Exploring some OOP code, and basic UML diagrams

To load the project in to IntelliJ:

1. Use the File -> New -> Project From Version Control -> Git menu.
2. Paste in the following URL: https://github.com/sdp−resources/chat.git
3. Click "Clone". You should now be looking at the project.
4. Select "VCS -> Git -> Branches", then pick the origin/simpleClassStructure option, and then "Checkout as..", then hit OK.
5. At the bottom of the screen there should be a "Maven projects need to be imported" pop-up. Click "Enable Auto-import".
6. Navigate into the chat/source/main/java folder in the project pane on the left. You should see 4 classes.

Task 1

Your first task in this activity is to produce a UML diagram describing these classes and their relations.

- Start by drawing a "square/card" for each class with the name near the top (leave room in it for variables, methods, and comments).
- Add into each square the instance and static variables of the class. Mark them with plus sign + (private) or minus sign - (public) on their left, and specify the type after a semicolon on their right. Underline the name if the variable is static.
- Add into each square the **public** or **package-private** methods of each class, including their return type and any arguments they take, and their type.
- If a class creates instances of another class, draw an dashed arrow from the former to the latter with the word <creates> next to the arrow.
- If a class *knows* of another class ("depends on it"), for example if it has an instance variable of that class or if one of its methods takes an argument of that class, draw a solid arrow from the former to the latter. If there is already a "creates" arrow, then *only* add another (separate arrow) if there is a different usage of that class. To find all places where a class is being referred to, place your cursor on the class name in its file, and choose the "Navigate -> Declaration" menu option.
- List in the comment section the kinds of "responsibilities" that this class has. What does it know how to do?
- Our classes use other classes provided by other packages. List all these dependencies in some separate section.

- A class' **fan-in** is the number of our classes that depend on it. Its **fan-out** is the number of our classes that it depends on. Compute the fan-ins and fan-outs of our four classes.

Task 2

Now in a new area on the board, do the same work for a slightly changed version of the code. To see that version, go to the "VCS -> Git -> Branches" menu, and pick the origin/basicClassStructure option and then "Checkout as..". Then compare and contrast the two different structures. What are their differences? What do you think of that?

Task 3

Consider the following: We want to expand our application so that there are multiple kinds of message-like notifications, instead of just the current ChatMessages. For example we might want a notification to show up when someone leaves or when someone joins, or we might want "video messages" rather than text messages. How might our application change to accomodate that, and which classes would be affected by this change?