



# Environments


Environments in conda are self-contained, isolated spaces where you can install specific versions of software packages, including dependencies, libraries, and Python versions. This isolation helps avoid conflicts between package versions and ensures that your projects have the exact libraries and tools they need.


## Why should I create a new environment?

There are several reasons you might want to create a new environment:

 **Isolation of dependencies** - Environments isolate software and their dependencies from the rest of the software installed on your machine. This means you can have both Python 3.9 and Python 3.10 installed on your machine and use both versions without encountering issues.

 **Reproducibility** - By creating an environment for each project, you can ensure that your code runs consistently across different machines. [Sharing the environment configuration](#) allows others to replicate your setup, ensuring that they have the same package versions and dependencies.

 **Ease of management** - Conda provides tools to easily [create, manage, and delete environments](#). You can quickly [switch between environments](#), making it simple to manage multiple projects with different requirements.

 **Testing and development** - Environments are perfect for testing new packages or libraries without affecting your stable development setups. You can experiment freely and remove the environment if things don't work out, without impacting your other projects.

## Why shouldn't I work in the base environment?

When first installing and using conda, you probably saw references to something called `base` or a “base environment”. This environment is where conda itself is installed, and should only be used for installing anaconda, conda, and conda-related packages, such as `anaconda-client` or `conda-build`.

For your projects, however, Anaconda *strongly* recommends creating new environments to work in. This protects your base environment from breaking due to complex dependency conflicts and allows you to easily manage and reproduce your environment on other machines.

## Working with environments

For convenience, the most common actions users take when managing environments are detailed here. For a full list of actions and more comprehensive guide, see [Manage environments](#) in the official conda documentation. Alternatively, follow along with our [Getting started with conda environments](#) tutorial on Anaconda Cloud.



### Tip

If you prefer to create and manage your environments via our graphical interface, Navigator, see [Managing environments](#).

## Creating an environment

Create a conda environment by opening Anaconda Prompt (Terminal on macOS/Linux) and running one of the following commands:

### Empty environment

### Environment with Python and packages

```
# Replace <ENV_NAME> with a name for your environment
# Replace <PACKAGE> with your desired package
# Replace <VERSION> with your desired version of Python
conda create -n <ENV_NAME> python=<VERSION> <PACKAGE>=<VERSION>
```



### Note

- This downloads the listed packages and their dependencies.
- If you do not specify a version of Python or other packages, conda will attempt to install the latest version from its available channels.

### Example:

```
conda create -n myenv python=3.11 beautifulsoup4 docutils jinja2=3.1.4 wheel
```



### Tip

It's best practice to install all the packages you want in your environment at the same time, for a few reasons:

- It will be faster, because each solve takes time.
- Any incompatibilities between the packages you want are discovered quickly.
- If incompatibilities are found, then you haven't left your environment in a half-built state.

# Activating an environment

Because environments are isolated spaces, you can only work with one at a time. Selecting an environment to work with is called activating it.

Activate an environment by running the following command:

```
# Replace <ENV_NAME> with the name of the environment you want to activate
conda activate <ENV_NAME>
```

## Switching between environments

When you're ready to switch between projects, simply activate the environment of your other project. Activating a different environment will deactivate your current one.

1. (Optional) View a list of all your environments by running the following command:

```
conda info --envs
```

2. To switch to a different environment, activate it by running the following command:

```
# Replace <ENV_NAME> with the name of the environment you want to switch to
conda activate <ENV_NAME>
```

## Deactivating an environment

It is best practice to deactivate your environment when you are finished working in it.

To deactivate your active environment, run the following command:

```
conda deactivate
```

## Sharing an environment

Sharing your environment with someone else allows them to use conda to recreate your environment on their machine.

To share an environment and its software packages, you must export your environment's configurations into a `.yaml` file.

### Caution

Simply copying your Anaconda or Miniconda files over to a new directory or another machine will not recreate the environment. You must export the environment as a whole.

## Exporting the environment configuration `.yaml` file

### Warning

If you already have an environment configuration `.yaml` file in your current directory, it will be overwritten during the export process.

1. Activate the environment you want to export by running the following command:

```
# Replace <ENV_NAME> with the name of the environment you want exported
conda activate <ENV_NAME>
```

2. Export the environment by running the following command:

```
conda env export > environment.yaml
```

### Note

This file handles both the environment's pip packages and conda packages.

3. Share the exported environment configuration `.yaml` file with another user.

## Creating an environment from an environment.yaml file

If someone has shared an environment with you—or you need to recreate your environment on a new machine—follow these steps to create a new environment using the environment configuration `.yaml` file:

1. Run the following command from the directory where you stored the `environment.yaml` file:

```
conda env create -f environment.yaml
```

The first line of the file sets the new environment's name. For more details, see [Creating an environment file manually](#).

2. Once your environment is successfully created, conda provides you with commands to activate it. Once activated, you can begin working in your new environment.

Was this helpful?

[Send feedback](#)