

# TECHNOLOGY



## Caltech

Center for Technology &  
Management Education

### Full Stack Java Developer

# TECHNOLOGY



## Angular



## Getting Started with Angular



# Learning Objectives

By the end of this lesson, you will be able to:

- 👁 Learn how to set up the environment to work with Angular
- 👁 Understand TypeScript to use Angular
- 👁 Illustrate custom HTML elements
- 👁 List the features of Angular core



# A Day in the Life of a Full Stack Developer

You are working in an organization and have been assigned an e-commerce web application. However, since the project has a tough deadline, you decide to make use of a framework that can create platform-based applications and also has in-built features to make the task easy.

You then decide to use Angular and TypeScript. It is fully compatible to create different platform-based applications. It also supports features of server-side programming.

Apart from this, it is an interpreted language and needs to be run to test its validity. It compiles the code.

To use the same, you need to set up Angular, understand its core features, and explore TypeScript in detail.

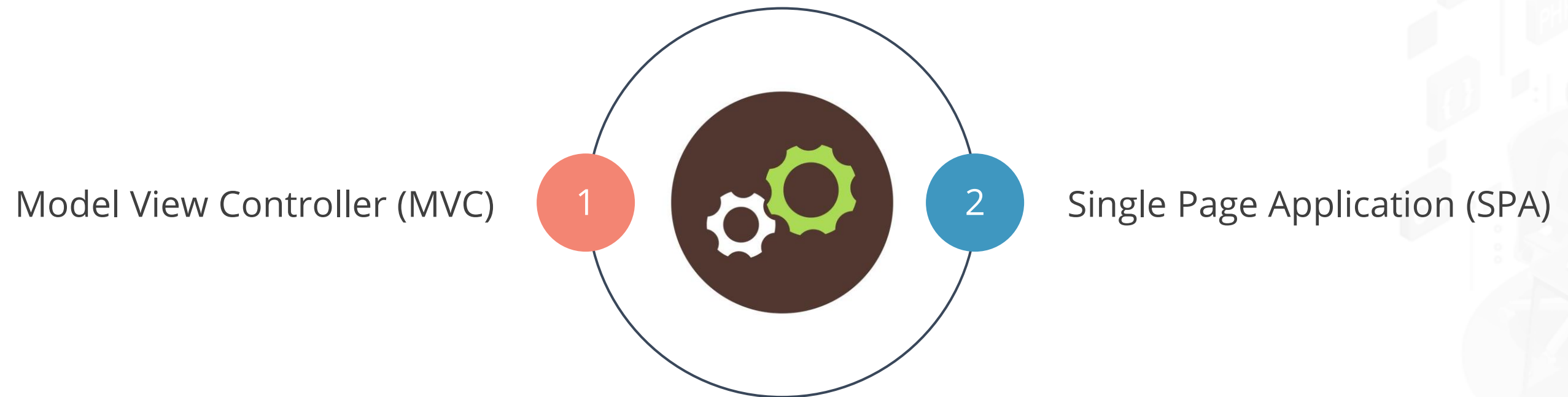




## What Is Angular?

# What Is Angular?

Angular is a client-side TypeScript-based full-stack web framework and a front-end development system.



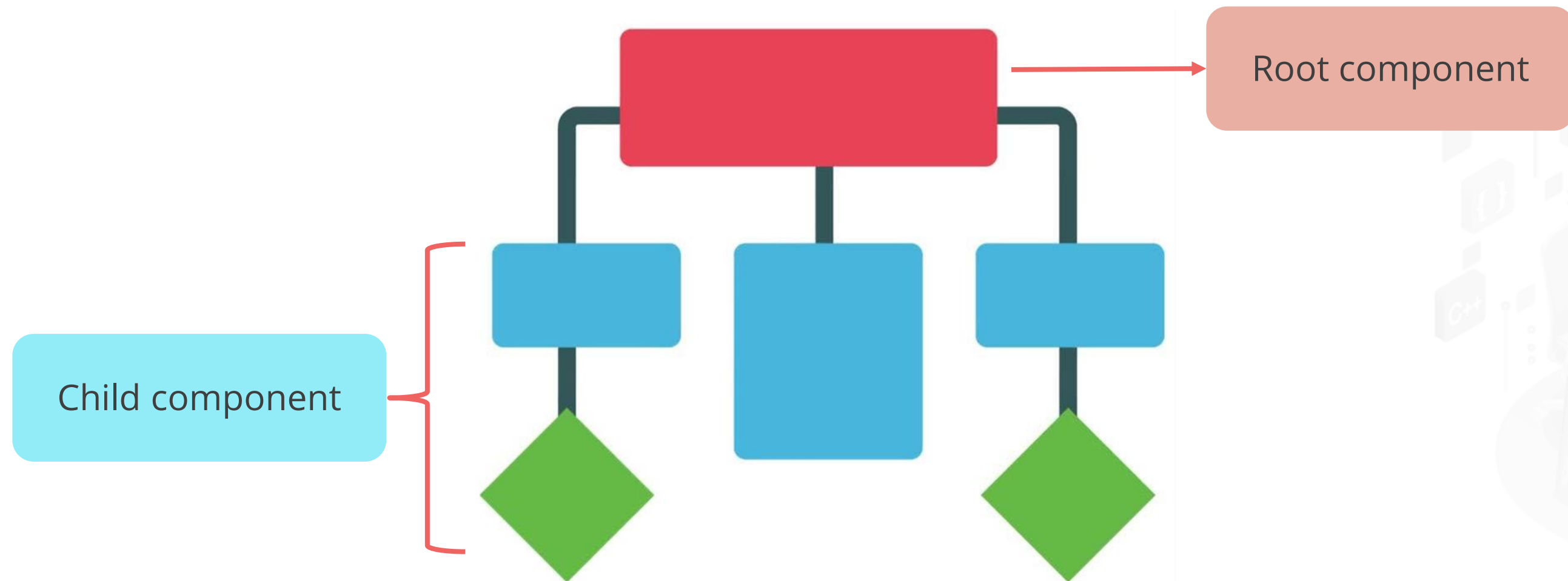
# Advantages of Angular

- 1 It provides a complete collection of well-integrated libraries and developer tools.
- 2 It is fully compatible to create different platform-based applications.
- 3 It consists of a user interface library and developer tools.
- 4 It supports features of server-side programming as well.



