😵 WindowsAgentArena Setup

Introduction

This is the WindowsAgentArena (WAA) setup with Agent S2 (and beyond). Why do we need a setup guide? Despite the thorough README.md, we have to include our code into their repository *and* fix up a number of setup issues from the WAA environment. Sadly, this isn't the most straightforward.

Initial WAA Setup

The initial WAA setup is straightforward. Follow the README.md on their repository. After you've finished this, try running `run-local.sh`. This will start up an experiment with their default `Navi` agent. At this point, the environment is *sufficient to run evaluation*, but it's incomplete and thus the evaluation won't be exactly correct due to environment issues.

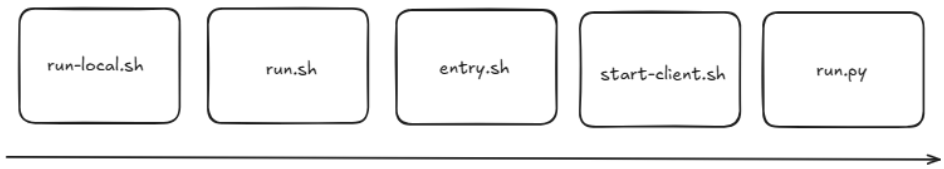

Figure 1: Bash script chain of execution.

While we're at it, look to understand the following things:

- the entire README.md (especially the Bring Your Own Agent guide)
- the *long* chain of bash scripts that start the run (Figure 1)
- the `run.py` to see how the agent/environment are instantiated and used together
- the folder structure of the repository and the purpose of each folder

Fixing Setup Issues

By now, your WAA environment should be set up to run locally. There are two major problems:

- setup issues
- the VM persists across examples (it won't reset after every example is completed which may make evaluation unfair)

Let's tackle the first one: setup issues.

Office Apps Aren't Installed

The first issue I ran into was the office apps aren't installed. Why is that? Turns out all apps installed in the VM during the initial setup stage install via the links from this file (`tools_config.json`). At the time of writing this, only the office links do not work. Try out all the links to make sure they work. If the links do not lead to a download (and some error occurs), then that app was not installed in the VM. What do we do? Two options:

- redo the entire initial setup stage (time consuming; ~**4** hours for me and even then, it would just not work a lot of the times; ideally, WAA is setup on Linux as I've had no issues so far with it)
- Enter the VM and install the apps manually (easier and faster)

We'll do the second approach.

You can access the VM via `https://localhost:8006`. You can turn the VM on by `run-local.sh`. There's probably a better/faster way to do it, but this doesn't take too much time anyways (~**1-2** mins). After the VM has started, enter the VM (the agent may be trying to take actions, but you can either just override the action in `run.py` with `import time; time.sleep(10000)` here or fight the agent for control of the VM!).

Inside the VM, navigate to their download page and download the latest LibreOffice version. After it's downloaded, complete the setup wizard and make sure to delete the downloaded `*.msi` file in the VM. Finally, test the download by opening up LibreOffice Writer and Calc.

Google Chrome Pop-ups

In Google Chrome, there a couple unexpected pop-ups.

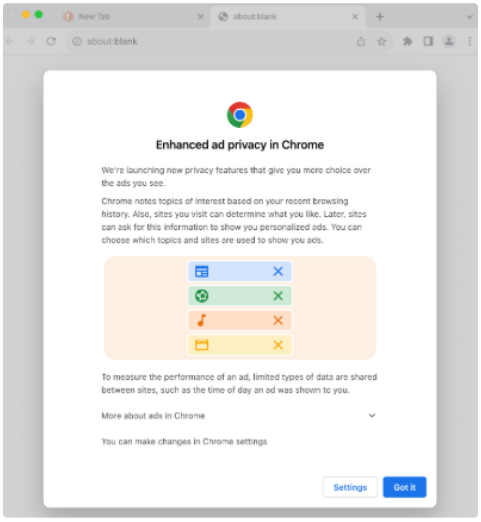

Figure 2: Pop-ups on Chrome.

Close all these pop-ups and make Google Chrome your default web browser.

VSCode Pop-ups

This isn't as important, but there are a couple initial pop-ups in VSCode that you can close.

