http://www.yegor256.com/2014/12/15/how-much-your-objects-encapsulate.html

How Much Your Objects Encapsulate?

15 December 2014 modified on 15 December 2014 Yegor Bugayenko

Which line do you like more, the first or the second:

```
new HTTP("http://www.google.com").read();
new HTTP().read("http://www.google.com");
```

What is the difference? The first class `HTTP` encapsulates a URL, while the second one expects it as an argument of method `read()`. Technically, both objects do exactly the same thing: they read the content of the Google home page. Which one is the right design? Usually I hate to say this, but in this case I have to — it depends.

© The Truman Show (1998) by Peter Weir

As we discussed before, a good object is a representative of a real-life entity. Such an entity exists outside of the object's living environment. The object knows how to access it and how to communicate with it.

What is that real-life entity in the example above? Each class gives its own answer. And the answer is given by the list of arguments its constructors accept. The first class accepts a single URL as an argument of its constructor. This tells us that the object of this class, after being constructed, will represent a web page. The second class accepts no arguments, which tells us that the object of it will represent ... the Universe.

I think this principle is applicable to all classes in object-oriented programming — in order to understand what real-life entity an object represents, look at its constructor. All arguments passed into the constructor and encapsulated by the object identify a real-life entity accessed and managed by the object.

Of course, I'm talking about good objects, which are immutable and don't have setters and getters.

Pay attention that I'm talking about arguments encapsulated by the object. The following class doesn't represent the Universe, even though it does have a no-arguments constructor:

```
class Time {
  private final long msec;
  public Time() {
    this(System.currentTimeMillis());
  }
  public Time(long time) {
    this.msec = time;
  }
}
```

This class has two constructors. One of them is the main one, and one is supplementary. We're interested in the main one, which implements the

encapsulation of arguments.

Now, the question is which is better: to represent a web page or the Universe? It depends, but I think that in general, the smaller the real-life entity we represent, the more solid and cohesive design we give to the object.

On the other hand, sometimes we have to have an object that represents the Universe. For example, we may have this:

```
class HTTP {
  public String read(String url) {
    // read via HTTP and return
  }
  public boolean online() {
    // check whether we're online
  }
}
```

This is not an elegant design, but it demonstrates when it may be necessary to represent the entire Universe. An object of this `HTTP` class can read any web page from the entire web (it is almost as big as the Universe, isn't it?), and it can check whether the entire web is accessible by it. Obviously, in this case, we don't need it to encapsulate anything.

I believe that objects representing the Universe are not good objects, mostly because there is only one Universe; why do we need many representatives of it? :)