# Angular Architecture

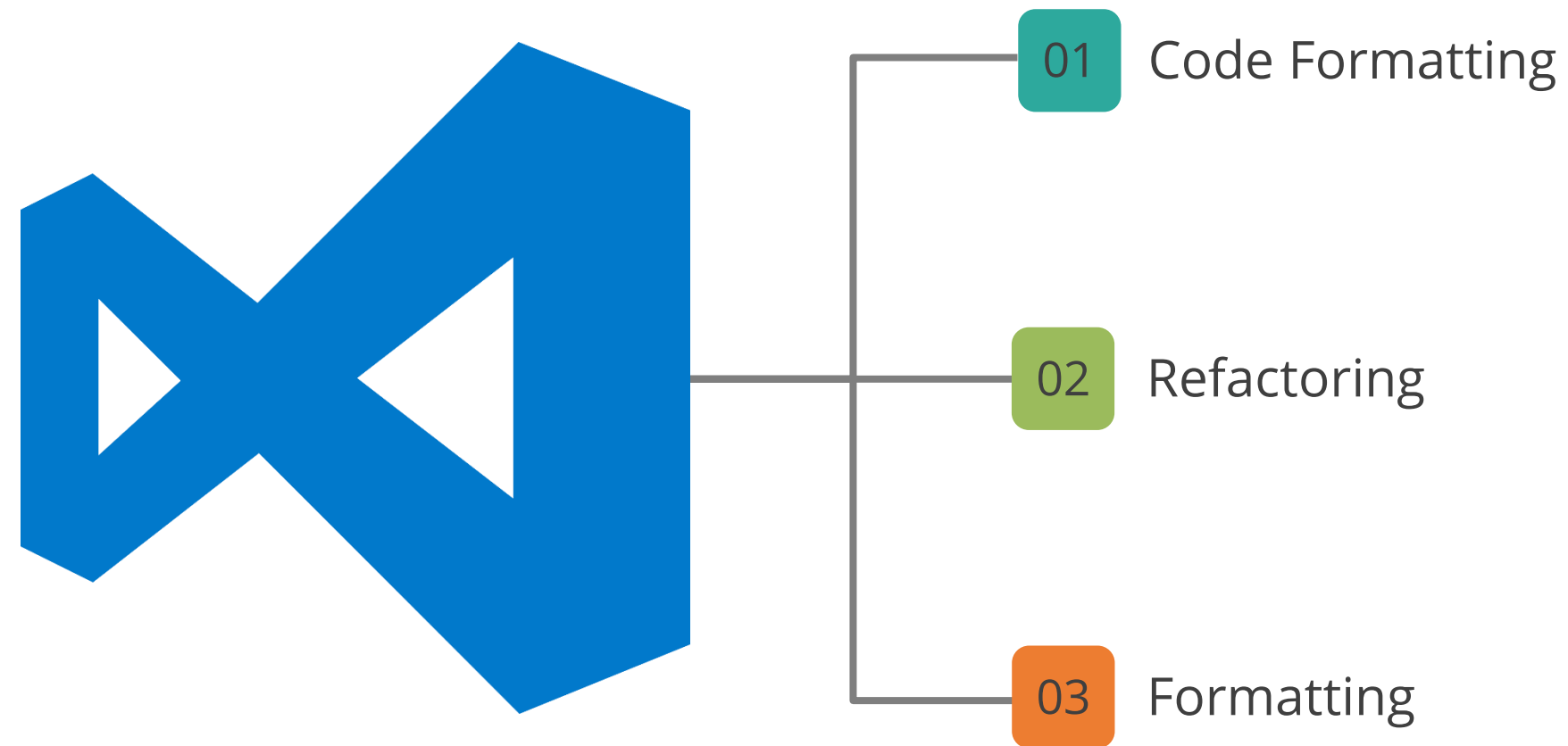
The architecture of an Angular Application is primarily based on the concept of components.



## Angular Installation

# Angular Installation

**Step 1:** Install an IDE(Integrated Development Environment), like Visual Studio Code or JetBrains WebStorm, to run the Angular app





# Angular Installation

JetBrains WebStorm is quick, appealing, and exceptionally simple to use programming.

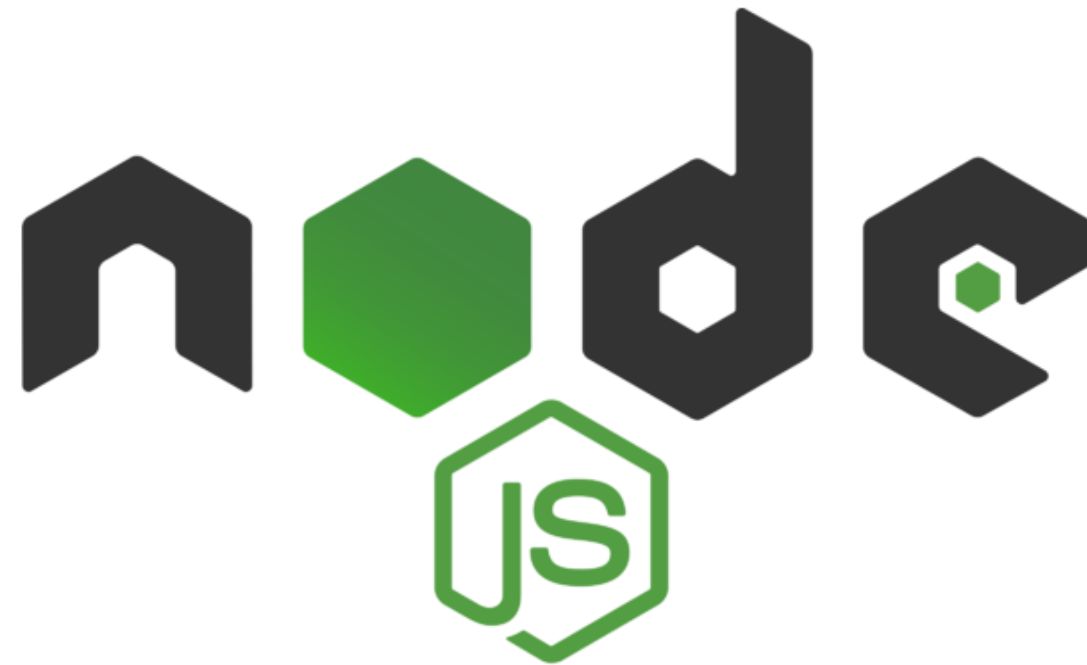


**30 Days Free of Charge**



# Angular Installation

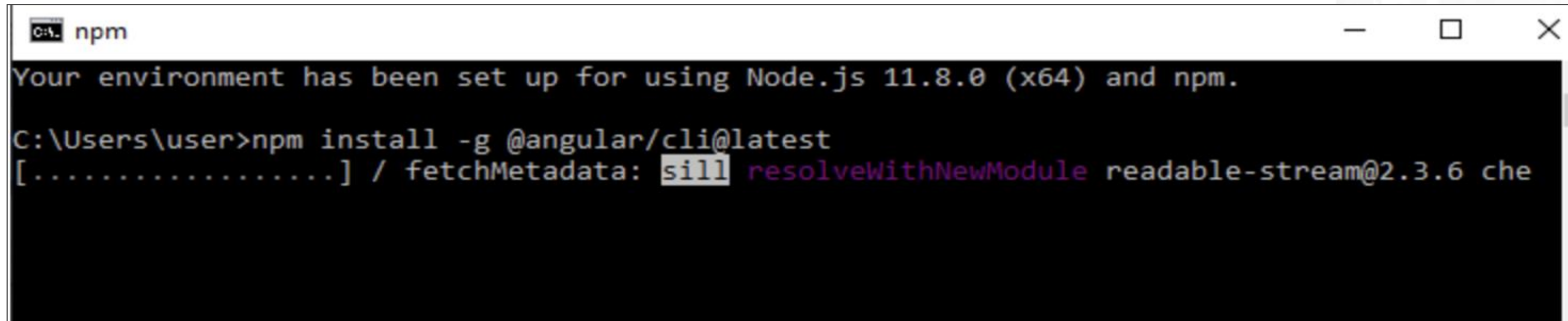
**Step 2:** Node.js provides the maximum libraries to execute the Angular app



# Angular Installation

**Step 3:** To install Angular CLI, run the following command using npm:

```
npm install -g @angular/CLI."
```

A screenshot of a Windows Command Prompt window titled 'C:\> npm'. The window shows the output of the command 'npm install -g @angular/cli@latest'. The first line says 'Your environment has been set up for using Node.js 11.8.0 (x64) and npm.' The second line shows the command being executed: 'C:\Users\user>npm install -g @angular/cli@latest'. The third line shows the progress of the installation: '[.....] / fetchMetadata: sill resolveWithNewModule readable-stream@2.3.6 che'.

```
C:\> npm
Your environment has been set up for using Node.js 11.8.0 (x64) and npm.
C:\Users\user>npm install -g @angular/cli@latest
[.....] / fetchMetadata: sill resolveWithNewModule readable-stream@2.3.6 che
```



## Angular CLI and Bootstrapping

# Angular CLI and Bootstrapping

Angular CLI is used to initialize, develop, and maintain Angular applications directly from a command shell.



