链滴

swagger3.0 文档——json 格式的配置怎么写?

作者：limanting

原文链接：https://ld246.com/article/1628936417214

来源网站：链滴

swagger文档：https://swagger.io/specification/

引入swagger-ui

使用 koa2-swagger-ui

```
const koaSwagger = require("koa2-swagger-ui").koaSwagger; // 引入包

app.use(
  koaSwagger({
    routePrefix: "/api/swagger", //写上swagger页面的路由
    swaggerOptions: {
      url: "swagger.json",// swagger.js文件
    },
  })
);
```

swagger配置文件的json写法

json文件里只有一个大json

基本配置

- "openapi": "3.0.3" swagger的版本，这里用的是3.0版本。
- "info": {} 描述文档的基本信息。其中可以填充"description" 、"version" 、"title" 这些字段。

例：

```
  "info": {
    "description": "iconfont平台node服务端",
    "version": "1.0.0",
    "title": "IconFont Node Server Interface"
  },
```

- "externalDocs": {} 该文档的其他文档，可增加链接。其中可以填充"description" 、"url"
- "host": "local:3000" 本地开发时的域名和端口
- "schemes":["http","https"] 本地开发时可选的协议
- "securityDefinitions" 定义安全策略

例：

```
  "securityDefinitions": {
    "cookie": {
      "type": "apiKey",
      "name": "cookie",
      "in": "header"
    }
  },
```

接口分类

● "tags":[] 数组中的元素是包含"name"（标记该类别的名称）和"description"（对该类别的描述）段的对象。

例：

```
  "tags": [
    {
      "name": "user",
      "description": "用户相关接口"
    },
    {
      "name": "icon",
      "description": "图标相关接口"
    },
    {
      "name": "atlas",
      "description": "图标仓库相关接口"
    },
    {
      "name": "project",
      "description": "项目相关接口"
    }
  ],
```

接口

所有接口都写在 "path"{}字段当中。

举例：

```
"path": {
    "/api/user/ldap": { // 字段是接口
      "get": { // 请求方法
        "tags": [ // 对应的接口分类
          "user"
        ],
        "summary": "获取当前用户的ldap", // 接口描述
        "description": "获取当前用户的ldap", // 接口描述
        "operationId": "login", // 操作id（仅标记作用）
        "responses": { // 返回情况的说明
          "200": {
            "description": "ok"
          },
          "401": {
            "description": "未登录，没有权限"
          }
        }
      }
    }
```

get 方法带参数

```
    "/api/project/detail/{id}": { // 接口如果带参数，用花括号包住
```

```
  "get": {
    "tags": [
      "project"
    ],
    "summary": "获取项目详情",
    "description": "返回项目详情",
    "operationId": "getMyProjectDetail",
    "parameters": [ // url中带有的参数描述
      {
        "name": "id", // 对应url中花括号包住的id
        "in": "path", //描述是在url的路径中还是在？号后的（“path” 或“query”）
        "require": true,
        "schema": { // 描述参数的类型
          "type": "integer"
        }
      }
    ],
    "responses": {
      "200": {
        "description": "ok"
      },
      "401": {
        "description": "未登录，没有权限"
      }
    }
  }
},
```

基本post方法

```
  "/api/project/search": {
    "post": { // post 方法
      "tags": [
        "project"
      ],
      "summary": "搜索项目",
      "description": "返回可能的项目",
      "operationId": "searchMyProject",
      "requestBody": { // 请求参数
        "description": "输入名字", //请求参数描述
        "content": { // 请求内容
          "application/json": { // 请求参数的类型
            "schema": { // 描述请求参数类型
              "type": "object",
              "properties": { // 包含的属性
                "projectName": {
                  "type": "string"
                }
              }
            }
          }
        },
        "required": true
      },
```

```
    "responses": {
      "200": {
        "description": "ok"
      },
      "401": {
        "description": "未登录，没有权限"
      }
    }
  }
 }
}
```

post 带有上传文件

```
  "/api/icon/create": {
    "post": {
      "tags": [
        "icon"
      ],
      "summary": "上传icon",
      "description": "返回更新后的该图集",
      "operationId": "addIcon",
      "requestBody": {
        "description": "icon的内容和图集的id",
        "content": {
          "multipart/form-data": {
            "schema": {
              "type": "object",
              "properties": {
                "atlasId": {
                  "type": "number"
                },
                "icon": {
                  "type": "string",
                  "format": "binary"  // 如果是文件，就这么写
                }
              }
            }
          }
        },
        "required": true
      },
      "responses": {
        "200": {
          "description": "ok"
        },
        "401": {
          "description": "未登录，没有权限"
        }
      }
    }
  },
```

文件传输到后端时，会将文件挂载到 ctx.request.files 下面，我们可以从 ctx.request.files.icon中拿

我们上传的icon文件。

数据结构的统一写法

写在 "components": { "schemas":{}}里面

例：

```
components": {
    "schemas":{
     "UserModel": { // 自定义的数据结构
      "type": "object",
      "properties": {
       "id": {
        "type": "integer",
        "format": "int64"
       },
       "ldapId": {
        "type": "string"
       }
      }
     },
}}
```

使用方法：

```
    "/api/project/create": {
     "post": {
      "tags": [
       "project"
      ],
      "summary": "新增项目",
      "description": "返回更新后的我管理的项目和我参加的项目",
      "operationId": "createMyProject",
      "requestBody": {
       "description": "返回更新后的我管理的项目和我参加的项目",
       "content": {
        "application/json": {
         "schema": {
          "$ref": "#/components/schemas/ProjectModel" // 在schema里面，字段为 $ref 表示
用 ， 内容为路径
         }
        }
       },
       "required": true
      },
      "responses": {
       "200": {
        "description": "ok"
       },
       "401": {
        "description": "未登录，没有权限"
       }
      }
```

```
    }
  },
```

原文链接：swagger3.0 文档——json 格式的配置怎么写?