FOR KOTLIN

MARC POPPLETON

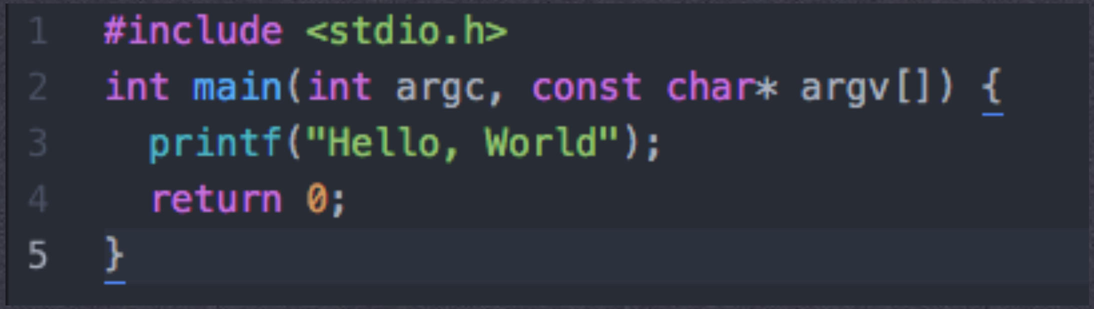
Code d'Armor

AU DÉBUT ÉTAIT LE VERBE

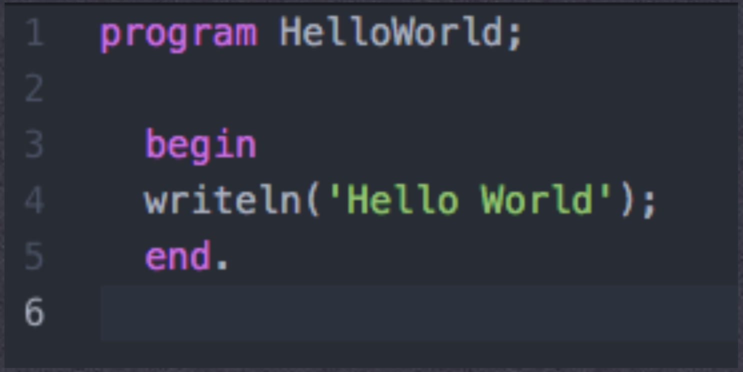
Code d'Armor

IMPÉRATIF

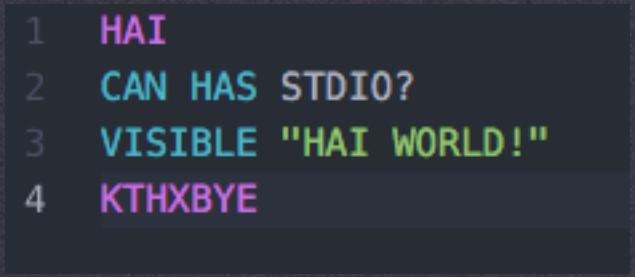
C
PASCAL
PERL
LOLCODE
…



```c
#include <stdio.h>
int main(int argc, const char* argv[]) {
  printf("Hello, World");
  return 0;
}
```



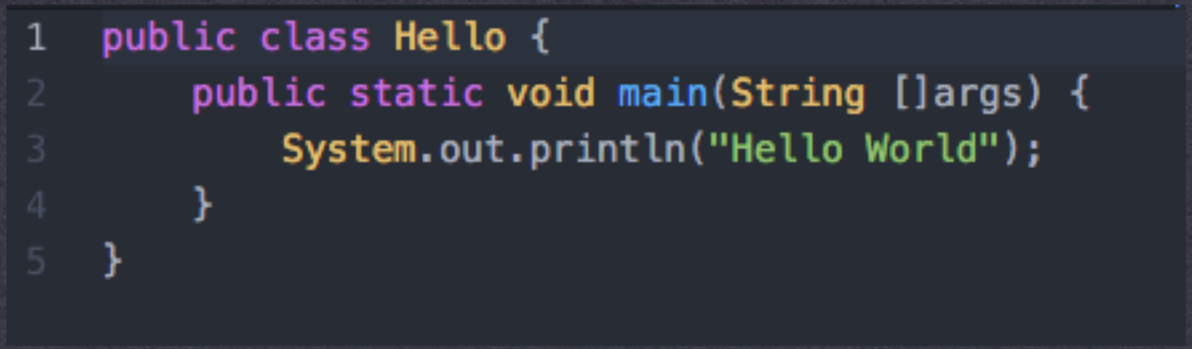
```pascal
program HelloWorld;

  begin
  writeln('Hello World');
  end.

```



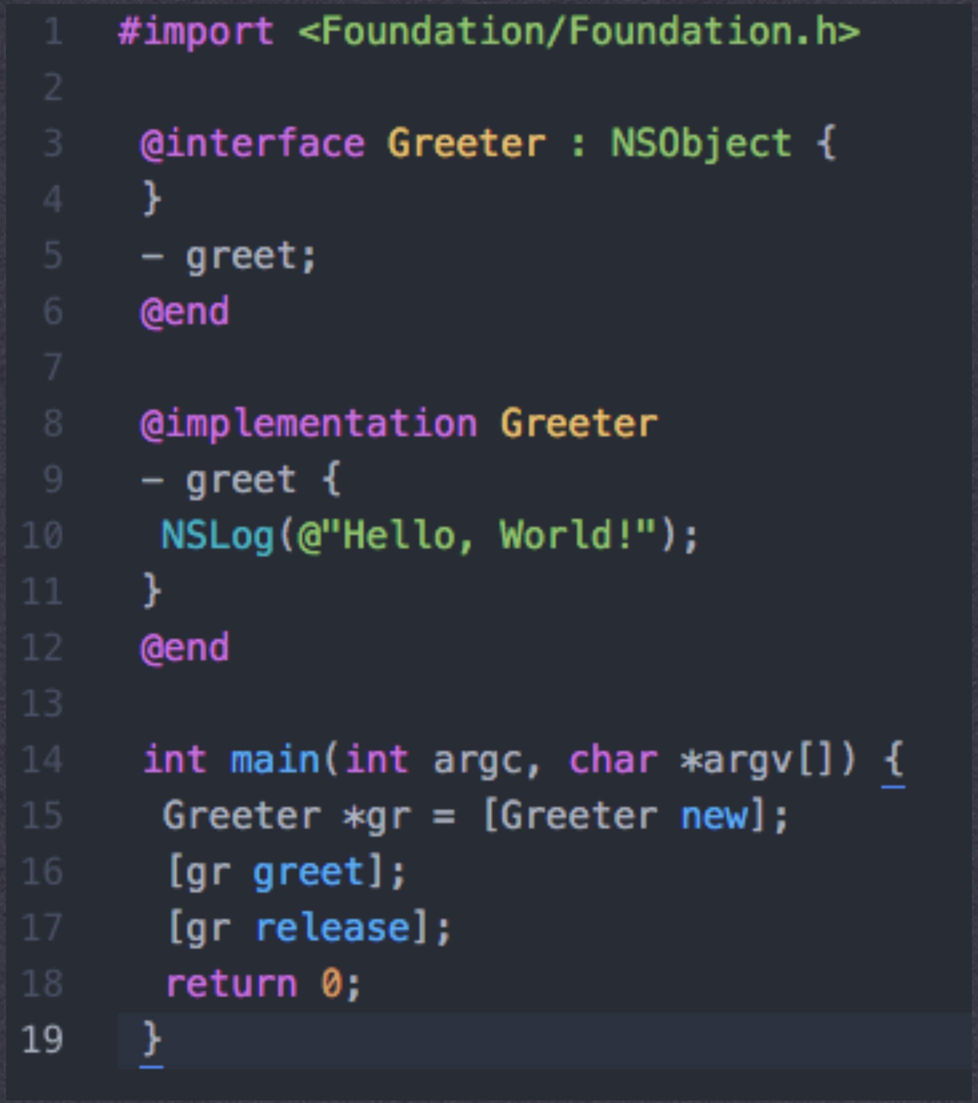
```
HAI
CAN HAS STDIO?
VISIBLE "HAI WORLD!"
KTHXBYE
```

ORIENTÉ OBJET

ADA
C++
JAVA
OBJECTIVE-C
PYTHON
JAVASCRIPT
RUBY
…



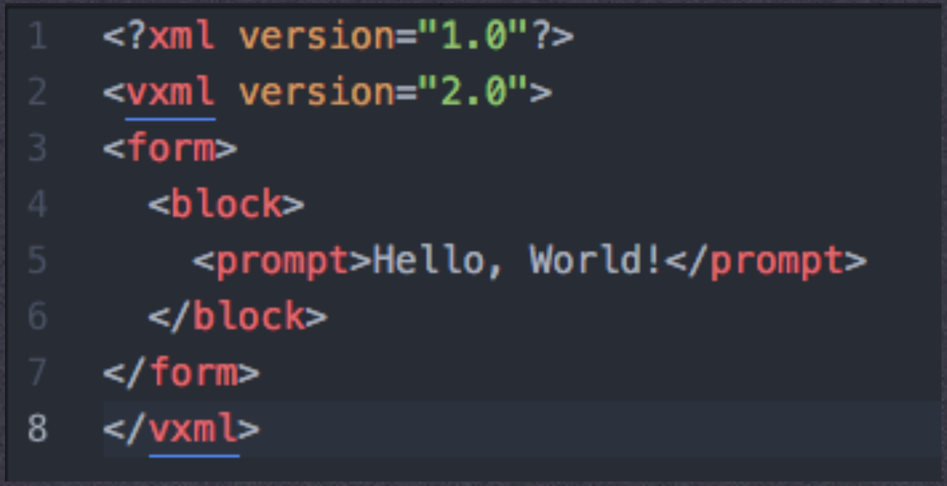
```java
public class Hello {
    public static void main(String []args) {
        System.out.println("Hello World");
    }
}
```



```objc
#import <Foundation/Foundation.h>

 @interface Greeter : NSObject {
 }
 - greet;
 @end

 @implementation Greeter
 - greet {
  NSLog(@"Hello, World!");
 }
 @end

 int main(int argc, char *argv[]) {
  Greeter *gr = [Greeter new];
  [gr greet];
  [gr release];
  return 0;
 }
```

DESCRIPTIF

HTML
XML
...

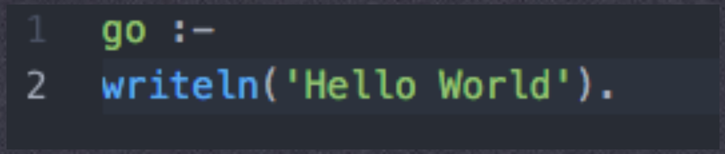


```
<?xml version="1.0"?>
<vxml version="2.0">
<form>
  <block>
    <prompt>Hello, World!</prompt>
  </block>
</form>
</vxml>
```

LOGIQUE

PROLOG

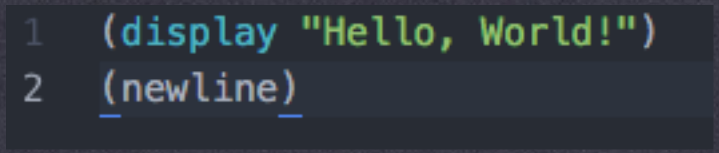
CLIPS



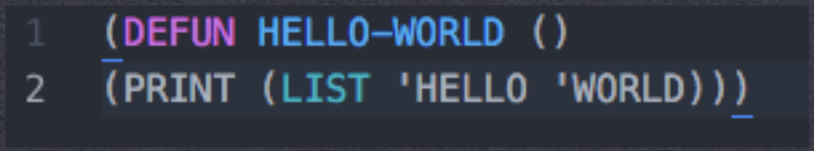
```
go :-
writeln('Hello World').
```

FONCTIONNEL

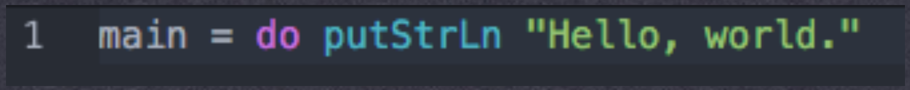
SCHEME
LISP
HASKELL
SCALA



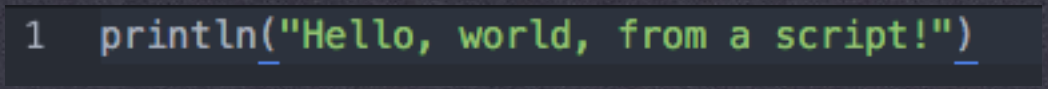
```scheme
(display "Hello, World!")
(newline)
```



```lisp
(DEFUN HELLO-WORLD ()
(PRINT (LIST 'HELLO 'WORLD)))
```



```haskell
main = do putStrLn "Hello, world."
```



```scala
println("Hello, world, from a script!")
```

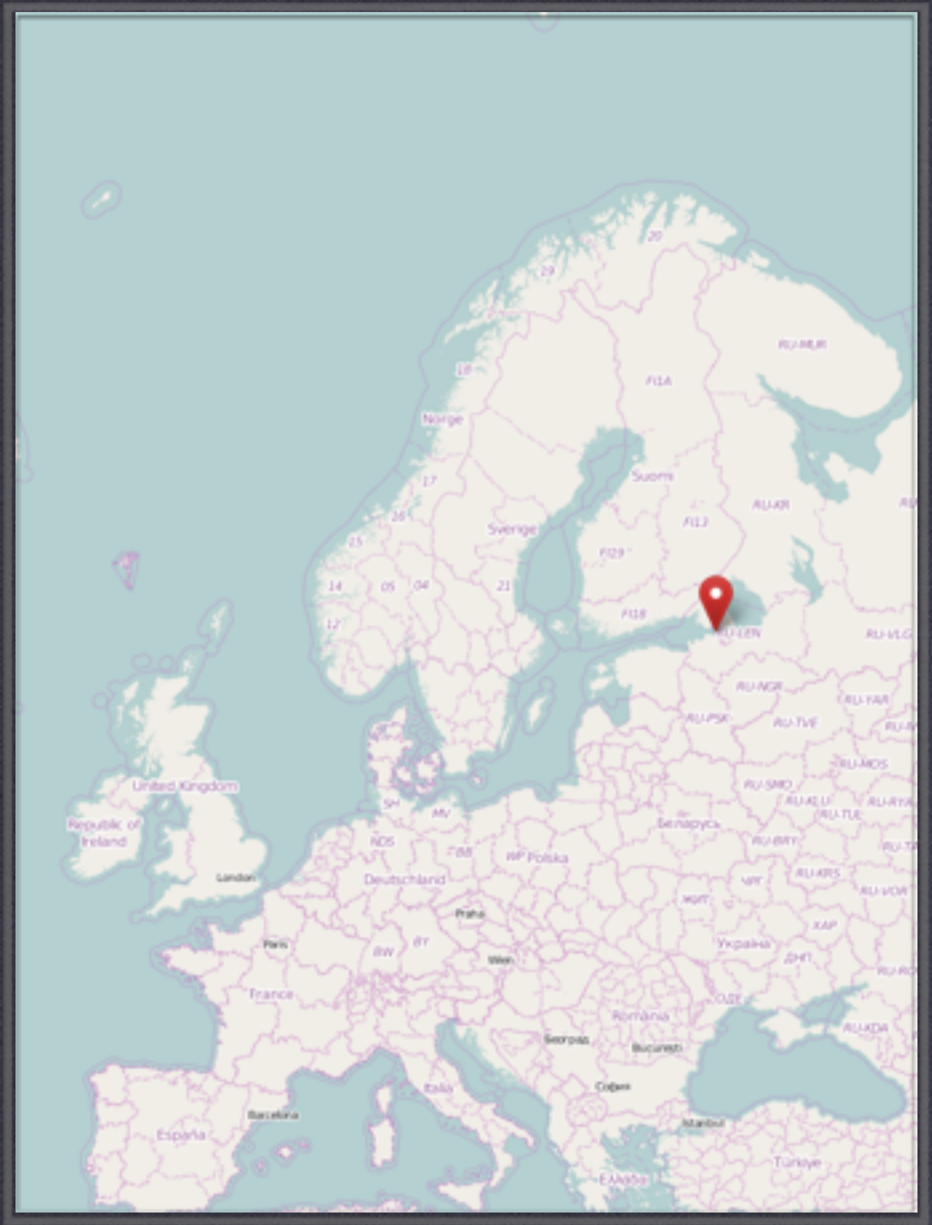
Code d'Armor

“C’est bien mais pas top”