It supports creating an Angular application with all the configuration files and packages in one single command.



It also helps to add features to existing Angular applications.



It builds the Angular Application using Typescript, Karma for unit testing, and Protractor for end-to-end testing.



# Angular CLI and Bootstrapping

To install Angular CLI:

```
npm install -g @angular/cli@latest
```

To find the currently installed Angular CLI version:

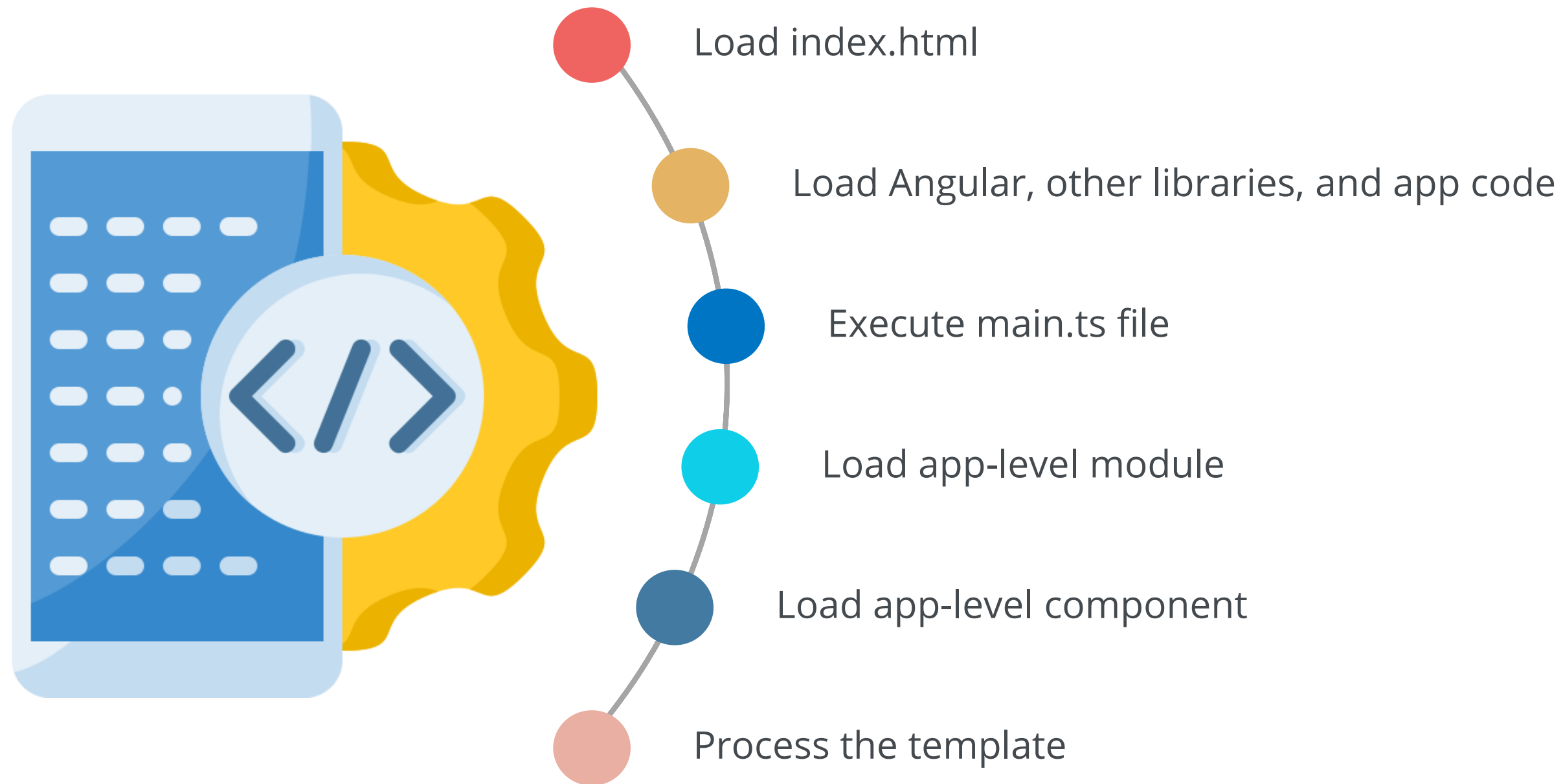
```
ng --version
```

Once the installation is complete, there is a need to initialize and load the angular application to start using it.



# Angular CLI and Bootstrapping

Angular uses certain steps to bootstrap the application, which are:

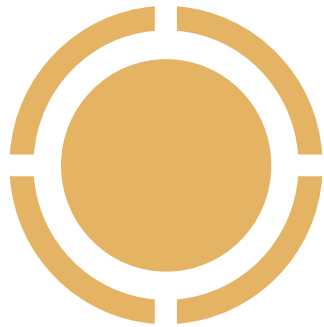


# TECHNOLOGY

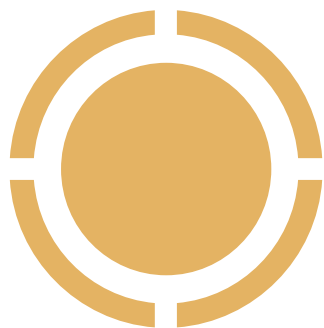
## TypeScript

# TypeScript

Angular is a client-side TypeScript-based full-stack web framework.



TypeScript is a specific, object-oriented, and compiled language.



TypeScript is both a language and a bunch of tools. It is a superset of JavaScript.





# Features of TypeScript



It begins and ends with JavaScript.



Its code is changed over into its JavaScript equivalent.



It can reuse the current JavaScript systems, tools, and libraries.



# Features of TypeScript



Any valid file with a .js extension can be changed to a .ts extension.



It is used across programs, gadgets, and working frameworks.



It does not need a devoted VM or a particular runtime environment to compile.



# Advantages of TypeScript

TypeScript is an interpreted language and needs to be run to test its validity. It compiles the code.



# Advantages of TypeScript

- ✓ It is not strongly typed and contains an optional static typing and types reasoning system through the TSL.
- ✓ It supports Object-oriented programming concepts that include classes, interfaces, and inheritance.

