link: null
title: 珠峰架构师成长计划
description: webpack.config.js
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=135 sentences=545, words=2753

1.Vue Loader 是什么？

- Vue Loader (https://vue-loader.vuejs.org/zh/) 是一个 webpack 的 loader，它允许你以一种名为单文件组件 (SFCs)的格式撰写 Vue 组件

2.起步

2.1 安装

```
npm install vue --save
npm install  webpack webpack-cli style-loader css-loader html-webpack-plugin vue-loader  vue-template-compiler  webpack-dev-server --save-dev
```

2.2 webpack.config.js

webpack.config.js

```
const { VueLoaderPlugin } = require('vue-loader')
const HtmlWebpackPlugin = require('html-webpack-plugin')
const webpack = require('webpack')
module.exports = {
    mode: 'development',
    devtool: false,
    entry: './src/main.js',
    module: {
        rules: [
            {
                test: /\.vue$/,
                loader: 'vue-loader'
            }
        ]
    },
    plugins: [
        new VueLoaderPlugin(),
        new HtmlWebpackPlugin({
            template: './src/index.html'
        }),
        new webpack.DefinePlugin({
            __VUE_OPTIONS_API__: true,
            __VUE_PROD_DEVTOOLS__: true
        })
    ]
}
```

2.3 main.js

src\main.js

```
import { createApp } from 'vue'
import App from './App.vue'
createApp(App).mount('#app')
```

2.4 App.vue

src\App.vue

```
console.log('App');
```

2.5 index.html

src\index.html

```
    vue-loader
```

3.loader实现

3.1 文件结构

- vue-loader内部主要有三个部分
    - `vue-loader\index.js` 实现了一个普通Loader,负责把SFC的不同区块转化为 `import`语句
    - `vue-loader\pitcher.js`实现pitch Loader,用于拼出完整的行内路径
    - `vue-loader\plugin.js` 负责动态修改webpack配置，注入新的loader到rules规则中

3.2 基础知识

3.2.1 LoaderContext

- Loader Context (https://webpack.docschina.org/api/loaders/#the-loader-context) 表示在 loader 内使用 this 可以访问的一些方法或属性
- this.callback (https://webpack.docschina.org/api/loaders/#thiscallback)可以同步或者异步调用的并返回多个结果的函数

3.2.2 pitching-loader

- pitching-loader (https://webpack.docschina.org/api/loaders#pitching-loader)loader 总是 从右到左被调用。在实际（从右到左）执行 loader 之前，会先 从左到右 调用 `loader` 上的 `pitch` 方法
- loader 可以通过 request 添加或者禁用内联前缀，这将影响到 pitch 和执行的顺序,请看Rule.enforce (https://webpack.docschina.org/configuration/module/#ruleenforce)

3.2.3 Rule.enforce

- Rule.enforce (https://webpack.docschina.org/configuration/module#ruleenforce)可以用于指定 `loader`种类

3.2.4 resource

- Rule.resource (https://webpack.docschina.org/configuration/module/#ruleresource)会匹配 resource
- Rule.resourceQuery (https://webpack.docschina.org/configuration/module/#ruleresourcequery)会匹配资源查询