–verbatim au sujet du commissaire Bialès

Kotlin

- pour la JVM et Javascript
- orienté objet
- fonctionnel
- statiquement typé
- open source Apache 2



source : openstreetmap.org

Code d'Armor

Kotlin, la promesse

- Orienté objet et fonctionnel
- La puissance de Scala
- La vitesse de compilation de Java
- Interopérabilité avec Java

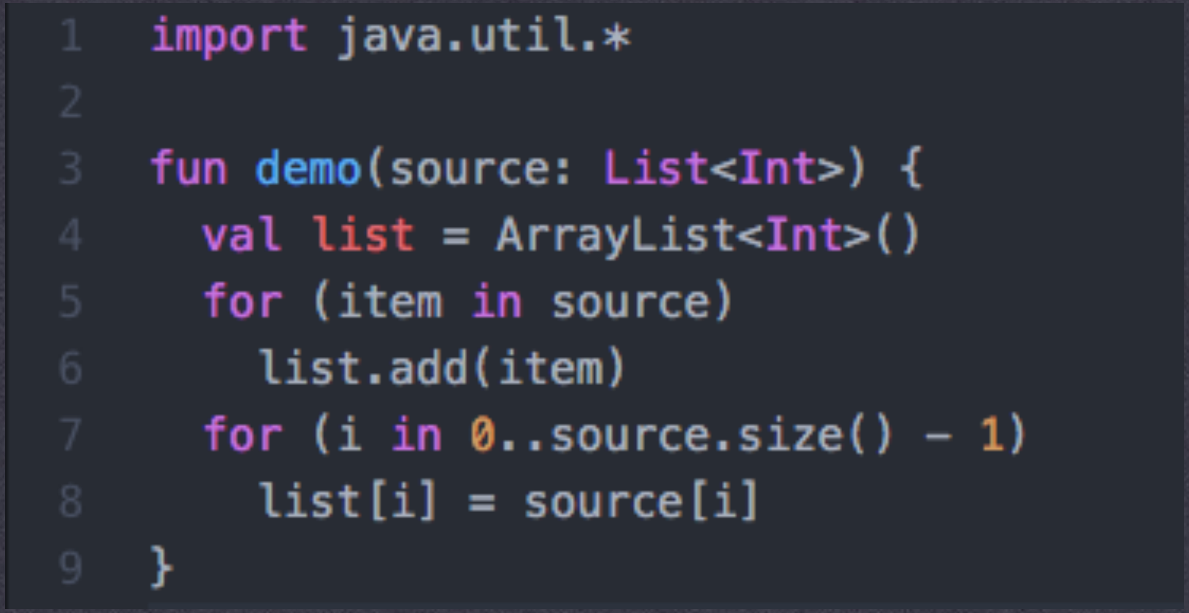
Code d'Armor

LES PRINCIPES DE KOTLIN

Code d'Armor

interopérabilité

- Kotlin peut utiliser des libs Java
- Java peut utiliser des idioms Kotlin
- Le runtime ne comporte que les fonctionnalités ajoutées par Kotlin



```kotlin
import java.util.*

fun demo(source: List<Int>) {
  val list = ArrayList<Int>()
  for (item in source)
    list.add(item)
  for (i in 0..source.size() - 1)
    list[i] = source[i]
}
```

Arguments

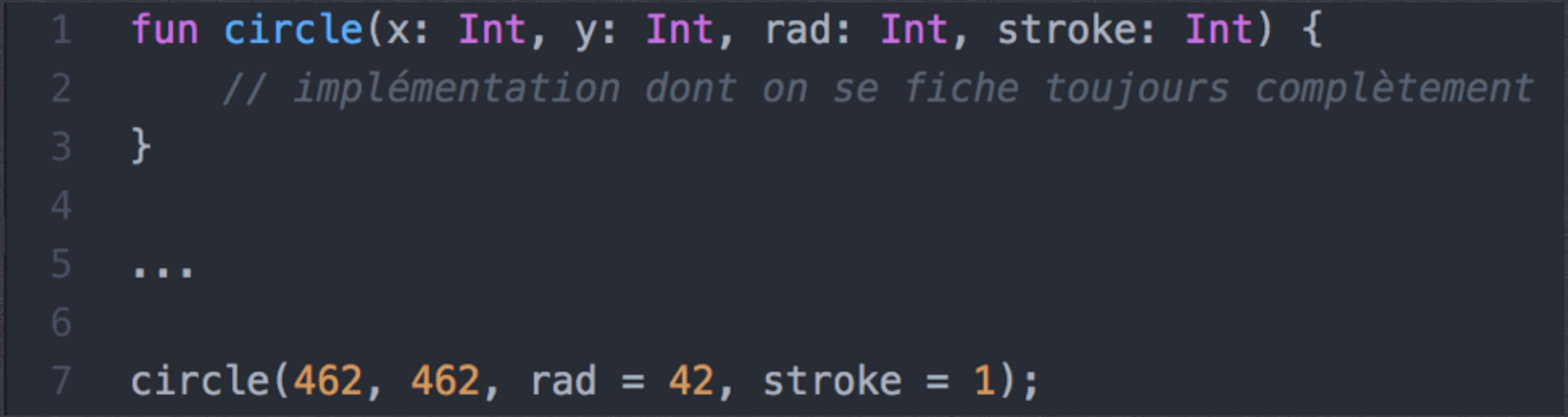
- nommés
- optionnels

Argument nommés

```
void circle(int x, int y, int rad, int stroke) {
    // implémentation dont on se fiche complètement
}

...

circle(462,462,42,1);
```

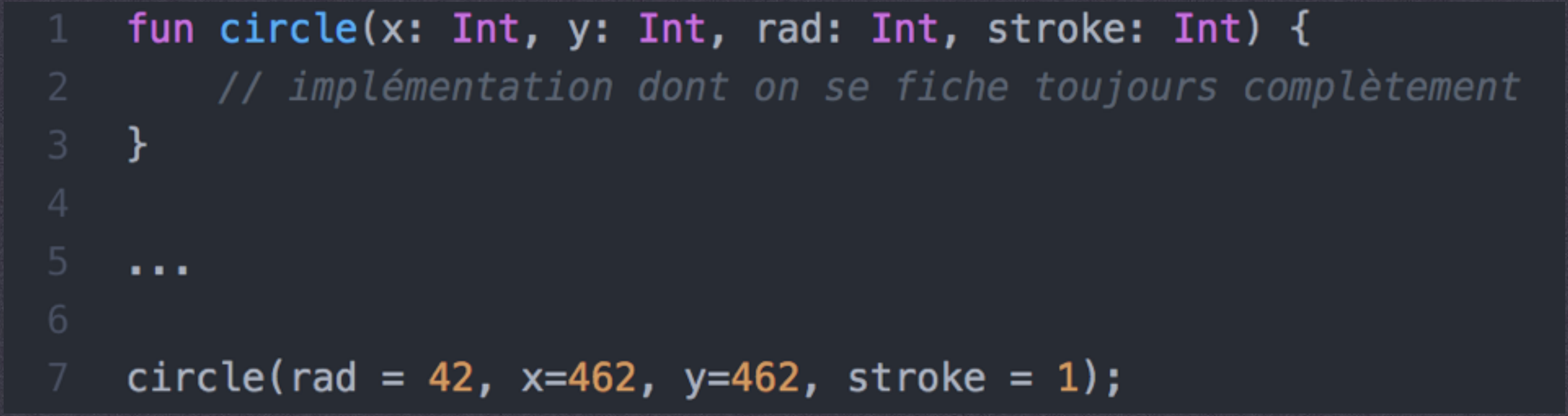
Code d'Armor

Arguments nommés



```kotlin
fun circle(x: Int, y: Int, rad: Int, stroke: Int) {
    // implémentation dont on se fiche toujours complètement
}

...

circle(462, 462, rad = 42, stroke = 1);
```

Arguments nommés



```kotlin
fun circle(x: Int, y: Int, rad: Int, stroke: Int) {
    // implémentation dont on se fiche toujours complètement
}

...

circle(rad = 42, x=462, y=462, stroke = 1);
```

Arguments optionnels

```
fun circle(x: Int, y: Int, rad: Int, stroke: Int = 1) {
    // implémentation dont on se fiche toujours complètement
}

...

circle(rad = 42, x=462, y=462);
circle(rad = 44, stroke= 2, x=462, y=462);
```

Code
d'Armor

Manipulation de Collections

- map
- groupBy
- filter
- fold (à gauche et à droite)
- reduce
- merge, partition
- sort
- …

Fonctions d’ordre supérieurs et λ

- Les fonctions d’ordre supérieur:
  - retournent une fonction ou
  - prennent une fonction en paramètre
- Les fonctions λ:
  - sont anonymes
  - sont passées sous la forme de leur expression

Code d’Armor

```kotlin
fun <T, R> List<T>.map(transform: (T) -> R): List<R> {
  val result = arrayListOf<R>()
  for (item in this)
    result.add(transform(item))
  return result
}
```

```kotlin
val doubled = ints.map { it -> it * 2 }
```

FONCTION D’ORDRE SUPÉRIEUR ET Λ

Le type Null

- Null est un type à part entière
- Une variable ne peut être null que si elle y est explicitement autorisée
- Plus de null check, Safe Call
- Calling Elvis ?
- Smooth Operator !!

Code d'Armor

```kotlin
var name: String = "Marc"
name = null // nope, le compilo refuse
```

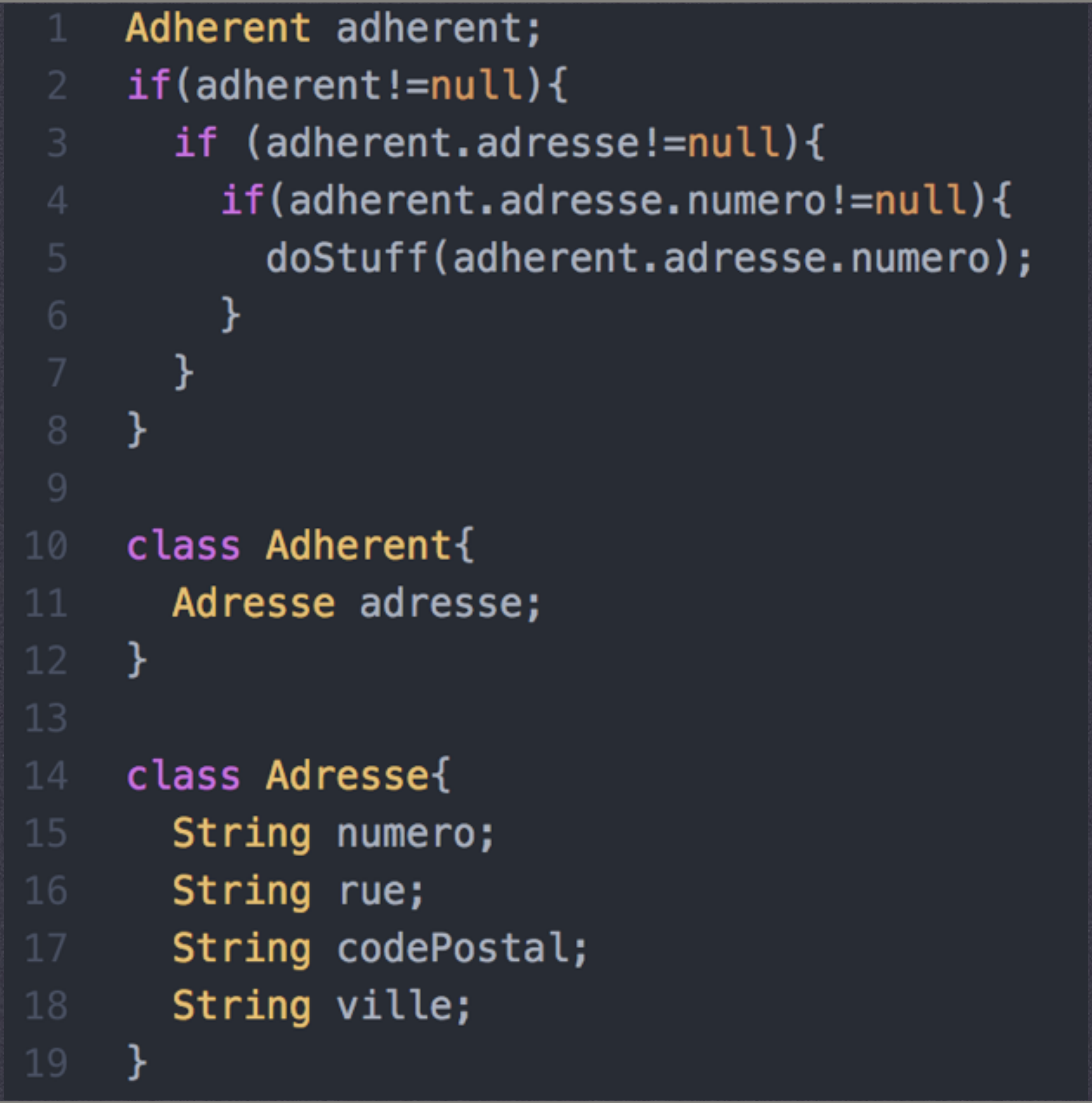
```kotlin
var nickname: String? = "Pop"
nickname = null // ça compile, merci Elvis
```

```kotlin
var nickname: String? = "Pop"
nickname = null // ça compile, merci Elvis
val l = nickname!!.length() // et paf NPE dans ta face!
```

```kotlin
var nickname: String? = "Pop"
nickname = null // ça compile, merci Elvis
val l = nickname?.length() // pas de NPE
```

