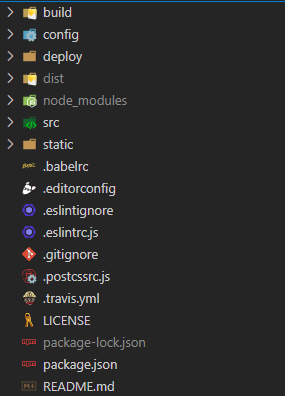
SKOJ开发文档

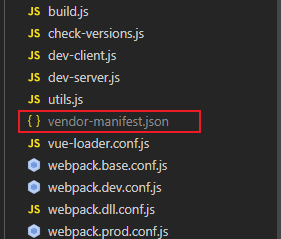
# 前端



node:18.13.0

npm:8.19.3

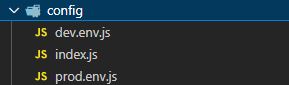
## build目录



构建项目相关的脚本-不一一说明了

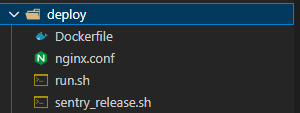
其中vendor-manifest.json是通过npm run build:dll生成的（同时生成的还有/static/js/vendor.dll\*.js）

## config目录



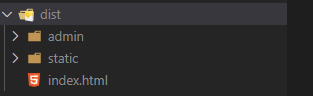
配置相关脚本（包含开发环境和生产环境的配置）

## deploy目录



部署相关脚本（nginx+docker）

## dist目录



打包生成的文件（npm run build）

## node\_modules目录

descript

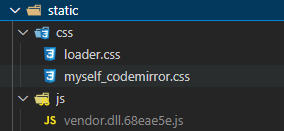
依赖相关目录

（

{ parser: "babylon" } is deprecated; we now treat it as { parser: "babel" }. 报错解决：/node\_modules/vue-loader/lib/template-compiler/index.js babylon 替换为 babel

）

## static目录



静态资源目录

[Vue项目中根目录.文件配置详解（3）.eslintignore、.eslintrc.js、.stylelintrc.js、.gitignore](https://blog.csdn.net/qq_37734787/article/details/109813918)

## .babelrc文件

ES6是2015年发布的下一代javascript语言标准，它引入了新的语法和API，使我们编写js代码更加得心应手，比如class，let,for…of promise等等这样的，但是可惜的是这些js新特性只被最新版本的浏览器支持，但是低版本浏览器并不支持，那么低版本浏览器下就需要一个转换工具，把es6代码转换成浏览器能识别的代码，babel就是这样的一个工具。可以理解为 babel是javascript语法的编译器。

## .editorconfig文件

EditorConfig 有助于为跨各种编辑器和 IDE 处理同一项目的多个开发人员维护一致的编码风格。EditorConfig 项目由用于定义编码样式的文件格式和一组文本编辑器插件组成，这些插件使编辑器能够读取文件格式并遵循定义的样式。

.editorconfig 自定义文件是用来定义编辑器的编码格式规范，编辑器的行为会与 .editorconfig 文件中定义的一致，并且其优先级比编辑器自身的设置要高。

## .eslintignore文件

主要作用是忽略哪些文件的语法检查

## .eslintrc.js

js代码的语法检查

代码提示出现很多奇奇怪怪的错误提示，因为是eslintrc.js文件没有配置相关命令

默认eslint规则：

代码末尾不能加分号 ;

代码中不能存在多行空行；

tab键不能使用，必须换成两个空格；

代码中不能存在声明了但未使用的变量；

最简单的方法，关闭eslint检测，把 build/webpack.base.conf.js 配置文件中的eslint rules注释掉即可。但不推荐这么做，eslint检测是有必要的，能保持良好的代码风格。

