link: null
title: 珠峰架构师成长计划
description: src\index.js
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=62 sentences=331, words=2345

1. css-loader

```
cnpm i webpack webpack-cli webpack-dev-server html-webpack-plugin css-loader css-selector-tokenizer file-loader less less-loader postcss style-loader to-string-loader -D
```

2. 使用CSS

```
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');
module.exports = {
    mode:'development',
    devtool:false,
    module:{
        rules:[
            {
                test:/\.css$/,
                use:[
                    "to-string-loader",
                    {
                        loader:'css-loader',
                        options:{
                            url:false,
                            import:false,
                            esModule: false
                        }
                    }
                ],
                include:path.resolve('src')
            }
        ]
    },
    plugins:[
        new HtmlWebpackPlugin({template:'./src/index.html'}),
    ]
}
```

src\index.js

```
const css = require("./global.css");
console.log(css);
```

src\global.css

```
body {
    background-color: green;
}
```

dist\main.js

```
(() => {
  var modules = {
    "css-loader/dist/cjs.js!./src/global.css": (module, exports, require) => {
      var ___CSS_LOADER_API_IMPORT___ = require("css-loader/dist/runtime/api.js");
      var ___CSS_LOADER_EXPORT___ = ___CSS_LOADER_API_IMPORT___(function (i) {
        return i[1];
      });
      ___CSS_LOADER_EXPORT___.push([
        module.id,
        "body {\r\n    background-color: green;\r\n  }",
        "",
      ]);
      module.exports = ___CSS_LOADER_EXPORT___;
    },
    "css-loader/dist/runtime/api.js": (module) => {
      module.exports = function () {
        var list = [];
        list.toString = function toString() {
          return this.join("");
        };
        return list;
      };
    },
    "./src/global.css": (module, exports, require) => {
      var result = require("css-loader/dist/cjs.js!./src/global.css");
      if (typeof result === "string") {
        module.exports = result;
      } else {
        module.exports = result.toString();
      }
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  (() => {
    const css = require("./src/global.css");
    console.log(css, css.toString());
  })();
})();
```

```
(() => {
  var modules = {
    "./src/global.css": (module, exports, require) => {
      module.exports = "body {\r\n    background-color: green;\r\n  }";
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  (() => {
    const css = require("./src/global.css");
    console.log(css);
  })();
})();
```

```
(() => {
  var modules = {
    "./src/global.css": (module, exports, require) => {debugger
      var list = [];
      list.push([
        module.id,
        "body {\r\n    background-color: green;\r\n  }",
        ""
      ]);
      let cssWithMappingToString = item=>item[1];
      let css =  list.map(item=>cssWithMappingToString(item)).join("");
      module.exports = css;
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  (() => {
    const css = require("./src/global.css");
    console.log(css);
  })();
})();
```

```
(() => {
  var modules = {
    "css-loader.js!./src/global.css": (module, exports, require) => {
      var api = require("api.js");
      var EXPORT = api(function (item) {
        return item[1];
      });
      EXPORT.push([
        module.id,
        "body {\r\n    background-color: green;\r\n}"
      ]);
      module.exports = EXPORT;
    },
    "api.js": (module) => {
      module.exports = function (cssWithMappingToString) {
        var list = [];
        list.toString = function toString() {
          return this.map(function (item) {
            var content = cssWithMappingToString(item);
            return content;
          }).join("");
        };
        return list;
      };
    },
    "./src/global.css": (module, exports, require) => {
      var result = require("css-loader.js!./src/global.css");
      module.exports = result.toString();
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  (() => {
    const css = require("./src/global.css");
    console.log(css);
  })();
})();
```