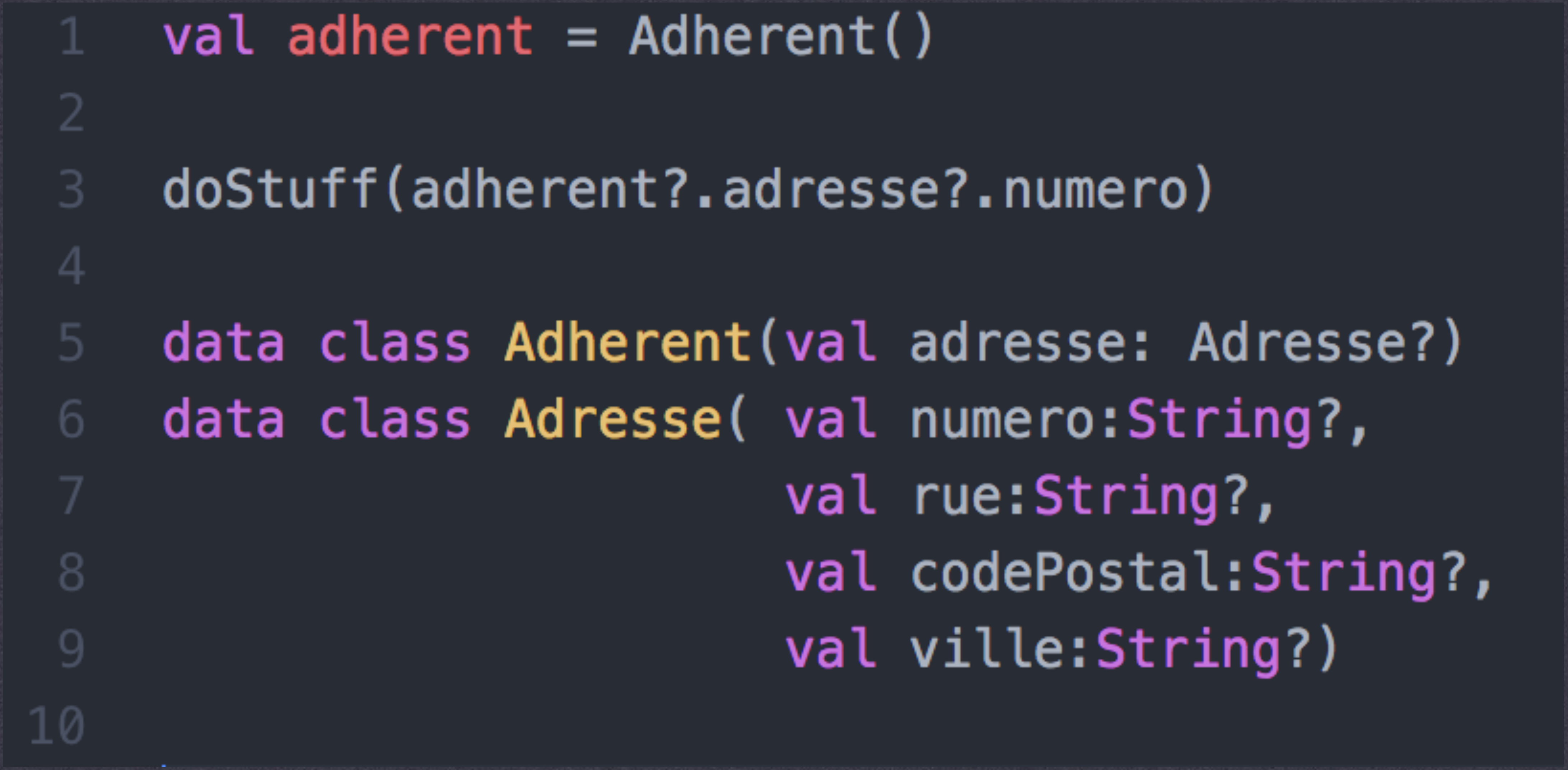
NULL

Code d'Armor

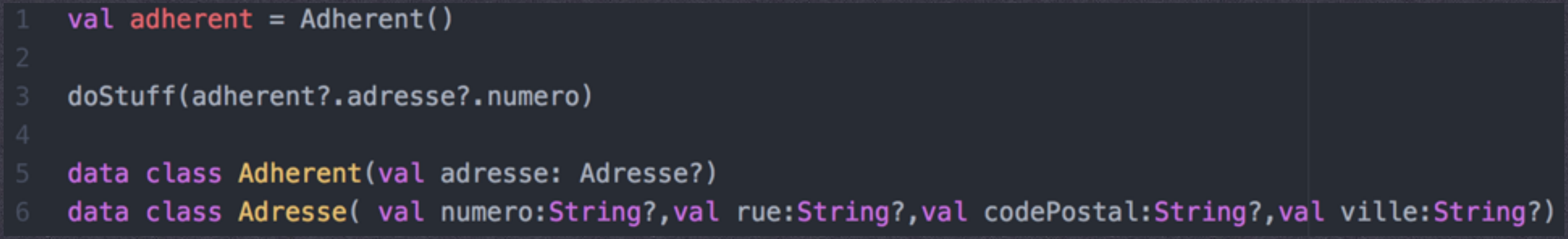


```java
Adherent adherent;
if(adherent!=null){
  if (adherent.adresse!=null){
    if(adherent.adresse.numero!=null){
      doStuff(adherent.adresse.numero);
    }
  }
}

class Adherent{
  Adresse adresse;
}

class Adresse{
  String numero;
  String rue;
  String codePostal;
  String ville;
}
```

NULL



```kotlin
val adherent = Adherent()

doStuff(adherent?.adresse?.numero)

data class Adherent(val adresse: Adresse?)
data class Adresse( val numero:String?,
                    val rue:String?,
                    val codePostal:String?,
                    val ville:String?)

```



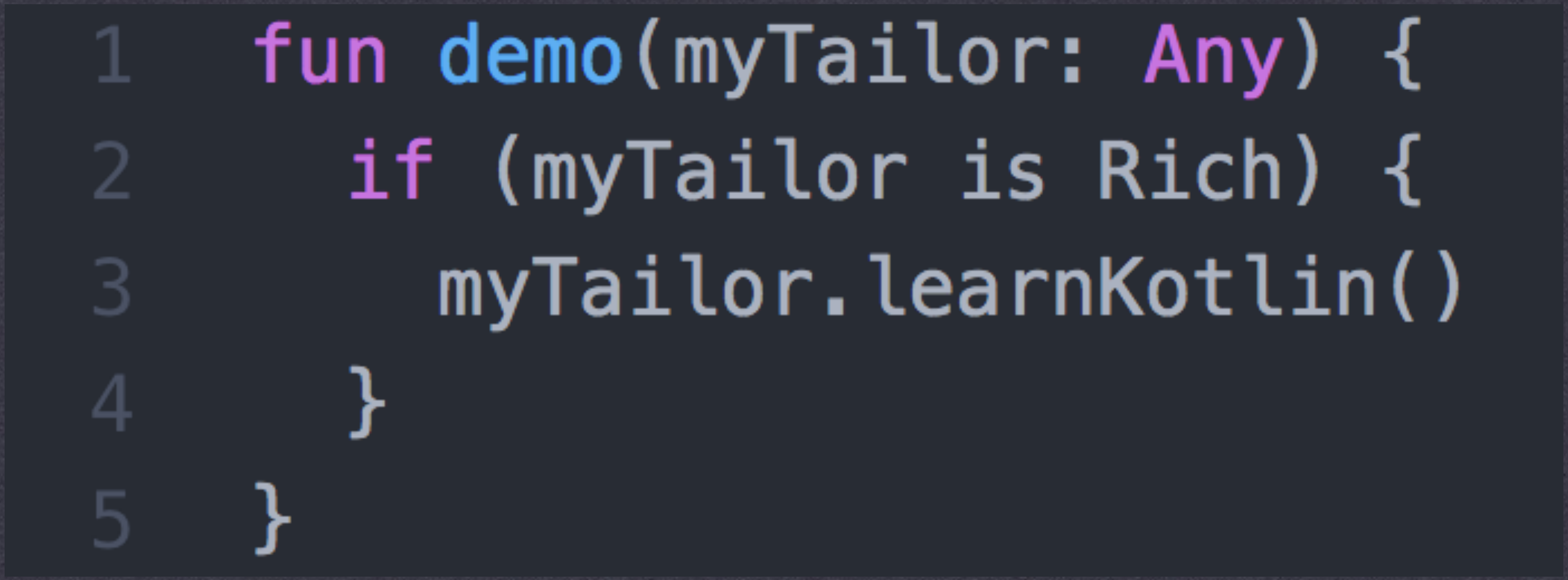
```kotlin
val adherent = Adherent()

doStuff(adherent?.adresse?.numero)

data class Adherent(val adresse: Adresse?)
data class Adresse( val numero:String?,val rue:String?,val codePostal:String?,val ville:String?)
```

NULL

Type & cast

- To be or not to be (is or !is)
- Unsafe cast (as)
- Safe cast (as?)

Code d'Armor

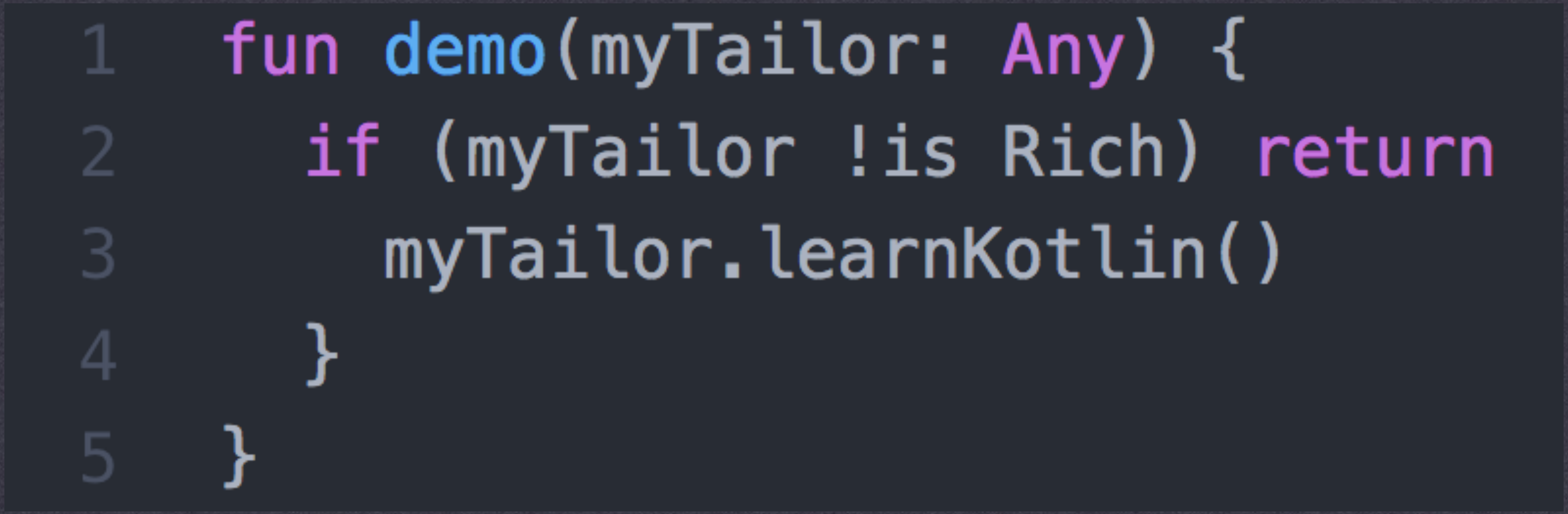


```kotlin
fun demo(myTailor: Any) {
  if (myTailor is Rich) {
    myTailor.learnKotlin()
  }
}
```

TYPE CHECK

SMART CAST

Code d'Armor

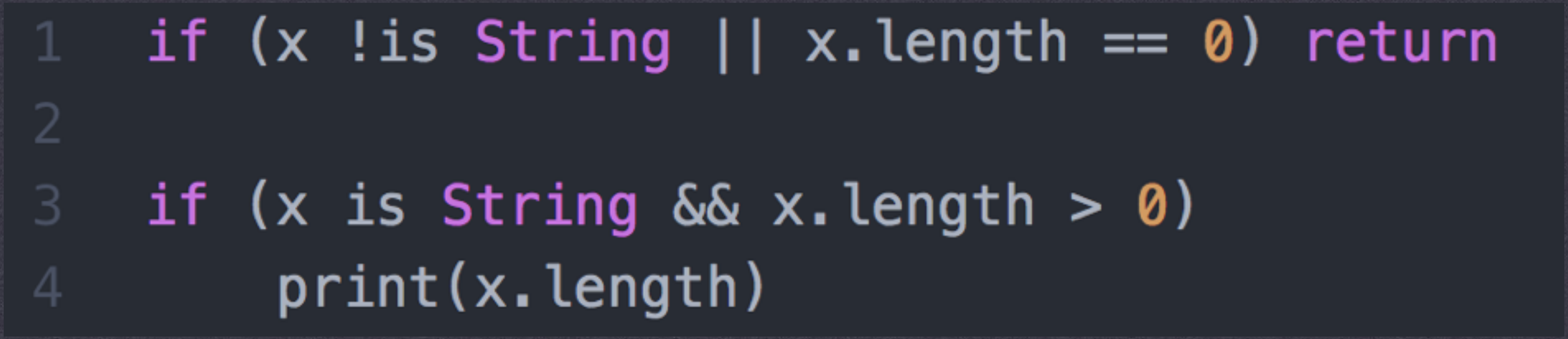


```kotlin
fun demo(myTailor: Any) {
  if (myTailor !is Rich) return
    myTailor.learnKotlin()
  }
}
```

TYPE CHECK

SMART CAST

Code d'Armor



```
if (x !is String || x.length == 0) return

if (x is String && x.length > 0)
    print(x.length)
```

TYPE CHECK

SMART CAST

Code d'Armor

```kotlin
val myTailor: Rich = someone as Rich
```

TYPE CHECK

UNSAFE CAST

Code d'Armor

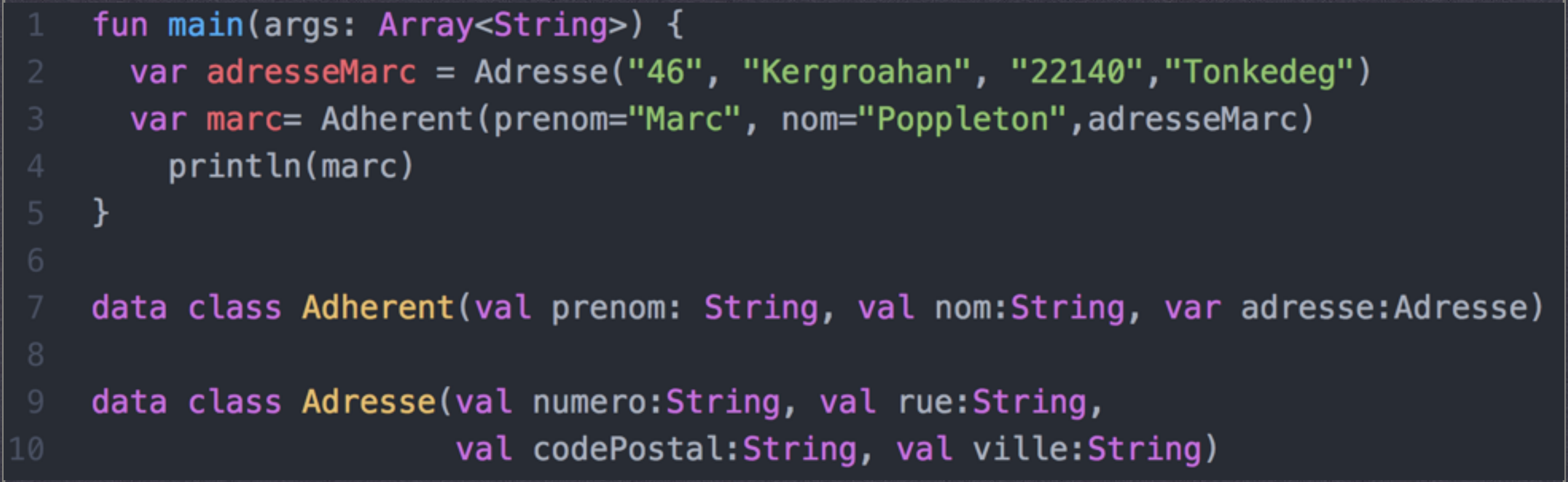
```kotlin
val myTailor: Rich? = someone as? Rich
```

