# How to install Cadnano on macOS

Getting started

This chapter describes how to set up and configure your local **macOS** desktop or laptop so you can begin designing DNA origami nanostructures with **Cadnano**. If you have a PC, please see the **Windows** version of this document. This tutorial matches how we set up our own machines in the Douglas Lab as of December 2021.

When we refer to **Cadnano1**, we mean two separate "legacy" versions of Cadnano. The first is **cadnano v0.2.3** which supports honeycomb-lattice designs, and the second is **cadnanoSQ v0.2.4**, which supports square-lattice designs). **Cadnano1** was written in the ActionScript language (similar to JavaScript) for the **Adobe AIR** platform. **Cadnano2** is the Python port of Cadnano1. It uses **PyQt5**, a library that provides Python bindings for **Qt** (pronounced "cute"), a cross-platform GUI framework, which replace the function of Adobe AIR.

We find that it is useful to have a more than one copy of **Cadnano** available on your system. For example, you might want to open two versions of a design and edit them side-by-side. Also, while an OS update may often cause one version to stop working, it is rare that both will break simultaneously.

## Step 1. Install both honeycomb and square-lattice versions of Cadnano1

1. Install **Adobe AIR** from: https://airsdk.harman.com/runtime
2. Download **cadnano** and **cadnanoSQ** from http://cadnano.org/legacy
3. Double-click each installers and follow the instructions. You should see an Application Install dialog window. Don't worry about the warning, it just means that we didn't pay a tax to become a known publisher. When you proceed, you can choose to add a desktop shortcut, or install in a different location.
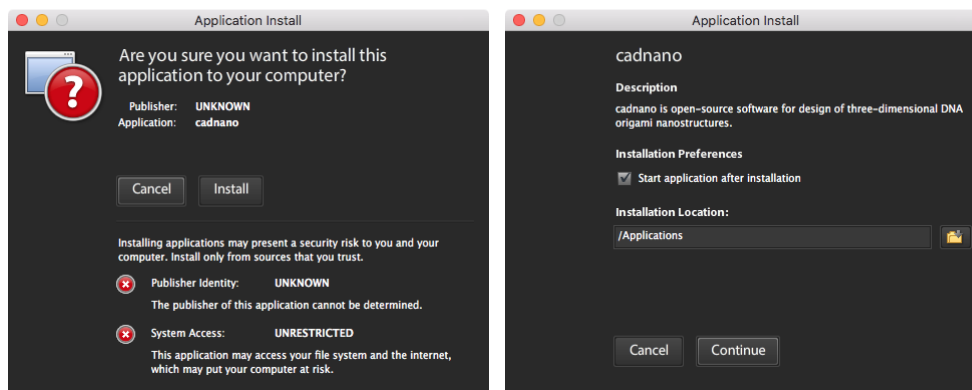
Step 2. Install Homebrew

Homebrew is a package manager for macOS that facilitates **python3** installation.

