***Rapporteur name:*** Ivan Karpovich.

***Presentations:*** SASS(SCSS).

***Quantity of slides:*** 18.

*Slide #1 (Intro)*

Hello everybody. My name is Ivan, and today, I'll tell you about SASS(SCSS). Let's start.

*Slide #2 (CSS preprocessor)*

– What is CSS-preprocessor? It is a preprocessor which extends the CSS language.

– The main task of the preprocessor – it is to simplify and accelerate the development of styles in the projects.

– CSS preprocessor converts code written using language preprocessor, into a clean and valid CSS-Code.

– As a good example, this image. It fully shows “how work CSS preprocessor”.

***– What do you know css preprocessor?*** I will help you. It is **SASS**( syntactically awesome stylesheets), **LESS** and **Stylus**.

* Ok, go next.

*Slide #3 (possibility SASS(SCSS))*

– Now I tell you about the possibilities SASS. It is the **Nesting, Variables, Math, Mixins, Imports, Cycles & C onditional expression** and **Extends & Placeholders**.

– Later I will tell you more about them.

*Slide #4 (compiling SASS(SCSS))*

– Before I tell you about the possibilities, I tell you about the compiling SASS.

– Browsers do not understand Sass, so you need to convert the file in the CSS format. There are several ways, this is one of them.

– One convenient online tools to edit Sass and get clear CSS, located below the link. It is the site “SassMeister”.

*Slide #5,6 (Nesting)*

– CSS lacks visual hierarchy while working with child selectors.

– You have to write selectors and their combinations in separate lines.

– Nesting provides a visual hierarchy as in the HTML and increases the readability.

– Here are a few examples of nesting convenience. Actually easier to read. **Go next**!

*Slide #7 (Variables)*

– The CSS is often a situation arises where the same value is applied several times. It could be anything.

– If you need to change everywhere, for example, one color to another, you will have to use the search and replacement text.

– Variables also allow you to specify all necessary values ​​in the beginning of the code, and then only to refer to them.

– It is necessary to edit the value of a variable, it will automatically be substituted in all the places where there is a variable.

– Any variable begins with the $ symbol.

– Take a look at an example.

*Slide #8 (Math)*

– Math operations can be used for standard arithmetic or unit conversions.

– SASS support arithmetic between different units. In addition to simple math, pre-processors also have complex math support such as ceiling, rounding, getting min or max value in a list etc.

– Take a look at an example.

*Slide #9,10,11 (Mixins)*

– Mixins is one of the most powerful pieces in Sass and its effect reminiscent functions in programming languages.

– In other words, this piece of code that can be inserted several times.

– To create mixins used **@mixin** command. Inside the braces writing style rules. At the point where we need to insert an impurity, add the command **@include** with her name.

– Look for examples.

*Slide #12 (Imports)*

– Splitting a single large document into multiple small simplifies development because the code is simpler and clearer, it is easier to debug.

– To download one file from another to use Sass **@import** command.

– The Sass has no performance issues or load time of pages, because a lot of small files in the end are combined into a single stylesheet.

– So easy to edit saved files and connect third-party libraries with high-speed downloads of styles.

*Slide #13,14 (cycles & conditional expression)*

– You all know how cycles work. I will only add that to create a cycle in the Sass is **@for** command.

– Cycles for convenient use CSS-sprites that use a background image containing multiple images.

– And you all know conditionak expression work. I will only add that SASS have @if command and @else command.

– Conditional expressions are convenient for setting triggers - the so-called check for compliance with some specified conditions.

­– Look for example.

*Slide #15,16 (Extends & Placeholders)*

– Some selectors use the same style rules. In order not to repeat the same code several times in the CSS is used for the convenience of the grouping representation and reduce code.

– In Sass, the **@extend** directive is an outstanding way to inherit already existing styles.

*Slide #17 (what is the difference SASS and SCSS)*

– If you look at the slide, then do all the notice. The difference is only one, and it is in braces. That all.

*Slide #18 (what is the difference SASS and SCSS)*

– Lastly, you want to be cool like Captain America, to simplify your life and Learn SASS.