1. One of your tasks was to make the Speaker, VibrationMotor, and Console classes Singletons. Why did we do this (other than to exercise the Singleton pattern)? Asked another way, what could potentially happen if we did *not* implement these as Singletons?

   In general, it holds an instantiated object and depending on how its created, it makes sure that it is thread safe and only does one initialization whether its lazy or eager. They can implement interfaces and be passed as parameters. If we do not use singletons, multiple instances can be created and when run, will provide errors, bugs, and misinformation due to it not being thread safe.

2. OK. Let's say we decide we want to make the Chat application a little more "context-aware" and we want to implement a functionality that makes it possible for a user's preferences to prevent "notifications" (i.e., beeps, vibrations, or texts) while the user is in an important meeting. Describe (perhaps with code snippets) what changes you would make and where you would make them. **Do not implement this solution, just explain your design.**

   From what I learned and what I think would be the best way to implement this, in the ChatListener, I would add a condition where it checks notifications if it is on or off. You can also create a singleton and command in which it to check if notifications are on and off, and pass this in order to verify if the user wants to be notified. This allows for the most up to date response when it comes to checking whether it is on or off.

3. Inside the run() method of ChatListener, you have to fill in the creation of the commands and do something with them that ultimately results in their execution. Why do you not just call execute() on the command that you just created? Hint: consider your answer for the previous question.

   In order to be thread safe, it is best to use the queue and with the help of the singletons, you can receive the notifications in order as well as the order in which they came. This allows for smooth run of the program and minimizing the bugs. If not, then the program would display incorrect information or have errors when it reads whether the message is a beep, vibration, or text.

4. In *your own words*, what does it mean when I say (in ChatClient) that the queue is "thread-safe"?

   From my understanding, it is thread safe because if nothing is in the queue, then the thread can wait until the queue has at least one object in which it can take, if not, it will just wait until something is added into the queue. Basically, it block threads if it is trying to modify the queue.