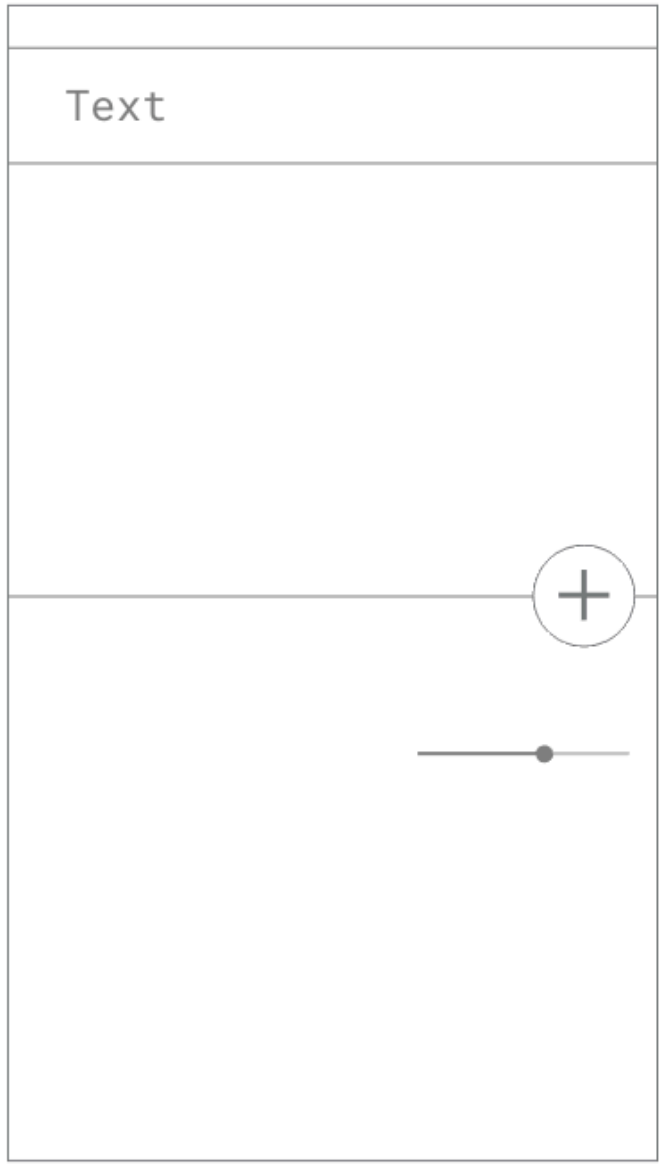
Formation Mobile Hybride

Ionic Cordova

Correction des exercices

```
git checkout step4
```

Theming



https://material.io/color

Theming

Fichier de définition globale des variables de theming

➔ src/theme/variables.scss

```
// Named Color Variables
// --------------------------------------------------
// Named colors makes it easy to reuse colors on various components.
// It's highly recommended to change the default colors
// to match your app's branding. Ionic uses a Sass map of
// colors so you can add, rename and remove colors as needed.
// The "primary" color is the only required color in the map.

$colors: (
  primary:    #488aff,
  secondary:  #32db64,
  danger:     #f53d3d,
  light:      #f4f4f4,
  dark:       #222,
  error:      #ff0000,
);
```

Theming

Principaux attributs de theming

text-left text-center text-right ….

text-uppercase text-lowercase text-capitalize

padding padding-top no-padding ….

margin margin-top no-margin ….

float float-left float-right …

http://ionicframework.com/docs/theming/css-utilities/

Theming

Les grids

```
<ion-grid>
  <ion-row>
    <ion-col col-12 col-sm>
      1 of 4
     </ion-col>
    <ion-col col-12 col-sm>
      2 of 4
    </ion-col>
    <ion-col col-12 col-sm>
      3 of 4
    </ion-col>
    <ion-col col-12 col-sm>
      4 of 4
    </ion-col>
  </ion-row>
</ion-grid>
```

Connaissez-vous ?

http://ionicframework.com/docs/theming/responsive-grid/#customizing-the-grid

Sass

Système qui étend les possibilités de **CSS**.

