What's the difference between a method and a function?

Asked 10 years, 10 months ago Active 6 months ago Viewed 639k times

1615

631

Can someone provide a simple explanation of **methods** vs. **functions** in OOP context?

function oop methods language-agnostic terminology

edited Jun 5 '18 at 12:25

Duc Filan

3,931 1 12 21

asked Sep 30 '08 at 23:45

willc2

18.2k 21 80 98

20 See also subroutine vs. function, function vs. procedure, coroutines. – gerrit Jan 29 '13 at 22:44

12 When the value of a property is a function, we call it a method – coder Aug 29 '14 at 19:21

2 What is the opposite of a method? That is my question. "Function" is the parent concept. Method is a type of function. What is the name for a type of function that is not a method, but that can be called directly by name? – Alexander Mills Sep 22 '16 at 17:56

It would be interesting to see another explanation outlining difference between methods and functions in **r**. It's interesting as method would usually *contain a function*. If method is an object-dependent function then function checking for object class `if(class(object)) { ... } else {stop()}` would be conceptually equivalent to method? – Konrad Sep 17 '17 at 19:18

31 Answers

1 2 next

1677

A **function** is a piece of code that is called by name. It can be passed data to operate on (i.e. the parameters) and can optionally return data (the return value). All data that is passed to a function is explicitly passed.

A **method** is a piece of code that is called by a name that is associated with an object. In most respects it is identical to a function except for two key differences:

1. A method is implicitly passed the object on which it was called.

2. A method is able to operate on data that is contained within the class (remembering that an object is an instance of a class - the class is the definition, the object is an instance of that data).

(this is a simplified explanation, ignoring issues of scope etc.)

edited Dec 19 '16 at 10:27

Nhan
2,989 6 24 30

answered Oct 1 '08 at 0:00

Andrew Edgecombe
30.8k 3 28 57

232 For 1), you should also add that methods in C++ are called *member* functions. Thus, the difference between functions and methods in this context is analogous to the difference between functions and member functions in C++. Furthermore, languages like Java only have methods. In this case, functions would be analogous to static methods and methods would have the same meaning. For 2), you should add that a method is able to operate on the *private* instance (member) data declared as part of the class. Any code can access public instance data. – Raffi Khatchadourian Jun 13 '12 at 21:59

43 a function is a mathematical construct. I would say all methods are functions but not all functions are methods – Tyler Gillies May 9 '13 at 20:22

28 Coming from a functional programming background, I feel there is a profound distinction between function and method. Mainly methods have side effects, and that functions should be pure thus giving a rather nice property of referential transparency – HHC Jul 2 '13 at 2:26

23 @TylerGillies and HHC, I agree it might be nice if "function" and "method" meant what you wanted them to, but your definitions do not reflect some very common uses of those terms. – Paul Draper Oct 15 '13 at 5:07

7 By this definition wouldn't a static method not actually be considered a method because it doesn't have anything to do with a particular instance? – Carcigenicate May 22 '15 at 16:44

920

A method is on an object.
A function is independent of an object.

For Java, there are only methods.
For C, there are only functions.

For C++ it would depend on whether or not you're in a class.

edited Feb 28 '13 at 15:54

J. Scott Elblein
1,098 3 21 45

answered Sep 30 '08 at 23:51

Aaron
13.2k 4 23 23

33 It is time to add C#. – Rosdi Kasim Mar 21 '14 at 8:15

5 @RosdiKasim For C#, there are only methods. (since even static methods need to be associated with an object) – Lynn Crumbling Apr 7 '14 at 13:14

34 How about static methods in a class? Those would be independent of an object (in java). Wouldn't that be a function then? – Squeazer Apr 15 '14 at 8:00

11 @Squeazer There is a question about that recently stackoverflow.com/questions/30402169/… The Java Language Specification just differs them as "class method"(static) and "instance method". So they're still all methods. – xji May 27 '15 at 8:31

11 Java has lambda expressions, which are functions that are not methods – Max Heiber Jan 7 '16 at 4:07

204

'method' is the object-oriented word for **'function'**. That's pretty much all there is to it (ie., no real difference).

