# Pygame Installation

Pygame requires Python; if you don't already have it, you can download it from python.org. **Use python 3.7.7** or greater, because it is much friendlier to newbies, and additionally runs faster.

The best way to install pygame is with the pip tool (which is what python uses to install packages). Note, this comes with python in recent versions. We use the --user flag to tell it to install into the home directory, rather than globally.

```
python3 -m pip install -U pygame --user
```

To see if it works, run one of the included examples:

```
python3 -m pygame.examples.aliens
```

If it works, you are ready to go! If not there are more detailed, platform-specific instructions further down the page.

# Next steps...

You could try:

- Having a go at one of the tutorials.
- Or dive right into the pygame Docs

## Further information on installation

Hopefully the installation instructions above worked for you. If not, please read some alternative installation methods, and extra details below.

Wheels are available for x86 and x64 architectures on Linux and Windows, and for x64 on Mac. If pip doesn't find a wheel for your platform, it will try to compile pygame from source (see below).

pygame requires a "newer" pip. If pygame starts compiling from source and fails, **first try upgrading pip.**

# Raspberry Pi

This comes with pygame already installed on the default raspbian installation.

# Windows installation

Make sure you install python3.6 with the "Add python 3.6 to PATH" option selected. This means that python, and pip will work for you from the command line.

There is documentation with python for the "windows installation steps"

```
py -m pip install -U pygame --user
```

```
py -m pygame.examples.aliens
```

If you get 'PermissionError: [WinError 5] Access is denied', before starting the command prompt right click and "Run as administrator".

# Mac installation

## Recent versions of Mac OS X require pygame 2

If your examples aren't running and you are using a recent version of Mac OS X; try this line to install pygame instead:

```
python3 -m pip install -U pygame==2.0.0.dev6 --user
```

## virtualenv issue

There are problems with '

```
virtualenv
```

' and pygame on mac, where it loses focus. More details here: details on the virtualenv issue. It does work if you create a venv with python -m venv however. Instructions if you want to use an virtualenv (some people have trouble with the window losing focus when inside a venv).

```
# create a virtualenv called 'anenv' and use it.
python3 -m virtualenv anenv
. ./anenv/bin/activate
# venvdotapp helps the python be a mac 'app'. So the pygame wind
python -m pip install venvdotapp
venvdotapp
python -m pip install pygame

# See if pygame works with the oo module, and the aliens example
python -m pygame.examples.aliens
```

That's just needed on mac if you use virtualenv, not if you install it with --user, or if you use -m venv.

## Anaconda Mac

Use

```
pythonw
```

instead of

```
python
```

(note the w on the end). If it doesn't work for you, please see the /wiki/MacCompile instructions.

# Unix Binary Packages

Many Linux and BSD distributions have their own packages of Pygame. These may have an older version of Pygame than the wheels, but have been carefully tested with other packages in that distribution.

## Debian/Ubuntu/Mint

```
sudo apt-get install python3-pygame
```

```
python3 -m pygame.examples.aliens
```

## Fedora/Red hat

```
sudo yum install python3-pygame
```

```
python3 -m pygame.examples.aliens
```

### Arch/Manjaro

```
sudo pamac install python-pygame
```

```
python -m pygame.examples.aliens
```

| **Distribution** | **Python 3 package** | **Python 2 package** | Debian/Ubuntu | python3-pygame |
|---|---|---|---|---|

FreeBSD also has an active pygame package. While techinicaly it isn't binary, it is automatically built and installed by the ports manager. See the FreeBSD package page for more information. http://www.freebsdports.info/ports/devel/py-game.html

Gentoo has a builtin package for pygame. This is compiled for your system as it installs, similar to BSD, http://packages.gentoo.org/package/dev-python/pygame

# Installing From Source

## Compilation

Compiling and installing pygame is handled by Python's distutils. Pygame also comes with some scripts to automatically configure the flags needed to build pygame. Use the "setup.py" script to start the installation.

The first time you run the setup script, it will call the "`config.py`" script. This will build a "`Setup`" file which stores all the information needed to compile. The "config.py" will do a good job of detecting what dependencies are available and where they are located. If it isn't perfect, it is easy to build your own, or edit the created "`Setup`" text file. This "`Setup`" file is a simple Makefile-like text file. It defines variables needed to use each dependency, and then enables all the pygame modules with found dependencies. If you have trouble compiling, you should be able to easily fix any problems inside the "`Setup`" file.

Running the "`setup.py`" script will call distutils to build and install the pygame package. Distutils actually supports a wide variety of compile and install options. running "`python setup.py help`" will start to show you the different options available. You can change many things like install locations, compiler to use, and more. Calling the "setup.py" script with no arguments and it will just ask you if you want the default flags needed to compile and install.

Some of the .c files are generated by Cython from .pyx files. Running "`setup.py cython`" will update them.

## Windows Compiling Info

> You can compile pygame on windows with mingw (gcc for windows) and also with visual studio. Up to date details can be found here: CompileWindows

## Unix Compiling Info

> Compiling from linux shouldn't give you any problems. One thing you must keep in mind is that most linux RPM packages separate the actual library from the "dev" files needed to compile. To build you will need to make sure the packages like "`SDL-dev`" are installed.
>
> You can check to see if SDL is ready to be built from by running the command `sdl-config` and seeing if it is found. If the `sdl-config` script is not on the path (or you have more than one?) Set the environment variable `SDL_CONFIG` to its location.
> Sometimes you will have the SDL libraries installed in once location, and the other SDL libraries in another. This tricks the pygame config scripts, but you can help it out by setting the environment `LOCALBASE` to a path prefix where the other libraries are. The common case for this is SDL installed in /usr and other SDL libs installed in /usr/local. The command for this situation is "`LOCALBASE=/usr/local python setup.py install`".

## Mac OS X Compiling Info

Up to date instructions for compiling on Mac OS X can be found here: MacCompile