## .gitignore

我们做的每个Git项目中都需要一个“.gitignore”文件，这个文件的作用就是告诉Git哪些文件不需要添加到版本管理中。

## .postcssrc.js

PostCss是一个处理 CSS 的处理工具，Vue CLI 内部使用了 PostCSS。本身功能比较单一，它主要负责解析 CSS 代码，再交由插件来进行处理，它的插件体系非常强大，所能进行的操作是多种多样的

## .travis.yml

[偶然在github开源项目中发现了.travis.yml这货](https://blog.csdn.net/albertsh/article/details/125581553)

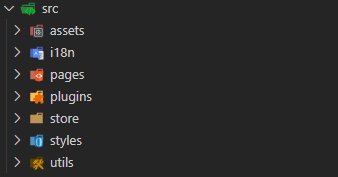
持续集成是将构建并测试的过程自动化，在你提交代码时，持续集成服务能够自动触发构建与测试过程，并反馈结果，加快开发周期，同时减少脏代码的引入，而持续集成工具有很多，常见的包括 Jenkins、Gitlab-CI、Travis CI 和 AppVeyor，github上项目的持续集成可以选择使用 Travis CI，也有项目使用 AppVeyor，它们都是开源持续集成云服务。

.travis.yml 是 github 用于说明持续集成步骤配置文件，使用的语言是 YAML。

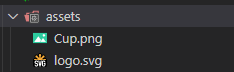
## package

依赖包相关版本

## src目录



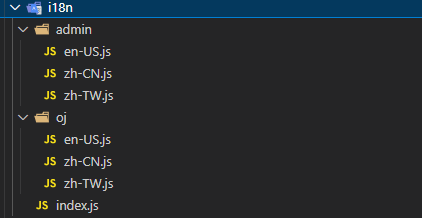
### assets目录



[vue中assets和static目录的区别](https://www.cnblogs.com/yanggb/p/12452389.html)

用于存放图片，比如logo等

### i18n目录



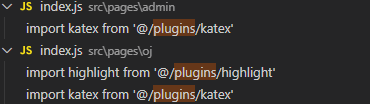
存放着系统的区域语言设置，支持多种字符集的转换，避免出现乱码

分为简体、繁体、英文，有部分映射不全，需要进一步补充

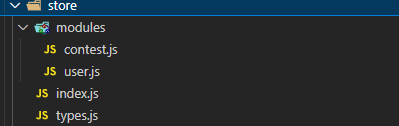
### plugins目录

descript

注册使用高亮插件，以及数学公式插件

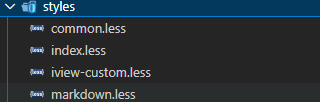


### store目录



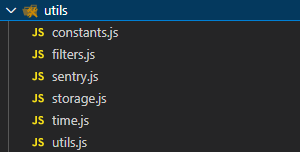
全局变量以及全局函数

### styles目录



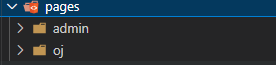
样式目录（本项目中仅支持less/css语法，scss需要重新更新下依赖以便防止冲突）