Unfortunately, I think a lot of the answers here are perpetuating or advancing the idea that there's some complex, meaningful difference.

Really - there isn't all that much to it, just different words for the same thing.

[late addition]

In fact, as Brian Neal pointed out in a comment to this question, the C++ standard never uses the term 'method' when refering to member functions. Some people may take that as an indication that C++ isn't really an object-oriented language; however, I prefer to take it as an indication that a pretty smart group of people didn't think there was a particularly strong reason to use a different term.

edited May 23 '17 at 12:02
Community ♦
1 1

answered Sep 30 '08 at 23:48
Michael Burr
289k 41 449 684

14 A method is a special type of function with an implicit argument passed (an instance of the class that the method is defined on). This is important as a function, in strict terms, should not use or modify anything not in its argument list. – ty1824 Nov 4 '14 at 9:39

2 @ty1824 it's not always the case that methods are passed implicit arguments. In Python, the `self` argument is explicit. And many languages have static methods, which are not passed an instance. – Max Heiber Jan 7 '16 at 4:09

1 @mheiber those are some valid points. With regards to self, you are correct, it is explicitly *defined*. The key is that the call is implicit, based on the original object reference. There are languages that support overriding a `this` or `self`, but those constructs are then usually referred to as functions, rather than methods. – ty1824 Jan 7 '16 at 4:17

1 @mheiber As far as static methods are concerned - java implemented them as a workaround so functions could be implemented without requiring a context. I would go so far as to propose that 'static method' is a misleading term and should be replaced... in an ideal theoretical world :) – ty1824 Jan 7 '16 at 4:20

1 @ty1824 C++, Scala, Python, Ruby, JavaScript, and Scala all have static methods, so I don't think it's just a Java thing. I don't like them either: they amount to using objects (bundles of state) as if they were namespaces. – Max Heiber Jan 7 '16 at 4:49

89

In general: methods are functions that belong to a class, functions can be on any other scope of the code so you could state that all methods are functions, but not all functions are methods:

Take the following python example:

```
class Door:
  def open(self):
    print 'hello stranger'

def knock_door:
  a_door = Door()
  Door.open(a_door)

knock_door()
```

The example given shows you a class called "Door" which has a method or action called "open", it is called a method because it was declared inside a class. There is another portion of code with "def" just below which defines a function, it is a function because it is not declared inside a class, this function calls the method we defined inside our class as you can see and finally the function is being called by itself.

As you can see you can call a function anywhere but if you want to call a method either you have to pass a new object of the same type as the class the method is declared (Class.method(object)) or you have to invoke the method inside the object (object.Method()), at least in python.

Think of methods as things only one entity can do, so if you have a Dog class it would make sense to have a bark function only inside that class and that would be a method, if you have also a Person class it could make sense to write a function "feed" for that doesn't belong to any class since both humans and dogs can be fed and you could call that a function since it does not belong to any class in particular.

edited Nov 17 '14 at 1:41

answered Oct 1 '08 at 0:33

Gustavo Rubio
7,085 4 31 54

1 This answer---mostly its first sentence---is by far the most concise and overall best answer to the question. – yourcomputergenius Aug 31 '18 at 22:19

42

Simple way to remember:

- **F**unction → **F**ree (Free means not belong to an object or class)
- **M**ethod → **M**ember (A member of an object or class)

answered Oct 8 '17 at 0:07

Duc Filan
3,931 1 12 21

40

A very general definition of the main difference between a **Function** and a **Method**:

> Functions are defined outside of classes, while Methods are defined inside of and part of classes.

edited Sep 1 '16 at 6:54

Andrew Tobilko
31.2k 10 50 98

answered Oct 28 '14 at 23:08

Jorgesys
98.9k 16 259 221

27

If you feel like reading here is "My introduction to OO methods"

The idea behind Object Oriented paradigm is to "threat" the software is composed of .. well "objects". Objects in real world have properties, for instance if you have an Employee, the employee has a name, an employee id, a position, he belongs to a department etc. etc.

