**Functions with no type. The use of void.**

The syntax shown above for functions:

*type name ( argument1, argument2 ...) { statements }*

Requires the declaration to begin with a *type*. This is the type of the value returned by the function. But what if the function does not need to return a value? In this case, the type to be used is *void*, which is a special type to represent the absence of value. For example, a function that simply prints a message may not need to return any value:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 | // void function example  #include <iostream>  using namespace std;  void printmessage ()  {  cout << "I'm a function!";  }  int main ()  {  printmessage ();  } | I'm a function! | [Edit & Run](https://cplusplus.com/doc/tutorial/functions/) |

*void* can also be used in the function's parameter list to explicitly specify that the function takes no actual parameters when called. For example, *printmessage* could have been declared as:

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 | void printmessage (void)  {  cout << "I'm a function!";  } |  |

In C++, an empty parameter list can be used instead of *void* with same meaning, but the use of *void* in the argument list was popularized by the C language, where this is a requirement.

Something that in no case is optional are the parentheses that follow the function name, neither in its declaration nor when calling it. And even when the function takes no parameters, at least an empty pair of parentheses shall always be appended to the function name. See how *printmessage* was called in an earlier example:

|  |  |  |
| --- | --- | --- |
|  | printmessage (); |  |

|  |  |
| --- | --- |
|  | printmessage; |

The parentheses are what differentiate functions from other kinds of declarations or statements. The following would not call the function:

Only *void* does not have *return type*, for other functions at the end should return something.