

Buku Digital

Praktek Menggunakan **CSS Preprocessor**

E.R. Nurwijayadi

Halaman ini sengaja dikosongkan.

Buku Digital Bebas Unduh

Praktek Menggunakan CSS Preprocessor

Edisi 0.1: November 2020

Target edisi 0.1:

Materi sampai bab empat.

Dekorasi halaman masih sederhana.

Akan dilanjutkan di edisi 0.2.

Oleh: E.R. Nurwijayadi

epsi-rns.gitlab.io

Praktek Menggunakan CSS Preprocessor

Copyright © 2020 E.R. Nurwijayadi

Buku ini dilisensikan sebagai CC BY-NC-SA,
sehingga selain bebas dibaca buku ini juga bebas digandakan
dan diubah untuk keperluan apa saja selain komersial,
dengan syarat menyebutkan secara lengkap,
nama penulis dan pernyataan lisensi ini,
menggunakan lisensi yang sama.

Mengenai E.R. Nurwijayadi

Di media sosial, penulis biasa dipanggil epsi.

Blog:

epsi-rns.gitlab.io

epsi-rns.github.io

Prakata

Buku digital ini ditulis untuk pembaca yang telah memiliki dasar [HTML](#) maupun [CSS](#), namun ingin mendalami ilmu stylesheet dengan melakukan kustomisasi lebih lanjut. Pendekatan buku ini adalah langsung ke penerapan sehari-hari, supaya sohib pembaca memiliki bayangan, mengenai manfaat [CSS Preprocessor](#).

Buku ini dibagi menjadi beberapa bab.

1. Pengenalan HTML dan CSS
2. Menggunakan SASS
3. Menggunakan LESS
4. Menggunakan PostCSS
5. Penerapan di Bootstrap
6. Penerapan di Bulma
7. Custom CSS dengan Tailwind CSS

Bagian pertama berisi motivasi untuk belajar CSS Preprocessor, dan gambaran besar secara sistematis mengenai bagaimana pembelajaran ini harus ditempuh tahap demi tahap.

Bagian kedua dibagi antara bab kedua sampai keempat, berisi mengenai looping di dalam CSS Preprocessor. Penulis sengaja memilih looping sebagai contoh kasus, karena sudah mencakup beberapa hal mulai dari penetapan variabel, pairs dan hal lain yang terkait dasar pemrograman.

Bagian ketiga berisi penerapan CSS Preprocessor dalam kehidupan nyata. Mulai dari Bootstrap yang populer di masa lalu. Bulma yang ringan, supaya pemula tahu hal lain selain Bootstrap. Dan tentunya Tailwind CSS untuk tingkat yang lebih lanjut.

Setelah menguasai CSS sebagai dasar, pembaca dapat beranjak ke buku digital berikutnya.

Bab Pertama

Pengenalan HTML dan CSS

Panduan perjalanan bagi pemula untuk menjadi web developer.

Bagaimana memulai belajar web development? Jadi sohib ingin membikin sesuatu yang berarti dalam hidup. Sohib ingin memutuskan menjadi pengembang web. Bagaimana seorang pemula harus memulai?

Peran

Pertama, tetapkan dengan jelas peran sohib ada di mana.

1. Apakah sohib ingin membikin sesuatu? atau
2. Apakah sohib ingin orang lain untuk membikinkan sesuatu untuk sohib.

Saya ingin menjadi pembikin sesuatu.

Maka sohib harus belajar cara membikin sesuatu.

Saya ingin seseorang membikinkan sesuatu untuk saya.

Maka sohib harus memperkerjakan pegawai. Atau membeli aplikasi dari *software house*.

Peta Jalan

Suatu website yaitu roadmap.sh, dengan lugas membagi peran para pengembang web sebagai berikut:

- ▶ *Frontend*
- ▶ *Backend*
- ▶ *Devops*

Bila ketiga hal tersebut digabungkan maka dikenal istilah lain yaitu *full stack web developer*. Dari sudut pandang secara umum, perusahaan yang memberikan lowongan pekerjaan *fullstack web developer*, akan tampak memiliki anggaran kecil, walaupun ini bukan patokan.

Mobile

Perlu dipahami pula bahwa pengembangan aplikasi berupa *mobile development*, berbeda dengan *web development*.

Belajar Mandiri

Di Mana Belajar?

Bila sohib adalah pemula, maka mulailah dari beberapa situs berikut:

- ▶ [w3schools.com](https://www.w3schools.com)
- ▶ roadmap.sh

Yang biasa penulis lakukan adalah meminta pemula untuk membaca [w3schools.com](https://www.w3schools.com). Dan untuk yang bukan pemula saya meminta untuk menelusuri diagram-diagram yang terdapat di roadmap.sh.

Bilamana dibutuhkan masih ada beberapa lagi.

- ▶ css-tricks.com
- ▶ <https://google.github.io/styleguide/htmlcssguide.html>

Penulis sendiri menulis presentasi di blog.

- ▶ <https://epsi-rns.gitlab.io/frontend/2020/10/11/slides-concept-css/>

Bilamana sohib pembaca membutuhkan panduan langkah demi langkah, maka [youtube.com](https://www.youtube.com) adalah rujukan yang cukup bagus. Banyak kanal yang bagus yang dapat ditonton, baik yang berbahasa Inggris, maupun yang berbahasa Indonesia.

Keahlian Dasar Wajib di Komunitas

- ▶ Berdiskusilah di group, dan hindari jalur pribadi kalau tidak kenal.
- ▶ Tips: Buatlah catatan harian.

1. Bahasa Inggris wajib paham.
2. Mengetahui cara mencari di [google.com](https://www.google.com) atau stackoverflow.com.
3. Mampu membaca dokumentasi resmi dan buku panduan *manual*.
4. Mengetahui cara membikin tangkapan layar (*screenshot*) untuk group.
5. Mengetahui cara berkomunikasi dengan baik di dalam group.

Membaca, Menulis, Berbagi.

Pendekatan

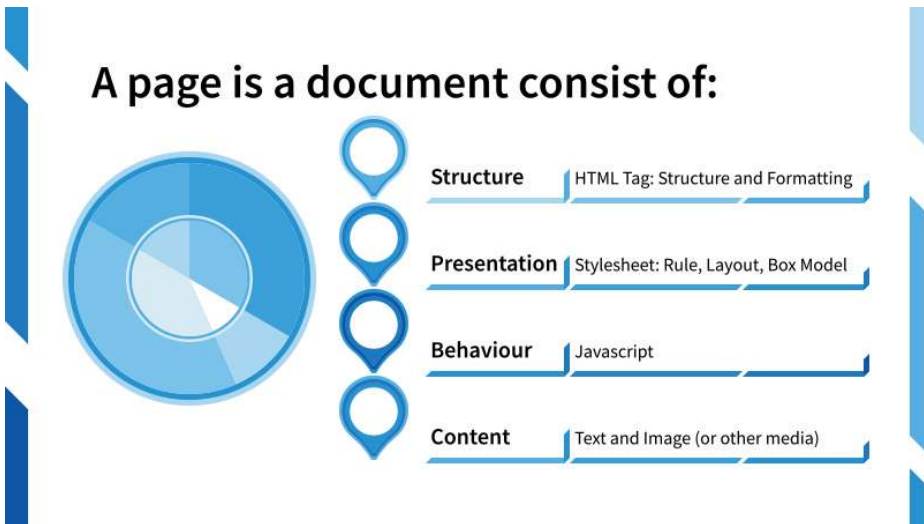
- *Stack* secara umum: `html+css+js`.