The object also know how to deal with its attributes and perform some operations on them. Let say if we want to know what an employee is doing right now we would ask him.

```
employe whatAreYouDoing.
```

That "whatAreYouDoing" is a "message" sent to the object. The object knows how to answer to that questions, it is said it has a "method" to resolve the question.

So, the way objects have to expose its behavior are called methods. Methods thus are the artifact object have to "do" something.

Other possible methods are

```
employee whatIsYourName
employee whatIsYourDepartmentsName
```

etc.

Functions in the other hand are ways a programming language has to compute some data, for instance you might have the function addValues(8 , 8) that returns 16

```
// pseudo-code
function addValues( int x, int y )  return x + y
// call it
result = addValues( 8,8 )
print result // output is 16...
```

Since first popular programming languages (such as fortran, c, pascal) didn't cover the OO paradigm, they only call to these artifacts "functions".

for instance the previous function in C would be:

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

It is not "natural" to say an object has a "function" to perform some action, because functions are more related to mathematical stuff while an Employee has little mathematic on it, but you can have methods that do exactly the same as functions, for instance in Java this would be the equivalent addValues function.

```
public static int addValues( int x, int y ) {
    return x + y;
}
```

Looks familiar? That´s because Java have its roots on C++ and C++ on C.

At the end is just a concept, in implementation they might look the same, but in the OO documentation these are called method.

Here´s an example of the previously Employee object in Java.

```
public class Employee {

    Department department;
    String name;

    public String whatsYourName(){
        return this.name;
    }
    public String whatsYourDeparmentsName(){
         return this.department.name();
    }
    public String whatAreYouDoing(){
        return "nothing";
    }
    // Ignore the following, only set here for completness
    public Employee( String name ) {
        this.name = name;
    }

}

// Usage sample.
Employee employee = new Employee( "John" ); // Creates an employee called John

// If I want to display what is this employee doing I could use its methods.
// to know it.
String name = employee.whatIsYourName():
String doingWhat = employee.whatAreYouDoint();

// Print the info to the console.

 System.out.printf("Employee %s is doing: %s", name, doingWhat );

Output:
Employee John is doing nothing.
```

The difference then, is on the "domain" where it is applied.

AppleScript have the idea of "natural language" matphor , that at some point OO had. For instance Smalltalk. I hope it may be reasonable easier for you to understand methods in objects after reading this.

NOTE: The code is not to be compiled, just to serve as an example. Feel free to modify the post and add Python example.

edited Oct 1 '08 at 0:35

community wiki
3 revs
OscarRyz

2 Bravo! Brilliant exposition on how methods are commands to objects. This is key to understanding OO and the lack of the rhetoric you display in this answer is why OO is so abused in the industry today. Excellent post. – moffdub Oct 1 '08 at 1:17

17

In OO world, the two are commonly used to mean the same thing.

From a pure Math and CS perspective, a function will always return the same result when called with the same arguments (f(x,y) = (x + y)). A method on the other hand, is typically associated with an instance of a class. Again though, most modern OO languages no longer use the term "function" for the most part. Many static methods can be quite like functions, as they typically have no state (not always true).

answered Sep 30 '08 at 23:56

TheSoftwareJedi
26.2k 19 95 143

15

Let's say a function is a block of code (usually with its own scope, and sometimes with its own closure) that may receive some arguments and may also return a result.

A method is a function that is owned by an object (in some object oriented systems, it is more correct to say it is owned by a class). Being "owned" by a object/class means that you refer to the method through the object/class; for example, in Java if you want to invoke a method "open()" owned by an object "door" you need to write "door.open()".

Usually methods also gain some extra attributes describing their behaviour within the object/class, for example: visibility (related to the object oriented concept of encapsulation) which defines from which objects (or classes) the method can be invoked.

