SYNTAX TECHNOLOGIES

JAVA Review Session

When to use conditional statement?

Conditional Statements are used to insert verification points and error handling.
Two types of Conditional statements in Java

1) if statement

2) switch Statement

What is an if statement?

An if statement is used to check a condition. So, *if* the condition is true, we run a block of statements (called the *if-block*), *else* we process another block of statements (called the *else-block*).

1) verification
2) this or that situation
3) selective execution

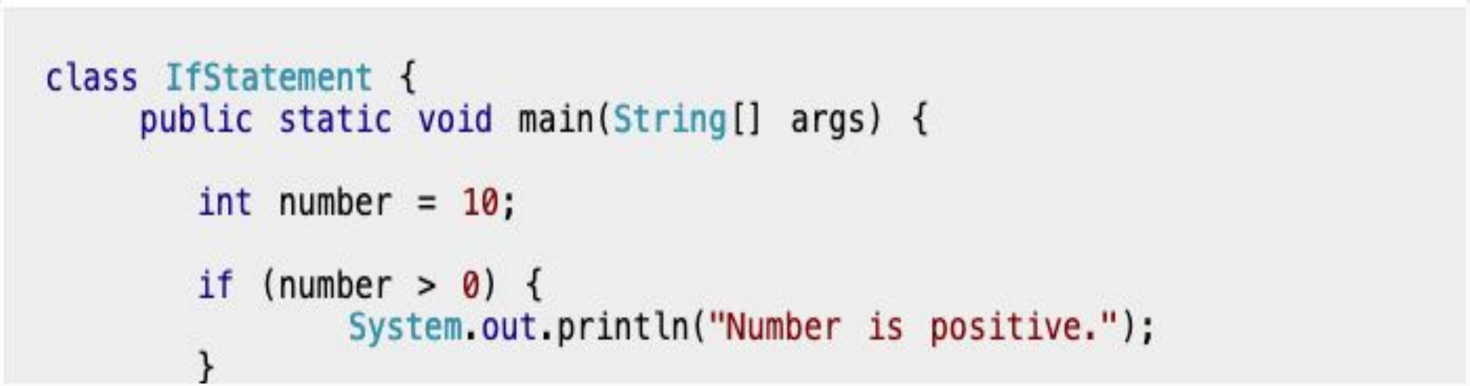
Syntax of if statement

```
if (condition) {
  // block of code to be executed if the condition is true
}
```

Example of if statement

Example:

```
 if(b==2)  {
 System.out.println("The value of b is 2");
}
```



```
class IfStatement {
    public static void main(String[] args) {

        int number = 10;

        if (number > 0) {
            System.out.println("Number is positive.");
        }
```

If else & else if Statement

if-else: The if statement alone tells us that if a condition is true it will execute a block of statements and if the condition is false it won’t. But what if we want to do something else if the condition is false. Here comes the else statement. We can use the else statement with if statement to execute a block of code when the condition is false.

else if - else if statements in Java is like another if condition, it's used in the program when if statement having multiple decisions.

Syntax of if-else statement

```
if (condition) {
  //  block of code to be executed if the condition is true
} else {
  //  block of code to be executed if the condition is false
}
```

Syntax of else-if statement

```
if (condition1) {
  // block of code to be executed if condition1 is true
} else if (condition2) {
  // block of code to be executed if the condition1 is false and condition2 is true
} else {
  // block of code to be executed if the condition1 is false and condition2 is false
}
```

Example of if-else statement

Example:

```
int i = 10;
if (i < 15) {
System.out.println("i is smaller than 15");
}else {
 System.out.println("i is greater than 15");
}
```

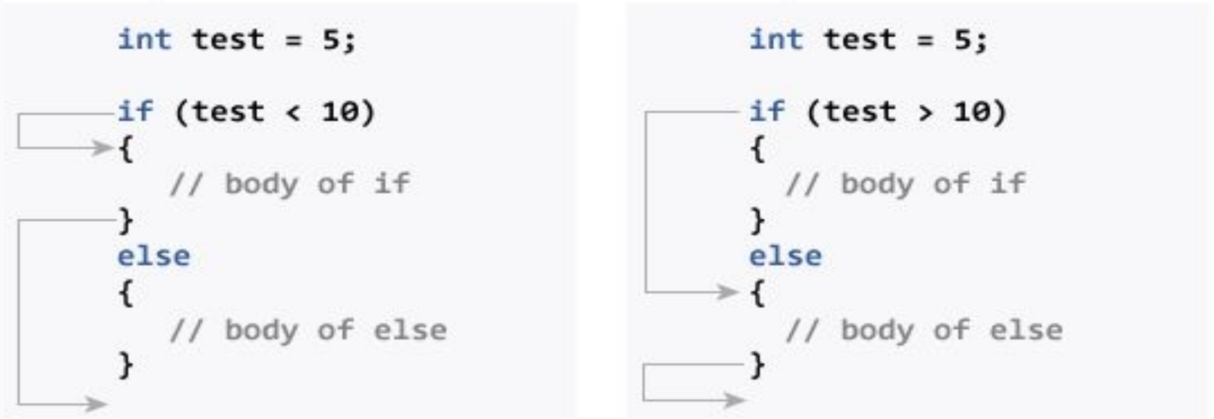
Example of else-if statement

```
public static void main(String args[]) {
        int a = 30, b = 30;

        if (b > a) {
            System.out.println("b is greater");
        }
        else if(a > b){
            System.out.println("a is greater");
        }
        else {
            System.out.println("Both are equal");
        }
```

How if...else statement works?