3.2.5 contextify

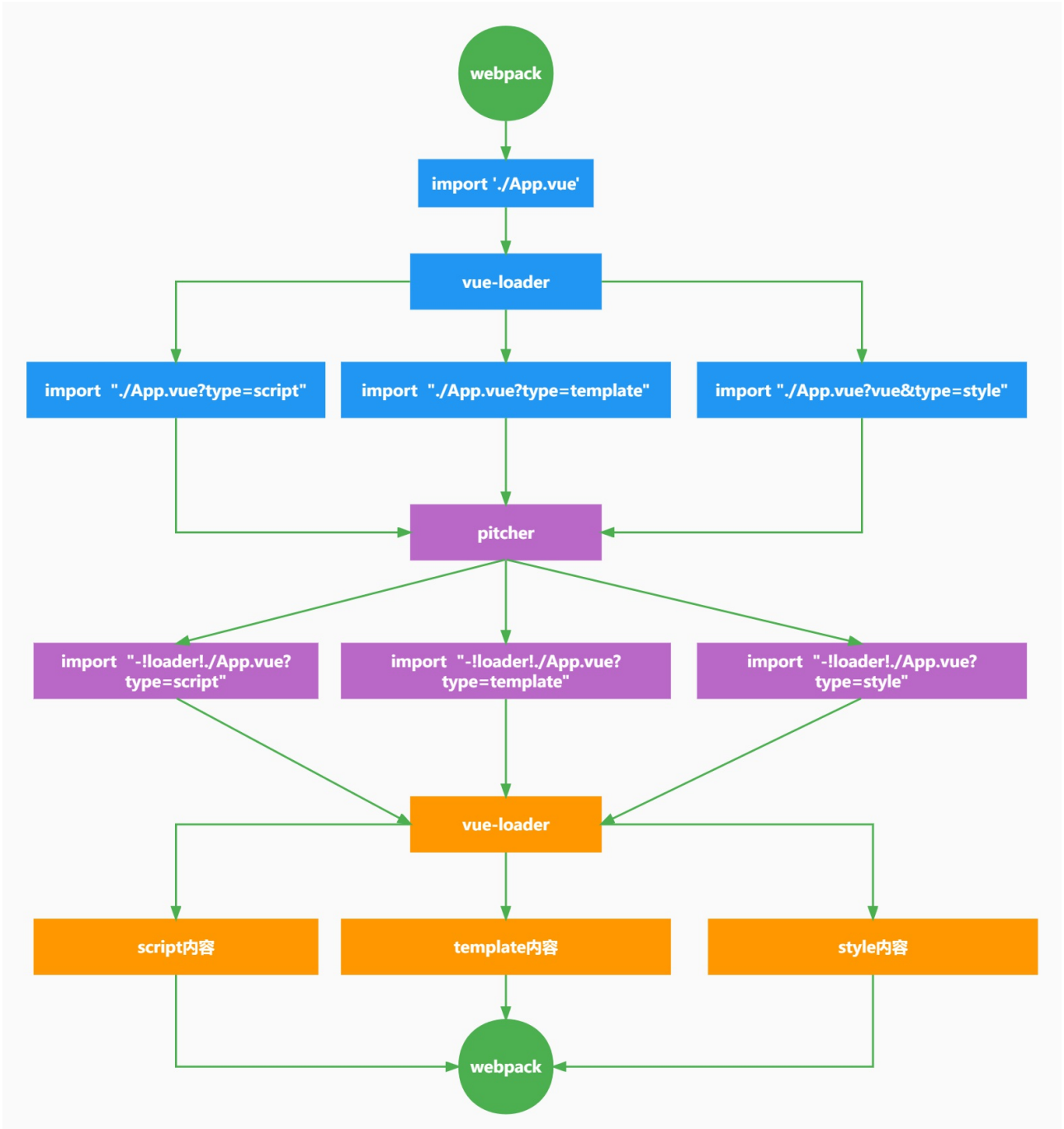
- contextify (https://webpack.docschina.org/api/loaders/#thisutils)返回一个新的请求字符串，尽可能避免使用绝对路径,将请求转换为可在内部使用 `require`或 `import`在避免绝对路径时使用的字符串

3.2.6 @vue/compiler-sfc

- compiler-sfc (https://www.npmjs.com/package/@vue/compiler-sfc)用于编译Vue单文件组件的低级实用程序

```
import script from '/project/foo.vue?vue&type=script'

import { render } from '/project/foo.vue?vue&type=template&id=xxxxxx'

import '/project/foo.vue?vue&type=style&index=0&id=xxxxxx'

script.render = render

script.__file = 'example.vue'
script.__scopeId = 'hash'
export default script
```

3.3 工作流程



3.3.1 原始内容

```
import App from './App.vue'
```

3.3.2 第1次转换

- 1.进入vue-loader的normal处理转换代码为

```
import script from "./App.vue?vue&type=script&id=4d69bc76&lang=js"
import {render} from "./App.vue?vue&type=template&id=4d69bc76&scoped=true&lang=js"
import "./App.vue?vue&type=style&index=0&id=4d69bc76&scoped=true&lang=css"
script.__scopeId = "data-v-4d69bc76"
script.render=render
export default script
```

3.3.3 第2次转换

- 2.进入pitcher，不同区块返回不同内容

```
export { default } from "-!../vue-loader/index.js!./App.vue?vue&type=script&id=4d69bc76&lang=js"; export * from "-!../vue-loader/index.js!./App.vue?
vue&type=script&id=4d69bc76&lang=js"

export * from "-!../vue-loader/templateLoader.js!../vue-loader/index.js!./App.vue?vue&type=template&id=4d69bc76&scoped=true&lang=js"

export * from "-!../node_modules/style-loader/dist/cjs.js!../node_modules/css-loader/dist/cjs.js!../vue-loader/stylePostLoader.js!../vue-
loader/index.js!./App.vue?vue&type=style&index=0&id=4d69bc76&scoped=true&lang=css"
```

