

Introdução a Customização de Arquivos CSS com o Pré-Processador SASS

APRESENTAR UMA OUTRA FORMA DE COMO TRABALHAR COM ARQUIVOS CSS PARA CRIAÇÃO DE FOLHAS DE ESTILO PARA SEREM UTILIZADAS EM PÁGINAS NA INTERNET.

AUTOR(A): PROF. EDSON MELO DE SOUZA

1. Introdução

O desenvolvimento de aplicações Web é centrada na utilização de tecnologias que permitem a manipulação de elementos a serem apresentados ao usuário, ou seja, qualquer tecnologia utilizada no *Back-End* (processamento do lado do servidor) é repassada ao usuário no formato HTML, assim como as demais tecnologias que podem ser agregadas para melhorar a interface e o desempenho de um site.

O desenvolvimento de páginas Web é de certa forma simples, entretanto a complexidade das aplicações demanda que o esforço para sua realização torna a tarefa complexa, além da necessidade de contornar problemas de compatibilidade com navegadores.

Neste tópico são lembradas as tecnologias HTML e o CSS e apresentado o SASS (*Syntactically Awesome StyleSheets*), que é um pré-processador de estilos CSS.

2. HTML 5

A linguagem HTML - *Hypertext Markup Language* ou Linguagem de Marcação de Hipertexto é utilizada para a publicação de conteúdo na Web (texto, imagem, vídeo, áudio, entre outros).

Esta linguagem possui elementos que permitem a manipulação de conteúdos por meio da utilização de *tags* que são instruções reduzidas ou marcas (W3C, 2017), que permitem ao desenvolvedor maior flexibilidade no desenvolvimento de páginas.

Um documento HTML é formado por etiquetas ou *tags*, as quais são os comandos da linguagem e que têm a função de marcar e formatar o texto para exibição no navegador (HARRIS, 2009).

O conjunto de tags disponíveis no HTML, atualmente na versão 5, é bastante extenso, o que permite desde a criação de simples parágrafos, até a construção de tabelas, formulários, animações, entre outras. Apesar de possuir um grande conjunto de etiquetas ou comandos, cada qual com uma função específica, a sintaxe do HTML é bastante simples e de fácil aprendizado.

O HTML é padronizado e utilizado apenas para a apresentação de páginas na Web, entretanto o mesmo fornece apenas conteúdo estático (LAWSON; SHARP, 2010). Uma página bem estruturada e com alguma funcionalidade, além da apresentação de textos, emprega, na grande maioria, tabelas e formulários, pois este último é a forma de enviar dados coletados em uma página para processamento no servidor.

O uso de formulários é amplamente empregado, pois eles são capazes de receber dados, os quais são submetidos ao servidor para o processamento. Existe uma grande variedade de formatos de formulários, embora os elementos que possibilitam sua construção são sempre os mesmos, sendo alterado apenas o layout em cada caso. Na caixa em destaque a seguir você pode obter uma lista completa das *tags* do HTML5.

Um outro aspecto, não menos importante, é a formatação de um documento HTML, pois as *tags* são apenas instrumentos de marcação, o que faz com que a aparência seja muito ruim. E, por esse motivo, entre então o CSS, que veremos a seguir.

3. CSS 3

CSS pode ser definido como “Uma linguagem que descreve um estilo para um documento HTML, ou seja, como um elemento deve ser apresentado no navegador.”.

A definição de estilos em documentos HTML é amplamente utilizada, pois elimina diversos elementos das páginas como: imagens e códigos embutidos. Além dessas características, a utilização do CSS facilita a manutenção das páginas por centralizar, em um ou mais arquivos, as definições a serem empregadas nas páginas.

O CSS 3 trouxe diversas melhorias em relação a versão 2, principalmente no que diz respeito à animação de elementos 2D e 3D, onde os mais utilizados são os de movimento, rotação e transição. Alguns elementos também dispensam a utilização de Javascript, como a definição de barras de rolagem com a tag `overflow-x` e `overflow-y`.

A criação de um estilo em CSS pode ir do mais simples até o mais complexo, dependendo da necessidade da aplicação do recurso no desenvolvimento de uma página HTML.

No exemplo a seguir é mostrado um código CSS que realiza uma formatação simples, ou seja, transforma um simples parágrafo com a tag “p” em um parágrafo com cor de fundo, dando destaque ao texto.

```
1. <!DOCTYPE html>
2. <html lang="pt-br">
3.     <head>
4.         <title></title>
5.
6.         <style type="text/css">
7.             p {
8.                 background-color: yellow;
9.                 font-size: 24px;
10.                padding: 50px;
11.            }
12.        </style>
13.    </head>
14.    <body>
15.        <p>Eu tenho o fundo amarelo</p>
16.    </body>
17. </html>
```

No código anterior o que realmente nos importa é o conteúdo que está dentro das *tags style*, é ali que estão os códigos que farão a formatação no HTML criado.

Na imagem a seguir é visualizado o resultado após ser renderizado no navegador, mostrando a frase “Eu tenho o fundo amarelo”.



Vejamos então apenas o fragmento do código CSS.

```
1. p {
2.     background-color: yellow;
3.     font-size: 24px;
4.     padding: 50px;
5. }
```

Na linha 1 é declarada uma *tag* “p” e um par de chaves “{}”. Dentro das chaves, linhas de 2 a 4, são declaradas propriedades do CSS, utilizada para criar a formatação no documento.

Neste caso a linha 2 está dizendo que é para ser atribuída a cor amarela ao fundo do parágrafo, pois tudo que está dentro das chaves, faz referência ao elemento que está fora dela, nesse caso o “p” de parágrafo. Na linha 3 é configurado o tamanho da fonte para 24px e, finalmente na linha 4, o código diz que haverá uma margem interna de 50px de cada lado.

