Working on projects

Creating a new project

You can create a new Python project using the uv init command:

```
uv init lesson3\hello-world
cd lesson3\hello-world
```

Alternatively, you can initialize a project in the working directory:

```
mkdir lesson3\hello-world
cd lesson3\hello-world
uv init
```

uv will create the following files:

```
├── .gitignore
├── .python-version
├── README.md
├── main.py
└── pyproject.toml
```

The `main.py` file contains a simple "Hello world" program. Try it out with `uv run`:

```
uv run main.py
=> Hello from hello-world!
```

Managing dependencies

You can add dependencies to your `pyproject.toml` with the `uv add` command. This will also update the lockfile and project environment:

```
uv add requests
```

You can also specify version constraints or alternative sources:

```
# Specify a version constraint
uv add "requests==2.31.0"

# Add a git dependency
uv add git+https://github.com/psf/requests
```

if you're migrating from a `requirements.txt` file, you can use `uv add` with the `-r` falg to add all dependencies from the file:

```
# Add all dependencies from "requirements.txt"
uv add -r requirements.txt -c constraints.txt
```

To remove a package, you can use `uv remove`:

```
uv remove requests
```

To upgrade a package, run `uv lock` with the `--upgrade-package` flag:

```
uv lock --upgrade-package requests
```

Init Project

```
mkdir lesson3\temp-project
cd lesson3\temp-project
uv init

# add package
uv add requests

# remove package
uv remove requests

# also, you can build your source
# check dist directory
uv build
```