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 September 2018.

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://get.adobe.com/air/
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, and allows for easy python3 installation. (Optional: Homebrew can install GUI applications too! See: Cask)

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 simply typing:

```
python3
```

---

**Step 4. Set up a virtual environment**

We highly recommend using virtual environments for managing cadnano installations, and Python projects in general.

Why use virtual environments? Normally, when you install Python modules from the Terminal, you are directly changing your system-wide configuration. However, some programs have different installation requirements that cannot easily co-exist, for example two different release versions of the same program. Virtual environments mitigate this problem by creating self-contained folders to manage custom sets of Python modules, and providing a set of Terminal commands to easily switch between different configurations to make it seem like you’re just working in a normal environment. The approach helps protect your system configuration since you avoid modifying it directly. And, if a virtualenv configuration gets messed up, you can easily delete it and start over.

1. If you don’t already have virtual environments set up, first, run this command from the Terminal:

```
pip3 install virtualenvwrapper
```
Installing `virtualenvwrapper` will also install its dependencies, including the `virtualenv` package.

2. `virtualenvwrapper` is invoked as a shell script to temporarily set up your environment. Because virtual environments work directly on the **Terminal**, we need to make sure that it runs every time you open a Terminal window. We do so by adding some lines to your `.bash_profile` with the following command:

```bash
cat <<END >> ~/.bash_profile
export WORKON_HOME=$HOME/.venvs
VIRTUALENVWRAPPER_PYTHON=/usr/local/bin/python3
source /usr/local/bin/virtualenvwrapper.sh
END
```

3. **Optional**: If you’d like to set your command prompt so the style matches the coloring in this document, you can copy the first two lines from [https://gist.github.com/sdouglas/324d2a6e760a301c1d85](https://gist.github.com/sdouglas/324d2a6e760a301c1d85) into your `~/.bash_profile` using `nano` (or the editor of your choice):

```
nano ~/.bash_profile
```

4. Next, open a new **Terminal** window. If everything worked, you should see `virtualenvwrapper.sh` run for the first time and create 12 files. The next time you open the **Terminal** it should just look like the normal command prompt.
Step 5. Create a virtualenv and install Cadnano2 dependencies: PyQt5 and networkx

**PyQt5** is required to run Cadnano2, similar to Adobe AIR for Cadnano1. The **networkx** package is used to handle some of the graph algorithms for the original version of the Autobreak plugin. We won’t be using that feature, but it’s easier to just install the dependency for now. We will be using pip, a command-line python package installer.

1. Let’s create our first virtual environment in preparation for installing cadnano2:

   ```shell
   mkvirtualenv cn2
   ```

   The new virtualenv will automatically activate upon creation, and you should see its name in parentheses at the start of your prompt, something like “(cn2) username ~ $” if you are working in your home directory.

2. Now we can install the cadnano2 dependencies PyQt5 and networkx inside the “cn2” virtualenv that we just created. First install PyQt5 using the following command:

   ```shell
   pip install PyQt5==5.10.1
   ```

   The “==5.10.1” part forces the installation of a specific version of PyQt5. We have found that the latest version of PyQt5 (version 5.11.2, as of September 2018) can cause some runtime errors, but 5.10.1 should work fine.

3. Install networkx using the following command:

   ```shell
   pip install networkx
   ```

   The above commands from steps 2 and 3 should output messages about downloading and installing the corresponding packages, similar to what we saw above when installing virtualenvwrapper. If you see warnings about upgrading pip to the newest version, you can ignore them. Or, follow the instructions if you wish.

Step 6: Install Cadnano2 by cloning the source git repository

1. Finally, we can clone the latest version of cadnano2, and run it.

   ```shell
   git clone https://github.com/douglaslab/cadnano2
   cd cadnano2
   python3 main.py
   ```

   ![](image1.png)

   Hopefully, the cadnano2 window launched!
Step 7: Running Cadnano2 in a new Terminal window

Once you have your virtual environment set up, open a new Terminal and run the following:

```
workon cn2
cd ~/cadnano2
python3 main.py
```

If you installed or moved cadnano2 somewhere other than your home directory, you’ll need to adapt the command to navigate there.

Note that you only need to follow steps 5 and 6 once to setup the virtualenv and download the cadnano2 source. After that, you only need to activate or deactivate the cn2 virtualenv as needed. Note that the “python3 main.py” command will not work until you activate the cn2 virtualenv.

In general, to re-activate your cn2 virtual environment, you use the `workon` command, e.g.

```
workon cn2
```

To switch back to your normal non-virtual environment, just open a new Terminal or use the `deactivate` command:

```
deactivate
```

Step 8: Create a Cadnano2 alias

You may wish to add the following line to your `.bash_profile` to create an alias to run all three commands together:

```
alias cadnano2='workon cn2 && cd ~/cadnano2 && python main.py'
```

With the alias, you can just type the following command to launch the app:

```
cadnano2
```

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.
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 commit. You can do from the Terminal by navigating to the cadnano2 source path, and then performing a `git pull` command:

```
   cd ~/cadnano2
   git pull
```

Assuming you haven’t made any local changes to the source code in the master branch, any updates should download and you’ll see some output similar to the above screenshot. If there haven’t been any updates, you’ll see the message “Already up to date”.