How to customize Pharo tools?

Extending Pharo with a Visualization

Milton Mamani Torres - 04/03/2022

WHO AM I?

- Milton Mamani Torres

- Bolivian

- Rmod Software Engineer

- Roassal developer for 7 years

- https://github.com/akevalion

Understanding code
In this case Baselines

- Creating a project is fascinating. But it can be easy to lose yourself in the projects.

- Big projects are hard to understand, not for machine, for us.

- That's why we have structures, rules, tools to code, to explain the source code.

Current state

Baseline = project configuration

- One specific tool is Baseline in Pharo.

- A baseline is a kind of map to load projects from repositories.

- This prototype that works with Baselines, is embedded in the system browser.
  Eg: BaselineOfMorphic.
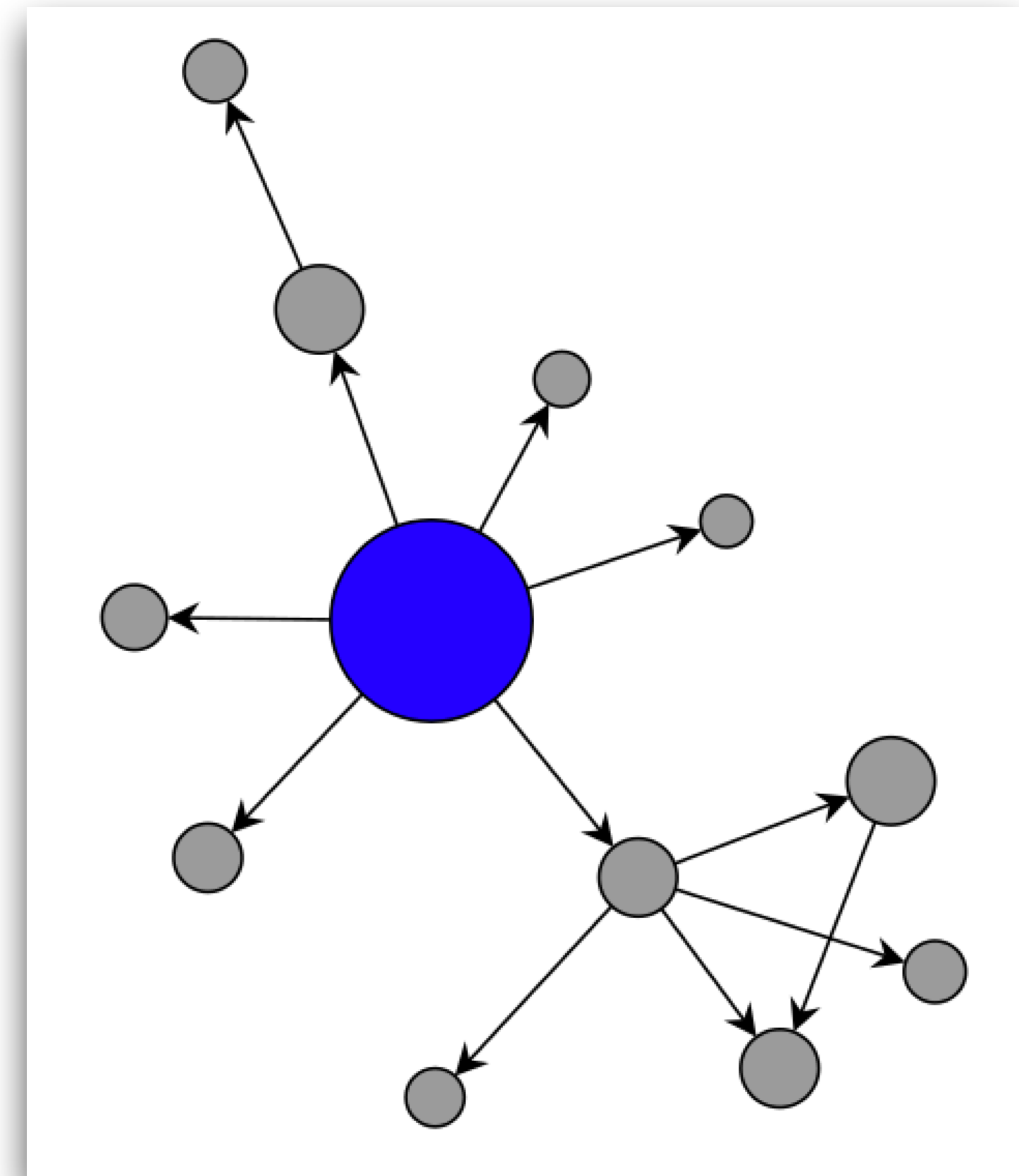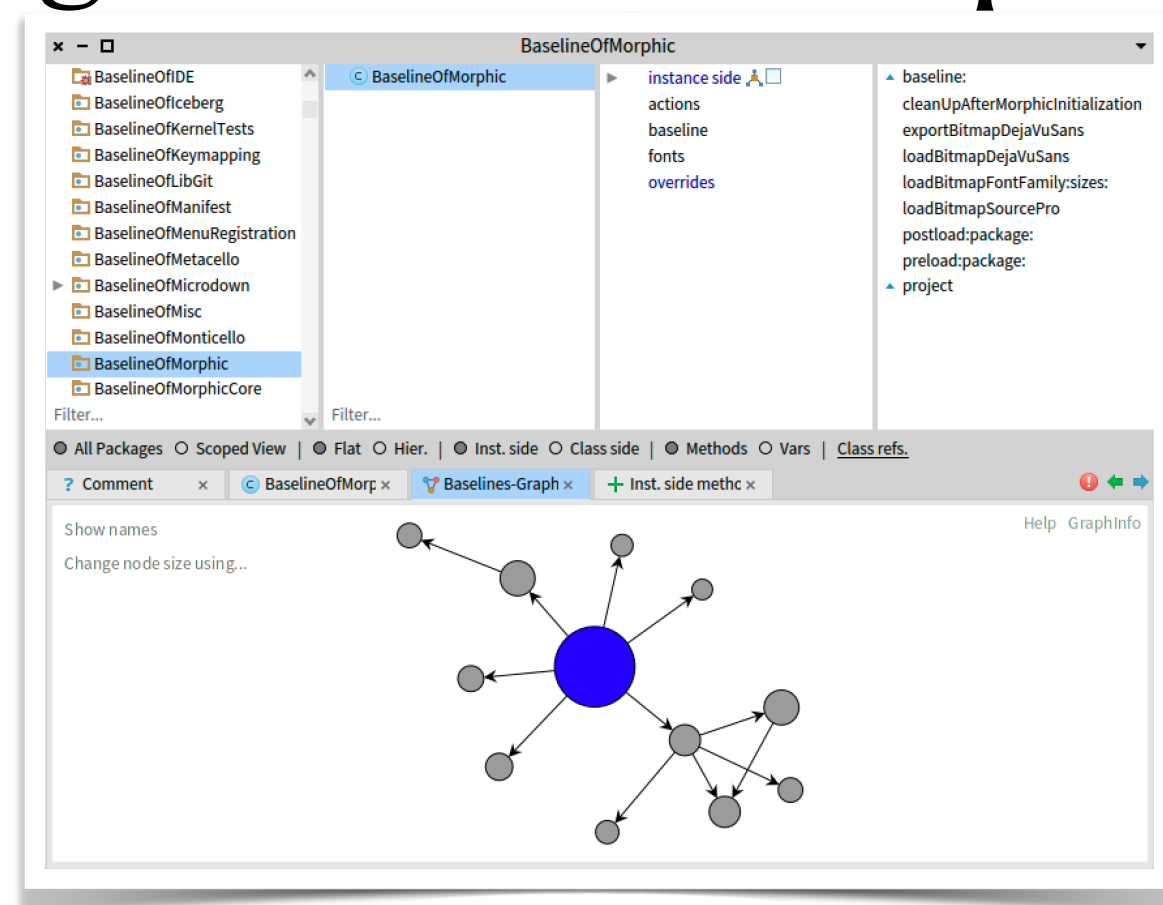
Challenge

