In this lecture, we will learn more about **Kotlin programming language**. Kotlin and Java are having the similar structure, however, the structure of Kotlin is much more simple, thus users learn it easier and faster. Moreover, Kotlin has some features which Java doesn't, therefore users are able to write a program with fewer lines of code compared to Java.

**Let's review some codes in Kotlin:**

## 1. What is variable?

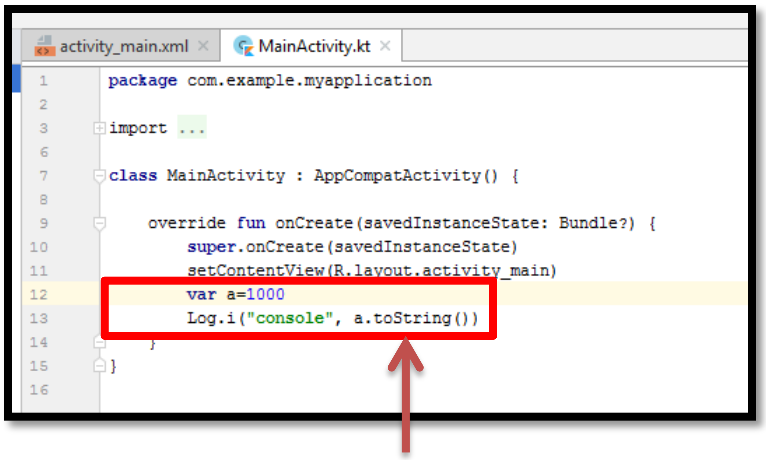
- It is a storage location paired with an associated symbolic name, which contains some quantity of information referred to as a value. The values of **Var** can be changed, depending on conditions or on information passed to the program.

  **Var name = value→ Var x=10**

- **Val** is a storage location paired with an associated symbolic name, which contains some quantity of information referred to as a value, **however**, the values of **Val** can**not** be changed once it has been assigned!

  **Val name = value→ Val pi=3.14 (the value of pi is constant)**

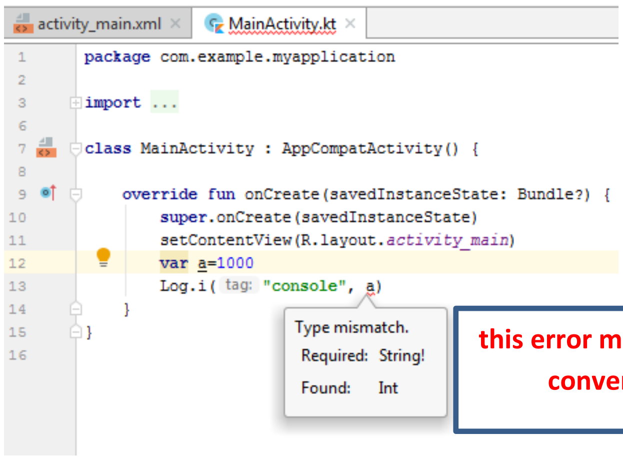
Sample: write the codes in the MainActivity.kt window.



```
package com.example.myapplication

import ...

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var a=1000
        Log.i("console", a.toString())
    }
}
```

The **Log.i()** method is used to **log** informational messages.

By default **Var/Val** recognizes the type of the assigned data, however, in order to see the result, it is required to convert it to **String** data type. Thus, we need to type: **toString()**.

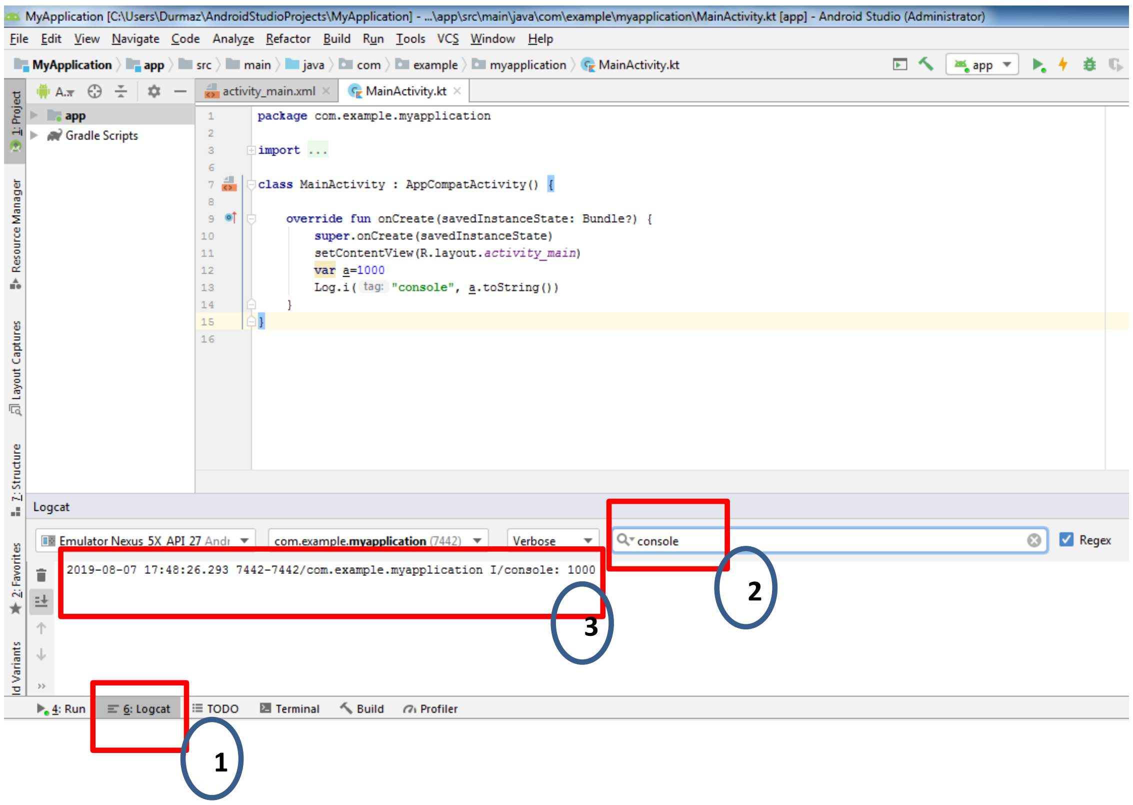


```
package com.example.myapplication

import ...

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var a=1000
        Log.i( tag: "console", a)
    }
}
```

```
Type mismatch.
 Required:  String!
 Found:     Int
```

