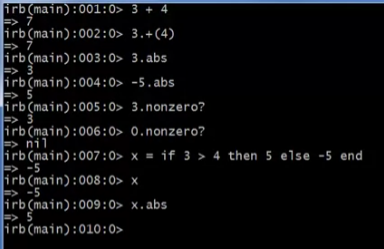
**Pure OOP**

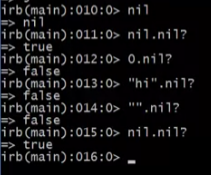
* Ruby is fully committed to OOP:
  + Every values is a reference to an object
* Simple, smaller semantics
* Can call methods on anything
  + May just get a dynamic “undefined method” error
* Almost everything is a method call

Example on numbers:

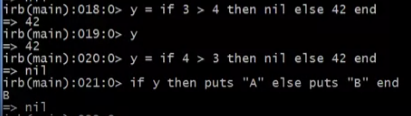


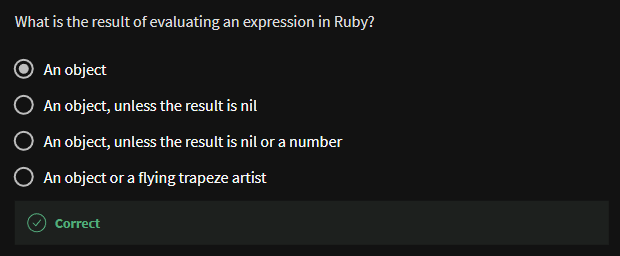
**nil Value**

* An object



* One of the 2 things that is **false** in ruby
  + Another one is the **false** object itself



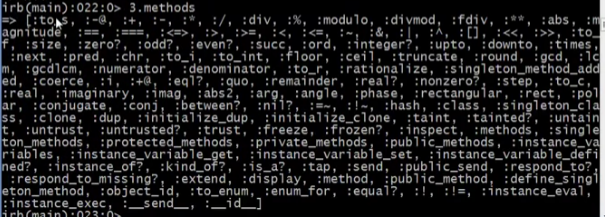


**All code is methods**

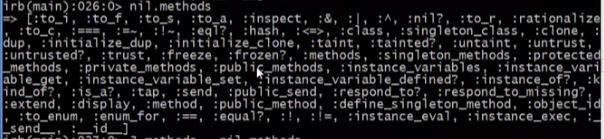
* All methods you define are part of a class
* Top-level methods (in file or REPL) just added to *Object* class
* Subclassing discussion coming later, but:
  + Since all classes you define are subclasses of Object, all inherit the top-level methods
  + So, you can call these methods anywhere in the program
  + Unless a class override (roughly-not-exactly, shadows) it by defining a method with the same name

**Reflection and exploratory programming**

* All object also has methods like:
  + Methods

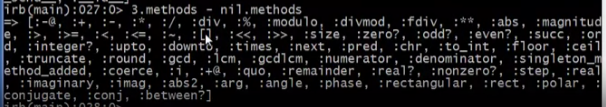






You can even subtract .methods of different objects

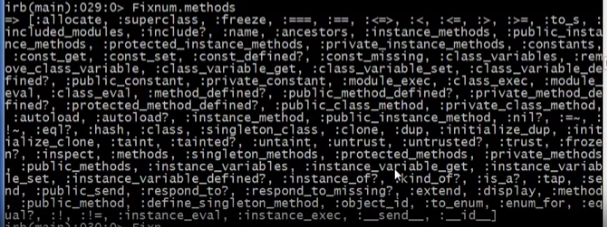
* + - This will give you the only the same methods on the 2 objects that you operate subtraction on



* + Class

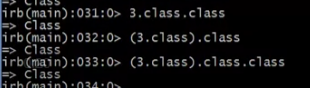


Fixnum also has a method & a class





Even classes are objects with a class



* Can use at run-time to query “what an object can do” and respond accordingly
  + Called *reflection*
* Also useful in the REPL to explore what methods are available
  + May be quicker than consulting full documentation
* Another example of “just objects and method call”