```
        poolDictionaries: ''
        category: 'Unknown'
```

We substitute the actual names for `NameOfSuperclass` and `NameOfSubclass`. In `instanceVariableNames`, we list the instance variables, in `classVariableNames` the class variables, and in `category`, we mark the category our class falls under (classes are grouped in categories in Smalltalk, similar to the namespaces or packages in other languages). The `poolDictionaries` slot lists the dictionaries that we are sharing with other classes; this is a mechanism for sharing variables in Smalltalk. When the details of the template are filled in, they are passed to the `subclass` selector of class `Class`:

```
    subclass: t instanceVariableNames: f classVariableNames: d
    poolDictionaries: s category: cat
        "This is the standard initialization message for creating a new class as a
        subclass of an existing class (the receiver)."
        ^(ClassBuilder new)
              superclass: self
              subclass: t
              instanceVariableNames: f
              classVariableNames: d
              poolDictionaries: s
              category: cat
```

The subclass selector creates an instance of class `ClassBuilder`, which creates new classes or modifies existing ones. We send the required information to the `ClassBuilder` instance so that the new class can be created according to what we have entered in the class template.

Doing everything on objects by sending messages gives us an economy of concepts that we have to grasp. It also allows us to limit the number of syntactic constructs in the language. Minimalism in programming languages goes a long way back. In the first paper on Lisp (McCarthy 1960), we find that Lisp comprised two classes of expressions: s-expressions (or syntactic expressions), which were expressions built from lists, and m-expressions (or meta-expressions), which were expressions using s-expressions as data. In the end, programmers opted for using s-expressions all the way, and hence Lisp became what we know today: a language with almost no syntax, as everything, program and data, is a list. Depending on where you stand on Lisp, this is proof that one single idea is enough for expressing the most complicated constructs (or that humans can be coerced into accepting anything).

Smalltalk does not limit itself to one syntactic element, but still, Smalltalk programs are composed of six building blocks:

1. Keywords, or pseudovariables, of which there are only six (`self`, `super`, `nil`, `true`, `false`, and `thisContext`)
2. Constants
3. Variable declarations
4. Assignments

Download at WoweBook.Com