3. @import

src\index.js

```
const css = require("./index.less");
console.log(css);
```

src\index.css

```
@import "./global.css";
body {
    color: red;
}
```

src\global.css

```
body {
    background-color: green;
}
```

src\index.less

```
@import "./global.less";
body {
    color: red;
}
```

src\global.less

```
body {
    background-color: green;
}
```

webpack.config.js

```
module.exports = {
+    resolveLoader:{
+        alias:{
+            'css-loader':path.resolve(__dirname,'loaders','css-loader.js')
+        }
+    },
+    module:{
+        rules:[
+            {
+                test:/\.css$/,
+                use:[
+                    "to-string-loader",
+                    {
+                        loader:'css-loader',
+                        options:{
+                            url:false,
+                            import:true,
+                            esModule: false
+                        }
+                    }
+                ],
+                include:path.resolve('src')
+            },
+            {
+                test:/\.less$/,
+                use:[
+                    "to-string-loader",
+                    {
+                        loader:'css-loader',
+                        options:{
+                            import:true,
+                            url:false,
+                            esModule: false,
+                            importLoaders:1
+                        }
+                    },
+                    {
+                        loader:'less-loader'
+                    }
+                ],
+                include:path.resolve('src')
+            },
+        ]
+    }
}
```

dist\main.js

```
(() => {
  var modules = {
    "css-loader.js!./src/global.css": (module, exports, require) => {
      var api = require("api.js");
      var EXPORT = api(function (i) {
        return i[1];
      });
      EXPORT.push([
        module.id,
        "body {\r\n    background-color: green;\r\n}\r\n",
        "",
      ]);
      module.exports = EXPORT;
    },
    "css-loader.js!./src/index.css": (module, exports, require) => {
      var api = require("api.js");
+     var GLOBAL_CSS = require("css-loader.js!./src/global.css");
      var EXPORT = api(function (item) {
        return item[1];
      });
+     EXPORT.i(GLOBAL_CSS);
      EXPORT.push([
        module.id,
        "body {\r\n    color: red;\r\n}",
        "",
      ]);
      module.exports = EXPORT;
    },
    "api.js": (module) => {
      module.exports = function (cssWithMappingToString) {
        var list = [];
        list.toString = function toString() {
          return this.map(function (item) {
            var content = cssWithMappingToString(item);
            return content;
          }).join("");
        };
+       list.i = function (modules) {
+         for (var i = 0; i < modules.length; i++) {
+           list.push(modules[i]);
+         }
+       };
        return list;
      };
    },
    "./src/index.css": (module, exports, require) => {
      var result = require("css-loader.js!./src/index.css");
      module.exports = result.toString();
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  (() => {
    const index = require("./src/index.css");
    console.log(index);
  })();
})();
```

loaders\to-string-loader.js

```
var loaderUtils = require("loader-utils");
module.exports = function() {}
module.exports.pitch = function(remainingRequest) {
    return `
        var result = require(${loaderUtils.stringifyRequest(this, "!!" + remainingRequest)});
        if (result && result.__esModule) {
            result = result.default;
        }
        if (typeof result === "string") {
            module.exports = result;
        } else {
            module.exports = result.toString();
        }
    `;
};
```

loaders\css-loader.js

```
var postcss = require("postcss");
var loaderUtils = require("loader-utils");
var Tokenizer = require("css-selector-tokenizer");
const { getOptions } = require("loader-utils");
const cssLoader = function (inputSource) {
  let loaderOptions = getOptions(this) || {};
  const cssPlugin = (options) => {
    return (root) => {
      root.walkAtRules(/^import$/i, (rule) => {
        rule.remove();
        options.imports.push(rule.params.slice(1, -1));
      });
    };
  };

  let callback = this.async();
  let options = { imports: [] };
  let pipeline = postcss([cssPlugin(options)]);
  pipeline.process(inputSource).then((result) => {
    let { loaders, loaderIndex } = this;
    let {importLoaders=0} = loaderOptions;
    const loadersRequest = loaders
    .slice(
      loaderIndex,
      loaderIndex + 1 + (typeof importLoaders !== "number" ? 0 : importLoaders)
    )
    .map((x) => x.request)
    .join("!");
    let importCss = options.imports
      .map((url) =>`list.push(...require(`+loaderUtils.stringifyRequest(this, `-!${loadersRequest}!${url}`)+`));`).join("\r\n");
    let script = `
      var list = [];
      list.toString = function () {return this.join('')}
      ${importCss}
      list.push(\`${result.css}\`);
      module.exports = list;
    `;
    callback(null, script);
  });
};

module.exports = cssLoader;
```

