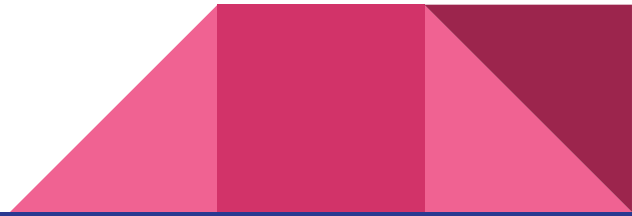
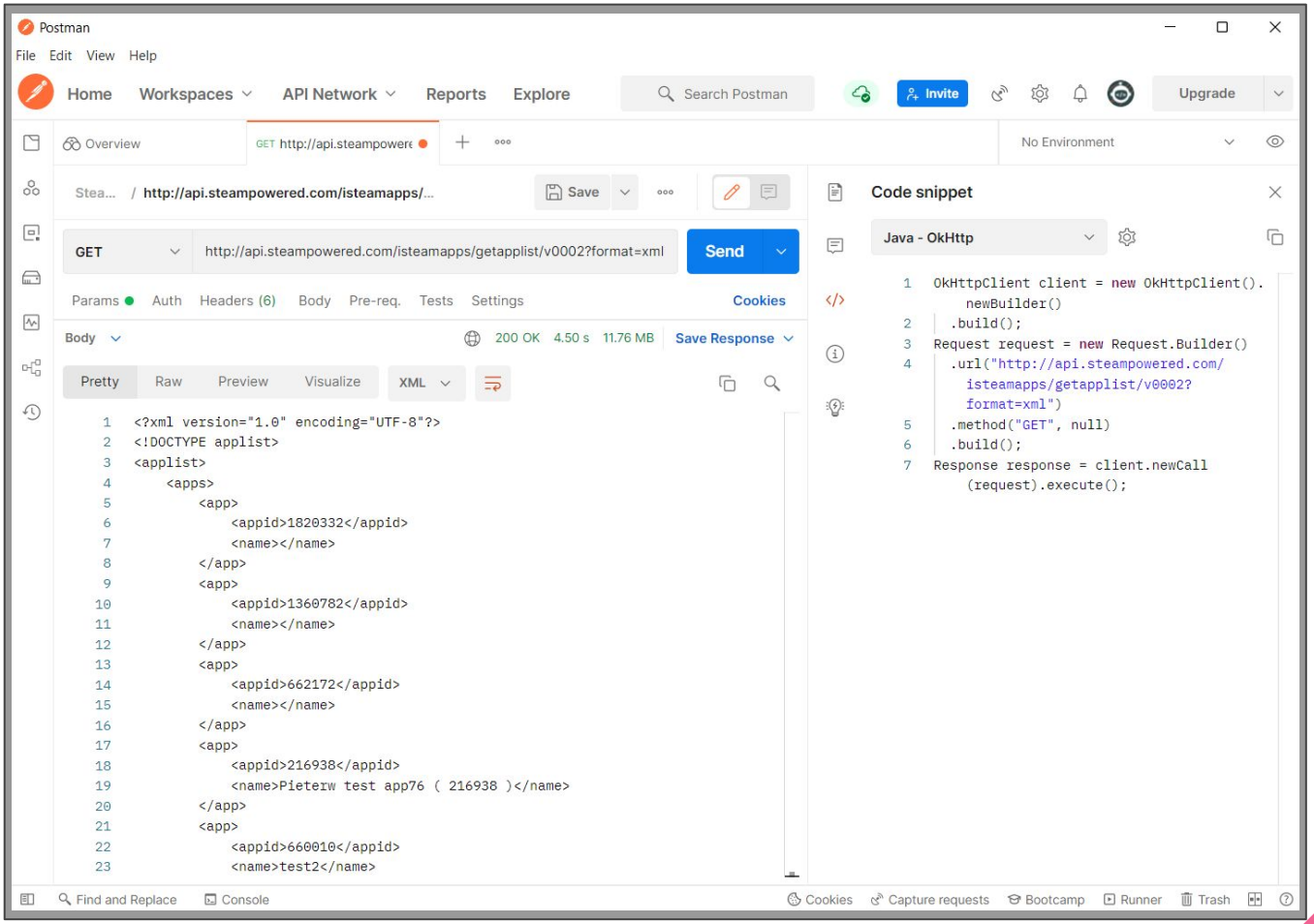
Team No Name Project

Steam Achievement Search Engine

Project Specifics

- Steam Achievement Search
- A search app that will return a list of achievements for a game on steam
- Joe, Alex, Vince
- The project is currently not finished. We needed to reduce the scope of what we were trying to achieve for time purposes.




Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
Overview
GET http://api.steampowere
No Environment
Stea... / http://api.steampowered.com/isteamapps/...
Save
GET
http://api.steampowered.com/isteamapps/getapplist/v0002?format=xml
Send
Params Auth Headers (6) Body Pre-req. Tests Settings
Cookies
Body
200 OK 4.50 s 11.76 MB
Save Response
Pretty Raw Preview Visualize XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE applist>
3 <applist>
4 <apps>
5 <app>
6 <appid>1820332</appid>
7 <name></name>
8 </app>
9 <app>
10 <appid>1360782</appid>
11 <name></name>
12 </app>
13 <app>
14 <appid>662172</appid>
15 <name></name>
16 </app>
17 <app>
18 <appid>216938</appid>
19 <name>Pieterw test app76 (216938)</name>
20 </app>
21 <app>
22 <appid>660010</appid>
23 <name>test2</name>
Code snippet
Java - OkHttp
1 OkHttpClient client = new OkHttpClient().
newBuilder()
2 .build();
3 Request request = new Request.Builder()
4 .url("http://api.steampowered.com/
isteamapps/getapplist/v0002?
format=xml")
5 .method("GET", null)
6 .build();
7 Response response = client.newCall
(request).execute();
Find and Replace
Console
Cookies
Capture requests
Bootcamp
Runner
Trash

eclipse-workspace - JavaFXControls/src/SteamGames.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

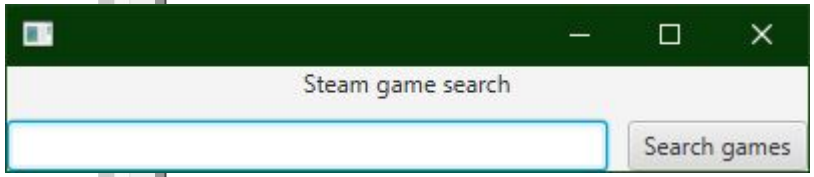
SteamGames.java

```
1  import javafx.application.Application;
15
16 public class SteamGames extends Application {
17
18     @Override
19     public void start(Stage mainStage) throws Exception {
20         // TODO Auto-generated method stub
21
22         //Button for searching steam games
23         Button searchButton = new Button("Search games");
24
25         //Textfield to get game user is searching for
26         TextField gameSearchTextField = new TextField();
27         gameSearchTextField.setPromptText("Enter a game to search for here...");
28         gameSearchTextField.setPrefWidth(300);
29
30         //Labels
31         Label applicationNameLabel = new Label("Steam game search");
32
33         //hboxes for horizontal placement
34         HBox applicationNameHBox = new HBox(applicationNameLabel);
35         applicationNameHBox.setAlignment(Pos.CENTER);
36         HBox searchGamesHBox = new HBox(10, gameSearchTextField, searchButton);
37
38         //vboxes for vertical placement.
39         //Build an HBox before this to place similar objects on the same horizontal plane
40         VBox mainVBox = new VBox(10, applicationNameHBox,searchGamesHBox);
41
42         //Adding top nested object container
43         //Currently this is mainVBox. This will be final unless we need to add GridPane.
44         Scene scene = new Scene(mainVBox);
45
46         mainStage.setScene(scene);
47         mainStage.show();
48
49     }
50
```

Console

<terminated> SteamGames [Java Application] C:\Program Files\Java\jdk-18\bin\javaw.exe (May 11, 2022, 9:45:52 AM – 9:45:57 AM) [pid: 75064]