this error message appears in case of not converting the value **to String**

To check the result on the Logcat, click on the Logcat, then type the tag name “console”, click on the RUN button to check the result. When you click on the **RUN** button, the android studio will create an APK (Android Programming Package) for the application and then shows the result.


MyApplication [C:\Users\Durmaz\AndroidStudioProjects\MyApplication] - ...\app\src\main\java\com\example\myapplication\MainActivity.kt [app] - Android Studio (Administrator)
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 myapplication 〉 MainActivity.kt
app
Gradle Scripts
activity_main.xml
MainActivity.kt
Logcat
Emulator Nexus 5X API 27 Andr
com.example.myapplication (7442)
Verbose
console
Regex
4: Run
6: Logcat
TODO
Terminal
Build
Profiler
1
2
3

```
package com.example.myapplication

import ...

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var a=1000
        Log.i( tag: "console", a.toString())
    }
}
```

```
2019-08-07 17:48:26.293 7442-7442/com.example.myapplication I/console: 1000
```

**Exercise1: Write a program to show the following result using 4 and 2 as variables.**

| | |
|---|---|
| Console: 6 | (4+2) |
| Console: 2 | (4-2) |
| Console: 2 | (4/2) |
| Console: 8 | (4*2) |

# Sample print code for Strings

| Input | Output |
|---|---|
| `var str= "welcome to android studio"` | welcome to android studio |
| `var str= "welcome to \”android studio\”"`<br><br>\symbol String \symbol, backslash is used to add any symbol in the string.<br><br>Ex: “welcome to \\android studio\\” | welcome to “android studio”<br><br>Output: welcome to \android studio\ |
| `var sub= str.subSequence(11,18)`<br><br>subsequence means substring<br>ex: “android” is a substring of this string-> “welcome to android studio” | android |
| `var len=str.length`<br><br>length shows the length of the string | 30 |
| `var a= str.contains("android")`<br><br>contains checks whether the substring “android” is included in the string of “welcome to android studio” or not. The result is a Boolean value, therefore it shows true/false. | True |
| `var name= "ali"`<br>`var surname= "hakan"`<br>`var fullname= "my name is $name $surname"` | my name is ali hakan |

## 2. Conditional Statements

### a. What is if?

i. An **if** the statement is a programming conditional statement that, if proved true, performs a function or displays information.

```
var a=4

var b=8


if(a == b)
 {

Log.i("console", "a is equal to b")

 }
```

```
If (condition)
{
do something if the condition is true
}
```

```
If (condition)
{
do something if the condition is true
}
else {
    do something else if none of the conditions are true
  }
```

```
If (condition1 && condition2)
{
do something if the condition is true
}
```

```
If (condition)
{
do something if this condition is true
}
else If (condition)
  {
    do something if this condition is true
  }
else If (condition)
  {
    do something if this condition is true
  }
else {
    do something else if none of the conditions are true
  }
```

**Different types for operators**

| | |
|---|---|
| **==** | **→ equal** |
| **>=** | **→greater or equal** |
| **<=** | **→ less or equal** |
| **!=** | **→ not equal** |
| **>** | **→ greater** |
| **<** | **→ less** |
| **&&** | **→ AND** |
| **\|\|** | **→ OR** |

**Exercise2: Assume that you want to buy a product, in order to make a purchase, you should check whether the account balance is enough or not.**

```
Var account_balance= 10000
Var product_price = 2500
    if (?)
        {
            -----------------???
        }
```

**Exercise3: Write a program to check the weather and show a message.**

less than 20 degree = weather is cold

21-25 degree = weather is good

26-30 = weather is cool

31-40 = weather is hot

none of above = no information

## b. What is When?

when is a conditional statement which is only available in Kotlin language.

When replaces the switch operator of C-like languages. In the simplest form, it looks like this

```
when (x) {
    1 -> print("x == 1")
    2 -> print("x == 2")
    else ->      { // Note the block
        print("x is neither 1 nor 2")
            }
        }
```

```
// program to show weekdays
var day=4
 when (day){
     0 -> Log.i("console", "saturday")
     1 -> Log.i("console", "sunday")
     2 -> Log.i("console", "monday")
     3 -> Log.i("console", "tuesday")
     4 -> Log.i("console", "wednesday")
     5 -> Log.i("console", "thursday")
     6 -> Log.i("console", "friday")
     else -> Log.i("console", "invalid numbr")
 }
```

```
// program to check range of numbers and show a message.
//  in range1..range2 -> print the message
// !in range1..range2 -> print the message

var mark=80

 when (mark){
     in 0..40 -> Log.i("console", "F")
     in 41..51 -> Log.i("console", "D")
     in 52..62 -> Log.i("console", "C")
     in 63..73 -> Log.i("console", "B")
     in 74..84 -> Log.i("console", "A-")
     in 85..100 -> Log.i("console", "A")
     else -> Log.i("console", "invalid value")
 }
```

**Exercise 4: Write the exercise 3 program using When statement.**