TYPE CHECK

SAFE CAST

Classes et Methodes

- Tout est final par défaut
- Constructeur primaire
- Constructeurs secondaires
- RW var
- Read Only val
- Getter et Setter automatiques

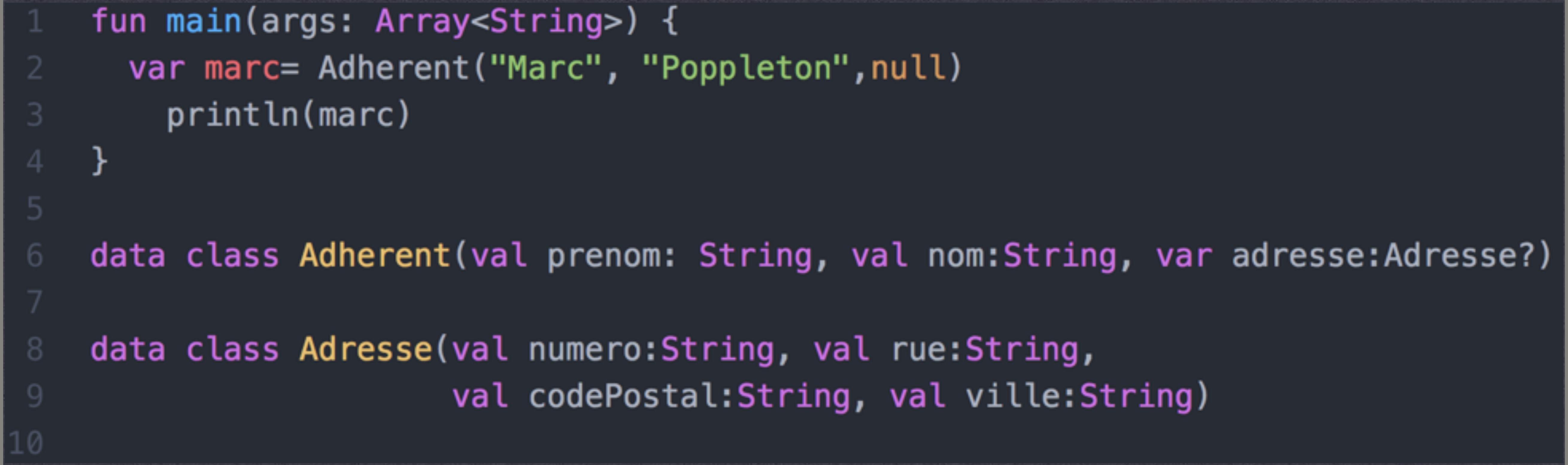


```
fun main(args: Array<String>) {
  var adresseMarc = Adresse("46", "Kergroahan", "22140","Tonkedeg")
  var marc= Adherent(prenom="Marc", nom="Poppleton",adresseMarc)
    println(marc)
}

data class Adherent(val prenom: String, val nom:String, var adresse:Adresse)

data class Adresse(val numero:String, val rue:String,
                   val codePostal:String, val ville:String)
```

CLASSES

CONSTRUCTEUR PRIMAIRE ET CONSTRUCTEURS SECONDAIRES

Code d'Armor



```kotlin
fun main(args: Array<String>) {
  var marc= Adherent("Marc", "Poppleton",null)
    println(marc)
}

data class Adherent(val prenom: String, val nom:String, var adresse:Adresse?)

data class Adresse(val numero:String, val rue:String,
                   val codePostal:String, val ville:String)

```

CLASSES

CONSTRUCTEUR PRIMAIRE ET CONSTRUCTEURS SECONDAIRES

```kotlin
fun main(args: Array<String>) {
  var marc= Adherent("Marc", "Poppleton")
    println(marc)
}

data class Adherent(val prenom: String, val nom:String, var adresse:Adresse?){
  constructor(prenom:String, nom:String) : this(prenom,nom,null)
}

data class Adresse(val numero:String, val rue:String,
                   val codePostal:String, val ville:String)
```

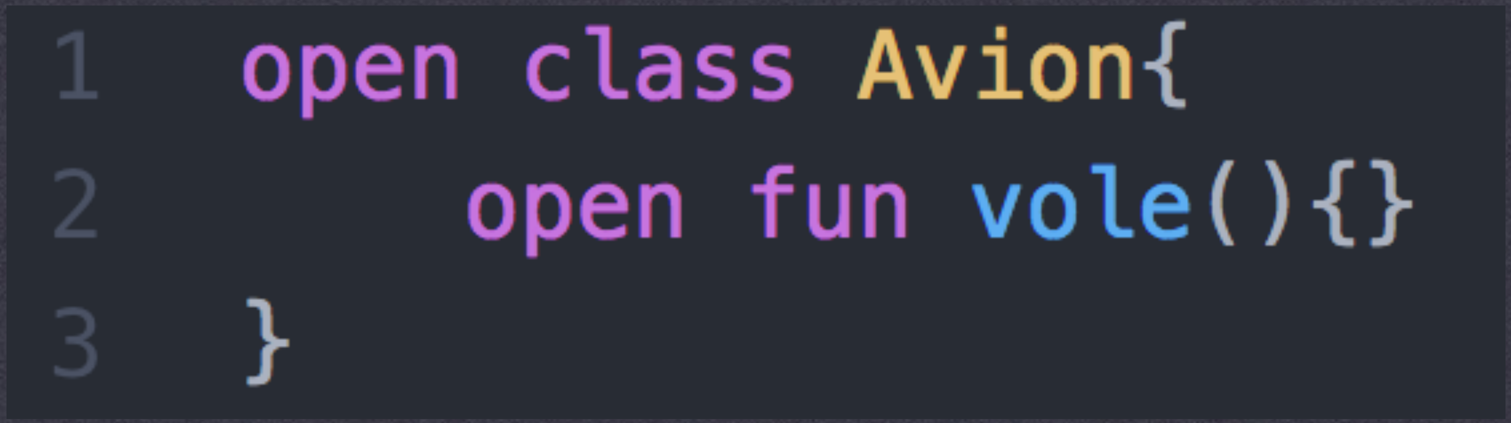
CLASSES

CONSTRUCTEUR PRIMAIRE ET CONSTRUCTEURS SECONDAIRES

Code d'Armor

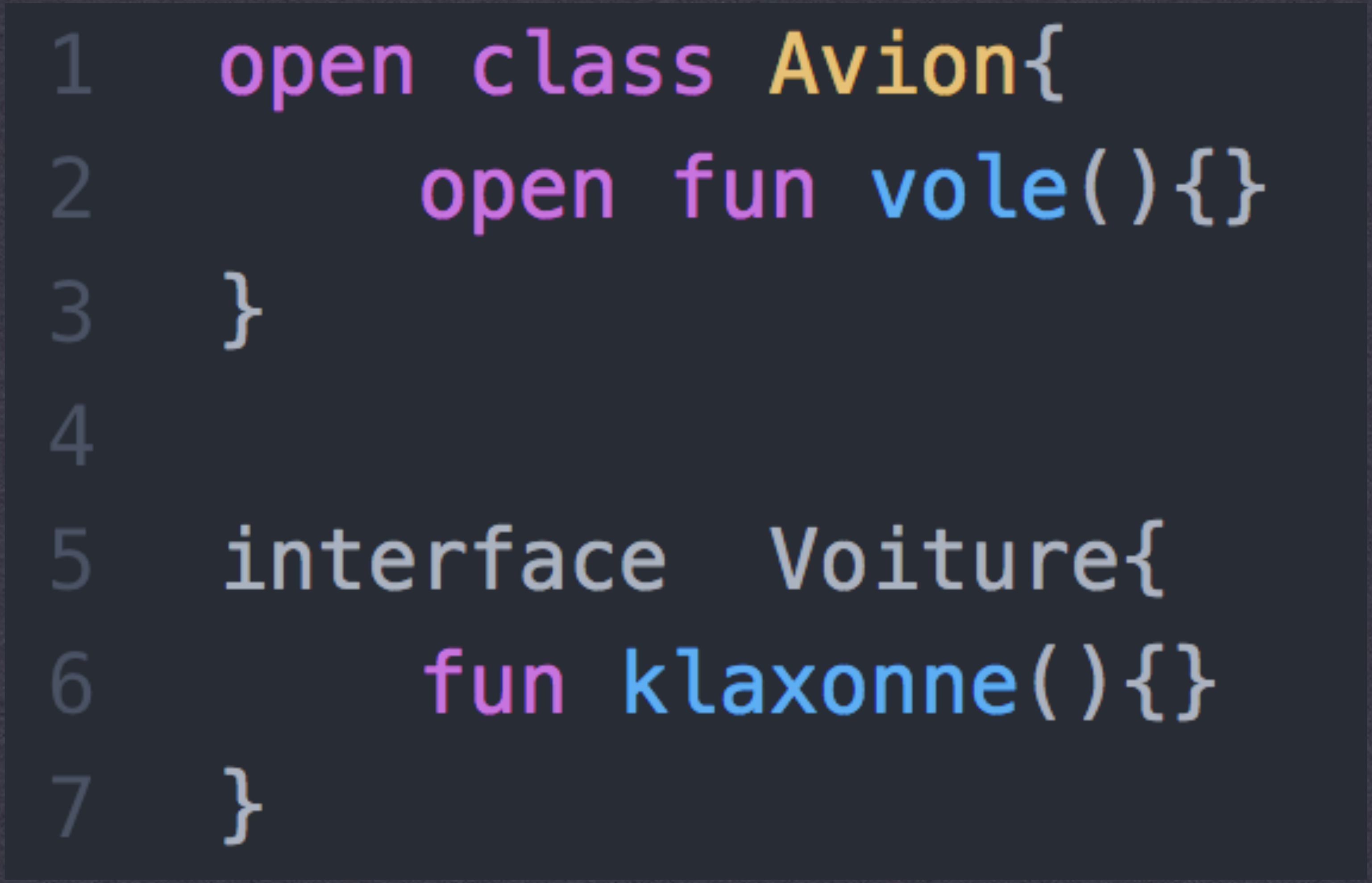
Interfaces

- avec implémentation de fonctions
- héritage = implémentation



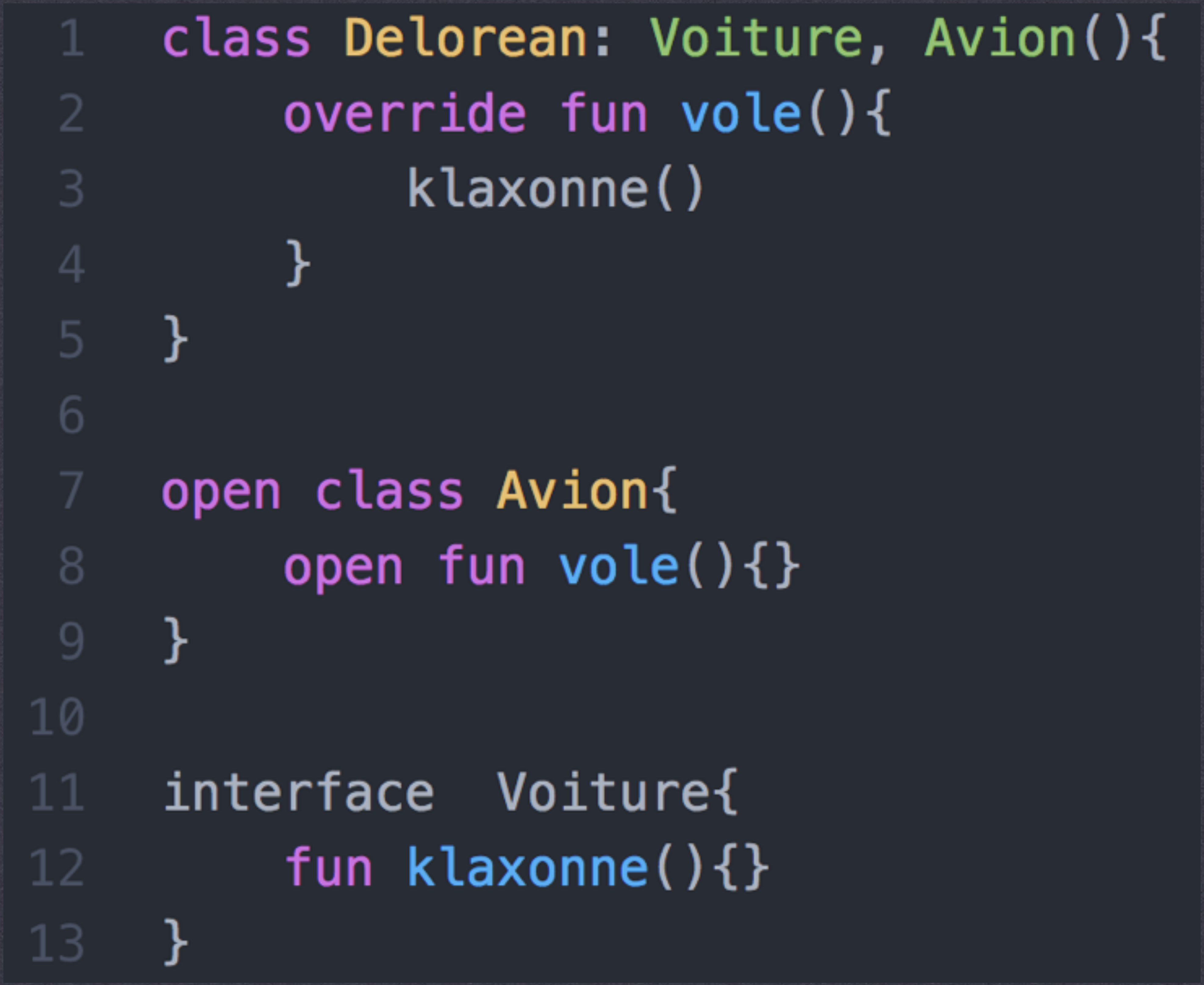
```kotlin
open class Avion{
    open fun vole(){}
}
```

