Program specialization and metaprogramming

Laboratory work No. 8

Analysis of cross-cutting concerns in Aspect Oriented Programming

1. Aims

To get acquainted with the advanced concepts of Aspect Oriented programming.

2. Tasks

Analyse crosscutting concerns of AspectJ applications.

3. Work

Run Eclipse IDE and select the Spacewar project from AspectJ examples.

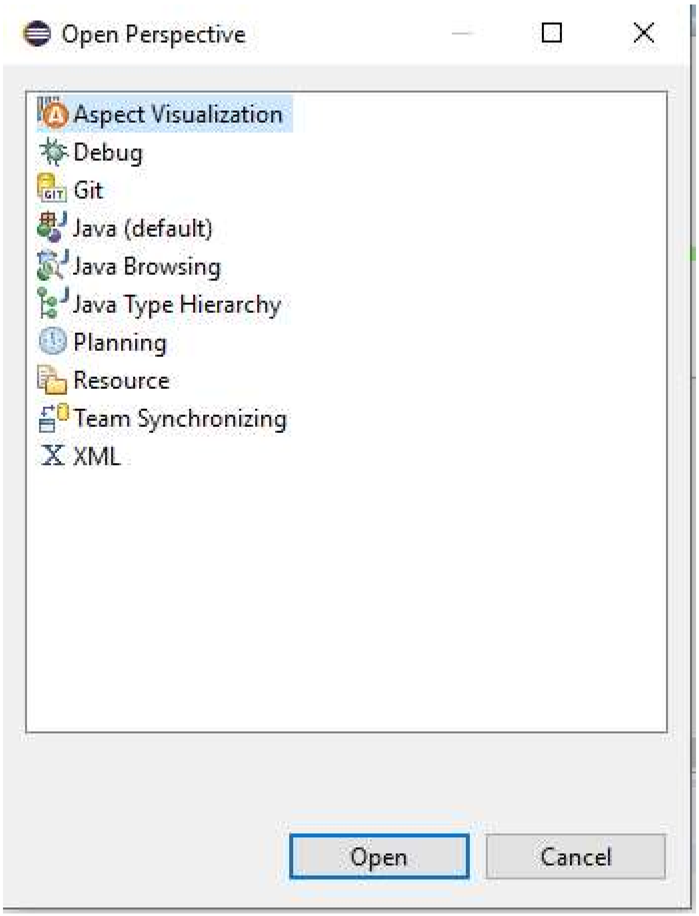
Inspect the classes and aspects of the Spacewar project.

What each class does? Which class has the main method?

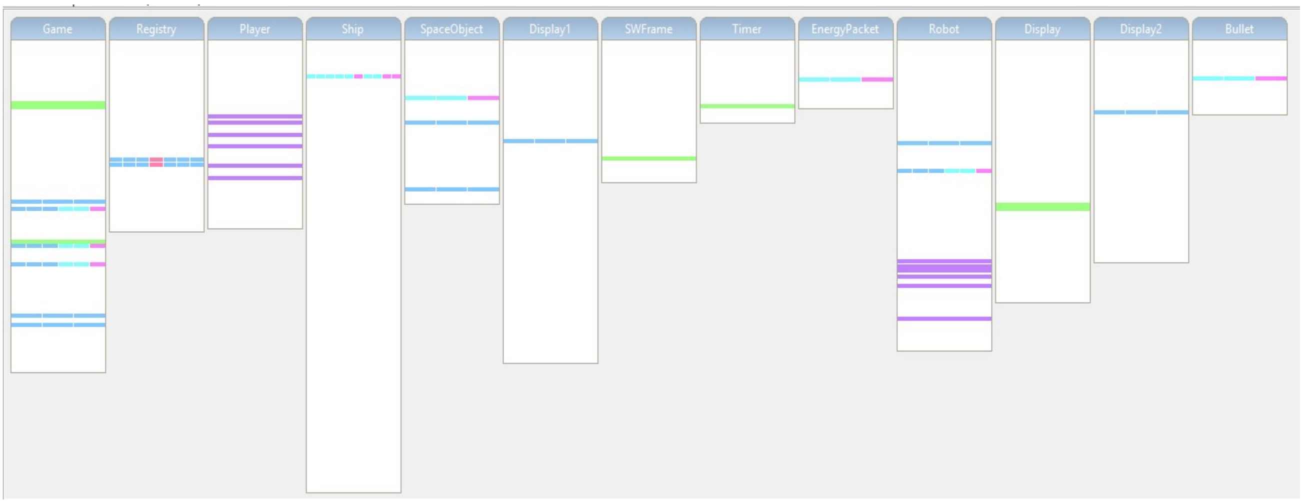
What each aspect does?

