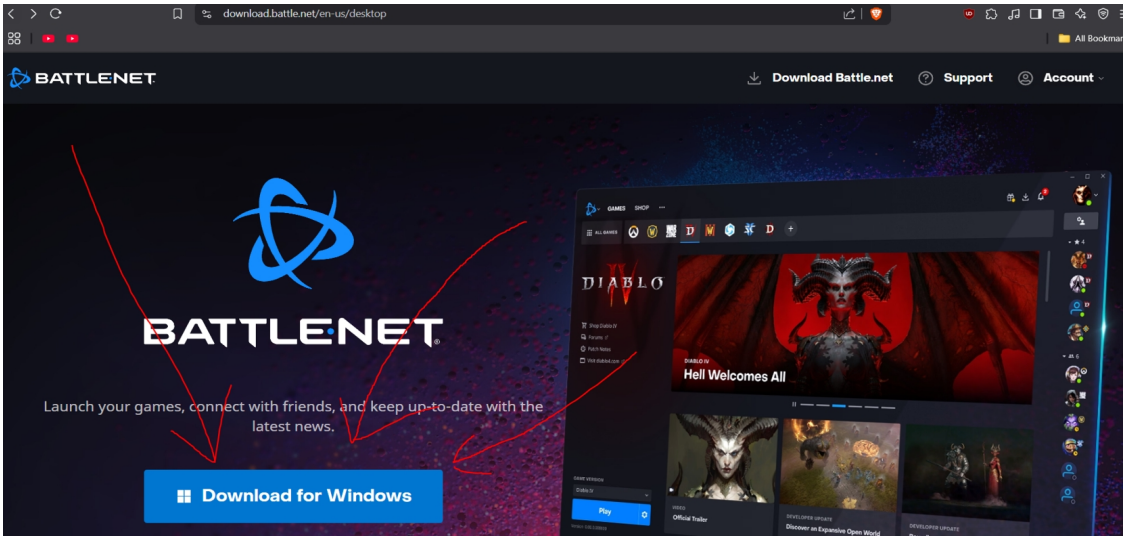
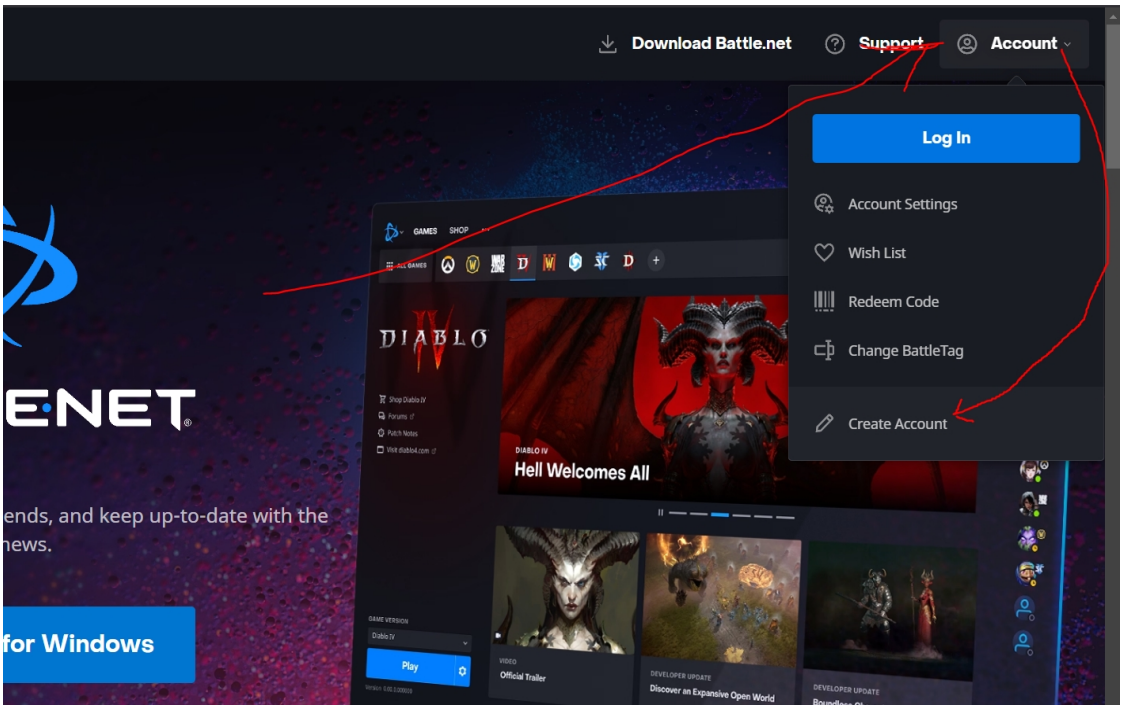
Instructions on how to run Starcraft II bot

