1 Concept Location

1.1 Methodology

The location of the classes that I identified was based on the following tools, which I used to locate the different classes and the different methods that I found was relevant for the feature I had chosen *Tools Palette*.

- Different search methods such as:
    - *Find all references.*
    - *Go to definition.*
    - *Quick search.*
    - *Global search.*
    - *Keyword search*
- Tree scaling both up and down with *Extension reference.*
- Removing code to see what functionality it would affect, thereby better understand what the different pieces of code did what and affects.

1.2 Table Content Overview

The table below provides an easy overview of the different tools and processes I used to locate the different classes that I found relevant for my chosen feature.

#	Domain classes	Tool(s) used	Comments
1	*AbstractToolbar*	*Quick Search* *Find all references* *Code removal*	I started by looking at the different abstract classes for the whole project. Here I found the *AbstractToolbar* class which looked like the right abstract class I was looking for when my features name is *Tool Palette*. I then tested with *code removal* to see what it would impact in the toolbar, but I just not see any changes to the behavior of the program itself when running, so I started looking at what the *AbstractToolbar* was extended from.
2	*JDisclosureToolbar*	*Code removal* *Extension reference* *Go to definition*	When I look at what *AbstractToolbar* was extended from, I found the abstract class named *JDisclosureToolbar,* I again tried *code removal*, this time giving my first result. The abstract class *JDisclosureToolbar* is responsible for the show/hide feature of the tool palette, which I needed for my user story *Display.*
3	*ToolsToolbar*	*Code removal* *Extension reference*	I then went back down the reference tree to see where in what class it would end. I ended up in the class *ToolsToolbar.* Here again with *code removal* I tried to see what the class was responsible for. I found that it was not the full toolbar as my first thought had been, but it was only a part of the whole tool palette.
4	*PaletteToolbarUI*	*Code removal* *Keyword Search*	After I hit a dead end with the *Extension reference* tool method, I tried to do a global search on different keywords such as *Tool, UI, Palette, Bar, ToolBar* and other keywords that

			could be assimilated with my feature. With this tool method I found the *PaletteToolbarUI* class which contained *handlers.* I tried to remove some of these handlers to see what it would affect.

2 Impact Analysis

2.1 Brief introduction

The impact analysis is used to understand the implications of changing a specific feature within the JHotDraw project. The project itself will be receive the different feature changes at different times, as the development team, consisting of 5 members, is working on different features of the application in their own pace and some members might be further ahead than other, at any giving time. After inserting the feature entry points into the JHotDraw project, I was able to get an output of different relevant figures: *Feature-code Characterization, Feature-code Correlation Grid* and *Feature-Package Correlation Graph.*
The feature entry points I used in this project are the following:

- *Tools-display*
- *Drag-drop*
  - *Pressed*
  - *Dragged*
  - *Released*

The *Tools-display* was as stated in the *Concept Location* chapter a class that references the *JDisclosureToolbar* class. The *Tools-display* is an handler call-back that has been add to a button, when pressed it will change the visibility of the chosen toolbar section from visible to hidden or vice versa.

The *Drag-drop* which consist of *Pressed, Dragged* and *Released.* Are handlers that are connected with the *PaletteToolbarUI.* They are activated in the following order:

- *Pressed* when a tool is pressed on with the click of a mouse.
- *Dragged* when a tool has been pressed and the mouse moves while the button is still being hold down by the user
- *Released* when the user releases the pressed mouse button again.