Flow Control Statements

Chapter-5

Λitrich

Flow Control Statements

Conditional statements, expressions, or simply conditionals are features of programming languages that tell the computer to execute certain actions, provided certain conditions are met.

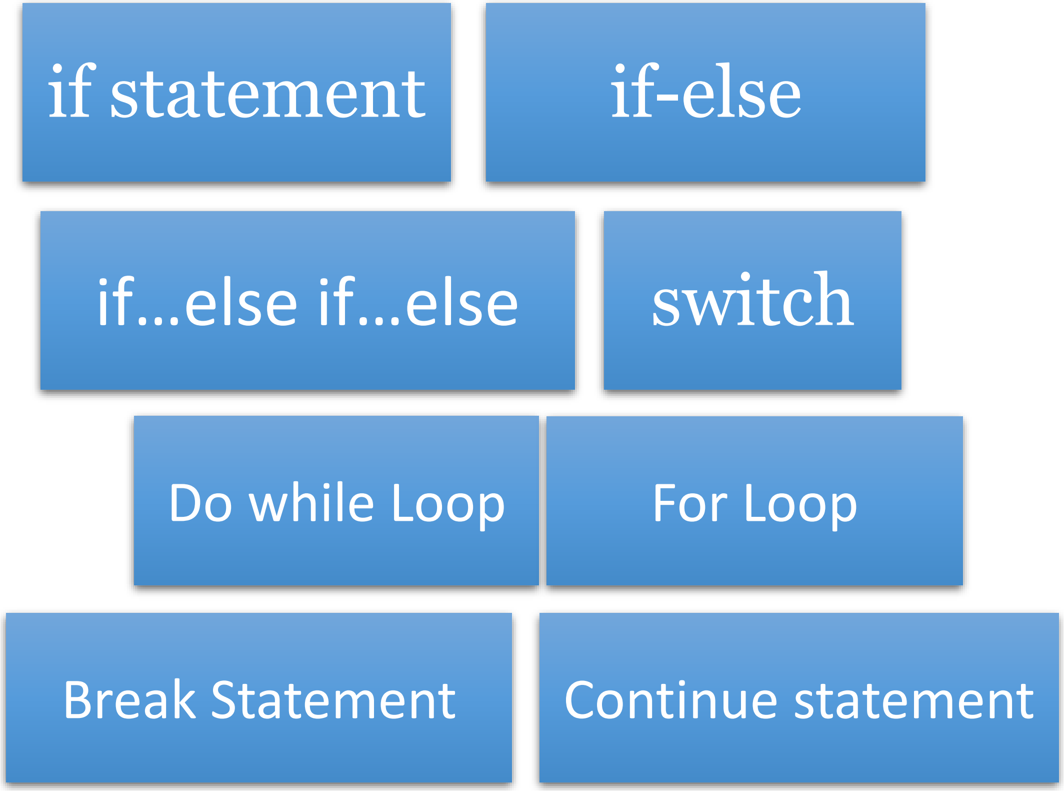
Conditional statements are used through the various programming languages to instruct the computer on the decision to make when given some conditions.

These decisions are made if and only if the pre-stated conditions are either true or false, depending on the functions the programmer has in mind.

Aitrich

Types of flow control statements

- We can use conditional control in TypeScript through the following statements



if statement

if-else

if…else if…else

switch

Do while Loop

For Loop

Break Statement

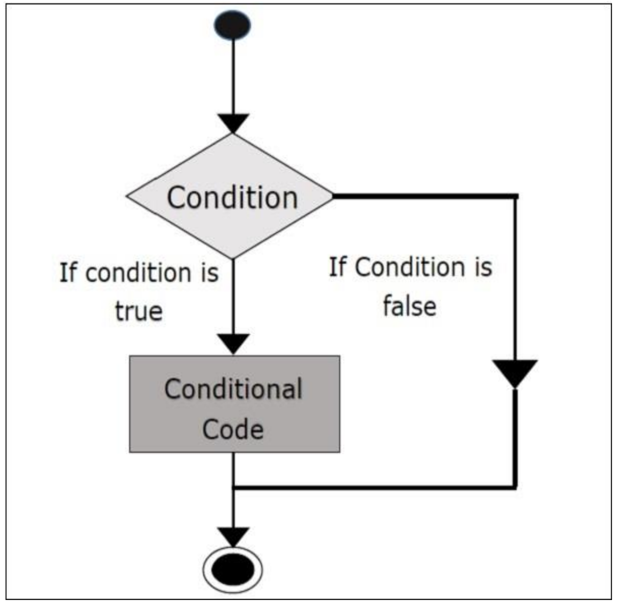
Continue statement

The if statement

- ➢The if statement will evaluate if a condition is true or not.

```
if(boolean_expression) {
   // statement(s) will execute if the
boolean  expression is true
}
```