Permet d'être plus productif et plus facile à maintenir

Né dans la communauté **Ruby On Rail**

Sass
Fonctionnement

Écrire son script **Sass**

Le préprocesseur interprète le script **Sass**

Les fichiers **css** sont produits par le préprocesseur

Ca vous fait pensez à quoi?

Sass
Syntaxe

2 syntaxes

SCSS

Syntaxe classique qui utilise **brackets** et **semicolon**
Tout ce qui est valide **css** est valide **scss**
C'est la syntaxe la plus utilisée (extension .scss)

SASS

À la place d'utiliser les **brackets** et **semicolon**,
Utilise l'indentation pour séparer les blocs de codes
Il utilise l'extension .**sass**

Sass
Vue générale

En + du CSS, 7 choses à connaître

1. Variables
2. Nesting
3. Partials
4. Import
5. Mixins
6. Extend/Inheritance
7. Operators

Sass
Variables

Fichier SCSS:

```scss
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

Fichier CSS compilé:

```css
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```

➔ src/theme/variables.scss

Sass
Nesting

Evite de devoir récrire à chaque fois le(s) parent(s)

Fichier SCSS:

```scss
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

Fichier CSS compilé:

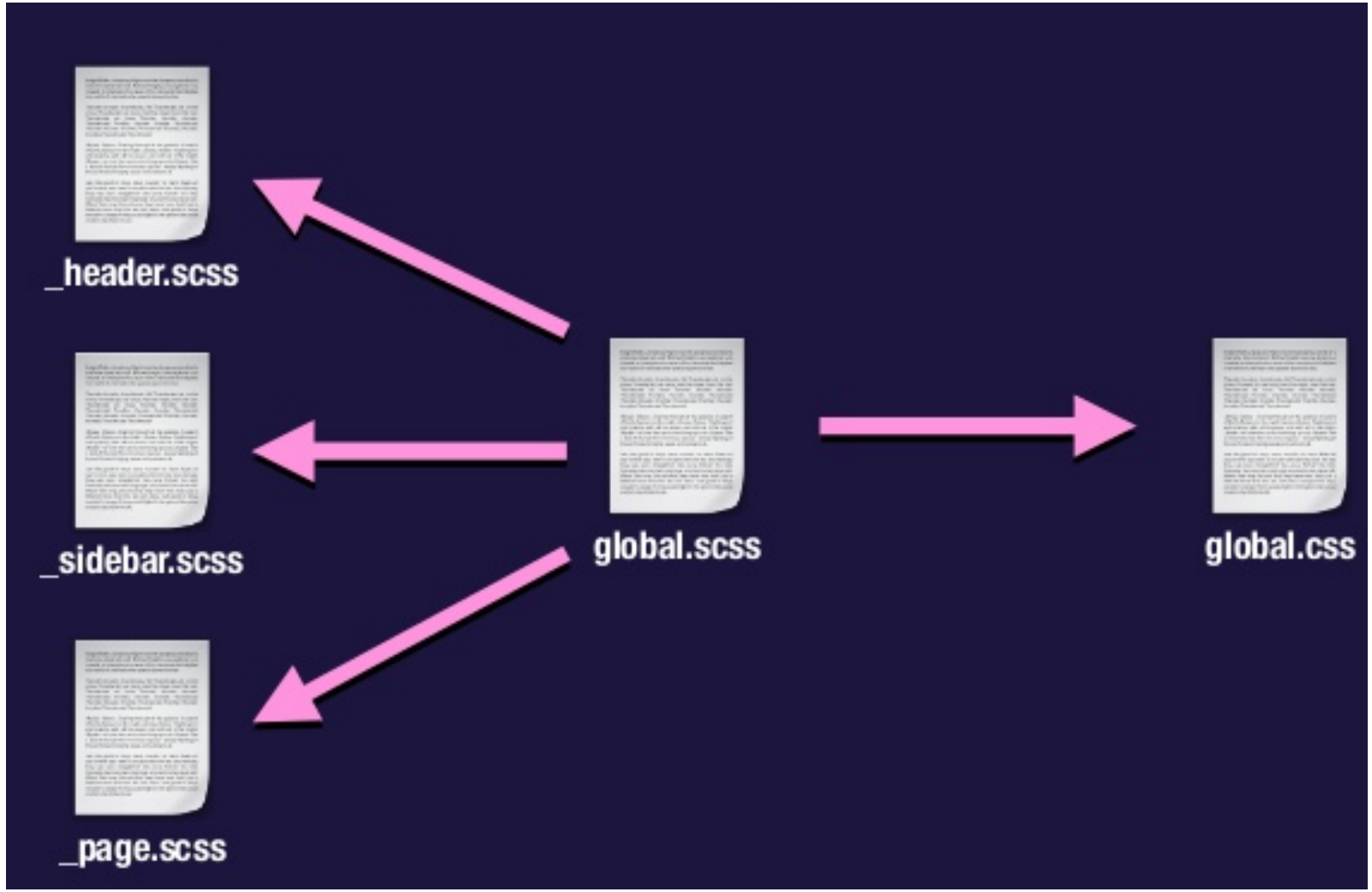
```css
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}

