Glamour packages - User study

Presentation

The goal of this study is to assess packages forming a application. You will assess packages through their classes and class references, analyzing what they use and how they are used, both internally and externally to the application.

This study is organized in three sections which require more and more in-depth look at packages and their classes. Each section implies that you play a different role when assessing the application, first as newcomer and potential client who wants to use the application, second as an architect who needs to assess the organization, third as a developer who performs maintenance. There are **11 questions** in this study, each question relating to a task.

You will perform the study on Glamour packages. Glamour is an engine for scripting browsers for any kind of models. If you are unfamiliar with Glamour, do not hesitate to test it before the study to get a basic understanding of Glamour capabilities. Information on usage and samples are available on: http://www.moosetechnology.org/tools/glamour

Tool used: System browser or OB System browser

Instructions

- Use only the tool indicated for the study. Do not use another browser/tool.
- Process questions in the given order (do not read questions in advance!)
- Please provide only accurate answers like the name of a class or a package, the association between two classes, the method which makes references.
- **Time yourself** for each question.
- Do not spend more than **20 minutes** on a question. If you reach this limit, write it down, stop the task, and proceed to the next question.
- You also have a time limit of **1h30** to answer the 11 questions so take care of your time.

A. Application assessment

As a potential client, you are assessing the package dependencies of the application. You want an idea about the size of the application and the kind of dependencies needed, especially if it involves new dependencies to be loaded with the application.

1) How big is the application?

Time taken: **2 min**

(a) In number of packages

11

(b) In number of classes (one of the following ranges):

[ ] <100; [x] 100-200; [ ] 200-300; [ ] > 300

158

2) What are the most important packages?

Time taken: **9 min**

(a) In terms of outgoing dependencies

Glamour-Tests

(b) In terms of incoming dependencies

Glamour-Browsers

(c) Overall, considering both outgoing and incoming dependencies

Glamour-Browsers

3) Focus on package Glamour-Morphic:

Time taken: **7 min**

(a) list all package dependencies which are external to Glamour.

#('Announcements' 'Balloon' 'Collections-Abstract' 'Collections-Arrayed' 'Collections-Sequenceable' 'Collections-Streams' 'Collections-Strings' 'Collections-Support' 'Collections-Text' 'Collections-Unordered' 'Compiler' 'DeprecatedPreferences' 'Exceptions' 'Files' 'FreeType' 'Graphics' 'Kernel' 'Magritte-Morph' 'Mondrian' 'Monticello' 'Morphic' 'Morphic-MorphTreeWidget' 'OB-Morphic' 'OB-Standard' 'Polymorph-Tools-Diff' 'Polymorph-Widgets' 'Refactoring-Core' 'SUnit' 'Shout' 'System-Platforms' 'Tools')

(b) in this list, please signal any external package which is not part of Pharo base (i.e., package must be loaded with Glamour).

Magritte-*, Mondrian, Morphic-MorphTreeWidget, OB-*, Refactoring

(c) are there other unexpected/unwanted package dependencies?

Other than what?

B. Application architecture assessment

As an architect, you now want to check the organization of your packages. You want your packages to have a good rationale for existence in the application. You want some parts of the application to be modular.

4) Please characterize **each** Glamour package as either:

Time taken: **2 min**

- a provider package for external clients (package with which external clients interact)

Glamour-Scripting, Glamour-Core, Glamour-Presentations, Glamour-Browsers

- an internal package (package which should not be accessed by external clients)

Glamour-Morphic, Glamour-Tests, Glamour-Helpers, Glamour-Tools, Glamour-Test-Morphic, Glamour-Announcements, Glamour-Examples

5) Are some Glamour packages optional/modular (package can be unloaded without impacting application core)?

Time taken: **5 min**

The Layers are as follows:

4. Glamour-Tests and Glamour-Example packages are optional, they test and exercise

3. Also Glamour-Morphic (Morphic UIs) and Glamour-Scripting (convenience) are optional

2. Glamour-Presentations, Glamour-Browsers and Glamour-Core form the core of the system

1. Glamour-Helpers and Glamour-Announcements could be used independently of the rest

6) What are the important classes (consider incoming, outgoing, inheritance dependencies) in Glamour-Core? If possible, explain their roles.

Time taken: **2 min**

GLMBrowser (abstract superclass of all browsers), GLMPresentation (abstract superclass of all presentation types), GLMRenderer (abstract builder class for browsers)

7) Are there direct cyclic dependencies from Glamour-Core to another package?
Time taken: **0 min**

No

C. Detailed assessment

As a developer, you want a detailed comprehension and assessment of dependencies between classes and packages and optionally to refactor such dependencies, assessing impact of change.

First give a precise answer then provide your explanation.

8) What are the most cohesive packages of the application?
Time taken: **1 min**
Glamour-Core (56 internal dependencies)
Glamour-Morphic (47 internal dependencies)

9) There is a dependency to DeprecatedPreferences in Glamour-Morphic. Can you detect the faulty class? Explain the dependency: do you see an easy way to solve it?
Time taken: **2 min**
Glamour-Morphic depends on DeprecatedPreferences:
GLMMorphicRenderer>>#treeMorphFor:and: references Preferences

10) Can you explain the organization of Glamour-Morphic and its relationship with other packages?
Time taken: **3 min**
Glamour-Morphic builds Morphic UIs of browsers modeled using the classes in Glamour-Browsers and Glamour-Core; Glamour-Morphic is essentially a Visitor (GLMMorphicRenderer) and a few morphic classes to perform the model-transformation to.

11) Multiple packages of Glamour have dependencies to external library Mondrian. List such packages. Could you extract this dependency and make it optional (you can propose a solution)?
Time taken: **2 min**
#('Glamour-Scripting' 'Glamour-Morphic' 'Glamour-Presentations' 'Glamour-Tests' 'Glamour-Examples' 'Glamour-Test-Morphic')
-> move to external package ‘Glamour-Mondrian’

D. Personal remarks

You can provide any additional remarks about the study itself, the tasks, the tool used.

Although it was requested to only use System Browser and OmniBrowser, I also used some reflective workspace scripts to answer the questions. I think it is totally pointless to browse code and to manually count (not-visible) dependencies with a tool made for something different; I don’t have that much time :-)