3.3.4 第3次转换

- 第二次执行 vue-loader,从SFC中提取对应的区块内容，交给后面的loader
- script内容直接编译返回
- template内容交给 templateLoader
- style内容交给 stylePostLoader

vue-loader\index.js

```
if (incomingQuery.get('type')) {
  return select.selectBlock(descriptor, id, loaderContext, incomingQuery);
}
```

4.编译script

- 第一次的时候只走 vue-loader,返回临时文件 import script from "./App.vue?vue&type=script&id=4d69bc76&lang=js"
- 第一次加载临时文件的时候会走 pitcher, pitcher会拼出行内loader和加载模块的完整路径

4.1 webpack.config.js

webpack.config.js

```
+const { VueLoaderPlugin } = require('./vue-loader')
const HtmlWebpackPlugin = require('html-webpack-plugin')
const webpack = require('webpack')
+const path = require('path')
module.exports = {
    mode: 'development',
    devtool: false,
    entry: './src/main.js',
    module: {
        rules: [
            {
                test: /\.vue$/,
+               loader: path.resolve(__dirname, 'vue-loader')
            }
        ]
    },
    plugins: [
        new VueLoaderPlugin(),
        new HtmlWebpackPlugin({
            template: './src/index.html'
        }),
        new webpack.DefinePlugin({
            __VUE_OPTIONS_API__: true,
            __VUE_PROD_DEVTOOLS__: true
        })
    ]
}
```

4.2 vue-loader\index.js

vue-loader\index.js

```
const compiler = require("vue/compiler-sfc");
const hash = require("hash-sum");
const VueLoaderPlugin = require("./plugin");
const select = require("./select");
function loader(source) {
    const loaderContext = this;
    const { resourcePath, resourceQuery = '' } = loaderContext;
    const rawQuery = resourceQuery.slice(1);
    const incomingQuery = new URLSearchParams(rawQuery);
    const { descriptor } = compiler.parse(source);
    const id = hash(resourcePath);
    if (incomingQuery.get('type')) {
        return select.selectBlock(descriptor, id, loaderContext, incomingQuery);
    }
    const code = [];
    const { script } = descriptor;
    if (script) {
        const query = `?vue&type=script&id=${id}&lang=js`;
        const scriptRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        code.push(`import script from ${scriptRequest}`);
    }
    code.push(`export default script`);
    return code.join('\n');
}
loader.VueLoaderPlugin = VueLoaderPlugin;
module.exports = loader;
```

4.3 plugin.js

vue-loader\plugin.js

```
class VueLoaderPlugin {
    apply(compiler) {
        const rules = compiler.options.module.rules;
        const pitcher = {
            loader: require.resolve('./pitcher'),

            resourceQuery: query => {
                if (!query) {
                    return false;
                }
                let parsed = new URLSearchParams(query.slice(1));
                return parsed.get('vue') !== null;
            }
        };

        compiler.options.module.rules = [pitcher, ...rules];
    }
}
module.exports = VueLoaderPlugin;
```

4.4 pitcher.js

vue-loader\pitcher.js

```
const pitcher = code => code;
const isNotPitcher = loader => loader.path !== __filename;
const pitch = function () {
    const context = this;
    const loaders = context.loaders.filter(isNotPitcher);
    const query = new URLSearchParams(context.resourceQuery.slice(1));
    return genProxyModule(loaders, context, query.get('type') !== 'template');
}
function genProxyModule(loaders, context, exportDefault = true) {
    const request = genRequest(loaders, context);
    return (exportDefault ? `export { default } from ${request}; ` : ``) + `export * from ${request}`;
}
function genRequest(loaders, context) {
    const loaderStrings = loaders.map(loader => loader.request);
    const resource = context.resourcePath + context.resourceQuery;
    return JSON.stringify(context.utils.contextify(context.context, '-!' + [...loaderStrings, resource].join('!')));
}
pitcher.pitch = pitch;
module.exports = pitcher;
```

4.5 select.js

vue-loader\select.js

```
const compiler_sfc = require("vue/compiler-sfc");
function selectBlock(descriptor, scopeId, loaderContext, query) {
    if (query.get('type') === `script`) {
        const script = compiler_sfc.compileScript(descriptor, { id: scopeId });
        loaderContext.callback(null, script.content);
        return;
    }
}
exports.selectBlock = selectBlock;
```

