In Dart, variables are declared using the var, final, const, or explicit data types like int, double, String, etc. Here's a breakdown of how to declare variables in Dart:

**1. Using var**

The var keyword lets the compiler infer the type automatically.

dart

Copy code

var name = 'Alice'; // String inferred

var age = 30; // int inferred

var height = 5.9; // double inferred

Once inferred, the type can't be changed later.

**2. Using Explicit Data Types**

You can specify the exact type of a variable.

dart

Copy code

String name = 'Bob';

int age = 25;

double height = 6.2;

bool isActive = true;

**3. Using final**

final is used for variables whose values won't change after being set. The value is assigned at runtime.

dart

Copy code

final city = 'New York';

final int zipCode = 10001;

**4. Using const**

const is used for compile-time constants. The value must be known at compile time and cannot change.

dart

Copy code

const pi = 3.14;

const String country = 'USA';

**5. Nullable Variables**

By default, Dart variables are non-nullable. To allow null values, use the ? symbol:

dart

Copy code

String? nullableName = null;

int? nullableAge = null;

**6. Late Variables**

Use late for variables that are initialized later, but guaranteed to be set before use.

dart

Copy code

late String description;

description = 'This is a late-initialized variable.';

**Example Code**

dart

Copy code

void main() {

var name = 'Alice';

final age = 25;

const country = 'Canada';

String? nickname = null;

print('$name is $age years old from $country.');

print('Nickname: $nickname');

}

This prints:

csharp

Copy code

Alice is 25 years old from Canada.

Nickname: null

Let me know if you need more clarification or details! 😊

In Dart, functions are blocks of reusable code that perform a specific task. You can define functions with or without parameters and return values. Here's a breakdown of how to use functions in Dart:

**1. Basic Function Syntax**

dart

Copy code

void greet() {

print('Hello, World!');

}

void main() {

greet(); // Output: Hello, World!

}

* **void** means the function doesn't return any value.
* The function is named greet.

**2. Function with Parameters**

You can pass data to a function using parameters.

dart

Copy code

void greetUser(String name) {

print('Hello, $name!');

}

void main() {

greetUser('Alice'); // Output: Hello, Alice!

}

**3. Function with Return Values**

Use a return type (like int, String, etc.) to return a value from a function.

dart

Copy code

int add(int a, int b) {

return a + b;

}

void main() {

int result = add(3, 5);

print('Result: $result'); // Output: Result: 8

}

**4. Optional Parameters**

You can specify optional parameters by using square brackets []. These parameters can be omitted when calling the function.

dart

Copy code

void greet(String name, [String? message]) {

if (message != null) {

print('$message, $name!');

} else {

print('Hello, $name!');

}

}

void main() {

greet('Alice'); // Output: Hello, Alice!

greet('Bob', 'Good morning'); // Output: Good morning, Bob!

}

**5. Named Parameters**

Named parameters allow you to specify which argument corresponds to which parameter. They are enclosed in curly braces {}.

dart

Copy code

void introduce({required String name, int age = 18}) {

print('Name: $name, Age: $age');

}

void main() {

introduce(name: 'Alice'); // Output: Name: Alice, Age: 18

introduce(name: 'Bob', age: 25); // Output: Name: Bob, Age: 25

}

* **required** makes a parameter mandatory.
* **Default values** (e.g., age = 18) can be provided.

**6. Anonymous Functions (Lambdas)**

Anonymous functions (also called lambdas) are useful for short, throwaway functions.

dart

Copy code

void main() {

var numbers = [1, 2, 3, 4];

numbers.forEach((num) => print(num \* 2));

// Output: 2, 4, 6, 8

}

**7. Arrow Syntax**

For concise, single-expression functions, use the => shorthand.

dart

Copy code

int square(int x) => x \* x;

void main() {

print(square(4)); // Output: 16

}

**8. Higher-Order Functions**

Functions can take other functions as parameters or return functions.

dart

Copy code

void executeOperation(int a, int b, int Function(int, int) operation) {

print('Result: ${operation(a, b)}');

}

void main() {

executeOperation(4, 2, (a, b) => a + b); // Output: Result: 6

}