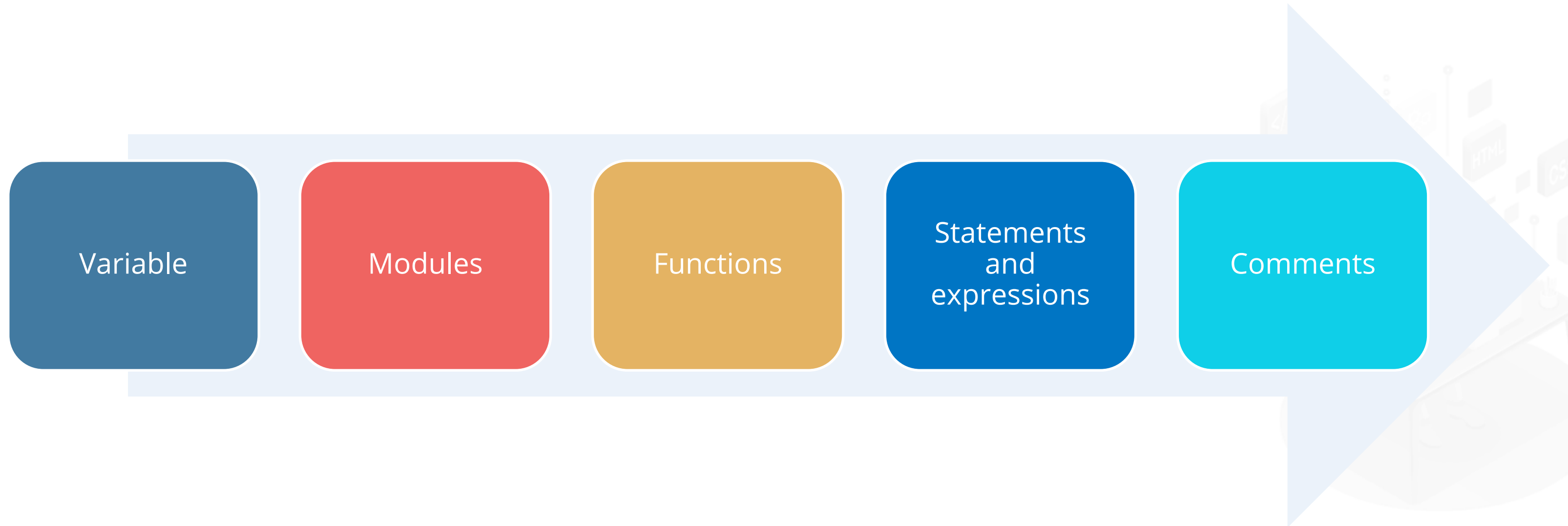
# TECHNOLOGY

## TypeScript Program



# TypeScript Program

TypeScript program consists of:



# Variable

A variable is a place in storage where data or value can be stored.



Variable names can contain upper-case, lower-case alphabets, and digits.



They cannot start with a digit.



\$ and \_ special characters can only be used. No other special characters are allowed.



# Modules

The module is used to create a group of related variables, classes, functions, interfaces, and more.

Modules execute not in global scope but in a local scope.

The module is created using the export keyword.



# Internal Modules

Modules are divided into internal modules and external modules.

## Internal Module

Used for logical grouping of interfaces, classes, functions, and variables into a single unit and export to another module

# Internal Modules

Syntax:

```
Module subtraction{  
  export function sub(a,b){  
    console.log("Difference is : " + (a-b))'  
  }  
}
```





# External Modules

## External Module

It is used to indicate the heap conditions between the various external JavaScript files.



ECMAScript 2015(ES6) module framework regards each file as a module.

**ES6**

# External Modules

Syntax:

```
export class Subtraction{
    constructor(private a?: number, private b?:
number) {
    }
    Subtract() {
        console.log("SUBTRACTION: " +(this.a -
this.b) );
    }
}
```



# Functions

Functions are the essential structure square of any application in JavaScript.

It is used to assemble layers of reflection.

It makes the code coherent, viable, and reusable.

It is used to impersonate classes, data hiding, and modules.

# Function Declaration

It gives the function name, function parameters, and return type to the compiler.

```
Function function_name( [arg_1, arg_2 , ... arg_n ]
```



# Function Definition

It contains actual statements that will be executed.

```
Function function_name([arg_1, arg_2 , ..., arg_N]){  
  //code statements  
}
```





# Function Call

It can be called from anywhere in the code.

```
function_name();
```



# Statements and Expressions

Statements and expressions are used in TypeScript to tell the program what to do and how to do it.



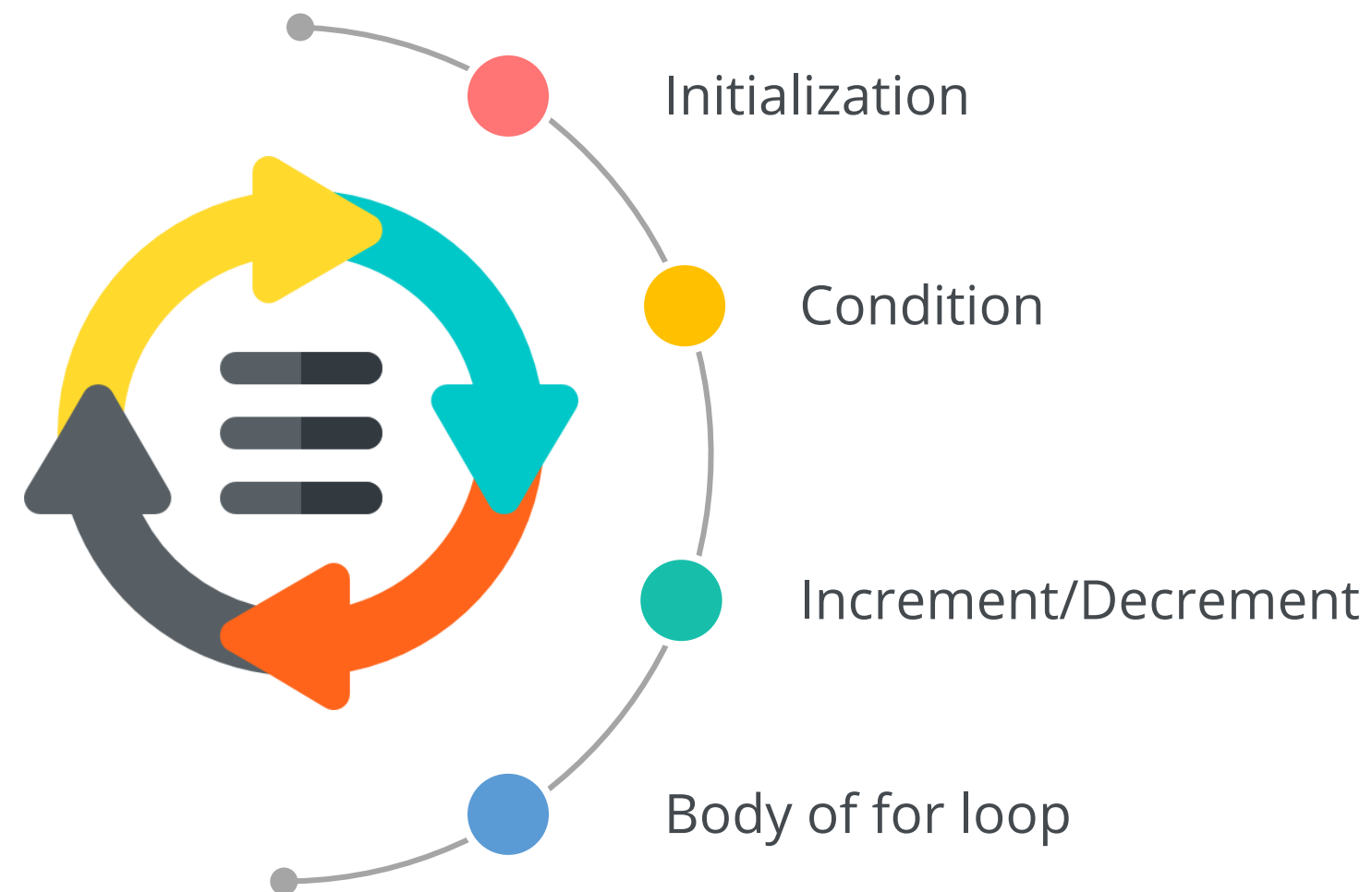
# Looping Statements

Loop statements are used to execute the same block of code at repetition, and to avoid duplicate code.



