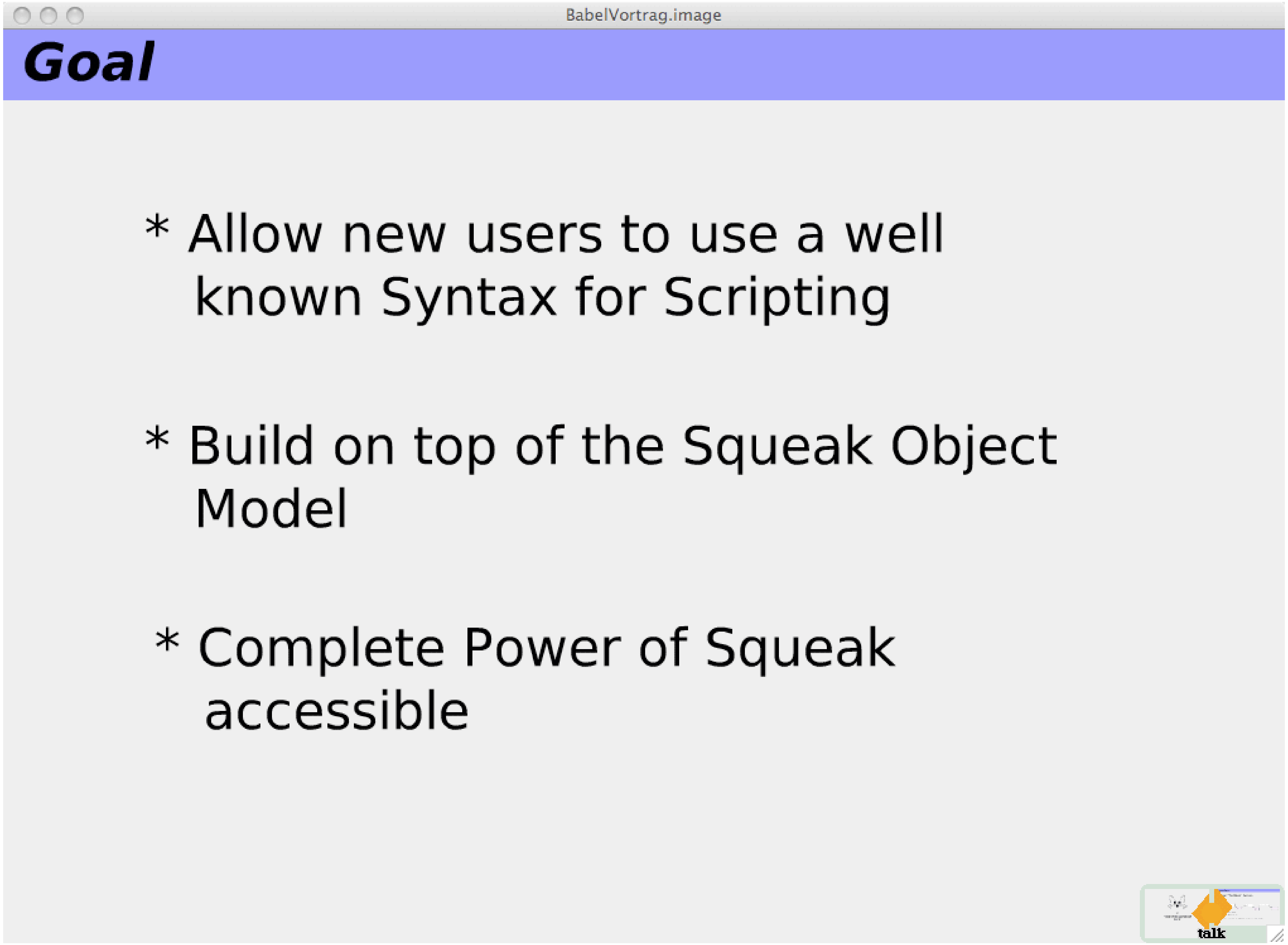
BabelVortrag.image

A ProgramingLanguage BABEL



Goal

* Allow new users to use a well known Syntax for Scripting

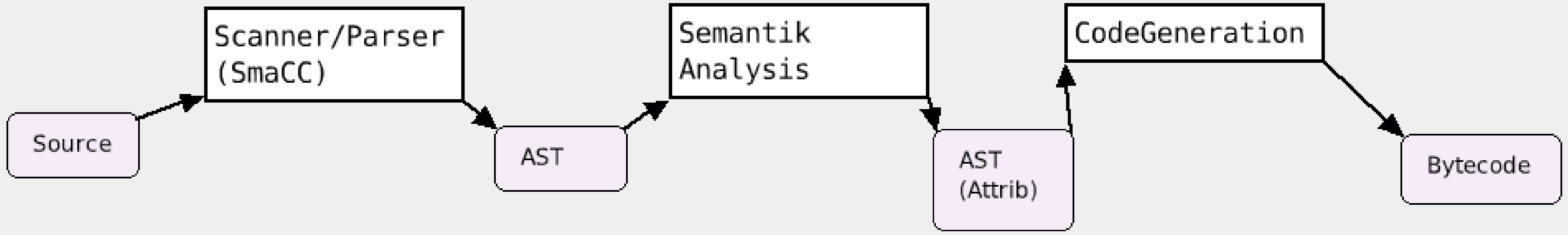