Launch Workbench and select Open Perspective at the top right end of the toolbar menu.

In the wizard window, select Aspect Visualization and press Open.



Select the spacewar package and observe the result in the Visualiser window.



The structure tree at the left can display different orderings and granularity for structure:

- The package hierarchy view shows the traditional hierarchy of package, class, and members.
- The inheritance view shows the hierarchy from topmost parent classes through subclasses to members.
- The crosscutting view shows the aspect members and the code they affect.

Additional buttons in the pane can be used to change the granularity and filter out items.

Whenever you select an item in the tree view, the source pane scrolls to that item

When working with aspects, it helps to be able to navigate between different program elements:

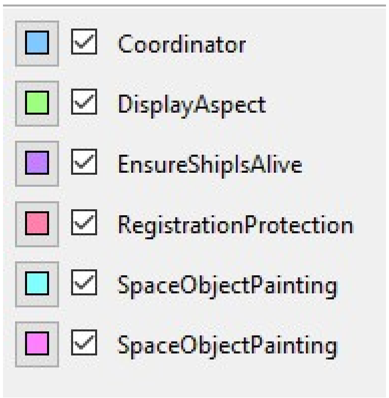
- When looking at a method, find the advice that affects it.
- When looking at a pointcut, find the advice that uses it.
- When looking at advice, find what it advises - e.g., method calls or executions, initializers, etc.
- When looking at a type, find any aspects that declare members or supertypes of the type, or vice-versa.

What do the columns and color rows in the Visualizer mean?

Inspect the cross-cutting concerns in the source code by pressing on the aspects represented by different color rows and observing the source code in the file view below.

Inspect the CrossReferences window at the bottom of the Workbench screen. Note the method and advice hierarchy it represents.

Perform selective visualization of concerns by selecting a specific aspect on the left part of the screen and observe the result.



🟦 ☑ Coordinator

🟩 ☑ DisplayAspect

🟪 ☑ EnsureShipIsAlive

🟥 ☑ RegistrationProtection

🟦 ☑ SpaceObjectPainting

🟪 ☑ SpaceObjectPainting

Which classes are affected (advised) by Coordinator aspect?

Which classes are affected (advised) by DisplayAspect aspect?

Which classes are affected (advised) by EnsureShipIsAlive aspect?

Which classes are affected (advised) by RegistrationProtection aspect?

Which classes are affected (advised) by SpaceObjectPainting aspect?

Modify the project by introducing new advice(s) and observe how the visualization of crosscutting concerns in the project has changed.

4. Self-study work

Explore other projects available in AspectJ examples using AspectJ visualization feature and discuss your observations and findings.

Discuss the complexity of source code in Aspect Oriented Programming versus Object Oriented Programming.

5. Report

Prepare a report of your work in this lab using a standard word processing application. The report is a single chapter of your final semester report. Final semester report will have to be uploaded to course Moodle page.

The content of report:

1. Title
2. Aims and tasks
3. Work done, described in steps and illustrated by screenshots and written or modified source code. Provide comments what you have done and what were the results (program outputs). Present answers to the questions given in the lab description.
4. Conclusions – what you have learned?