

Unit 13: SASS

CONTENTS

Objectives

Introduction

13.1 Sass (Systematically Awesome Style Sheets)

13.2 LESS

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After studying this unit, you will be able to:

- Understand Sass
- Design web page using SaSS
- Analyze LESS

Introduction

Sass(Syntactically Awesome Style Sheet) is a CSS pre-processor that is fully compatible with every version of CSS. It is a scripting language (like JS) that is compiled to CSS in the browser. It provides additional benefits over normal CSS styling that enhances the way of writing CSS styles. Since browsers are unable to read a SASS file, so, we are required to use a sass compiler that converts its file to a normal CSS file. It also helps reduces the overall length of the code by discarding the repeated CSS code and therefore saves time. It was designed by Hampton Catlin and developed by Natalie Weizenbaum in 2006.

13.1 Sass (Systematically Awesome Style Sheets)

Sass stands for Systematically Awesome Style Sheets.

It is a CSS pre-processor. It is an extension of CSS that is used to add power and elegance to the basic language. It facilitates you to add variables, nested rules, mixins, inline imports, inheritance and more, all with fully CSS-compatible syntax.

Sass is more stable and powerful CSS extension language that describes style of document cleanly and structurally. It is very useful to handle large style sheets by keeping them well organized and running quickly small style sheets.

Features of SASS:

1. SASS is CSS-compatible i.e. it is fully compatible with every version of CSS.
2. It supports the various extension of Languages such as variables, nesting, and mixins.

3. It is well-formatted with supports the customizable output.
4. It facilitates a number of useful functions for manipulating colors and other values, etc.
5. It provides the Sass pre-processor that helps the web browser recognize the Sass codes into simple standard CSS.
6. Sass is fully CSS-compatible.
7. It is more stable, powerful and elegant than CSS.
8. It is based on JavaScript and is superset of CSS.
9. It has its own syntax and compiles to readable CSS.
10. It is an open-source pre processor that is interpreted into CSS.
11. It supports language extensions such as variables, nesting, and mixins.

The conversion of one syntax to another can automatically be done using the sass-convert command-line tool in the file. SASS can be implemented in 5 ways:

- Using the import statement
- Using the node & npm
- Using the Command-line tool
- Using the Standalone-Ruby module
- Using installing the Plugin

Why use Sass

- Sass is a pre-processing language and it has its own syntax for CSS.
- It is easy, short and clean in a programming construct.
- It has some features that are used for creating awesome style sheets and facilitates writing code more efficiently and easy to maintain.
- It contains all the features of CSS along with some advance features.
- It provides document style presentation better than flat CSS.
- It facilitates reusability methods, logic statements and some of the built in functions like color manipulation, mathematics and parameter lists.
- It facilitates you to keep your responsive design project more organized.
- You don't need to repeat similar CSS again and again in your project.



Notes

Syntax for comments in regular CSS starts with /* comments...*/, while in SASS there are two type of comment, the single line comments // and the multiline CSS comments with /**/.

Installation Process of SASS

System Requirements for SASS

Operating System – Cross-platform

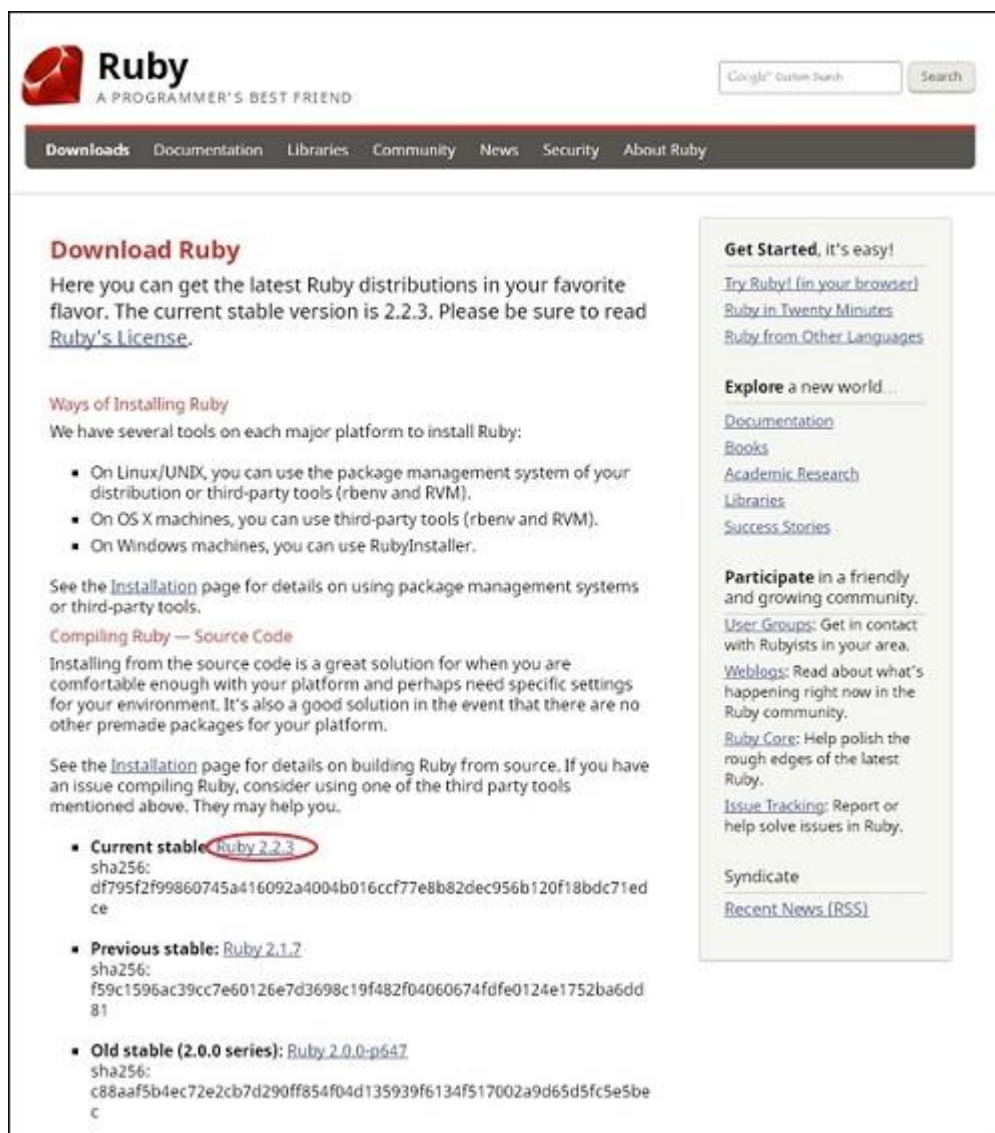
Browser Support – IE (Internet Explorer 8+), Firefox, Google Chrome, Safari, Opera