### utils目录

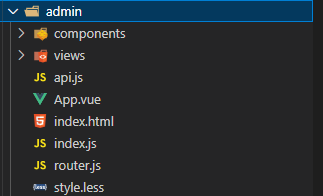


工具目录，常量、工具函数等

### pages目录



#### admin目录-后台页面



各个页面之间的逻辑可以通过查看route.js查明

components是提供复用的组件

views是各个页面

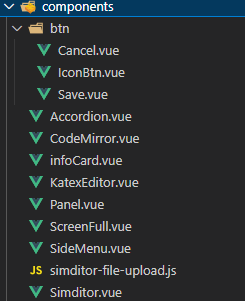
api.js是封装的接口调用

App.vue Vue项目文件

index.html Vue项目文件

index.js Vue项目文件

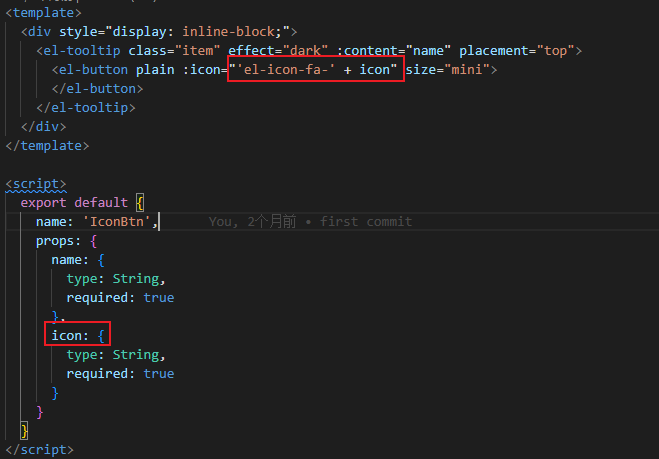
##### components目录



###### Cancel.vue

普通的el-button复用---显示关闭

###### IconBtn.vue:



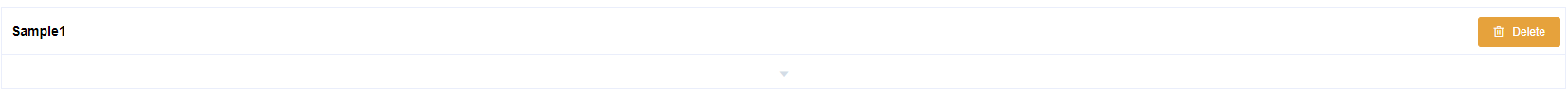
使用的font-awesome