Terdiri Dari Apakah Halaman Web?

Suatu halaman web sebetulnya hanya berisi ini

1. *Structure + Presentation + Behaviour*
2. *Custom User Content*: Teks dan gambar (atau media lain).

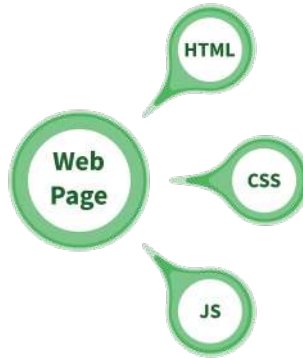
Yang pertama adalah, *stack* yang kita kenal sebagai `html+css+js`. Sedangkan



yang kedua adalah, isi apa saja yang dapat dimasukkan ke dalam berkas halaman tersebut.

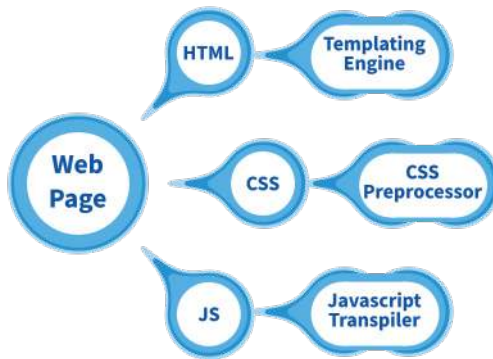
Secara Teknis

- Bagaimana cara menerangkan dari hal dasar ke era modern?



Ikatan Tersembunyi

- Tidak adanya panduan lengkap mengenai *template engine*.



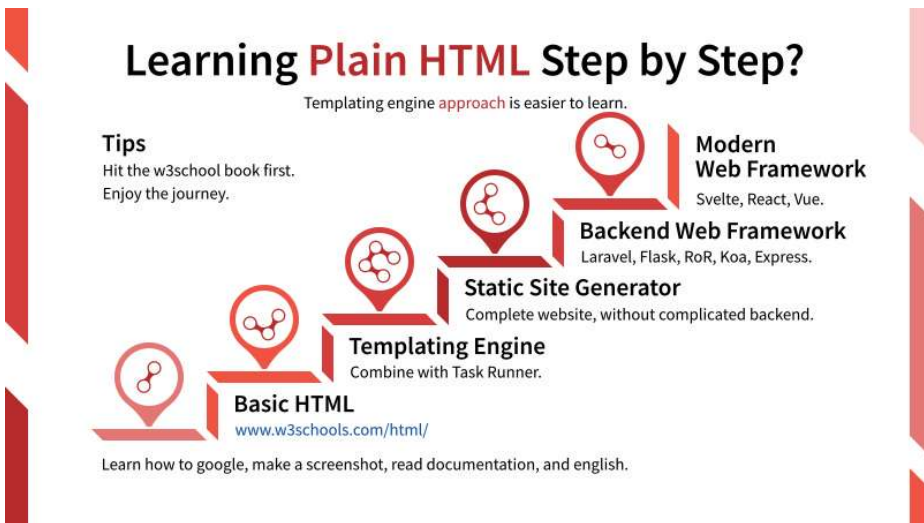
Ya, ada celah kekurangan, di dalam *tutorial* untuk *web development* pada umumnya.

HTML

- Lebih mudah untuk belajar HTML yang murni, dengan pendekatan template engine.

HTML5 adalah topik yang luas. HTML5 bukan hanya terkait HTML tag. Pertimbangkanlah untuk mempelajari dahulu hanya bagian dasarnya saja, kemudian langsung praktekkan untuk membuat berkas HTML, di dalam suatu proyek pribadi yang sederhana. *Blog* atau *portfolio*, adalah pilihan bagus untuk proyek pribadi yang sederhana. Ini disebabkan karena untuk membikin suatu *blog*, tidak ada beban untuk membuat *database* yang rumit, dan juga tidak ada kebutuhan untuk melakukan autentikasi melalui *login*.

Langkah-langkahnya adalah semudah berikut ini:



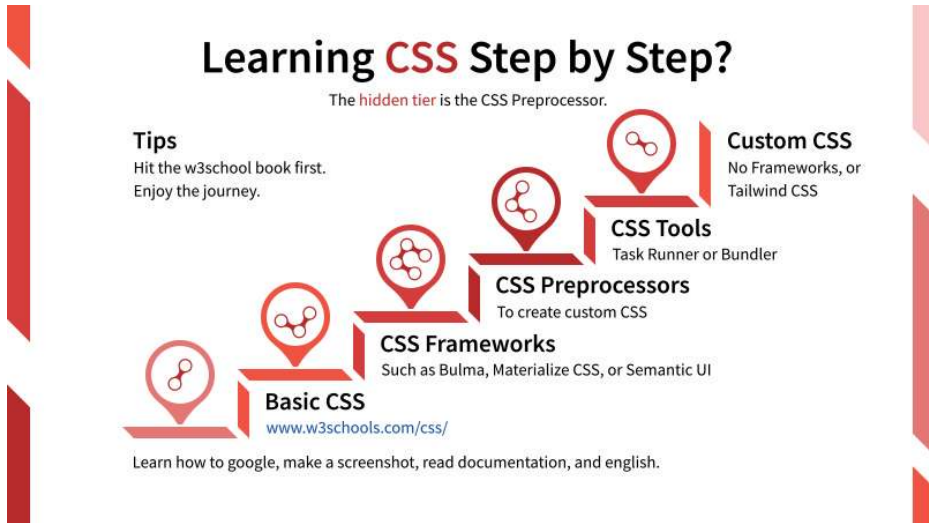
1. Dasar HTML: w3schools.com/html/.
2. *Template Engine*: Bersama dengan *Task Runner*.
3. *Static Site Generator*: Situs lengkap, tanpa *backend* yang rumit.
4. *Backend Web Framework*: [Laravel](#), [Flask](#), [RoR](#), [Koa](#), [Express](#).
5. *Modern Web Framework*: [Svelte](#), [React](#), [Vue](#).

Template engine adalah bidang backend.

Stylesheet

- Bagian yang tersembunyi adalah **CSS Preprocessors**.

Sohib dapat belajar menggunakan **CSS Framework**, lalu belajar menggunakan **CSS Preprocessor**, lalu kembali untuk melakukan kustomisasi di **CSS Framework**.



