POSTCSS

原理与使用

刘洋鑫

CONTENTS

1.St

为什么要使用postcss

- css预处理器的不足
- postcss的优势

CSS预处理器的不足

Libsass: 110 files, 21 300 LOC of C++
Stylus: 72 files, 7 900 LOC
Less: 105 files, 9 800 LOC

postcss的优势

处理速度快

CSS Preprocessor Benchmark

处理200 KB的简单语法 CSS 文件用

- Sass: 4.9 sec
- Sass: 2.5 sec (with warm `.sass-cache`)
- libsass: 0.2 sec
- Stylus: 1.7 sec
- Rework: 0.2 sec
- LESS: 0.5 sec
- r.js: 0.2 sec

PostCSS Benchmarks

Css处理器在解析使用嵌套，混合，变量等语法时用时（代码量小于100行）

```
PostCSS:    39 ms
Rework:     68 ms   (1.8 times slower)
libsass:    100 ms  (2.6 times slower)
Less:       134 ms  (3.5 times slower)
Stylus:     171 ms  (4.4 times slower)
Stylecow:   296 ms  (7.6 times slower)
Ruby Sass: 1692 ms (43.6 times slower)
```

postcss的优势

工具而非模版语言，体积小

做到sass做不到的事

Css 选择器 level4

功能扩展

可以对功能进行私人定制

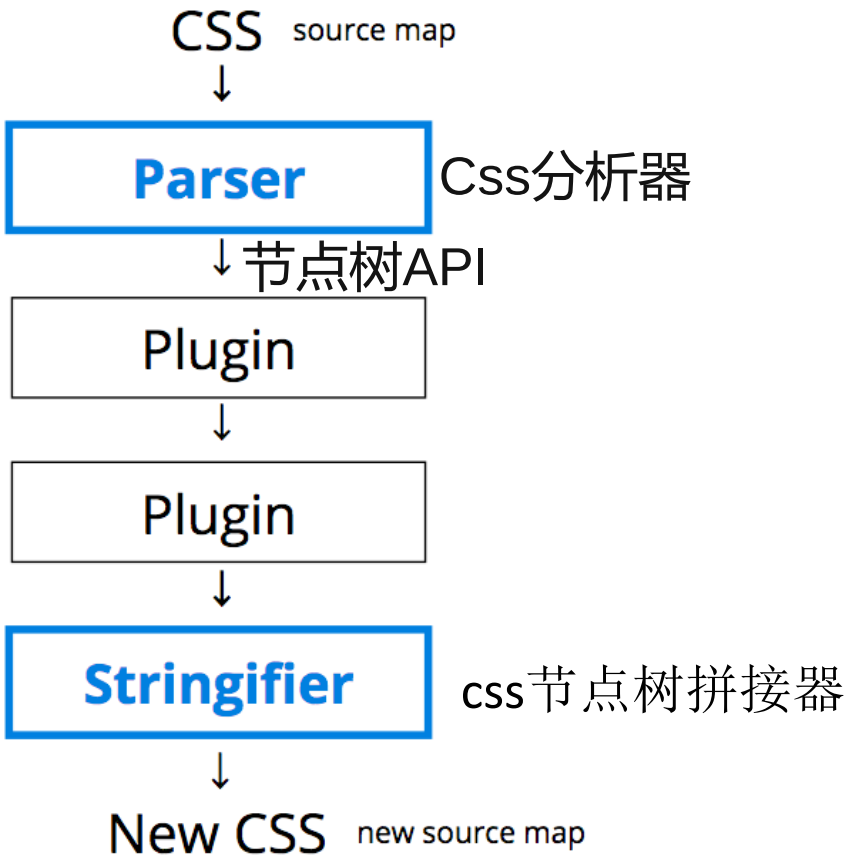
无需学习新的预处理器

并不是一种新的模版语言

2.nd

Postcss 的原理

原理



css分析器读取样式规则，得到节点树；

插件通过节点树的API进行转换，

最后由节点树拼接器将节点树重新组成css字符

Css 分析器结果

CSS 文件在经过分析器转化后可以得到json对象即抽象语法树 (AST)：

```
{
  "type": "root",
  "nodes": [
    {
      "type": "rule",
      "selector": ".progress-bar",
      "nodes": [
        {
          "type": "decl",
          "prop": "width",
          "value": "calc(random * 100%)"
        }
      ]
    }
  ]
}
```

root: 整个CSS 代码段

nodes: 节点

rule:一个CSS class 范围内的代码段

decl: 单行CSS ,包含属性与值

```
.icon-close {
    background-image: url(../slice/icon-close.png);
    font-size: 14px;
}
```

Width: calc(random * 100)

prop: CSS 属性　value: 值

3.rd

自己编写插件

利用postcss的API编写插件

插件最基础的构成

```
var postcss = require('postcss');
module.exports = postcss.plugin('PLUGIN_NAME', function () {});
```

```
css.walkRules(function (rule) {          // 遍历所有 CSS
        rule.walkDeclsfunction (decl) {      // 遍历每条 CSS 规则
                rule.append({…});            // 添加规则
}});
```