Até aqui sem novidades. Mas, e quando temos um arquivo CSS extremamente grande, com muitas classes e id's para controlarmos? Pois bem, é aqui que entra o SASS (*Syntactically Awesome StyleSheets*) que veremos agora.

SAIBA MAIS!

Mozilla Foundation - https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5_element_list
(https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5/HTML5_element_list)

O SASS (*Syntactically Awesome StyleSheets*)



Syntatically Awesome Style Sheets

O SASS é um pré-processador de arquivos CSS que facilita a vida do desenvolvedor quando há a necessidade de escrever diferentes estilos dentro de um mesmo projeto, ou até mesmo para outros projetos.

Trabalhar com o CSS é fácil e intuitivo, entretanto quanto mais vamos aprimorando as interfaces, maior é número de repetições que temos que realizar dentro dos arquivos CSS, pois muitos elementos devem ser utilizados em lugares diferentes.

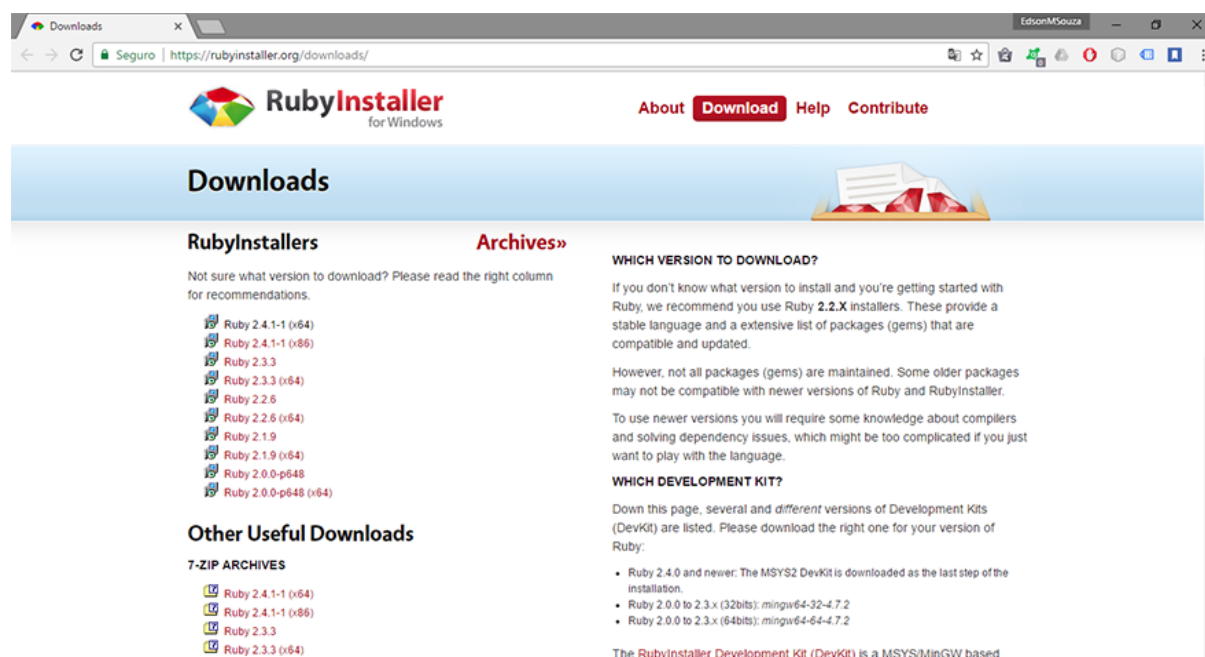
Uma das vantagens em utilizar o SASS é o ganho de produtividade, pois, quando necessitamos trocar alguma configuração em um ou mais arquivos CSS, temos que fazê-lo um a um, mas como o SASS, podemos fazer tudo de uma só vez. Ao final de uma edição, basta gerarmos então o arquivo CSS final que será interpretado pelo HTML.

O SASS utiliza conceitos de linguagens de programação, onde é possível criar variáveis, análise de expressões matemáticas (para dimensionamento de telas), decisões baseadas em opções, entre outros.

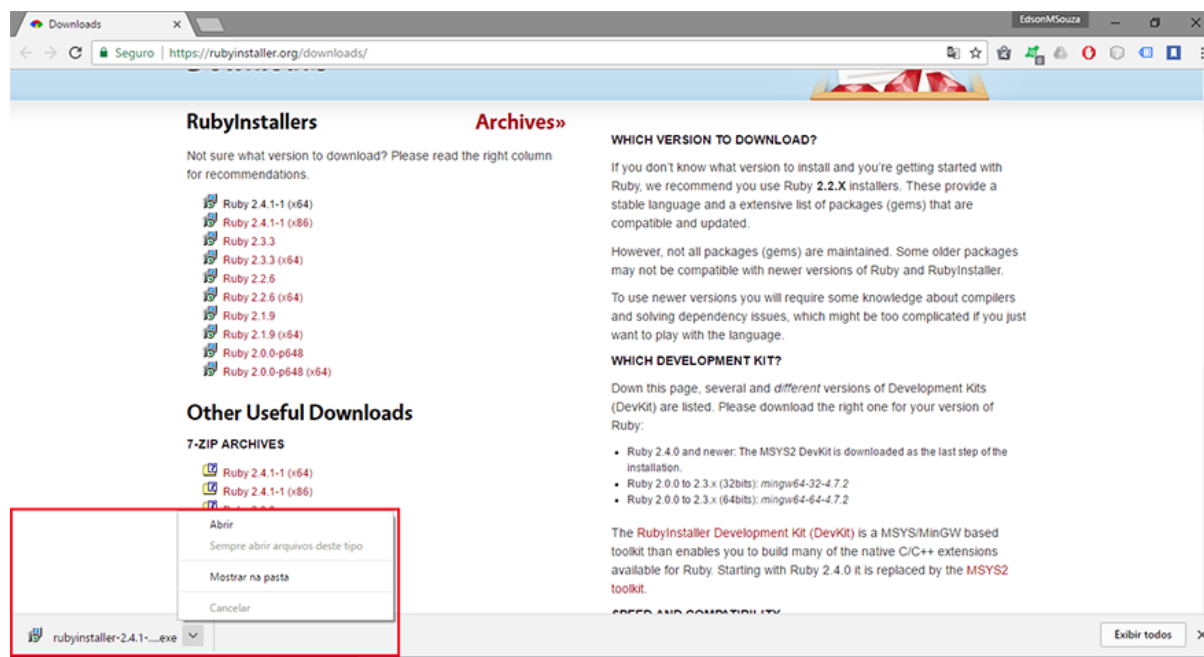
Downloads e Instalações (*Ambiente Windows*)