Adding a new dependency visualization

- New visual tool with boxes.

- It will be available on system browser, inspector, debugger, etc. if necessary.

- It provides a path to grow and be extended by other users.

- For example this mock image made by hand:

BaselineOfIDE			

BaselineOfMorphic	BaselineOfMetachello	BaselineOfCalypso

BaselineOfKeyMap	BaselineOfFuel	BaselineOfCommander	BaselineOfClassAnnotation

BaselineOFuelPlatform
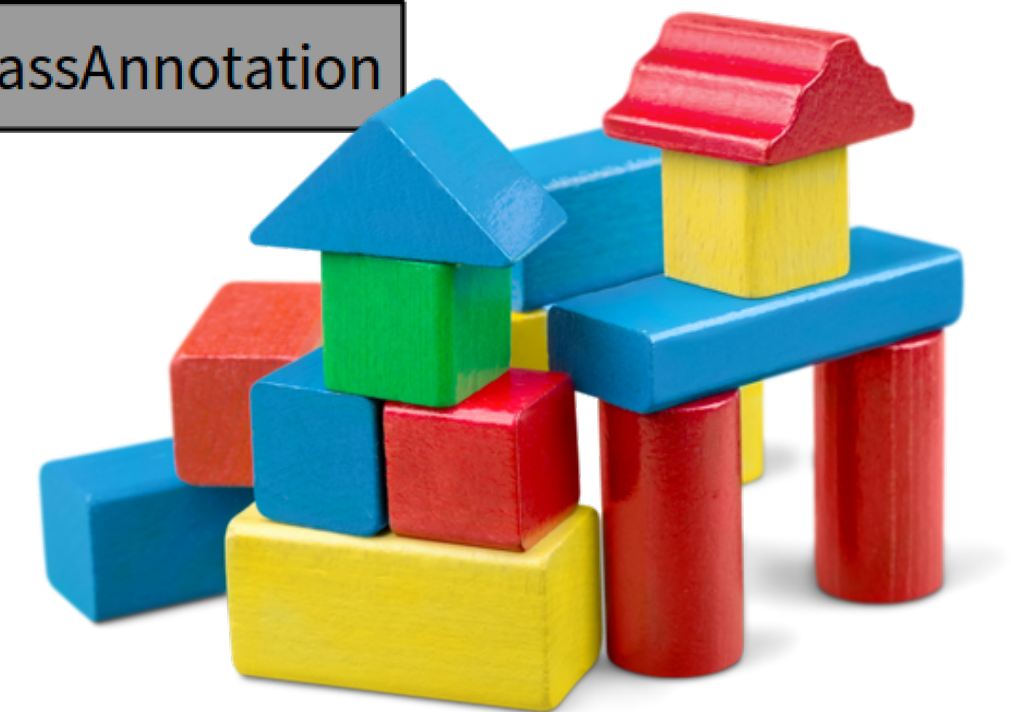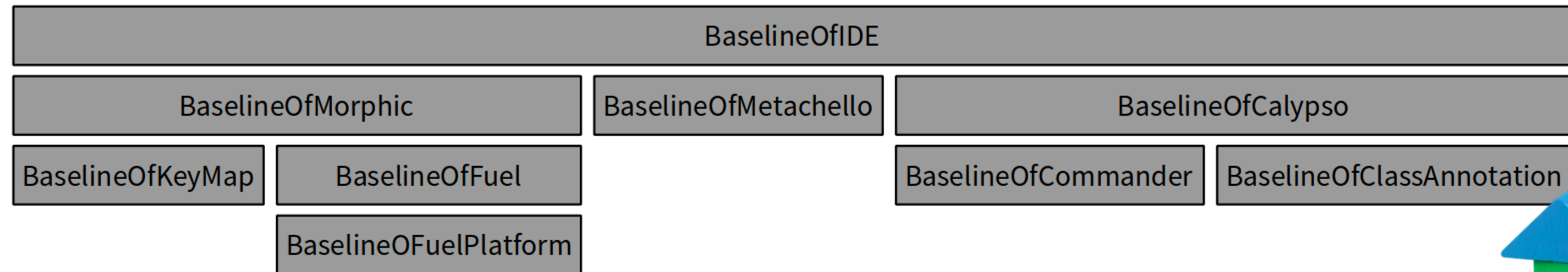
This talk you will see how to create a basic visualization and transform it into a useable tool.