###### Save.vue

普通的el-button复用---显示保存

###### Accordion.vue





折叠面板

###### CodeMirror.vue

代码编辑组件，目前在admin中未使用，可以用作预览std代码

###### infoCard.vue

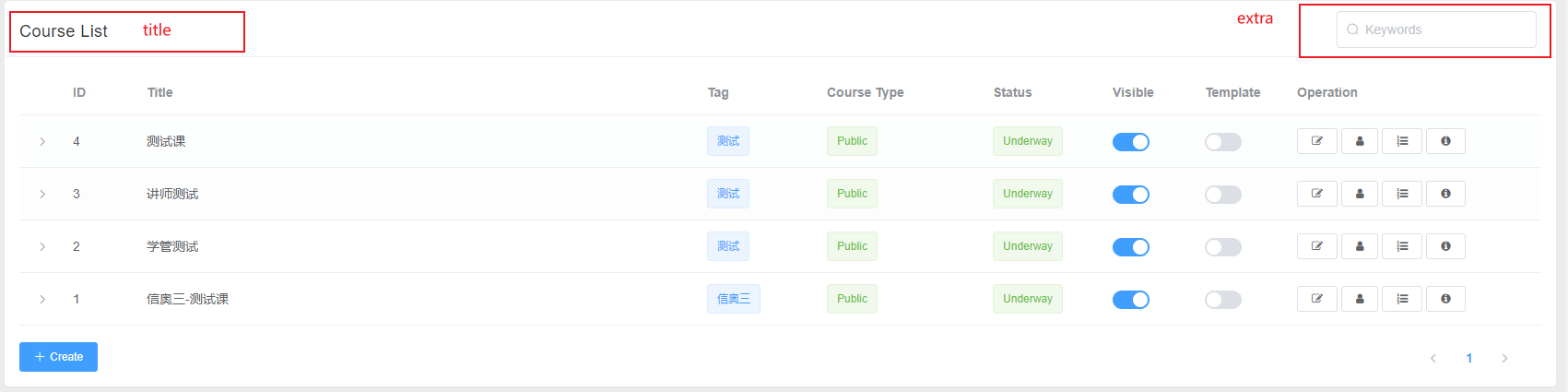
暂未使用

###### KatexEditor.vue



数学公式编辑组件

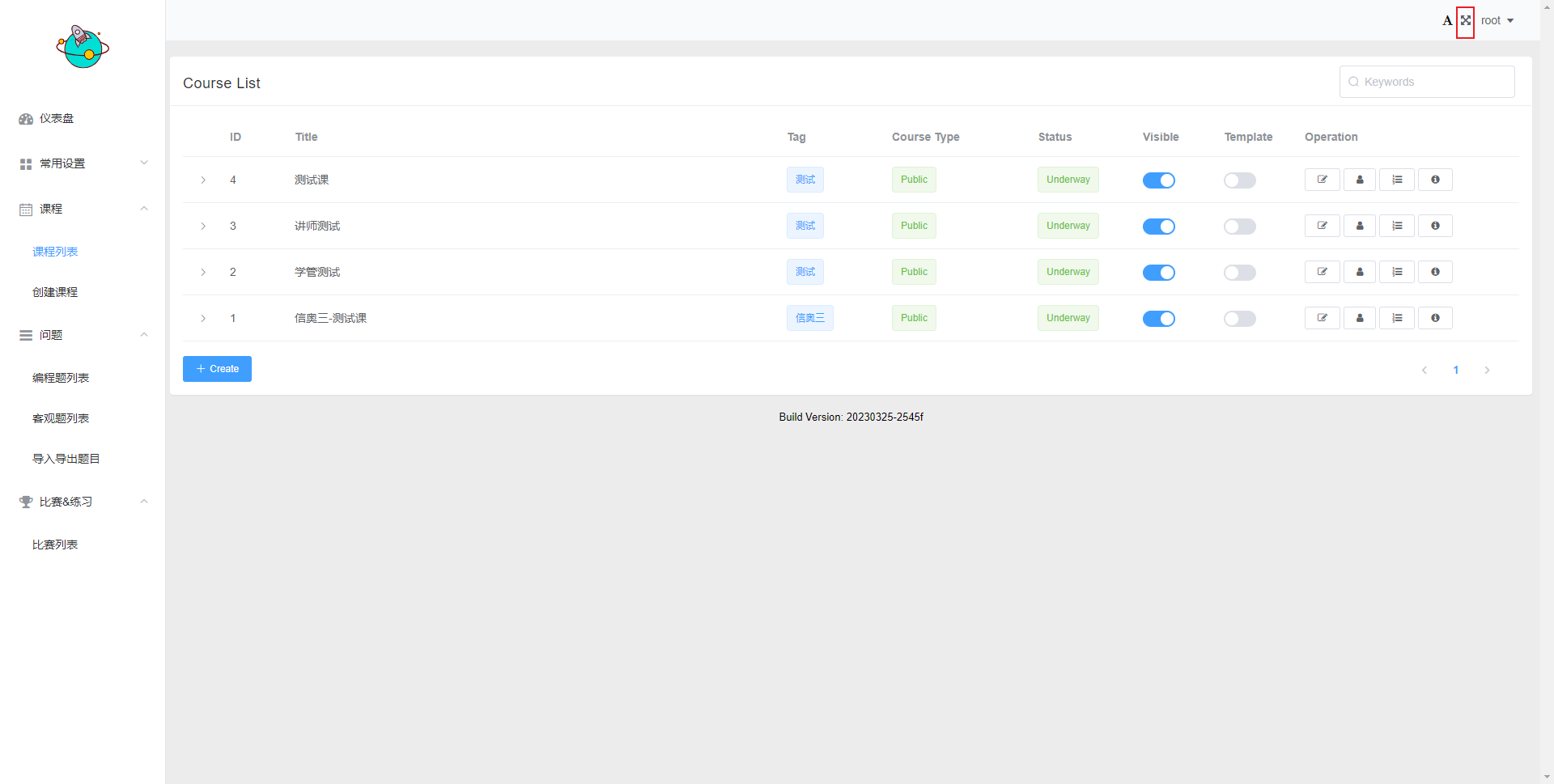
###### Panel.vue



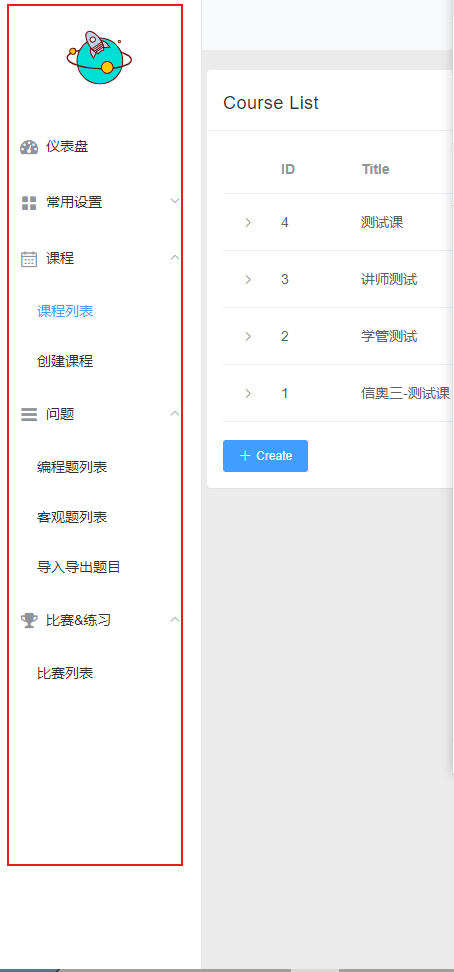
面板组件-复用很多次

###### ScreenFull.vue

全屏功能

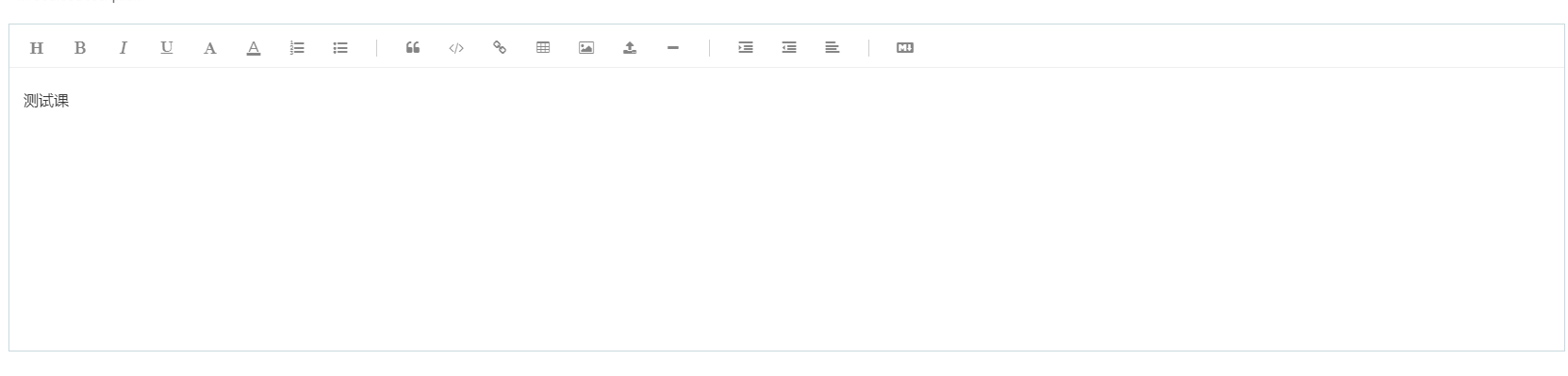


###### SideMenu.vue



侧边导航菜单