* Build on top of the Squeak Object Model

* Complete Power of Squeak accessible

BabelVortrag.image

Overview

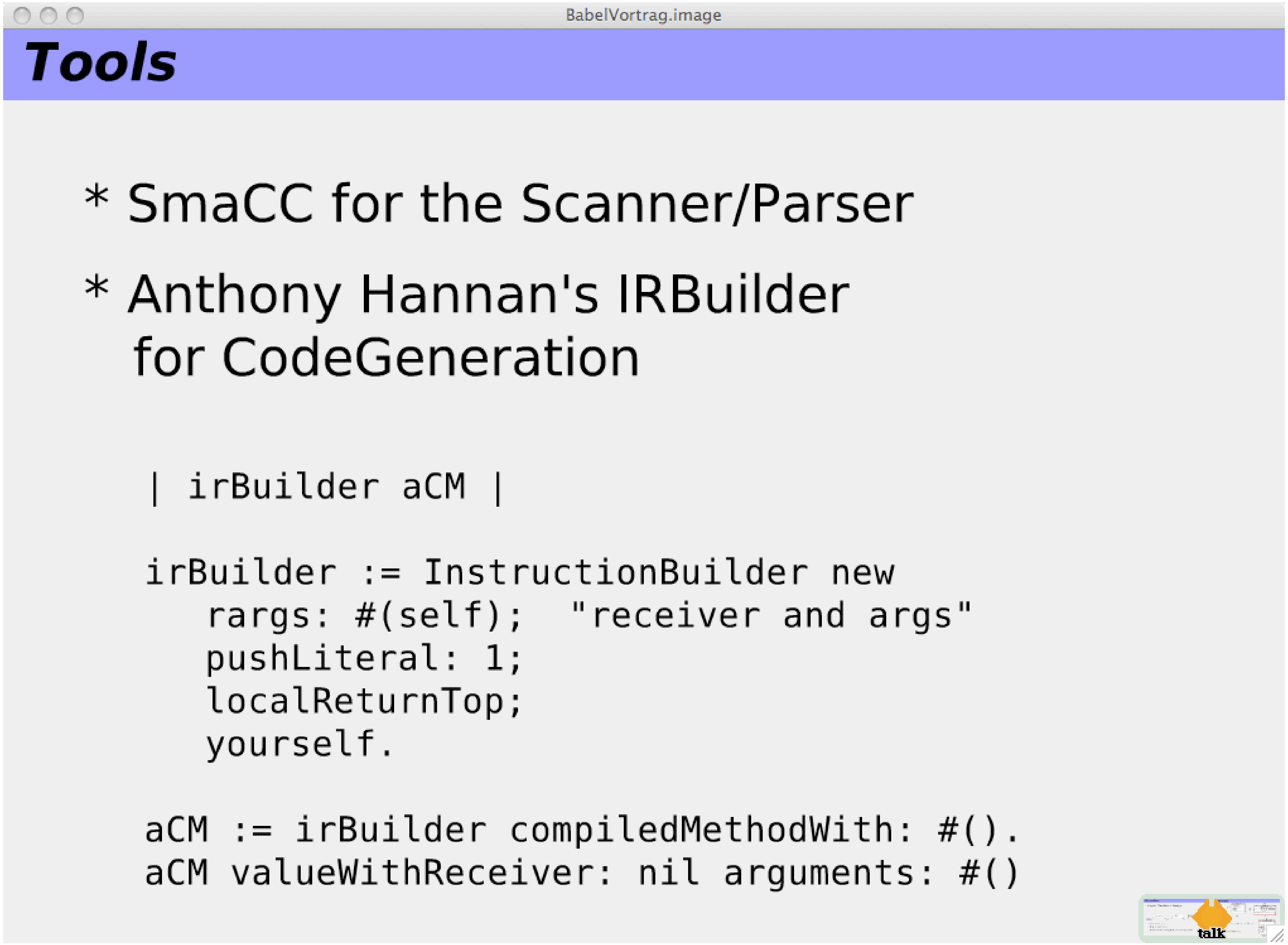
Simple "TextBook" Design:



Speed: Don't Care

Memory: Don't Care

"Do the simplest Thing that could possibly work"



Tools

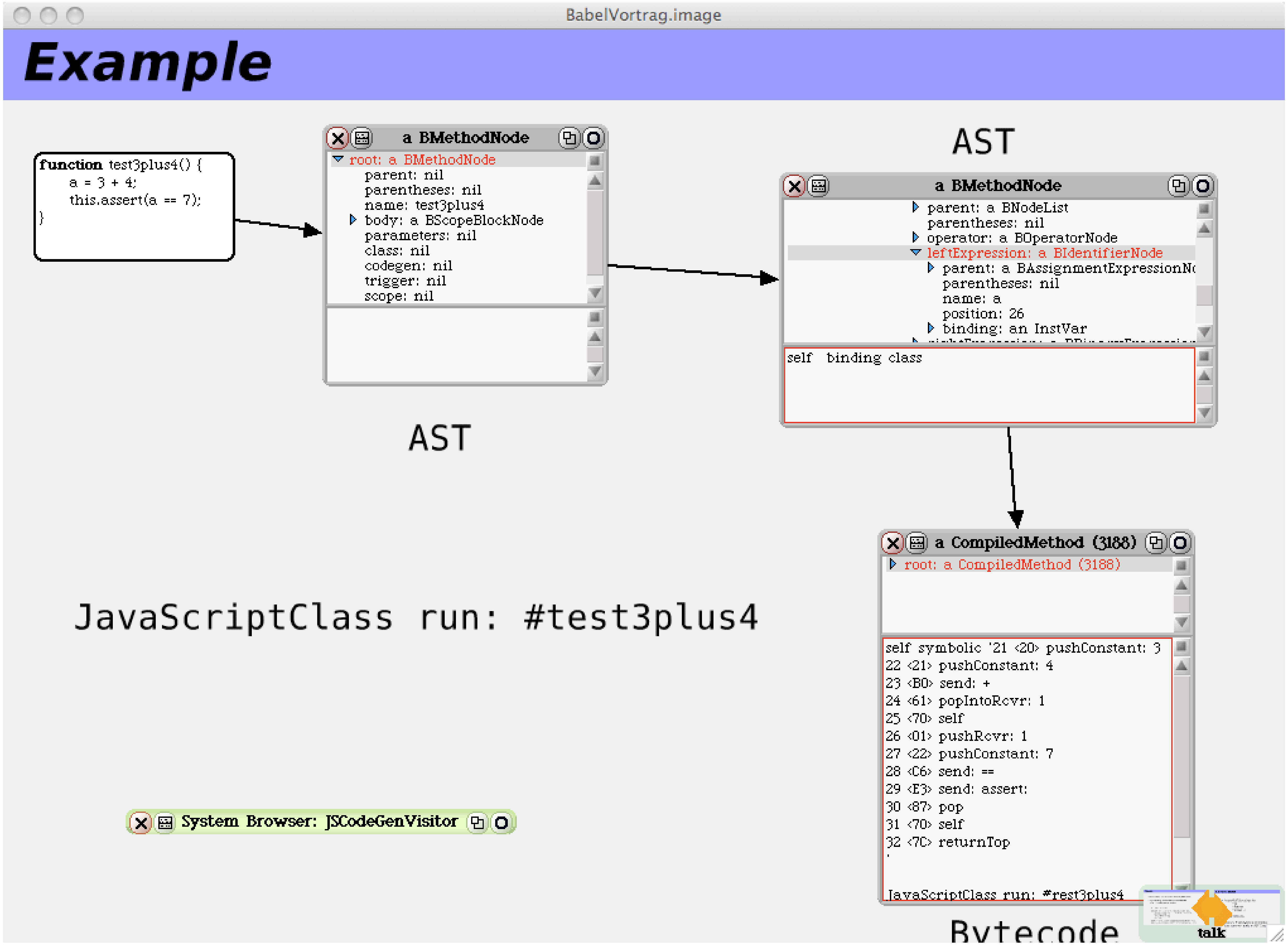
* SmaCC for the Scanner/Parser

* 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);
}
```

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

```
▷ parent: a BNodeList
  parentheses: nil
▷ operator: a BOperatorNode
▽ leftExpression: a BIdentifierNode
  ▷ parent: a BAssignmentExpressionN
    parentheses: nil
    name: a
    position: 26