Writable | Smart Insert | 62 : 6 : 2130



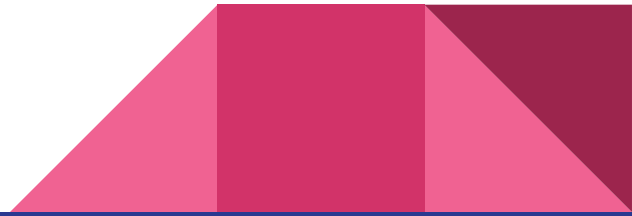
Functioning API Call

```
public ArrayList<Response> OkHttp() throws IOException {

    OkHttpClient client = new OkHttpClient().newBuilder().build();
    Request request = new Request.Builder()
              .url("http://api.steampowered.com/isteamapps/getapplist/v0002?format=XML")
              .method("GET", null)
              .build();
    Response response = client.newCall(request).execute();
    ArrayList<Response> apiResponse = new ArrayList<Response>();
    apiResponse.add(response);
    return apiResponse;
}
```

Background

- We’re all gamers, so it seemed like a fun idea
- Initially we wanted to create an app where you could get all you achievements, and compare them against your friends
- No one was personally tied to the idea, but we all liked the idea of doing something with a hobby we all personally enjoyed



Technology

- Initially we planned to make a web application, but found that difficult due to all the extra technology that would need to be involved. We decided to switch to Java, as we all had recent experience with it.
- We used Eclipse for doing any work directly to the UI, but styling was sometimes done using replit
- We used Discord for communication as a team outside of class
- Github was a great help, since it allowed us to also use replit
- The original technology to make the program a web app proved difficult. It was going to require the use of HTML/CSS, along with PHP and SQL on the back end. Java provided a simpler way to interact with the steam API, while not requiring a multitude of supporting technology to get started.

Human Factor

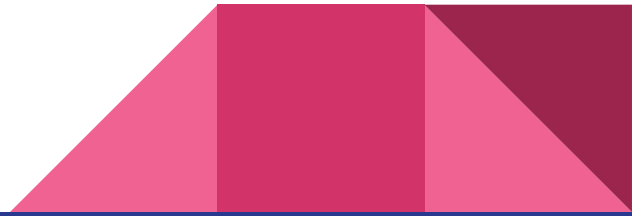
- As a group we decided to have Joe as the lead, due to experience with running a project in another class. Alex stepped up to handle a lot of documentation related to the project by his own choice. Finally, Vince handled the styling as he expressed an interest in it the most out of the group.
- We mainly decided to meet in class each week, and reach out to each other over discord whenever possible. Occasionally we met on the weekend to voice chat for planning the direction of the project.
- Life is busy, so the main issue we encountered was mainly just a lack of time due to work, family, or school.

Preparation

- Having a weekly assignment to see what weekly progress was planned was helpful.

Lesson learned Joe

- I would have started with a much easier idea to begin. It also would of been good to start with the same technology in terms of what language we used, instead of changing halfway through the project, but time dictated changing to something easier to understand
- Maybe more class time, and less lecture. For our group, the class time was almost the only time we were able to all get together and work collectively.

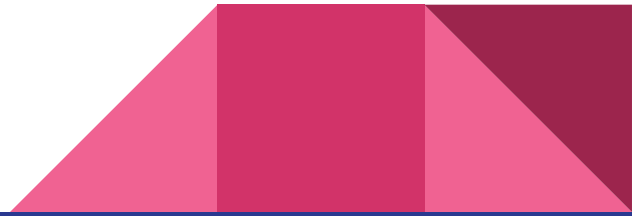


Lessons learned Alex

- I think I would keep the focus narrower and not try to reach crazy goals early on. Getting a project running in small sections at a time seem to work better than trying to navigate to an end-goal early on.
- Working with a team has its ups and downs. Relying on team members isn't something I'm used to so it took a little getting used to.
- For future classes, I think it would be a good idea to maybe give more time for coding and less time going over the ins and outs of GitHub etc. Maybe there's room in an earlier class to introduce GitHub, so you don't have to focus so heavily on it in this class.

Lessons learned Vince

- Set aside more personal time outside of class for actual coding
- Less lecture time in class and more code/planning time



Group lessons learned

- Big projects don't have to be stressful
- Working with a team is just as important as final product