Langkah-langkah-nya adalah semudah berikut ini:

1. Dasar **CSS**: w3schools.com/css/
2. **CSS Frameworks**: Misalnya **Bulma**, **Materialize CSS**, atau **Semantic UI**.
3. **CSS Preprocessors**: Untuk membikin tambahan kustomisasi **CSS**
4. **CSS Tools**: **Task Runner**, ataupun **Bundler**.
5. **Custom CSS**: Tanpa **Framework**, atau dengan **Tailwind CSS**.

Stylesheet adalah bidang frontend.

Javascript

- ▶ tidak dibahas di sini.

Javascript adalah topik yang sangat luas, dan layak mendapatkan penjelasan yang khusus.

Content

Content ini dapat berisi macam-macam, misalnya untuk sistem informasi content dapat berupa *database*. Sedangkan di media sosial, *content* dapat berupa *chat*.

Untuk proyek sederhana, misalnya *blogging* ataupun pembikinan *portfolio*, maka sohib harus membikin *content* sendiri. Karena itu perlu ada keahlian tambahan, yaitu untuk mendapat teks yang tepat dan mengolah gambar dengan baik.

1. Teks

- ▶ Belajar menulis esai.
- ▶ Bahasa Inggris, ataupun bahasa Indonesia, yang tepat.

2. Gambar

- ▶ Mengambil gambar dengan *smartphone*.
- ▶ Membikin ilustrasi *raster*: belajar menggunakan *GIMP*.
- ▶ Membikin ilustrasi *vektor*: belajar menggunakan *Inkscape*.

Penutup

Pahami konsepnya, alih-alih hanya sekedar belajar cara memakai sesuatu.

Mari bersenang-senang dengan melakukan coding.

Halaman ini sengaja dikosongkan.

Bab Kedua

Menggunakan SASS

*Mulai dari yang mudah dan populer
Makan bubur mulai dari pinggirnya.*

Mari kita memulai dengan mempersiapkan lingkungan SASS, baru kemudian masuk ke urusan *coding*.

Mempersiapkan Lingkungan

Penulis sudah lama menjadi pengguna GNU/Linux, yang untuk selanjutnya saya sebut saja dengan kata Linux, karena tidak semua Linux memakai GNU. Yang lebih penting adalah pembaca paham maksudnya. Ternyata Linux sangat memudahkan *programmer* untuk menjadi nyaman dalam keseharian. Bagaimanapun penulis juga sadar, kalau kebanyakan pemula adalah pengguna Windows. Maka penulis mencoba sebisanya mempersiapkan lingkungan di Windows. Kebetulan penulis tidak pernah memakai Windows 10. Maka harap maklum kalau contoh yang diberikan adalah Windows 7.

Command Line Interface

Lingkungan apa yang perlu dipersiapkan? Yaitu membiasakan diri menggunakan CLI (*command line interface*) ataupun dikenal juga dengan istilah *terminal shell*. Kalau di Windows dapat memakai MS-DOS prompt (`cmd`), atau dengan `powershell`. Sedangkan di Linux pilihannya banyak karena sifatnya yang modular. Misalnya shell dapat menggunakan `bash`, atau `zsh`. Sedangkan *terminal*-nya dapat memakai `xfce4-terminal`, `urxvt`, atau `st` (*simple terminal*).

Lingkungan Windows 7

Dukungan *terminal* di Windows 7 masih sangat terbatas. Untungnya ada beberapa peralatan tambahan dari pihak ketiga yang dapat kita pakai:

- ▶ `cmdr`: *terminal* yang dengan tampilan *cantique*, sebagai pengganti `cmd` di Windows 7.
- ▶ `chocolatey`: *package manager* untuk Windows
- ▶ `schoop`: *installer* alternatif, selain dari `chocolatey`.

Setelah beberapa saat, beberapa aplikasi tambahan yang tersebut dapat membuat *Windows* nyaman dipakai lagi. Sebagai kombinasi dari *choco* and *cmd*, jadinya kita memiliki *package manager* di dalam *terminal* yang *cantique*. Sebagai pengguna *Linux*, saya merasa seolah berada di rumah lagi.

Choco Package Manager

Seperti yang telah saya sampaikan sebelumnya, sebetulnya ada dua alternatif yang fungsinya kurang lebih sama.

- ▶ chocolatey.org: *Package Manager* untuk *Windows*.
- ▶ scoop.sh: *Installer* untuk *Windows*.

Penulis pilihkan *choco* yang kegunaannya lebih luas.

Install Choco

Langsung saja ke situsnya:

- ▶ <https://chocolatey.org/docs/installation#more-install-options>

Sohib tentunya dapat melihat perintah panjang *command line* yang dibutuhkan di website, yaitu untuk dijalankan di *cmd*.

```
@ "%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" bla
bla bla ....
```

Sekarang bukalah *cmd* dengan *privilege* sebagai *administrator*, lalu salin-tempel (*copy-paste*) perintah tersebut. Jangan lupa untuk menekan tombol *enter* untuk menjalankan perintah tersebut.

```
Administrator: C:\Windows\System32\cmd.exe - "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

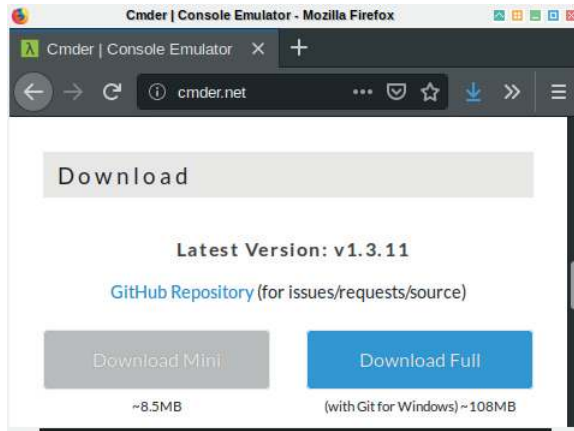
C:\Windows\system32>@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" & { $cmd = "Set-System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))"
Unable to set PowerShell to use TLS 1.2 and TLS 1.1 due to old .NET Framework in
to do one or more of the following: (1) upgrade to .NET Framework 4.5+ and Power
DownloadUrl prior to install or host the package internally). (3) use the Downlo
install options.
Getting latest version of the Chocolatey package for download.
Getting Chocolatey from https://chocolatey.org/api/v2/package/chocolatey/0.10.11
Extracting C:\Users\admin\AppData\Local\Temp\chocolatey\chocInstall\chocolatey.z
Installing chocolatey on this machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very likely you will need to close and reopen your shell
before you can use choco.
```

Cmder Console

Sekarang bagian yang membuat sohib nampak sakti, yaitu *terminal*.

► cmder.net.

Sebagaimana terlihat, ada dua pilihan:



Pilih benar satu dari dua pilihan tersebut, unduh, ekstrak, dan buka.