herw



The if statement Example

```
var  password : string= " lessile007";
var confirm : string= " lessile007 ";
if(password==confirm)
{
        console.log("password & confirm
              password  are     equal");
}
```

if-else statement

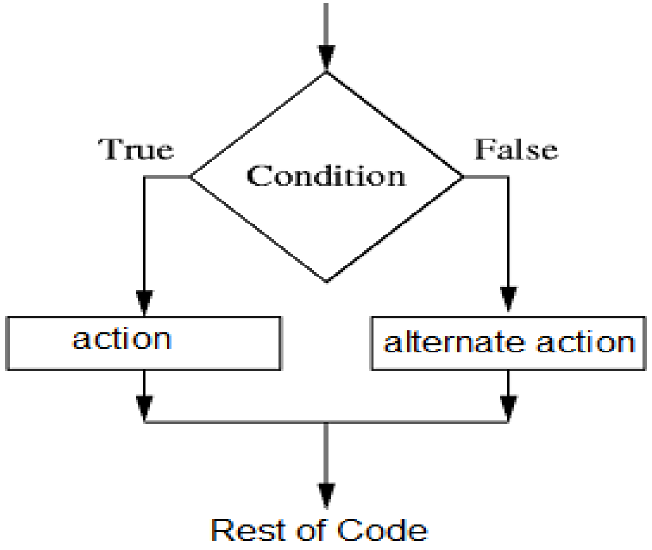
- An if else condition includes two blocks - if block and an else block. If the if condition evaluates to true, then the if block is executed. Otherwies, the else block is executed.

```
if(condition) {

  // code to be executed

} else {

   // code to be executed
}
```



The if-else statement Example

```
var  password : string= " lessile007";
var confirm : string= " lessile007 ";

if(password==confirm)
{

     console.log("password & confirm password are equal");

}
else
{
      console.log("password and confirm pasword not
             match");
}
```

if…else if…else

- The if…else if…else statement can have one or more else if branches but only one else branch.

```
if (condition1) {
 //Statement to execute if Condition1 is true

} else if (condition2) {


  //Statement to execute if Condition2 is true

} else if (condition3) {
  //Statement to execute if Condition3 is true

} else {
  //Statement to execute if all the above conditions are
false
}
```

Nested-If

- A nested if is an if statement, which is present inside the body of another if or else statement.

```
if (condition1)
{
      if (condition2)
   {

     if (condition3)
     {

      //Executes when both condition1 & condition3 is true
   }
   }
   else if (condition4)
   {
      //Executes when both condition1 & condition4 is true
   }
}
```

switch statement

- The Typescript switch statement evaluates a given expression.
- It then matches the result of that expression with the values in the case clause. If it finds a match, then it executes the statements associated with that matching case clause.
- It also executes the statements in cases that follow the matching case. You can break out of a switch using the break statement or using the return statement.

```
switch(expression){
  case expression1:
    //code to be executed;
    break;  //optional
  case expression2:
    //code to be executed;
    break;  //optional
    ........
default:
    //when no case is matched, this block will be executed;
    break;  //optional
}
```

