FREE eBook

LEARNING pharo

Free unaffiliated eBook created from
Stack Overflow contributors.

#pharo

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: pharo

It is an unofficial and free pharo ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official pharo.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with pharo

Remarks

Pharo provides an excellent way to combat complexity in software development. With its fast feedback with instantaneous compilation of the currently edited method, its objects all the way from compiler to the run-time stack, and tools designed to be customizable by the individual developer, it supports programming your way out of a mess again. It has excellent refactoring tools and makes it easy to build domain specific languages. It values innovation and effectiveness over run-time efficiency.

Examples

Installation or Setup

Easy Install

Go to http://pharo.org/download and select a fitting download and run it.

Details

There are a lot of different ways to install Pharo. Pharo itself consists of a vm and an image. In addition it needs its sources and plugins, and has some dependencies:

- It is a cross-platform environment, running on OS-X (and iOS), Windows and several unix variants (a.o. Ubuntu and Android).
- It runs on a virtual machine that can run on several processor architectures (Intel, ARM). The virtual machine is shared with Squeak, Cuis and Newspeak. With Pharo 5 a new and much faster vm has been introduced using a different image format and FFI.
- There are 32-bit and 64-bit variants.
- In addition to the standard image there is the PharoLauncher that integrates with our CI infrastructure and supports downloading and running all kinds of images, a.o. preconfigured seaside, magritte and moose ones, older releases and the latest development versions of Pharo.

Read Getting started with pharo online: https://riptutorial.com/pharo/topic/8799/getting-started-with-pharo

Chapter 2: Creating a Class

Introduction

Creating a class in Pharo is as simple as sending `subclass:` message to another class object. But most of the classes are created and modified in a system browser (currently Nautilus).

