Variable Tracker

Preparation for Bachelor project
SCG seminar project presentation
Overview

• Bachelor/seminar project
• Implementation
  – Reflectivity
  – How is it done?
  – Caching + Optimization
  – Speed comparison
• Demo
Bachelor project

• Give a man a dynamically typed language, will he write dynamically typed code?

```ruby
|a|
a := 42.
a := 'hello world'.
a := Dictionary new.
```

• Do developers need/use this behaviour?

```ruby
|a b c|
a := 42.
b := 'hello world'.
c := Dictionary new.
```
Seminar project

• Program to monitor how types change during execution
• Data is stored to MongoDB
• Program to analyze the results from the database

http://www.mongodb.org/
Interaction

VariableTracker + Reflectivity

Voyage

MongoDB Java Driver

Analyze results in database
Overview

- Bachelor/seminar project

**Implementation**
- Reflectivity
- How is it done?
- Caching + Optimization
- Speed comparison

**Demo**
Reflectivity

«Reflectivity is a tool to annotate AST nodes with metalinks. A metalink corresponds to a message sent to an arbitrary object.»

Example:

MyClass methods do: [ :method |
    method ast
    forAllNodes: [ :node | node isAssignment ]
    putAfter: [ RFMetalink fromExpression: 'Transcript crShow: ''variable written''' ];
    installWrapper ]

http://smalltalkhub.com/#!/~RMoD/Reflectivity
addWrapper: aMethod

aMethod ast

forallNodes: [:node | node isAssignment]

putAfter: [:node | RFMetalink fromExpression:
'((VariableInformation new',
' variable: ', node variable name asString,
' name: ''', node variable name asString, '''',
' isInstanceVariable: ', node variable isInstance asString,
' class: self class name asString',
' method: thisContext method selector asString',
' isClassSide: self class isClassSide asString',
') save']];

installWrapper.
How is it done?

```
VariableTracker
activateOnClass:
activateOnMethod:
deactivateOnClass:
...
```

```
MyClass
methodDict
```

![Diagram showing relationships between VariableTracker and MyClass with methodDict connections]
# Speed comparison

<table>
<thead>
<tr>
<th></th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Without tracker</strong></td>
<td></td>
</tr>
<tr>
<td>First start</td>
<td>311ms</td>
</tr>
<tr>
<td>Next starts</td>
<td>~264ms</td>
</tr>
<tr>
<td><strong>VariableTracker</strong></td>
<td></td>
</tr>
<tr>
<td>First start</td>
<td>16.5s</td>
</tr>
<tr>
<td>Next starts</td>
<td>~11s</td>
</tr>
</tbody>
</table>

**Action:**
Starting Nautilus browser

<table>
<thead>
<tr>
<th></th>
<th>Variables written</th>
</tr>
</thead>
<tbody>
<tr>
<td>First start</td>
<td>~800</td>
</tr>
<tr>
<td>Next starts</td>
<td>~160</td>
</tr>
</tbody>
</table>
Caching + Optimization

VariableTracker
activateOnClass:
activateOnMethod:
deactivateOnMethod:
...

originalMethods

MyClass

methodDict

01001
10011

01001
10011
# Speed comparison

<table>
<thead>
<tr>
<th>Action: Starting Nautilus browser</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Without tracker</strong></td>
</tr>
<tr>
<td>First start</td>
</tr>
<tr>
<td>Next starts</td>
</tr>
</tbody>
</table>

| **VariableTracker without caching** |
| First start                           | 16.5s                          |
| Next starts                           | ~11s                           |

| **VariableTracker with caching** |
| First start                           | 1.496s                         |
| Next starts                           | ~413ms                         |

- First call: **11x** faster
- Next calls: **26x** faster
Overview

• Bachelor/seminar project
• Implementation
  – Reflectivity
  – How is it done?
  – Caching + Optimization
  – Speed comparison
• Demo