Para podermos trabalhar com o SASS é necessário instalar primeiramente o Ruby, que é uma linguagem de programação. Depois de instalada, vamos obter o SASS.

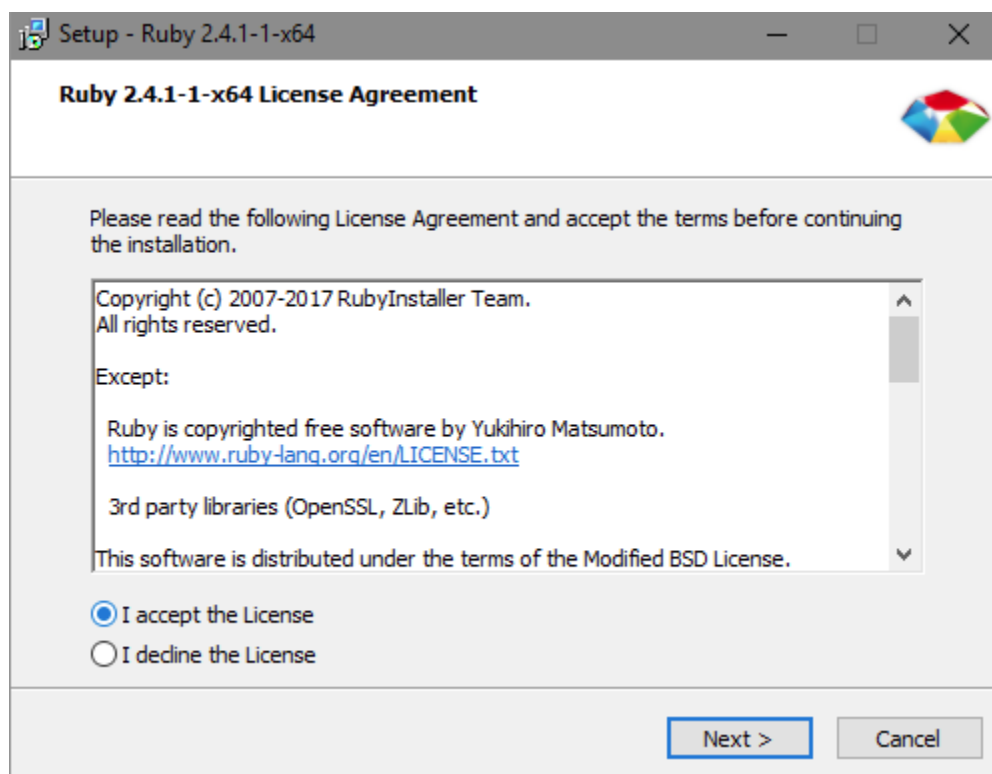
Faça o *download* do Ruby, acessando este endereço <https://rubyinstaller.org/downloads/> (<https://rubyinstaller.org/downloads/>), conforme a figura a seguir:



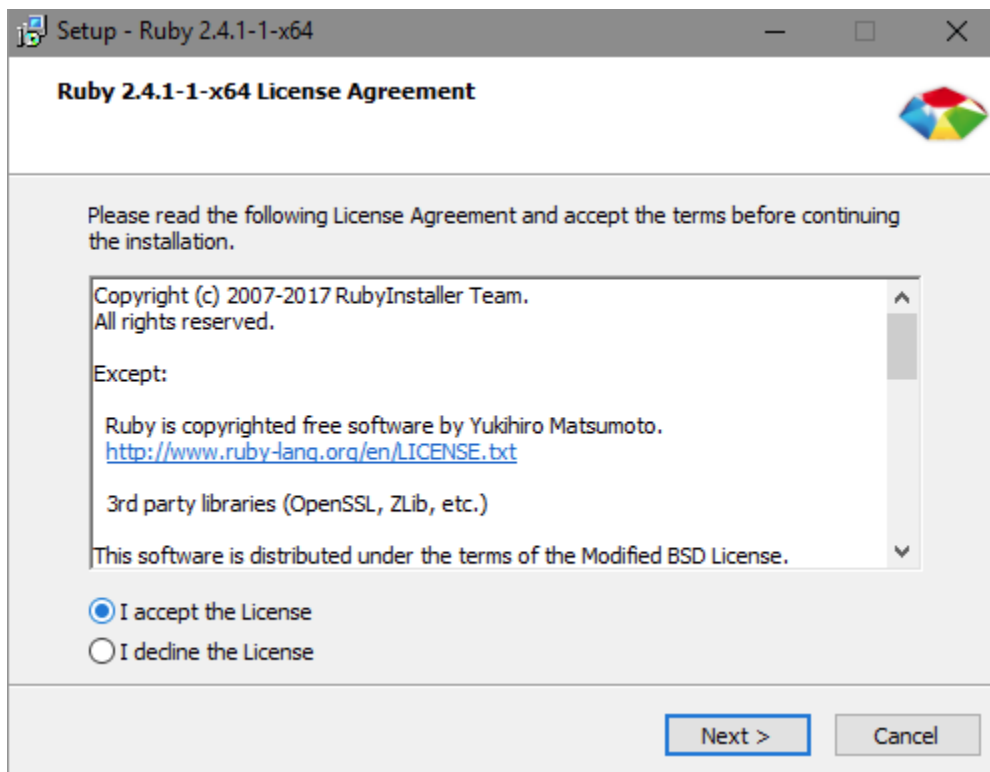
Selecione a versão mais adequada ao seu ambiente. Para esse tópico foi selecionada a versão Ruby 2.4.1-1(x64) e clique para baixar, conforme a figura a seguir:



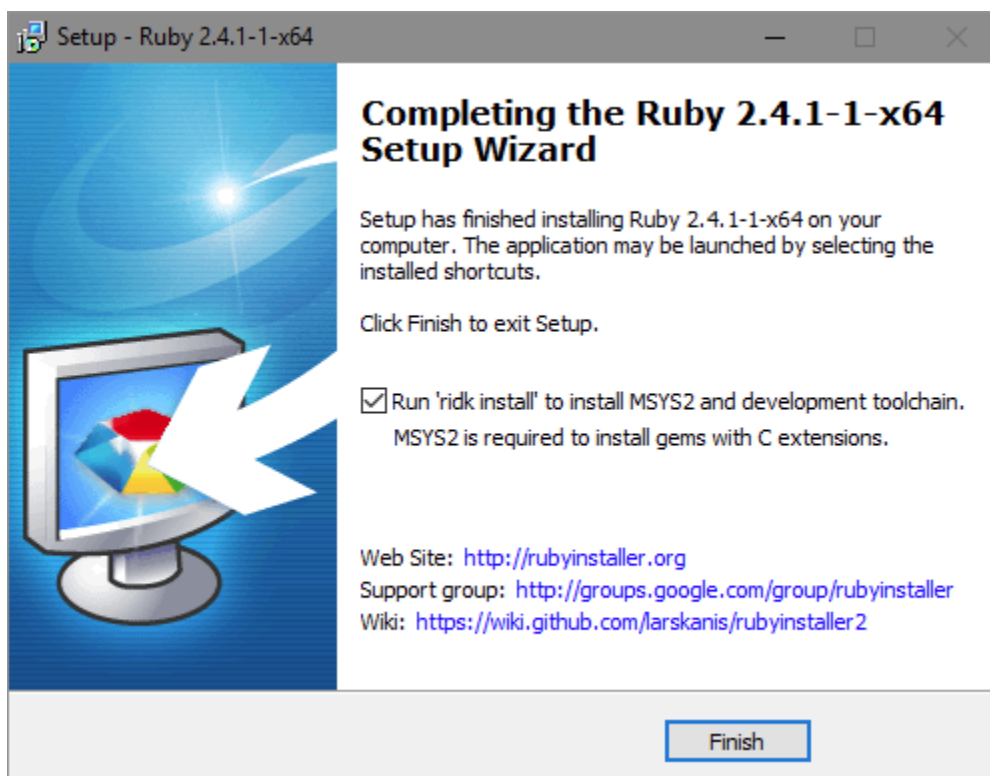
Após finalizado o download, inicie a instalação e siga as orientações conforme as figuras que serão apresentadas passo a passo.



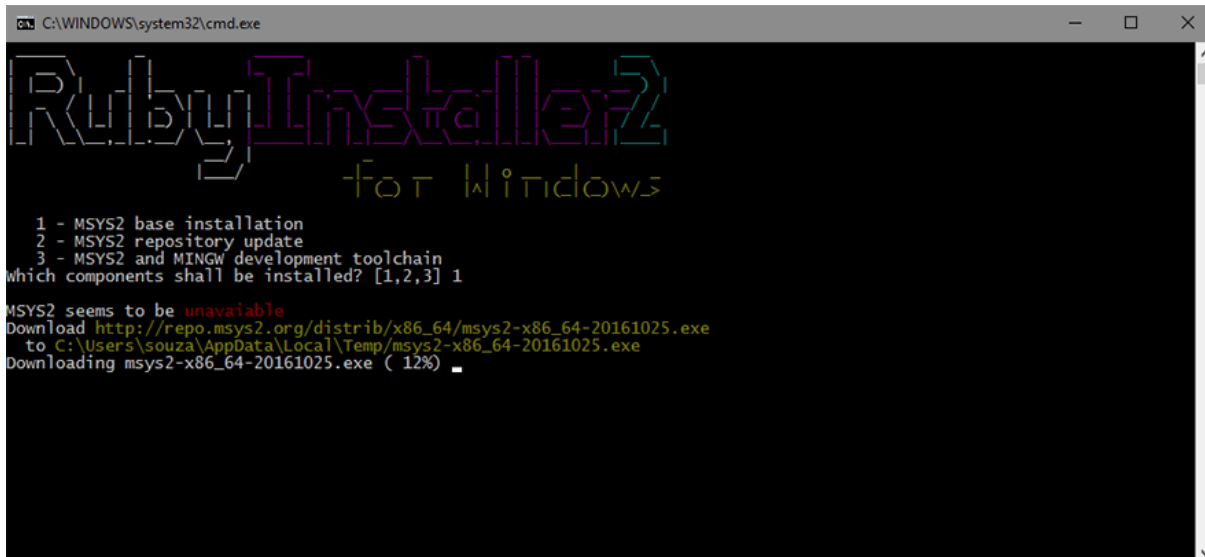
Na próxima tela, selecione o local de instalação. Neste tópico foi usado o caminho sugerido.



Após o processo de instalação, você receberá a próxima tela como mensagem.