Programming Language – Ruby

Installation of Ruby

Step 1 – Open the link <https://www.ruby-lang.org/en/downloads/>, you will see a screen as shown below

–



The screenshot shows the Ruby website's 'Download Ruby' page. The header includes the Ruby logo, the tagline 'A PROGRAMMER'S BEST FRIEND', a search bar, and a navigation menu with links to Downloads, Documentation, Libraries, Community, News, Security, and About Ruby. The main content area is titled 'Download Ruby' and explains that the current stable version is 2.2.3. It provides instructions on how to install Ruby on Linux/UNIX, OS X, and Windows. A sidebar on the right offers links to 'Get Started' resources, 'Explore a new world...' (Documentation, Books, Academic Research, Libraries, Success Stories), and 'Participate' in the community (User Groups, Weblogs, Ruby Core, Issue Tracking). The bottom section lists three versions of Ruby with their SHA256 hashes: Current stable (2.2.3), Previous stable (2.1.2), and Old stable (2.0.0 series).

Download Ruby

Here you can get the latest Ruby distributions in your favorite flavor. The current stable version is 2.2.3. Please be sure to read [Ruby's License](#).

Ways of Installing Ruby

We have several tools on each major platform to install Ruby:

- On Linux/UNIX, you can use the package management system of your distribution or third-party tools (rvm and RVM).
- On OS X machines, you can use third-party tools (rvm and RVM).
- On Windows machines, you can use RubyInstaller.

See the [Installation](#) page for details on using package management systems or third-party tools.

Compiling Ruby — Source Code

Installing from the source code is a great solution for when you are comfortable enough with your platform and perhaps need specific settings for your environment. It's also a good solution in the event that there are no other premade packages for your platform.

See the [Installation](#) page for details on building Ruby from source. If you have an issue compiling Ruby, consider using one of the third party tools mentioned above. They may help you.

- Current stable:** [Ruby 2.2.3](#)
sha256:
df795f2f99860745a416092a4004b016ccf77e8b82dec956b120f18bdc71ed
ce
- Previous stable:** [Ruby 2.1.2](#)
sha256:
f59c1595ac39cc7e60126e7d3698c19f482f04060674fdfe0124e1752ba6dd
81
- Old stable (2.0.0 series):** [Ruby 2.0.0-p647](#)
sha256:
c88aaf5b4ec72e2cb7d290ff854f04d135939f6134f517002a9d65d5fc5e5be
c

Get Started, it's easy!

[Try Ruby! \(in your browser\)](#)
[Ruby in Twenty Minutes](#)
[Ruby from Other Languages](#)

Explore a new world...

[Documentation](#)
[Books](#)
[Academic Research](#)
[Libraries](#)
[Success Stories](#)

Participate in a friendly and growing community.

[User Groups](#): Get in contact with Rubyists in your area.
[Weblogs](#): Read about what's happening right now in the Ruby community.
[Ruby Core](#): Help polish the rough edges of the latest Ruby.
[Issue Tracking](#): Report or help solve issues in Ruby.

Syndicate

[Recent News \(RSS\)](#)

Download the *Current stable* version of the zip file.

Step 2 – Next, run the setup to install **Ruby** on the System.

