

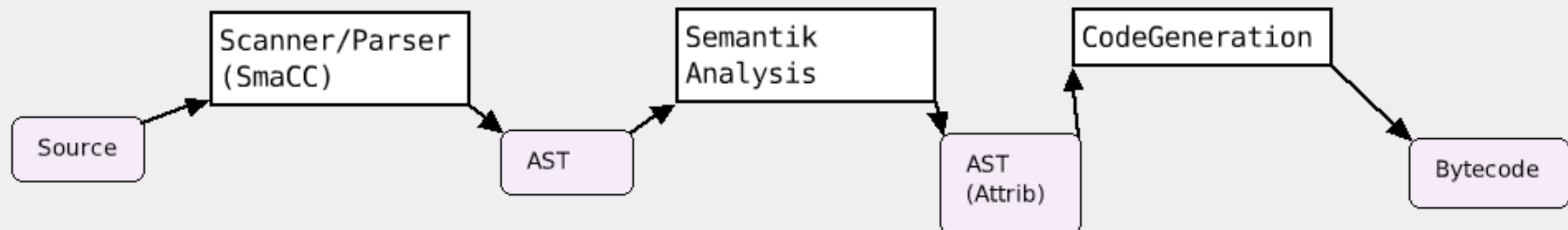
A  
Programming Language  
BABEL

# Goal

- \* Allow new users to use a well known Syntax for Scripting
- \* Build on top of the Squeak Object Model
- \* Complete Power of Squeak accessible

# Overview

## Simple "TextBook" Design:



Speed: Don't Care

Memory: Don't Care

"Do the simplest Thing that could possibly work"

# Tools

- \* SmaCC for the Scanner/Parser
- \* Anthony Hannan's IRBuilder for CodeGeneration

```
| irBuilder aCM |
```

```
irBuilder := InstructionBuilder new  
    rargs: #(self); "receiver and args"  
    pushLiteral: 1;  
    localReturnTop;  
    yourself.
```

```
aCM := irBuilder compiledMethodWith: #().  
aCM valueWithReceiver: nil arguments: #()
```

# Example

```
function test3plus4() {
  a = 3 + 4;
  this.assert(a == 7);
}
```

**a BMethodNode**

- root: a BMethodNode
- parent: nil
- parentheses: nil
- name: test3plus4
- body: a BScopeBlockNode
- parameters: nil
- class: nil
- codegen: nil
- trigger: nil
- scope: nil

AST

**AST**

**a BMethodNode**

- parent: a BNodeList
- parentheses: nil
- operator: a BOperatorNode
- leftExpression: a BIdentifierNode
  - parent: a BAssignmentExpressionNode
  - parentheses: nil
  - name: a
  - position: 26
  - binding: an InstVar

self binding class

JavaScriptClass run: #test3plus4

**a CompiledMethod (3188)**

```

self symbolic '21 <20> pushConstant: 3
22 <21> pushConstant: 4
23 <B0> send: +
24 <61> popIntoRcvr: 1
25 <70> self
26 <01> pushRcvr: 1
27 <22> pushConstant: 7
28 <C6> send: ==
29 <E3> send: assert:
30 <87> pop
31 <70> self
32 <7C> returnTop
.
```

JavaScriptClass run: #rest3plus4

Bytecode



# ***Current state***

- > Parser/AST/CodeGen for
  - \* JS
  - \* Python
  - \* LOGO ;-)
  
- > Parser for Ruby
  
- > Slowly a Framework is emerging  
(e.g. common parts of AST Classes)

# The Language: js

## JavaScript-like Syntax

```
function testForIn2() {ifFalse:
  var a,c,i;
  a = new OrderedCollection;
  a.add(1);
  a.add(2);
  c = 0;
  for (i in a) { c += i; }
  this.assert(c == 3);
}
```