Na próxima tela será solicitada a escolha de uma versão para instalação do MSYS2, que é um software que tem a função de emular um ambiente POSIX (Unix ou Linux) para realizar as compilações do Ruby. Escolha a primeira opção, digitando 1 pressionando ENTER, conforme a figura a seguir:



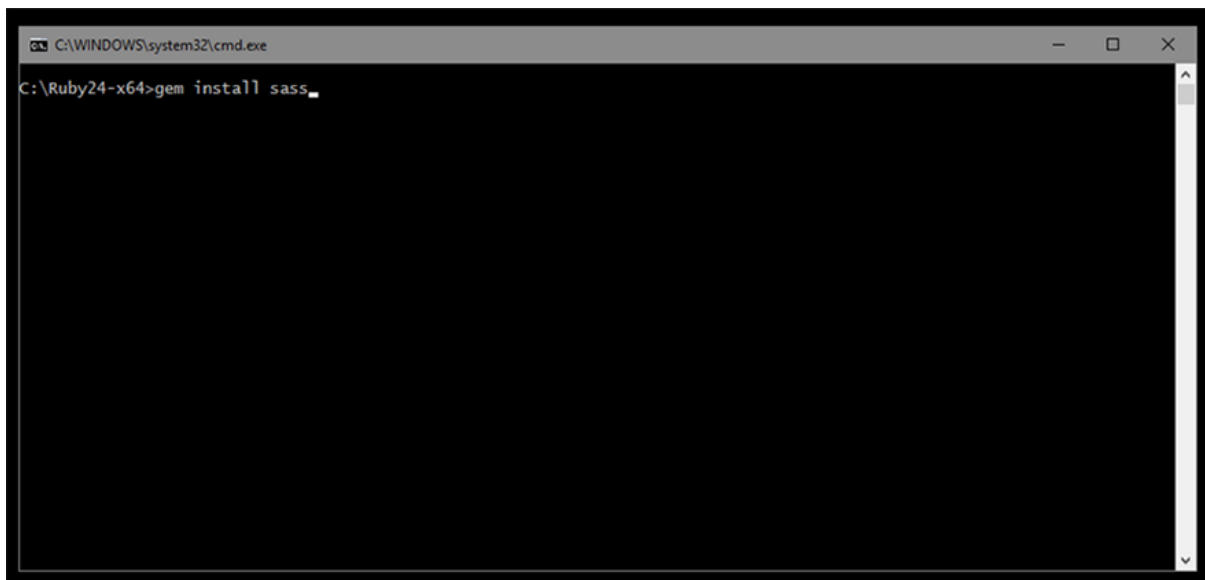
```
C:\WINDOWS\system32\cmd.exe

RubyInstaller2
for Windows

1 - MSYS2 base installation
2 - MSYS2 repository update
3 - MSYS2 and MINGW development toolchain
which components shall be installed? [1,2,3] 1

MSYS2 seems to be unavailable
Download http://repo.msys2.org/distrib/x86_64/msys2-x86_64-20161025.exe
to C:\Users\souza\AppData\Local\Temp\msys2-x86_64-20161025.exe
Downloading msys2-x86_64-20161025.exe ( 12%)
```

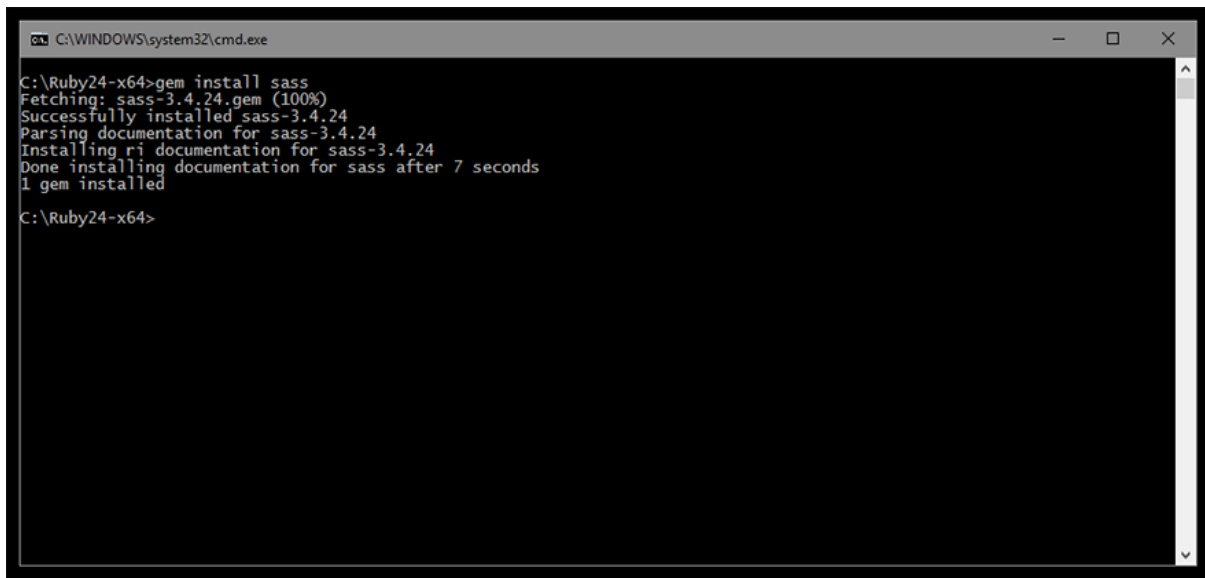
Responda às perguntas e prossiga com a instalação até seu término. Após finalizar a instalação, acesse a pasta “c:\Ruby24-x64” ou a que você escolheu na instalação e digite o comando “gem install sass”, conforme a figura a seguir:



```
C:\WINDOWS\system32\cmd.exe

C:\Ruby24-x64>gem install sass
```

Ao término do processo você deverá visualizar uma tela como na figura a seguir:



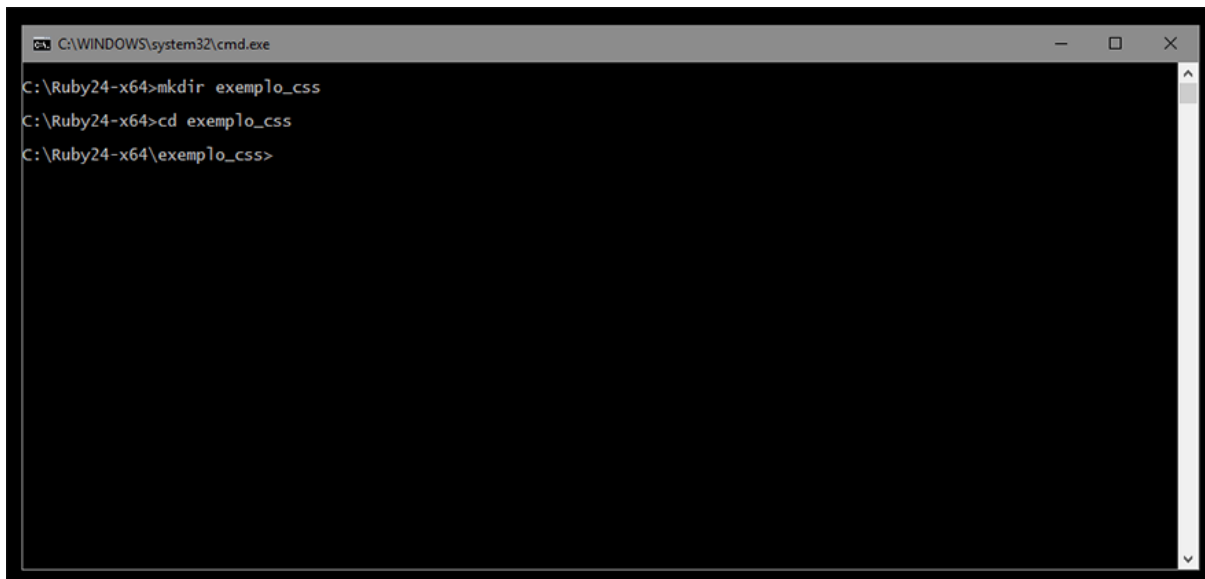
```
C:\WINDOWS\system32\cmd.exe
C:\Ruby24-x64>gem install sass
Fetching: sass-3.4.24.gem (100%)
Successfully installed sass-3.4.24
Parsing documentation for sass-3.4.24
Installing ri documentation for sass-3.4.24
Done installing documentation for sass after 7 seconds
1 gem installed
C:\Ruby24-x64>
```

Após terminar a instalação do Ruby, pode fechar todas as janelas.

Iniciando o SASS

Antes de iniciarmos com os exemplos, vamos colocar o SAS em execução.

Abra o terminal do Windows (Windows + R), acesse o diretório “c:\Ruby24-x64\” ou então no diretório (pasta) que você criou na instalação e crie uma pasta como o nome “exemplo_css”. O caminho completo ficará “c:\Ruby24-x64\exemplo_css”, conforme a figura a seguir:



```
C:\WINDOWS\system32\cmd.exe
C:\Ruby24-x64>mkdir exemplo_css
C:\Ruby24-x64>cd exemplo_css
C:\Ruby24-x64\exemplo_css>
```

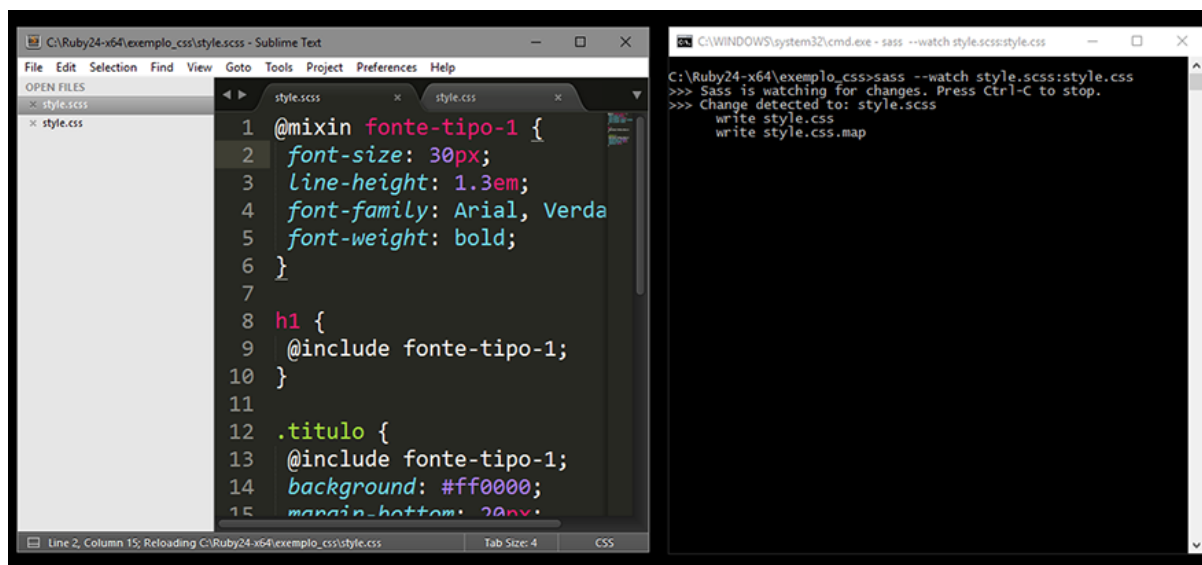
Digite então o comando:

```
1. sass --watch style.scss:style.css
```

e pressione ENTER

Você receberá a seguinte informação: >>> Sass is watching for changes. Press Ctrl-C to stop.