Starting part

In a playground window

In the beginning I started with model and UI.

```
| model modelFor visualization visualizationFor |
modelFor := [ :baseline | self flag: 'TODO' ].
model := modelFor value: BaselineOfMorphic.
visualizationFor := [ :anObject| self flag: 'TODO' ].
visualization := visualizationFor value: model.
visualization open.
```
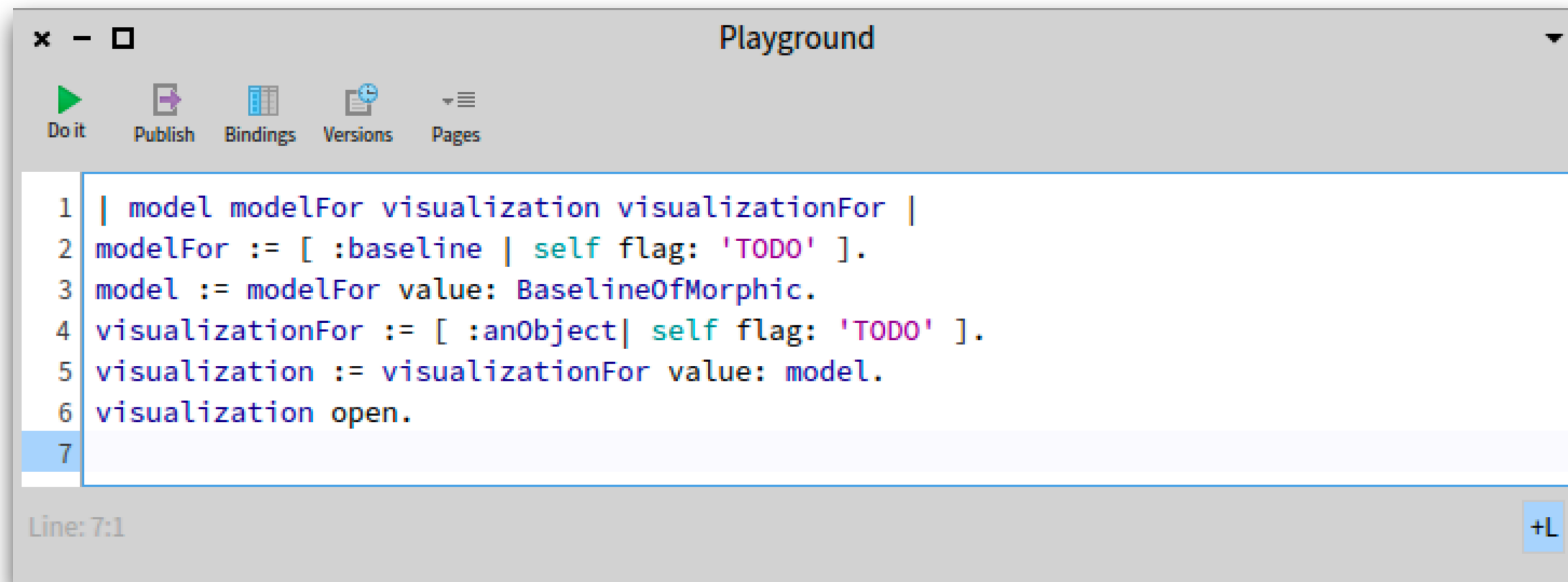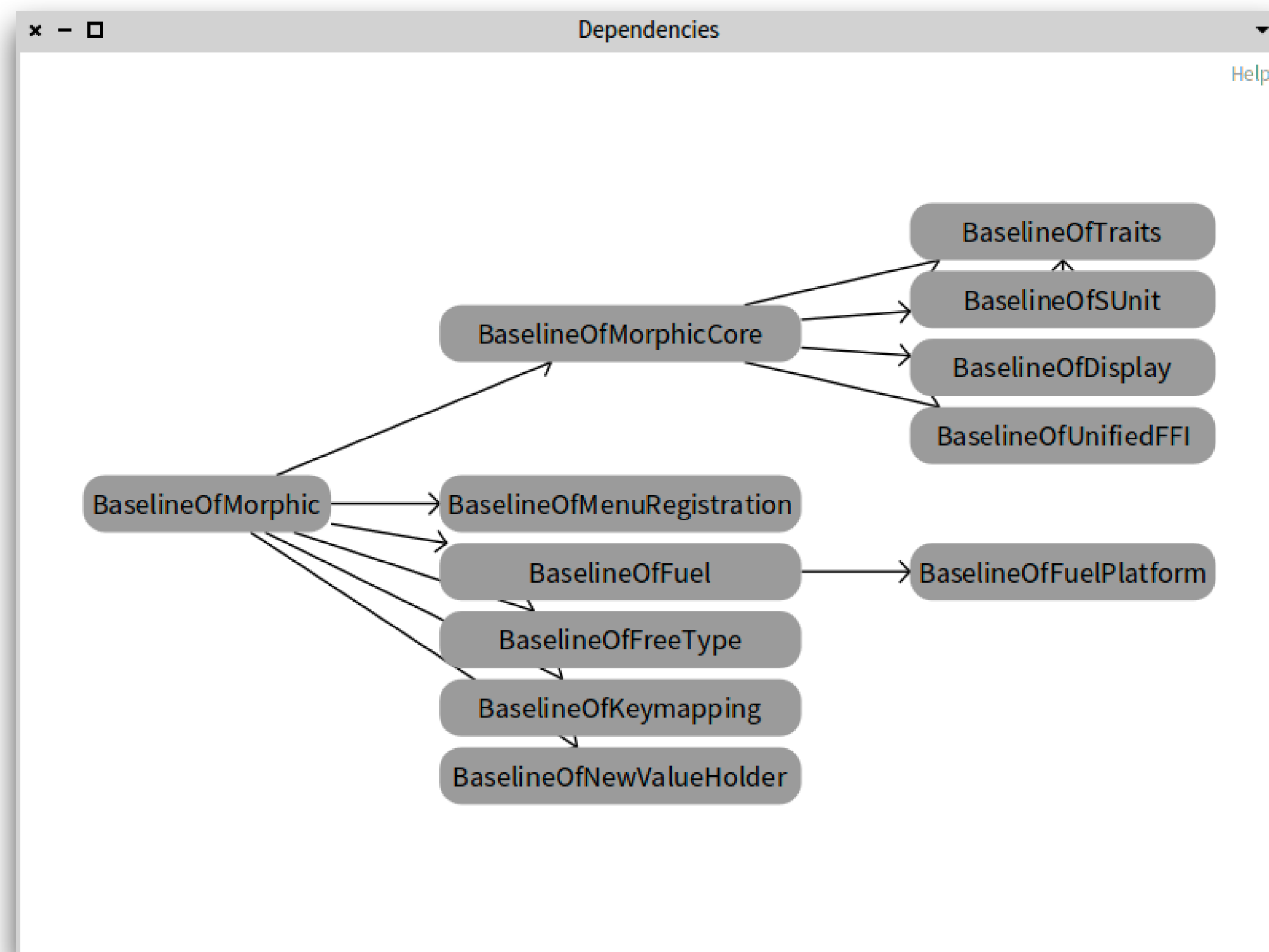
Starting part