INTERFACES



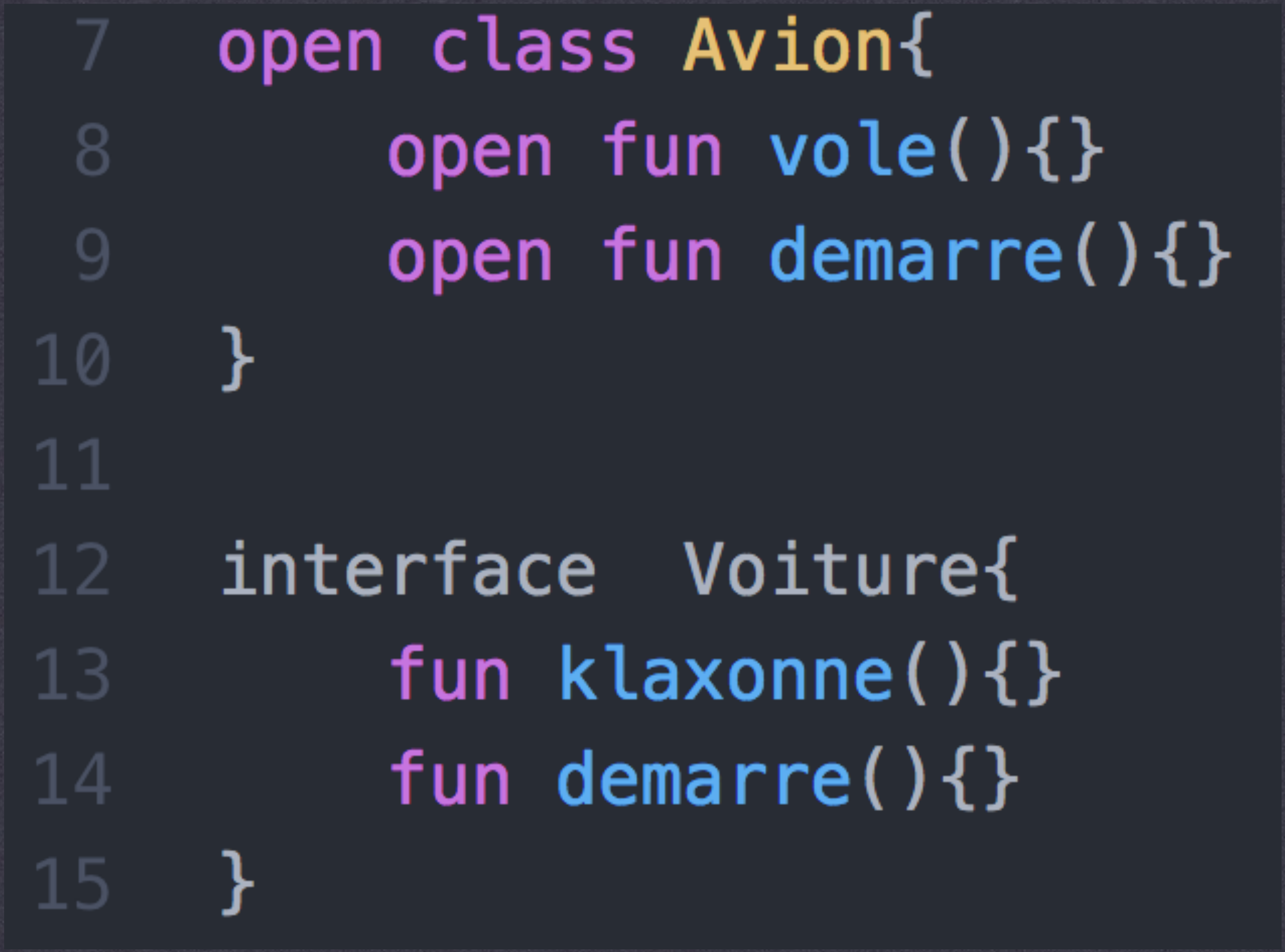
```kotlin
open class Avion{
      open fun vole(){}
}

interface  Voiture{
      fun klaxonne(){}
}
```

INTERFACES



```
class Delorean: Voiture, Avion(){
    override fun vole(){
        klaxonne()
    }
}

open class Avion{
    open fun vole(){}
}

interface  Voiture{
    fun klaxonne(){}
}
```

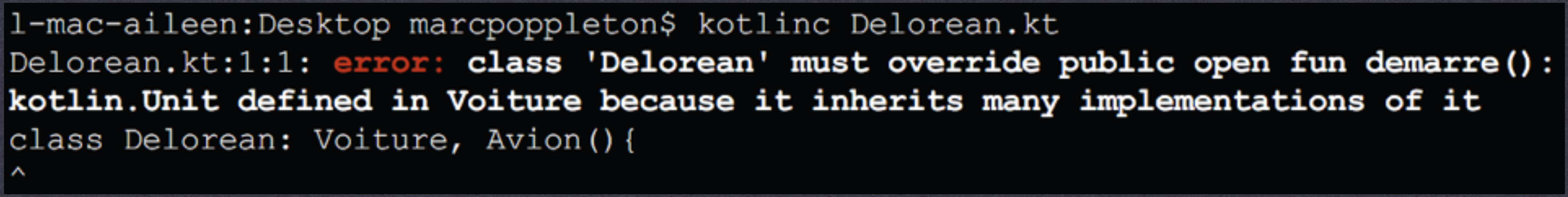
INTERFACES



```kotlin
open class Avion{
    open fun vole(){}
    open fun demarre(){}
}

interface  Voiture{
    fun klaxonne(){}
    fun demarre(){}
}
```

INTERFACES

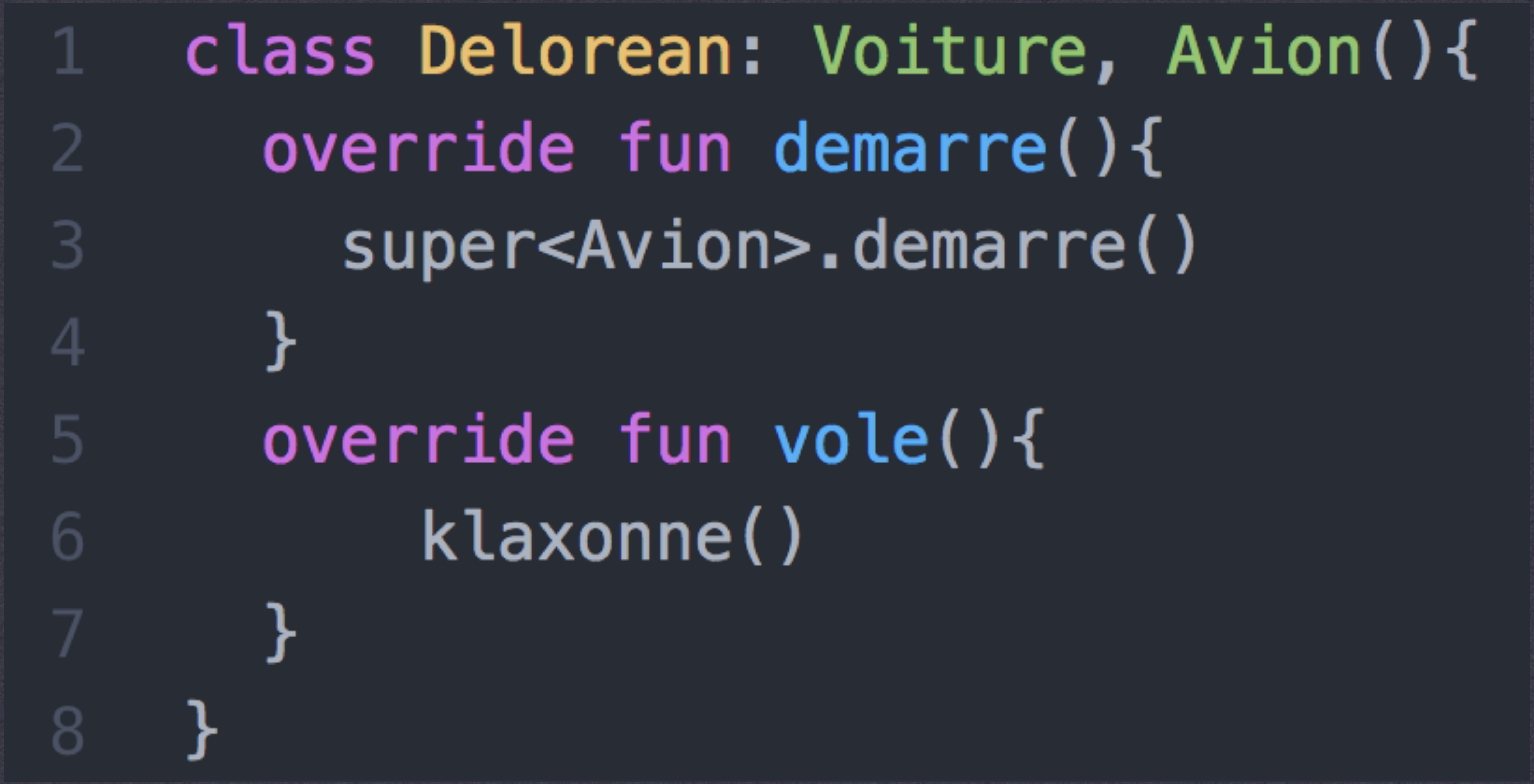
Code d'Armor



```
l-mac-aileen:Desktop marcpoppleton$ kotlinc Delorean.kt
Delorean.kt:1:1: error: class 'Delorean' must override public open fun demarre():
kotlin.Unit defined in Voiture because it inherits many implementations of it
class Delorean: Voiture, Avion(){
^
```

INTERFACES

Code d'Armor

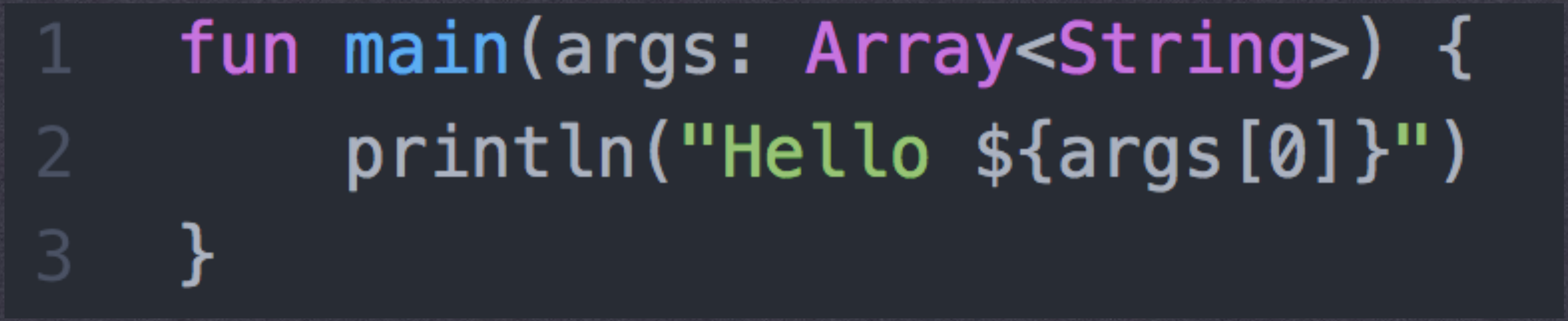


```kotlin
class Delorean: Voiture, Avion(){
  override fun demarre(){
    super<Avion>.demarre()
  }
  override fun vole(){
      klaxonne()
  }
}
```

INTERFACES

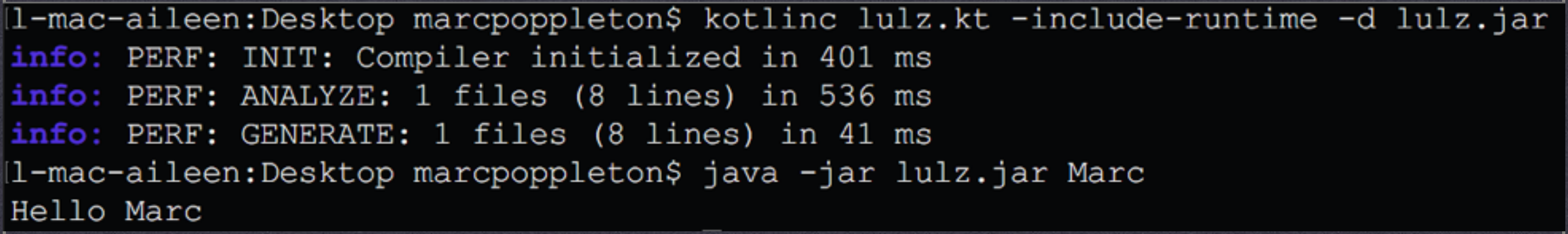
Extensions

- Etendre une classe sans héritage
- Plus besoin de classes utilitaires



```kotlin
fun main(args: Array<String>) {
    println("Hello ${args[0]}")
}
```

EXTENSIONS



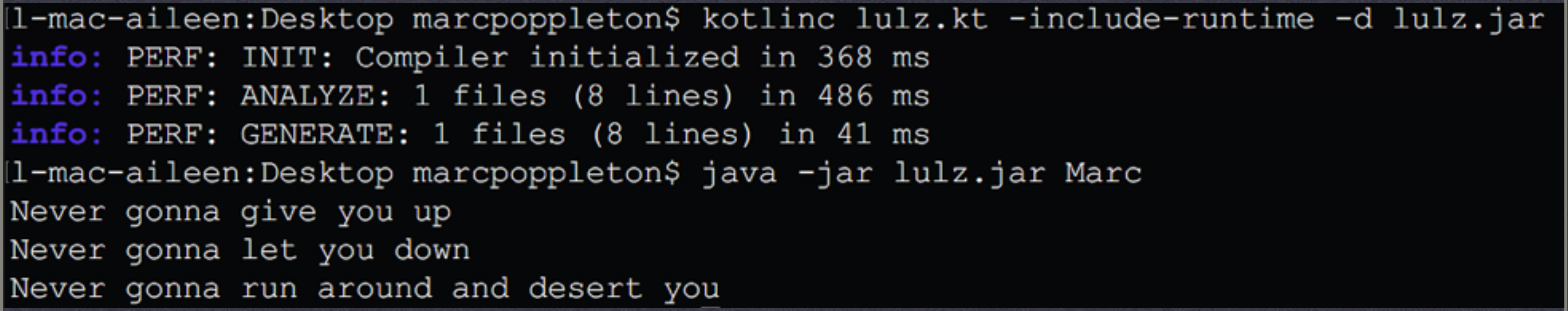
```
l-mac-aileen:Desktop marcpoppleton$ kotlinc lulz.kt -include-runtime -d lulz.jar
info: PERF: INIT: Compiler initialized in 401 ms
info: PERF: ANALYZE: 1 files (8 lines) in 536 ms
info: PERF: GENERATE: 1 files (8 lines) in 41 ms
l-mac-aileen:Desktop marcpoppleton$ java -jar lulz.jar Marc
Hello Marc
```

EXTENSIONS

Code d'Armor

```kotlin
fun main(args: Array<String>) {
    println(args[0].rickroll())
}

fun String.rickroll(): String {
    return "Never gonna give you up\nNever gonna let you down\n" +
            "Never gonna run around and desert you"
}
```

EXTENSIONS



```
[l-mac-aileen:Desktop marcpoppleton$ kotlinc lulz.kt -include-runtime -d lulz.jar
info: PERF: INIT: Compiler initialized in 368 ms
info: PERF: ANALYZE: 1 files (8 lines) in 486 ms
info: PERF: GENERATE: 1 files (8 lines) in 41 ms
[l-mac-aileen:Desktop marcpoppleton$ java -jar lulz.jar Marc
Never gonna give you up
Never gonna let you down
Never gonna run around and desert you
```