# For Loop

The four main parts of for loop are:



# For Loop

Syntax:

```
for(initializer; condition; increment/decrement) {  
    // statements  
}
```





# While Loop

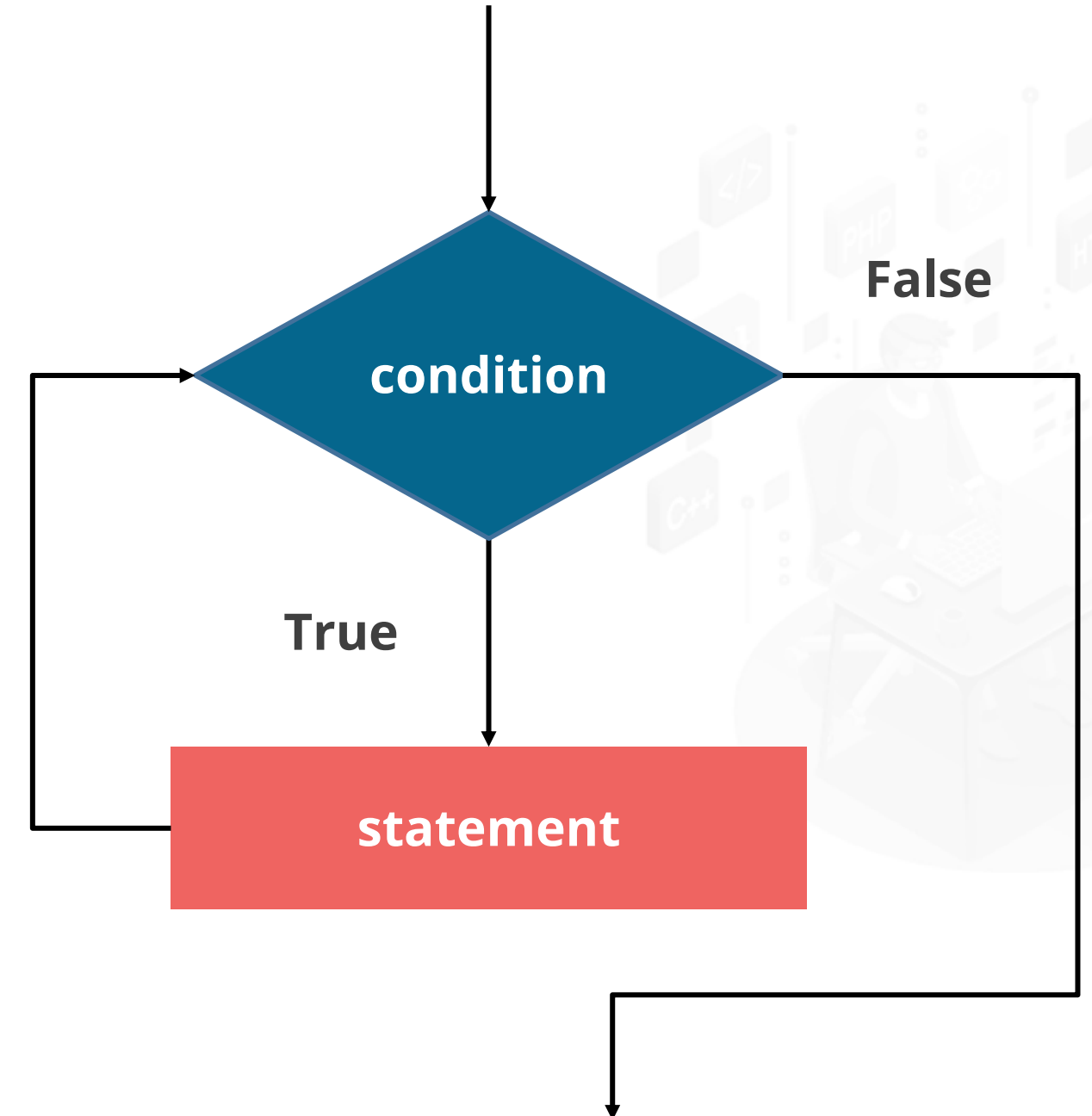
It is used to emphasize a piece of the program over and over until the predetermined Boolean condition is valid.



# While Loop

Syntax:

```
Initializer;  
while(condition) {  
    increment/decrement;  
}
```



## Do-While Loop

It is used to utilized to iterate a piece of the program over and over until the predefined condition is valid.



The do-while looks at the condition toward the finish of the loop body.



# Do-While Loop

Syntax:

```
initializer;  
do {  
    // statements to be executed  
    increment/decrement;  
} while(condition);
```



# Decision Making Statements

The software developers use decision-making to determine at least one condition to be assessed by the program.

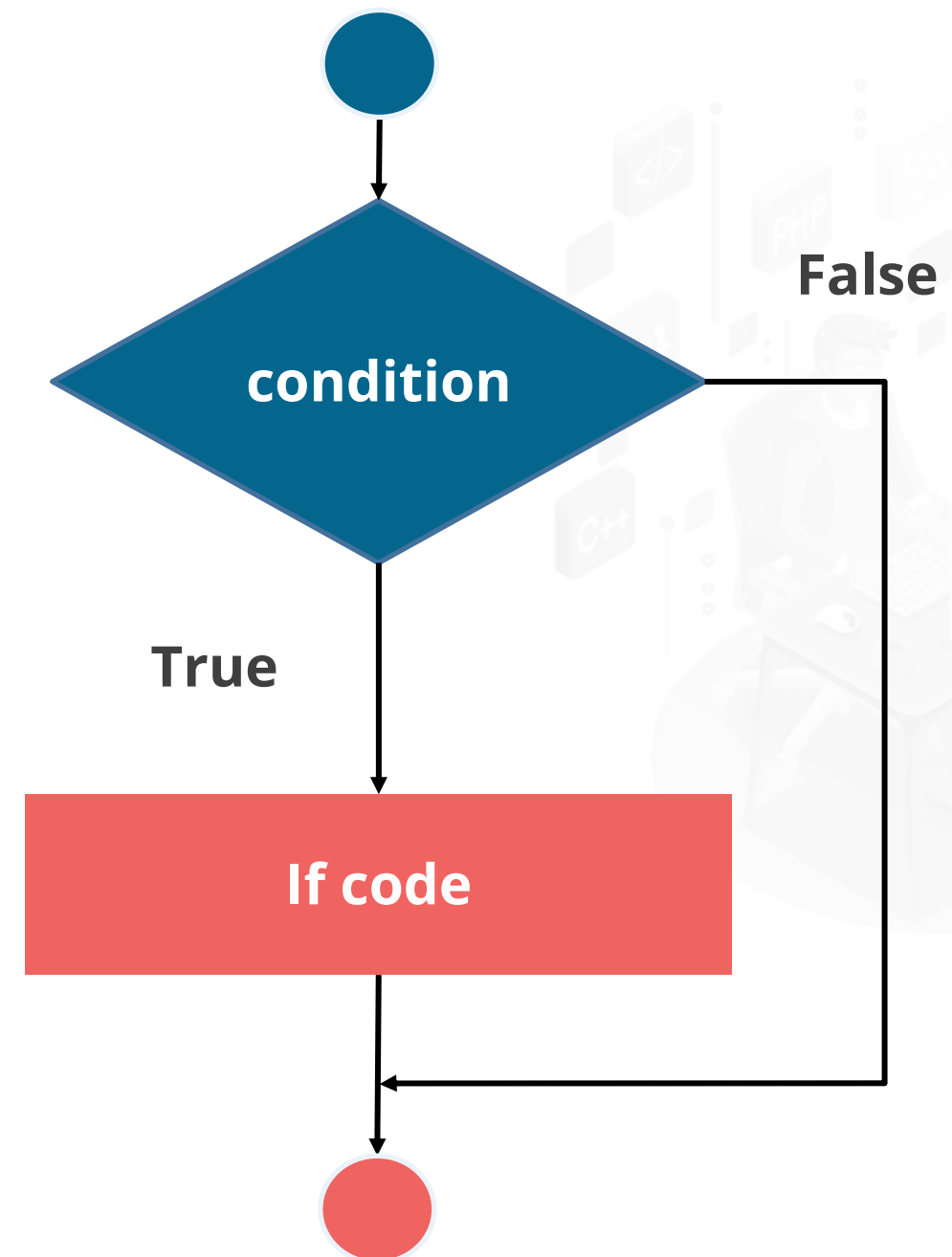
- 1 if statement
- 2 if-else statement
- 3 If-else-ladder statement
- 4 Nested If statement
- 5 switch statement