4.@url

```
module.exports = {
    mode:'development',
    devtool:false,
    resolveLoader:{
        alias:{
            'css-loader':path.resolve(__dirname,'loaders','css-loader.js')
        },
         modules:[path.resolve(__dirname,'loaders'),'node_modules']
    },
    module:{
        rules:[
            {
                test:/\.css$/,
                use:[
                    "to-string-loader",
                    {
                        loader:'css-loader',
                        options:{
                            url:true,
                            import:true,
                            esModule: false
                        }
                    },
                    "logger1-loader",
                    "logger2-loader"
                ],
                include:path.resolve('src')
            },
            {
                test:/\.less$/,
                use:[
                    "to-string-loader",
                    {
                        loader:'css-loader',
                        options:{
                            import:true,
+                            url:true,
                            esModule: false,
                            importLoaders:1
                        }
                    },
                    {
                        loader:'less-loader'
                    }
                ],
                include:path.resolve('src')
            },
+            {
+                test:/\.jpg/,
+                use:[{
+                    loader:'file-loader',
+                    options:{
+                        esModule:false
+                    }
+                }]
+            }
        ]
    },
    plugins:[
        new HtmlWebpackPlugin({template:'./src/index.html'}),
    ]
}
```

src\index.css

```
body {
    color: red;
}
+#root{
+    background-image: url(./images/kf.jpg);
+    width:100px;
+    height:100px;
+}
```

dist\main.js

```
(() => {
  var modules = {
    "css-loader.js!./src/index.css": (module, exports, require) => {
      var api = require("api.js");
      var EXPORT = api(function (i) {
        return i[1];
      });
      EXPORT.push([
        module.id,
+        "body {\r\ncolor: red;\r\n}\r\n#root{\r\nbackground-image: url(" +require("./src/images/kf.jpg") +");\r\nwidth:100px;\r\n    height:100px;\r\n}"
      ]);
      module.exports = EXPORT;
    },
    "api.js": (module) => {
      module.exports = function (cssWithMappingToString) {
        var list = [];
        list.toString = function toString() {
          return this.map(function (item) {
            var content = cssWithMappingToString(item);
            return content;
          }).join("");
        };
        list.i = function (modules) {
          for (var i = 0; i < modules.length; i++) {
            list.push(modules[i]);
          }
        };
        return list;
      };
    },
    "./src/images/kf.jpg": (module, exports, require) => {
      module.exports = require.p + "4a783413fe7beffd284711d5fdfd1549.jpg";
    },
    "./src/index.css": (module, exports, require) => {
      var result = require("css-loader.js!./src/index.css");
      module.exports = result.toString();
    },
  };
  var cache = {};
  function require(moduleId) {
    if (cache[moduleId]) {
      return cache[moduleId].exports;
    }
    var module = (cache[moduleId] = {
      id: moduleId,
      exports: {},
    });
    modules[moduleId](module, module.exports, require);
    return module.exports;
  }
  require.p="";
  const index = require("./src/index.css");
  console.log(index);
})();
```

loaders\css-loader.js

```
var postcss = require("postcss");
var loaderUtils = require("loader-utils");
var Tokenizer = require("css-selector-tokenizer");
const { getOptions } = require("loader-utils");
const cssLoader = function (inputSource) {
  let loaderOptions = getOptions(this) || {};
  const cssPlugin = (options) => {
    return (root) => {
      root.walkAtRules(/^import$/i, (rule) => {
        rule.remove();
        options.imports.push(rule.params.slice(1, -1));
      });
+      root.walkDecls((decl) => {
+          var values = Tokenizer.parseValues(decl.value);
+          values.nodes.forEach(function (value) {
+            value.nodes.forEach((item) => {
+              if (item.type === "url") {
+               let url = loaderUtils.stringifyRequest(this, item.url);
+               item.stringType = "'";
+               item.url ="`+require("+url+")+`";
+              }
+            });
+          });
+          let value = Tokenizer.stringifyValues(values);
+          decl.value = value;
+      });
    };
  };

  let callback = this.async();
  let options = { imports: [] };
  let pipeline = postcss([cssPlugin(options)]);
  pipeline.process(inputSource).then((result) => {
    let { loaders, loaderIndex } = this;
    let {importLoaders=0} = loaderOptions;
    const loadersRequest = loaders
    .slice(
      loaderIndex,
      loaderIndex + 1 + (typeof importLoaders !== "number" ? 0 : importLoaders)
    )
    .map((x) => x.request)
    .join("!");
    let importCss = options.imports
      .map((url) =>`list.push(...require(`+loaderUtils.stringifyRequest(this, `-!${loadersRequest}!${url}`)+`));`).join("\r\n");
    let script = `
      var list = [];
      list.toString = function () {return this.join('')}
      ${importCss}
      list.push(\`${result.css}\`);
      module.exports = list;
    `;
    callback(null, script);
  });
};

module.exports = cssLoader;
```

