COMS W4115 Project Proposal
CSSLang

Wayne Yang
wy2149@columbia.edu

June 6, 2010

1 Introduction

In the past few years, there has been a growing trend of building web sites that comply with web standards to ensure greater accessibility, usability, and interoperability of web pages. These web standards commonly refer to Hypertex Markup Language (HTML), Cascading Style Sheets (CSS), and ECMAScript (or Javascript). CSS is a style sheet language primarily used to describe the presentations of web pages [1]. It enables the separation of presentation from the semantic or structure of web pages and allows web designers to reuse style sheets across web pages [2].

Throughout this document, the term "CSS" refers to the the CSS Level 2 Revision 1 (CSS 2.1) specification by W3C. Although CSS can also be applied to XML documents, this project focuses on CSS in the HTML context.

This project is to design a language CSSLang that adds features typically seen in programming languages to CSS.

2 Motivations

Currently, CSS has no notion of variables which makes it diffcult for users to reuse style properties or create CSS rule sets that have parameterized property values. For example, to specify similar border and margin style for two class selectors *header* and *footer*, one has to duplicate style properties with slight variations in values:

```
#header {
   border-width: 1px;
   margin-width: 1px;
}
#footer {
   border-width: 5px;
   margin-width: 5px;
}
```

In addition, CSS lacks support for expressions in the property value such as the following:

```
#header {
   border-width: 5px + 3em;
}
```

Inpsired by *LESS* and *Sass* which attempts to address these shortcomings of CSS, CSSLang implements some of the features and syntax of these languages.

3 CSSLang

A CSSLang source file is compiled into a valid CSS style sheet with file extension .css.

3.1 Rule sets

CSSLang will implement the syntax of a simplified CSS rule set which consists of a selector followed by a declaration block. A declaration block consists of one of more pairs of style property and its value separated by a colon. Each pair of property and value is terminated by a semicolon. An example of a CSS rule set in CSSLang looks like the following:

```
selector {
    property1: value1;
    property2: value2;
    ...
}
```

CSS selectors are pattern matching rules that determine which style rules apply to elements in a web page [1]. In CSSLangl, only ID and class selectors will be implemented. An ID selector starts with a '#' followed by the ID name. So *#content* is a selector that match HTML element with ID attribute equal to *content*. On the other hand, class selectors are defined with a prefix '.'. An example of a class selector *.footer* will match HTML elements with class attribute equal to *footer*.

3.2 Variables

Variables in CSSLang are declared in the following syntax:

```
$variable-name: variable-value
```

where variable-value can be any valid CSS property value. A variable can be used inside a rule set as follows:

```
$length: 1px;
#header {
    border-width: $length;
}
```

The values of variables persist after the variable declarations until the end of the file. CSSLang supports the following type of values for the variables:

- numbers: 1, 2px, and -3
- strings: unquoted strings
- color: #FF0000

3.3 Operators

The basic arithmetic operators are supported in CSSLang: +, -, *, and / which are addition, substraction, multiplications, and divisions respectively. Operators can be used in expressions in the property values of a rule set.

3.4 Control structures

As in the Sass language, CSSLang will support @if directive for conditional declaration of style properties. This directive is used inside a declaration of style properties. A basic usage is as follows:

```
$font-style : italic;
div {
    font-style: $font-style;
    @if font-style = normal {
        font-size: 5px;
    } @else if $font-style == italic {
        font-size: 10px;
    } @else {
        font-size: 15px;
    }
}
```

This is compiled to:

```
div {
    font-style: italic;
    font-size: 10px;
}
```

3.5 Rule templates

Rule templates (or just templates) are rule sets that can be re-used throughout a CSSLang source file. They can be customized by taking parameters. For example, a template can be declared as

```
@def foo-header($color, $align) {
    text-align: $align;
    color: $color;
}
```

To use template *foo-header*, use the @include directive. For example:

```
h1 {
    @include foo-header(#ffffff, center);
}
```

is compiled to:

```
h1 {
    text-align: center;
    color: #ffffff;
}
```

4 Examples

4.1 First example

As mentioned in the earlier in this document, rule sets such as

```
#header {
   border-width: 1px;
   margin-width: 1px;
}
#footer {
   border-width: 5px;
   margin-width: 5px;
}
```

can be rewritten in CSSLang as

```
@def space-before-content ($width) {
   border-width: $width;
   margin-width: $width;
}
#header {
   @include space-before-content(1px);
}
#footer {
   @include space-before-content(5px);
}
```

5 Second example

```
@def div-style ($type, $width) {
   @if $type = header {
      color: red;
      border-width: $width + 10px;
   } @else if $type = body {
      color: blue;
      border-width: $width + 5px;
   } @else {
      color: green;
      border-width: $width;
   }
}
div.header {
   float: right;
   @include div-style(header, 1px);
}
div.footer {
   height: 5px;
   clear: both;
   @include div-style(footer, 1px);
}
```

is compiled to:

```
div.header {
   float: right;
   color: red;
   border-width: 11px;
}
div.footer {
   height: 5px;
   clear: both;
   color: green;
   border-width: 1px;
}
```

References

[1] Cascading Style Sheets Level 2 Revision 1 Specification, http://www.w3.org/TR/CSS2/

[2] Cascading Style Sheets, http://en.wikipedia.org/wiki/CSS

[3] LESS documentation, http://lesscss.org/docs

[4] Sass Language Reference, http://sass-lang.com/docs/yardoc/file.SASS_REFERENCE.html