# If Statement

It decides whether the statement will be executed. This means that it verifies the condition as true if the given condition is true.

```
if(condition) {  
    // statements to be executed  
}
```

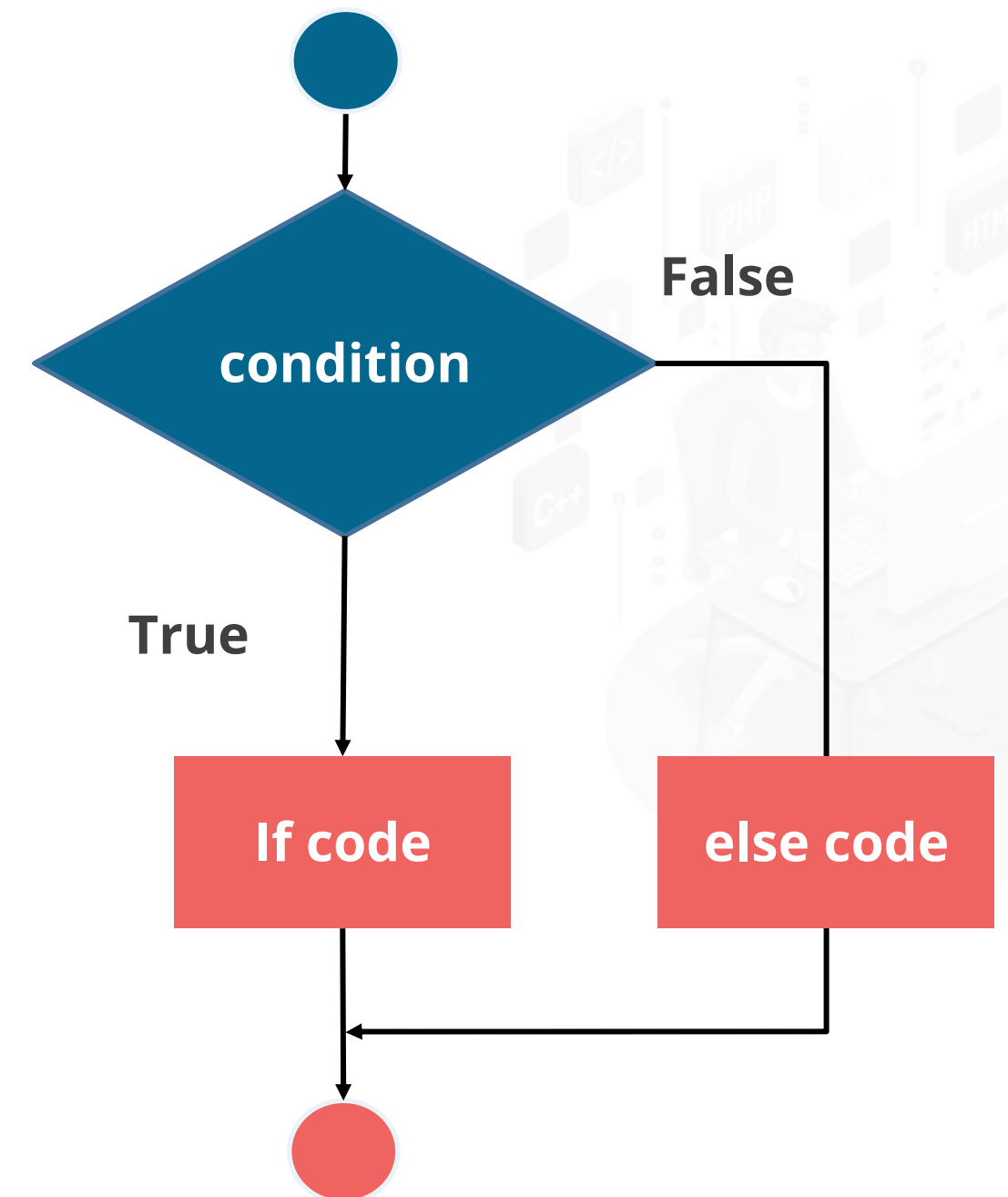




# If-else Statement

Use an if-else statement to return something when the condition is false.