In many object oriented languages, all "functions" belong to some object (or class) and so in these languages there are no functions that are not methods.

edited Oct 1 '08 at 0:05

answered Sep 30 '08 at 23:55

Mike Tunnicliffe
9,325 3 24 46

1 I belabored that methods could be owned by an object or class since Javascript (which you mentioned in your question) is one of the languages that bucks the general trend of object oriented languages by having methods owned by objects and not classes (although similar structures to classes exist). – Mike Tunnicliffe Oct 1 '08 at 0:10

13

Methods are functions of classes. In normal jargon, people interchange method and function all over. Basically you can think of them as the same thing (not sure if global functions are called methods).

http://en.wikipedia.org/wiki/Method_(computer_science)

answered Sep 30 '08 at 23:52

Statement
1,720 3 28 43

12

A function is a mathematical concept. For example:

```
f(x,y) = sin(x) + cos(y)
```

says that function f() will return the sin of the first parameter added to the cosine of the second parameter. It's just math. As it happens sin() and cos() are also functions. A function has another property: all calls to a function with the same parameters, should return the same result.

A method, on the other hand, is a function that is related to an object in an object-oriented language. It has one implicit parameter: the object being acted upon (and it's state).

So, if you have an object Z with a method g(x), you might see the following:

```
Z.g(x) = sin(x) + cos(Z.y)
```

In this case, the parameter x is passed in, the same as in the function example earlier. However, the parameter to cos() is a value that lives inside the object Z. Z and the data that lives inside it (Z.y) are implicit parameters to Z's g() method.

answered Oct 1 '08 at 0:54

Bradley Mazurek
476 7 9

Function or a method is a named callable piece of code which performs some operations and optionally returns a value.

12

In **C** language the term function is used. **Java** & **C#** people would say it a method (and a function in this case is defined within a class/object).

A **C++** programmer might call it a function or sometimes method (depending on if they are writing procedural style c++ code or are doing object oriented way of C++, also a C/C++ only programmer would likely call it a function because term 'method' is less often used in C/C++ literature).

You use a function by just calling it's name like,

```
result = mySum(num1, num2);
```

You would call a method by referencing its object first like,

```
result = MyCalc.mySum(num1,num2);
```

edited Mar 26 '15 at 23:10

Tshepang
6,389 18 74 115

answered Feb 6 '14 at 5:03

Abdullah Leghari
1,132 3 19 33

12

Historically, there may have been a subtle difference with a "method" being something which does not return a value, and a "function" one which does.Each language has its own **lexicon of terms** with special meaning.

In **"C"**, the word **"function"** means **a program routine**.

In **Java**, the term **"function"** does not have any special meaning. Whereas **"method"** means one of the routines that forms the implementation of a class.

In **C#** that would translate as:

```
public void DoSomething() {} // method
public int DoSomethingAndReturnMeANumber(){} // function
```

But really, I re-iterate that there is really no difference in the 2 concepts. If you use the term "function" in informal discussions about Java, people will assume you meant "method" and carry on. Don't use it in proper documents or presentations about Java, or you will look silly.

edited Feb 4 '16 at 10:33

answered Jan 12 '16 at 2:13

Jaimin Patel
3,375 2 25 27

Actually, in your example both are `method` ... or I would call It `procedure` and `function` . – Yousha Aleayoub Mar 11 '17 at 9:16