###### simditor.vue

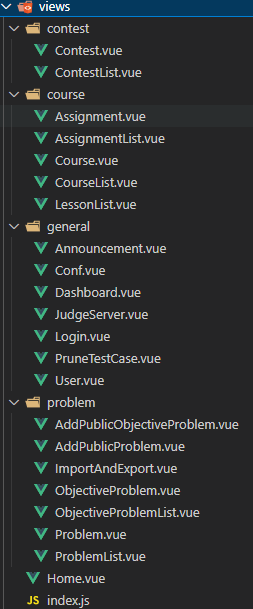


富文本编辑组件

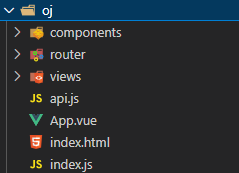
###### simditor-file-upload.js

管理文件上传到后端的功能实现

##### views目录



#### oj目录-前台页面



各个页面之间的逻辑可以通过查看router/route.js查明

components是提供复用的组件

router中有着全局身份确认以及路由配置

views是各个页面

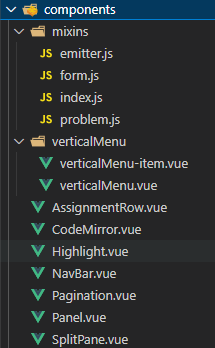
api.js是封装的接口调用

App.vue Vue项目文件

index.html Vue项目文件

index.js Vue项目文件

##### components目录



###### mixins

复用部分：emitter(跳转路由)，problem(通过率)，form(表单检验)