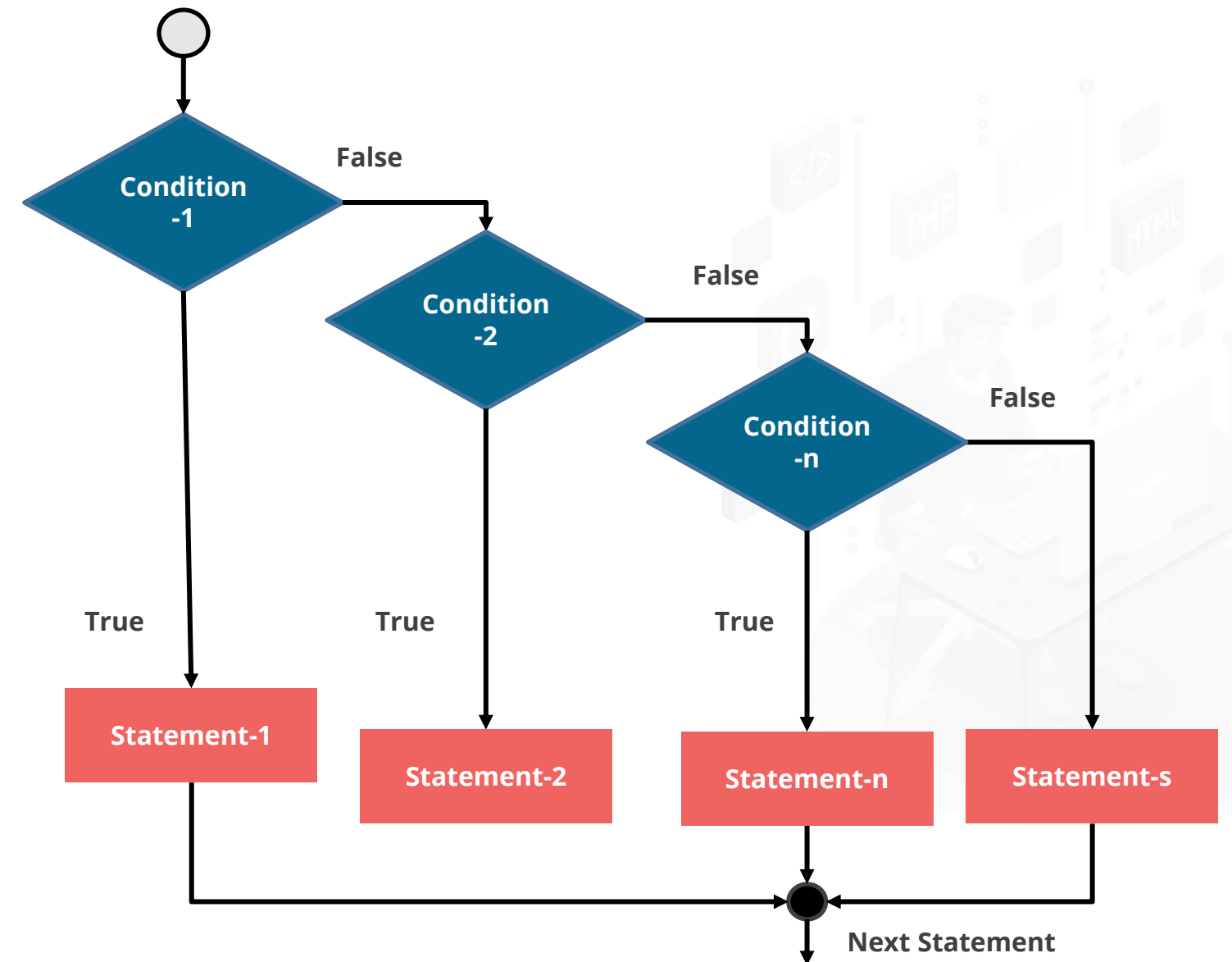
```
if(condition) {  
    // code to be executed  
} else {  
    // code to be executed  
}
```



# If-else Ladder Statement

Users can make choose from multiple options, and it runs in a top-down approach.

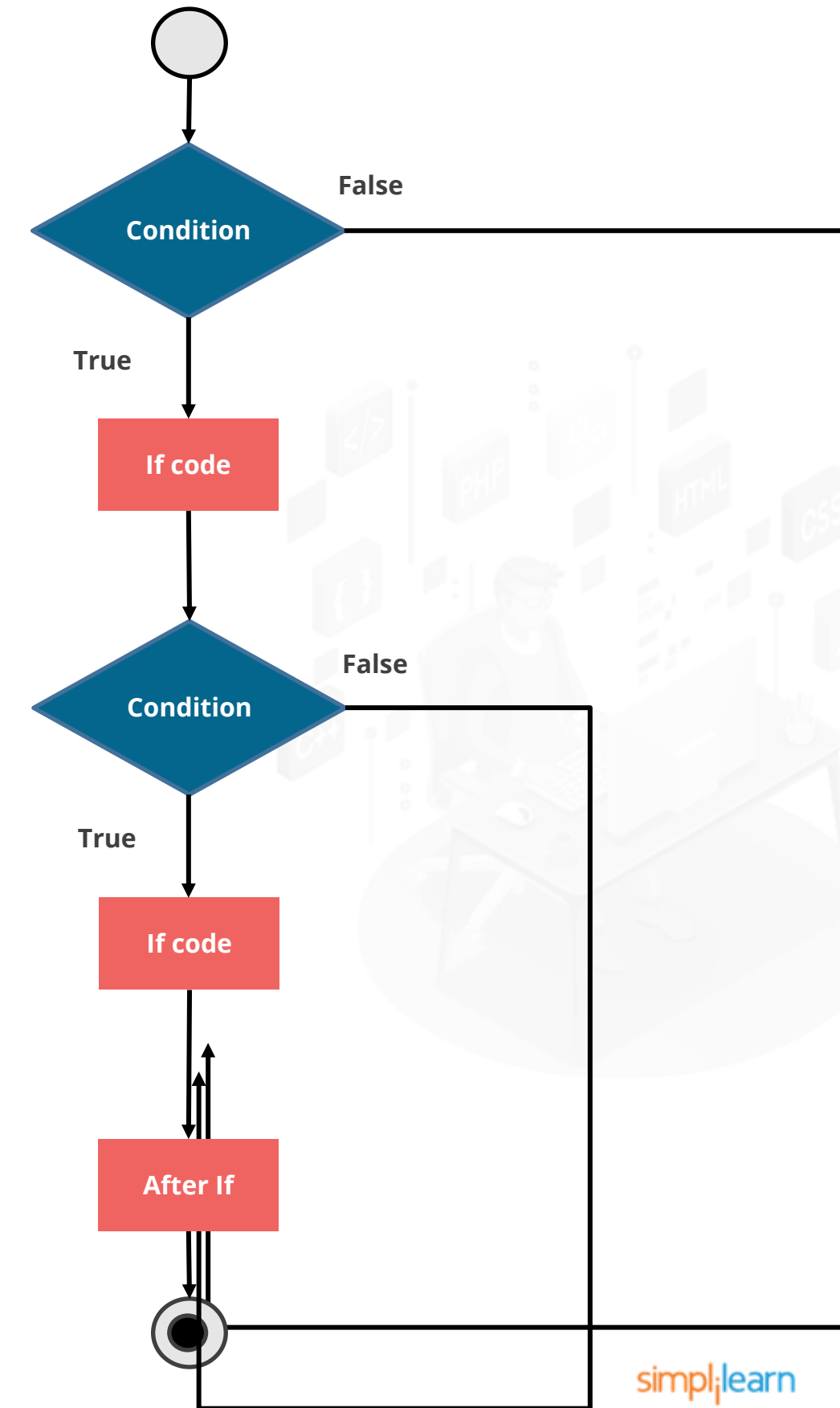
```
if(condition_1){  
  //statement to be executed if condition_1 is  
  true  
}else if(condition_2){  
  //statement to be executed if condition_2 is  
  true  
}  
  else if(condition_3){  
    //statement to be executed if condition_3 is  
    true  
  }  
else{  
  //statement to be executed if all the  
  conditions are false  
}
```



# Nested-If Statement

In this statement, the if statement focuses on another if statement.

```
if(condition1) {  
    //Nested if else inside the body of  
    "if"  
    if(condition2) {  
        //Code inside the body of nested  
        "if"  
    }  
    else {  
        //Code inside the body of nested  
        "else"  
    }  
}  
else {  
    //Code inside the body of "else."  
}
```



# Switch Statement

The switch statement executes one statement from numerous conditions.

