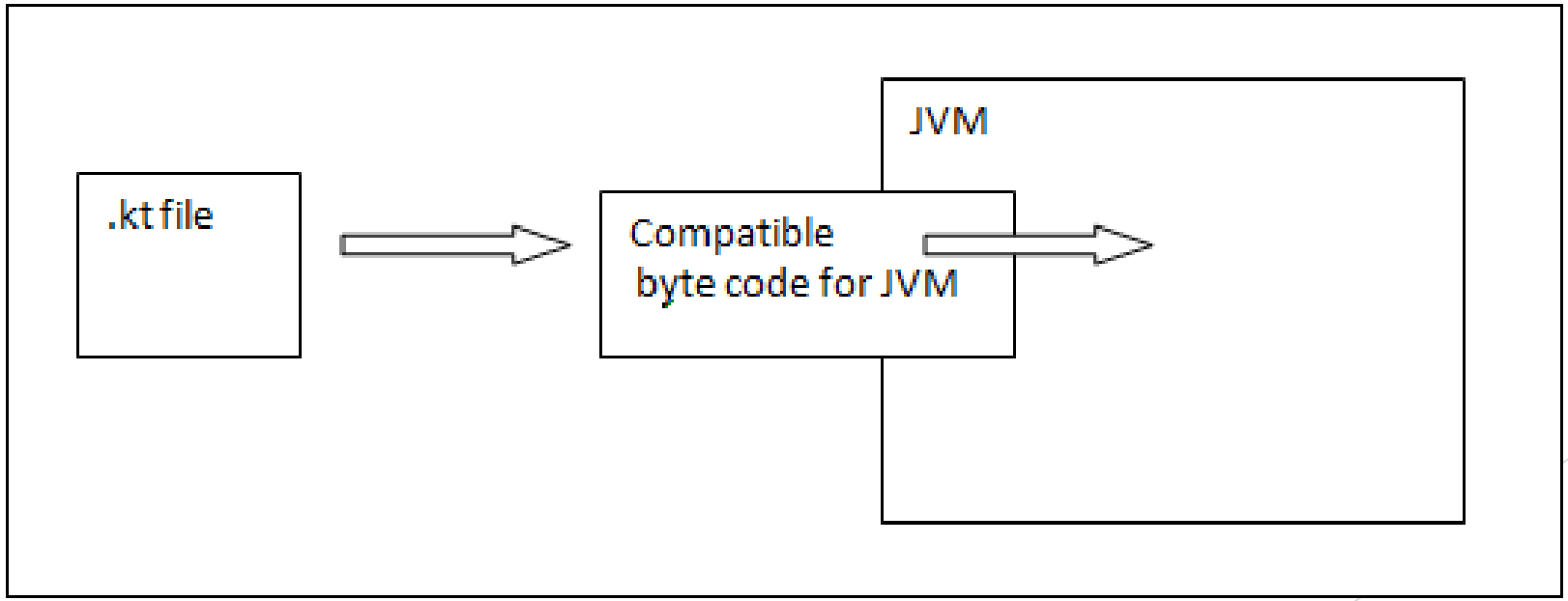
Kotlin

Introduction:

- Kotlin is a programming language introduced by JetBrains, the official designer of the most intelligent Java IDE, named Intellij IDEA. This is a strongly statically typed language that runs on JVM.
- In 2017, Google announced Kotlin is an official language for android development. Kotlin is an open source programming language that combines object-oriented programming and functional features into a unique platform.

Architecture:

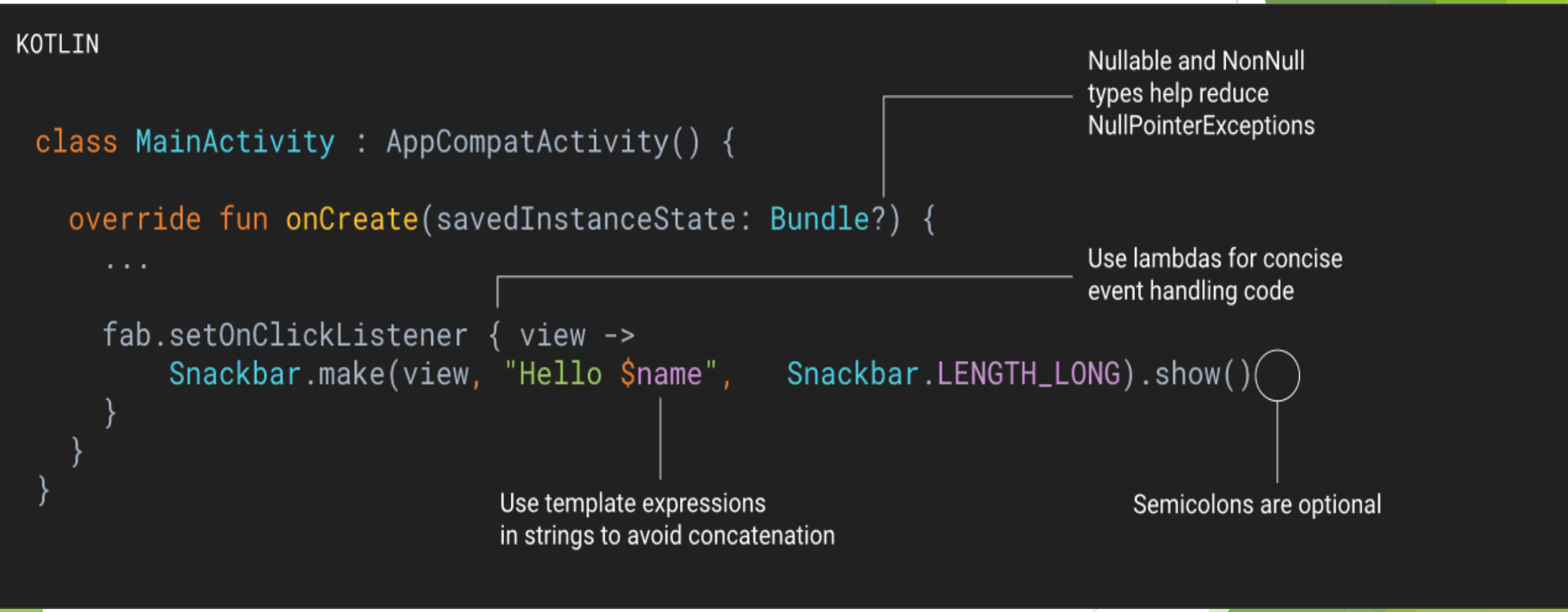
Kotlin compiler creates a byte code and that byte code can run on the JVM, which is exactly equal to the byte code generated by the Java **.class** file. Whenever two byte coded file runs on the JVM, they can communicate with each other and this is how an interoperable feature is established in Kotlin for Java.



Advantages of Kotlin language:

- Easy to learn – Basic is almost similar to java. If anybody has worked in java then easily understand in no time.
- Kotlin is multi-platform – Kotlin is supported by all IDEs of java so you can write your program and execute them on any machine which supports JVM.
- It’s much safer than Java.
- It allows using the Java frameworks and libraries in your new Kotlin projects by using advanced frameworks without any need to change the whole project in Java.
- Kotlin programming language, including the compiler, libraries and all the tooling is completely free and open source and available on github.

What does Kotlin code look like?



Java

```
public class MyActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity);
  }
}
```

Kotlin

```
class MyActivity : AppCompatActivity() {
  override fun onCreate(savedInstanceState: Bundle?)
  {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity)
  }
}
```

Java

```
FloatingActionButton fab = (FloatingActionButton)
findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View view) {
    ...
  }
});
```

Kotlin

```
val fab = findViewById(R.id.fab) as FloatingActionButton
fab.setOnClickListener {
  ...
}
```

Variable declaration

Kotlin uses two different keywords to declare variables:
val and **var.**

- Use val for a variable whose value never changes. You can't reassign a value to a variable that was declared using val.
- Use var for a variable whose value can change.

In the example below, count is a variable of type Int that is assigned an initial value of 10:

```
var count: Int = 10
```

The var keyword means that you can reassign values to count as needed. For example, you can change the value of count from 10 to 15:

```
var count: Int = 10
count = 15
```

Java vs Kotlin

```
public class Station {
    private String id;
    private String address;

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
}
```

Java vs Kotlin

In Kotlin:

```
data class Station(val id: String, val address:String)
```

Here, we can see very well that there is no need to generate the getters, setters and write the implementations of toString(), equals() methods, etc.

Reference Links

https://swayam.gov.in/

https://storage.googleapis.com/uniquecourses/online.html

https://spoken-tutorial.org/tutorial-search/?search_foss=Android+app+using+Kotlin&search_language=English

https://developer.android.com/kotlin

https://kotlinlang.org/docs/tutorials/kotlin-for-py/introduction.html

https://www.tutorialspoint.com/kotlin/index.htm