  ▷ binding: an InstVar
```

```
self  binding class
```

JavaScriptClass run: #test3plus4

```
▷ root: 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: #test3plus4
```

BabelVortrag.image

Current state

-> Parser/AST/CodeGen for
- * JS
- * Python
- * LOGO ;-)

-> Parser for Ruby

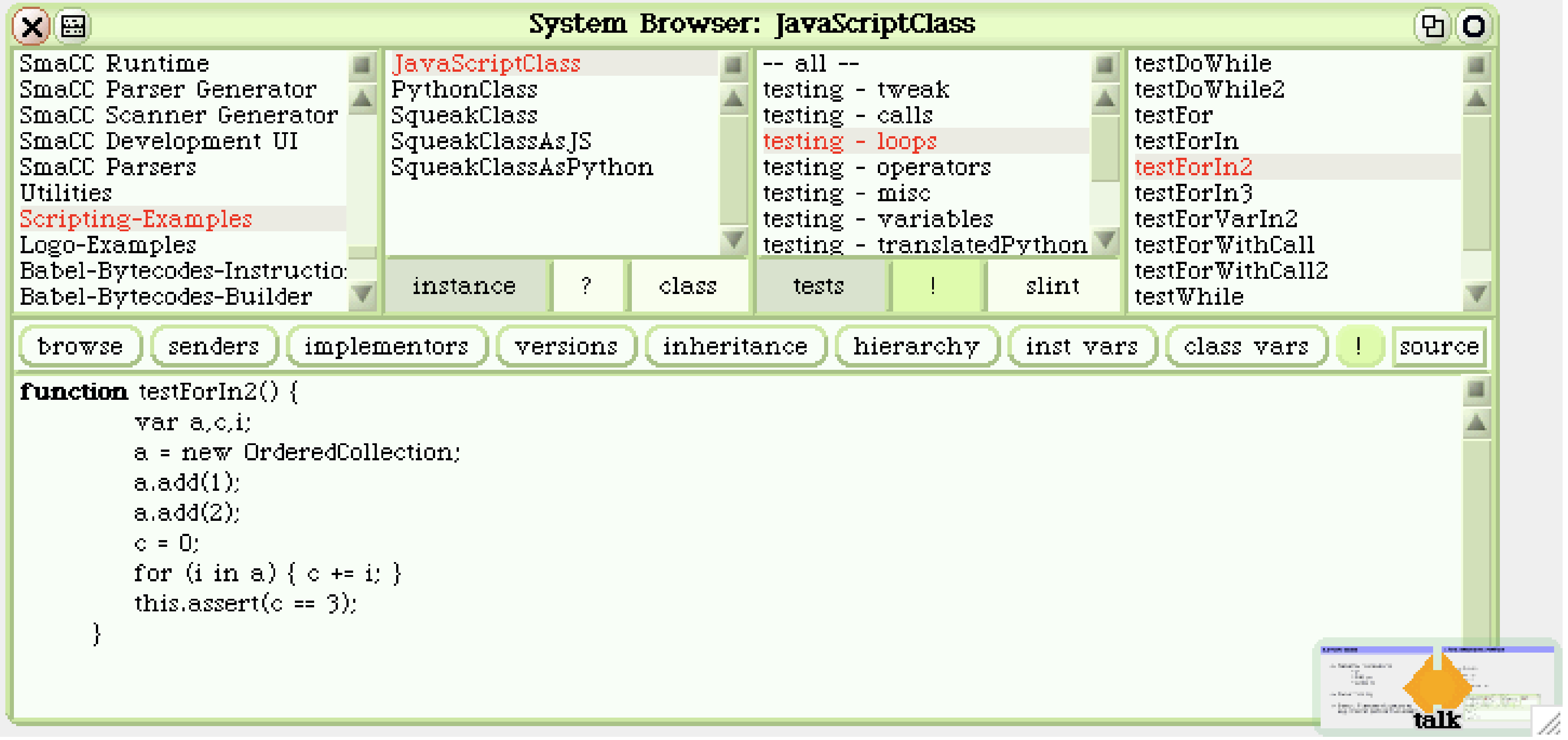
-> Slowly a Framework is emerging
(e.g. common parts of AST Classes)

BabelVortrag.image

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);
    }
```