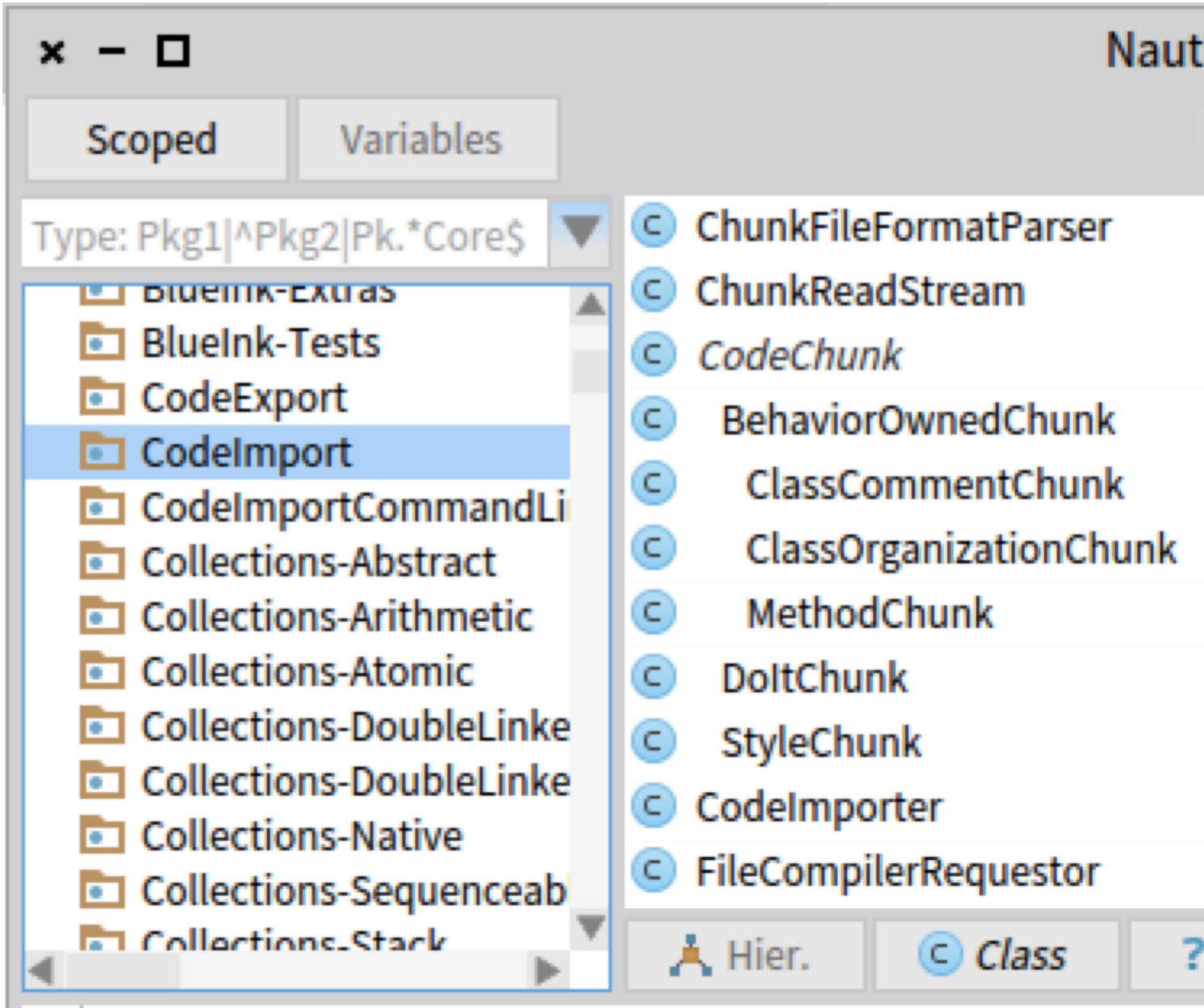
Examples

Adding a class in a system browser

The most common way to add (and edit classes) in from the system browser In the Nautilus system browser have nothing selected or select a package (first column) or a class (second column). Depending on the selection, the code editor will display a slightly different class template:

Selection	Template
None	Empty class template
Package	A class template with pre-filled package name (based on the selected package)
Class	An actual definition of the selected class

The following image demonstrates the Nautilus window with a selected package:


Naut
Scoped
Variables
Type: Pkg1|^Pkg2|Pk.*Core$
BlueInk-Tests
CodeExport
CodeImport
CodeImportCommandLi
Collections-Abstract
Collections-Arithmetic
Collections-Atomic
Collections-DoubleLinke
Collections-DoubleLinke
Collections-Native
Collections-Sequenceab
ChunkFileFormatParser
ChunkReadStream
CodeChunk
BehaviorOwnedChunk
ClassCommentChunk
ClassOrganizationChunk
MethodChunk
DoItChunk
StyleChunk
CodeImporter
FileCompilerRequestor
Hier.
Class
Object subclass: #NameOfSubclass
instanceVariableNames: ''
classVariableNames: ''
package: 'CodeImport'

. If you don't have any particular superclass in your mind it's advised to subclass from `Object` but this brings a tautological confusion into the previous example. Let's say that you want to create a class *PriorityStack* with class *Stack* as a superclass, then you need to evaluate:

```
Stack subclass: #PriorityStack
```

Anonymous class

You can create classes without names that are not installed in the system by sending `newAnonymousSubclass` to a class.

For example

```
anonymousSet := Set newAnonymousSubclass
```

will assign an anonymous subclass of Set to `anonymousSet` variable. Then you can compile methods in this class and instantiate it, or swap it with a real class.

Useful for test resources of for proxying

Read Creating a Class online: https://riptutorial.com/pharo/topic/8834/creating-a-class

Chapter 3: Seaside

Introduction

Seaside is a web framework for Pharo and other smalltalks. It is ideal for complex applications with a rich domain model.

Examples

Droppable

A Seaside component (subclass of `WAComponent`) needs to override `#renderContentOn:`. It is a smalltalk class that can use all the normal ways of structuring an application. Here it delegates to three different methods.

```
JQDroppableFunctionalTest>>renderContentOn: html
   self renderInstructionsOn: html.
   self renderInventoryOn: html.
   self renderSelectedOn: html
```

As a parameter it gets a html canvas object that understands messages relevant to building up the html and javascript. It uses a fluent interface, where `#with:` is the last message send to the current canvas context.

```
JQDroppableFunctionalTest>>renderInventoryOn: html
   html div class: 'inventory ui-corner-all'; with: [
      self colors do: [ :each |
         html div
            class: each;
            passenger: each;
            script: (html jQuery new draggable
               revert: true) ] ]

JQDroppableFunctionalTest>>renderSelectedOn: html
   html div
      class: 'selected ui-corner-all';
      script: (html jQuery new droppable
         onDrop: (html jQuery this load
            callbackDroppable: [ :event |
                self selected add: (event at: #draggable) ];
            html: [ :r | self renderSelectedItemsOn: r ]));
      with: [ self renderSelectedItemsOn: html ]

JQDroppableFunctionalTest>>renderSelectedItemsOn: html
   self selected do: [ :each |
      html div
         class: each;
         passenger: each;
         script: (html jQuery new draggable
            onStop: (html jQuery this effect
               percent: 1; puff;
```

```
                onComplete: (html jQuery this parent load html: [ :r |
                   self selected remove: each ifAbsent: [ ].
                   self renderSelectedItemsOn: r ])) ]
```

Read Seaside online: https://riptutorial.com/pharo/topic/8805/seaside

Credits

S. No	Chapters	Contributors
1	Getting started with pharo	Community, Stephan Eggermont
2	Creating a Class	Uko
3	Seaside	Stephan Eggermont