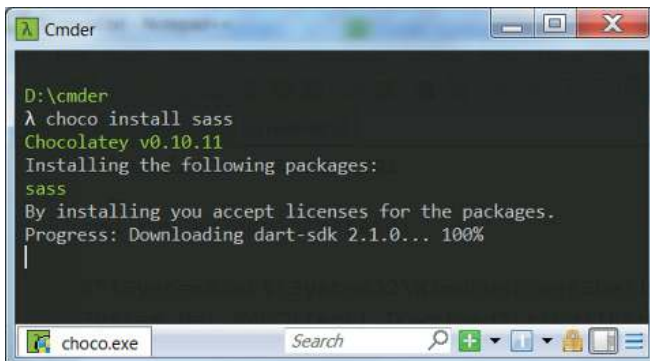
Berikutnya sohib dapat melakukan *pin* ke *taskbar* untuk supaya *cmder* mudah digunakan. Sohib juga dapat menggunakan *cmder* sebagai *administrator*, kapanpun dibutuhkan.

Di antara keduanya, penulis mendapatkan bahwa pilihan *cmder mini* sudah cukup. Tentunya ada rinciannya, namun buku ini bukan tempatnya.

Memasang SASS

Memasang SASS cukup mudah.

```
$ choco install sass
```



PRAKTEK MENGGUNAKAN CSS PREPROCESSOR

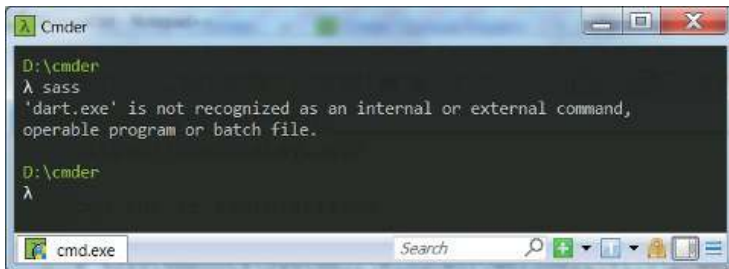
Pemulis menggunakan tanda `$` sebagai command prompt yang umum di [Linux](#). Di [Windows](#) biasanya memakai tampilan `C:>`.

Banyak implementasi [SASS](#). Penulis menggunakan [SASS](#) yang dibikin di atas bahasa [dart](#) yang tersedia di daftar paket di [chocolatey](#). Sebagai alternatif, sohib dapat menggunakan [node-sass](#) yang dapat dipasang melalui [npm](#).

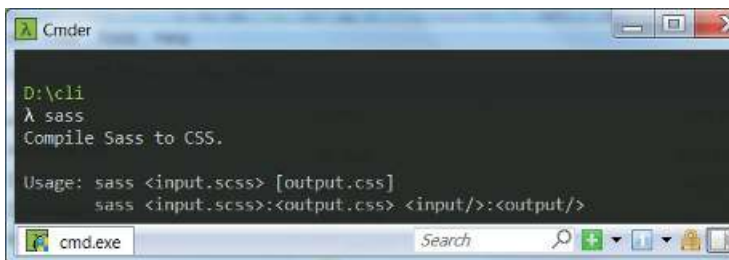
Sohib mungkin mengalami hal tersebut di bawah, tepat saat pertama kali menjalankan `dart`.

```
$ sass
```

```
'dart.exe' is not recognized as an internal or external command,  
operable program or batch file.
```



Cara penyelesaiannya adalah cukup dengan menutup *terminal*, dan buka [SASS](#) di terminal yang baru. Maka [SASS](#) akan berjalan dengan baik-baik saja.



Penulis tidak ingin berkulat terlalu lama di bagian pemasangan, supaya pembaca dapat segera melakukan praktek.

Lingkungan Linux

Seperti telah disampaikan sebelumnya. Ada beberapa implementasi SASS. Selayaknya budaya *RTFM* (*read the fine manual*) di komunitas *Linux*, sebelum kita masuk lebih dalam, ada baiknya kita membaca dahulu dokumentasi resminya.

- ▶ sass-lang.com.

Beberapa Pilihan

Ada beberapa pilihan untuk mengkompilasi SASS.

- ▶ *Compiler* Umum: *ruby-sass*, *dart-sass*, dan *node-sass*.
- ▶ *LibSass Wrapper*: *SassC*, *sass.cr*, *go-libsass*, *jsass*, *sass.js*, *lua-sass*, *libsass-net*, *node-sass*, *CSS: :Sass*, *SassPHP*, *libsass-python*, *sassc-ruby*, *sass_rs*, dan *Sass-Scala*.
- ▶ *Task Runner*: *Gulp*, *Grunt*. Dan juga *bundler*: *Rollup*, *Webpack*, and *Parcel*.

1: ruby-sass, yang kadaluarsa

Secara resmi, *Ruby Sass* sudah usang (*deprecated*). *Ruby Sass* ini sangat umum, dan ada berbagai tutorial tersedia di internet, maka tidak perlu dibahas lagi di sini. Bilamana ingin tahu, baca saja dokumentasi resminya.

- ▶ sass-lang.com/ruby-sass.

Mari kita lihat saja baris perintahnya di terminal supaya kita dapat membedakan antara satu implementasi dengan lainnya.

```
$ sass \
  --watch -I sass sass/themes/oriclonestatic/assets/css \
  --style compressed --sourcemap=none
```

Penulis menggunakan *backslash* (\) untuk memisahkan perintah panjang, sebagaimana umum dilakukan di *bash*.

2: dart-sass

Saat ini `ruby-sass` yang usang, sudah digantikan dengan `dart-sass`. Dokumentasi resmi juga ada:

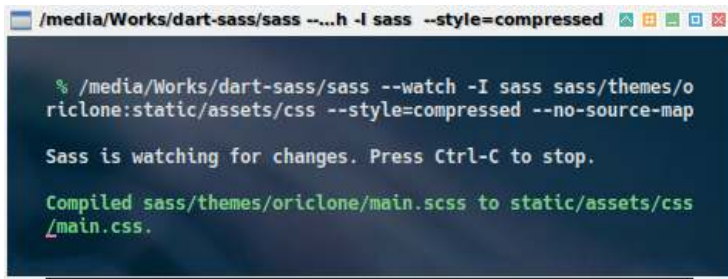
- ▶ sass-lang.com/dart-sass.

Memasang dart juga cukup mudah, langsung saja unduh, ekstrak dan langsung dapat dijalankan.

- ▶ <https://github.com/sass/dart-sass/releases/tag/1.29.0>

Mari kita lihat lagi baris perintahnya di terminal

```
$ /media/Works/dart-sass/sass --watch \
-I sass sass/themes/oriclone:static/assets/css \
--style=compressed --no-source-map
```



```
/media/Works/dart-sass/sass --...h -I sass --style=compressed

% /media/Works/dart-sass/sass --watch -I sass sass/themes/o
riclone:static/assets/css --style=compressed --no-source-map

Sass is watching for changes. Press Ctrl-C to stop.

Compiled sass/themes/oriclone/main.scss to static/assets/css
/main.css.
```

Demi kemudahan penggunaan *terminal* sehari-hari, maka sohib dapat menggunakan alias, yaitu di `.bashrc` ataupun di `.zshrc`.

```
alias dart-sass=/media/Works/dart-sass/sass
```

3: node-sass

Pemakaian `node-sass` adalah sangat umum, dan sudah banyak tutorial di internet. Seperti biasa, langsung saja jenguk dokumentasi resminya.