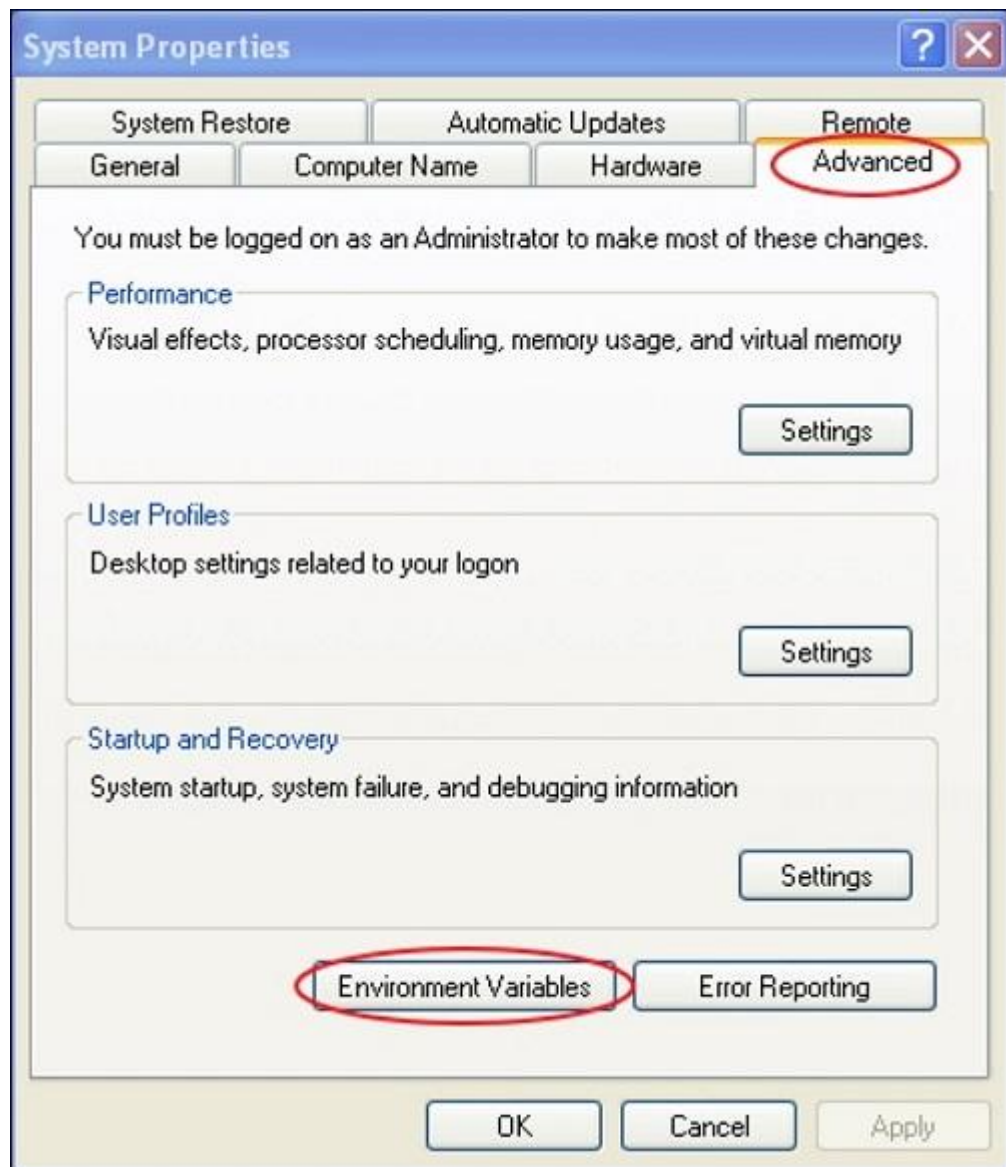
Step 3 – Next, add Ruby bin folder to your *PATH User Variable* and *System Variable* to work with gem command.

Path User Variable –

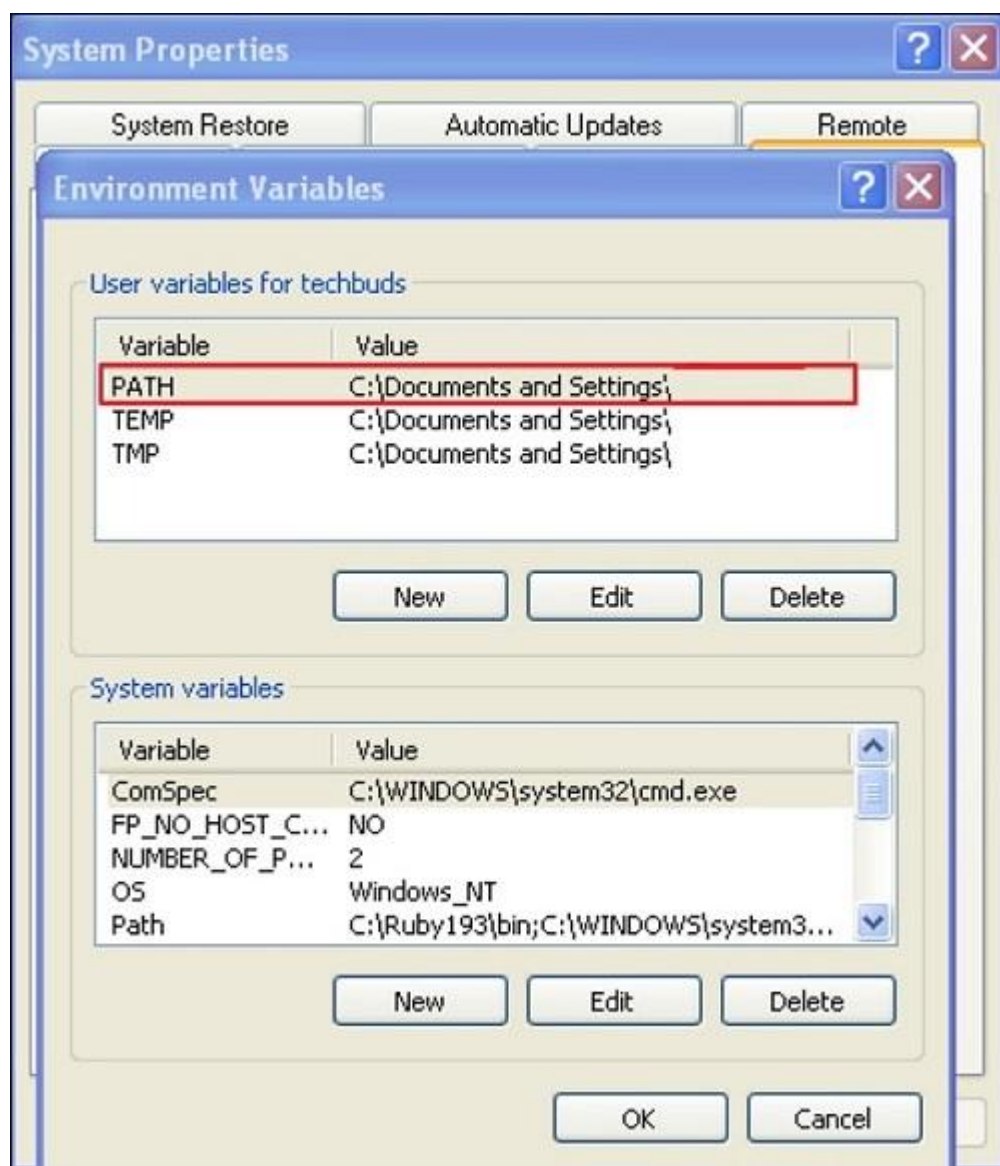
Right Click the **My Computer** icon.