1 Install Starcraft II

- 1.1 Install Battlenet and create an account.
  - If you already have battlenet, skip step 1.1.
  - Blizzard is the company that made Starcraft II and their games are downloadable through Battlenet. I believe this the website for Battlenet -> https://download.battle.net/en-us/desktop. Creating an account is also necessary.

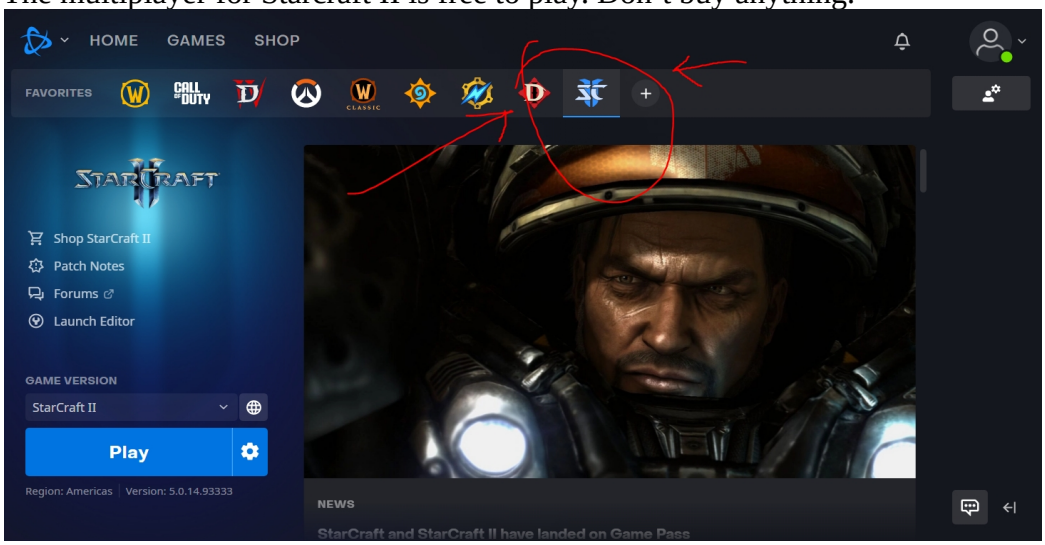




Creating an account and downloading an account is necessary because blizzard decided to make their games all run through battlenet. Having an account is necessary to view replays and download Starcraft II. Playing local bot games does not require login. The name used for the account does not matter. It will only be used for downloading Starcraft II and watching replays. The api uses “local player”.

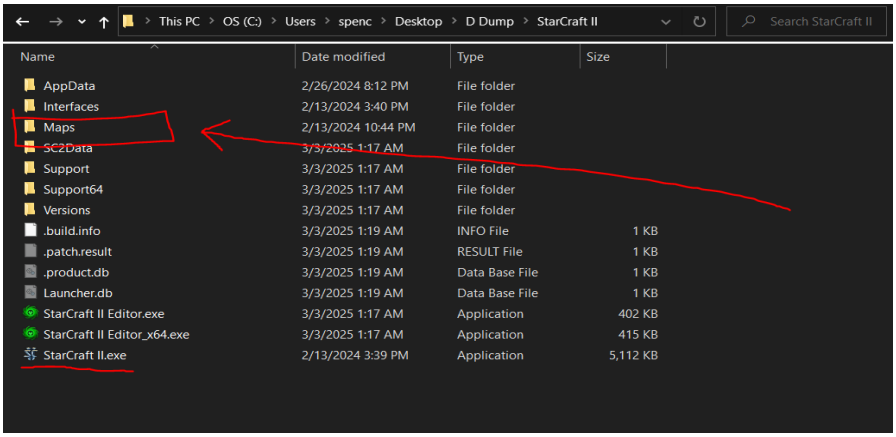
- 1.2 Install Starcraft II

- If you already have Starcraft II, skip step 1.2.
- The multiplayer for Starcraft II is free to play. Don’t buy anything.