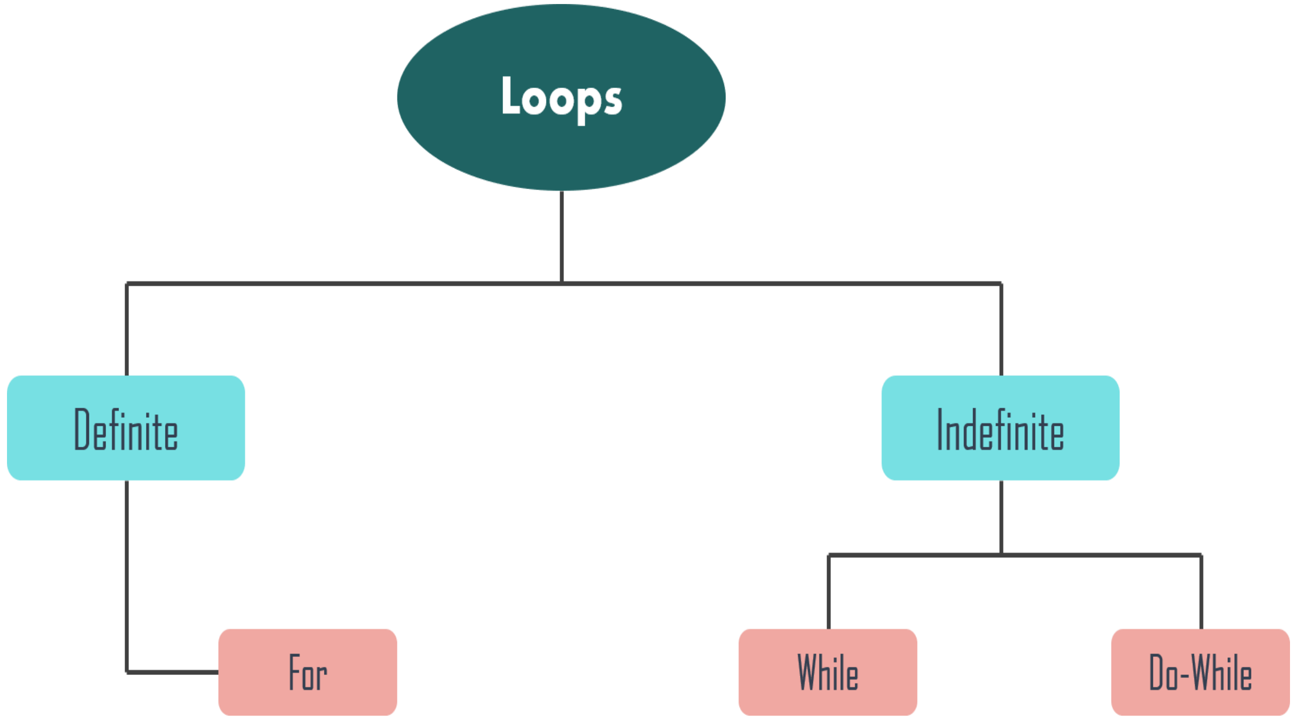
switch statement Example

```
let select-job : number = 4;

switch (select-job) {
    case 1:
        console . log("Java Developer");
        break;
    case 2:
        console . log("Angular Developer");
        break;
         ..........
  default:
        console . log("No such job exists!");
        break;
}
```

TypeScript-Loops

- A loop statement allows us to execute a statement or group of statements multiple times.



