Développement mobile iOS 1

Mathieu Fancello - Head of Frontend @ GoodBarber
mathieu@goodbarber.com

TP1 - Application Météo

Le but de ce TP est de réaliser une application simple, permettant d'afficher le calendrier et les résultats d'une équipe de Foot.

Attention: Vous devez créer tous vos objets en lignes de code et ne pas utiliser l'interface graphique (xib ou storyboard).

1. **Réalisez uniquement par le code**, l'écran d'application suivant :

Vous trouverez les icônes nécessaires à cette URL :
http://mathieu.goodbarber.com/docs/weather_icons.zip

2. **Téléchargez le fichier JSON** suivant : https://mathieu.goodbarber.com/docs/ajaccio.json et **insérez le dans votre projet XCode**.

   Ce fichier JSON contient les données météo de la ville d'Ajaccio à une date donnée.

   Vous devrez maintenant parser ce JSON.

   a) Dans un premier temps, nous allons récupérer l'intégralité de ce fichier.

   Pour cela dans la méthode *viewDidLoad* de votre *ViewController,* insérez la ligne de code suivante :

```
let path = Bundle.main.path(forResource: [FILE_NAME], ofType: [FILE_TYPE])
```

   Remplacez ***[FILE_NAME]*** par le nom de votre fichier et ***[FILE_TYPE]*** par le type de votre fichier.

   Cette méthode permet de récupérer le chemin de votre fichier dans le file system. Affichez maintenant dans le debugger le résultat de cette méthode.

   b) Maintenant que vous avez le chemin de ce fichier, vous pouvez désormais récupérer son contenu.

   A la suite de la méthode précédente, rajoutez la suivante :

```
let data:NSData = try! NSData(contentsOfFile: [FILE_PATH]!)
```

   Remplacez ***[FILE_PATH]*** par le chemin que vous avez précédemment récupéré. Affichez maintenant le résultat cette méthode dans le debugger.

   Si de nombreux caractères étranges s'affichent, c'est que ça a fonctionné (la console est incapable d'afficher de la donnée brute, Si vous souhaitez la rendre lisible, vous devez la transformer en *String)*.

c) Il faut désormais parser cette data pour obtenir un objet utilisable dans votre code.

En iOS, lorsque vous parsez du JSON vous obtenez soit des tableaux d'objets, soit des dictionnaires du type « clés du JSON » -> objet associé.

Créez une variable d'instance dans votre *ViewController* de type *[String:Any]*.
Initialisez la à *[:]*.

A la suite de votre récupération de data, insérez le code suivant :

```
  do {
        [JSON_VARIABLE] = try JSONSerialization.jsonObject(with: [FILE_DATA]
as Data, options: .allowFragments) as! [String:Any]
  } catch let error as NSError {
        print(error)
  }
```

Remplacez **[FILE_DATA]** par la data que vous avez précédemment récupérée et **[JSON_VARIABLE]** par votre variable d'instance. Vous avez maintenant dans cette variable l'intégralité du JSON sous la forme Array/Dictionnaire.

d) Affichez dans le debugger les informations suivantes :

- Nom de la ville
- Température
- Temps qu'il fait

3. **Une fois ces informations récupérées**, le contenu de votre écran devra être dynamique et se baser sur ces informations.
    - Le nom de la ville en titre
    - La température

- L'icône du temps qu'il fait (les icônes sont toutes nommées en fonction de la clé que vous récupèrerez dans le fichier JSON.

- La couleur du background devra changer en fonction de l'heure de la journée
  - Gris la nuit (23h à 5h)
  - Bleu le matin (5h à 9h)
  - Vert la journée (9h à 15h)
  - Jaune en fin d'après-midi (15h à 19h)
  - Orange le soir (19h à 23h)

Vous pouvez tester en changeant l'heure du simulateur

4. **Vous devrez maintenant non plus parser le JSON** situé dans votre en projet en local. Mais le télécharger à distance depuis cette URL : https://api.openweathermap.org/data/2.5/weather?q=Ajaccio&units=metric&appid=b8c0162f208b810fd4c2e82e370a98a4

   Vous devrez aussi, à chaque fois que l'utilisateur clique sur le bouton refresh, relancez cette requête, et mettre à jour votre contenu.

   Voici comment télécharger un fichier JSON distant :

```
let data:NSData = try! NSData(contentsOf: URL(string:[URL])!)
```

5. **Récupérez les coordonnées GPS courantes du device**. Pour se faire:

   a) Ajoutez le framework *CoreLocation.framework* à votre projet.

   b) Importez ce framework dans votre classe *ViewController*.

   c) Vous devrez déclarer une variable d'instance *locationManager* de type *CLLocationManager.*

   d) Dans la méthode viewDidLoad, instanciez votre *locationManager.*

e) Ajoutez la ligne de code suivante :

```
locationManager.delegate = self
```

Elle permet de vous abonner aux évènements du locationManager

f) La ligne de code suivante, vous permet de demander la permission à l'utilisateur d'utiliser la géolocalisation du device :

```
locationManager.requestWhenInUseAuthorization()
```

g) Vous pouvez maintenant demander la géolocalisation avec la méthode suivante :

```
locationManager.startUpdatingLocation()
```

h) Implémentez la méthode suivante :

```
  func locationManager(_ manager:CLLocationManager, didUpdateLocations
locations: [CLLocation]) {

}
```

Elle sera automatiquement appelée (évènement) lorsque la géolocalisation sera trouvée ou mise à jour. A vous de trouver dans la documentation des paramètres de cette méthode, ou se trouve la géolocalisation

6. **Passez maintenant ces coordonnées GPS en paramètre de la requête de la façon suivante** :

https://api.openweathermap.org/data/2.5/weather?lat=8.738635&lon=41.919229&appid=b8c0162f208b810fd4c2e82e370a98a4

Vous pouvez modifier la géolocalisation du simulateur pour effectuer vos tests.

Pour aller plus loin…

A chaque fois que l'utilisateur secoue le device, lancer un refresh. *Vous pouvez simuler une secousse sur le simulateur.*