In a playground window

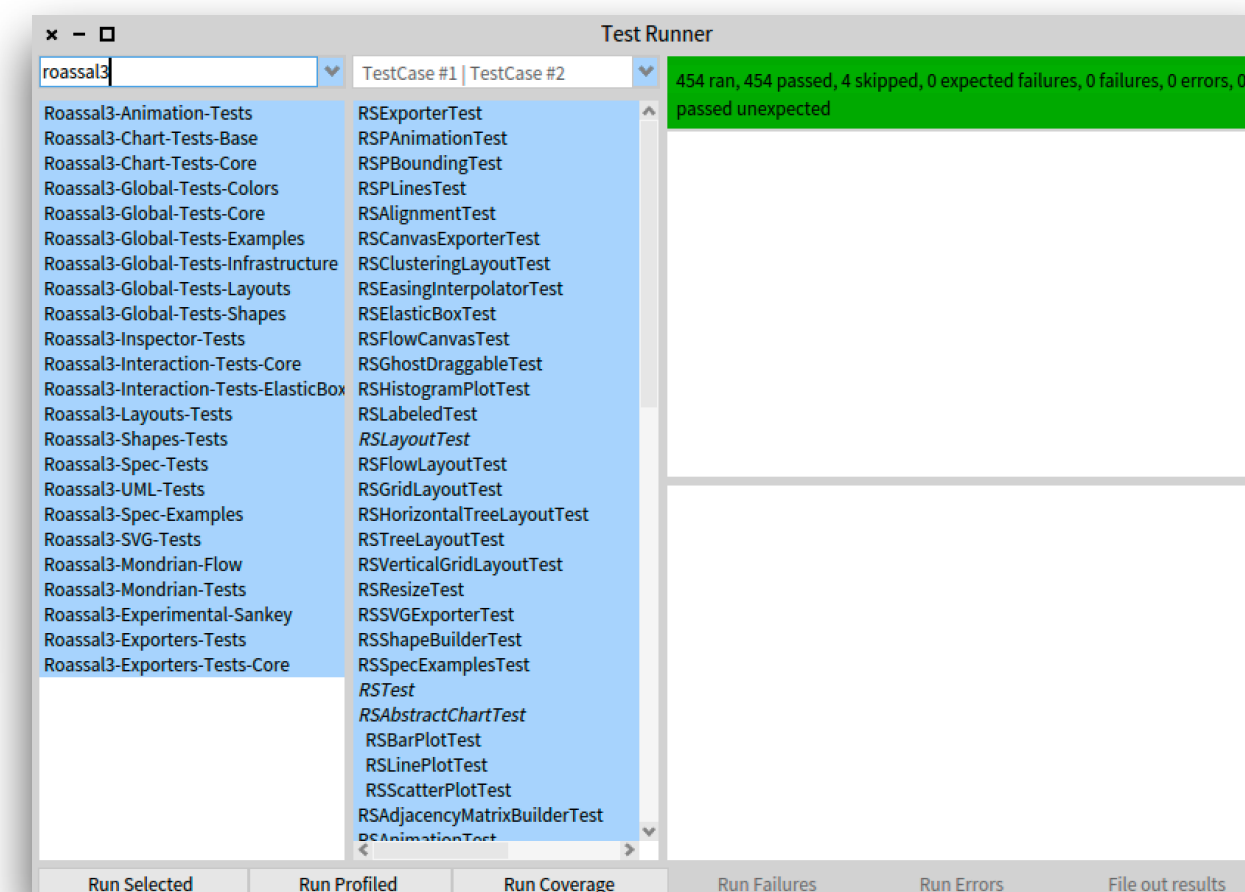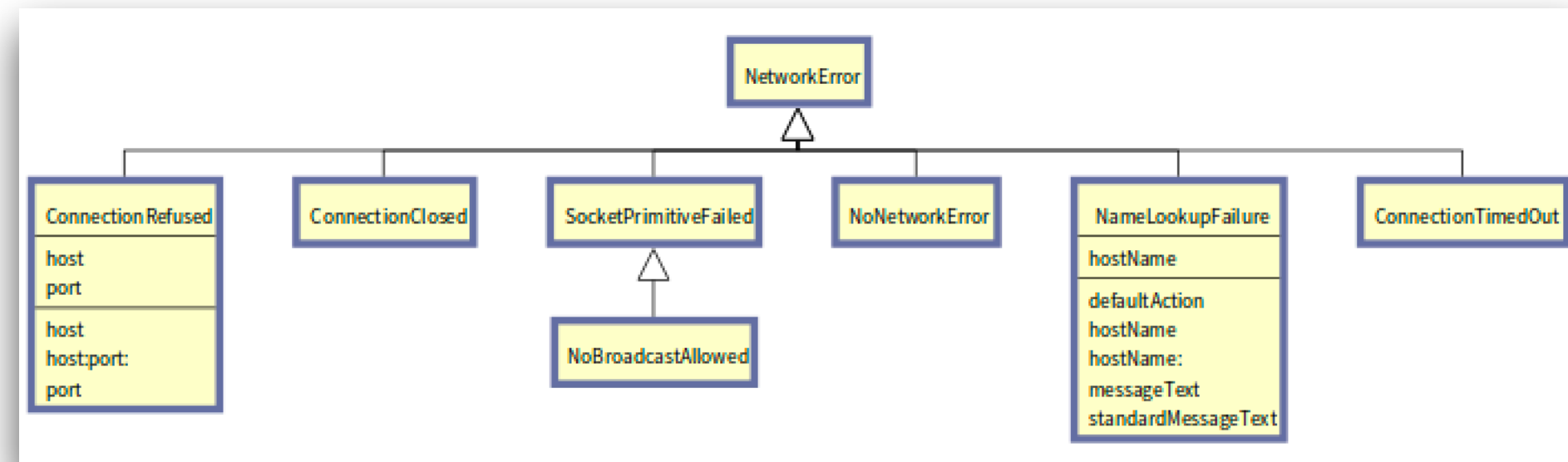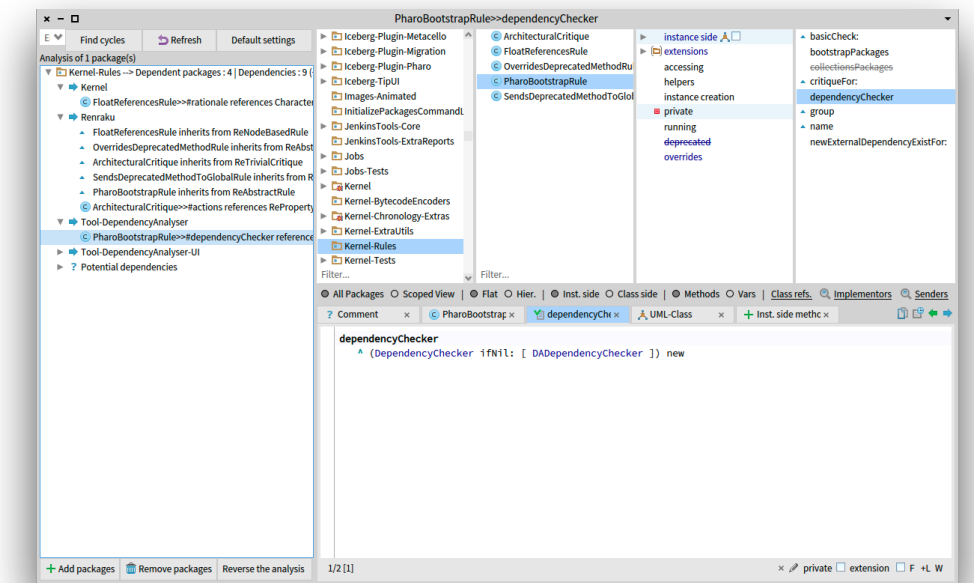At the end I create 85 lines of code that produces a basic visualization.

