02-ViewModel

1 - Créer un nouveau projet dans le dossier Part1

2 - Add dependencies to the project

Dans le fichier `./app/src/build.gradle`, ajouter :

```
    def lifecycle_version = "1.1.1"
    implementation "android.arch.lifecycle:extensions:$lifecycle_version"
    annotationProcessor "android.arch.lifecycle:compiler:$lifecycle_version"

    def kotlin_version = "1.5.30"
    implementation "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    implementation "org.jetbrains.kotlin:kotlin-reflect:$kotlin_version"
```

Les trois dernières lignes fixent la version de Kotlin

On n'oublie pas de sync le gradle

1 - ajout d'un textview

Dans le TextView de activity_main.xml, on ajoute :

```
android:text="Number"
android:textSize="25dp"
```

On crée une classe DataGenerator :

```
package iutinfo.lp.devmob.viewmodeldemo

import android.util.Log
import java.util.*

class DataGenerator {
    private lateinit var myRandomNumber : String

    fun getNumber () : String {
        Log.i(TAG, "Get Number")
        if (!::myRandomNumber.isInitialized) {
            this.createNumber()
        }
```

```
        return  myRandomNumber
    }

    private fun createNumber() {
        Log.i(TAG, "Create new number")
        val random = Random()
        myRandomNumber = "Number : " + (random.nextInt(10-1) + 1)
    }

    companion object {
        private val TAG = DataGenerator::class.simpleName
    }
}
```

Dans les plugins du build.gradle (du module), ajouter :

```
id 'kotlin-android-extensions'
```

C'est une extension qui permet de synthétiser les éléments du XML dans notre code Kotlin (on peut récupérer direct un élément dans le code Kotlin avec son id)

On retourne dans le MainActivity et on ajoute :

```
import kotlinx.android.synthetic.main.activity_main.*
```

On modifie la classe MainActivity:

```
class MainActivity : AppCompatActivity() {

    lateinit var data : DataGenerator

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        data = DataGenerator()
        val myRandomNumber = data.getNumber()
        tvNumber.text = myRandomNumber
        Log.i(TAG, "Random Number Set")
    }

    companion object {
        private val TAG = "Demo : " + DataGenerator::class.simpleName
    }
}
```

Lorsque l'on tourne le téléphone, on se rend compte que le numéro change car l'activité se regénére à chaque rotation

On copie/colle dans le dossier Part2 notre projet

Part 2

On vérifie que toutes les dépendances sont précisés dans le build.gradle du module App

On transforme notre classe DataGenerator en ViewModel :

```
class DataGenerator : ViewModel() {
    ...
}
```

Le ViewModel est un pattern singleton donc on ne peut que hériter d'une instance (d'où le ())

Attacher un LifecycleOwner à notre ViewModel

Dans le MainActivity, on change l'instanciation de DataGenerator :

```
 //data = DataGenerator()
data = ViewModelProviders.of(this).get(DataGenerator::class.java)
```

Maintenant, lorsque l'on tourne l'écran, la valeur number ne change pas

Dans la classe DataGenerator, on ajoute :

```
override fun onCleared() {
        Log.i(TAG, "ViewModel destroyed")
}
```