编写插件：字体组简写

目标：　　font-family: "Open Sans", fontstack("Arial");

font-family: "Open Sans", Arial, "Helvetica Neue", Helvetica, sans-serif;

1、定义字体组

```
var postcss = require('postcss');

// 定义字体组   Font stacks from http://www.cssfontstack.com/
var fontstacks_config = {
    'Arial': 'Arial, "Helvetica Neue", Helvetica, sans-serif',
    'Times New Roman': 'TimesNewRoman, "Times New Roman", Times, Baskerville, Georgia, serif'
}
```

编写插件：字体组简写

2、获取样式。"Open Sans", fontstack("Arial");

```
css.walkRules(function (rule) {
    rule.walkDecls(function (decl, i) {
        var value = decl.value;
```

3、获取字体组简写值 Arial

```
if (value.indexOf('fontstack(') !== -1) {
    // 通过匹配fontstack()括号中的字符串，得到需要请求的字体组，然后替换其中的双引号或单引号
    var fontstack_requested = value.match(/\(([^)]+)\)/)[1].replace(/["']/g, "");
    // 首字母大写
    function toTitleCase(str) {
        return str.replace(/\w\S*/g, function (txt) {
            return txt.charAt(0).toUpperCase() + txt.substr(1).toLowerCase(); });
    }
    fontstack_requested = toTitleCase(fontstack_requested);
```

编写插件：字体组简写

4、查找“Arial”对应的字体并与fontstack()之前的字体名"Open Sans"合并

```
var fontstack = fontstacks_config[fontstack_requested];
// 查找并存储fontstack()之前的任意字体名
var first_font = value.substr(0, value.indexOf('fontstack('));
// 通过合并first_font和fontstack变量创建一个新值
var new_value = first_font + fontstack;
```

5、将新值赋给样式表

```
// 将新值返回样式表中
decl.value = new_value;
```

4.th

推荐插件及写法

推荐插件

postcss-sassy-mixins

```
@mixin images($img) {
        background-image: url('./../assets/images/$img.jpg');
}
@include images(luck);
```

postcss-extend

```
%fixed  {
    position:  fixed;
}
.Toolbar {
    @extend %fixed;
}
```

推荐插件

cssnano

压缩优化

- 删除空格和最后一个分号
- 删除注释
- Calc 计算

postcss-assets

未更新的插件，版本依赖postcss5.2.17

- 省略路径 background: resolve('bg1.png');
- 内联图片，把图片转换成 Base64 background: inline(‘img.jpg');
- 取图片宽高和尺寸，指定除数

```
body {
  width: width('images/foobar.png', 2); /* 160px */
  height: height('images/foobar.png', 2); /* 120px */
  background-size: size('images/foobar.png', 2); /* 160px 120px */
}
```

推荐插件

1. 包含AutoPrefixer自动添加浏览器前缀

2. Css 层级嵌套　　& .title

3. CSS Selectors Level 4
:not
:matches 语法糖
:valid、:invalid <input type="email">
:placeholder-shown

4. 自定义选择器

@custom-selector :--heading h1, h2, h3, h4, h5, h6;

5. 变量

```
:root { --mainColor: red; }
a { color: var(--mainColor); }
```

推荐插件

1. 限制嵌套层级 max-nesting-depth
2. 最大空行数 max-empty-lines
3. 选择器类型检查

```
"selector-type-no-unknown": [ true, {
        "ignoreTypes": ["/^--/"],
}],
```

4. 选择器大小写检查 selector-type-case
5. 注释前空行 comment-empty-line-before

赋null关闭选项

推荐插件

postcss-reporter

美化插件的打印信息

postcss-import

@import引入外部css文件必须

推荐插件

postcss-triangle

画三角形

```
triangle: pointing-right;
width: 150px;
height: 115px;
background-color: red;
```