Next step: classes - packages

- Classes enables a better structure for big projects

- The project can use tests classes, and test runners

- Scalable, classes can be transformed into a major project

- More tools like dependency analyzer, repositories, versions, documentation, etc.

- Reutilizable, users choose to use or not your tool.
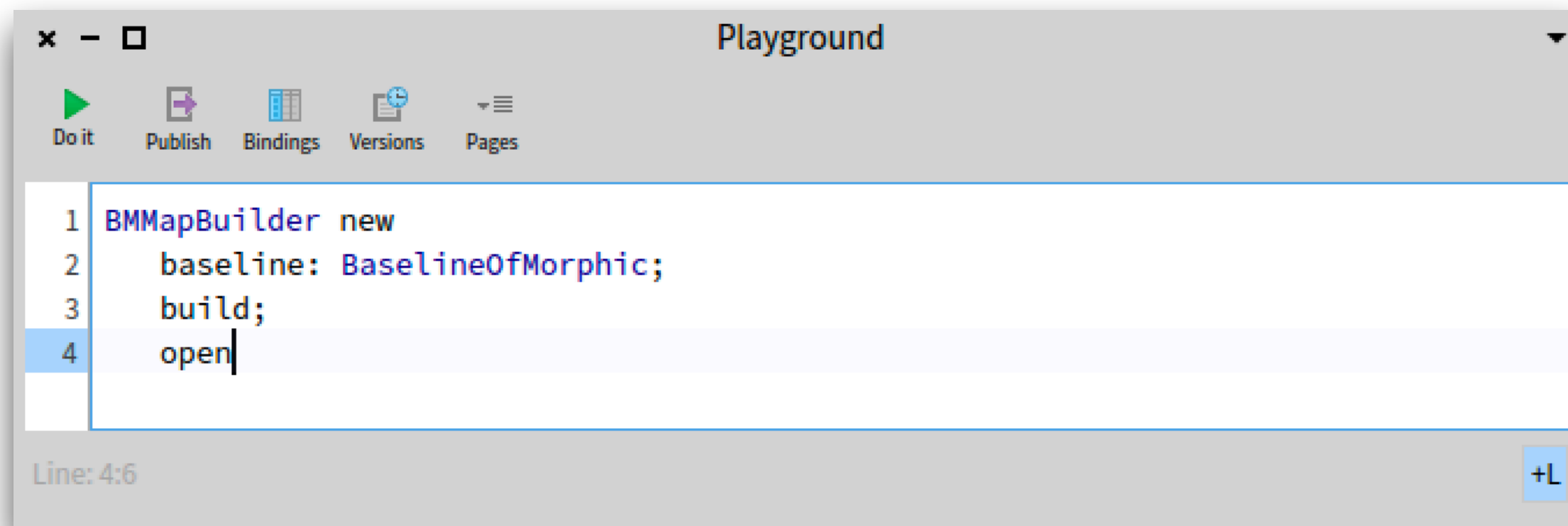
BaselineMap

The new project

- I have created a GitHub repository called https://github.com/akevalion/BaselineMap

- Then I have moved all the code from the script into a class called, **BMMapBuilder** now it has 105 lines of code.

- It should be easy to read, by other people

- It should be easy to modify avoiding hundreds of changes for new requirements
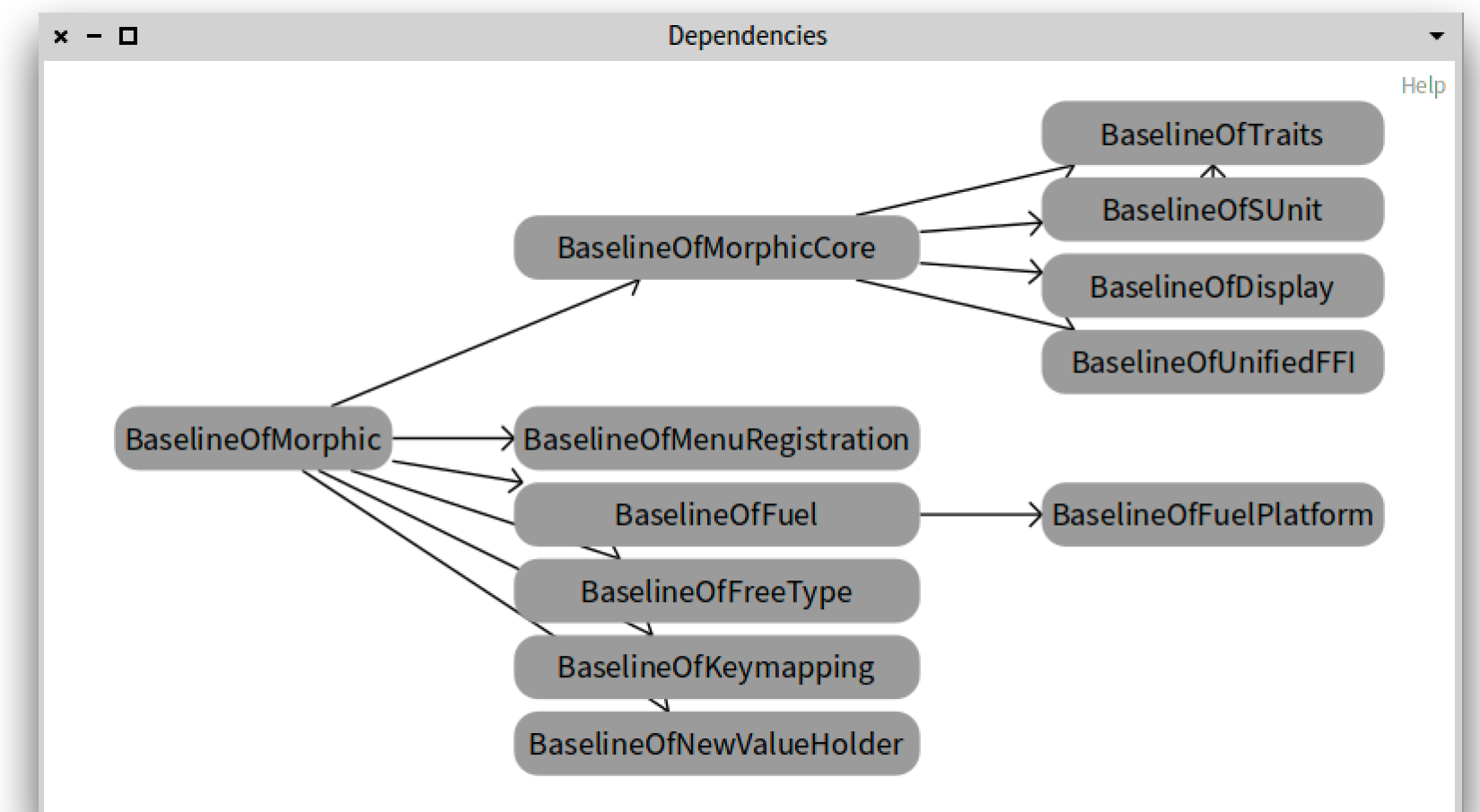
Baseline map

In the system browser

- Currently system browser of Pharo is made in Calypso project, and it provides a system to plug in new visual tools.