nav li {
  display: inline-block;
}

nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none;
}
```

Sass
Partials

Séparer les feuilles de style pour rendre le code plus modulable et organisé



Sass
Import

En lien avec « partials »: importe le code d'un fichier dans un autre

Fichier SCSS:

```scss
// _reset.scss

html,
body,
ul,
ol {
   margin: 0;
  padding: 0;
}
```

```scss
// base.scss

@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

Fichier CSS compilé:

```css
html, body, ul, ol {
  margin: 0;
  padding: 0;
}

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

N.B: On omet l'underscore et l'extension.scss

Sass
Mixins

Défini des « fonctions » qui sont réutilisés partout dans la css

Fichier SCSS:

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

.box { @include border-radius(10px); }
```

Fichier CSS compilé:

```css
.box {
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
  -ms-border-radius: 10px;
  border-radius: 10px;
}
```

Particulièrement utile pour les « préfixes vendeurs » CSS
(https://developer.mozilla.org/fr/docs/Glossaire/Pr%C3%A9fixe_Vendeur)

Sass
Extend / Inheritance

Partager un ensemble de propriétés CSS d'un sélecteur à l'autre

Fichier SCSS:

```scss
.message {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}

.success {
  @extend .message;
  border-color: green;
}

.error {
  @extend .message;
  border-color: red;
}

.warning {
  @extend .message;
  border-color: yellow;
}
```

Fichier CSS compilé:

```css
.message, .success, .error, .warning {
  border: 1px solid #cccccc;
  padding: 10px;
  color: #333;
}

.success {
  border-color: green;
}

.error {
  border-color: red;
}

.warning {
  border-color: yellow;
}
```

Sass

Pour les avancés

Il existe également des:

- functions
- conditions
- boucles
- Listes et maps
- …

En savoir +

➔ https://sass-guidelin.es/fr

Exercices

Consignes

Retranscrire la feuille de style CSS dans le dossier Exercice SASS en utilisant les 7 différentes techniques indiquées dans la partie « Learn Sass » du site officiel

Référence

http://sass-lang.com/

Paramètres dans la navigation

```
Let user = {id: 25, name: "john"};
this.navCtrl.push('testPage',
                  {userParams: user, other: 25});
```

```
export class MyClass{
 constructor(public navParams: NavParams){
   // userParams is an object we have in our nav-parameters
   this.navParams.get('userParams');
 }
}
```

Lifecycle Hooks

ionViewDidLoad

ionViewWillEnter

ionViewDidEnter

ionViewWillLeave

ionViewDidLeave

ionViewWillUnload

ionViewCanEnter

ionViewCanLeave

Retourne booléen ou promesse
Utilisé pour authguard

Utiliser des API existantes

Installer la librarie Gmap pour type script

```
npm install @types/google-maps --save
```

Index.html

```
<script
src"https://maps.googleapis.com/maps/api/js?key
=YOUR_API_KEY"></script>
```

