PostCSS

宁勃

“ PostCSS 是一个用 JavaScript 工具和插件转换 CSS 代码的工具。”

目录

PostCSS 是什么

- PostCSS 是处理 CSS 的一个插件体系
- 可以对 CSS 进行各种不同的转换和处理
- 把繁琐复杂的工作交由程序去处理
- 把开发人员解放出来

PostCSS 如何使用

- PostCSS 一般不单独使用，**而是与已有的构建工具进行集成**
- PostCSS 与主流的构建工具，如 Webpack、Grunt 和 Gulp 都可以进行集成
- 完成集成之后，选择满足功能需求的 PostCSS 插件并进行配置。

PostCSS 有哪些插件

- 插件查询地址：https://www.postcss.parts/
- 常用插件列表：https://github.com/postcss/postcss/blob/master/docs/plugins.md

PostCSS 插件列举

- Autoprefixer：自动补全浏览器私有前缀
- precss：CSS预处理（整合Sass、LESS或Stylus功能，语法基本和Sass的相同）
- postcss-import：通过@import，整合多个CSS文件
- css-mqpacker：将相同的CSS媒体查询规则合并为一个
- cssnano：压缩CSS文件
- postcss-color-rgba-fallback：给rgba颜色创建降级方案(添加备用颜色)
- postcss-opacity：给opacity提供降级方案（给IE浏览器添加滤镜属性）
- node-pixrem：让IE8支持rem单位
- postcss-pseudoelements：将伪元素的::转换为:(IE8不支持::)

PostCSS 语法介绍

- Autoprefixer：自动补全浏览器私有前缀
- postcss-preset-env：支持现代的 css 语法
- precss：CSS预处理（整合Sass、LESS或Stylus功能，语法基本和Sass的相同）
- cssnano：压缩CSS文件

Autoprefixer 插件介绍

Autoprefixer：自动补全浏览器私有前缀，参考CanIUse

```
p{
    transform: scale(1);
    animation: ani 1s linear;
}
```

```
p{
    -webkit-transform:scale(1);
            transform:scale(1);
    -webkit-animation:ani 1s linear;
            animation:ani 1s linear;
}
```

postcss-preset-env 插件介绍

postcss-preset-env：支持现代的 css 语法

- 重置标签所有属性
- 统一锚点各状态的样式
- 自定义媒体查询
- 自定义常量
- 自定义选择器
- 支持嵌套规则
- overflow 简写

postcss-preset-env 插件介绍

```css
a{
    all: initial;
}

a:any-link{
    background-color: yellow;
}

@custom-media --narrow-window (max-width: 30em);
@media (--narrow-window) { }

:root{
    --some-length: 32px;
}
p {
    height: var(--some-length);
    width: var(--some-length);
}

@custom-selector :--heading h1, h2, h3, h4, h5, h6;
:--heading {
    margin-block: 0;
}

html {
    overflow: hidden auto;
}
```

```css
a{
    -webkit-animation:none 0s ease 0s 1 normal none running;
    -webkit-backface-visibility:visible;
    -o-border-image:none;
    ……
}

a:-webkit-any-link{
    background-color:#ff0;
}

a:-moz-any-link{
    background-color:#ff0;
}

a:link,a:visited{
    background-color:#ff0;
}

:root{
    --some-length:32px;
}
p{
    height:32px;
    height:var(--some-length);
    width:32px;
    width:var(--some-length);
}
h1,h2,h3,h4,h5,h6{
    margin-top:0;
    margin-bottom:0;
}


html{
    overflow-x:hidden;
    overflow-y:auto;
    overflow:hidden auto;
}
```

precss 插件介绍

支持类似sass语法，并支持未来语法

- 嵌套可以使用 & 符，把父选择器复制到子选择器中
- 使用 $ 符声明变量，比如 $color: #ccc
- 用 @if 和 @else 来控制循环
- 用 @for 和 @each 来循环
    - @for 循环：用一个计数器变量，来设置循环的周期
    - @each 循环：循环周期是一个列表，而不是数字
- mixin
    - 通过 @mixin mixin_name($arg1, $arg2) {...} 来声明
    - 使用 @include mixin_name($arg1, $arg2) 来调用
- @extend 创建模板
- @at-root 生成到根部
- 直接引用css属性的值，例如@color