EXTENSIONS

Code
d'Armor

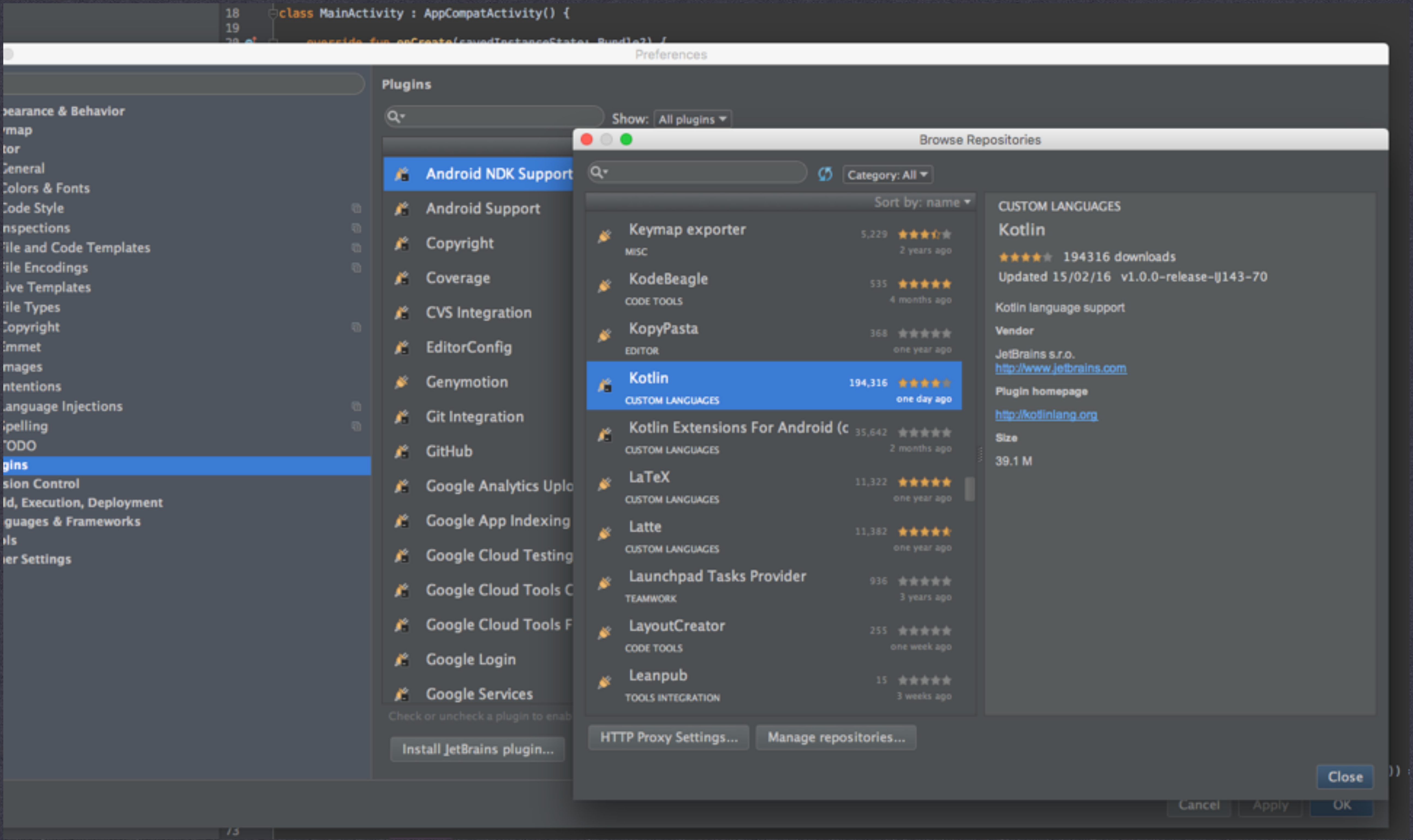
KOTLIN ET ANDROID

Code
d'Armor

Outillage

- Android Studio et IntelliJ IDEA
  - Ajouter le plugin (sauf IntelliJ IDEA 15 et plus)
  - Ajouter la dépendance Gradle
  - Synchroniser
- Eclipse
  - Ajouter le plugin
  - Attendre une semaine que le plugin se télécharge
- Je suis un barbu, je code avec vi et compile en ligne de commande
  - Télécharger le compilateur sur Github
- Je suis un barbu hipster, je code avec Atom pour OSX et compile en ligne de commande
  - brew install kotlin
  - Installer le plugin Atom language-kotlin
  - Aller se chercher en fixie un moccacino fait à l'aeropress

Code d'Armor



```
class MainActivity : AppCompatActivity() {
```

Preferences

earance & Behavior
map
tor
General
Colors & Fonts
Code Style
nspections
File and Code Templates
File Encodings
Live Templates
File Types
Copyright
Emmet
mages
ntentions
Language Injections
Spelling
TODO
gins
sion Control
ld, Execution, Deployment
guages & Frameworks
ls
er Settings

Plugins

Show: All plugins

Android NDK Support
Android Support
Copyright
Coverage
CVS Integration
EditorConfig
Genymotion
Git Integration
GitHub
Google Analytics Uplo
Google App Indexing
Google Cloud Testing
Google Cloud Tools C
Google Cloud Tools F
Google Login
Google Services

Check or uncheck a plugin to enab

Install JetBrains plugin...

Browse Repositories

Category: All

Sort by: name

Plugin	Category	Downloads	Updated
Keymap exporter	MISC	5,229	2 years ago
KodeBeagle	CODE TOOLS	535	4 months ago
KopyPasta	EDITOR	368	one year ago
Kotlin	CUSTOM LANGUAGES	194,316	one day ago
Kotlin Extensions For Android (c	CUSTOM LANGUAGES	35,642	2 months ago
LaTeX	CUSTOM LANGUAGES	11,322	one year ago
Latte	CUSTOM LANGUAGES	11,382	one year ago
Launchpad Tasks Provider	TEAMWORK	936	3 years ago
LayoutCreator	CODE TOOLS	255	one week ago
Leanpub	TOOLS INTEGRATION	15	3 weeks ago

CUSTOM LANGUAGES

Kotlin

★★★★ 194316 downloads

Updated 15/02/16 v1.0.0-release-IJ143-70

Kotlin language support

Vendor

JetBrains s.r.o.
http://www.jetbrains.com

Plugin homepage

http://kotlinlang.org

Size

39.1 M

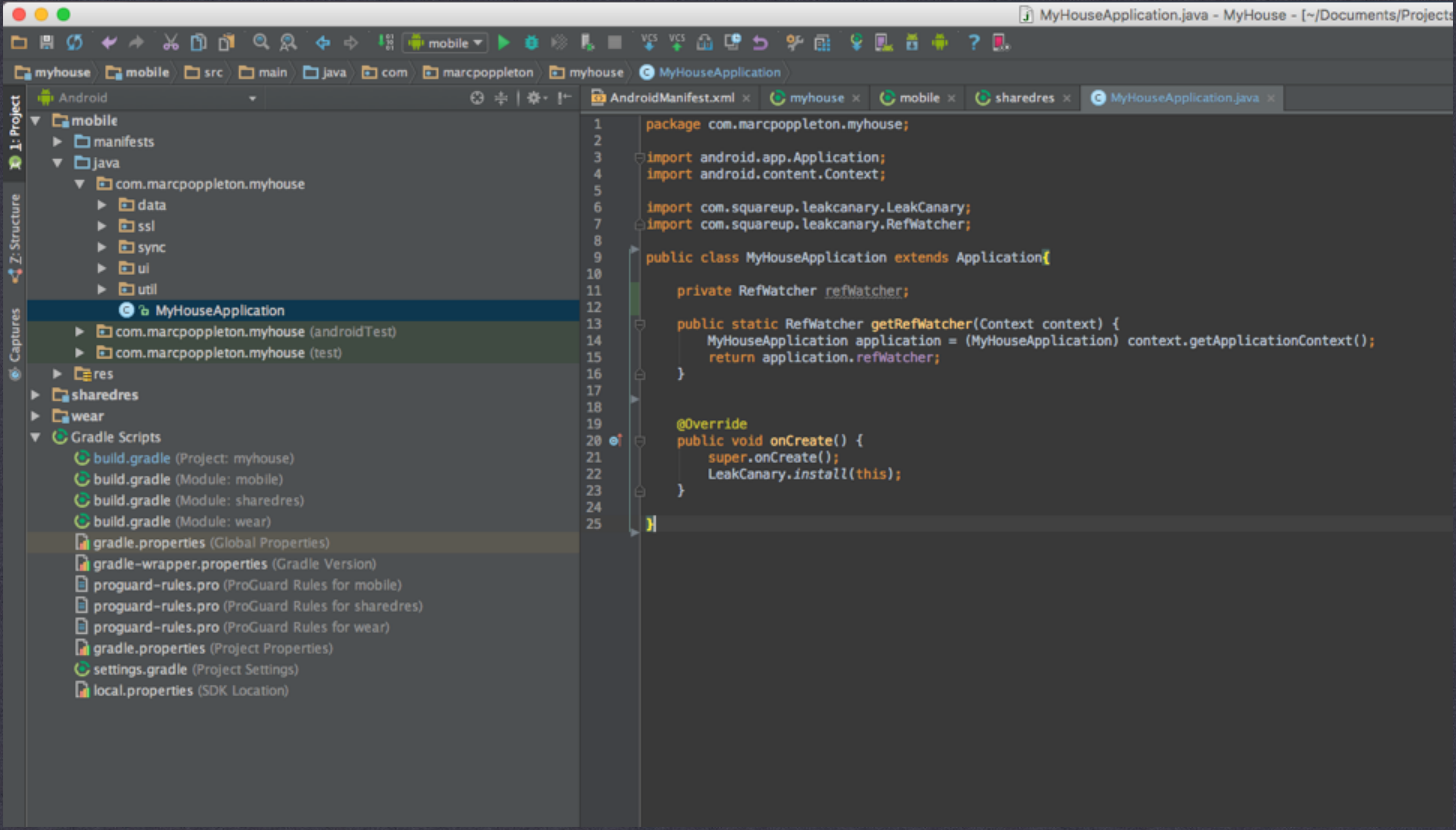
HTTP Proxy Settings... Manage repositories... Close

Cancel Apply OK

KOTLIN & ANDROID

MIGRER UN SOURCE EXISTANT

Code d'Armor



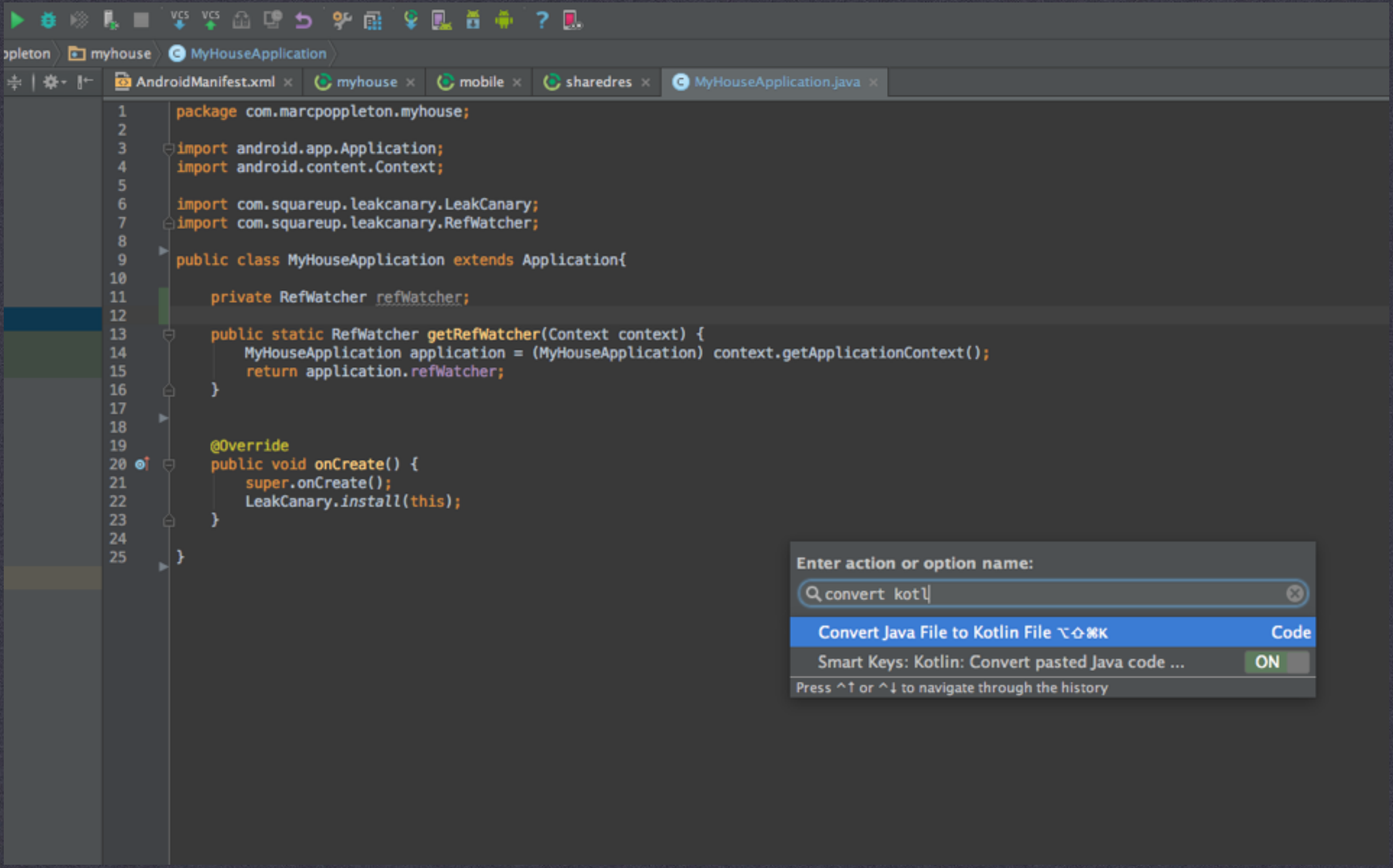
```
mobile
  manifests
  java
    com.marcpoppleton.myhouse
      data
      ssl
      sync
      ui
      util
      MyHouseApplication
    com.marcpoppleton.myhouse (androidTest)
    com.marcpoppleton.myhouse (test)
  res
sharedres
wear
Gradle Scripts
  build.gradle (Project: myhouse)
  build.gradle (Module: mobile)
  build.gradle (Module: sharedres)
  build.gradle (Module: wear)
  gradle.properties (Global Properties)
  gradle-wrapper.properties (Gradle Version)
  proguard-rules.pro (ProGuard Rules for mobile)
  proguard-rules.pro (ProGuard Rules for sharedres)
  proguard-rules.pro (ProGuard Rules for wear)
  gradle.properties (Project Properties)
  settings.gradle (Project Settings)
  local.properties (SDK Location)
```

```java
package com.marcpoppleton.myhouse;

import android.app.Application;
import android.content.Context;

import com.squareup.leakcanary.LeakCanary;
import com.squareup.leakcanary.RefWatcher;

public class MyHouseApplication extends Application{

    private RefWatcher refWatcher;

    public static RefWatcher getRefWatcher(Context context) {
        MyHouseApplication application = (MyHouseApplication) context.getApplicationContext();
        return application.refWatcher;
    }


    @Override
    public void onCreate() {
        super.onCreate();
        LeakCanary.install(this);
    }

}
```

KOTLIN & ANDROID

MIGRER UN SOURCE EXISTANT

Code d'Armor



```java
package com.marcpoppleton.myhouse;

import android.app.Application;
import android.content.Context;

import com.squareup.leakcanary.LeakCanary;
import com.squareup.leakcanary.RefWatcher;

public class MyHouseApplication extends Application{

    private RefWatcher refWatcher;

    public static RefWatcher getRefWatcher(Context context) {
        MyHouseApplication application = (MyHouseApplication) context.getApplicationContext();
        return application.refWatcher;
    }


    @Override
    public void onCreate() {
        super.onCreate();
        LeakCanary.install(this);
    }

}
```