Aitrich

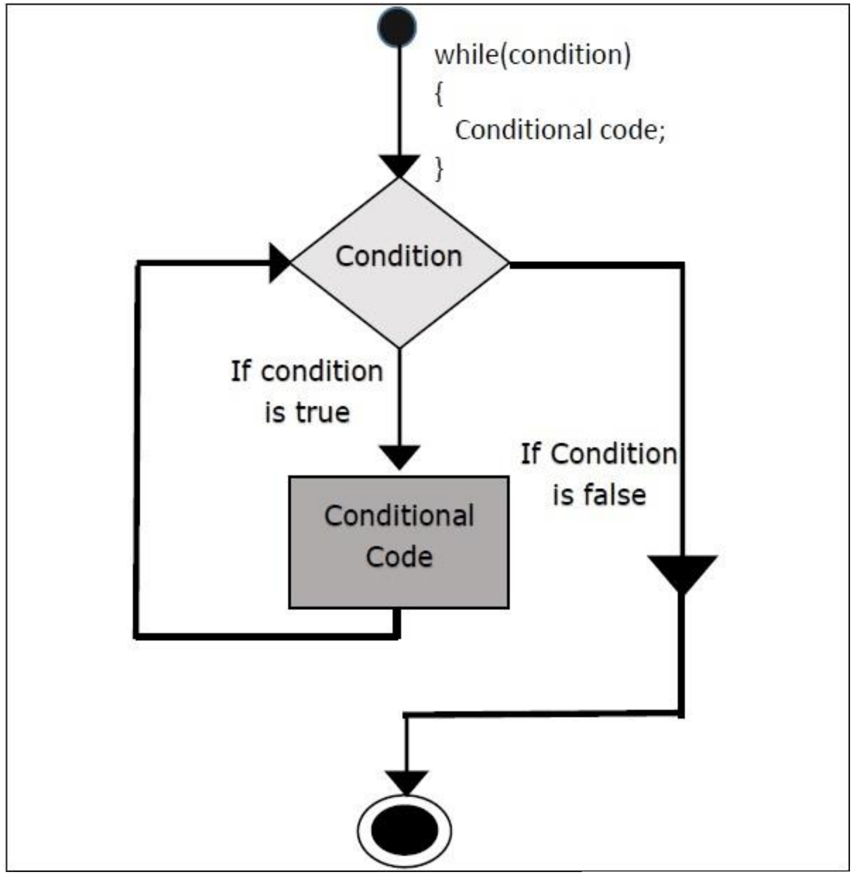
While Loop

- How it works
  - step1:The loop evaluates its condition.
  - step2: If the condition is true then the While body executes. If the condition is false loop ends.
  - step:3Goto to step 2

Aitrich

While Loop Example

```
let i: number = 2;
while (i < 4) {
    console.log( "Block statement
execution no." + i )
    i++;
}
```

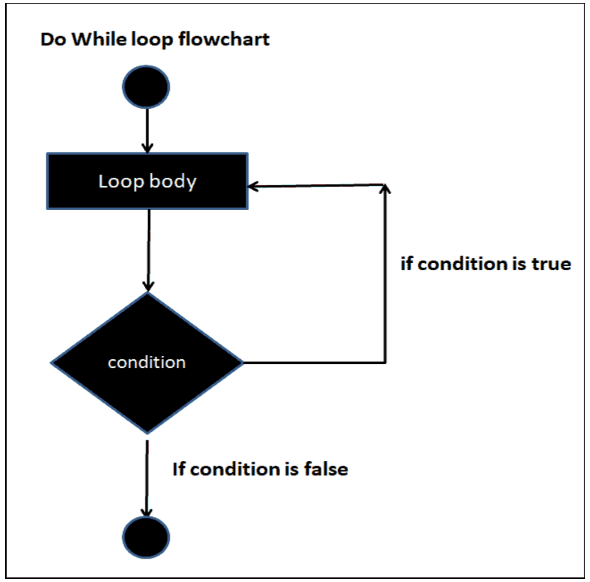


```
while(condition)
{
  Conditional code;
}
```

Do-While Loop

- The do while loop tests its condition after it executes the while body. Hence the while body executes at least once.
  - Step1:The loop body executes
  - Step2:loop evaluates its condition.
  - Step3:If the condition is true the loop body executes. If the condition is false loop ends.
  - Step4:Goto to step 2

```
let i: number = 2;
do {
    console.log( "Block statement execution no." + i )
    i++;
}
while (i < 4) ;
```



Do While loop flowchart

Loop body

condition

if condition is true

If condition is false

For Loop

- It starts with for keyword followed by three optional expressions inside parentheses separated by semi-colons. Finally, a block of statements inside curly braces follows them. These block statements execute for each iteration of the loop.

```
for ([initial-expression]; [condition]; [final-expression])
{ statements }
```

- The three optional expressions are
  - Initial-expression
  - Condition
  - Final-expression

For Loop

When a For loop executes, the following occurs

- step1:Executes the initial-expression. This runs only once
- step2:Evaluates the condition. If the value is True then continue to step 3. If false the loop terminates.
- step3:Executes the statements.
- step:4Executes the final-expression.
- step5:Control returns to Step 2.

```
for (let i = 0; i < 5; i++) {
  console.log(i);
}
console.log("End of Loop");
```

The break statement

The break statement to break out of a For loop.

```
let i = 0;
for (; ; i++) {
  if (i >= 5) break;
  console.log(i);
}
```

The Continue statement

- Use the continue statement to skip the rest of the for loop and move over to the next iteration.

```
➢let i = 0;
➢for (let i = 0; i < 5; i++) {
➢   if (i < 2) continue;
➢   console.log(i);
➢}
```

THANK
YOU