Function naming conventions

Asked 9 years, 6 months ago Active 2 months ago Viewed 35k times

32

19

I am writing a library, so, I want its functions to be named as clearly and cleverly as possible. Currently, I use the following principles:

1. Self-explanatory names: a function getName() will tell the developer what it returns as well as setAddress(), isMale(), etc.
2. Short: a function name must be as short as possible so that it's simple to type as well as easy to remember. A function getNumberOfPagesInTheBook() is not good, something like getBookPageCount() is better.
3. Use of prefixes: I always use prefixes in the functions such as getName(), setName(), hasHair(), isBlond(), etc.

I'm interested in knowing if there's something I'm missing. Also, can you think of some other prefixes other than is, has, get and set?

function naming-conventions

edited Jan 2 '10 at 11:35

Adriaan Stander
136k 25 248 262

asked Jan 2 '10 at 11:25

Tower
46.3k 101 306 475

4 This is language-dependent to some degree, since different languages often have different naming/coding conventions. – skaffman Jan 2 '10 at 11:26

getters and setters can be an indication of poor design, if you have too many of them. – anon Jan 2 '10 at 11:28

@Neil Butterworth: Interesting! Curious to know why and what would be better approach then? (This is about getters and setters) – Madhu Jan 2 '10 at 11:31

2 If you need many get/set functions, it is often an indication that your class is simply a "record" with no real behaviour. – anon Jan 2 '10 at 11:34

Subtle but telling point. – Kzqai Jan 9 '10 at 22:49

8 Answers

One of the more universal, yet simple rules is: Function names should be verbs if the function changes the state of the program, and nouns if they're used to return a certain value.

56

answered Jan 2 '10 at 11:31

Carl Smotricz
55.9k 16 112 156

20 so that means getter functions are all named wrong?... ie getSomeValue(); – timh Aug 6 '12 at 22:24

1 @timh Accessors are a bit of a special case, they usually return properties of an object, something you'd just use property access syntax for in C# or AS3. – futlib May 23 '13 at 3:05

7 @timh yes, they are named wrong – Kamil Tomšík Apr 6 '14 at 15:44

1 @timh yes they are wrong thats why C# fixed theirs with get;set property – electricalbah Dec 3 '15 at 23:50

1 This isn't a good rule in languages with named closures and computed properties. A better rule for languages with those features is to always name functions as verbs or predicates. You have to think about what the function will look like when stored as a closure. A noun won't make sense; that gives the impression that it's the result of invocation. – Jessy Apr 30 '18 at 3:44

4

One more important thing to do when writing a library is to use the same word to describe the same action every time. don't write a function named **getName** in one class and another function named **retrieveNumber** in another class.

answered Jan 2 '10 at 11:31

Moshe Levi
3,073 15 23

I typically use `get` when it gets the data locally from class variable and or has a calculation (e.g `circle.getArea()`), and then I use `retrieve` when it must get this data from the outside, (e.g from database: `db.retrieveUserById(123)` – David Callanan Dec 18 '18 at 9:19

2

Have a look at

- Naming Conventions for .NET / C# Projects
- Naming Guidelines

answered Jan 2 '10 at 11:29

Adriaan Stander


136k 25 248 262

1

Other prefixes? Possibly "isa", though that is only applicable in some situations.

Some languages can communicate "get" and/or "set" with other constructs (specifically, in Common Lisp you can make (setf (get* ...) blah) do the same as what you would've wanted (set* ... blah) do).

answered Jan 2 '10 at 11:39


Vatine
16.9k 2 45 66

0

1. Naming conventions for Java
2. Naming conventions for .Net
3. Naming conventions for C++

answered Jan 2 '10 at 11:33


Suraj Chandran
19.3k 10 54 90

0

If there is a universal rule, I think it should be to be consistent.

There is also the "on" prefix, which is widely used when dealing with events (i.e. Java Android: `onViewCreated`). Some other prefixes or short and/or generic verbs (such as has, get and set) widely used are:

- can
- put / post

I prefer using nouns for simple getters when there is very little logic involved (i.e. properties) but I would use the "get" prefix for complex actions:

```
func center() {
   return (a + b) / 2
}
```

However, in some languages where using explicitly the "get" prefix is widely extended (i.e. Android - Java), the common practice is using some verb such as "compute" (i.e. `computeVerticalScrollOffset()`)

Furthermore, in some languages (e.g. swift) you can also use property setters so you don't really use the "set" prefix:

```
var x: X {
  get {
    return foo(x)
  }
  set {
    x = bar(newValue)
  }
}

// Set x
x = y
```

And finally, there are many widely used constructions such as `instanceof` , `indexOf` , ...

answered Jan 17 '16 at 12:31

FranMowinckel
2,693 19 22

0

Pro get/set

When a class has many methods, it is better to use verb prefixes, such as get/set, to distinguish methods from each other.

PHP example:

```
$foo->setText('Hello world!');
$foo->prependText('So. ');
$foo->appendText(' And welcome');
$x = $foo->getText();
```

By the way, in Hungarian notation prefixes go with a small letter and will not detract from keyword.

Counter get/set

When you need only two methods, it is easier to use the same noun in the context of using parameters.

jQuery example:

```
$('.foo').html();                //get
$('.foo').html('Hello world!'); //set
```

Examples

For functions and static methods with arrays as parameters I use the following rule:

If changes should occur only at run time:

```
setFoo($arr); // Replace/delete all properties, i.e. if some elements are not passed,
the corresponding properties will get empty values.
setFoo([]); // Delete all properties
setFoo(); // Set all properties by default
delFoo($arr); // Delete specified properties
addFoo($arr); // Add/replace specified properties
```

If changes will be made forever (in DB or files):

```
deleteFoo(...); // Delete specified properties
insertFoo(...); // Add specified properties
replaceFoo(...); // Add or replace specified properties
updateFoo(...); // Update specified properties
```

For both cases:

```
$arr = getFoo(); // Get all properties
$val = getFoo($level1, $level2, ...); // You can obtain the value of the given level,
placing the list of arguments
or
$val=getFoo()[$level1][$level2];
```

edited Mar 28 '16 at 5:27

answered Mar 27 '16 at 8:51

Vlad Alivanov

526 5 8

0

Here is a great resource advising the same as @Carl's answer: https://swift.org/documentation/api-design-guidelines/#strive-for-fluent-usage

> **Name functions and methods according to their side-effects**
>
> - Those without side-effects should read as noun phrases, e.g. `x.distance(to: y)`, `i.successor()`.
> - Those with side-effects should read as imperative verb phrases, e.g., `print(x)`, `x.sort()`, `x.append(y)`.

answered May 1 at 0:50

gamliela
1,511 1 20 34