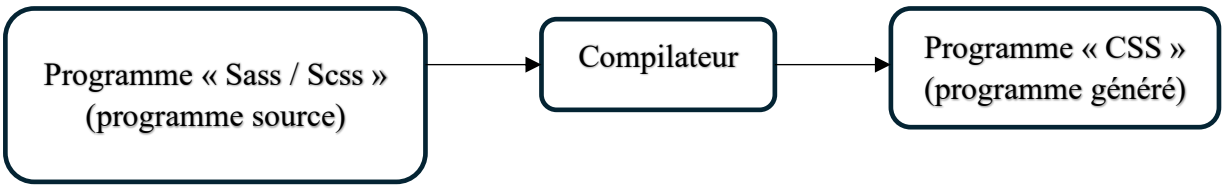
UNIVERSITÉ DE LORRAINE | IUT nancy Charlemagne Département Informatique

BUT Informatique 3ème année
« Compilation » - S5 RA-IL 01-3

Projet (de la semaine 40 à la semaine 43, soit 16 heures au total)

Objectif : Développer un compilateur (à l'aide d'ANTLR4) capable de traduire un source écrit en SASS / SCSS en CSS.



Deux possibilités pour la grammaire « Sass / Scss » à utiliser :

1. Utiliser « telle quelle » la vraie grammaire de Sass / Scss :
   - https://github.com/antlr/grammars-v4/tree/master/scss
2. À partir de cette grammaire, construire un sous-ensemble réduit contenant les éléments « Sass / Scss » - de base - décrits ci-dessous.
   - Si vous faites ce choix, car cela peut impliquer un travail important, ce sera pris en compte au niveau de l'évaluation.

Eléments de base « Sass / Scss » à considérer :

1. Variables.
2. Mixins (@mixin).
3. Include (@include).
4. For (@for) et each (@each), éventuellement imbriqués.
5. If (@if) et else (@else), éventuellement imbriqués.

Le code « sass / scss » de la page suivante illustre l'utilisation des « éléments de base » à considérer :

```
$primary-color: #333;
$margin: 10px;

@mixin border-radius($radius) {
  -webkit-border-radius: $radius;
  -moz-border-radius: $radius;
  border-radius: $radius;
}

button {
  color: $primary-color;
  @include border-radius(5px);

  @if $primary-color == #333 {
    background-color: white;
  } @else {
    background-color: black;
  }

  @for $i from 1 through 3 {
    margin-#{$i}: $margin * $i;
  }
}

@each $item in a, b, c {
  .#{$item}-class {
    background: $primary-color;
  }
}
```

Travail à faire : en vous inspirant des « visitors » des TP5 et TP6, programmer un « visitor » transformant le code « sass / scss » en css en tenant compte des « éléments de base » indiqués ci-dessus.

Étapes à suivre :

1. Création d'un nouveau projet IntelliJ IDEA.
2. Pour le groupe RA-IL-1 (S. Cruz-Lara), la grammaire du langage « Sass / Scss » se trouve dans le repository créé à partir du lien github classroom qui vous a été envoyé par e-mail (répertoire « src »). Pour le groupe RA-IL-2 (B. Mangeol), la grammaire pourra être récupérée directement sur : https://github.com/antlr/grammars-v4/tree/master/scss.
3. Génération de l'analyseur lexical (i.e., lexer) et de l'analyseur syntaxique (i.e., parser) par antlr4. Rappel : antlr4 génère également le code java de base des « visitors ». Ceux-ci sont la base de la « génération de code ».
4. Vérifiez, en utilisant plusieurs sources « sass / scss » que le parser arrive à produire un arbre de dérivation sans erreur.
5. Développez ensuite, un « visitor » appelé « ScssToCssGenerator » qui, en parcourant l'arbre de dérivation généré par le parser, va générer le code « css » correspondant au code « sass / scss » du programme source analysé.
    - Attention ! Si vous utilisez la grammaire complète de « sass / scss », ne programmez que les « visitors » concernant les « éléments de base ».
6. Testez votre « visitor » aussi extensivement que possible.
    - Le cas échéant, utilisez l'extension « Watch Sass » de Visual Studio Code, pour vérifier que le programme « css » généré correspond bien au code « sass / scss » analysé.

7. Le code « sass / scss » doit être lu à partir d'un fichier texte « input.scss » et le code « css » généré doit être écrit dans un fichier texte « output.css ».
8. Travail optionnel : vous pouvez aller au-delà des « éléments de base » (par exemple, prise en compte de « @import »). Si c'est le cas, ce sera pris en compte, cela va de soi, au niveau de l'évaluation.

Que la force soit avec vous 😉.