Select **Properties**.

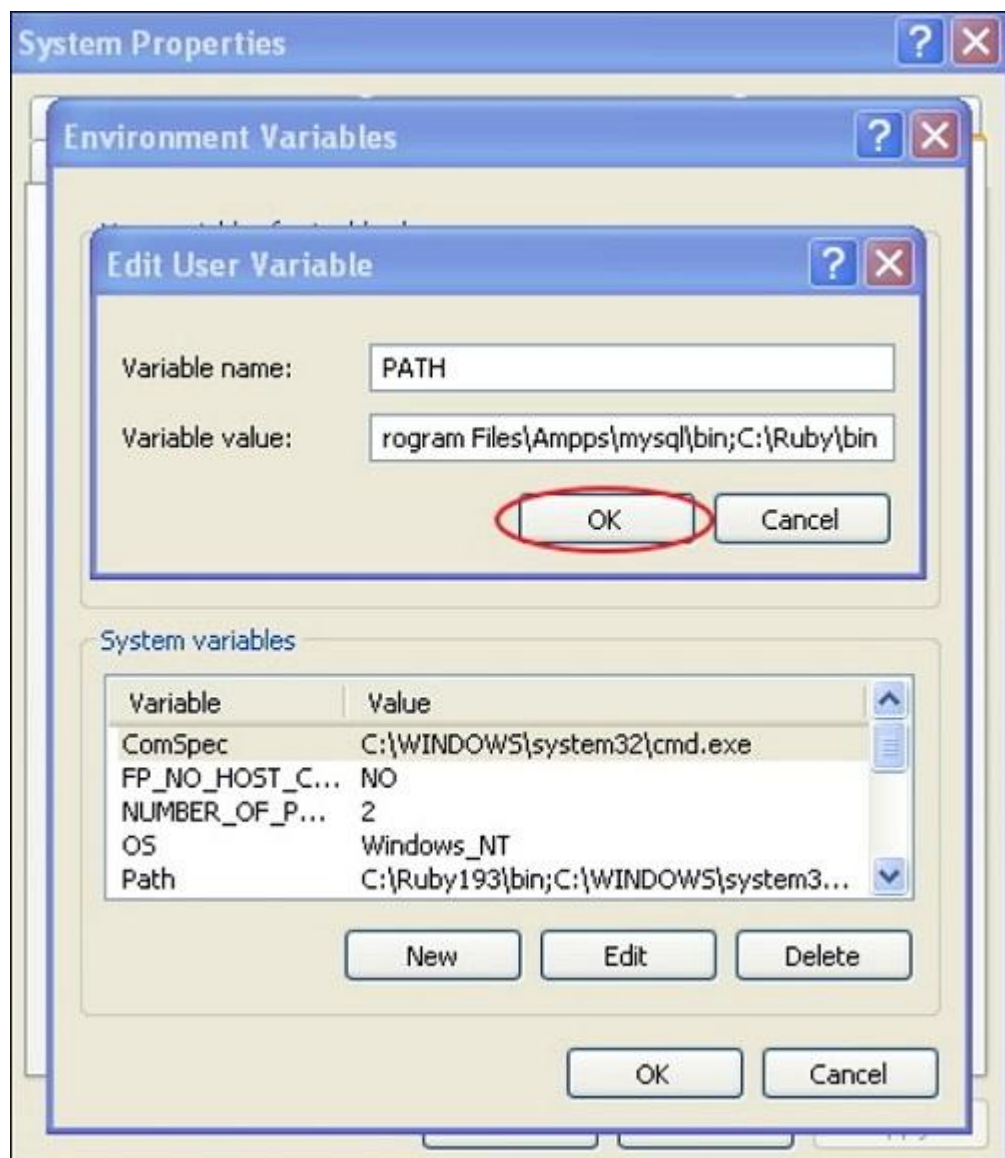
Next, click the **Advanced** tab and click **Environment Variables**.



