The present and future of Pharo
What?

A progressive, open-source Smalltalk platform for professional use.
What?

A flexible environment to support the research of new language concepts.
What?

Pharo = Language + IDE

Pure Object-Oriented Language

Dynamically Typed
History

Based on Squeak Smalltalk

+ Major Cleanups (MVC, eToys)
+ New UI Look / TrueType
+ Tools
+ Block Closures
+ **Lots** of bugfixes and small improvements
699 Bug-reports closed

377 Updates

Release Planned: August 2009
methodReturnTop

"Return Top Of Stack bytecode."
changeNotifier

notify: self

ofSystemChangesOfItem: AbstractEvent methodKind
change: AddedEvent changeKind using: #methodAdditionNotifiedWith;

notify: self

ofSystemChangesOfItem: AbstractEvent methodKind
change: ModifiedEvent changeKind using: #methodModificationNotifiedWith;
Getting Started
(if needed)
seaside The fast lane to HTTP

About
- Screenshots
- Success Stories
- Examples
- What others think
- Hosting
- Support
- Trivia
- Merchandise

Documentation
- FAQ
- Tutorials
- Migration
- Videos

Community
- Weblogs
- Mailing Lists
- Development
- Contribute
- Merchandise
- Extensions
- Projects

News
- [Smalltalk Tidbits, Industry Rants] Industry Misinterpretations 144: Children of Smalltalk 12 July 2009
  You'll have to listen to the entire podcast to understand where the title comes from :) This week ...

- London Geek Night 10 July 2009
  I have some pictures to share for people that missed the London Geek Nights with the title Seaside: ...

- Web Velocity released 29 June 2009
  Big news from Cincom on the Seaside mailing list today: they've made an evaluation version of their ...

- London 23 June 2009
  I'll be in London (UK) on Wednesday June 24 for 10 days or so. If anyone around there wants to meet ...

- Seaside 2.8.4 and Pharo 5 June 2009
  We just announced Seaside 2.8.4, the fourth maintenance release for Seaside 2.8.Again this release ...
Pier is a content management system that is light, flexible and free. It is light as in double click and go (download it). It is flexible as in make it be what you want (learn more). It is free as in freedom (MIT license).
Smalltalk with OO-Database

Pharo is the IDE
Future
...cleaner
...smaller
...faster
(of course)
Enable Evolution
Enable Experiments
Examples

- Compiler
- Slots
- JIT and AOStA
Old Compiler
- From 1976
- Not OO
- Hard to understand. Hard to change

We need a better Compiler!
New Compiler

- Originally by Anthony Hannan
- Based on Visitors
- Reusable and Pluggable Backend (IRBuilder)
- Uses RB AST
- SmaCC Parser

Easier to understand.
Easier to change
New Compiler

Code generation in detail

Scanner / Parser → AST → Semantic Analysis → AST → Code Generation → Bytecode

Build IR → IR → Bytecode Generation → Bytecode

ASTTranslator
IRBuilder

IRTranslator
BytecodeBuilder
Plans...

- New BlockClosure Format
- Plugin architecture: Parser, Checker, CodeGen
- Use RB Parser (Faster, Error messages)
Examples

- Compiler
- Slots
- JIT and AOSTA
Instance Vars

- Not objects
- Change behavior?
- Change memory layout?

Instance state should be more powerful!
Slots

- Are objects
- Allow custom subclasses
- Change behavior and memory layout

Without Performance penalty!
Auto-Accessor

Create accessor-methods at compile-time
MopObject subclass: #SlotExample
slots: '{ActiveSlot named: #hello
   action: [Beeper beep]}'
classVariableNames: ''
poolDictionaries: ''
category: 'SlotExamples'

Evaluate block on read
Sparse Slot

MopObject subclass: #SlotExample
  slots: '{SparseSlot named: #a.
  SparseSlot named: #b.
  SparseSlot named: #c}'
  classVariableNames: ''
  poolDictionaries: ''
  category: 'SlotExamples'

Store values in one Dictionary
iVar Slot

Normal instance Variable
MopObject subclass: #SlotExample

slots: '{AutoAccessorSlot named: #hello}'

classVariableNames: ''

poolDictionaries: ''

category: 'SlotExamples'

Create accessor-methods at compiled-time
Slot subclass: #AutoAccessorSlot

instanceVariableNames: ''

....

postCreationAction

self generateGetter.

self generateSetter.

generateGetter

class compile:

(String streamContents: [:stream |stream

nextPutAll: self name; crtab;

nextPutAll: '^', self name asString])
MopObject subclass: #SlotExample

slots: '{ActiveSlot named: #hello
  action: [Beeper beep]}'

classVariableNames: ''

poolDictionaries: ''

category: 'SlotExamples'

Evaluate block on read
AutoAccessorSlot subclass: #ActiveSlot
    instanceVariableNames: 'action'

    emitReadUsing: methodBuilder
        methodBuilder
            pushLiteral: action;
            send: #value.
        super emitReadUsing: methodBuilder
Status

- First Prototype
- Integration with ClassBuilder
- Compatibility with iVars
- Explore different designs
Examples

- Compiler
- Slots
- JIT and AOSTa
VM

- Virtual Machine uses Interpreter

- JIT Compiler under development (Eliot Miranda)

How to optimize even more?
AOStA

- Provide API for accessing runtime data
- PIC: Polymorphic Inline Cache
  Record class per send instruction
- Hotness Counter
  Where do we spend time?
Optimize SSA-Form
(Static Single Assignment)
AOStA

- Use PIC-Data for specialization
- Inline often-called methods
- Special Bytecode for primitive types (Floats)
AOStA

- Bytecode-to-Bytecode optimization
- Purely realized in Smaltalk
- Accessible in the image. Reflection?
More...

- Improve Tools
- Real Modules
- Reflection
- ........
Join Us!

Goal: learning and having fun

http://pharo-project.org