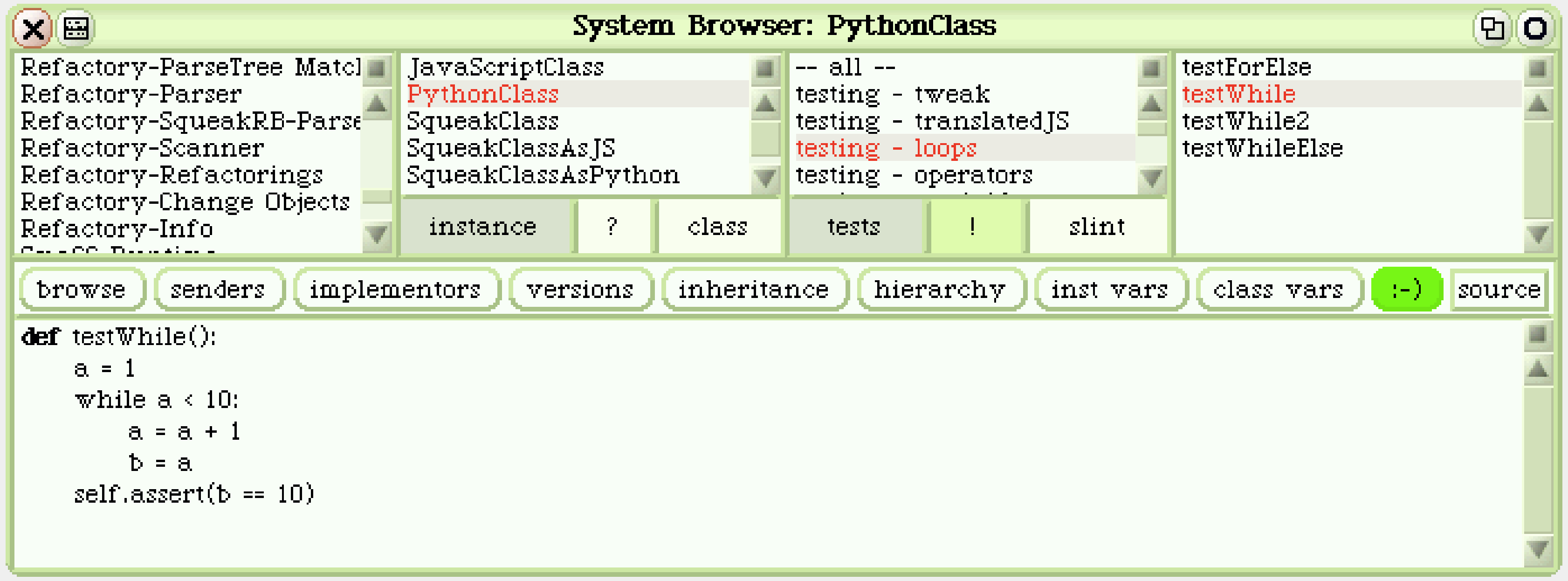


```
function testForIn2() {
        var a,c,i;
        a = new OrderedCollection;
        a.add(1);
        a.add(2);
        c = 0;
        for (i in a) { c += i; }
        this.assert(c == 3);
   }
```