插件顺序

postcss-sassy-mixins

stylelint

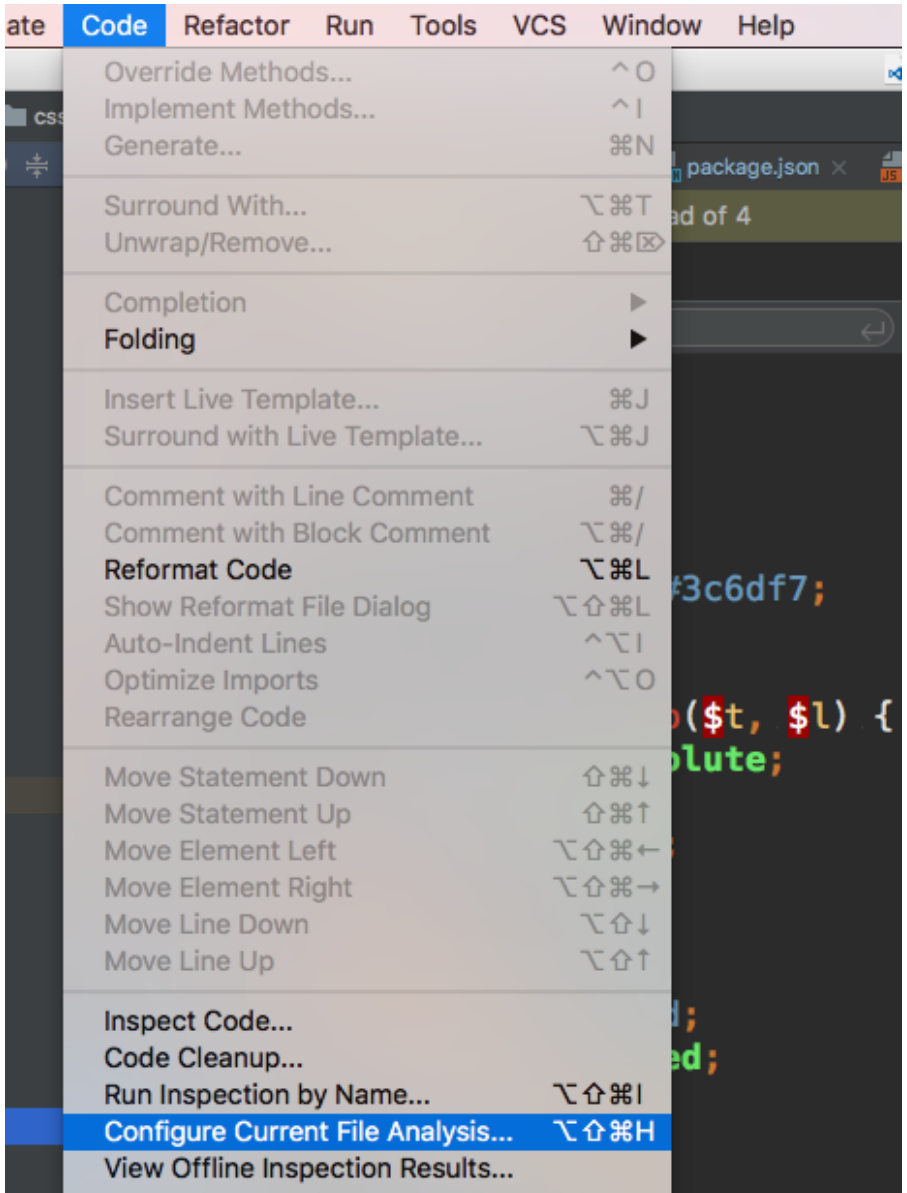
postcss-cssnext

postcss-extend

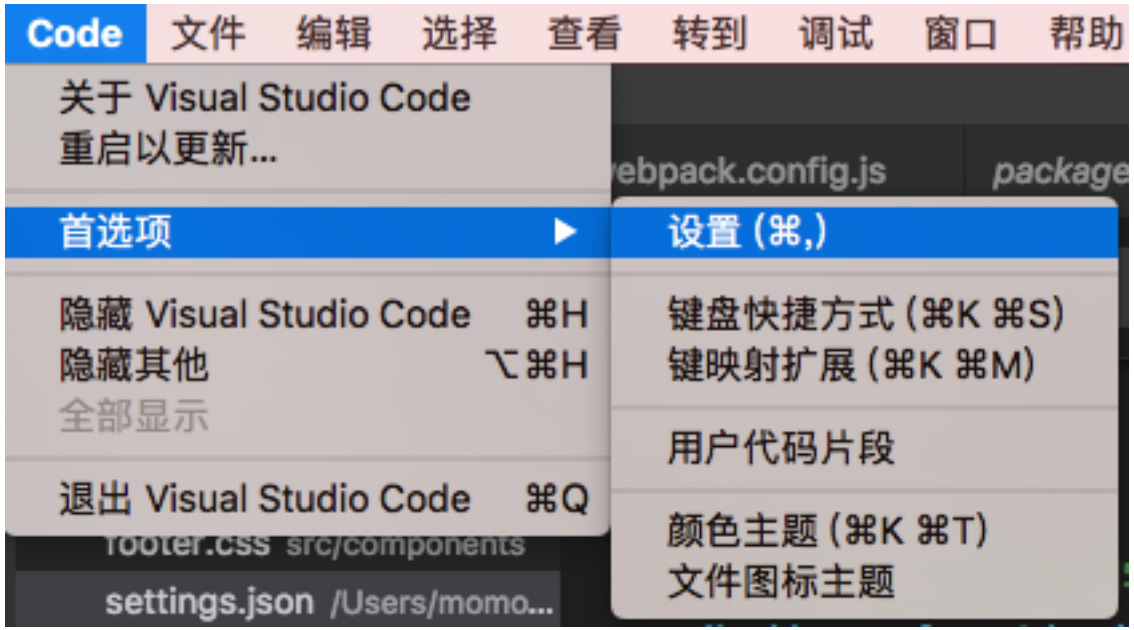
cssnano

关闭编辑器检查

phpstorm



vscode



样式模块化

cssModules

- 与postcss并列
- style标签中写 module<style module>
- 模版中由$style注入类名 :class="$style.navItem”
- “-”改写 $style.normalPopWrapper”
- 不支持多类名

THANKS

Thank you for watching