5.编译template

5.1 src\App.vue

src\App.vue

```
    hello

console.log('App');
```

5.2 vue-loader\index.js

vue-loader\index.js

```
const compiler = require("vue/compiler-sfc");
const hash = require("hash-sum");
const VueLoaderPlugin = require("./plugin");
const select = require("./select");
function loader(source) {
    const loaderContext = this;
    const { resourcePath, resourceQuery = '' } = loaderContext;
    const rawQuery = resourceQuery.slice(1);
    const incomingQuery = new URLSearchParams(rawQuery);
    const { descriptor } = compiler.parse(source);
    const id = hash(resourcePath);
    if (incomingQuery.get('type')) {
        return select.selectBlock(descriptor, id, loaderContext, incomingQuery);
    }
    const code = [];
    const { script } = descriptor;
    if (script) {
        const query = `?vue&type=script&id=${id}&lang=js`;
        const scriptRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        console.log(scriptRequest);
        code.push(`import script from ${scriptRequest}`);
    }
+   if (descriptor.template) {
+       const query = `?vue&type=template&id=${id}&lang=js`;
+       const templateRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
+       code.push(`import {render} from ${templateRequest}`);
+   }
+   code.push(`script.render=render`);
    code.push(`export default script`);
    return code.join('\n');
}
loader.VueLoaderPlugin = VueLoaderPlugin;
module.exports = loader;
```

5.3 plugin.js

vue-loader\plugin.js

```
class VueLoaderPlugin {
    apply(compiler) {
        const rules = compiler.options.module.rules;
        const pitcher = {
            loader: require.resolve('./pitcher'),
            resourceQuery: query => {
                if (!query) {
                    return false;
                }
                let parsed = new URLSearchParams(query.slice(1));
                return parsed.get('vue') !== null;
            }
        };
+       const templateCompilerRule = {
+           loader: require.resolve('./templateLoader'),
+           resourceQuery: query => {
+               if (!query) {
+                   return false;
+               }
+               const parsed = new URLSearchParams(query.slice(1));
+               return parsed.get('vue') != null && parsed.get('type') === 'template';
+           }
+       };
+       compiler.options.module.rules = [pitcher, templateCompilerRule, ...rules];
    }
}
module.exports = VueLoaderPlugin;
```

5.4 select.js

vue-loader\select.js

```
const compiler_sfc = require("vue/compiler-sfc");
function selectBlock(descriptor, scopeId, loaderContext, query) {
    if (query.get('type')
        const script = compiler_sfc.compileScript(descriptor, { id: scopeId });
        loaderContext.callback(null, script.content);
        return;
    }
+   if (query.get('type') === `template`) {
+       const template = descriptor.template;
+       loaderContext.callback(null, template.content);
+       return;
+   }
}
exports.selectBlock = selectBlock;
```

5.5 templateLoader.js

vue-loader\templateLoader.js

```
const compiler_sfc = require("vue/compiler-sfc");
const TemplateLoader = function (source) {
    const loaderContext = this;
    const query = new URLSearchParams(loaderContext.resourceQuery.slice(1));
    const scopeId = query.get('id');
    const { code } = compiler_sfc.compileTemplate({
        source,
        id: scopeId
    });
    loaderContext.callback(null, code);
}
module.exports = TemplateLoader;
```

6.编译style

6.1 webpack.config.js

webpack.config.js

```
const { VueLoaderPlugin } = require('./vue-loader')
const HtmlWebpackPlugin = require('html-webpack-plugin')
const webpack = require('webpack')
const path = require('path')
module.exports = {
    mode: 'development',
    devtool: false,
    entry: './src/main.js',
    module: {
        rules: [
            {
                test: /\.vue$/,
                loader: path.resolve(__dirname, 'vue-loader')
            },
+           {
+               test: /\.css$/,
+               use: [
+                   'style-loader',
+                   'css-loader'
+               ]
+           }
        ]
    },
    plugins: [
        new VueLoaderPlugin(),
        new HtmlWebpackPlugin({
            template: './src/index.html'
        }),
        new webpack.DefinePlugin({
            __VUE_OPTIONS_API__: true,
            __VUE_PROD_DEVTOOLS__: true
        })
    ]
}
```