O comando anterior irá colocar o SASS no modo “escuta”. Isso significa que, quando fizermos qualquer alteração e salvarmos o arquivo “.scss”, ele automaticamente atualizará o arquivo “.css” final, conforme pode ser observado na figura a seguir:



Principais Funções do SASS

O SASS permite que sejam criadas variáveis, funções, repetição (*loops*) e muito mais, o que nos dá amplas possibilidades para criação e manutenção dos arquivos CSS.

Para nossos exemplos, será criado um arquivo com o nome de “style.scss”. Preste bem a atenção que a extensão do arquivo é “.scss” e não “.css”. Vamos ver então alguns desses recursos.

Variáveis - o uso de variáveis, assim como no JavaScript, servem para que possamos controlar alterações de forma geral dentro de um arquivo CSS, pois, ao alterar o valor de uma variável, todos os valores serão atualizados no arquivo final CSS e, conseqüentemente, também no site. Veja o exemplo a seguir:

```
1. $azul: #002db3;
2. $verde: #33cc33;
3. $vermelho: #ff0000;
4. $fonte: Arial, Verdana;
5.
6. a{
7.     color: $verde;
8.     font-family: $fonte;
9. }
10.
11. div{
12.     background: $azul;
13.     border: 3px solid $verde;
14. }
15.
16. .cabecalho{
17.     border: 2px solid $vermelho;
18. }
```

Nas linhas de 1 a 4 são criadas variáveis utilizando o símbolo cifrão “\$” antes do nome da variável. As variáveis são preenchidas com os valores e então na linha 6 é criado um estilo para tag “a”, para a qual são configuradas as propriedades “color” e “font-family”.

Na linha 11 é criada uma tag “div” que configura as propriedades “background” e “border”. Por fim, na linha 16, é criada uma classe denominada “cabecalho” e, e seu interior, o valor da propriedade “border” é atribuído.

Veja que nas linhas indicadas que estão atribuindo os valores as propriedades, existe a inclusão da variável que foi criada anteriormente. desta forma, quando o arquivo for salvo, o SASS fará a substituição dos valores das variáveis e criará o novo arquivo, no nosso caso “style.css” que conterà o código a seguir:

```
1. a {  
2.   color: #33cc33;  
3.   font-family: Arial, Verdana; }  
4.  
5. div {  
6.   background: #002db3;  
7.   border: 3px solid #33cc33; }  
8.  
9. .cabecalho {  
10.  border: 2px solid #ff0000; }
```

Agora é possível perceber onde que o SASS atuou, ou seja, ele trocou o valor das variáveis, por valores literais. A partir de agora já podemos utilizar nosso arquivo em nossa páginas web.

Repetição (*Loops*)

Em muitas situações necessitamos, por exemplo, criar cabeçalhos “h” com diferentes tamanhos e, dentro do CSS, somos forçados a criar várias referências. Com o laço de repetição fica muito mais fácil, pois, utilizando variáveis, podemos fazer a criação de referências e não só a substituição de valores.

Veja o código a seguir e observe que temos quatro formatações para “h” que vão de “h1” até “h4”.

```
1. h1{  
2.   font-size: 30px;  
3. }  
4. h2{  
5.   font-size: 22px;  
6. }  
7. h3{  
8.   font-size: 16px;  
9. }  
10. h4{  
11.   font-size: 12px;  
12. }
```

Mas, e se fosse necessário alterar o tamanho da fonte dessas referências e ainda criar mais uma entrada como “h5”? Percebeu o trabalho que daria editar cada uma delas individualmente e ainda criar ou na mão. Para isso há uma saída brilhante com o uso da estrutura de repetição “*foreach*”, onde podemos configurar como a repetição será realizada, permitindo a criação dinâmica de elementos.

Vamos ver então como fazer isso com o SASS e ganhar na produtividade no próximo exemplo.

```
1. $valor_inicial: 30px;
2. @each $h in h1, h2, h3, h4, h5{
3.
4.     #{ $h} {
5.         font-size: $valor_inicial;
6.     }
7.
8.     $valor_inicial: $valor_inicial - 4;
9. }
```

Na linha 1 é declarada uma variável “valor_inicial” com o valor de 30px. Esse valor será atribuído ao “h1”.

Na linha 2 o símbolo “@each” informa que se trata de uma repetição, onde a variável “\$h” receberá os valores descritos em “in” que são de h1 até h5.

Na linha 4 será criada uma entrada “h” que está descrita na linha 2. Na linha 5 é atribuído o valor da propriedade “font-size” de acordo com o valor contido em “valor_inicial”. Você pode ver que na linha 8 a variável “valor_inicial” está sendo realimentada com uma subtração de 4, ou seja, a cada iteração do laço é diminuído em 4 o valor da “variável_inicial”. Dessa forma, criamos então as referências “h” com valores bem distribuídos.

O resultado do processamento do código gerou o seguinte arquivos CSS:

```
1. h1 {
2.     font-size: 30px; }
3.
4. h2 {
5.     font-size: 26px; }
6.
7. h3 {
8.     font-size: 22px; }
9.
10. h4 {
11.     font-size: 18px; }
12.
13. h5 {
14.     font-size: 14px; }
```

Você pode perceber como esse tipo de automação pode aumentar sua produtividade, pois a redução de tempo é muito grande.

Mixins

Esse recurso do SASS é outro ponto muito interessante porque permite que uma parte de um código seja reaproveitado em outros lugares. Por exemplo, você tem tipos derivados de fontes ao longo do site ou páginas para cada tipo de área. Suponha que as fontes no “Painel de Controle do Usuário” sejam personalizáveis de acordo com o sexo dele. Então, você deverá criar dois arquivos CSS com valores diferentes para a formatação das páginas.