Enter action or option name:

convert kotl

Convert Java File to Kotlin File ⌥⇧⌘K — Code

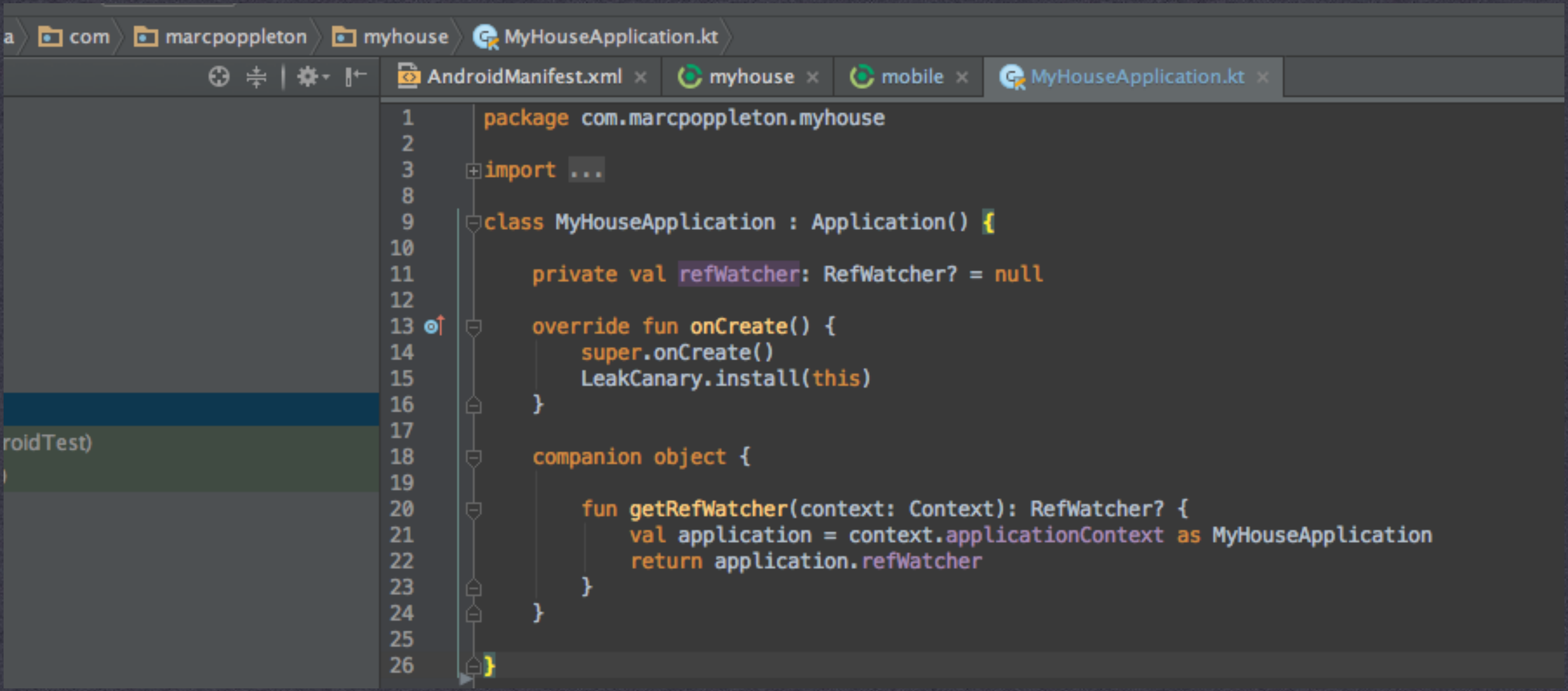
Smart Keys: Kotlin: Convert pasted Java code ... — ON

Press ^↑ or ^↓ to navigate through the history

KOTLIN & ANDROID

MIGRER UN SOURCE EXISTANT

Code d'Armor

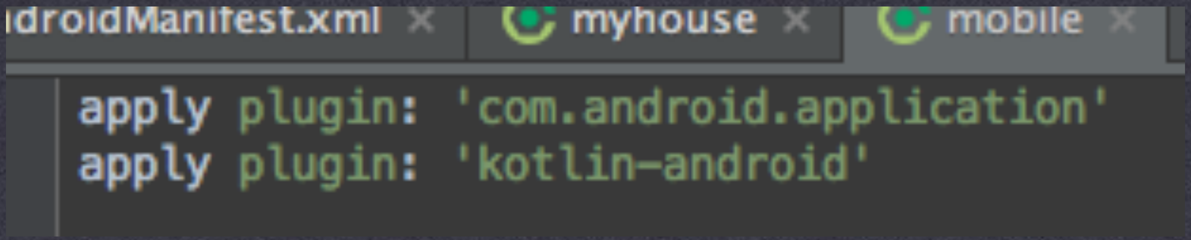


```kotlin
package com.marcpoppleton.myhouse

import ...

class MyHouseApplication : Application() {

    private val refWatcher: RefWatcher? = null

    override fun onCreate() {
        super.onCreate()
        LeakCanary.install(this)
    }

    companion object {

        fun getRefWatcher(context: Context): RefWatcher? {
            val application = context.applicationContext as MyHouseApplication
            return application.refWatcher
        }
    }

}
```

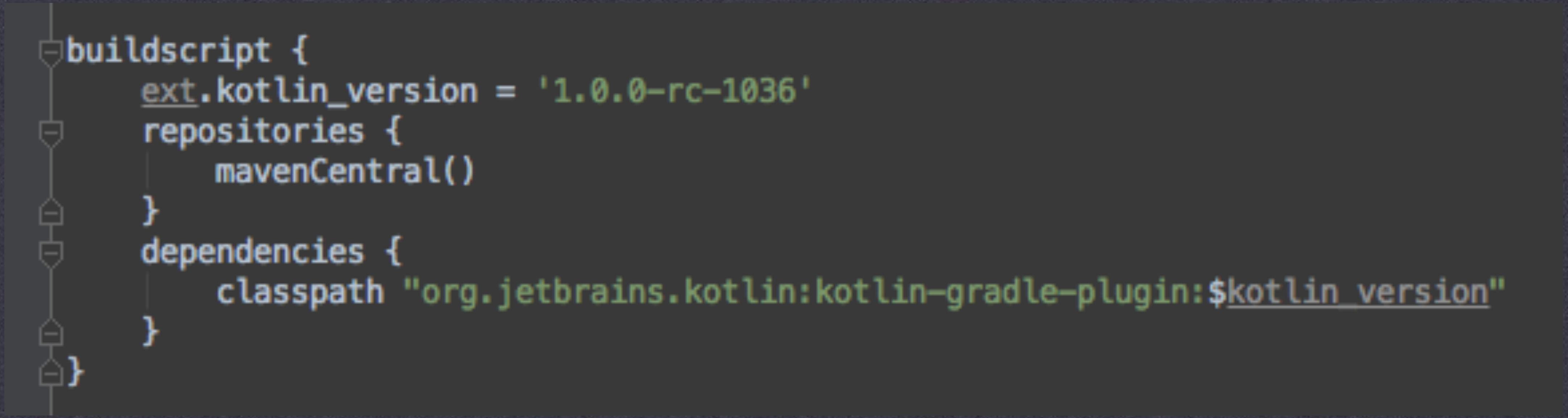
KOTLIN & ANDROID

MIGRER UN SOURCE EXISTANT

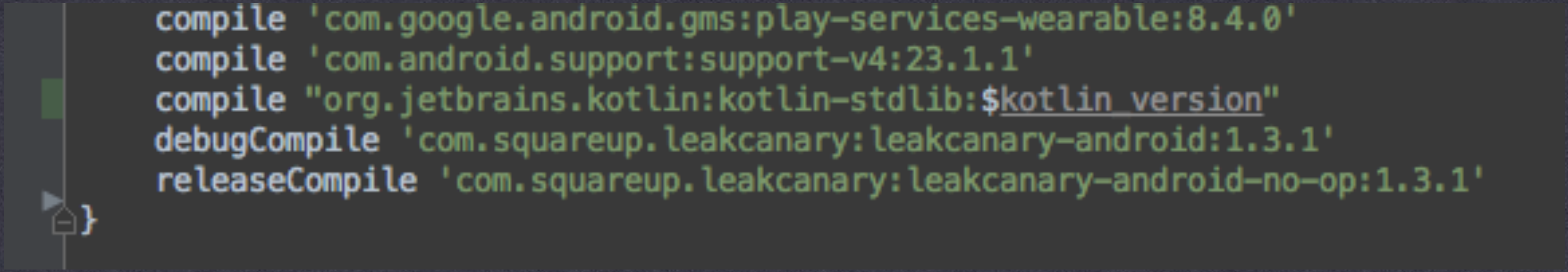
Code d'Armor



```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
```



```
buildscript {
    ext.kotlin_version = '1.0.0-rc-1036'
    repositories {
        mavenCentral()
    }
    dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
    }
}
```

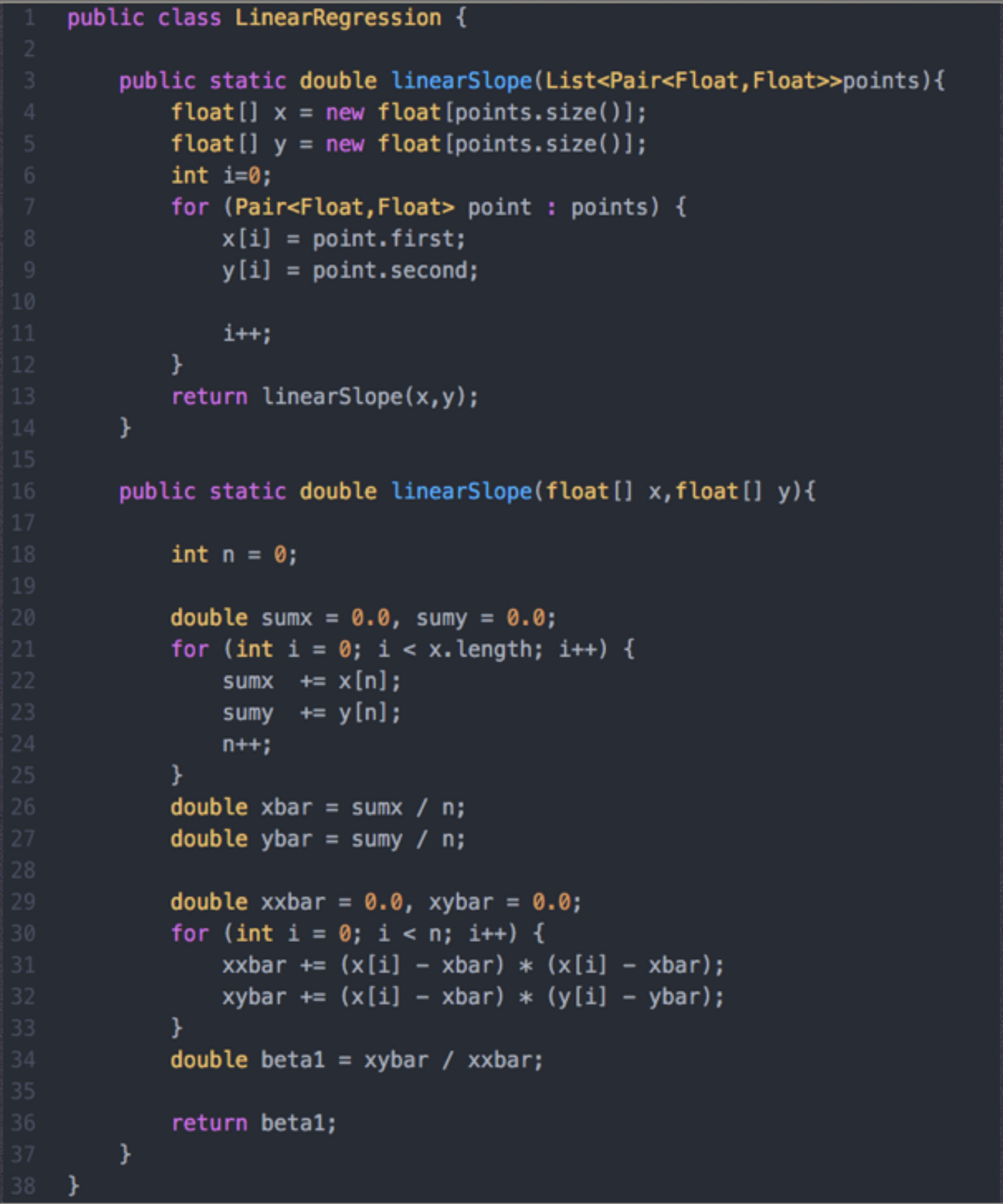


```
    compile 'com.google.android.gms:play-services-wearable:8.4.0'
    compile 'com.android.support:support-v4:23.1.1'
    compile "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    debugCompile 'com.squareup.leakcanary:leakcanary-android:1.3.1'
    releaseCompile 'com.squareup.leakcanary:leakcanary-android-no-op:1.3.1'
}
```