1. Open a **Terminal** (the app can be found by typing `terminal` into Spotlight search, or navigating via Finder to /Applications/Utilities/Terminal).
2. Open a web browser and visit the URL https://brew.sh/
3. Copy & paste the "Install Homebrew" command into the **Terminal**, it will be something like:
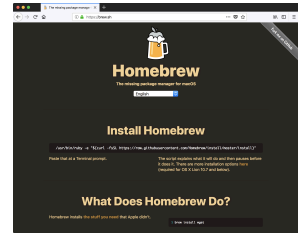
```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

4. Press RETURN when prompted. You may need to enter your password to install **Xcode Command Line Tools** or fix file permissions.
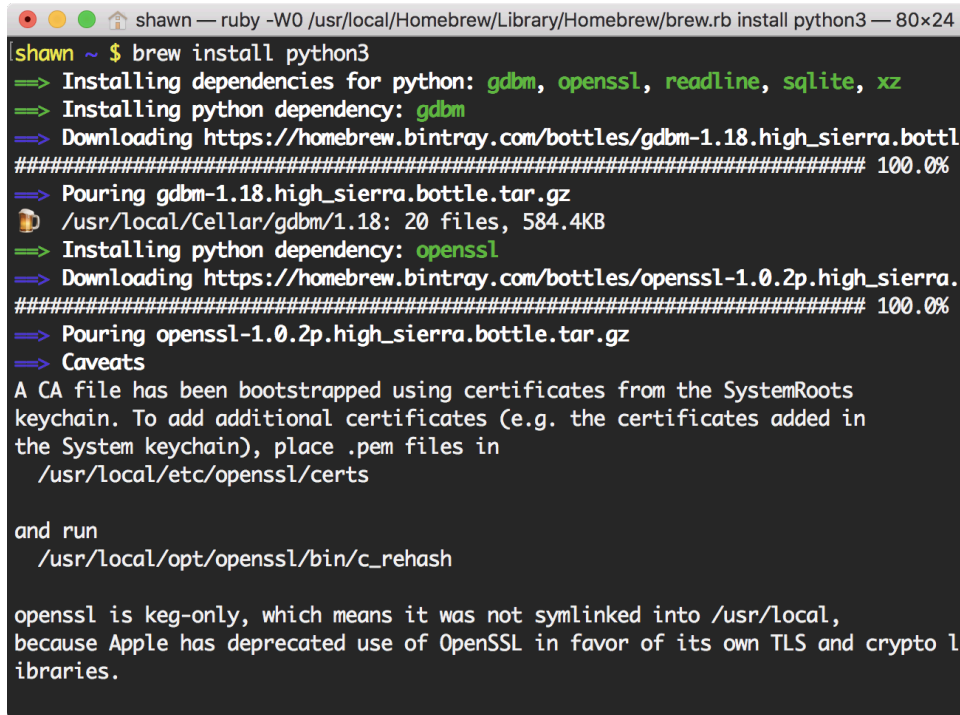
Step 3. Install Python3

1. Once **Homebrew** is successfully installed, type the following command to install python3:
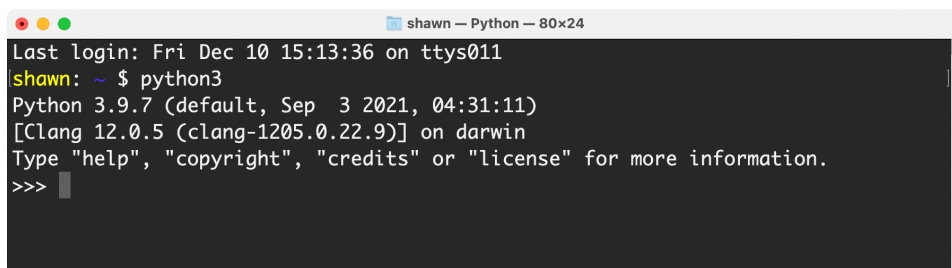
   ```
   brew install python3
   ```


2. If successful, you should be able to launch **python3** from the terminal by typing:

   ```
   python3
   ```


Step 4. Set up a virtual environment

We recommend using a virtual environment to manage your Cadnano2 installation. Fortunately, python3 provides native support for virtual environments. Virtual environments simplify the process of managing multiple packages that may have incompatible dependencies. They also protect your root Python3 installation — if you make a mistake, you can simply delete the virtual environment folder and start over.
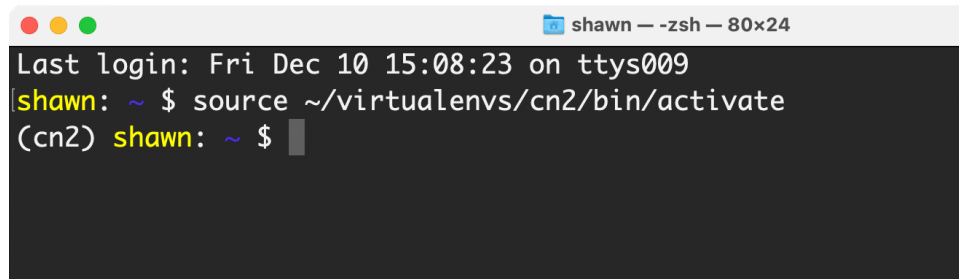
1. Create a new virtual environment called `cn2` by running this command from the **Terminal**:

   `python3 -m venv ~/virtualenvs/cn2`

2. Run this command to confirm that you can successfully activate the new virtual environment:

   `source ~/virtualenvs/cn2/bin/activate`

   If everything worked, your terminal prompt should include a `(cn2)` prefix.

