">重置</el-button>
      </el-form-item>
    </el-form>

    <!-- 列表 -->
    <el-table v-loading="loading" :data="list">
        <!-- 省略每一列... -->
    </el-table>
```

```
    <!-- 分页组件 -->
    <pagination v-show="total > 0" :total="total" :page.sync="queryParams.pageNc
                @pagination="getList"/>

</template>

<script>
import { getTenantPage } from "@/api/system/tenant";

export default {
    name: "Tenant",
    components: {},
    data() {
      // 遮罩层
      return {
        // 遮罩层
        loading: true,
        // 显示搜索条件
        showSearch: true,
        // 总条数
        total: 0,
        // 租户列表
        list: [],
        // 查询参数
        queryParams: {
          pageNo: 1,
          pageSize: 10,
          // 搜索条件
          name: null,
          contactName: null,
          contactMobile: null,
          status: undefined,
        },
      }
    },
    created() {
      this.getList();
    },
    methods: {
      /** 查询列表 */
      getList() {
        this.loading = true;
        // 处理查询参数
        let params = {...this.queryParams};
        // 执行查询
        getTenantPage(params).then(response => {
```

```
            this.list = response.data.list;
            this.total = response.data.total;
            this.loading = false;
          });
      },
      /** 搜索按钮操作 */
      handleQuery() {
        this.queryParams.pageNo = 1;
        this.getList();
      },
      /** 重置按钮操作 */
      resetQuery() {
        this.resetForm("queryForm");
        this.handleQuery();
      }
    }
}
</script>
```

1.2 API 请求

请求 `system/tenant.js` 相关的代码如下：

```
import request from '@/utils/request'

// 获得租户分页
export function getTenantPage(query) {
  return request({
    url: '/system/tenant/page',
    method: 'get',
    params: query
  })
}
```

2. 后端分页实现

2.1 Controller 接口

在 TenantController 类中，定义 `/admin-api/system/tenant/page` 接口。代码如下：

```
@Tag(name = "管理后台 - 租户")
@RestController
@RequestMapping("/system/tenant")
```

```
public class TenantController {

    @Resource
    private TenantService tenantService;

    @GetMapping("/page")
    @Operation(summary = "获得租户分页")
    @PreAuthorize("@ss.hasPermission('system:tenant:query')")
    public CommonResult<PageResult<TenantRespVO>> getTenantPage(@Valid TenantPag
        PageResult<TenantDO> pageResult = tenantService.getTenantPage(pageVO);
        return success(TenantConvert.INSTANCE.convertPage(pageResult));
    }

}
```

- Request 分页请求，使用 TenantPageReqVO 类，它继承 PageParam 类
- Response 分页结果，使用 PageResult 类，每一项是 TenantRespVO 类

分页请求，需要继承 PageParam 类。代码如下：

```
@Schema(description="分页参数")
@Data
public class PageParam implements Serializable {

    private static final Integer PAGE_NO = 1;
    private static final Integer PAGE_SIZE = 10;

    @Schema(description = "页码，从 1 开始", required = true,example = "1")
    @NotNull(message = "页码不能为空")
    @Min(value = 1, message = "页码最小值为 1")
    private Integer pageNo = PAGE_NO;

    @Schema(description = "每页条数，最大值为 100", required = true, example = "10
    @NotNull(message = "每页条数不能为空")
    @Min(value = 1, message = "每页条数最小值为 1")
    @Max(value = 100, message = "每页条数最大值为 100")
    private Integer pageSize = PAGE_SIZE;

}
```

分页条件，在子类中进行定义。以 TenantPageReqVO 举例子，代码如下：

```
@Schema(description = "管理后台 - 租户分页 Request VO")
@Data
@EqualsAndHashCode(callSuper = true)
@ToString(callSuper = true)
public class TenantPageReqVO extends PageParam {

    @Schema(description = "租户名", example = "芋道")
    private String name;

    @Schema(description = "联系人", example = "芋艿")
    private String contactName;

    @Schema(description = "联系手机", example = "15601691300")
    private String contactMobile;

    @Schema(description = "租户状态（0正常 1停用）", example = "1")
    private Integer status;

    @DateTimeFormat(pattern = FORMAT_YEAR_MONTH_DAY_HOUR_MINUTE_SECOND)
    @Schema(description = "创建时间")
    private LocalDateTime[] createTime;

}
```

2.1.2 分页结果 PageResult

分页结果 PageResult 类，代码如下：

```
@Schema(description = "分页结果")
@Data
public final class PageResult<T> implements Serializable {

    @Schema(description = "数据", required = true)
    private List<T> list;

    @Schema(description = "总量", required = true)
    private Long total;

}
```

分页结果的数据 `list` 的每一项，通过自定义 VO 类，例如说 TenantRespVO 类。

2.2 Mapper 查询

在 TenantMapper 类中，定义 selectPage 查询方法。代码如下：

```
@Mapper
public interface TenantMapper extends BaseMapperX<TenantDO> {

    default PageResult<TenantDO> selectPage(TenantPageReqVO reqVO) {
        return selectPage(reqVO, new LambdaQueryWrapperX<TenantDO>()
                .likeIfPresent(TenantDO::getName, reqVO.getName()) // 如果 name
                .likeIfPresent(TenantDO::getContactName, reqVO.getContactName())
                .likeIfPresent(TenantDO::getContactMobile, reqVO.getContactMobil
                .eqIfPresent(TenantDO::getStatus, reqVO.getStatus()) // 如果 sta
                .betweenIfPresent(TenantDO::getCreateTime, reqVO.getBeginCreateT
                .orderByDesc(TenantDO::getId)); // 按照 id 倒序
    }

}
```

针对 MyBatis Plus 分页查询的二次分装，在 BaseMapperX 中实现，主要是将 MyBatis 的分页结果 IPage，转换成项目的分页结果 PageResult。代码如下图：

← 参数校验 文件存储（上传下载）→