Generator Methods

Generator methods are defined by the use of the **yield** keyword:

```
def fibonacci():
   a, b = 0, 1
   while true:
      yield b
      a, b = b, a+b
```

Given the definition above the following program would print the first five elements of the Fibonacci series:

```
for index as int, element in zip(range(5), fibonacci()):
   print "$(index+1): $element"
```

So although the generator definition itself is unbounded (a while true loop) only the necessary elements will be computed, five in this particular case as the zip builtin will stop asking for more when the range is exausted.

Generator methods are also a great way of encapsulating iteration logic:

```
def selectElements(element as XmlElement, tagName as string):
   for node as XmlNode in element.ChildNodes:
      if node isa XmlElement and tagName == node.Name:
         yield node
```