



How do you find the Sine function,  
if you don't know its name?

Ted Kaehler

**This material is based upon work supported in part by the National Science Foundation under Grant No. 0639876. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.**

VPRI Memo M-2009-012

## How do you find the Sine function, if you don't know its name?

*Also known as the "Semantic Types" problem.*

*Introducing the Method Tormentor.*

**Ted Kaehler, October, 2009 (revised from materials written in May 2009).**

Why is this important?

When an object is plunked down in a foreign world, it should find the resources it needs in order to run correctly. It may not know the names of the methods it should call. It needs to be able to find a method without knowing what it happens to be called in a foreign world.

A program that knows semantic types can write glue code. The programmer does not need to do it. It can hook things up.

When the slow-running specification disagrees with the fast optimization, this capability can help to discover what is wrong. It can match up methods with different names to see if they produce the same results.

This capability allows us to build a better Method Finder.

Solution 1 -- Use the Method Finder

Yes, the Squeak Method Finder can discover what message to send from one or three examples of its use. Let's find out how good this is. Suppose you pick a random method invocation on a real object in the system. Suppose you then remove all knowledge of the selector. Can that selector be found again? We need to be able to run repeatable experiments, invoke an unknown method, and an error should not injure the system. Doing this in an isolated World would be nice, but there is another way. Squeak includes many test methods, which are protected in case they fail, know how to set themselves up and know how to tear themselves down. These are in the SUnit classes.

My "Method Tormentor" program passes over every test method in the system. It makes a copy of the method and removes each selector in turn. It then tries to discover the selector with no prior knowledge.

A few selectors are too dangerous to use randomly, and a few tests can run amok and hurt the system. There is nothing wrong with factorial, except when you invoke it on the number of seconds since the year 1904.

I (painfully) identified 216 dangerous selectors, and 52 test methods that injure the system, take too long when modified, or don't work at all.

With these blocked out, I ran a modified MethodFinder against all selectors used in 72 test classes that contain 622 test methods. The Method Tormentor found 1216 selectors again after each been removed separately. This entire task took about 10 minutes.

When you have a concrete receiver and arguments, and a method that can fail without injuring the system, it is perfectly possible to discover the method that needs to be invoked. It should be noted that the test unit method was run as a whole. The replaced selector was buried inside it. This is a harder problem than just running a single invocation of a method.

These were "perfectly matched" invocations. The arguments were all correct, and the correct answer was known.

A slightly harder problem is what to do when a little glue code is needed. The MethodFinder already knows how to do this in some circumstances. It can convert a string to a number, and substitute true for #true, leave out extra arguments, and not be fooled by slightly wrong floating point values.

This new "Method Tormentor" capability will allow the successful docking of an object in a new environment.

#### Solution 2 -- Any Description is Fair Game

Here is a second way to discover the name of a function. The motto is "Any way that you can describe a function is legitimate." If we passed over all methods and annotated them, and then allowed search on the annotations, we could find the method. Here is a taxonomy of ways to describe a method. In this taxonomy, just check the items that apply to the method you need.

1. The method is a Request for Info. No side effects.
  - 1.A Pure request
    - 1.A.1 get value of an instance variable
    - 1.A.2 A computed function
      - 1.A.2.A arithmetic
      - 1.A.2.B trig functions
  - 1.B Create a new object for later use.
  - 1.C Show something on the screen
  - 1.D Create a copy
    - 1.D.1 complete copy
    - 1.D.2. partial copy
  - 1.E Give me a handle to an object I can see on the screen
  - 1.F Copy the receiver and Sort or Reorder
  - 1.G Search
    - 1.G.1 Search for one item
    - 1.G.2 Search for a combination of items
2. Modifies the receiver
3. Modifies an Arg (i.e. printOn: aStream)

4. Modifies the system
5. The Answer (output) Range is known
  - 5.A Class of the Range
  - 5.B It is a Boolean test function
6. It is One-to-One:
  - 6.A One-to-One between domain and range, the same as information-preserving and it is reversible.
  - 6.B Almost One-to-One. One-to-One over part of its domain.
7. Outputs a list. The range is a list of multiple answers (other criteria apply to each answer)
8. Receiver and Args: We know the Domain (receiver and args)
  - 8.A Receiver class
  - 8.B Args classes
  - 8.C Some Arg value (incomplete) (may only provide the class of arg)
  - 8.D The number of args: 0, 1, 2, 3
9. Supply an Example:
  - 9.A One example
  - 9.B Three examples (for if-then-else, etc)
  - 9.C A straight line scenario -- provides a long list of examples. Several scenarios can be merged to form a program.
  - 9.D Examples but of the wrong type. Need trivial transform  $3 \rightarrow '3'$ ,  $'abc' \rightarrow \#abc$ ,  $\#foo \rightarrow \#(foo)$ , etc.  $\#(true) \Rightarrow \{true\}$
10. Similar Method. This method does the same thing as XXX except the receiver or an argument is of class YYY instead of ZZZ.
11. Constraints. Write constraints of before and after the method is run.
12. Description
  - 12.A Describe the function in mathematical language.
  - 12.B Use a taxonomy of functions to say what it is. Is there already an accepted taxonomy of math functions? Is there standard naming?
  - 12.C Describe in Computer Science terminology. Is there an algorithm taxonomy? Just name the chapter and volume in Knuth's books?
  - 12.D. "Deep Semantic" description.