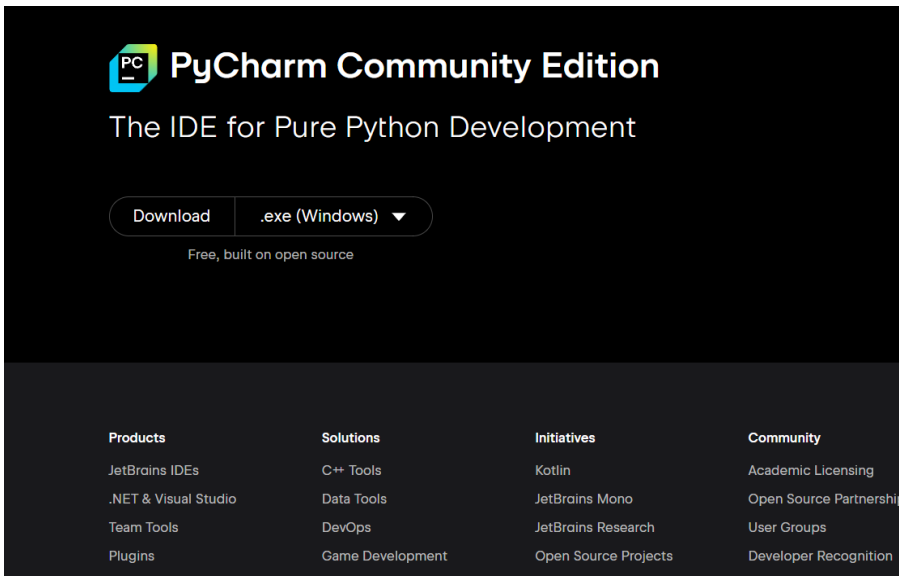


2 Set up things

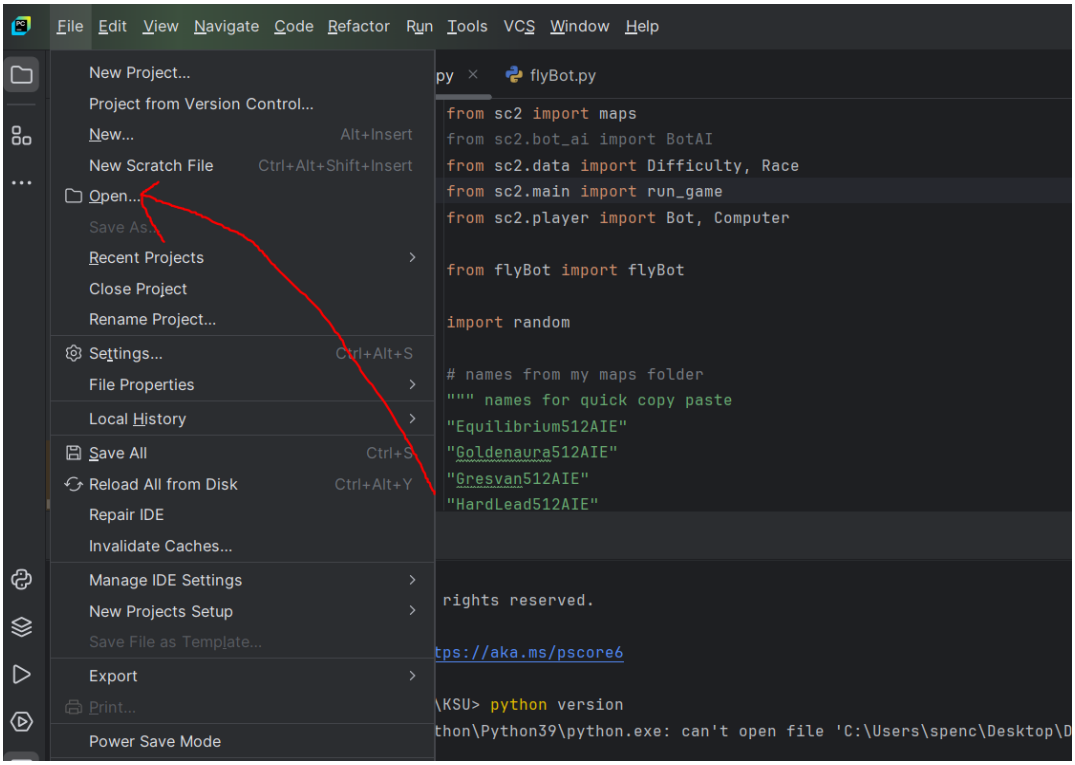
- I thought about trying to set the project as an .exe and decided it was a bad idea.
  - It is more modifiable as a python project.
  - The hardest part of this project is setting up the environment and that would be required even if it was an executable.
- 2.1 Set up “Maps” folders
  - The the api can not run local games unless there is a map folder with maps in it. I’m going to include a map folder with 1 map and set up my test script to reference that map. There is a maximum file size and the map might be too large. I intend to include a map folder named “Maps” in my project submission. Copy and paste the map folder to same folder where Starcraft II.exe is located. If I did not include a map folder or you want to add your own maps, you can find maps here https://aiarena.net/wiki/maps/. I am 99% sure I am using Sc2 AI Arena 2024 Season 1 as my map pool of commented out maps. The default locations for windows PC to install Starcraft II is here "`C:\Program Files (x86)\StarCraft II\`” according to the wiki. The file path can be changed. My filepath is different.