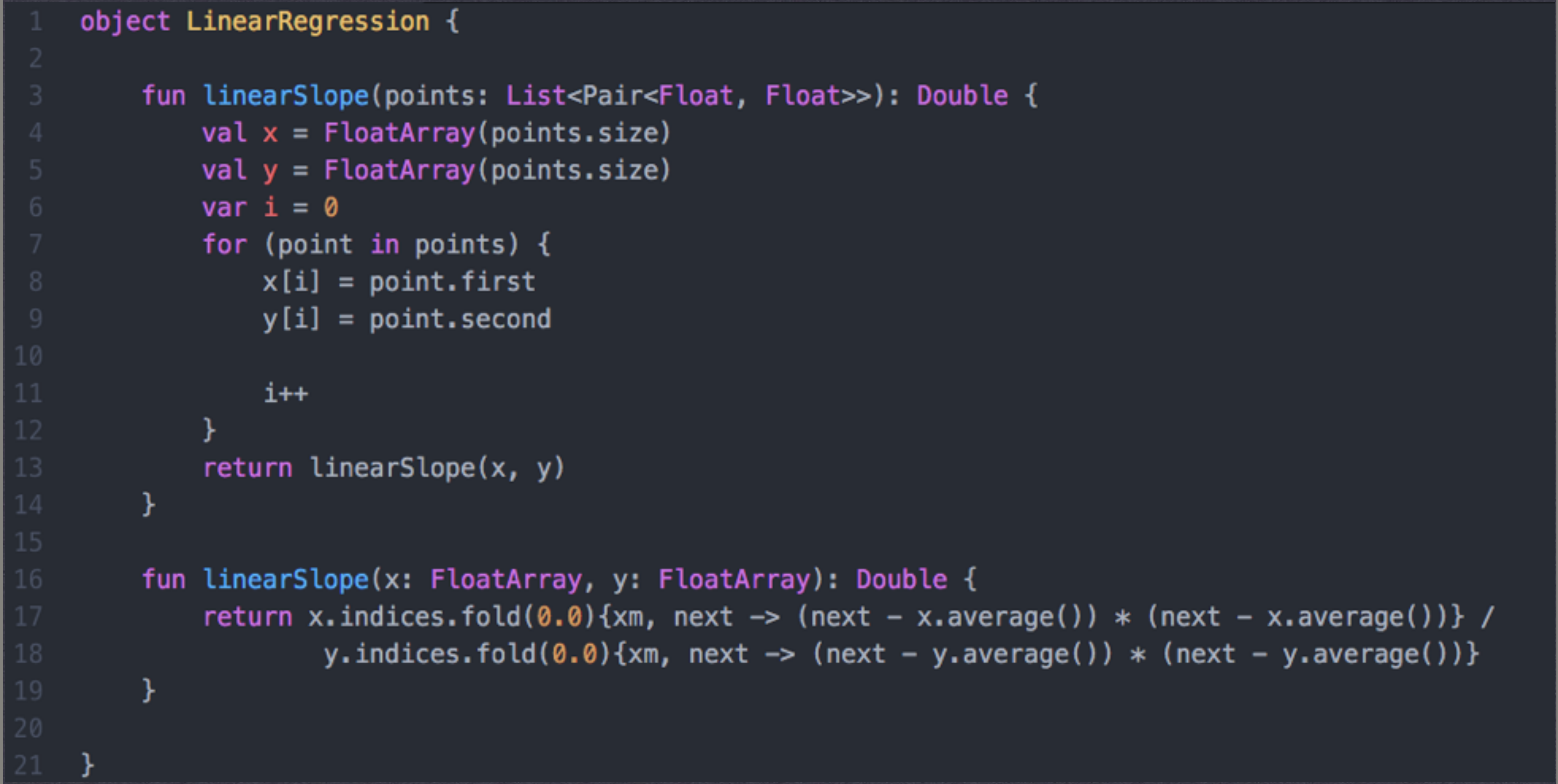
KOTLIN & ANDROID

MIGRER UN SOURCE EXISTANT

UN EXAMPLE!



```java
public class LinearRegression {

    public static double linearSlope(List<Pair<Float,Float>>points){
        float[] x = new float[points.size()];
        float[] y = new float[points.size()];
        int i=0;
        for (Pair<Float,Float> point : points) {
            x[i] = point.first;
            y[i] = point.second;

            i++;
        }
        return linearSlope(x,y);
    }

    public static double linearSlope(float[] x,float[] y){

        int n = 0;

        double sumx = 0.0, sumy = 0.0;
        for (int i = 0; i < x.length; i++) {
            sumx  += x[n];
            sumy  += y[n];
            n++;
        }
        double xbar = sumx / n;
        double ybar = sumy / n;

        double xxbar = 0.0, xybar = 0.0;
        for (int i = 0; i < n; i++) {
            xxbar += (x[i] - xbar) * (x[i] - xbar);
            xybar += (x[i] - xbar) * (y[i] - ybar);
        }
        double beta1 = xybar / xxbar;

        return beta1;
    }
}
```


```
object LinearRegression {

    fun linearSlope(points: List<Pair<Float, Float>>): Double {
        val x = FloatArray(points.size)
        val y = FloatArray(points.size)
        var i = 0
        for (point in points) {
            x[i] = point.first
            y[i] = point.second

            i++
        }
        return linearSlope(x, y)
    }

    fun linearSlope(x: FloatArray, y: FloatArray): Double {
        return x.indices.fold(0.0){xm, next -> (next - x.average()) * (next - x.average())} /
                y.indices.fold(0.0){xm, next -> (next - y.average()) * (next - y.average())}
    }

}
```

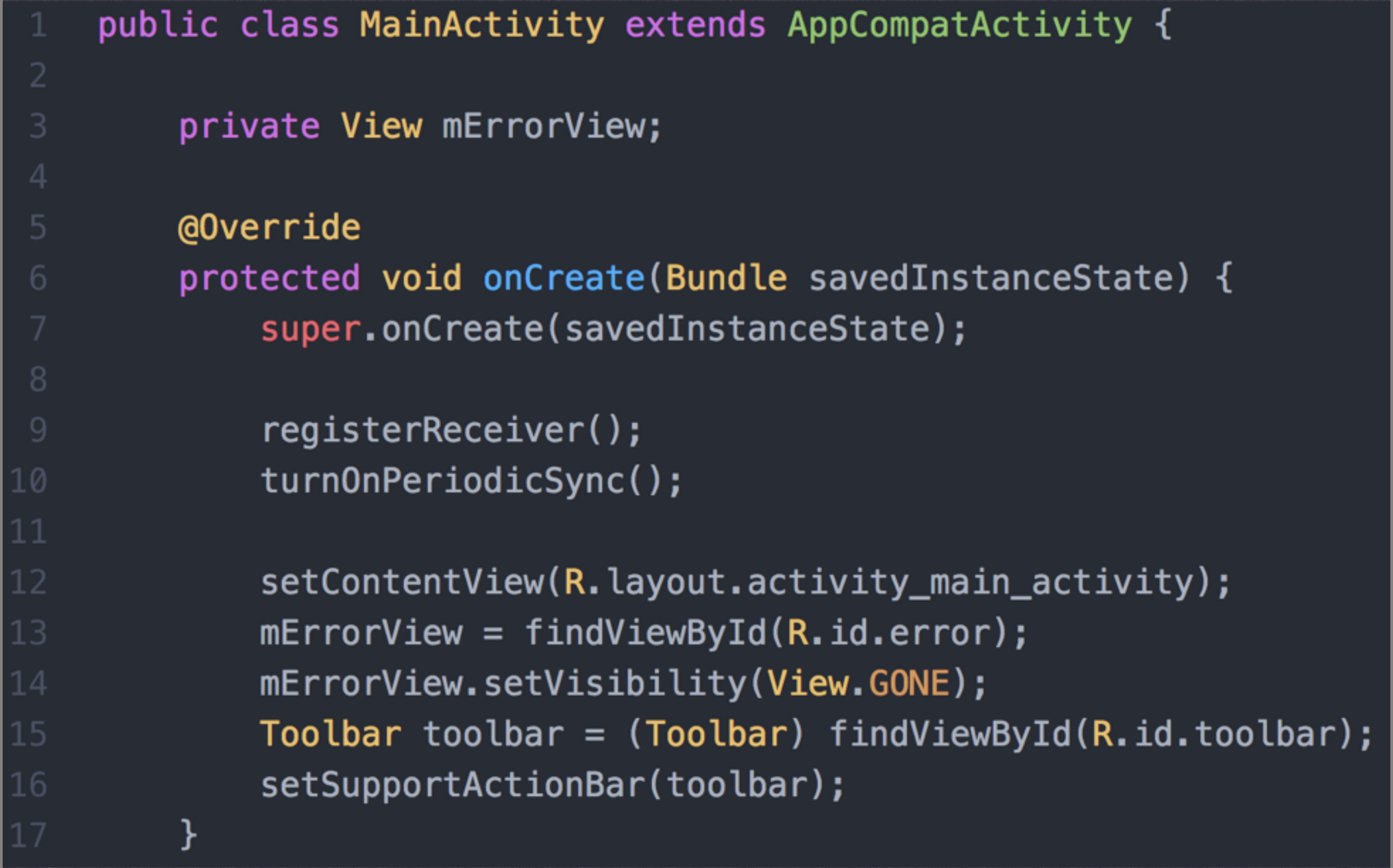
KOTLIN ANDROID EXTENSIONS

Code d'Armor

Kotlin Android Extensions

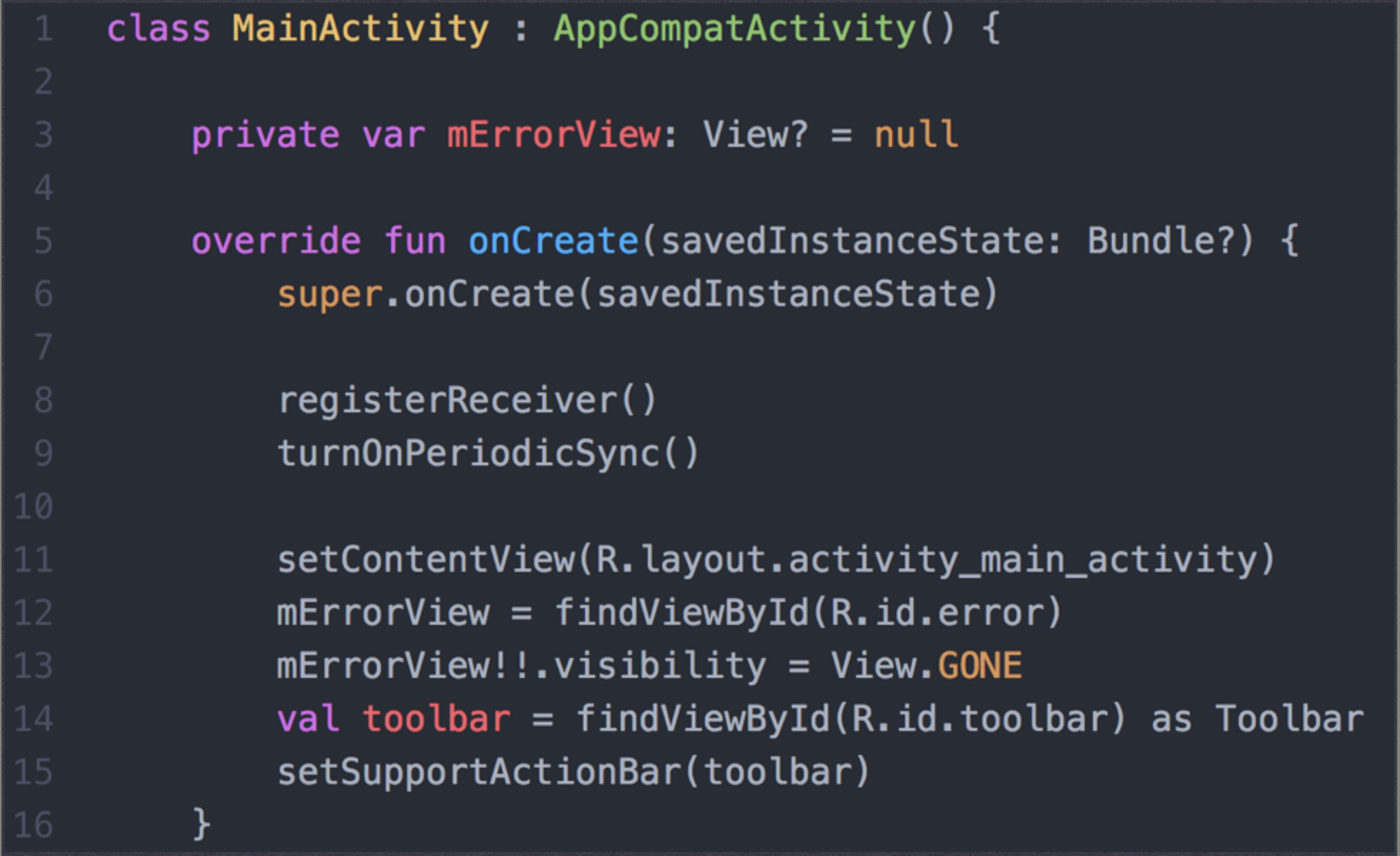
- end of findViewById(R.id.wtf_is_the_id_of_the_view)
- pas de lib à ajouter
- apply plugin: 'kotlin-android-extensions'

Code d'Armor



```java
public class MainActivity extends AppCompatActivity {

    private View mErrorView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        registerReceiver();
        turnOnPeriodicSync();

        setContentView(R.layout.activity_main_activity);
        mErrorView = findViewById(R.id.error);
        mErrorView.setVisibility(View.GONE);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
    }
```

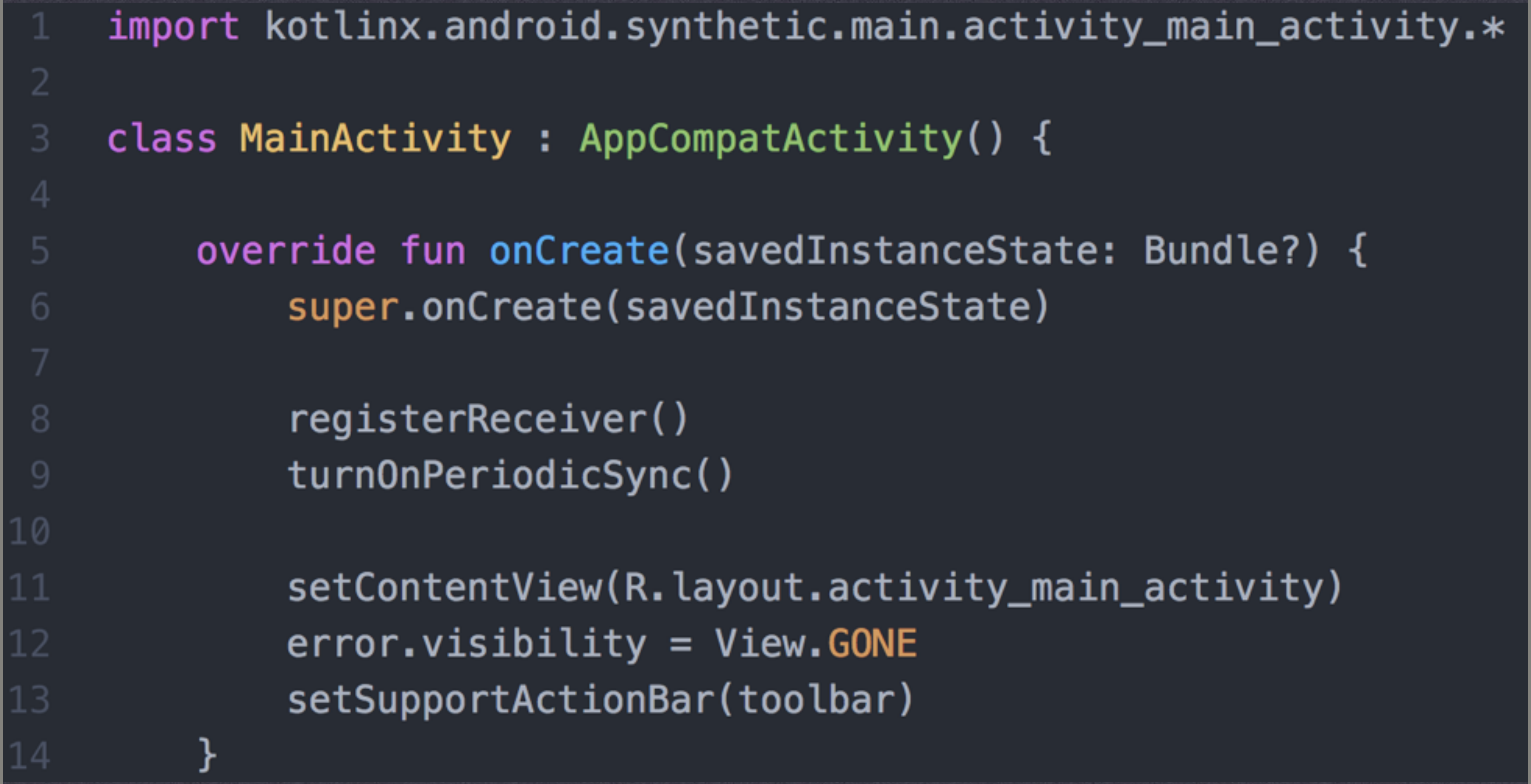
KOTLIN ANDROID EXTENSIONS



```kotlin
class MainActivity : AppCompatActivity() {

    private var mErrorView: View? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        registerReceiver()
        turnOnPeriodicSync()

        setContentView(R.layout.activity_main_activity)
        mErrorView = findViewById(R.id.error)
        mErrorView!!.visibility = View.GONE
        val toolbar = findViewById(R.id.toolbar) as Toolbar
        setSupportActionBar(toolbar)
    }
```

KOTLIN ANDROID EXTENSIONS

Code
d'Armor



```kotlin
import kotlinx.android.synthetic.main.activity_main_activity.*

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        registerReceiver()
        turnOnPeriodicSync()

        setContentView(R.layout.activity_main_activity)
        error.visibility = View.GONE
        setSupportActionBar(toolbar)
    }
```

KOTLIN ANDROID EXTENSIONS

Code
d'Armor

QUI S’EN SERT ALORS?

Code d’Armor

Square

JET BRAINS

Prezi

Code d'Armor

Plus d'infos

- https://kotlinlang.org
- http://try.kotlinlang.org/
- http://blog.jetbrains.com/kotlin/

MERCI

@MARCPOPPLETON

Code
d'Armor