Mobile Application Development with Flutter

Lecture 8: Functions in Dart

Functions in Dart

- **Functions** are the block of code that performs a specific task
- The function helps reusability of the code in the program

- Function Advantages:
  - Avoid Code Repetition
  - Easy to divide the complex program into smaller parts
  - Helps to write a clean code
  - Improve readability
  - Easier debugging

Syntax of Function

```
returnType functionName(parameters) {
  // body
}
```

```
void sayHello() {
  print("Hello");
}
```

```
int add(int a, int b) {
  return a + b;
}
```

Function Parameters

Required Parameters

```
void greet(String name) {
  print("Hello $name");
}
```

Optional Positional Parameters

```
void greet(String name, [String? title]) {
  print("Hello $title $name");
}
```

```
greet("Ali");
greet("Ali", "Mr.");
```

Function Parameters

Optional Named Parameters

```
void register({String? name, int? age}) {
  print("$name is $age years old");
}
```

```
register(name: "Ali", age: 20);
```

Required Named Parameters

```
void register({required String name, required int age}) {
  print("$name is $age years old");
}
```

Example 1: Function That Prints Name

```
// writing function outside main function.
void printName(){
  print("My name is Raj Sharma. I am from function.");
}
// this is our main function.
void main(){
  printName();
}
```

Example 2: Function To Find Sum of Two Numbers

```
void add(int num1, int num2){
   int sum = num1 + num2;
    print("The sum is $sum");
}

void main(){
   add(10, 20);
}
```

Example 3: Function To Find Sum of Two Numbers

```
// this function add two numbers
int add(int a, int b) {
  int sum = a + b;
  return sum;
}

void main() {
  int num1 = 10;
  int num2 = 20;

  int total = add(num1, num2);
  print("The sum is $total.");
}
```

Example 4: Default Value of Positional Parameter

```
void add(int num1, int num2, [int num3=5]){
  int sum;
  sum = num1 + num2 + num3;
  print("The sum is $sum");
}

void main(){
  add(10, 20);
  add(10, 20, 30);
}
```

Anonymous Function

- A function without a name
- Syntax

```
(parameterList){
// statements
}
```

- Anonymous functions are used in:
  - Button onPressed
  - List operations
  - Callbacks
  - Event handling

Example 1: Anonymous Function In Dart

```dart
void main() {
  const fruits = ["Apple", "Mango", "Banana", "Orange"];

  fruits.forEach((fruit) {
    print(fruit);
  });
}
```

Example 2: Anonymous Function In Dart

```dart
void main() {
// Anonymous function
  var cube = (int number) {
    return number * number * number;
  };

  print("The cube of 2 is ${cube(2)}");
  print("The cube of 3 is ${cube(3)}");
}
```

Arrow Functions

- Dart has a special syntax for the function body, which is only one line
- The arrow function is represented by => symbol
- It is a shorthand syntax for any function that has only one expression
- Syntax

```
returnType functionName(parameters...) => expression;
```

Example 1

- Short syntax for single-expression functions

```
int add(int a, int b) => a + b;
```

- Equivalent to:

```
int add(int a, int b) {
  return a + b;
}
```

- Arrow with forEach

```
numbers.forEach((num) => print(num));
```

Functions as First-Class Objects

- Dart treats functions as objects
- We can:

✓ Store in variable
✓ Pass as argument
✓ Return from another function

Example1: Pass a Function as a Parameter

```
void execute(Function action) {
  action();
}

execute(() {
  print("Running");
});
```

Example2: Pass a Function as a Parameter

```
void sayHello() {
  print("Hello");
}

void runFunction(Function f) {
  f();
}

runFunction(sayHello);
```

Function Return From Another Function

```
Function getGreeting() {
  return () {
    print("Hello!");
  };
}
```

```
var greet = getGreeting();
greet();
```