- 2.2
  - Install Python. I am using Python 3.9.13. I think this is the website for python. https://www.python.org/downloads/
- 2.3
  - Install Pycharm. This is the website for the windows version. https://www.jetbrains.com/pycharm/download/?section=windows. If you are not using windows, there is probably a different place to find the correct version. The free version of pycharm requires scrolling to the bottom of the page. I do not know why they had the free version at the bottom.

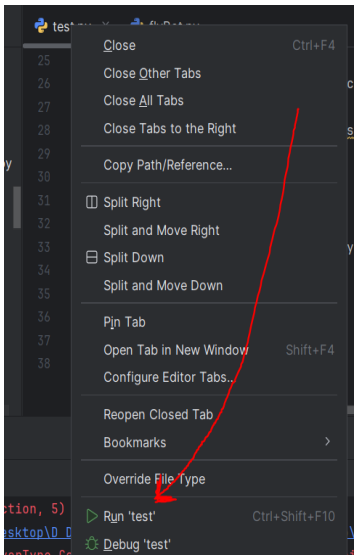


- 2.4 Open the project in Pycharm
  - Get Pycharm working. After Pycharm is working. Open the project.



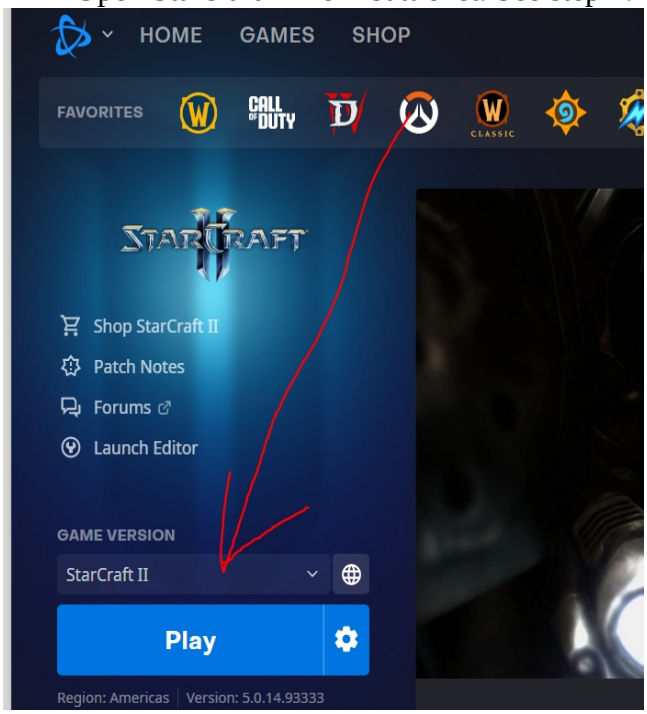
3 Change and Run the Test File.

