perplexity

pre-commit question

I usually run `pre-commit run --all-files` before `git commit`

We also run pre-commit-ci

I want to add a script for in my repo that is run as part of `pre-commit run --all-files` and pre-commit-ci. Is that supported? What are the steps to set this up

Yes, it is fully supported to add a custom script to your repository that runs as part of both `pre-commit run --all-files` and pre-commit.ci. Here's how you can set this up:

Steps to Add a Custom Script to pre-commit and pre-commit.ci

1. Write Your Script

Create your script (for example, `scripts/my-check.sh`) and make sure it is executable. For shell scripts, start with a shebang (e.g., `#!/bin/sh`) and ensure any dependencies are available in the environment where it will run[1] [2].

2. Configure the Hook in `.pre-commit-config.yaml`

Add a `local` repo entry in your `.pre-commit-config.yaml` at the root of your project. Here's an example:

```
repos:
  - repo: local
    hooks:
      - id: my-custom-check
        name: My Custom Check
        entry: scripts/my-check.sh
        language: script
        pass_filenames: false  # Set to true if your script expects filenames as argument
        always_run: true       # Ensures it runs even if no files match
```

- `entry` points to your script.
- `language: script` tells pre-commit to run it as a shell script.
- `pass_filenames` determines if pre-commit passes file names to your script.
- `always_run: true` ensures the hook runs on every invocation, even if no files match[1].

3. (Optional) Use `.pre-commit-hooks.yaml` for Shared Repos

If you want to share your hook as a reusable repo, add a `.pre-commit-hooks.yaml` file describing your hook (not necessary for local hooks)[3].

4. Install or Update pre-commit

Run:

```
pre-commit install
pre-commit install-hooks
```

This sets up the git hook scripts and preps the environments[4] [2].

5. Test Locally

Run:

```
pre-commit run --all-files
```

This will execute your new hook on all files in the repository, just as pre-commit.ci would[4] [2].

6. pre-commit.ci Integration

pre-commit.ci automatically runs all hooks defined in `.pre-commit-config.yaml` on every pull request and commit, including your custom script-no extra configuration is needed[5]. If your script is part of the config, it will run in CI.

Notes and Tips

- If your script relies on certain tools or dependencies, ensure they are available in the CI environment. For `language: script`, pre-commit.ci provides a standard shell environment.
- If your script should only run on certain file types or patterns, use the `files:` or `types:` keys to restrict its scope[2].
- If your hook is being skipped, make sure to use `--all-files` or set `always_run: true` to force execution[1].

Example `.pre-commit-config.yaml` Snippet

```
repos:
  - repo: local
    hooks:
      - id: my-custom-check
        name: My Custom Check
        entry: scripts/my-check.sh
        language: script
        always_run: true
```

Once configured, your script will run both when you use `pre-commit run --all-files` and automatically in pre-commit.ci on pull requests and pushes[5] [1] [2].

⁂

Is `entry: python` supported?

No, setting `entry: python` by itself is not supported in pre-commit hook configuration. The `entry` field specifies the command or script that should be executed when the hook runs. If you set `entry: python`, pre-commit will simply try to run the `python` command with no arguments, which does nothing and will not execute your intended script[6].

If you want to run a Python script as a pre-commit hook, you have two main options:

1. Specify the Python script directly (with a shebang):

- Make sure your script (e.g., `my_script.py`) starts with a shebang line like `#!/usr/bin/env python` and is executable.
- Set `entry: ./my_script.py` and `language: python` in your hook definition.

2. Use `python my_script.py` as the entry:

- Set `entry: python ./my_script.py` in your hook configuration. This explicitly tells pre-commit to run the Python interpreter with your script as an argument[6].

Example:

```
- repo: local
  hooks:
    - id: my-python-hook
      name: Run my Python script
      entry: python ./my_script.py
      language: python
```

or, if your script is executable with a shebang:

```
- repo: local
  hooks:
    - id: my-python-hook
      name: Run my Python script
      entry: ./my_script.py
      language: python
```

Summary Table

entry value	Supported?	Result
entry: python	✖	Runs `python` with no script; does nothing
entry: python my.py	✅	Runs `python my.py` as expected
entry: ./my.py	✅	Runs the script directly (needs shebang)

References:

- [pre-commit documentation][7]

- [Stack Overflow discussion][6] [7]: https://pre-commit.com[6]: https://stackoverflow.com/questions/58921711/using-the-python-library-pre-commit-how-can-i-add-a-git-pre-commit-hook-that

⁂

The shebang approach looks good. Will this work with pre-commit-ci (GitHub Actions)?

Yes, the shebang approach for Python scripts works with pre-commit.ci and GitHub Actions.

If your Python script has a proper shebang (e.g., `#!/usr/bin/env python3`), is marked as executable, and is referenced in your `.pre-commit-config.yaml` as a local hook with `language: script`, it will run correctly both locally and in CI environments like pre-commit.ci and GitHub Actions[8] [9] [10]. This is a standard and supported pattern for custom hooks.

Key points:

- pre-commit.ci and GitHub Actions both use environments where `/usr/bin/env python3` (or `python`) is available by default.
- The script must be executable (`chmod +x your_script.py`).
- Your `.pre-commit-config.yaml` should point to the script using `entry: ./your_script.py` and `language: script`.

This setup ensures your script runs identically in local and CI contexts, as both environments honor the shebang line to select the interpreter[8] [9] [10].

⁂

1. https://stackoverflow.com/questions/59499061/how-to-run-custom-shell-script-file-before-pre-commit-hook
2. https://verdantfox.com/blog/how-to-use-git-pre-commit-hooks-the-hard-way-and-the-easy-way
3. https://stefaniemolin.com/articles/devx/pre-commit/hook-creation-guide/
4. https://pre-commit.com
5. https://pre-commit.ci
6. https://stackoverflow.com/questions/58921711/using-the-python-library-pre-commit-how-can-i-add-a-git-pre-commit-hook-that
7. https://pre-commit.com
8. https://pre-commit.com
9. https://github.com/pre-commit/pre-commit-hooks
10. https://github.com/marketplace/actions/pre-commit