Expression is true.

Expression is false.



```
int test = 5;

if (test < 10)
{
   // body of if
}
else
{
   // body of else
}
```

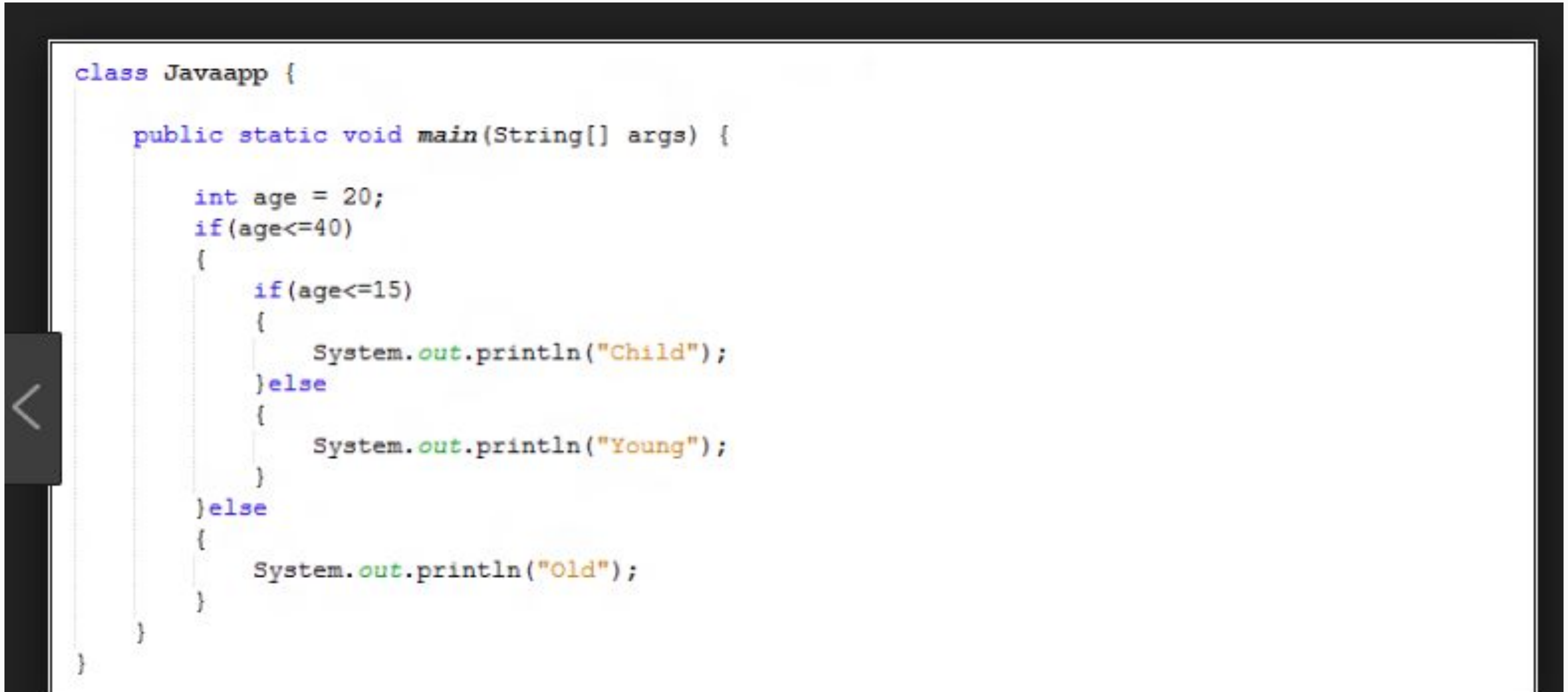
```
int test = 5;

if (test > 10)
{
   // body of if
}
else
{
   // body of else
}
```

Nested-if

Nested-if: A nested if is an if statement that is the target of another if or else. Nested if statements means an if statement inside an if statement. Yes, java allows us to nest if statements within if statements. i.e, we can place an if statement inside another if statement.

Nested-if Syntax

```
if (condition1)
{
   // Executes when condition1 is true
   if (condition2)
   {
      // Executes when condition2 is true
   }
}
```



```
class Javaapp {

    public static void main(String[] args) {

        int age = 20;
        if(age<=40)
        {
            if(age<=15)
            {
                System.out.println("Child");
            }else
            {
                System.out.println("Young");
            }
        }else
        {
            System.out.println("Old");
        }
    }
}
```

What is the Scanner Class ?

Programs typically need some way to read input provided by the user and some way to display results back to the user.
A fundamental part of all programs is user interaction.

The first step in using a Scanner object is setting up an import statement.

In Java, the **import** statement is used to bring certain classes or the entire packages, into visibility. As soon as imported, a class can be referred to directly by using only its name.

Next Step

```
Scanner input; // Declare a Scanner variable
input = new Scanner(System.in); // Create a Scanner object
```

or as a single, combined step:

```
 Create a Scanner object
Scanner input = new Scanner(System.in);
```

Last Step

Once the Scanner is set up, we operate it by calling methods that ask the Scanner to read various kinds of input.

For example, in the program above we asked the Scanner to read a double:

```
double radius = input.nextDouble();
```

Syntax for Scanner Class

```
import java.util.Scanner;

Scanner sc = new Scanner(System.in);
System.out.println("text");
String name = sc.nextLine();
System.out.println("text "+name);
```

Examples of Scanner Class

```
Scanner scan= new Scanner(System.in);
System.out.println(“What is your name”);
String name = scan.nextLine();
System.out.println(“Your name is : ”+name);
```