- It gets a little bit complicated.
- 3.1 There are things that can be changed in the test script
  - If you don't want to change anything, skip step 3.1.
  - In my test script,
    - I have a variable named "myMap". myMap is a string for the map that is being run. It references a file name in the map folders previously mentioned in step 2.1. The file extension is "SC2Map". Do not include the ".SC2Map" at the end of the file name. Changing the filename enabled playing on different maps.
    - I have a variable named enemyFaction. That is the faction the enemy plays.
      - The game has main 3 factions and a 4th option called "Random". The enemy faction shows up on loading Screen. Although there are technically 3 factions to choose from, if random is chosen it displays as random on the load screen and is quarried as random in the api.
    - There is a line that reads "c = Computer(enemyFaction, Difficulty.VeryHard)". Difficulty can be changed to a different difficulty. I believe there is a bug in the api with difficulty. High difficulties are one higher.
    - Inside of method being called that is named "run_game", there is a thing called realtime. It is set to true or false. It is the most massive thing that can be changed in the test script. "realtime" changes how "async def on_step(self, iteration):" from the bot class works. There are 22.4 game steps per second in Starcraft 2. When realtime is set to false, the game runs at the same speed as bot iterations. If the bot the bot is written optimally, "realtime=false" will run faster than "realtime=true". Having it set to false also makes it so that bot performance is more consistent because it has the bot calculations linked to to the game step. My bot uses specific timings to make units group together better before attacking. Having it set to "realtime=true" makes it play at normal speed. "true" is better for humans. It is bad for bots because the timing isn't linked to game step anymore.
    - Do you want to play against the bot?
      - I have some commented out code underneath my test script. It is set up for human vs bot. It's not part of the scope of the project and the quality is going to be low but it is an option.
- 3.2 Run the test script.
  - I like to right click the test script and then click to run it.

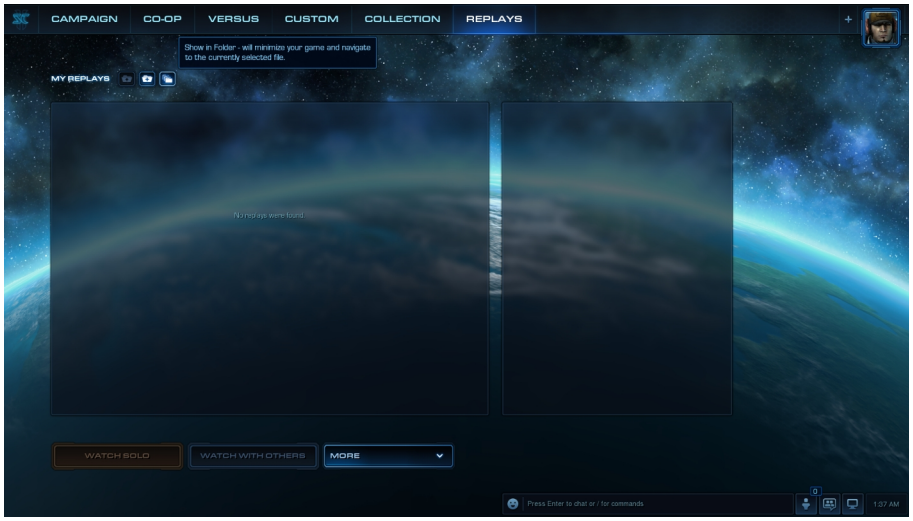


4 Replays

- The advantage of realtime=true for bot vs bot/computer games is that it is more human viewable. Replays are an alternative for this. Bots perform better on realtime=false. Re-watching using the replay feature in sc2 allows for watching in realtime without compromising the bots performance. It also allows for a higher level analysis detail.
- 4.1 Finish one or more games.
  - I think replays only save if the game finishes and they save in a weird place.
- 4.2 Open Starcraft II, the actual game.
  - Open Starcraft II from battlenet. See step 1.1.



  - If you get to the load screen you should be fine. With clicking from file explorer.
- 4.3 Watch replay
  - There are two ways of I know of on how to watch replays. Both require step 4.2. The normal way is to open starcraft2 and click “Show in folder - …”. It will open the file explorer. It is also possible to manually open file explorer. After finding the file, double click it.