- ▶ sass-lang.com/install

Bilamana sohib adalah pemula di bidang `NodeJS`, maka sohib perlu memahami mengenai pengaturan lingkungan `NPM`.

- ▶ Tidak perlu `sudo`!

Saat menjadi pemula di bidang `NodeJS`, penulis berulang kali terbentur masalah terkait *permission*. Sampai penulis menemukan artikel ini

- ▶ <https://docs.npmjs.com/resolving-eacces-permissions-errors-when-installing-packages-globally>

Pada dasarnya, ini hanyalah mengatur *global path* untuk pengguna Linux tanpa *root privilege*.



```

epsi@andalan: ~
% mkdir ~/.npm-global
% npm config set prefix '~/.npm-global'
% export PATH=~/.npm-global/bin:$PATH
%
  
```

Jangan lupa menambahkan di `.bashrc` ataupun di `.zshrc`.

```
export PATH="$PATH:$HOME/.npm-global/bin"
```

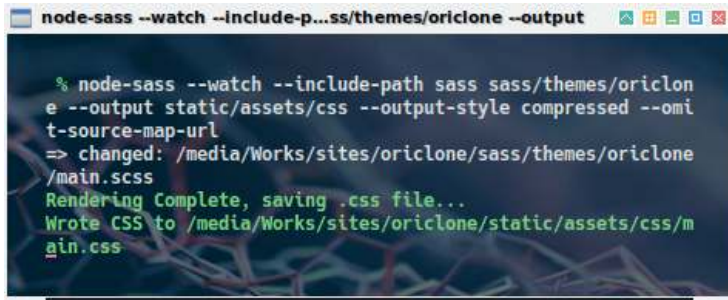
Dengan cara ini maka, pengguna biasa dapat memasang `node-sass` dengan bebas, tanpa membutuhkan perintah `sudo`.

```
$ npm install -g node-sass
```

Sekarang kita dapat menggunakan `node-sass` di dalam *terminal*, dengan pilihan *parameter* yang berbeda dengan `ruby-sass` sebagaimana berikut:

```

$ node-sass --watch \
  --include-path sass sass/themes/oriclone \
  --output static/assets/css \
  --output-style compressed \
  --omit-source-map-url
  
```



```
node-sass --watch --include-p...ss/themes/oriclone --output

% node-sass --watch --include-path sass/themes/oriclone
e --output static/assets/css --output-style compressed --omi
t-source-map-url
=> changed: /media/Works/sites/oriclone/sass/themes/oriclone
/main.scss
Rendering Complete, saving .css file...
Wrote CSS to /media/Works/sites/oriclone/static/assets/css/m
ain.css
```

Berulang kali mengetik di *terminal* rawan dari kesalahan, maka ada baiknya perintah tersebut kita bungkus dalam bentuk *script* di dalam berkas `packages.json`. Salah satu contoh yang bagus ada di dokumentasi Bulma.

- <https://bulma.io/documentation/customize/with-node-sass/>

Penulis ringkas saja dan ambil yang perlu yaitu:

```
"scripts": {
  "css-build": "node-sass
    --include-path sass sass/themes/oriclone
    --output static/assets/css",
  "css-watch": "npm run css-build --
    --watch --output-style compressed
    --omit-source-map-url",
  "start": "npm run css-watch"
},
```

Sekarang kita cukup melakukan perintah pendek ini.

```
$ npm run css-watch
```

Bagaimana pengaturan yang sesuai, tentunya terserah saja kepada pembaca, karena ini sangat tergantung keadaan setempat. Semua orang memiliki *use case* yang berbeda, karena itu kebutuhannya berbeda pula.

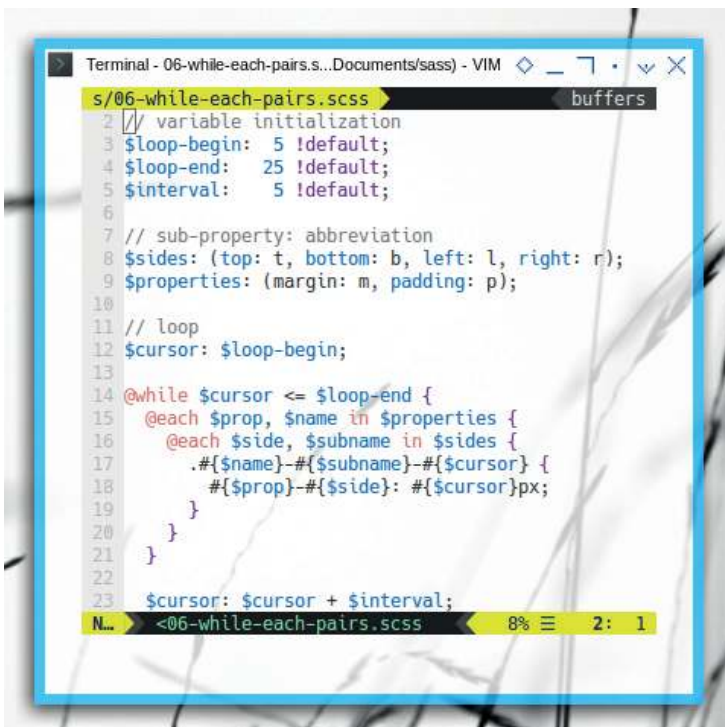
*langkah demi langkah, unjuk kemampuan dari SASS,
untuk menghasilkan margin class maupun padding class.*

Spacing Class dengan SCSS

- Tujuan: Menghasilkan spacing class dengan loop di SASS.

Pendahuluan

Pembikinan theme, membutuhkan *custom CSS*. Ini juga berlaku, bahkan saat menggunakan *CSS Framework*. Salah satu yang penulis butuhkan di masa lalu adalah membuat *spacing class* di *Bulma* maupun *Materialize CSS*. Walaupun akhirnya di tahun 2019, *Bulma 0.8* telah memiliki *spacing class* sendiri, namun penulis tetap membutuhkan untuk proyek lainnya. *Spacing class* ini dapat dibikin dengan menggunakan *SASS*.



```
Terminal - 06-while-each-pairs.s...Documents/sass) - VIM
s/06-while-each-pairs.scss buffers
2 // variable initialization
3 $loop-begin: 5 !default;
4 $loop-end: 25 !default;
5 $interval: 5 !default;
6
7 // sub-property: abbreviation
8 $sides: (top: t, bottom: b, left: l, right: r);
9 $properties: (margin: m, padding: p);
10
11 // loop
12 $cursor: $loop-begin;
13
14 @while $cursor <= $loop-end {
15   @each $prop, $name in $properties {
16     @each $side, $subname in $sides {
17       .#{$name}-#{$subname}-#{$cursor} {
18         #{$prop}-#{$side}: #{$cursor}px;
19       }
20     }
21   }
22   $cursor: $cursor + $interval;
23 }
N... <06-while-each-pairs.scss 8% 2: 1
```

Panduan ini akan menghemat banyak waktu *coding*, karena sudah diberikan contoh. Dan tentunya juga menghemat banyaknya ketikan, karena dengan otomasi maka tidak perlu mengetik *CSS* secara *manual*.

