Practice 9

Deadline: 2 weeks from now. Should be checked onsite (during labs).

Task 1

In this task, we'll implement a simple Pokémon query service in which multiple clients could simultaneously query a server to ask for basic information about a given pokémon.

Server

The server should keep listening for clients. Once the server establishs a connection with a client, it could get the name of a pokémon sent by the client. Then, the server should make a REST request to PokeAPI to get the basic information about this pokémon.

PokeAPI (https://pokeapi.co/) is a free RESTful service for querying Pokémon data. For instance, we could query all data w.r.t a specific pokémon using its Pokemon endpoint:

```
GET https://pokeapi.co/api/v2/pokemon/{id or name}/
```

If the client sent "pikachu", the server should:

- Query the data for "pikachu" using the Pokemon endpoint.
- Get the JSON response.
- Retrieve pikachu's height, weight, and a list of abilities by analyzing the JSON data (you may use any JSON APIs or libraries for parsing).
- Send the information back to the client.

Client

After connecting to the server, the client could keep typing names of pokémon, and print out the information about this pokémon sent from the server. If the client sent "QUIT", the connection will be closed.

Demo

You should start the server, and start at least two clients at the same time.

Server

```
Waiting for clients to connect...

Client connected.
Info of pikachu sent

Client connected.
Info of arbok sent
Info of psyduck sent
Client quits.
```

```
Client quits.
```

Client 1

```
Enter the pokemon name: pikachu
Name: pikachu
Height: 4
Weight: 60
Abilities: [static, lightning-rod]

Enter the pokemon name: psyduck
Name: psyduck
Height: 8
Weight: 196
Abilities: [damp, cloud-nine, swift-swim]

Enter the pokemon name: QUIT
```

Client 2

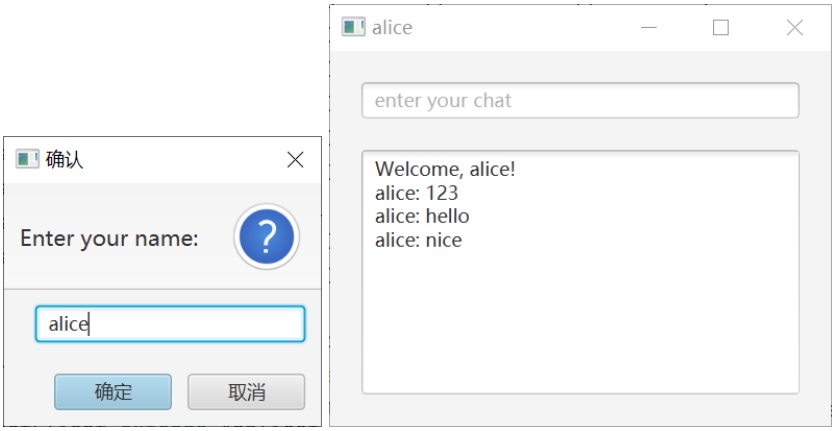
```
Enter the pokemon name: arbok
Name: arbok
Height: 35
Weight: 650
Abilities: [intimidate, shed-skin, unnerve]

Enter the pokemon name: QUIT
```

Task 2

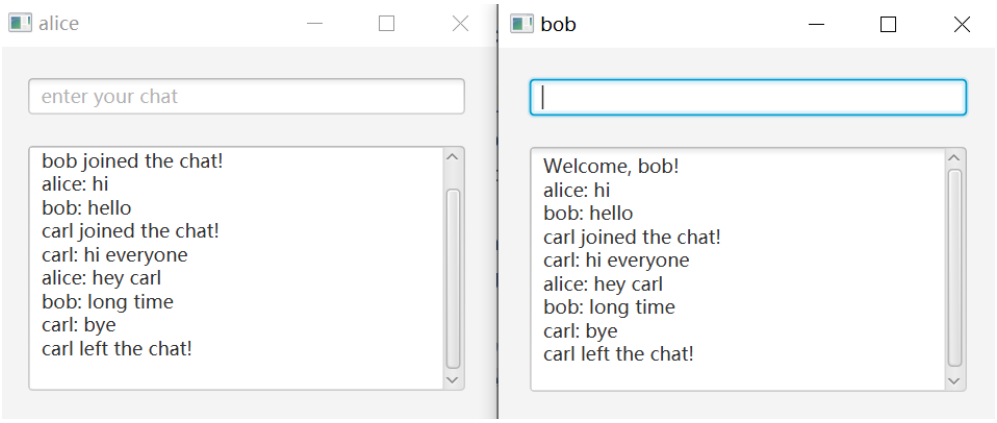
In this task, we'll turn the broadcast console program from the tutorial to a GUI program with JavaFX.

To get you started, we provide an incomplete implementation that allows user to input name, type anything in the text field, press Enter, then display the input in the text area. Download the `BroadcastFX.zip` from BB to access the code.



Note that, the controller class can define an `initialize()` method, which is called after all `@FXML` annotated fields have been injected. Basically, in the controller class, the constructor is called first, then any `@FXML` annotated fields are populated, then `initialize()` is called. So, the constructor does not have access to `@FXML` fields, while `initialize()` does have access to them.

Your task is to turn this code into a C/S broadcast GUI program as in the tutorial. Multiple clients could connect to the server, and send messages to the server, which will broadcast the messages to all clients.



Tips

- You may reuse the `ChatServer` from our tutorial, which is also available on BB.
- You may update `ChatClientController` to add socket communication and multi-threading.
- You may use `Platform.runLater()` to update the GUI from a non-GUI thread.
- You may use `stage.setOnCloseRequest(...)` to handle the event when clients click X to close the window.

Evaluation

The practice will be checked by teachers or SAs. What will be tested:

1. That you understand every line of your own code, not just copy from somewhere
2. That your program compiles correctly (javac)
3. Correctness of the program logic
4. That the result is obtained in a reasonable time

Late submissions after the deadline will incur a 20% penalty, meaning that you can only get 80% of this practice's score.