8

Methods on a class act on the instance of the class, called the object.

```
class Example
{
   public int data = 0; // Each instance of Example holds its internal data. This is a
"field", or "member variable".

   public void UpdateData() // .. and manipulates it (This is a method by the way)
   {
      data = data + 1;
   }

   public void PrintData() // This is also a method
   {
      Console.WriteLine(data);
   }
}

class Program
{
   public static void Main()
   {
       Example exampleObject1 = new Example();
       Example exampleObject2 = new Example();

       exampleObject1.UpdateData();
       exampleObject1.UpdateData();

       exampleObject2.UpdateData();

       exampleObject1.PrintData(); // Prints "2"
       exampleObject2.PrintData(); // Prints "1"
   }
}
```

answered Sep 30 '08 at 23:57


Statement
1,720 3 28 43

8

In OO languages such as Object Pascal or C++, a "method" is a function associated with an object. So, for example, a "Dog" object might have a "bark" function and this would be considered a "Method". In contrast, the "StrLen" function stands alone (it provides the length of a string provided as an argument). It is thus *just* a "function." Javascript is technically Object Oriented as well but faces many limitations compared to a full-blown language like C++, C# or Pascal. Nonetheless, the distinction should still hold.

A couple of additional facts: C# is fully object oriented so you cannot create standalone "functions." In C# every function is bound to an object and is thus, technically, a "method." The kicker is that few people in C# refer to them as "methods" - they just use the term "functions" because there isn't any real distinction to be made.

Finally - just so any Pascal gurus don't jump on me here - Pascal also differentiates between "functions" (which return a value) and "procedures" which do not. C# does not make this distinction explicitly although you can, of course, choose to return a value or not.

answered Sep 30 '08 at 23:59


Mark Brittingham
25.1k 11 73 104

8

Since you mentioned Python, the following might be a useful illustration of the relationship between methods and objects in most modern object-oriented languages. In a nutshell what they call a "method" is just a function that gets passed an extra argument (as other answers have pointed out), but Python makes that more explicit than most languages.

```
# perfectly normal function
def hello(greetee):
  print "Hello", greetee

# generalise a bit (still a function though)
def greet(greeting, greetee):
  print greeting, greetee

# hide the greeting behind a layer of abstraction (still a function!)
def greet_with_greeter(greeter, greetee):
  print greeter.greeting, greetee

# very simple class we can pass to greet_with_greeter
class Greeter(object):
  def __init__(self, greeting):
    self.greeting = greeting
```

```
  # while we're at it, here's a method that uses self.greeting...
  def greet(self, greetee):
    print self.greeting, greetee

# save an object of class Greeter for later
hello_greeter = Greeter("Hello")

# now all of the following print the same message
hello("World")
greet("Hello", "World")
greet_with_greeter(hello_greeter, "World")
hello_greeter.greet("World")
```

Now compare the function `greet_with_greeter` and the method `greet` : the only difference is the name of the first parameter (in the function I called it "greeter", in the method I called it "self"). So I can use the `greet` method in exactly the same way as I use the `greet_with_greeter` function (using the "dot" syntax to get at it, since I defined it inside a class):

```
Greeter.greet(hello_greeter, "World")
```

So I've effectively turned a method into a function. Can I turn a function into a method? Well, as Python lets you mess with classes after they're defined, let's try:

```
Greeter.greet2 = greet_with_greeter
hello_greeter.greet2("World")
```

Yes, the function `greet_with_greeter` is now also known as the method `greet2` . This shows the only real difference between a method and a function: when you call a method "on" an object by calling `object.method(args)` , the language magically turns it into `method(object, args)` .

(OO purists might argue a method is something different from a function, and if you get into advanced Python or Ruby - or Smalltalk! - you will start to see their point. Also some languages give methods special access to bits of an object. But the main conceptual difference is still the hidden extra parameter.)

answered Oct 1 '08 at 1:20

Sam Stokes

11.7k 7 32 32

Function is a set of logic that can be used to manipulate data.

8

While, Method is function that is used to manipulate the data of the object where it belongs. So technically, if you have a function that is not completely related to your class but was declared in the class, its not a method; It's called a bad design.

answered Dec 3 '08 at 9:47

gian ebao

7

for me: the function of a method and a function is the same if I agree that:

- a function may return a value
- may expect parameters

Just like any piece of code you may have objects you put in and you may have an object that comes as a result. During doing that they might change the state of an object but that would not change their basic functioning for me.