- Finding the file is more complicated than it should be. I do not know the correct terminology but the local data is in a different place than the game files. The local files which include replays and account data are located in “\Documents\StarCraft II” for me. It is a different file location than where Starcraft II was installed.
- You may have noticed from my previous image but, the replay files are not displayed on the Starcraft II game under replays. That is fine and expected. The reason why is because the file system is complicated. It is displaying player replays.
  - On my PC, player replays are located here “C:\Users\spenc\Documents\StarCraft II\Accounts\880887116\1-S2-1-13247786\Replays”
  - On my PC, Bot replays are located here “C:\Users\spenc\Documents\StarCraft II\Replays”
- I can speculate on why it saves them in two different places. I’m trying to be objective, so I’m not speculating. “It is what it is”

5 Debugging

```
    ability=bot_object.game_data.abilities[proto.ability_id],
KeyError: 4449
```

- If you get any error that has “KeyError” and some kind of id, there is a chance that Id constants aren’t synced with the game version. Pretty much every time the game updates, the Ids change. Updates are not common enough to be a major problem but it does happen. Ids are similar to Enums and have the same functionality.
  - The api I’m using has a script in for updating Id constants. It is located here “\ Fall2025StarCraftBot\.venv\Lib\site-packages\sc2\generate_ids.py”
- So the game is running and it decides to randomly crash after 5 minutes? That is because I make coding errors. It happens.

6 What have I not finished yet?

- I think I accomplished most of what I agreed to do.

NF1: Adhere to bot rules.	Some Bot Rules were posted later in the document
NF2: Installability	Bot needs to be packaged in a way that makes it runnable on any machine that has same IDE, programming language, and Starcraft II installed
NF3: Usability	Bot must be simple and easy to run
NF4: Reliability	Bot must run without crashing over 90% of the time.
NF5: Security	Bot must not do anything beyond playing Starcraft II
NF6: Performance	Bot must run at a speed that is same speed or faster than game speed steps.
NF7: Adaptability(Map Tolerance)	Bot can play on any ladder map with normal functionality.
NF8: Adaptability(Enemy Tolerance)	Bot can play against each other faction with normal functionality
F1A: Beat in game bot	Bot must be able to beat at least one in game bot difficulty.
F1B: Beat difficulty Hard	Bot must be able to beat hard difficulty bot
F1C: Beat difficulty Harder	Bot must be able to beat harder difficulty bot
F1D: Beat difficulty Very Hard	Bot must be able to beat very hard difficulty bot
F1E: Beat difficulty Elite	Bot must be able to beat elite difficulty bot
F2: Unit tier	Bot must tech into at least one max tier unit(s).
F3: Upgrades	Bot must research unit upgrades
F4: Expansion	Bot must expand based on strategic factors.
F5: Unit Mirco	Units must react to nearby stimuli such as enemies.
F6: Buildings/replacing structures	Bot must build structures when needed.
F7: Produce units	Buildings that can produce units produce units when needed.
F8: Scouting	Bot must be able to deduce enemy location trough scouting or other means.
F9: Rally	Bot must be able to send most units to go to location for various reasons.

  - Most of usability issues come from the required set up. I could write directions to make it more usable. The only other way I could make set up easier would be if I set up a script to install maps and Starcraft 2. I can't do that because it is extremely sketchy. It would be a violation of Non-functional requirement 5.
  - I can test for Adaptability(Map Tolerance) by testing more maps. I only have 6 maps in my map pool. One map has a ramp that doesn't work and the bot adapts to map not working. There are more chaotic maps I can. I do not have red shift in my map pool which is one of the most controversial ladder maps ever added. Red shift is a wild map.
- My bot still has weakness and I can fix those weaknesses.
  - My bot is weak to early rushes.
    - This is acceptable by game theory. My bot should lose to early rushes. Most games have some kind of rock paper scissors. In starcraft2, early rush beats eco rush, eco rush beats standard, and standard beats early rush.
  - My bot builds 2nd stargate too soon. I can delay it.
  - My bot tends to have + 4 workers over saturation at 3 bases. I can either change the worker production logic or expansion logic.
  - My bot still builds too close to mineral line.
  - My bot still occasionally walls itself off by building bad.
    - Nexus have warp and that is a band aid solution
  - Bot needs to overreact less. That needs to be patched.
    - Sends a few too many workers when doing a worker defend
    - Attack nearest army macro logic needs to be refined. One zergling is not worth sending the whole army.