vue 组件

vue3 脚手架基础版本 - 自动生成单文件组件

本章创建一个vue3 项目的最最基本脚手架，可以自动生成vue 单文件组件。

为什么是最最基本，不是很完善的？

本来想可以弄的很完善，但是因为考虑每个项目文件组织不同，也可能每一个人的习惯不同。所以本章完成一个比较通用的。目的是为了给各位小伙伴最最通用和可扩展的，让大家可以自由发挥，而且不用从0开始。

简单描述

基于promise 完成的，每一个功能都划分很仔细，代码不长。对扩展非常友好。

翠花，上代码

```
const inquirer = require('inquirer');
const path = require('path');
const fs = require('fs');
const singleFileComponentTemplate = `<template>
</template>
<script>
</script>
<style>
</style>
`;
const defaultConfig = {
  version: '1.0.0',
  componentName: '',
  type: '',
  desc: '',
  sort: '',
};
const componentsPath = path.join(__dirname, '../src/components/');
boot();
function boot() {
  inquirer
    .prompt([
      {
        type: 'input',
        name: 'componentName',
```

```
        message:
          '组件的名字（vue3建议组件名字是英文组成，首字母大写，比如 MyComponent）',
        validate(value) {
          return value && value.match(/^[A-Za-z]/) && true;
        },
      },
      {
        type: 'input',
        name: 'desc',
        message: '组件的简单描述，最多30个字: ',
        validate(value) {
          return true;
        },
      },
      {
        type: 'rawlist',
        name: 'type',
        message: '请选择类型: ',
        choices: ['component'],
        validate(value) {
          return value && /^[1]$/.test(value) ? true : '输入正确编号';
        },
      },
      {
        type: 'rawlist',
        name: 'sort',
        message: '请选择分类: ',
        choices: ['singleFileComponent'],
        validate(value) {
          return value && /^[1]$/.test(value) ? true : '输入正确分类编号';
        },
      },
    ])
    .then((answers) => {
      Object.assign(defaultConfig, answers) && generateComponent();
    });
}
function checkIfFileExists(componentName) {
  return new Promise((resolve, reject) => {
    const filePath = componentsPath + componentName + '.vue';
    if (!fs.existsSync(filePath)) {
      console.log('✅ 文件路径可用，将创建文件');
      resolve(filePath);
    } else {
      reject(`${filePath}文件已经存在`);
    }
  });
}
function generateFile(filePath) {
  return new Promise((resolve, reject) => {
    fs.open(filePath, 'w', (err, file) => {
```

```
        if (err) reject(err);
        console.log('✔ 文件创建成功,正在生成模版内容');
        resolve(filePath);
      });
    });
}
function generateFileContent(filePath) {
    return new Promise((resolve, reject) => {
      fs.writeFile(filePath, singleFileComponentTemplate, (err) => {
        if (err) throw err;
        resolve('✔ 文件模版添加成功');
      });
    });
}
function generateComponent() {
    checkIfFileExists(defaultConfig.componentName)
      .then((filePath) => {
        if (defaultConfig.type == 'component') {
          return generateFile(filePath);
        }
        return;
      })
      .then((filePath) => {
        return generateFileContent(filePath);
      })
      .then(() => {
        console.log('✔ 组件文件创建成功');
        process.exit();
      })
      .catch((err) => {
        console.log(err);
      });
}
```

交互部分使用 inquirer, 大家记得安装一下，

```
npm i -D inquirer
```

感觉代码大家都能懂就不在这里啰嗦了。