In the *Environment Variables* window, double click the *PATH* as shown in the screenshot given below –



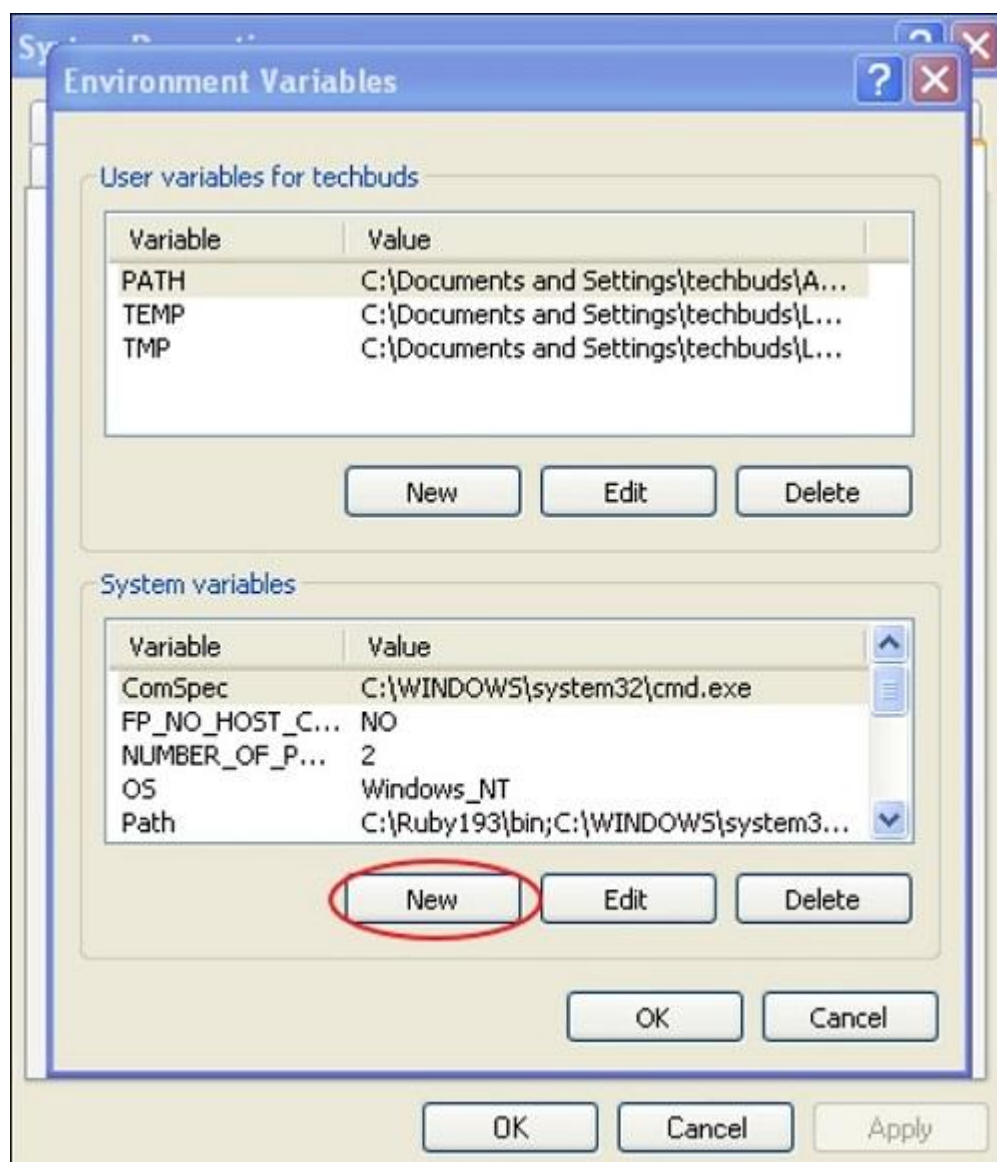
You will get an *Edit User Variable* box as shown. Add ruby bin folder path in the *Variable value* field as **C:\Ruby\bin**. If path is already set for other files, then put semicolon after that and add the Ruby folder path as shown below.