Bacaan Rujukan

Dasar Perulangan (*looping*)

- ▶ <http://thesassway.com/intermediate/if-for-each-while>

Sebagian besar `coding` di bab ini terilhami oleh:

- ▶ github.com/jmaczan/.../margin-padding.sass

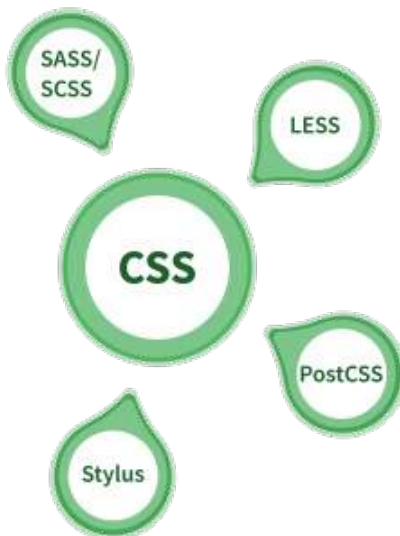
Bacaan lain:

- ▶ <https://alexwenzel.de/2017/707/sass-spacing-helper-classes>

Keluarga CSS Preprocessor

SASS memiliki dua format, yang pertama adalah **.scss** yang menggunakan kurung kurawal dan memisahkan dengan titik koma (*semicolon*), lalu yang kedua adalah **.sass**. yang mengandalkan indentasi. Untuk pembelajaran kita pakai jenis yang pertama, yang secara *syntax* mirip dengan **CSS**.

SASS/SCSS adalah salah satu alternatif diantara beberapa keluarga **CSS preprocessor**. Kebetulan **SASS/SCSS** ini adalah juga yang paling digemari karena dipakai di **Bootstrap**, **Bulma**, maupun **Materialize CSS**. Selanjutnya, mari langsung kita mulai saja. Praktek dengan contoh nyata.



1: @for

Bayangkan sohib membutuhkan beberapa kelas (*class*) yang berurut untuk menangani `margin`, sebagaimana CSS berikut:

```
.m-1 {
  margin: 1px;
}
```

Hasil jadi dari naskah di atas dapat dicapai dengan perulangan, tepatnya adalah dengan `loop constructor` sebagaimana berikut di bawah:

```
@for ... from ... through ... {
  ...
}
```

Penamaan *CSS Class* yang ingin dicapai adalah sebagai berikut:

```
.m-#{ $number }
```

Supaya nyaman maka perlu jelas, kita perlu menentukan nilai awal dari perulangan.

```
// variable initialization

$loop-begin: 1 !default;
$loop-stop: 3 !default;

// loop

@for $cursor from $loop-begin through $loop-stop {
  .m-#{ $cursor } {
    margin: #{ $cursor }px;
  }
}
```

Di dalam perulangan tersebut kita menjumpai interpolasi variabel yaitu `#{...}`, yang digunakan untuk meng-ekstrak suatu nilai ke dalam CSS.

- <https://sass-lang.com/documentation/interpolation>

Dengan `#{$cursor}` menangani isi dari perulangan, maka kita mendapatkan hasil sebagaimana berikut:

```
.m-1 {
  margin: 1px;
}
```

```
.m-2 {
  margin: 2px;
}
```

```
.m-3 {
  margin: 3px;
}
```

Yang menjadi keterbatasan dari `@for` adalah kita tidak dapat menggunakan *interval*. Kita dapat menghasilkan nilai berurut yaitu `[1, 2, 3, 4, 5]`, namun kita tidak dapat menghasilkan nilai berurut berupa `[5, 10, 15, 20, 25]`.

2: @while

Untuk menyelesaikan permasalahan nilai berurut `[5, 10, 15, 20, 25]` maka kita dapat menggunakan `@while`, sebagaimana di bawah ini:

```
$cursor: $loop-begin;

@while $cursor <= $loop-end {
  ...

  $cursor: $cursor + $interval;
}
```

Penamaan *CSS Class* yang ingin dicapai masih sama, yaitu sebagai berikut:

```
.m-#{ $number }
```

Maka kode sumber dari perulangan `@while`-nya adalah sebagaimana berikut:

```
// variable initialization
$loop-begin: 5 !default;
$loop-end: 25 !default;
$interval: 5 !default;

// loop
$cursor: $loop-begin;

@while $cursor <= $loop-end {
  .m-#{ $cursor } {
    margin: #{ $cursor }px;
  }
  $cursor: $cursor + $interval;
}
```

Sehingga kita mendapatkan hasil naskah CSS seperti tampak di bawah:

```
.m-5 {
  margin: 5px;
}

.m-10 {
  margin: 10px;
}

.m-15 {
  margin: 15px;
}

.m-20 {
  margin: 20px;
}

.m-25 {
  margin: 25px;
}
```

3: @each

Sebelum melanjutkan ke situasi yang lebih rumit, kerjakan contoh yang sederhana terlebih dahulu. Tantangannya adalah membikin *margin property* yang memiliki beberapa jenis varian, sebagaimana berikut:

- ▶ *margin*,
- ▶ *margin-top*,
- ▶ *margin-bottom*,
- ▶ *margin-left*,
- ▶ *margin-right*.

Untuk kesederhanaan, maka penulis tidak mengikutsertakan varian *margin*.

Penamaan *CSS Class* yang ingin dicapai berbeda, yaitu sebagai berikut:

```
.m-#{ $side }-5
```

Variabel SASS dapat menangani deklarasi *list*, yaitu:

```
$sides: (top, bottom, left, right);
```

Sedangkan daftar list tersebut dapat di-akses dengan *iterator*, yaitu *@each*:

```
@each ... in ... {
  ...
}
```

Mari kita lihat bagaimana cara menangani variabel *\$sides* ini:

```
// property
$sides: (top, bottom, left, right);

@each $side in $sides {
  .m-#{ $side }-5 {
    margin-#{ $side }: 5px;
  }
}
```

Sekarang kita dapat melihat naskah hasil jadinya sebagaimana di bawah:

```
.m-top-5 {  
  margin-top: 5px;  
}  
  
.m-bottom-5 {  
  margin-bottom: 5px;  
}  
  
.m-left-5 {  
  margin-left: 5px;  
}  
  
.m-right-5 {  
  margin-right: 5px;  
}
```

Ternyata naskah hasil jadinya masih tidak sesuai dengan harapan, karena yang kita inginkan adalah berbentuk `.m-t-5`, alih-alih berbentuk `.m-top-5`.

4: @each untuk Pasangan

SASS memiliki solusi untuk permasalahan ini.

