Menu
Log in
HTML CSS

C Function Parameters

❮ Previous
Next ❯

Parameters and Arguments

Information can be passed to functions as a parameter. Parameters act as variables inside the function.

Parameters are specified after the function name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma:

Syntax

```
returnType functionName(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

The following function that takes a string of characters with **name** as parameter. When the function is called, we pass along a name, which is used inside the function to print "Hello" and the name of each person.

Example

```
void myFunction(char name[]) {
  printf("Hello %s\n", name);
}

int main() {
  myFunction("Liam");
  myFunction("Jenny");
  myFunction("Anja");
  return 0;
}

// Hello Liam
// Hello Jenny
// Hello Anja
```

Try it Yourself »

When a **parameter** is passed to the function, it is called an **argument**. So, from the example above: `name` is a **parameter**, while `Liam` , `Jenny` and `Anja` are **arguments**.

Multiple Parameters

Inside the function, you can add as many parameters as you want:

Example

```
void myFunction(char name[], int age) {
  printf("Hello %s. You are %d years old.\n", name, age);
}

int main() {
  myFunction("Liam", 3);
  myFunction("Jenny", 14);
  myFunction("Anja", 30);
  return 0;
}

// Hello Liam. You are 3 years old.
// Hello Jenny. You are 14 years old.
// Hello Anja. You are 30 years old.
```

Try it Yourself »

Note that when you are working with multiple parameters, the function call must have the same number of arguments as there are parameters, and the arguments must be passed in the same order.

Return Values

The `void` keyword, used in the previous examples, indicates that the function should not return a value. If you want the function to return a value, you can use a data type (such as `int` or `float`, etc.) instead of `void`, and use the `return` keyword inside the function:

Example

```
int myFunction(int x) {
  return 5 + x;
}

int main() {
  printf("Result is: %d", myFunction(3));

  return 0;
}

// Outputs 8 (5 + 3)
```

Try it Yourself »

This example returns the sum of a function with **two parameters**:

Example

```
int myFunction(int x, int y) {
  return x + y;
}

int main() {
  printf("Result is: %d", myFunction(5, 3));

  return 0;
}

// Outputs 8 (5 + 3)
```

Try it Yourself »

You can also store the result in a variable:

Example

```
int myFunction(int x, int y) {
  return x + y;
}

int main() {
  int result = myFunction(5, 3);
  printf("Result is = %d", result);

  return 0;
}
// Outputs 8 (5 + 3)
```

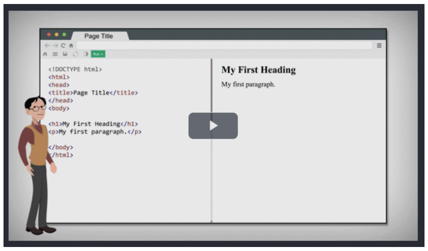
Try it Yourself »

❮ Previous

Next ❯

NEW

We just launched
W3Schools videos



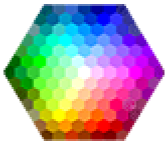
```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

My First Heading

My first paragraph.

Explore now

COLOR PICKER



Get certified
by completing

a course today!

Get started

CODE GAME

Play Game

Report Error

Spaces

Pro

Get Certified

Top Tutorials

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples

XML Examples
jQuery Examples

Get Certified

HTML Certificate
CSS Certificate
JavaScript Certificate
Front End Certificate
SQL Certificate
Python Certificate
PHP Certificate
jQuery Certificate
Java Certificate
C++ Certificate
C# Certificate
XML Certificate

FORUM | ABOUT

W3Schools is optimized for learning and training. Examples might be simplified to improve reading and learning. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using W3Schools, you agree to have read and accepted our terms of use, cookie and privacy policy.

W3Schools is Powered by W3.CSS.