###### verticalMenu

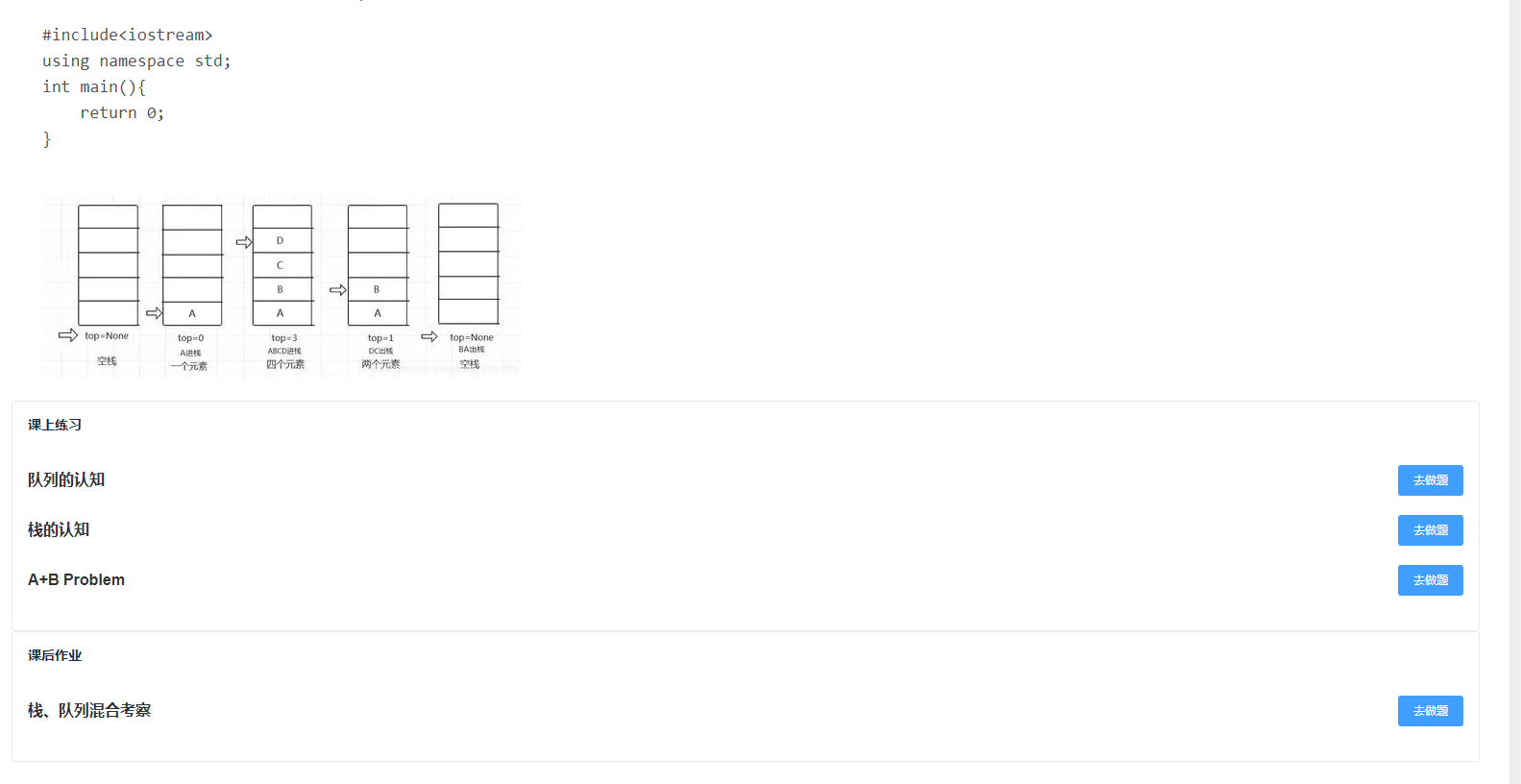


主要运用在竞赛页面，作为左侧悬浮的面板组件

###### AssignmentRow.vue



用来展示单个作业任务(作业任务内涵多个竞赛(竞赛内部内含多个题目))，可以点击上方的标题展开



（后期需要改变竞赛的显示页面----改成试卷的风格，可以有个导航组件用来跳转题目）

###### CodeMirror.vue

代码编辑组件，在编程详情页使用

###### Highlight.vue

代码高亮组件，在编程题提交记录详情页使用(后期提交记录需要做成弹出框，而不是单独的页面)

###### NavBar.vue

顶部导航栏

###### Pagination.vue

分页组件，底部页码组件，在列表页会进行使用

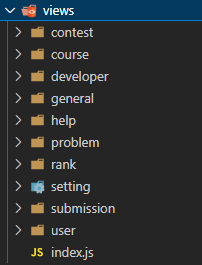
###### Panel.vue

面板组件，是对Card组件的修改样式后的复用