Penamaan *CSS Class* yang ingin dicapai adalah menggunakan singkatan, yaitu sebagai berikut:

```
.m-#{ $abbreviation }-5.
```

Untungnya SASS memperbolehkan kita menulis pasangan *pairs* dalam daftar *list*. Sehingga kita mendapatkan *sub-property* sebagaimana berikut:

```
$sides: (top: t, bottom: b, left: l, right: r);
```

Mari kita tulis ulang kode sebelumnya menjadi sebagai berikut:

```
// property: abbreviation
$sides: (top: t, bottom: b, left: l, right: r);

@each $side, $name in $sides {
  .m-#{$name}-5 {
    margin-#{$side}: 5px;
  }
}
```

Sekarang kita mendapatkan naskah hasil jadi yang kita inginkan.

```
.m-t-5 {
  margin-top: 5px;
}

.m-b-5 {
  margin-bottom: 5px;
}

.m-l-5 {
  margin-left: 5px;
}

.m-r-5 {
  margin-right: 5px;
}
```

Perhatikan bahwa, penulis menggunakan `.m-t-5`, alih-alih `.mt-5`, untuk membedakan dengan spacing classes yang ada di salah satu *CSS Frameworks* yaitu [Bootstrap](#).

5: @each in @while

Pertimbangkan untuk meningkatkan kode untuk situasi yang lebih rumit. Kita membutuhkan varian dari kelas-kelas tadi untuk beberapa jarak angka yang berbeda secara berurut.

Penamaan *CSS Class* yang ingin dicapai adalah sebagai berikut:

```
.m-#{ $subname }-#{ $number }
```

Perhatikan kerangka kode-nya. Kita meletakkan *iterator*, yaitu `@each` ke dalam perulangan `@while`.

```
@while $cursor <= $loop-end {

    @each $side, $name in $sides {
        ...
    }

    $cursor: $cursor + $interval;
}
```

Maka kode sumber *SASS* selengkapnya adalah sebagai berikut:

```
// variable initialization
$loop-begin: 5 !default;
$loop-end: 25 !default;
$interval: 5 !default;

// sub-property: abbreviation
$sides: (top: t, bottom: b, left: l, right: r);

// loop
$cursor: $loop-begin;
```

Karena panjang, kita lanjutkan di halaman berikutnya.

```
@while $cursor <= $loop-end {

    @each $side, $name in $sides {
        .m-#{ $name }-#{ $cursor } {
            margin-#{ $side }: #{ $cursor }px;
        }
    }

    $cursor: $cursor + $interval;
}
```

Sekarang mulai dapat kita nikmati hasilnya secara perlahan yaitu beberapa kelas secara berurut:

```
.m-t-5 {
    margin-top: 5px;
}

.m-b-5 {
    margin-bottom: 5px;
}

.m-l-5 {
    margin-left: 5px;
}

.m-r-5 {
    margin-right: 5px;
}

.m-t-10 {
    margin-top: 10px;
}
```

```
.m-b-10 {  
  margin-bottom: 10px;  
}
```

```
.m-l-10 {  
  margin-left: 10px;  
}
```

```
.m-r-10 {  
  margin-right: 10px;  
}
```

...

...

```
.m-t-25 {  
  margin-top: 25px;  
}
```

```
.m-b-25 {  
  margin-bottom: 25px;  
}
```

```
.m-l-25 {  
  margin-left: 25px;  
}
```

```
.m-r-25 {  
  margin-right: 25px;  
}
```

Sengaja penulis potong di bagian tengah supaya tidak memenuhi isi buku.

6: Nested @while

Mari kita tambah kesulitannya.

Penamaan *CSS Class* yang ingin dicapai adalah sebagai berikut:

```
.#{$name}-#{$subname}-#{$number}
```

Kita tentukan dahulu deklarasi *list* untuk tiap-tiap *property* yang akan dihasilkan, yaitu:

- ▶ *Property*: *margin* dan *padding*.
- ▶ Masing-masing memiliki *sub-property*: *top*, *bottom*, *left*, *right*.

Tepatnya disajikan sebagai berikut:

```
$sides: (top: t, bottom: b, left: l, right: r);
$properties: (margin: m, padding: p);
```

Perhatikan kerangka kode-nya. Kita meletakkan dua *iterator* secara bersarang, yaitu *@each* ke dalam perulangan *@while*.

```
@while $cursor <= $loop-end {

    @each $prop, $name in $properties {
        @each $side, $subname in $sides {
            ...
        }
    }
    $cursor: $cursor + $interval;
}
```

Selanjutnya seperti biasa, yaitu kode sumber [SASS](#) selengkapnya sebagai berikut:

```
// variable initialization
$loop-begin: 5 !default;
$loop-end: 25 !default;
$interval: 5 !default;

// sub-property: abbreviation
$sides: (top: t, bottom: b, left: l, right: r);
$properties: (margin: m, padding: p);

// loop
$cursor: $loop-begin;

@while $cursor <= $loop-end {
  @each $prop, $name in $properties {
    @each $side, $subname in $sides {
      .#{$name}-#{$subname}-#{$cursor} {
        #{$prop}-#{$side}: #{$cursor}px;
      }
    }
  }
  $cursor: $cursor + $interval;
}
```

Hasilnya bukan hanya [margin](#), namun juga [padding](#).

```
.m-t-5 {
  margin-top: 5px;
}

.m-b-5 {
  margin-bottom: 5px;
}
```

```
.m-l-5 {  
    margin-left: 5px;  
}  
  
.m-r-5 {  
    margin-right: 5px;  
}  
  
.p-t-5 {  
    padding-top: 5px;  
}  
  
.p-b-5 {  
    padding-bottom: 5px;  
}  
  
.p-l-5 {  
    padding-left: 5px;  
}  
  
.p-r-5 {  
    padding-right: 5px;  
}  
  
.m-t-10 {  
    margin-top: 10px;  
}  
  
.m-b-10 {  
    margin-bottom: 10px;  
}  
  
...
```

Dan selanjutnya. Sengaja penulis potong lagi supaya tidak memenuhi isi buku.

Property Utama

Menjelang Final. Mari kita sempurnakan lagi, dengan `margin` dan `padding`, tanpa *sub-property*.

