Name: *Lastname, Firstname*

*GitHub Username:*

Student ID:

Project:

Date: *mm/dd/yyyy*

*Everything in italic should be changed as appropriate by you and should not be italic when submitted. Also remember code is not changed until the Actualization stage, so don’t put “I changed” or similar until section 4 of the report.*

*Comic Template (#108)*

## 1. Change Request

As a user, I would like to create a comic template so it's possible to create own comics. This enhancement will give an opportunity to create own comics with a possibility to choose one of many ready templates. I also would like to create an option to add quotes, change font and size in these quotes. As additional I want to be able to change the background of each figure of template.

*In this section, describe the change that you were requested to do. Give any relevant background information or any essential details.*

## 2. Concept Location

*Explain the methodology that you have used to locate each concept that was part of your change request. Using Table 1, list all the files in the order that you have visited them (2nd column). Explain how you have found each file (3rd column). You can simply read the source code, but we encourage you to use the features provided by Netbeans and Featureous: “Quick Find”, “Find Symbol”, “Go To Definition”, “Call Browser”, “Find all References”, ”Class View”, “Insert Breakpoint”, or any other software tools that you want to use.*

*Furtmore, Featureous Inspector is a good starting point for localizing classes based on their features.*

*In the fourth column, mention if the class is related to the concept. Use one of the following terms:*

* *Use* ***“Unchanged”*** *if the class has no relation to the concept but you have visited it.*
* *Use* ***“Propagating”*** *if you read the source code of the class and it guided you to the location of the concept, but you will not change it.*
* *Use* ***“Changed”*** *if the class will be changed.*

*In the fifth column, write what you have learned about the class.*

**Table 1. The list of all the classes visited during concept location.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **File name** | **Tool used** | **Located?** | **Comments** |
|  | ToolsToolBar.java | Featureous |  |  |
|  | ButtonFactory.java | Find in project |  |  |
|  | AbstractToolBar.java | Featureous |  |  |
|  | SVGDrawingPanel.java | Featureous |  |  |
|  |  |  |  |  |

tb.setBackground(Color.RED); ButtonFactory.java

tb er Tools panel, hvor der skal tilføjes et ekstra knap.

toolsPane.setBackground(Color.RED);

SVGDrawingPanel.java

(hele panel I bottom af program)

billeder:

/Users/Karol/NetBeansProjects/SB5-MAI-E17/JHotDraw/target/classes/org/jhotdraw/samples/svg/action

*Use Featureous Feature call-tree to provide a tree-based visualization of the runtime call graphs of methods implementing the features of your change request. This view provides an execution- based alternative to the hierarchical fashion of browsing features supported of the mentioned feature inspector view.*

## 3. Impact Analysis

*Using table 2 list the classes that you visited after you located the concept. On the first row, include the class where you have located the concept, i.e. the class that will be changed (2nd column). Explain how you have found each of the classes, i.e. which tools have you used (3rd column). Do you think that the class will be impacted by the change? (4th column). Write short comments explaining what have you learned about each class. What other tools you would like to have in your IDE so that impact analysis would be faster?*

**Table 2. The list of all the classes visited during impact analysis.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Class name** | **Tool used** | **Impacted?** | **Comments** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

*Use Featureous Feature-Code Characterization View to illustrate Scattering and Tangling of each feature that is involved in your Change Request.*

*Use Featureous Feature Relations Characterization view to relate your change request features to each other with respect to how they depend on the objects created by other features and how they exchange data by sharing these objects with one another.*

*Use Feature-code correlation graph and feature-code correlation grid to illustrate detailed investigations of the correspondences between source code units and the features of your change request.*

## 4. Refactoring

*Describe the code smell that triggered your refactoring, see Chapter 4 in* [*[Ker05]*](#kix.mbwfsntqmgyh)*.*

*Describe how you have refactored to, towards or away from patterns.*

*Why did you do the refactoring?*

*Describe the implementation details of the refactorings. Remember there is not one way to implement a pattern.*

*Which of the refactorings from* [*[Ker05]*](#kix.mbwfsntqmgyh) *did you apply and what was the reasoning behind it?*

## 5. Implementation

*Explain where and why you made changes in the source code. That is, explain the difference between the actual change set compared to the estimated impacted set from the impact analysis. Which classes or methods you created, or changed. Furthermore, describe used design patterns and reflections about improved design to improve future software maintenance.*

## 6. Verification *Document how your have verified your implemented change.*

*Document the result of your* ***acceptance*** *test that test your new or improved feature from your change request.*

*At class level document* ***unit test*** *of important business functionality.*

## 7. Conclusion

*Explain your experience with creating the final baseline of JHotDraw.*

*How did you manage to merge in your changes to the baseline ?*

*How was the system tested?*

*Describe your reflection on what went well and what went not so well.*

*Did you have to cut the scope of your change request and did you have to put some issues back into the backlog.*

*What can be done to avoid future problems, i.e. what have you learned from the iteration.*

## 8. Source Code

*Submit all your code changes to your local Git repository and finally the remote GitHub repository.**You can use links to GitHub if you want to reference source code.*

## 9. Planning

*Release backlog table after each day of work, see Table 12.8 page 194 in* [*[Raj13]*](#74rfajcjyvsx)*.*

## References

*Include any sources that you cited or used information from.*

[Ker05] Joshua Kerievsky. Refactoring to patterns. Addison-Wesley, 2005.

[Raj13]Vaclav Rajlich. Software Engineering: The Current Practice, volume 38. ACM, New York, NY, USA, November 2013.

## 