- ClyBrowserToolMorph is the entry here to add new tools to browser.

- By creating a new subclass of ClyBrowserToolMorph you can add a new tool.
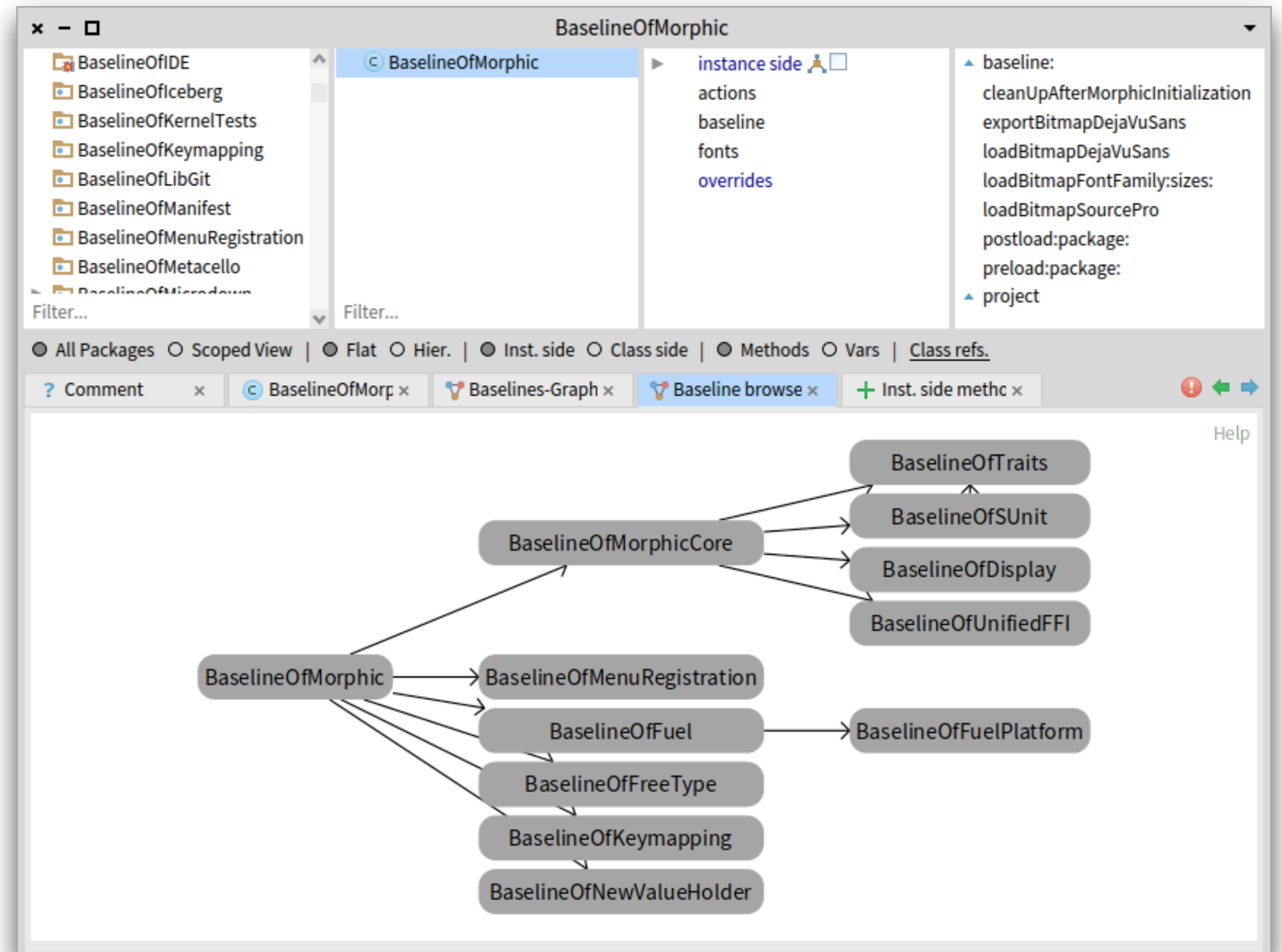
Baseline map

In the inspector

- Inspector in other hand provides an annotation for an object inspector:

```
1  inspectionBaselineMap
2      <inspectorPresentationOrder: 10 title: 'Dependencies Map'>
3      | canvas |
4      canvas := BMSettings classMapBuilder new
5          baseline: self class;
6          build;
7          canvas.
8
9      ^ SpRoassal3InspectorPresenter new
10         canvas: canvas;
11         yourself
```
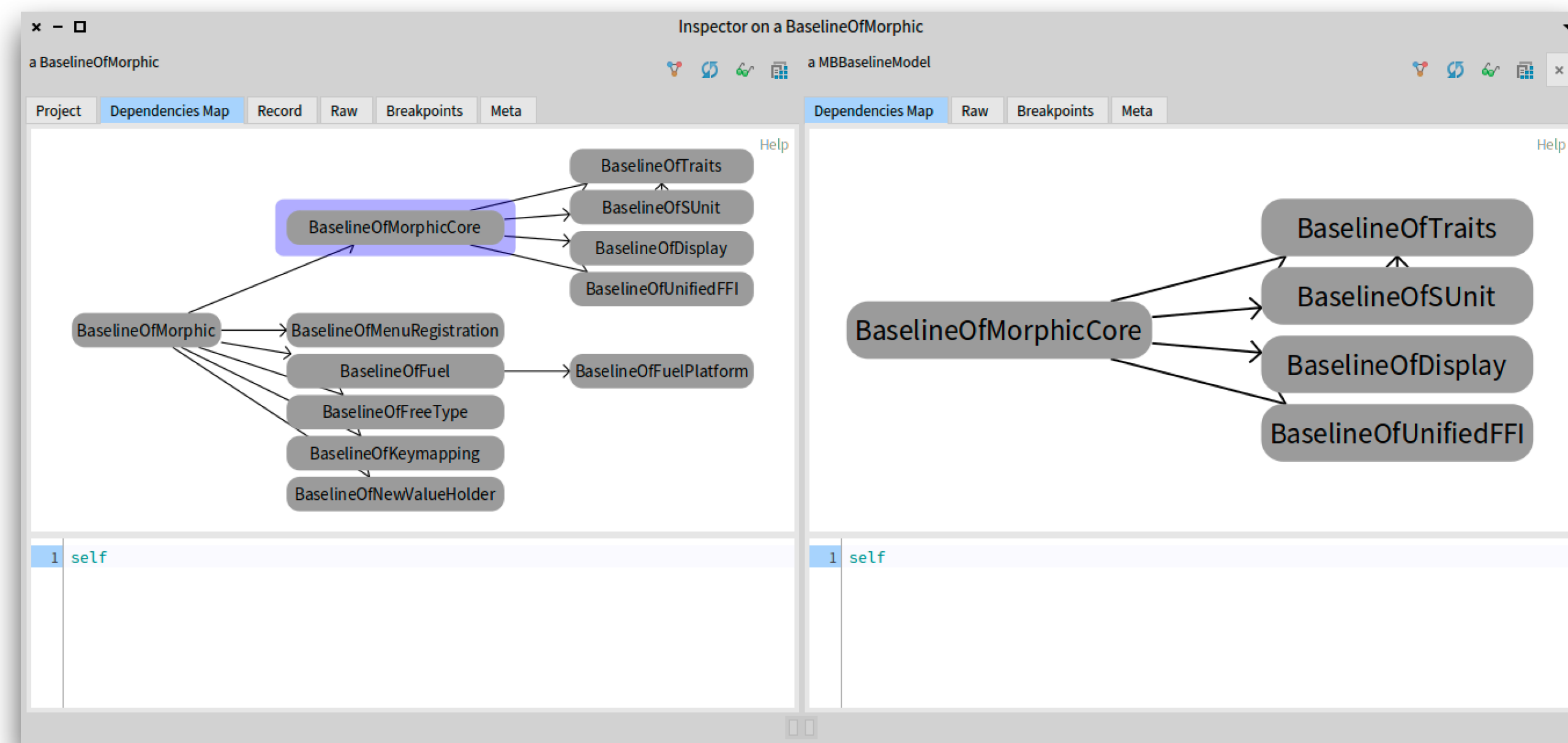
BaselineMap