BabelVortrag.image

The Languages: Python

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



```
Refactory-ParseTree Matcl    JavaScriptClass        -- all --                  testForElse
Refactory-Parser             PythonClass            testing - tweak            testWhile
Refactory-SqueakRB-Parse     SqueakClass            testing - translatedJS     testWhile2
Refactory-Scanner            SqueakClassAsJS        testing - loops            testWhileElse
Refactory-Refactorings       SqueakClassAsPython    testing - operators
Refactory-Change Objects
Refactory-Info
```

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

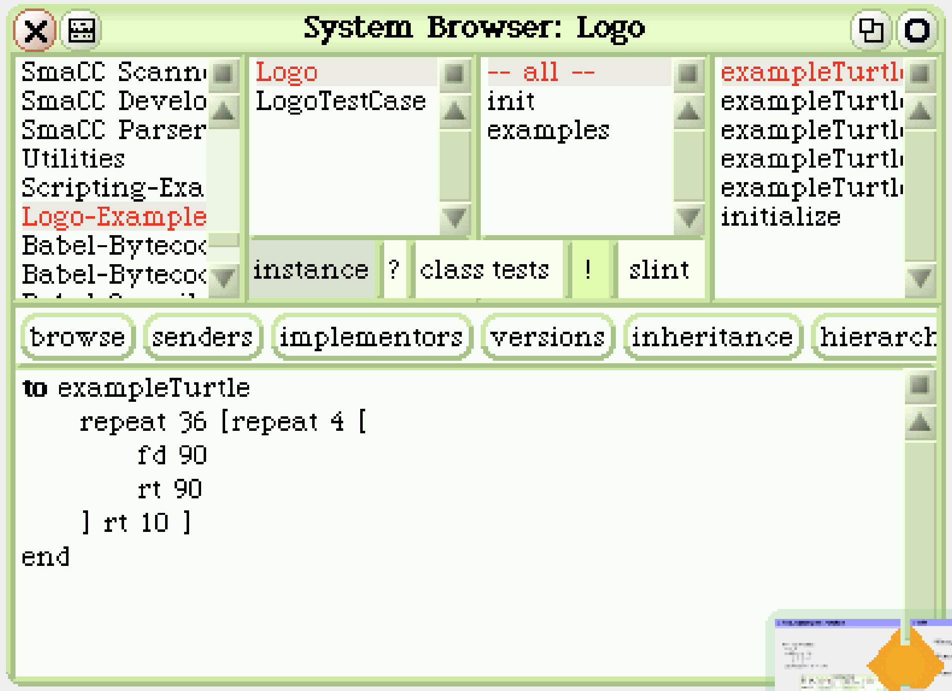
BabelVortrag.image

The Languages: LOGO

A real Compiler.

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

```
Display restoreAfter:
  [Logo new exampleTurtle]
```



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

BabelVortrag.image

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