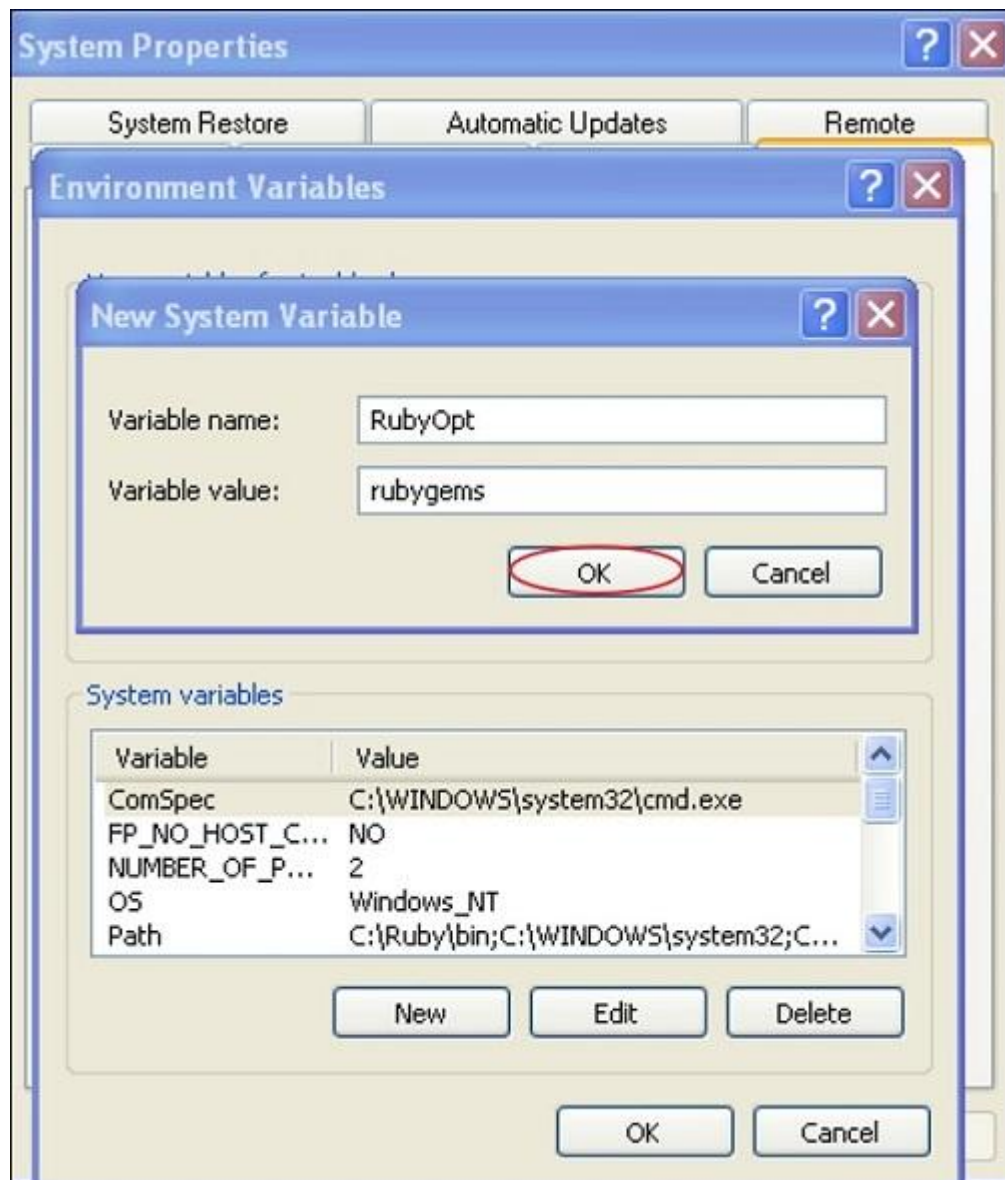
Click the **OK** button.

System Variable –

Click the **New** button.



Next, the **New System Variable** block is displayed as shown below.



Enter **RubyOpt** in the *Variable name* field and **rubygems** in the *Variable value* field. After writing the *Variable name* and *value*, click the **OK** button.

Step 4 – Open the command prompt in your system and enter the following line –

gem install sass

Step 5 – Next, you will see the following screen after installing SASS successfully.

```

C:\>cd ruby
C:\Ruby>gem install sass
Fetching: sass-3.4.19.gem (100%)
Successfully installed sass-3.4.19
1 gem installed
Installing ri documentation for sass-3.4.19...
Installing RDoc documentation for sass-3.4.19...
C:\Ruby>

```



Example

The following is a simple example of SASS.


```

<html>
<head>
<title> Import example of sass</title>
<linkrel="stylesheet"type="text/css"href="style.css"/>
</head>

<body>
<h1>Simple Example</h1>
<h3>Welcome to TutorialsPoint</h3>
</body>
</html>

```

Now, we will create file as *style.scss*, which is quite similar to CSS and the only one difference is that it will be saved with *.scss* extension. Both, *.htm* and *.scss* files should be created inside the folder **ruby**. You can save your *.scss* file in the folder **ruby\lib\sass** (before this process, create a folder as **sass** in lib directory).

```

h1{
  color: #AF80ED;
}

```

```

h3{
  color: #DE5E85;
}

```

You can tell SASS to watch the file and update the CSS whenever SASS file changes, by using the following command –

```
sass --watch C:\ruby\lib\sass\style.scss:style.css
```



```

C:\>cd ruby
C:\Ruby>sass --watch C:\ruby\lib\sass\style.scss:style.css
>>> Sass is watching for changes. Press Ctrl-C to stop.
      write style.css
      write style.css.map

```

When you run the above command, it will create the *style.css* file automatically. Whenever you change the SCSS file, the *style.css* file will be updated automatically.

