Lab 10: Querying Web API with Async Calls

Submission deadline: 2359, Sunday, November 12, 2017.

Setup

The skeleton code from Lab 10 is available on `cs2030-i` under the directory `~cs2030/lab10`. The code is functional, but uses synchronous calls to make query to a Web API for bus information.

Finding Bus Routes

The given program, `LabTen`, takes in a source and destination bus stop, for instance:

```
java LabTen 17009 18331
```

and output possible routes to go from the source bus stop to the destination:

```
From 17009 Clementi Int to 18331 Kent Ridge Stn Exit A/NUH:
take 96, transfer at 16199 NUS Fac Of Design & Env to 95
take 196, transfer at 18149 Opp Anglo-Chinese JC to 95
take 196, transfer at 18141 Aft Anglo-Chinese JC to 95
take 96, transfer at 16179 Opp Yusof Ishak Hse to 95
take one of [14, 166], transfer at 18121 Opp Ayer Rajah Ind Est
take 96, transfer at 16189 Computer Ctr to 95
take one of [14, 166], transfer at 18129 Ayer Rajah Ind Est to 9
take one of [165, 7], transfer at 11261 Holland Village to 95
take one of [196, 147], transfer at 11181 Opp Blk 43 to 95
take one of [196, 147], transfer at 11189 Blk 43 to 95
take 196, transfer at 11361 Buona Vista Stn Exit C to 95
take 196, transfer at 11369 Buona Vista Stn Exit D to 95
take 14, alight at 18101 PSB Science Pk Bldg and walk
take 14, alight at 18109 Opp PSB Science Pk Bldg and walk
take 166, alight at 18101 PSB Science Pk Bldg and walk
take 166, alight at 18109 Opp PSB Science Pk Bldg and walk
```

There are three types of routes being considered:

- A direct route (none in the example above), where we can take a bus directly from the source to the destination.
- A direct route with walking, where we can take a bus directly from the source to a stop near the destination, and then walk to the destination (`take ... alight at ... and walk`);
- A route with one transfer, where we can take a bus from the source, a change to another bus at an intermediate stop (`take ... transfer at ... to ...`).

There are other possibilities that we do not consider in this lab (such as walking to a nearby bus stop from the source and take a direct bus to destination, or walk from one intermediate bus stop to another to transfer).

The program uses the same data files as Lab 8, but only the list of bus stop IDs and its human friendly names are available to you (in the file `bus-stops.csv`). The rest of the information is available via a Web API call:

- `https://cs2030-bus-api.herokuapp.com/bus_services/<bus service id>` returns a list of bus stop ids served by a given bus. Example [https://cs2030-bus-api.herokuapp.com/bus_services/95]
- `https://cs2030-bus-api.herokuapp.com/bus_stops/<bus stop id>` returns a list of bus service ids that serves a given bus stop. Example [https://cs2030-bus-api.herokuapp.com/bus_stops/18331]
- `https://cs2030-bus-api.herokuapp.com/bus_stops/<bus stop id>/nearby` returns a list of bus stops id within 5-minutes walk (400 meters) of a given bus stop. Example [https://cs2030-bus-api.herokuapp.com/bus_stops/18331/nearby]

The basic code to query the Web API is provided in the skeleton code. See `BusStop.java` and `BusService.java` .

Since we do not have any information about the actual bus schedule and bus direction, we do not consider them in producing the output. So, the routes produced may not match the actual bus route in real world -- do not use this lab to plan you travel in real life :)

Task

The goal of this lab is to change the code from synchronous to asynchronous, using `CompletableFuture`. At the very least, all methods that involve querying the Web API should be asynchronous, since they are slow. The search for routes can be made faster through asynchronous calls as well.

My suggestion on how to approach this lab:

- Read the code given in `BusStop.java`, `BusService.java`, and `BusSg.java` to understand the algorithms used to search for the routes.
- Convert the code one-by-one to asynchronous calls, using `CompletableFuture`'s `runAsync` or `supplyAsync`. If a method returns `T`, you may want to return `CompletableFuture<T>`.
- You can change how the search code is written in `BusSg.java` (e.g., break it into smaller methods so that asynchronous calls is easier to read).
- You should handle exception appropriately, catching them and replacing the return values with empty list / set (for instance, if the Web API fails).
- Thread-safe collections are used liberally in the given code (`CopyOnWriteArrayList` and `ConcurrentHashMap`). If you need to add a new collection in part of the code that is access concurrently, make sure you use a collection from `java.util.concurrent` package.
- Make sure that the program waits for all the asynchronous calls to complete before the program exits, using `CompletableFuture.allOf(...).join()`.

NOTE: The sequence of output is not important for correctness in this lab.

As usual, you should also:

- follows the CS2030 Coding Style [../style/index.html]
- clearly documented with `javadoc` [../javadoc/index.html]

You need an Internet connection to run LabTen.

Grading

This lab contributes another 4 marks to your final grade (100 marks).

You get:

- 1 mark for correctly and asynchronously finding the direct bus route (`findDirectBusServicesBetween`)
- 1 mark for correctly and asynchronously finding the direct bus route + walking (`findBusServicesToNearby`)
- 2 marks for correctly and asynchronously finding the bus route with a single transfer (`findBusServicesWithTransferBetween`)

You can get -0.5 mark deduction for serious violation of style.

Submission

When you are ready to submit your lab, on `cs2030-i` , run the script

```
~cs2030/submit10
```

which will copy all files matching `*.java` (and nothing else) from your `~/lab10` directory on `cs2030-i` to an internal grading directory. We will test compile and test run with a tiny sample input to make sure that your submission is OK.

You can submit multiple times, but only the most recent submission will be graded.

Warning
Make sure your code are in the right place -- it must be in subdirectory named `lab10` , directly under your home directory, in other words `~/lab10` . If you place it anywhere else, it will not get submitted.