Class 17: Callbacks & Graphical User Interfaces

6.102 — Software Construction
Spring 2024

Warmup

Exercise:  yellkey.com/**TODO**

Nanoquiz:  yellkey.com/**TODO**

Open `warmup.ts`

`npm run warmup` to see it in action

Refactor to convert `await` into `then()`

- start with `getBalance()`
  - and run it again
- then convert `main()`

Preparing to edit your GUI

- `npm run watch` so that the TS code is being constantly recompiled
- open `chat.html` in your web browser
- open the browser's JavaScript console so you can see errors
  - right-click on the page / Inspect / Console tab

Back in VS Code, find TODO1 at the top of `controller.ts`

- edit the code to uncomment the print statement and save the file
- watch it be recompiled in the VS Code terminal ("[watch] build finished")

Back in your browser:

- reload the page
- look for the print statement to appear in the browser's JavaScript console

Controller

In `controller.ts`:

- TODO2: the Go button should display what the user typed using `display()`
  - hint: textboxes have `value` property
- TODO3: Go should also clear the user's textbox so they can easily type something new
  - hint: textbox `value` property is assignable
- TODO4: the Clear button should clear the chat textarea
- TODO5: pressing Enter in the userTextbox should do the same thing as clicking Go
  - hint: `KeyboardEvent` has a `key` property

Model

In `model.ts`, first study the code:

- what is `Chat`?
- when does it call its listeners?
- what argument does the listener take?

Model

In `model.ts`, first study the code:

- what is `Chat`?
- when does it call its listeners?
- what argument does the listener take?

Then, update `controller.ts`:

- TODO6
  - Go button should now send messages to the `Chat` model, not call `display()` directly
  - listen to the Chat model and `display()` on all messages it gets

Model

yellkey.com/**TODO**

In `model.ts`, first study the code:
• what is `Chat`? • when does it call its listeners? • what argument does the listener take?

Then, update `controller.ts`:

- TODO6
  - Go button should now send messages to the `Chat` model, not call `display()` directly
  - listen to the Chat model and `display()` on all messages it gets

- TODO7: implement `inviteBot()` to attach a bot to the `Chat` model
  - write a listener that sends the bot every message and puts its reply back on the chat
  - `WishBot` is already invited to the chat; look at its code in `bots.ts`
  - try typing a message that `WishBot` will reply to, e.g. `I wish I had an apple`

Model

yellkey.com/**TODO**

In `model.ts`, first study the code:
• what is `Chat`? • when does it call its listeners? • what argument does the listener take?

Then, update `controller.ts`:

- TODO6
  - Go button should now send messages to the `Chat` model, not call `display()` directly
  - listen to the Chat model and `display()` on all messages it gets

- TODO7: implement `inviteBot()` to attach a bot to the `Chat` model
  - write a listener that sends the bot every message and puts its reply back on the chat
  - `WishBot` is already invited to the chat; look at its code in `bots.ts`
  - try typing a message that `WishBot` will reply to, e.g. `I wish I had an apple`

- TODO8: the `Elmo` bot
  - in `bots.ts`, finish `ElmoBot`
  - try typing a message that both `WishBot` and `ElmoBot` will reply to, e.g. `I wish I had a banana`