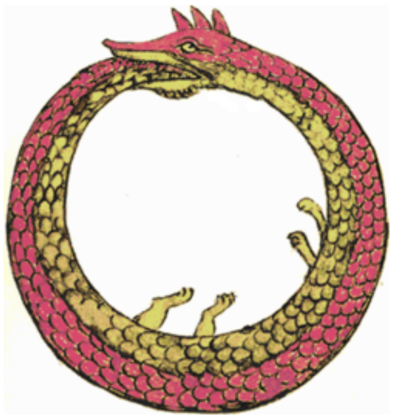
Protalk

PROTALK extends Smalltalk with prototype-based inheritance, ie delegation.

PROTALK turns Smalltalk object into real prototypes, without any change to the virtual machine. Protalk objects share the same object space with Smalltalk objects—actually, they are pure Smalltalk objects: an instance of a Protalk prototype is no different than an instance of a Smalltalk class. PROTALK uses the same inheritance and the same method lookup as Smalltalk.



Download: **http://www.squeaksource.com/Prototype.html**

Acknowledgments: Protalk was born on a boat trip at ESUG 2007 during a discussion with **Travis Griggs**. Protalk was originally implemented for Visualworks Smalltalk, and has recently been ported to Squeak with the kind help of Matthieu Suen.

How does PROTALK work?

Protalk objects are pure Smalltalk objects, with one small difference: each Protalk object is its own class. Protalk is pure Smalltalk, it relies on two features that had been hiding in plain sight since Smalltalk-80

- Every object can possible be used as a class
- The class of an object can change after instantiation

Change the Class of an Object

We all know how to get the class of an object

```
(3@4) class ⇒ Point
```

What about setting the class of an object, does that work?

```
(3@4) class: Assocation
```

Yes it works, but you must write it as follows

```
(3@4) primitiveChangeClassTo: Association basicNew ⇒ 3->4
```

This changes the class of the receiver into the class of the argument, given that (R = receiver, A = argument)

- the format of R matches the format of A
- neither A nor R are SmallIntegers
- either both A's and R's class are compact or none
- either both A and B are fixed sized or both are variable sized with matching sizes

A Class of its Own

Now, as we know how to can change the class of an object, we can create an object that is its own class.

```
oroboros := Class new.
oroboros superclass: Class.
oroboros methodDictionary: MethodDictionery new.
oroboros setFormat: Class format.
oroboros primitiveChangeClassTo: oroboros basicNew

oroboros class = oroboros ⇒ true
```

Everything is a Class

We all know, everything is an object and everything has a class. But did you know that everything is (possibly) a class? Any object can create instances of itself, given that

- it implements a method that calls primitive 70

- the 1st instance variable refers to its superclas
- the 2rd instance variable contains a method dictionary
- the 3nd instance variable contains the format, ie a magic number

If your object satisfies these constraints, the VM accepts it as a class and you can use your object to create instances of itself.

Let’s do that. First we have to create a new class `Thing` that inherits from object, ie instances of `Thing` are just normal objects. Then we will use an instance of `Thing` to create an instance of an instance of `Thing`.

Create a new class `Thing` with three instance variables

```
Object subclass: #Thing
    instanceVariableNames: 'delegate methods format'
    classVariableNames: ''
    poolDictionaries: ''
    category: 'Sandbox'
```

Implement an initializer and a method `new2` that calls primitive 70.

```
Thing >> initialize
    delegate := Object.
    methods := MethodDictionary new.
    format := superthing format.

Thing >> newToo
    <primitive: 70>;
```

Now, execute in a workspace:

```
object := Thing new.
abomination := object newToo.
```

```
abomination class ⇒ object abomination class class ⇒ Thing
```

Hence, instead of the usual two levels

```
        object --isa--> Thing
```

we have now three level of instantiation

```
        abomination --isa--> object --isa--> Thing
```

In the same way we can go to an unlimited number of instantiation levels. Combining this with the oroboros example above, we can create a prototype-based subspace within the class-based object space of a Smalltalk image.

To be continued...

Literals are not Constants

To be continued...

Any object is a Compiled Method

To be continued...

Other work

- Marvin, a prototype extension of Squeak
- Russell Allen's prototypes for Squeak
- Slate by Brian Rice ?
- Whatever by Michael Lucas-Smith

Last changed by admin on 21 April 2009