5.style-loader

```
const path = require('path');
const HtmlWebpackPlugin = require('html-webpack-plugin');
module.exports = {
    mode:'development',
    devtool:false,
    resolveLoader:{
        alias:{
            'css-loader':path.resolve(__dirname,'loaders','css-loader.js'),
+            'style-loader':path.resolve(__dirname,'loaders','style-loader.js')
        }
    },
    module:{
        rules:[
            {
                test:/\.css$/,
                use:[
+                    "style-loader",
                    {
                        loader:'css-loader',
                        options:{
                            url:true,
                            import:true,
                            esModule: false
                        }
                    }
                ],
                include:path.resolve('src')
            },
            {
                test:/\.less$/,
                use:[
+                    "style-loader",
                    {
                        loader:'css-loader',
                        options:{
                            import:true,
                            url:true,
                            esModule: false,
                            importLoaders:1
                        }
                    },
                    {
                        loader:'less-loader'
                    }
                ],
                include:path.resolve('src')
            },
            {
                 test:/\.jpg/,
                 use:[{
                     loader:'file-loader',
                     options:{
                         esModule:false
                     }
                 }]
             }
        ]
    },
    plugins:[
        new HtmlWebpackPlugin({template:'./src/index.html'}),
    ]
}
```

loaders\style-loader.js

```
let loaderUtils = require("loader-utils");
function loader(source) {}

loader.pitch = function (remainingRequest, previousRequest, data) {let style = `
    var style = document.createElement("style");
    style.innerHTML = require(${loaderUtils.stringifyRequest(
      this,
      "!!" + remainingRequest
    )});
    document.head.appendChild(style);
 `;
  return style;
};
module.exports = loader;
```

src\index.html

```
    css-loader

+    root
```

7. PostCSS

- CSS AST主要有3种父类型
    - `AtRule` @xxx的这种类型，如@screen
    - `Comment` 注释
    - `Rule` 普通的css规则
- 子类型
    - `decl` 指的是每条具体的css规则
    - `rule` 作用于某个选择器上的css规则集合
- nodes: CSS规则的节点信息集合
    - decl: 每条css规则的节点信息
    - prop: 样式名,如width

    - value: 样式值,如10px
- type: 类型
- source: 包括start和end的位置信息，start和end里都有line和column表示行和列
- selector: type为rule时的选择器
- name: type为atRule时@紧接rule名，譬如@import 'xxx.css'中的import
- params: type为atRule时@紧接rule名后的值，譬如@import 'xxx.css'中的xxx.css
- text: type为comment时的注释内容
- walk: 遍历所有节点信息，无论是atRule、rule、comment的父类型，还是rule、 decl的子类型
    - walkAtRules：遍历所有的AtRules
    - walkComments 遍历所有的Comments
    - walkDecls 遍历所有的Decls
    - walkRules 遍历所有的Rules
- postCss给出了很多操作CSS ([AtRule](http://api.postcss.org/AtRule.html))的方法
    - postcss插件如同babel插件一样，有固定的格式
    - 注册个插件名，并获取插件配置参数 `opts`
    - 返回值是个函数，这个函数主体是你的处理逻辑,第一个参数是AST的根节点

```
var postcss = require("postcss");
const cssPlugin = (options) => {
    return (root,result) => {
      root.walkAtRules();
      root.walkDecls((decl) => {
        if(decl.value.endsWith('px')){
            decl.value=parseFloat(decl.value)/75+'rem';
        }
      });
    };
  };
let options = {};
let pipeline = postcss([cssPlugin(options)]);
let inputSource = `
#root{
    width:750px;
}`;
pipeline.process(inputSource).then((result) => {
    console.log(result.css);
})
```