```
●  ●  ●                    🗂 shawn — -zsh — 80×24
Last login: Fri Dec 10 15:08:23 on ttys009
[shawn: ~ $ source ~/virtualenvs/cn2/bin/activate
(cn2) shawn: ~ $ ▊
```

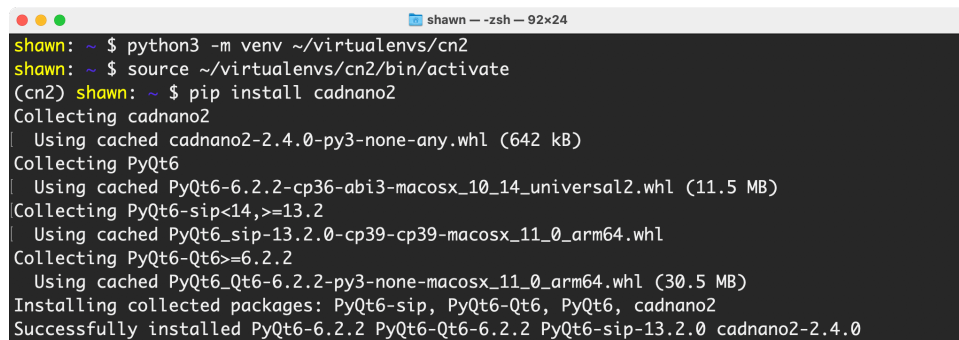## Step 5. Install Cadnano2 package

We will be using **pip**, a built-in command-line python package installer, to install **cadnano2**.

1. With your `cn2` virtual environment active in the **Terminal**, run the following command:

   `pip install cadnano2`

   You should see up to four packages install: `PyQt6`, `PyQt6-sip`, `PyQt6-Qt6`, and `cadnano2`

```
●  ●  ●                        🗂 shawn — -zsh — 92×24
shawn: ~ $ python3 -m venv ~/virtualenvs/cn2
shawn: ~ $ source ~/virtualenvs/cn2/bin/activate
(cn2) shawn: ~ $ pip install cadnano2
Collecting cadnano2
  Using cached cadnano2-2.4.0-py3-none-any.whl (642 kB)
Collecting PyQt6
  Using cached PyQt6-6.2.2-cp36-abi3-macosx_10_14_universal2.whl (11.5 MB)
[Collecting PyQt6-sip<14,>=13.2
  Using cached PyQt6_sip-13.2.0-cp39-cp39-macosx_11_0_arm64.whl
Collecting PyQt6-Qt6>=6.2.2
  Using cached PyQt6_Qt6-6.2.2-py3-none-macosx_11_0_arm64.whl (30.5 MB)
Installing collected packages: PyQt6-sip, PyQt6-Qt6, PyQt6, cadnano2
Successfully installed PyQt6-6.2.2 PyQt6-Qt6-6.2.2 PyQt6-sip-13.2.0 cadnano2-2.4.0
```

Note: the exact versions may differ.

Step 6: Run Cadnano2

1. The python package adds a console script to the path:

```
cadnano2
```
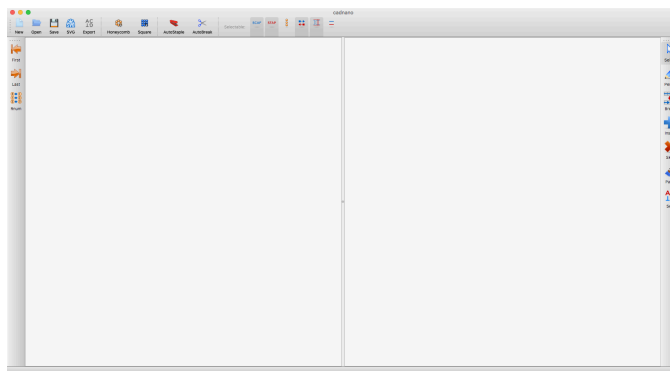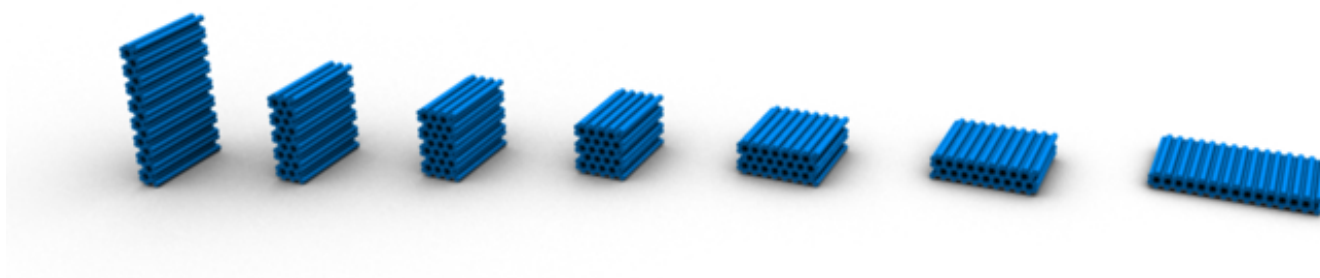

Hopefully, the **cadnano2** window launched!

Both honeycomb and square lattice designs are handled by the same version of **cadnano2**, so we are pretty much ready to start designing! You can click on the **[Honeycomb]** or **[Square]** button in the interface to get started, or visit http://cadnano.org/gallery.html to download some published DNA origami designs to open and play with.

Step 7: Create an alias to launch cadnano2 from a single command

Once cadnano2 is working in your virtual environment, open a new **Terminal** and run the following:

```
cat <<END >> ~/.zprofile
alias cn2="source ~/virtualenvs/cn2/bin/activate && cadnano2"
END
```

The command above creates an alias that first activates your virtual env, and then runs the cadnano2 script. If successful, you should be able to open another new Terminal window and run one command to launch:

```
cn2
```

If you see an error, make sure that you didn't copy any special characters from the PDF.

Additional Instructions

Reporting Bugs

The **douglaslab/cadnano2** application is not bug free. You should get in the habit of saving your designs frequently, and reporting bugs when it crashes so we can make it more stable over time. Typically, you will get a Traceback in the **Terminal** window with some information about the line number and function that was executing when an exception was thrown.

If you encounter this situation, please copy the full traceback and paste it into new "Issue" along with detailed steps to reproduce at https://github.com/douglaslab/cadnano2/issues. If you have any example files that will be useful in reproducing the bug, please upload them too.

If you poke around in the source code and think that you've fixed the bug, please submit a pull request along with your Issue. If we incorporate your fix, you will be added to the list of contributors.

Updating Cadnano2 when it's been patched

If you submit any bugs, hopefully, we will be able to patch the source code quickly. Once that is done, you'll need to synchronize your local copy of **cadnano2** with the latest release. You can do so from the **Terminal** by activating your virtual env, and then performing an upgrade command:

```
source ~/virtualenvs/cn2/bin/activate
pip install —upgrade cadnano2
```