Penamaan *CSS Class* yang ingin dicapai adalah sebagai berikut:

```
.#{$name}-#{$number}
.#{$name}-#{$subname}-#{$number}
```

Masih dengan meletakkan dua *iterator* secara bersarang, yaitu `@each` ke dalam perulangan `@while`. Kerangkanya sedikit berubah.

```
@while $cursor <= $loop-end {

    @each $prop, $name in $properties {
        ...

        @each $side, $subname in $sides {
            ...
        }
    }

    $cursor: $cursor + $interval;
}
```

Selanjutnya seperti biasa, yaitu kode sumber *SASS* selengkapnya.

```
// variable initialization
$loop-begin: 5 !default;
$loop-end: 25 !default;
$interval: 5 !default;

// sub-property: abbreviation
$sides: (top: t, bottom: b, left: l, right: r);
$properties: (margin: m, padding: p);

// loop
$cursor: $loop-begin;
```

Kita lanjutkan di halaman berikutnya, karena kode-nya panjang.

```
@while $cursor <= $loop-end {
  @each $prop, $name in $properties {
    .#{$name}-#{$cursor} {
      #{$prop}: #{$cursor}px !important;
    }

    @each $side, $subname in $sides {
      .#{$name}-#{$subname}-#{$cursor} {
        #{$prop}-#{$side}: #{$cursor}px !important;
      }
    }
  }

  $cursor: $cursor + $interval;
}
```

Perhatikan, bahwa kita juga menerapkan aturan *rule*, yaitu `!important` ke dalam value dari CSS di masing-masing kelas. Sehingga hasilnya sebagai berikut:

```
.m-5 {
  margin: 5px !important;
}

.m-t-5 {
  margin-top: 5px !important;
}

...

.p-5 {
  padding: 5px !important;
}

...
```

Dan seterusnya. Kita hampir dapat menggunakan *custom spacing class* ini di suatu wesbite ataupun proyek pribadi.

Penerapan di Dunia Nyata

Untuk alasan praktis, ada beberapa hal yang perlu diperhatikan.

- ▶ Menggunakan format `.sass`, alih-alih `.scss`, supaya ada contoh bagi pembaca.
- ▶ Memulai dari angka `0` (nol), alih-alih dari `5` (lima), supaya tersedia pengaturan tanpa *margin* dan tanpa *padding*.
- ▶ Menggunakan `X`, dan `Y`, supaya mirip `Bootstrap`. Hal ini memudahkan saat migrasi dari `bootstrap` ke *custom class*.

Hasil naskah jadi, yang ingin kita dapatkan adalah sebagai berikut.

```
.m-y-5 {  
  margin-top: 5px !important;  
  margin-bottom: 5px !important;  
}  
  
.m-x-5 {  
  margin-left: 5px !important;  
  margin-right: 5px !important;  
}
```

Semakin lengkap belum tentu semakin baik. Kekurangan dari cara ini adalah, ukuran *stylesheet* menjadi gemuk, karena ukurannya besar.

Seperti telah disebutkan sebelumnya, `SASS` memperbolehkan pasangan di dalam deklarasi *list*, sehingga sekarang kita memiliki *sub-property* sebagaimana berikut:

```
$properties: (margin: m, padding: p)  
$sidesy: (top, bottom)  
$sidesx: (left, right)
```

Perhatikan, kalau kode-nya tidak memakai titik-koma dan tidak nantinya juga tidak memakai kurung kurawal. Yang dipakai adalah indentasi.

Kemudian mari kita lihat kerangkanya, perulangan yang dibikin menjadi juga membutuhkan perubahan.

```
// loop
$cursor: $loop-begin

@while $cursor <= $loop-end
  @each $prop, $name in $properties

    .#{$name}-y-#{ $cursor}
      @each $side in $sidesy
        #{$prop}-#{ $side}: #{ $cursor}px !important

    .#{$name}-x-#{ $cursor}
      @each $side in $sidesx
        #{$prop}-#{ $side}: #{ $cursor}px !important

$cursor: $cursor + $interval
```

Berikut kode sumber [SASS](#) selengkapnya, untuk keperluan sehari-hari.

```
// variable initialization
$loop-begin: 0 !default
$loop-end: 25 !default
$interval: 5 !default

// sub-property: abbreviation
$sides: (top: t, bottom: b, left: l, right: r)
$sidesy: (top, bottom)
$sidesx: (left, right)
$properties: (margin: m, padding: p)

// loop
$cursor: $loop-begin

@while $cursor <= $loop-end
  @each $prop, $name in $properties
    .#{$name}-#{$cursor}
      #{$prop}: #{$cursor}px !important

  @each $side, $subname in $sides
    .#{$name}-#{$subname}-#{$cursor}
      #{$prop}-#{$side}: #{$cursor}px !important

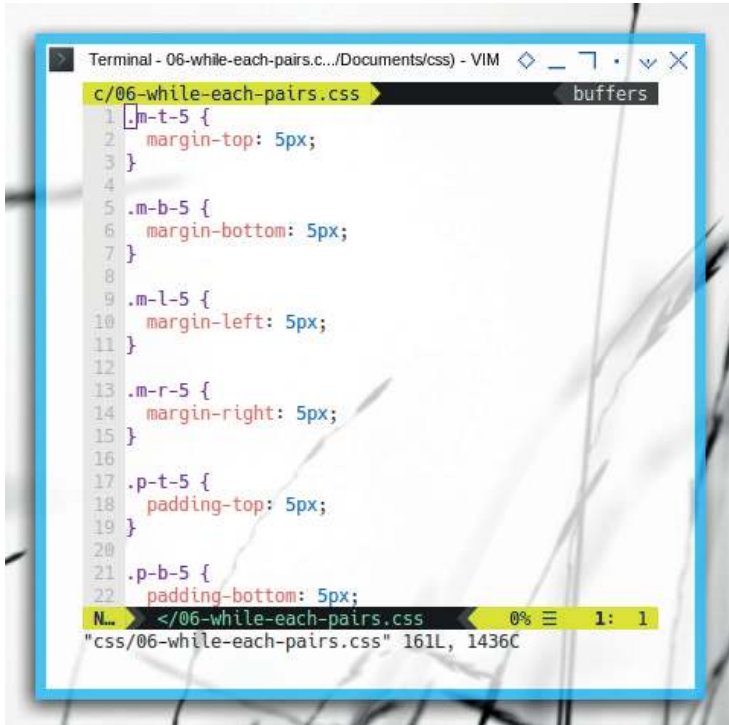
  .#{$name}-y-#{$cursor}
    @each $side in $sidesy
      #{$prop}-#{$side}: #{$cursor}px !important

  .#{$name}-x-#{$cursor}
    @each $side in $sidesx
      #{$prop}-#{$side}: #{$cursor}px !important

$cursor: $cursor + $interval
```


PRAKTEK MENGGUNAKAN CSS PREPROCESSOR

Akhirnya *custom spacing classes* ini siap dipakai. Mengenai hasilnya bagaimana, dapat dicoba sendiri sebagai latihan.

A screenshot of a VIM terminal window. The title bar reads "Terminal - 06-while-each-pairs.c.../Documents/css) - VIM". The editor shows a file named "c/06-while-each-pairs.css" with the following content:

```
1 .m-t-5 {  
2   margin-top: 5px;  
3 }  
4  
5 .m-b-5 {  
6   margin-bottom: 5px;  
7 }  
8  
9 .m-l-5 {  
10  margin-left: 5px;  
11 }  
12  
13 .m-r-5 {  
14  margin-right: 5px;  
15 }  
16  
17 .p-t-5 {  
18  padding-top: 5px;  
19 }  
20  
21 .p-b-5 {  
22  padding-bottom: 5px;  
23 }
```

The status bar at the bottom shows "N... </06-while-each-pairs.css 0% 1: 1" and a file path "css/06-while-each-pairs.css" with line 161L and column 1436C.

Bagaimana menurut sohib pembaca?