6.2 App.vue

src\App.vue

```

+   hello

console.log('App');

+</span>
<span class="hljs-addition">+.title {</span>
<span class="hljs-addition">+    color: red;</span>
<span class="hljs-addition">+}</span>
<span class="hljs-addition">+
```

6.3 vue-loader\index.js

vue-loader\index.js

```
const compiler = require("vue/compiler-sfc");
const hash = require("hash-sum");
const VueLoaderPlugin = require("./plugin");
const select = require("./select");
function loader(source) {
    const loaderContext = this;
    const { resourcePath, resourceQuery = '' } = loaderContext;
    const rawQuery = resourceQuery.slice(1);
    const incomingQuery = new URLSearchParams(rawQuery);
    const { descriptor } = compiler.parse(source);
    const id = hash(resourcePath);
    if (incomingQuery.get('type')) {
        return select.selectBlock(descriptor, id, loaderContext, incomingQuery);
    }
    const code = [];
    const { script } = descriptor;
    if (script) {
        const query = `?vue&type=script&id=${id}&lang=js`;
        const scriptRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        console.log(scriptRequest);
        code.push(`import script from ${scriptRequest}`);
    }
    if (descriptor.template) {
        const query = `?vue&type=template&id=${id}&lang=js`;
        const templateRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        code.push(`import {render} from ${templateRequest}`);
    }
+   if (descriptor.styles.length) {
+       descriptor.styles.forEach((style, i) => {
+           const query = `?vue&type=style&index=${i}&id=${id}&lang=css`;
+           const styleRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
+           code.push(`import ${styleRequest}`);
+       })
+   }
    code.push(`script.render=render`);
    code.push(`export default script`);
    return code.join('\n');
}
loader.VueLoaderPlugin = VueLoaderPlugin;
module.exports = loader;
```

6.4 plugin.js

vue-loader\plugin.js

```
+const langBlockRuleResource = (query, resource) => `${resource}.${query.get('lang')}`;
class VueLoaderPlugin {
    apply(compiler) {
        const rules = compiler.options.module.rules;
        const pitcher = {
            loader: require.resolve('./pitcher'),
            resourceQuery: query => {
                if (!query) {
                    return false;
                }
                let parsed = new URLSearchParams(query.slice(1));
                return parsed.get('vue') !== null;
            }
        };
+       const vueRule = rules.find(rule => 'foo.vue'.match(rule.test));
+       const clonedRules = rules.filter(rule => rule !== vueRule)
+           .map(rule => cloneRule(rule, langBlockRuleResource));
        const templateCompilerRule = {
            loader: require.resolve('./templateLoader'),
            resourceQuery: query => {
                if (!query) {
                    return false;
                }
                const parsed = new URLSearchParams(query.slice(1));
                return parsed.get('vue') != null && parsed.get('type')
            }
        };
+       compiler.options.module.rules = [pitcher, templateCompilerRule, ...clonedRules, ...rules];
    }
}
+function cloneRule(rule, ruleResource) {
+    let currentResource;
+    const res = Object.assign(Object.assign({}, rule), {
+        resource: resources => {
+            currentResource = resources;
+            return true;
+        },
+        resourceQuery: query => {
+            if (!query) {
+                return false;
+            }
+            const parsed = new URLSearchParams(query.slice(1));
+            if (parsed.get('vue') === null) {
+                return false;
+            }
+            //取出路径中的lang参数，生成一个虚拟路径，传入规则中判断是否满足
+            //通过这种方式，vue-loader可以为不同的区块匹配rule规则
+            const fakeResourcePath = ruleResource(parsed, currentResource);
+            if (!fakeResourcePath.match(rule.test)) {
+                return false;
+            }
+            return true;
+        }
+    });
+    delete res.test;
+    return res;
+}
module.exports = VueLoaderPlugin;
```

6.5 select.js

vue-loader\select.js

```
const compiler_sfc = require("vue/compiler-sfc");
function selectBlock(descriptor, scopeId, loaderContext, query) {
    if (query.get('type')
        const script = compiler_sfc.compileScript(descriptor, { id: scopeId });
        loaderContext.callback(null, script.content);
        return;
    }
    if (query.get('type')
        const template = descriptor.template;
        loaderContext.callback(null, template.content);
        return;
    }
+   if (query.get('type') === `style` && query.get('index') != null) {
+       const style = descriptor.styles[Number(query.get('index'))];
+       loaderContext.callback(null, style.content);
+       return;
+   }
}
exports.selectBlock = selectBlock;
```