precss 插件介绍

```
article {
    margin-top: 20px;
    & p {
        color: #333;
    }
}
```

```
$text_color: #232323;
$border_comn: 1px solid red;
body{
    color: $text_color;
    border: $border_comn;
}
```

```
$column_layout: 2;
.column{
    @if $column_layout == 2{
        width: 50%;
        float: left;
    }@else{
        width: 100%;
    }
}
```

```
article{
    margin-top:20px
}
article p{
        color:#333;
    }
body{
    color:#232323;
    border:1px solid red;
}
.column{
        width:50%;
        float:left
}
```

precss 插件介绍

```
@for $i from 1 to 3{
    p:nth-of-type($i){
        margin-left: calc(100% / $i);
    }
}
```

```
@for $i from 1 to 5{
    p:nth-of-type($i){
        margin-left: calc(100% / $i);
    }
}
```

```
@for $count from 1 to 5 by 2 {
    .col-$count {
        width: $count%;
    }
}
```

```
$social: twitter,facebook,youtube;
@each $icon in ($social){
    .icon-$(icon){
        background: url('img/$(icon).png');
    }
}
```

```
p:first-of-type{
        margin-left:100%;
    }
p:nth-of-type(2){
        margin-left:50%;
    }
p:nth-of-type(3){
        margin-left:33.33333%;
    }
p:nth-of-type(4){
        margin-left:25%;
    }
p:nth-of-type(5){
        margin-left:20%;
    }
.col-1{
        width:1%;
    }
.col-3{
        width:3%;
    }
.col-5{
        width:5%;
    }

.icon-twitter{
        background:url(img/twitter.png);
    }

.icon-facebook{
        background:url(img/facebook.png);
    }

.icon-youtube{
        background:url(img/youtube.png);
    }
```

precss 插件介绍

```scss
@mixin heading-text($color: #242424, $font-size: 4em) {
  color: $color;
  font-size: $font-size;
}

h1, h2, h3 {
  @include heading-text;
}

.some-heading-component > :first-child {
  @include heading-text(#111111, 6em);
}


%thick-border {
  border: thick dotted red;
}
.modal {
  @extend %thick-border;
  color: red;
}


.parent {
    font-size: 20px;
  @at-root{
    .child {
      font-size: 25px;
    }
  }
}


.Test {
  margin-left: 20px;
  margin-right: @margin-left;
  color: red;
  background: @color url('test.png');
  line-height: 1.5;
  font-size: @(line-height)em;
}
```

```css
h1,h2,h3{
  color:#242424;
  font-size:4em;
}

.some-heading-component>:first-child{
  color:#111;
  font-size:6em;
}
.modal{border:thick dotted red;color:red;}


.child{
      font-size:25px;
    }


.parent{
    font-size:20px
}


.Test{
  margin-left:20px;
  margin-right:20px;
  color:red;
  background:red url(test.png);
  line-height:1.5;
  font-size:1.5em;
}
```

自定义插件

一个PostCSS 插件最基础的构成如下：

```
var postcss = require('postcss');
module.exports = postcss.plugin('PLUGIN_NAME', function (opts) {
    opts = opts || {};
    // 传入配置相关的代码
    return function (root, result) {
        // 转化CSS 的功能代码
    };
});
```

然后就是不同的需求情况来决定是否引入第三方模块，是否有额外配置项，然后在包含root,result 的匿名函数中进行最为核心的转换代码功能编写。

- root(css)： 也是整个 CSS 代码段，包含多个 rule
- rule: 包含一个 CSS class 范围内的代码段
- nodes: 代指 rule 中 {} 中间的多个 decl 部分。
- decl: 单行 CSS ,即有属性与值的部分
- prop ， value

“感谢聆听”