Structure to be extended

- With hooks methods, object models, design patterns, tests. This class is ready to be extended in a future.

- Github opens collaboration between users => community.

- Test packages guarantee that my code within my parameters works. Also it can be used by Github CI

Customization

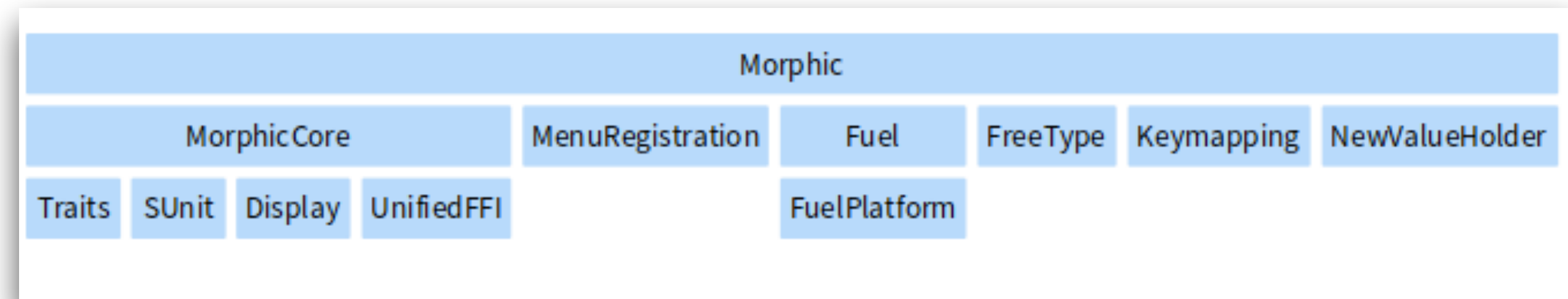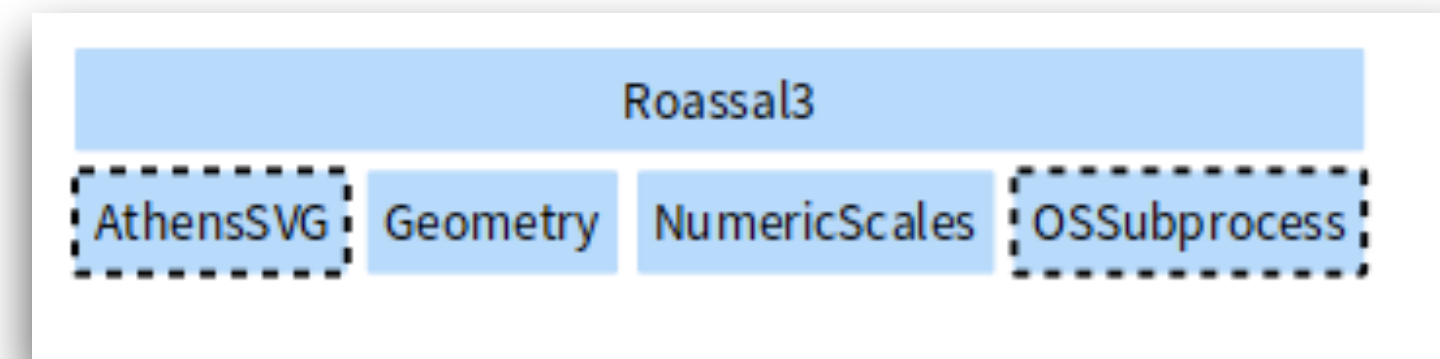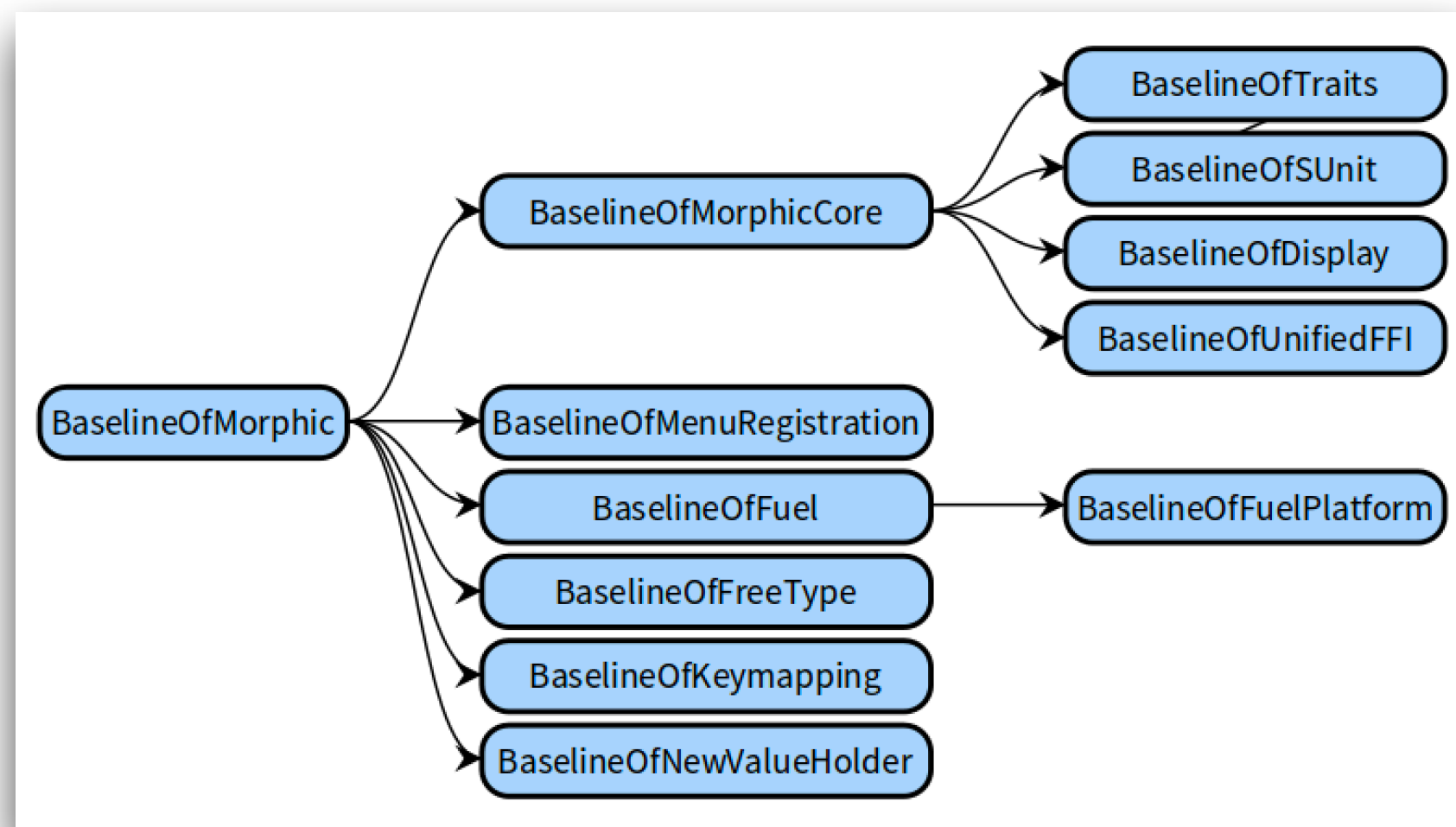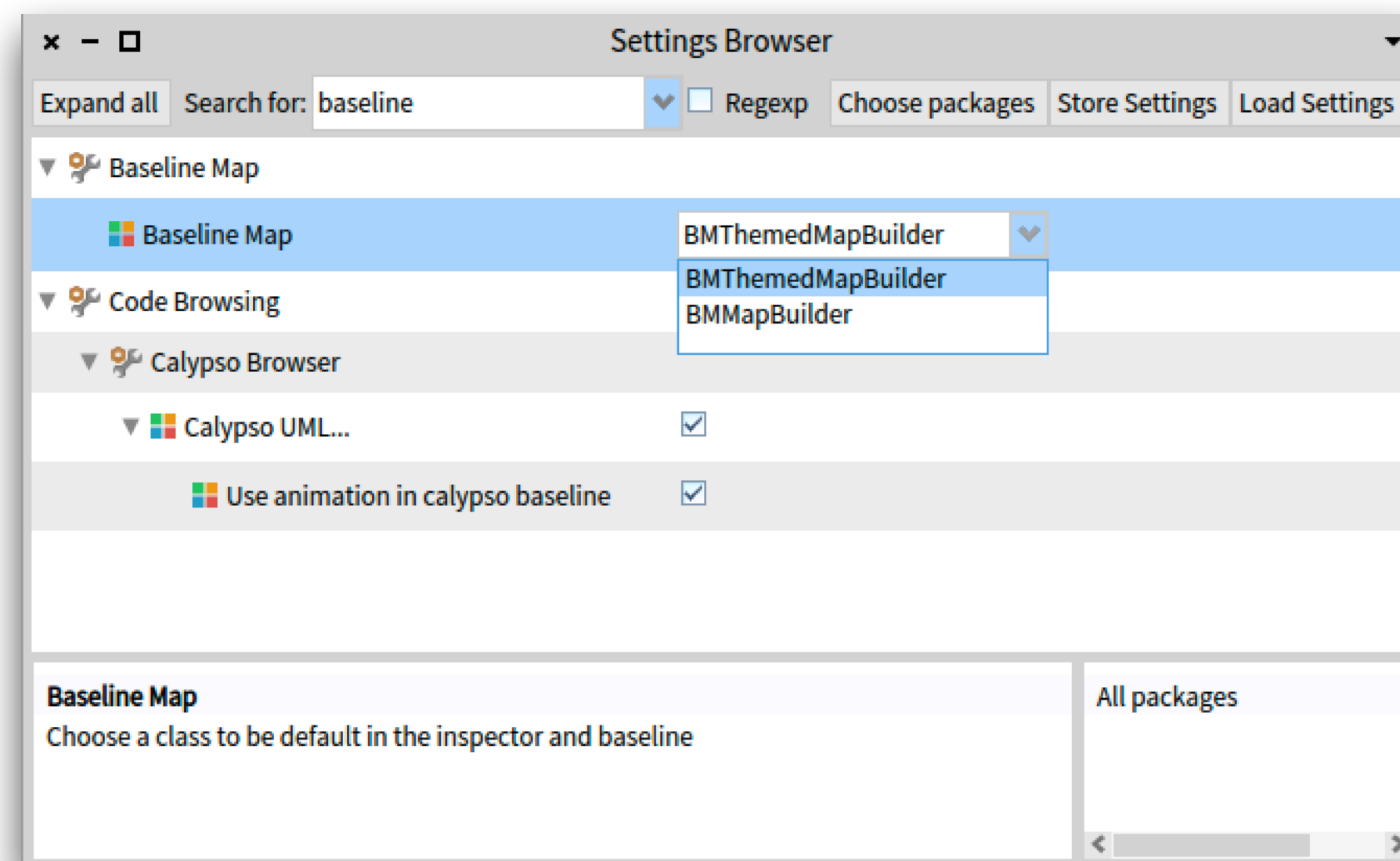## New baseline maps

- Because it is free to use, other users can create their own versions.

New maps

Collaboration with users

- With new versions of this visualization, this project can be extended to its limits.

- Now with a configuration class, users can use Settings Browser a tool created in Pharo to customize the application.

Demo

Reflexion

Communication

- After an idea, for a new project. Smalltalkers should create an entendible and extendible application, in this way your project can be used or extended by other developers.

- Rules, organization, patterns are very important.

- Documentation, tests, comments, correct names for classes, methods, attributes, examples, all this matters.

- Plugging a new tool in the ecosystem implies that you will collaborate with other people project's.

Thank you!