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
Simple "TextBook" Design:

- Source → Scanner/Parser (SmaCC) → AST → Semantik Analysis → AST (Attrib) → CodeGeneration → Bytecode

- Speed: Don't Care
- Memory: Don't Care
- "Do the simplest Thing that could possibly work"
* SmaCC for the Scanner/Parser

* Anthony Hannan's IRBuilder for CodeGeneration

```plaintext
| irBuilder aCM |

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

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

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

```
JavaScriptClass run: #test3plus4
```

---

### AST

- **Root**: a BMethodNode
  - **Parent**: nil
  - **Parentheses**: nil
  - **Name**: test3plus4
  - **Body**: a BScopeBlockNode
  - **Parameters**: nil
  - **Class**: nil
  - **Codegen**: nil
  - **Trigger**: nil
  - **Scope**: nil

- **LeftExpression**: a BOperatorNode
  - **Parent**: a BAssignmentExpressionNode
    - **Parentheses**: nil
    - **Operator**: nil
    - **Name**: a
    - **Position**: 26
    - **Binding**: an InstanceVar

- **RightExpression**: a BInstanceExpression

### Bytecode

```
System Browser: JSCodeGenVisitor
```

```
JavaScriptClass run: #test3plus4
```

```
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)
JavaScript-like Syntax

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