There might be a definition differencing in calling functions of objects or other codes. But isn't that something for a verbal differenciations and that's why people interchange them? The mentions example of computation I would be careful with. because I hire employes to do my calculations:

```
new Employer().calculateSum( 8, 8 );
```

By doing it that way I can rely on an employer being responsible for calculations. If he wants more money I free him and let the carbage collector's function of disposing unused employees do the rest and get a new employee.

Even arguing that a method is an objects function and a function is unconnected computation will not help me. The function descriptor itself and ideally the function's documentation will tell me what it needs and what it may return. The rest, like manipulating some object's state is not really transparent to me. I do expect both functions and methods to deliver and manipulate what they claim to without needing to know in detail how they do it. Even a pure computational function might change the console's state or append to a logfile.

answered Apr 13 '12 at 9:50

Dirk Schumacher

811 9 22

From my understanding a method is any operation which can be performed on a class. It is a general term used in programming.

4

In many languages methods are represented by functions and subroutines. The main distinction that most languages use for these is that functions may return a value back to the caller and a subroutine may not. However many modern languages only have functions, but these can optionally not return any value.

For example, lets say you want to describe a cat and you would like that to be able to yawn. You would create a Cat class, with a Yawn method, which would most likely be a function without any return value.

answered Sep 30 '08 at 23:53

marcj

869 1 7 10

4

To a first order approximation, a method (in C++ style OO) is another word for a member function, that is a function that is part of a class.

In languages like C/C++ you can have functions which are not members of a class; you don't call a function not associated with a class a method.

answered Oct 1 '08 at 0:00

Captain Segfault

1,492 7 11

3

IMHO people just wanted to invent new word for easier communication between programmers when they wanted to refer to functions inside objects.

If you are saying methods you mean functions inside the class. If you are saying functions you mean simply functions outside the class.

The truth is that both words are used to describe functions. Even if you used it wrongly nothing wrong happens. Both words describe well what you want to achieve in your code.

Function is a code that has to play a role (*a function*) of doing something. Method is *a method* to resolve the problem.

It does the same thing. It is the same thing. If you want to be super precise and go along with the convention you can call methods as the functions inside objects.

edited Jun 27 '15 at 16:15

answered Mar 31 '15 at 22:34

Morfidon


526 1 8 19

3

Function is the concept mainly belonging to **Procedure oriented programming** where a function is an an entity which can process data and returns you value

Method is the concept of **Object Oriented programming** where a method is a member of a class which mostly does processing on the class members.

answered Nov 10 '15 at 14:23


Akshay Khale
4,173 3 34 44

3

Let's not over complicate what should be a very simple answer. Methods and functions are the same thing. You call a function a function when it is outside of a class, and you call a function a method when it is written inside a class.

answered Dec 14 '17 at 14:03


SamyCode
346 1 4 20

2

I am not an expert, but this is what I know:

1. Function is C language term, it refers to a piece of code and the function name will be the identifier to use this function.
2. Method is the OO term, typically it has a this pointer in the function parameter. You can not invoke this piece of code like C, you need to use object to invoke it.
3. The invoke methods are also different. Here invoke meaning to find the address of this piece of code. C/C++, the linking time will use the function symbol to locate.
4. Objecive-C is different. Invoke meaning a C function to use data structure to find the address. It means everything is known at run time.

edited Mar 26 '15 at 23:10


Tshepang
6,389 18 74 115

answered Jul 30 '13 at 4:02


Marvin
21 1

2

I know many others have already answered, but I found following is a simple, yet effective single line answer. Though it doesn't look a lot better than others answers here, but if you read it carefully, it has everything you need to know about the method vs function.

A method is a function that has a defined receiver, in OOP terms, a method is a function on an instance of an object.

answered Apr 23 '17 at 23:11

f-society
1,759 17 15

2

Difference Between Methods and Functions

From reading this doc on Microsoft

> Members that contain executable code are collectively known as the function members of a class. The preceding section describes methods, which are the primary kind of function members. This section describes the other kinds of function members supported by C#: constructors, properties, indexers, events, operators, and finalizers.