API KEY: AIzaSyB16sGmIekuGIvYOfNoW9T44377IU2d2Es

Utiliser des API existantes

Créer le composant mapComponent

```
@Component({
  selector: 'map',
  template: '<div class="map_canvas" id="map_canvas_{{ mapID }}"></div>'
})
export class MapComponent {

  @Input() mapID: string = "";
  private map: google.maps.Map;

  public init(lat:number, long:number, zoom:number=12) {

    this.map = new google.maps.Map(document.getElementById(`map_canvas_${this.mapID}`), {
        center: new google.maps.LatLng(lat, long),
        zoom: zoom,
        mapTypeId: google.maps.MapTypeId.ROADMAP
     });

  }

}
```

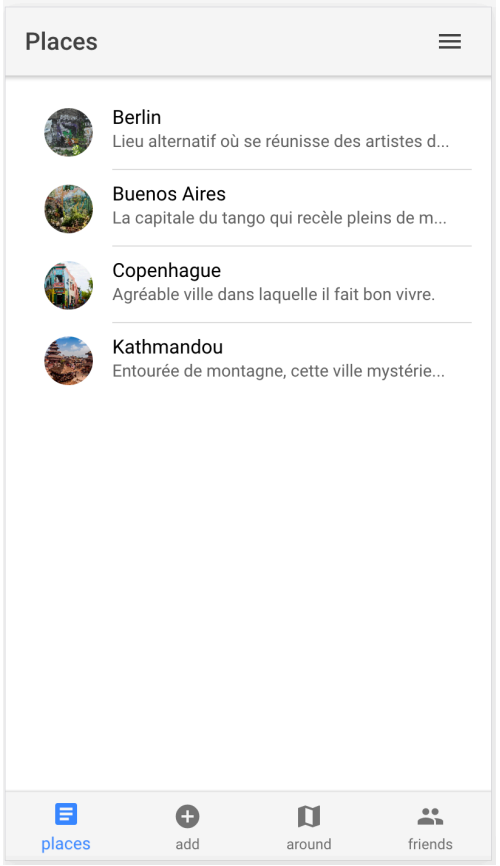
Utiliser des API existantes

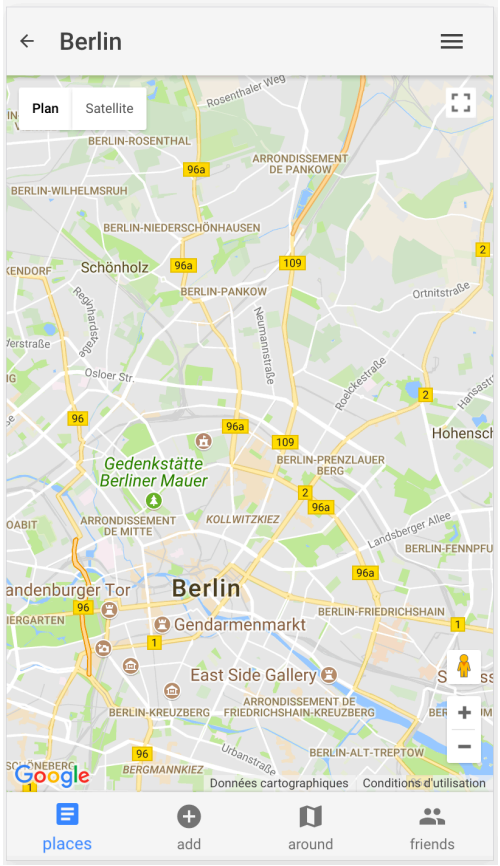
Exercices

Consignes

Créer un provider placeProvider qui va rechercher les différentes « places » du back-end, en mettant à jour endpointsProvider (cf. fichier postman).

Au sein de placesPage, implémenter le code qui permettra d'afficher les places dans places.html. La méthode selectItem() permet de diriger vers une nouvelle page placePage, qui affiche une carte pointant sur l'endroit choisi.






```
git checkout step5
```

Merci de votre attention