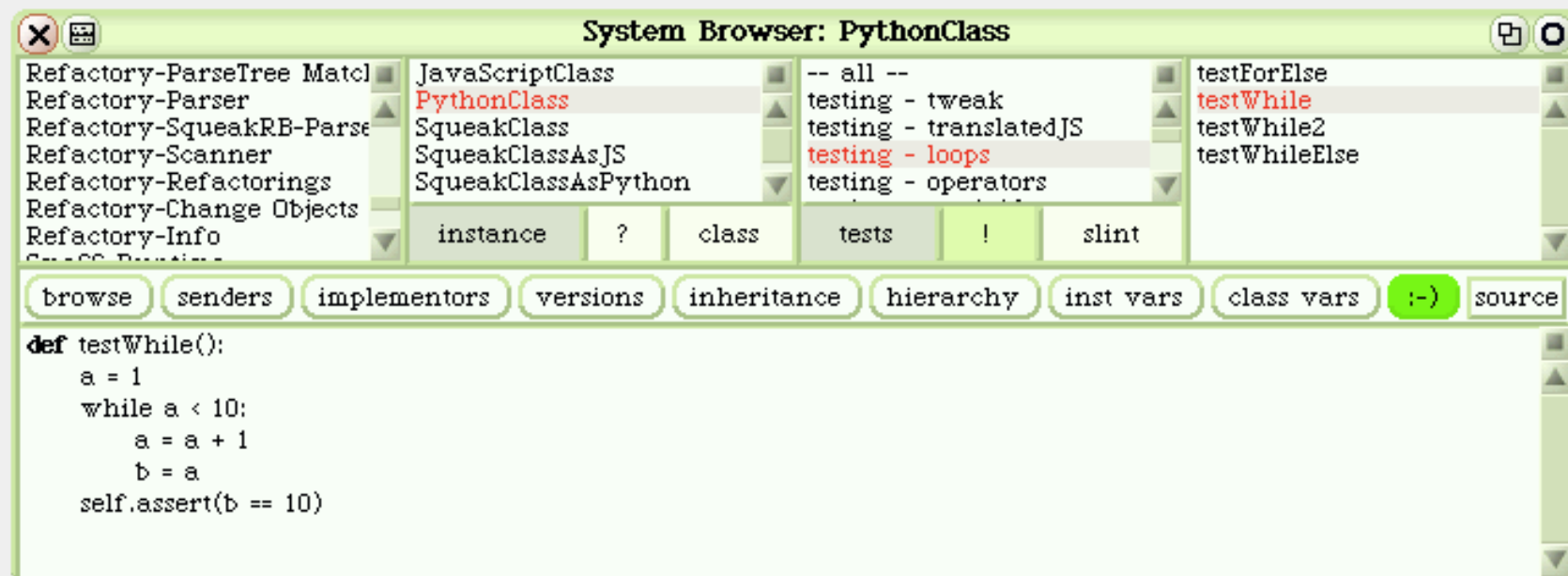
The screenshot shows the 'System Browser: JavaScriptClass' window. The left pane lists various classes, with 'JavaScriptClass' selected. The middle pane shows the class hierarchy, including 'PythonClass', 'SqueakClass', 'SqueakClassAsJS', and 'SqueakClassAsPython'. The right pane shows a list of methods, with 'testForIn2' highlighted. Below the panes are buttons for 'browse', 'senders', 'implementors', 'versions', 'inheritance', 'hierarchy', 'inst vars', 'class vars', and 'source'. The 'source' button is active, displaying the following code:

```
function testForIn2() {
  var a,c,i;
  a = new OrderedCollection;
  a.add(1);
  a.add(2);
  c = 0;
  for (i in a) { c += i; }
  this.assert(c == 3);
}
```

In the bottom right corner, there is a small 'talk' icon with a speech bubble.

# The Languages: Python

```
def testWhile():
    a = 1
    while a < 10:
        a = a + 1
        b = a
    self.assert(b == 10)
```



The screenshot shows the 'System Browser: PythonClass' window. The left pane lists various classes, with 'PythonClass' selected. The right pane shows a list of methods, including 'testWhile'. Below the panes are buttons for 'browse', 'senders', 'implementors', 'versions', 'inheritance', 'hierarchy', 'inst vars', 'class vars', and 'source'. The 'source' button is highlighted in green. The source code for 'testWhile()' is displayed in the bottom pane, matching the code shown above.

Next: LOGO





# The Languages: LOGO

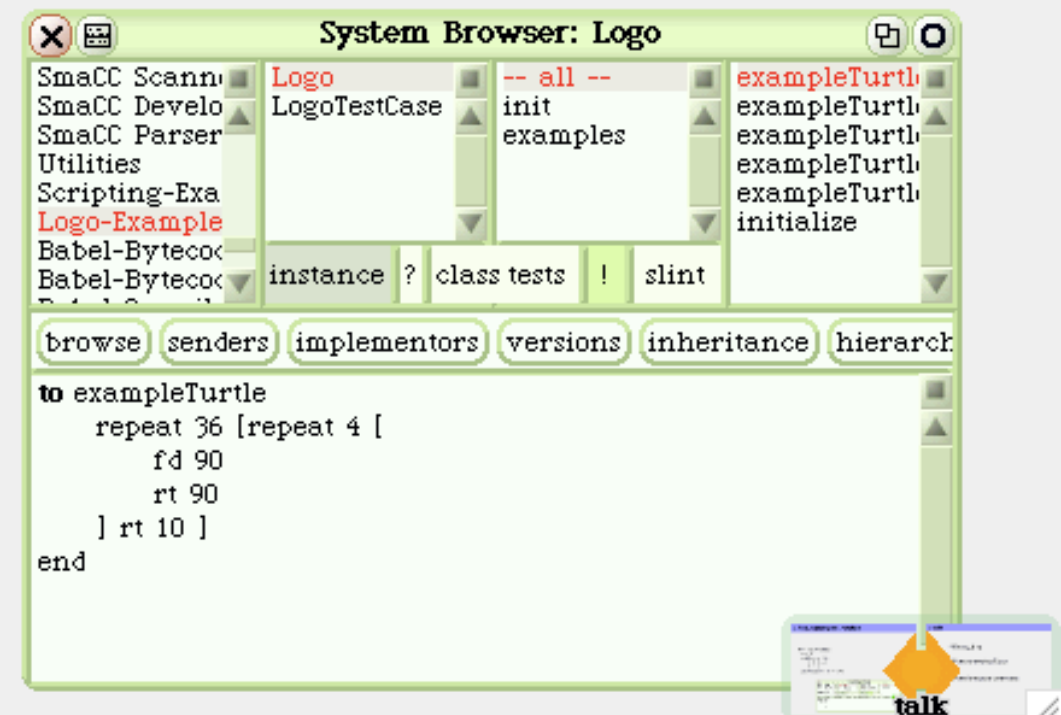
A real Compiler.

```

to exampleTurtle
  repeat 36 [repeat 4 [
    fd 90
    rt 90
  ] rt 10 ]
end

```

Display restoreAfter:  
[Logo new exampleTurtle]



# Todo

- \* Debugging
- \* More examples/Tests
- \* More language constructs

For more current work in this direction:

## **LanguageBoxes**

<http://scg.unibe.ch/research/languageboxes>

## **Helvetia.**

Context Specific Languages with Homogeneous Tool Integration

<http://scg.unibe.ch/research/helvetia>