Boolean

Number

Byte

Short

Int

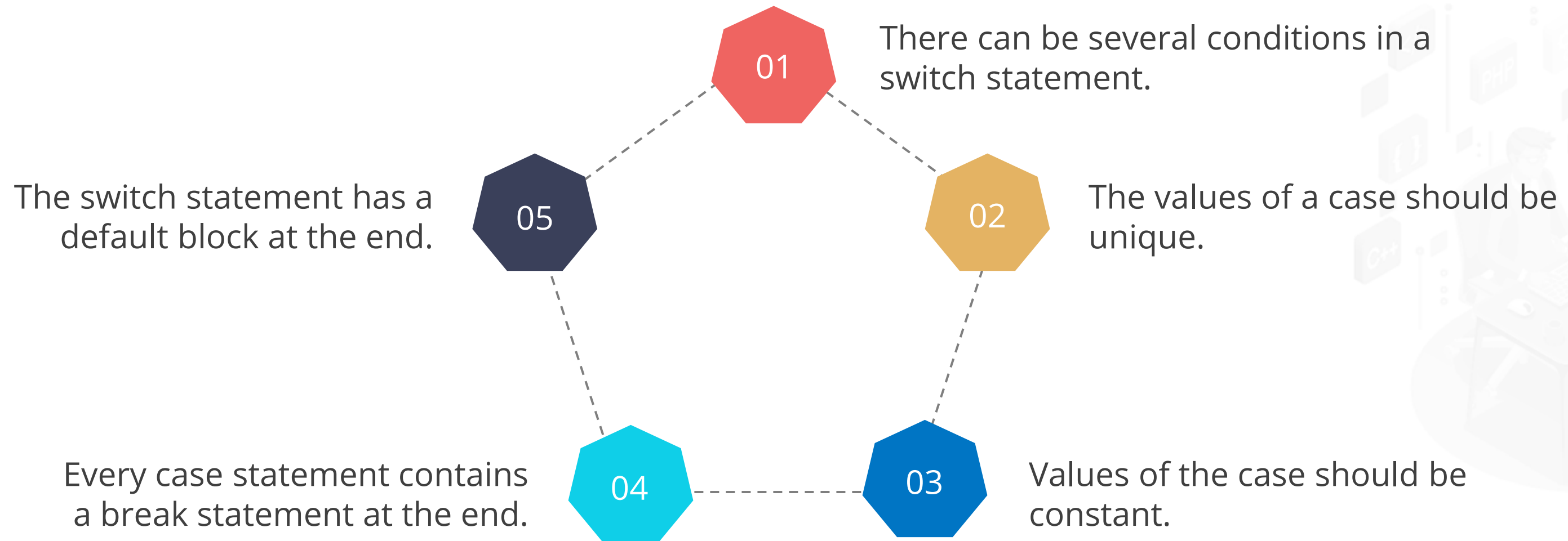
Long

Enum Type

String

# Switch Statement

Here are a few points to remember:



# Switch Statement

Syntax:

```
switch(expression) {  
    case value1:  
        //code to be executed if the cases are  
matched;  
break;  
    case value2:  
        //code to be executed if the cases are  
matched;  
break;  
    default:  
        // code will be executed if all the  
cases are not matched;  
}
```



# Switch Statement

The switch statement also contains:

## Case

It is followed by a constant and then a semicolon. The case cannot contain another expression or variable.

## Default

It should be written at the end. It executes when there will be no case matched.

## Break

The break must be written at the end to terminate the switch after executing a case block.

## Comments

They are intended to make the source code simpler for people to comprehend and are ignored by interpreters and compilers.



# Switch Statement

Syntax:

```
/**  
    statements  
*/
```



## Compiling and Executing a TypeScript Program

# Compiling and Executing a TypeScript Program

Steps to compile and execute a Typescript program are:

Start by saving the file with a TypeScript extension

Open the saved file with the .ts extension in the command prompt

Use this command to compile the program:  
"tsc file\_name.ts"

Use this command to run the program:  
"Node file\_name.js"

Use this command to compile multiple files at once: "tsc file\_1.ts file\_2.ts"



## Data Types

# Data Types

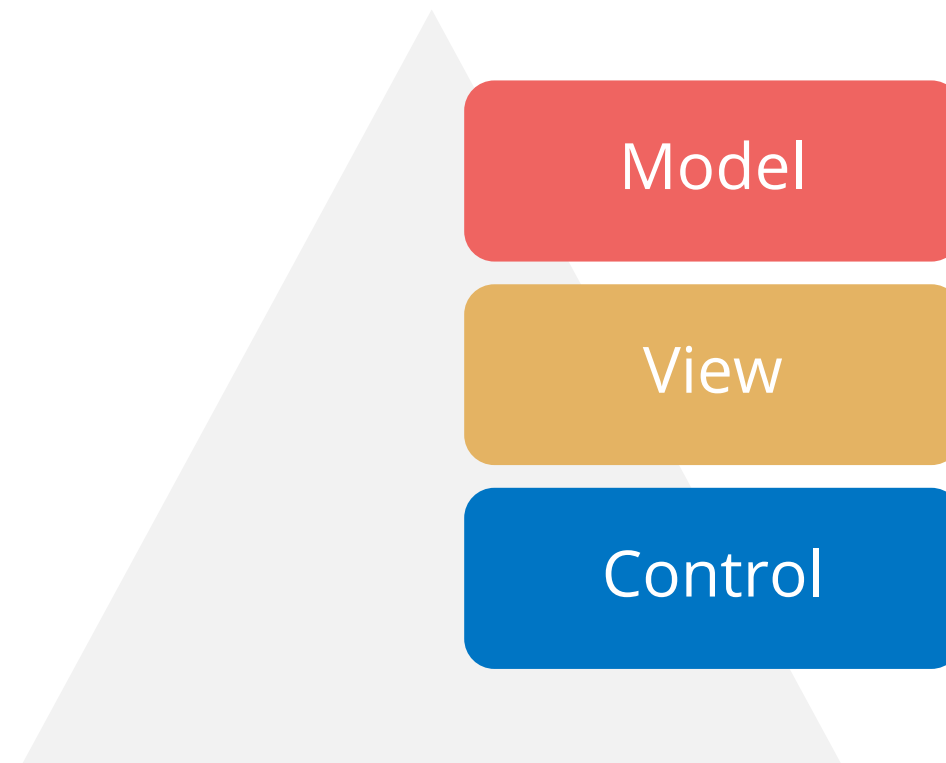
The list of data types in TypeScript are:

Data Types	Uses	Keyword
Any	It can store any type of data in any type of variable.	any
Number	It is used for integers, fractions, or any numbers.	number
String	It is used for sequences of Unicode or chars.	string
Boolean	It is used for logical values, i.e., true and false.	boolean
Void	It is used for function return types for representing non-returning functions.	void
Null	It is used for the absence of value of an object.	Null
Undefined	It is used for all uninitialized variables.	undefined

## Key Features of Angular Core

# MVC Architecture

MVC Architecture is used to develop web applications and consists of three parts:





# Key Features of Angular Core

1

## **Efficient two-way binding:**

The view layer represents the model and works in perfect synchronization.

2

## **Less code framework:**

It does not need different codes to make the connection between the MVC layers.



# Key Features of Angular Core

The basic CLI of Angular is:

`ng new`

This is used to make an Angular app. It is a primary initial for Angular development.

`ng generate`

This is used to generate a component, service, module, pipes, and routes.

`ng serve`

It allows testing the app on the local server.

# Key Features of Angular Core

Angular allows using the dynamic syntax instead of a string for lazy-loaded modules.

```
{ path: '/products', loadChildren: ()  
=>  
import('./products/product.module').t  
hen(s => p.ProductModule) }
```



## Key Takeaways

- Angular is a client-side TypeScript-based full-stack web framework and a front-end development system.
- Angular CLI is used to initialize, develop, and maintain Angular applications directly from a command shell.
- MVC Architecture is used to develop web applications and consists of the model, view, and controller.
- Angular allows using the dynamic syntax instead of a string for lazy-loaded modules.



# TECHNOLOGY

**Thank You**