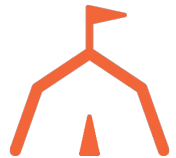
Methods



What Is A Method?

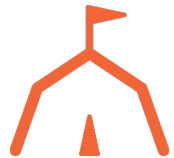
A method is a sequence of instructions that performs a specific task, and those instructions are packaged as a single unit.

Description

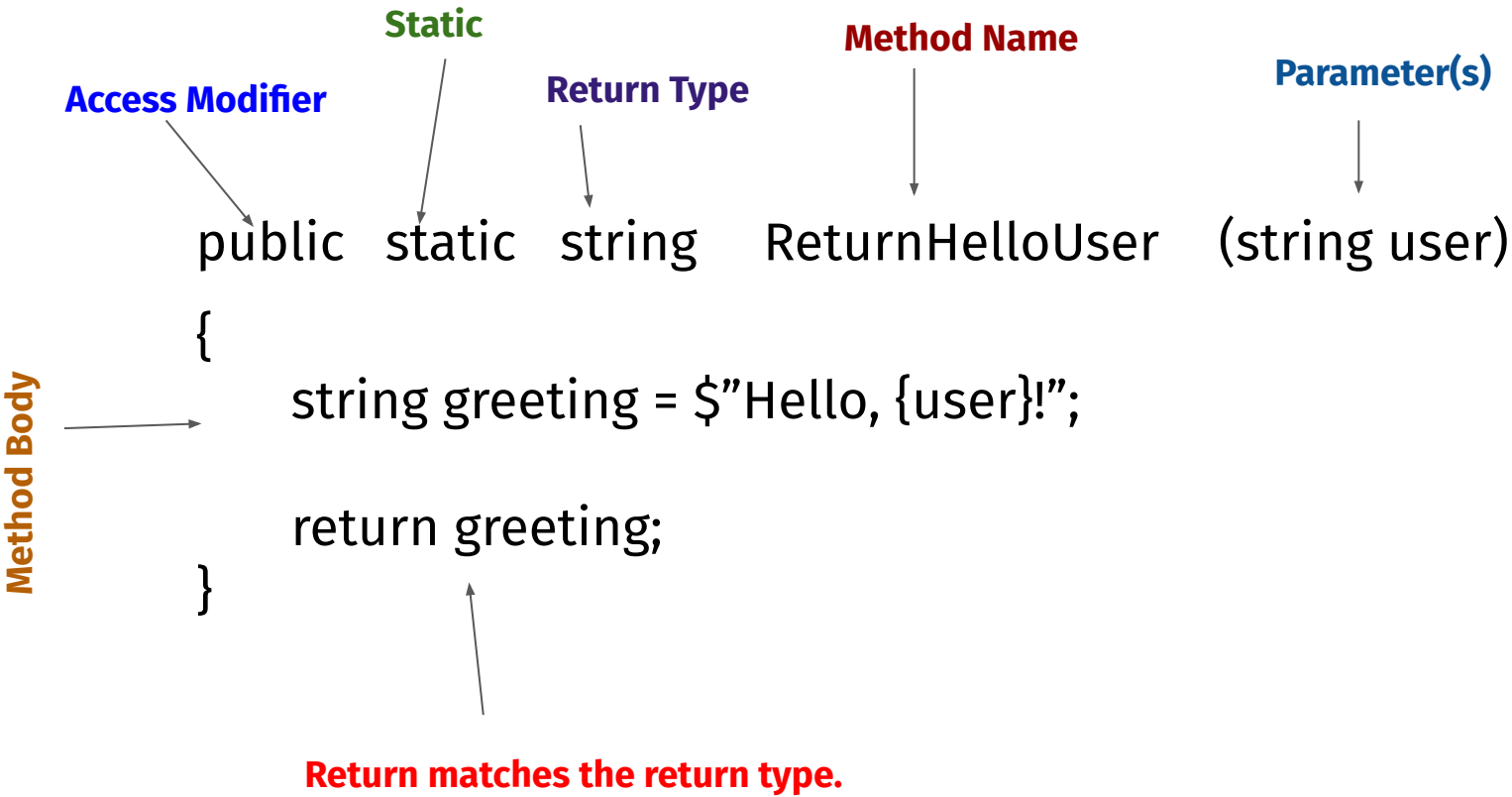
- To allow other classes to access a method, use the public keyword. To prevent other classes from accessing it, use the private keyword.
- Methods that don’t return any data should have the void keyword as the return type.

Description

Names of methods should typically start with verbs. Two very commonly used examples of this are methods which either set or return the values of instance variables, called setters and getters respectively.



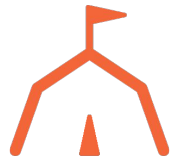
Syntax


Static
Method Name
Access Modifier
Return Type
Parameter(s)
public static string ReturnHelloUser (string user)
{
string greeting = $"Hello, {user}!";
return greeting;
}
Method Body
Return matches the return type.

Method Overloading

When you create two or more methods with the same name but with different:

- Parameter type
- Parameter order
- Parameter count
- Passing method (in, out, ref)



Calling Methods

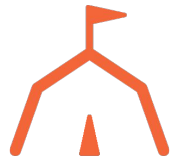
When calling a method with no arguments, we include an empty set of parentheses after the method name.

Calling Methods

If a method returns a value, we can code an assignment statement to assign the return value to a variable. The data type of that variable must be compatible with the data type of the return value.

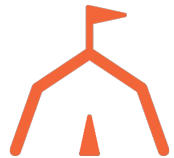
Syntax

```
objectName.methodName(argumentList)
```



Examples

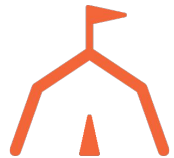
```
product.printToConsole();

product.setCode(productCode);

double price = product.getPrice();
```



Passing Parameters To Methods

There are three ways to send parameters to methods in C#

- Pass by value
- Pass by reference
- Pass by out



Recap

- What are methods and why are they important
- Syntax for writing methods
- How to do method overloading
- How to call methods
- Passing parameters to methods