CHAPTER 1

Understanding Existing Code

In this exercise session, you will practice how to approach a system and understand it.

1.1 Setup

Download Pharo with Pillar from the Pillar website on Smalltalkhub http://www.smalltalkhub.com/#!/~Pier/Pillar.

To answer the following questions. We sugget that you use the different navigation tools of Pharo looking for class references (references), method implementations (implementers) and message sent (senders).

1.2 High-Level Structure

1. Draw a package diagram explaining which package depends on which one and the reason of this dependency
2. For the main packages, draw a class diagram with the main entities (classes, instance variables and methods)
3. How does test packages depend on other packages?

1.3 High-Level Behavior

1. Explain the different phases (and related classes) transforming a Pillar file into an HTML file.

2. Pillar has tests for abstractions (*i.e.*, abstract classes) such as `PRList` and `PRFormat`. These tests must hold for each implementation of these abstractions (*i.e.*, all subclasses). Explain how this works in practice.
3. What is responsible for parsing the command-line arguments (*e.g.*, ~~`baseDirectory and`~~ `to`) passed to the pillar command? How is the command-line interface implemented?

1.4 Studying Patterns

1. Find at least one instance of these design patterns:
   a) template method

   b) visitor

   c) composite

   d) interpreter

1.5 Visiting the Visitor

1. Draw a sequence diagram explaining how the super calls work in visiting methods (*e.g.*, `visitBoldFormat:` of `PRHTMLWriter`).
2. What is the role of the brush, canvas and writer classes in `Pillar-ExporterCore`?
3. What is the role of a transformer? What are the different types of transformers?

1.6 Technical Details

1. The methods in the `transformers` protocol of some subclasses of `PRPhase` are never called directly (*i.e.*, there are no *senders*). When are they executed? What does the `<pillarTransformer:*>` line at the beginning of each of these methods mean? Draw a sequence diagram.
2. How does Pillar maps a language name (*e.g.*, `"smalltalk"` and `"bash"`) to a particular subclass of `PRAbstractScriptLanguage`?
3. Pillar supports multiple ways of writing heading numbers (*e.g.*, ”See Section A.1.b”). Explain how the different numbering styles are implemented in Pillar and how they can be configured by the user.