Imagine o trabalho de ter que editar as propriedades referentes a cor, tamanho, tipo de fonte, etc. Para isso existe os *mixins* que permitem que seja realizado o reaproveitamento do código. Vamos ver como escrever o arquivos “.scss” para essa funcionalidade a seguir.

```
1. @mixin fonte-tipo-1 {
2.     font-size: 28px;
3.     line-height: 1.3em;
4.     font-family: Arial, Verdana;
5.     font-weight: bold;
6. }
7.
8. h1 {
9.     @include fonte-tipo-1;
10. }
11.
12. .titulo {
13.     @include fonte-tipo-1;
14.     background: #ff0000;
15.     margin-bottom: 20px;
16.     padding: 30px;
17. }
```

Na linha 1 é declarado o “@mixin” com a atribuição de um identificador, ou seja, esse nome será utilizado para ser incluído em outras partes do código. Nas linhas de 2 a 5 são atribuídos os valores estáticos para as propriedades.

Na linha 9 e 13 está sendo referenciado o “mixin” por meio da instrução “@include”, significando que, quando for processado o SASS, os valores declarados “@mixin” serão inseridos nos locais indicados.

Veja a seguir o arquivo gerado pelo SASS após esse processamento:

```
1. h1 {  
2.     font-size: 28px;  
3.     line-height: 1.3em;  
4.     font-family: Arial, Verdana;  
5.     font-weight: bold; }  
6.  
7. .titulo {  
8.     font-size: 28px;  
9.     line-height: 1.3em;  
10.    font-family: Arial, Verdana;  
11.    font-weight: bold;  
12.    background: #ff0000;  
13.    margin-bottom: 20px;  
14.    padding: 30px; }
```

Mais uma vez você pode perceber como o SASS fez a criação das propriedades da fonte personalizado para as duas entradas no CSS.

Encerramos aqui essa breve introdução à personalização de arquivos CSS. O recomendado é que você explore os links em destaque a seguir para se aprofundar no assunto e descobrir muito mais sobre esse pré-processador CSS.

SAIBA MAIS!

SASS Oficial - <http://sass-lang.com/> (<http://sass-lang.com/>)

O Que é SASS - <http://www.ronaldodiniz.com.br/design/css-sass-vantagens.html>
(<http://www.ronaldodiniz.com.br/design/css-sass-vantagens.html>)

Tableless - <https://tableless.com.br/sass-um-outro-metodo-de-escrever-css/>
(<https://tableless.com.br/sass-um-outro-metodo-de-escrever-css/>)

TidBits - <http://www.tidbits.com.br/desenvolvendo-css-de-forma-mais-produtiva-usando-sass>
(<http://www.tidbits.com.br/desenvolvendo-css-de-forma-mais-produtiva-usando-sass>)

Resumo

Neste tópico foi apresentada uma breve introdução sobre o pré-processador SASS para personalização de arquivos CSS, mostrando como instalar e configurar o Ruby e o SASS, além de trazer exemplos e explicações sobre o funcionamento da ferramenta.

Conclusão

Trabalhar com desenvolvimento de interfaces, principalmente as dinâmicas, requer um amplo conhecimento de tecnologias e ferramentas que auxiliem na construção e na manutenção de páginas.

O SASS é uma dessas ferramentas que vai lhe auxiliar muito quando você ganhar experiência e estiver criando páginas e sites complexos. Portanto, coloque a mão na massa, explorando os links em destaque, pois assim você estará se preparando para o mercado de trabalho.

ATIVIDADE FINAL

Entre as alternativas a seguir, qual delas é uma vantagem em utilizar o SASS?

- A. Ganho de produtividade.
- B. Geração de arquivos CSS menores.
- C. Melhora o desempenho de uma página HTML.
- D. Gera aspectos visuais com melhor qualidade.

Qual a função das variáveis no SASS?

- A. Controlar alterações de forma geral dentro de um arquivo CSS.
- B. Configurar o CSS para obter uma formatação melhor.
- C. Economizar processamento do lado do cliente.
- D. Economizar processamento do lado do servidor.

Qual a função da instrução " @include"?

- A. Fazer a injeção de uma porção de código em outras partes do CSS.
- B. Cria uma entrada para inclusão no arquivo HTML.
- C. Melhorar a qualidade do arquivo CSS final.
- D. Indica o início de um código SASS.

REFERÊNCIA

ANICHE, Maurício F.; GEROSA, Marco A. Boas e Más Práticas no Desenvolvimento Web com MVC: Resultados de Um Questionário com Profissionais. 2015.

CATLIN, Hampton; CATLIN, Michael Lintorn. Pragmatic guide to Sass. Pragmatic Bookshelf, 2011.

DINIZ, Ronaldo. O que é SASS e quais suas vantagens para edição de CSS. 2015. Disponível em: <<http://www.ronaldodiniz.com.br/design/css-sass-vantagens.html>

(<http://www.ronaldodiniz.com.br/design/css-sass-vantagens.html>)>. Acesso em: 27 maio 2017.

SASS. CSS with superpowers. 2015. Hampton Catlin, Natalie Weizenbaum, Chris Eppstein. Disponível em: <<http://sass-lang.com/> (<http://sass-lang.com/>)>. Acesso em: 27 maio 2014.

ORIGAMID CODEX. CSS com SASS. 2016. Disponível em: <<https://www.origamid.com/codex/css-com-sass/> (<https://www.origamid.com/codex/css-com-sass/>)>. Acesso em: 26 maio 2017.

POPLADE, Thaiana. O que é Sass? Entenda esse outro método de escrever CSS. 2013. Disponível em: <<https://tableless.com.br/sass-um-outro-metodo-de-escrever-css/> (<https://tableless.com.br/sass-um-outro-metodo-de-escrever-css/>)>. Acesso em: 25 maio 2017.