Note: `set_cell_values`

❗ *Important if you're using* `set_cell_values`

Agent S2 uses a special grounding function called `set_cell_values` that takes advantage of the `soffice` CLI and `unotools` Python library. TL; DR, this function lets the agent set the cell values for a given spreadsheet and sheet.

For this function to work on WAA, the set up is a bit messy...

1. Connect into the VM

2. Open up a terminal and run `python --version`, you should see you're using the GIMP Python which is `2.x` 😤. This won't let you use the `soffice` CLI or `import uno` in Python code.
3. In the `Desktop` directory within a terminal, do `pip freeze > requirements.txt` to save all the PYPI libraries from the GIMP Python to a `requirements.txt`.
4. Configuring Python path to LibreOffice's Python
   a. In the File Explorer, locate the `python.exe` file from LibreOffice. You can do this with `where python`. Copy this path.
   b. In the Search bar in the bottom task bar inside the VM, search for "environment variables".
   c. Click on "Environment Variables" and click on "Path" under "System variables". Paste the copied path from step (a) into there and ensure this path is *above* the GIMP Python path so it takes precedence.
   d. Reopen a terminal and run `soffice` to ensure it is now working. Create a temporary python file and ensure `import uno` works.
5. LibreOffice's Python should be `3.10` or above. However, it does not come with pip. To install pip, download this file and execute `python get-pip.py` to install it. Ensure the `python` here is LibreOffice's Python. Next, install `pip install -r requirements.txt` using the `requirements.txt` from step 3. This is to ensure LibreOffice's Python has all the dependencies needed for evaluation (pyautogui, etc).
6. Clean up all installer files. Then, inside the WAA repository code, change this line

```
command_list = ["python", "-c", self.pkgs_prefix.format(command=command)]
```

to:

```
command_list = ["absolute/path/to/libreoffice/python", "-c", self.pkgs_prefix.format(command=command)]
```

This ensures that the subprocess running in the flask server inside the VM will use that specific Python version.

Double Checking...

Double check all apps can be used and no unexpected pop-ups or issues are in the way. Any apps you open make sure to close them upon finishing your clean-up. Make sure any installation files you have in `Downloads` are deleted (and removed from Recycle Bin) to keep the environment clean. At the end, this is our **golden image**. You may want to save a copy of this VM somewhere safe so that you can always copy it back into the WAA repository to be reused (refer to this).

Set up Agent S2 with WAA Locally

Take the time to understand the Agent-S repository.

Then, take a look at this setup guide for Agent S1. This setup won't be used for Agent S2, but it's a useful reference.

1. Instead of following the README.md for Agent S2, you need to clone the repository then `pip install -r requirements.txt`
2. Move the s2 folder to the mm_agents folder in WAA. Follow the Bring Your Own Agent guide.
   a. You will need to move the `agent_s.py` file out to the `s2` folder and update all the relevant import statements
3. Make the necessary changes in `run.py` and `lib_run_single.py` to accommodate Agent S2 (replace the Navi Agent with Agent S2).
4. Test it by running the experiments! Don't forget when you do `run-local.sh`, now you need to specify Agent S2 instead of the navi agent `agent="agent_s"`.
5. You may have some import errors and these libraries need to be installed inside the `winarena` container (I think). You can just add the pip install commands to the bash script where the error stems from (hacky).

Perplexica

There may be a Perplexica issue. The Perplexica URL must be configured so that the agent in the `winarena` Docker container can communicate with `localhost:3001` which is the forwarded port from the Perplexica container. On Mac/Windows this can be fixed by changing the `PERPLEXICA_URL` to `http://host.docker.internal:3001/api/search`. On Linux, I just disabled it... I haven't tried,

but you can add `--add-host=host.docker.internal:host-gateway` as a flag to the docker command here (run.sh). This may let you use `http://host.docker.internal:3001/api/search` as the `PERPLEXICA_URL`

Agent S2 with WAA on Azure

1. Ensure you have:
   a. a **clean copy** of the golden image
   b. the correct Azure subscription (so you're not using your own payment method)
2. Follow the Azure deployment in the README.md.
3. Test it! If this works, then we have a resettable golden image and WAA can be ran in parallel, making evaluation much *much* faster! Good luck!

References