So methods are the subset of the functions. Every method is a function but not every function is a method, for example, a `constructor` can't be said as a **method** but it is a function.

answered Dec 15 '17 at 18:42

Siraj Alam
2,129 1 19 38

> Properties, indexers and events are not functions. The getters, setters (for properties and indexers), adders and removers (for events) are functions. Function members are members that are associated with one or more functions, but not necessarily by being a function. – Jon Hanna Jan 12 '18 at 20:50

2

Function - A function in an independent piece of code which includes some logic and **must be called independently and are defined outside of class.**

Method - A method is an independent piece of code which is **called in reference to some object and are be defined inside the class.**

answered Jan 17 at 8:35

A H M

582 6 12

1

Here is some explanation for method vs. function using JavaScript examples:

`test(20, 50);` **is function define and use to run some steps or return something back that can be stored/used somewhere.**

> You can reuse code: Define the code once and use it many times.
>
> You can use the same code many times with different arguments, to produce different results.

```
var x = myFunction(4, 3);   // Function is called, return value will end up in x

function myFunction(a, b) {
    return a * b;           // Function returns the product of a and b
}
```

`var test = something.test();` **here test() can be a method of some object or custom defined a prototype for inbuilt objects, here is more explanation:**

> JavaScript methods are the actions that can be performed on objects.
>
> A JavaScript method is a property containing a function definition.

Built-in property/method for strings in javascript:

```
var message = "Hello world!";
var x = message.toUpperCase();
//Output: HELLO WORLD!
```

Custom example:

```
function person(firstName, lastName, age, eyeColor) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.age = age;
    this.eyeColor = eyeColor;
```

```
    this.changeName = function (name) {
        this.lastName = name;
    };
}

something.changeName("SomeName"); //This will change 'something' objject's name to
'SomeName'
```

You can define properties for String, Array, etc as well for example

```
String.prototype.distance = function (char) {
    var index = this.indexOf(char);

    if (index === -1) {
        console.log(char + " does not appear in " + this);
    } else {
        console.log(char + " is " + (this.length - index) + " characters from the end of
the string!");
    }
};

var something = "ThisIsSomeString"

// now use distance like this, run and check console log

something.distance("m");
```

Run code snippet Expand snippet

Some references: Javascript Object Method, Functions, More info on prototype

edited Sep 28 '17 at 14:32

answered Sep 28 '17 at 14:12

Lahar Shah
2,740 1 12 32

Please don't post identical answers to multiple questions. Post one good answer, then vote/flag to close the other questions as duplicates. If the question is not a duplicate, *tailor your answers to the question.* – Paul Roub Sep 28 '17 at 14:17

Thank you for the feedback, I didn't tailor other answer, I deleted that anwer. :-) – Lahar Shah Sep 28 '17 at 14:30

0

In C++, sometimes, method is used to reflect the notion of member function of a class. However, recently I found a statement in the book «The C++ Programming Language 4th Edition», on page 586 "Derived Classes"

> A virtual function is sometimes called a method.

This is a little bit confusing, but he said **sometimes**, so it roughly makes sense, C++ creator tends to see methods as functions can be invoked on objects and can behave polymorphic.

answered Sep 6 '17 at 4:02

Lance LI
36 1 5

0

A class is the collection of some data and function optionally with a constructor.

While you creating an instance (copy,replication) of that particular class the constructor initialize the class and return an object.

Now the class become object (without constructor) & Functions are known as method in the object context.

So basically

Class <==new==>Object

Function <==new==>Method

In java the it is generally told as that the constructor name same as class name but in real that constructor is like instance block and static block but with having a user define return type(i.e. Class type)

While the class can have an static block,instance block,constructor, function The object generally have only data & method.

edited Dec 25 '17 at 4:02

answered Dec 25 '17 at 3:55

Subrat Kumar Palhar
41 5

---

1 2 next

protected by Jorgesys Jun 15 '15 at 17:25

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?