7.Scoped CSS

- 当 style标签有scoped (https://vue-loader.vuejs.org/zh/guide/scoped-css.html)属性时，它的 CSS 只作用于当前组件中的元素

7.1 App.vue

src\App.vue

```
   hello

console.log('App');

+</span>
<span class="hljs-addition">+.title {</span>
<span class="hljs-addition">+    color: red;</span>
<span class="hljs-addition">+}</span>
<span class="hljs-addition">+
```

7.2 vue-loader\index.js

vue-loader\index.js

```
const compiler = require("vue/compiler-sfc");
const hash = require("hash-sum");
const VueLoaderPlugin = require("./plugin");
const select = require("./select");
function loader(source) {
    const loaderContext = this;
    const { resourcePath, resourceQuery = '' } = loaderContext;
    const rawQuery = resourceQuery.slice(1);
    const incomingQuery = new URLSearchParams(rawQuery);
    const { descriptor } = compiler.parse(source);
    const id = hash(resourcePath);
    if (incomingQuery.get('type')) {
        return select.selectBlock(descriptor, id, loaderContext, incomingQuery);
    }
+    const hasScoped = descriptor.styles.some(s => s.scoped);
    const code = [];
    const { script } = descriptor;
    if (script) {
        const query = `?vue&type=script&id=${id}&lang=js`;
        const scriptRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        code.push(`import script from ${scriptRequest}`);
    }
    if (descriptor.template) {
+       const scopedQuery = hasScoped ? `&scoped=true` : ``;
+       const query = `?vue&type=template&id=${id}${scopedQuery}&lang=js`;
        const templateRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
        code.push(`import {render} from ${templateRequest}`);
    }
    if (descriptor.styles.length) {
        descriptor.styles.forEach((style, i) => {
+           const scopedQuery = style.scoped ? `&scoped=true` : ``;
+           const query = `?vue&type=style&index=${i}&id=${id}${scopedQuery}&lang=css`;
            const styleRequest = JSON.stringify(loaderContext.utils.contextify(loaderContext.context, resourcePath + query));
            code.push(`import ${styleRequest}`);
        })
    }
+   if (hasScoped) {
+       code.push(`script.__scopeId = "data-v-${id}"`);
+   }
    code.push(`script.render=render`);
    code.push(`export default script`);
    return code.join('\n');
}
loader.VueLoaderPlugin = VueLoaderPlugin;
module.exports = loader;
```

7.3 pitcher.js

vue-loader\pitcher.js

```
+const isCSSLoader = loader => /css-loader/.test(loader.path);
+const stylePostLoaderPath = require.resolve('./stylePostLoader');
const pitcher = code => code;
const isNotPitcher = loader => loader.path !== __filename;
const pitch = function () {
    const context = this;
    const loaders = context.loaders.filter(isNotPitcher);
    const query = new URLSearchParams(context.resourceQuery.slice(1));
+   if (query.get('type') === `style`) {
+       const cssLoaderIndex = loaders.findIndex(isCSSLoader);
+       if (cssLoaderIndex > -1) {
+           const afterLoaders = loaders.slice(0, cssLoaderIndex + 1);
+           const beforeLoaders = loaders.slice(cssLoaderIndex + 1);
+           return genProxyModule([...afterLoaders, stylePostLoaderPath, ...beforeLoaders], context);
+       }
+   }
    return genProxyModule(loaders, context, query.get('type') !== 'template');
}
function genProxyModule(loaders, context, exportDefault = true) {
    const request = genRequest(loaders, context);
    return (exportDefault ? `export { default } from ${request}; ` : ``) + `export * from ${request}`;
}
function genRequest(loaders, context) {
+   const loaderStrings = loaders.map(loader => loader.request || loader);
    const resource = context.resourcePath + context.resourceQuery;
    return JSON.stringify(context.utils.contextify(context.context, '-!' + [...loaderStrings, resource].join('!')));
}
pitcher.pitch = pitch;
module.exports = pitcher;
```

7.4 stylePostLoader.js

vue-loader\stylePostLoader.js

```
const compiler_sfc = require("vue/compiler-sfc");
const StylePostLoader = function (source) {
    const query = new URLSearchParams(this.resourceQuery.slice(1));
    const { code } = compiler_sfc.compileStyle({
        source,
        id: `data-v-${query.get('id')}`,
        scoped: !!query.get('scoped')
    });
    this.callback(null, code);
};
module.exports = StylePostLoader;
```