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: Package blueprint browser

Instructions

- Use only the tool indicated for the study. Do not use another browser/tool.
- Browse the documentation before performing the study:
    - **PackageBlueprintsPrinciples.pdf** explains the basics of package blueprints
    - **packageblueprints.mov** shows interactions with the package blueprint browser
    - a draft version of the journal paper describing package blueprints is provided
- 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 with 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: 00:37.8

(a) In number of packages

11

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

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

2) What are the most important packages?
Time taken: 02:54.7
(Glamour-xxxx)
(a) In terms of outgoing dependencies
Tests, Morphic, Examples, Core
(b) In terms of incoming dependencies
Core, Browsers, Presentations, Helpers
(c) Overall, considering both outgoing and incoming dependencies
Core, Presentations, Browsers, Helpers

3) Focus on package Glamour-Morphic:
Time taken: 03:13.2
(a) list all package dependencies which are external to Glamour.
Morphic, Polymorph-Widgets, Mondrian, Morphic-MorphTreeWidget, Balloon, Shout, Margritte-Morph, Graphics, Collections-Strings, Collections-Sequenceable, DeprecatedPreferences, Collections-Arrayed, FreeType, Polymorphic-Tools-Diff, Collections-Unordered
(b) in this list, please signal any external package which is not part of Pharo base (i.e., package must be loaded with Glamour).
Mondrian, Shout, Margritte-Morph

(c) are there other unexpected/unwanted package dependencies?
(this is not clear) but i found weird to find DeprecatedPreferences

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: 03:01.1
- a provider package for external clients (package with which external clients interact)
Core, Browsers, Presentations, Helpers, Announcements, Tools.
Tests, Morphic not for external clients outside Glamour
Scripting does not appear in the incoming blueprint. It should be a provider package
- an internal package (package which should not be accessed by external clients)
Test-Morphic, Examples

5) Are some Glamour packages optional/modular (package can be unloaded without impacting application core)?
Time taken: 01:36.8
Test-Morphic, Examples, Tests, Morphic, Tools

6) What are the important classes (consider incoming, outgoing, inheritance dependencies) in Glamour-Core? If possible, explain their roles.
Time taken: 08:45.6
GLMTransmission
GLMBrowser

GLMPresentation
GLMPane
GLMPanePort
GLMPort
GLMCompositePresentation
GLMLoggedObject

7) Are there direct cyclic dependencies from Glamour-Core to another package?
Time taken: 01.32.7
Core (i) - Browsers (i)
Core (i) - Presentations (i)
Core (o) - Helper (i)
Core (o) - Announcements (i)

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: 01:45.1

Test, Core, Morphic, Presentations, Browsers. Considering the size of top surfaces
Core, Morphic, Browsers. Considering internal cohesion

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: 00:58.1
GLMMorphicRenderer -> Preferences
treeMorphFor:and: invokes Preferences standardMenuFont when creating treeColumns

A class method should be defined to retrieve the needed font for Glamour

10) Can you explain the organization of Glamour-Morphic and its relationship with other packages?
Time taken: 04:57.6
By inspecting inheritance blueprint the organization as layers:
styles (windows, theme, tabs)
morphic renderer
shout morph
schollers
morphic model + actions/events
morphic + list model
tree behavior

This package is centralized by its GLMRenderer class. This class almost refers to all of the internal classes, and to the majority of external packages (including 3 of 4 of the Glamour packages: Browsers, Announcements and Core)

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: 05:56.8

Tests, Morphic, Examples, Test-Morphic, Presentations
There is a small set of classes of Mondrian which is referred by Glamour (ViewRenderer, 2 shapes, Canvas, 4 events, 2 layouts). In particular Presentations (part of the core) is only referring to ViewRenderer for the mondrian presentation, few classes by Morphic and the rest by Examples and Test(s). The mondrian presentation (+ related behavior) could be defined in a separated package for external presentations and not as part of the core.

D. Personal remarks

You can provide any additional remarks about the study itself, the tasks, the tool used.
I want to mention that I had not previous knowledge about Package blueprints. But instead I have a good knowledge about Glamour, how its classes are distributed and their purposes. So, probably this impacted in the evaluation.

The study itself is well done, and the tasks are concrete. For me in particular, knowing Glamour made easy to find many of the answers, without really knowing all the potential of Package blueprint.. But in other case, I would probably needed a lot more of time... For understanding your tool I needed at least an hour to go quickly through the documentation. Based on this, the study requires more time just to prepare the user and I don’t know if was a good idea to evaluate it on an application that I know.

E. Personal evaluation of Package blueprints

1	2	3	4	5
Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree

Question	1	2	3	4	5
Does package blueprint help you to understand dependencies between packages?					x
Would you use package blueprint when you need to understand packages?					x
Did the outgoing view help you?					x
Did the incoming view help you?					x
Did the inheritance view help you?				x	
Was package blueprint useful to get an impression of the most used classes in a package?					x
Was package blueprint useful to get an impression of the most referencing classes in a package?				x	

Was package blueprint useful to get an impression of package cohesion?				x	