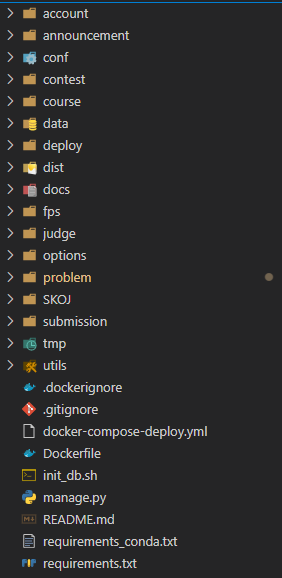
###### SplitPane.vue

可拖拽变换左右布局的组件，目前暂未使用，期望在编程题中使用

##### views目录



# 后端



# 数据库

# 其他

## 权限问题

权限主要集中在课程、题目、提交记录

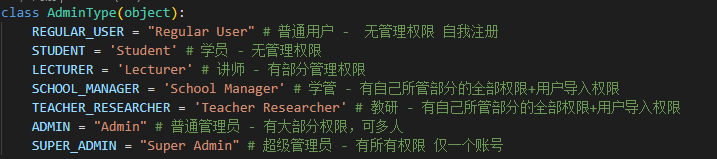
故有两种实现方式，对于每个账号均设置各个维度上的权限/对于账号进行特定的分权限管理

对于第一种方式：

后端有一部分已经如此实现了：

每个维度有着None,Own,All三个级别

第二种方式下：



目前是两种方式混用

因为对于管理级别的角色有三种，讲师、学管、教研，但实际中可能有一人身兼多职，需要到多种权限，切换账号显然不是一个好的处理方式，故可以在原有的权限上，扩展其他模块的权限

也就是

**对于student与regularuser都是固定的权限**

**对于具有管理权限的角色有着默认的权限，并且可以通过超级管理员继续权限的升级。**