The *style.css* file will have the following code when you run the above given command –

```

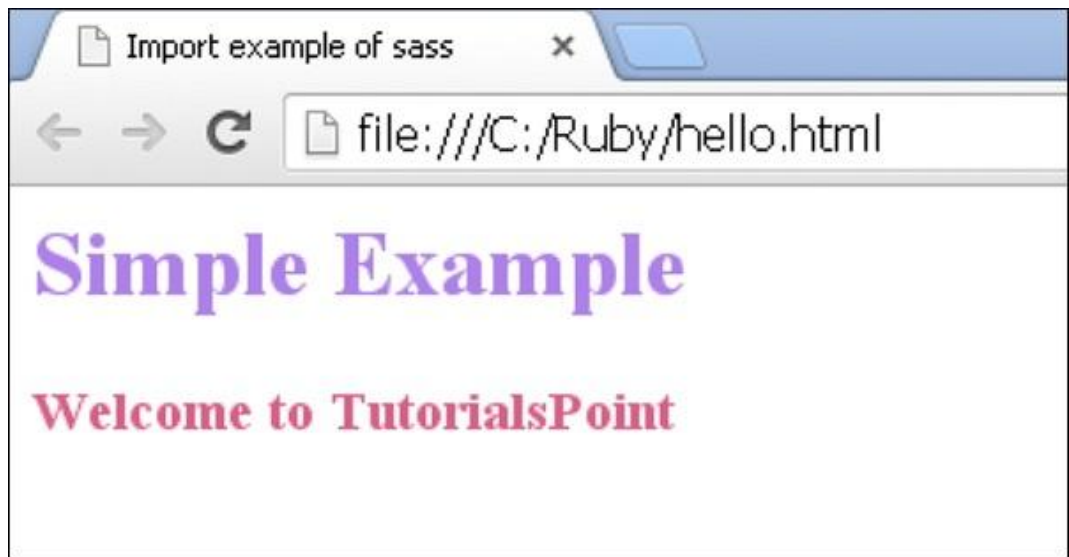
style.css
h1 {
  color: #AF80ED;
}
h3 {
  color: #DE5E85;
}

```

Let us carry out the following steps to see how the above given code works –

Save the above given code in **hello.html** file.

Open this HTML file in a browser.



Example

Mystyle.scss

```
$myFont: Helvetica, sans-serif;  
$myColor: red;  
$myFontSize: 18px;  
$myWidth: 680px;
```

```
body {  
  font-family: $myFont;  
  font-size: $myFontSize;  
  color: $myColor;  
}
```

```
#container {  
  width: $myWidth;  
}
```

Mystyle.css

```
body {  
  font-family: Helvetica, sans-serif;  
  font-size: 18px;  
  color: red;  
}
```

```
#container {  
  width: 680px;  
  border: 1px blue double;  
}
```

Page.html

```
<!DOCTYPE html>  
<html>  
<link rel="stylesheet" href="mystyle.css;?>">  
<body>  
  
<h1>Hello World</h1>  
  
<p>This is a paragraph.</p>  
  
<div id="container">This is some text inside a container.</div>  
  
</body>  
</html>
```

Output

Hello World

This is a paragraph.

This is some text inside a container.

13.2 LESS

Less is a CSS pre-processor that facilitates you to customize, manage and reuse the style sheets for the webpage. Less is an extension of CSS and a dynamic style sheet language which can be run on client side or server side.

Less is an open source language. It is also cross browser compatible.

Features of Less

- Less is clean, compact, more readable code and written in a well organized way.
- Less supports cross-browser compatibility.
- Less is faster and easier.
- Less is written in JavaScript. It compiles faster than other preprocessor of CSS.
- Less provides variables which makes its maintenance faster.
- Less provides nesting which makes the code short, clean and organized in a specific way

- Less facilitates you to define styles which can be reused throughout the code.
- Less is an extension of CSS. So it is also called super set of CSS.
- Less is capable enough to sort out the problem of code redundancy.

Advantages of Less

1. Less is a CSS preprocessor. After compilation, it generates simple CSS which works across the browsers.
2. Less supports cross-browser compatibility.
3. Less codes are simple, clean and well organized because the use of nesting.
4. Maintenance in Less can be achieved faster because the use of variables.
5. Less provides a list of operators which makes coding faster and time saving.
6. Less facilitates you to reuse the whole classes easily by referencing them in your rule-set.
7. Less is new and preferred over the conventional CSS because may ease the lengthy styling.

Disadvantages of Less

1. You must have to spend some time to learn Less if you are new to CSS preprocessing.
2. Less provides fewer frameworks as compared to older preprocessor like SASS which contains frameworks like Compass, Gravity and Susy.
3. In Less, there is a tight coupling between modules so it takes more effort to reuse and/or test dependent modules.

Less Comments

Less comments are non-executable statements that are placed inside the source code. These comments are written to make source code clear and easier to understand by other developers and testers. Comments can be written in block style and inline within the Less code, but single line comments are not appeared in CSS code.

There are two types of comments supported in Less.

Single line comments: In Less, single line comments are written using `//` followed by comments. The single line comments are not displayed in generated CSS output.

Multiline comments: In Less, multiline comments are written between `/* */`. The multiline comments are preserved in generated CSS output.

Less code looks like:-

```
@color-base: #2d5e8b;

.class1 {
  background-color: @color-base;
  .class2 {
    background-color: #fff;
    color: @color-base;
  }
}
```

Why Sass is considered better than LESS?

- Sass allows you to write reusable methods and use logic statements, e., loops, and conditionals
- Sass user can access Compass library and use some awesome features like dynamic sprite map generation, legacy browser hacks and cross-browser support for CSS3 features
- Compass also allows you to add an external framework like Blueprint, Foundation or Bootstrap on top
- In LESS, you can write a basic logic statement using a 'guarded mixin', which is equivalent to Sass if statements
- In LESS, you can loop through numeric values using recursive functions while Sass allows you to iterate any kind of data
- In Sass, you can write your own handy functions

LESS VS SASS

LESS	SaSS
LESS uses JavaScript and processed at client-side	Sass is coded in Ruby and thus processed to server-side
Variable names are prefaced with the @symbol	Variable name are prefaced with \$ symbol
LESS does not inherit multiple selectors with one set of properties	Sass inherits multiple selectors with one set of properties
LESS does not work with "unknown" units neither it returns syntax error notification for incompatible units or maths related syntax error	Sass allows you to work with "unknown" units also returns a syntax error notification for incompatible units

Summary

- Less (which stands for Leaner Style Sheets) is a backwards-compatible language extension for CSS. This is the official documentation for Less, the language and Less.js, the JavaScript tool that converts your Less styles to CSS styles.
- Sass is completely compatible with all versions of CSS. We take this compatibility seriously, so that you can seamlessly use any available CSS libraries.
- Sass boasts more features and abilities than any other CSS extension language out there. The Sass Core Team has worked endlessly to not only keep up, but stay ahead.
- There are an endless number of frameworks built with Sass. Bootstrap, Bourbon, and Susy just to name a few.

Keywords

Less: Less is a dynamic preprocessor style sheet language that can be compiled into Cascading Style Sheets and run on the client side or server side. Designed by Alexis Sellier, Less is influenced by Sass and has influenced the newer "SCSS" syntax of Sass, which adapted its CSS-like block formatting syntax.

SaSS: Sass is a preprocessor scripting language that is interpreted or compiled into Cascading Style Sheets. SassScript is the scripting language itself.

SelfAssessment

1. What are the most attractive features of SASS?
 - A. It is more stable, powerful and fully compatible to CSS3
 - B. It is time saving because it facilitates you to write CSS in less code
 - C. It uses its own syntax
 - D. All of these

2. Who is the inventor of SASS?
 - A. James Gosling
 - B. Guido van Rossum
 - C. MiskoHevery
 - D. Hampton Catlin

3. What are the key features for Sass include
 - A. Full CSS3-compatible
 - B. Language extensions such as nesting, variables, and mixins
 - C. Many useful functions for manipulating colors and other values
 - D. All of the above

4. What are the reasons behind using SASS?
 - A. You can write codes easily and efficiently and they are easy to maintain
 - B. It is a pre-processing language which provides its own syntax for CSS
 - C. It is a superset of CSS which contains all the features of CSS and is an open source pre-processor, coded in Ruby
 - D. All of the above

5. From the following what are the reasons behind using SASS?
 - A. You can write codes easily and efficiently and they are easy to maintain
 - B. It is a pre-processing language which provides its own syntax for CSS
 - C. It is more stable and powerful CSS extension and style documents more clearly and structurally
 - D. All of the above

6. What are the data types that SassScript supports?
 - A. Numbers
 - B. Strings of texts
 - C. Colors

- D. All of these
7. What is the difference between Sass and SCSS?
- A. Sass is a CSS pre-processor with syntax advancements and an extension of CSS3
 - B. Sass has two syntax
 - C. The first syntax is SCSS" and it uses the .scss extension
 - D. All of the above
8. How many ways SASS can be used?
- A. It can be used as a command line tool
 - B. It can be used as a standalone Ruby module
 - C. It can be used as a plugin for any Rack-enabled framework
 - D. All the above mentioned
9. Which of the following are types of mixin arguments?
- A. Keyword Arguments
 - B. variable Arguments
 - C. All of the above
 - D. None
10. What is the correct full form of SASS?
- A. System Asymmetric Style Sheets
 - B. Symbolic Asymmetric Style Sheets
 - C. Systematic Awesome Style Sheets
 - D. Syntactically Awesome Style Sheets
11. Why is SASS called a superset of CSS?
- A. SASS is called a superset of CSS because it is better than CSS.
 - B. SASS is called a superset of CSS because it is an open-source pre-processor written in the Ruby programming language.
 - C. SASS is called a superset of CSS because it contains all the features of CSS.
 - D. None of the Above
12. CSS stands for -
- A. Cascade style sheets
 - B. Color and style sheets
 - C. Cascading style sheets
 - D. None of the above
13. Which of the following is the correct syntax for referring the external style sheet?
- A. <style src = example.css>
 - B. <style src = "example.css" >

- C. <stylesheet> example.css </stylesheet>
 D. <link rel="stylesheet" type="text/css" href="example.css">
14. Which of the following directive is used to set the style rule to different media types?
 A. @media
 B. @import
 C. @extend
 D. @debug
15. What is the correct way to define a variable in SASS?
 A. \$primary-color: #888
 B. @primary-color: #888
 C. %primary-color: #888
 D. #primary-color: #888

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. D | 2. D | 3. D | 4. D | 5. D |
| 6. D | 7. D | 8. D | 9. C | 10. D |
| 11. C | 12. C | 13. D | 14. A | 15. A |

Review Questions

1. Explain what is Sass? How it can be used?
2. List out the key features for Sass?
3. Explain what is the difference between Sass and SCSS?
4. Why is Sass considered better than LESS?
5. Explain how Sass comments are different from regular CSS?



Further Readings

Bootstrap by Example, Silvio Morato
 Learning Bootstrap 4, Matt Lambert
 Bootstrap in 24 Hours, By Sams
 Jump Start Bootstrap,



Web Links

<https://sass-lang.com/>
<https://lesscss.org/>