# Linux

Независимо от того, какая ОС Linux у вас стоит, я советую установить следующее приложение – Alacarte:

* Apt-get install alacarte

Если у вас стоит что-то более экзотичное, типа Red Hat, Fedora, то исспользуйте yum:

* Yum install alacarte

В зависимости от настроек Linux может потребоваться использовать sudo.

Для чего нужна alacarte? Всё довольно просто. В Linux не все приложения устанавливаются, как «приложения». Зачастую нет красивеньких значков, ярлычков, а иногда даже непонятно откуда запускать программу. Если вы используете Linux давно, то флаг, как говорится, в руки, можно запускать приложения через командную строку и пугать простых пользователей видом терминала со скачущими буковками. Но если вы недавно начали использовать Linux или, если вам просто лень каждый раз искать, где там лежит этот скрипт, или лень прописывать что-то, есть красивое решение, с теми самыми ярлыками, с возможностью запуска всего этого через меню с картинками.

Итак, установив alacarte, в системах типа Ubuntu (кстати, легендарная Kali тоже входит сюда), вы должны зайти в «Applications», затем Accessories, иногда этот подпункт меню может находиться в Usual Applications, там будет находиться Alacarte Menu Editor. В некоторых случаях надо нажать на Main Menu.

Начинаем редактирование (создание нового удобного способа запуска приложения). Нажмите либо на Add Item, либо на New Item. Появится окошко, требующее от вас задать параметры приложения. Прежде всего название Name, например, PyCharm или любое другое название для любого другого приложения…

Следующее немного сложнее – command. Это способ вызова необходимого нам приложения. Если неясно, а что там собственно делать, нажмите на кнопку Browse. Она выкинет вас в /usr/bin, где можно подсмотреть путь к нужному нам приложению. Если вдруг мы не в /usr/bin то просто и спокойно переходим туда. В правой части открывшегося окошка есть «быстрые кнопки», по которым можно перейти в нужное место. Скорее всего там будет такая «/». Это, читай, начало всех начал, в этом месте будет много разных папок, в том числе и та, столь нужная нам «/usr», внутри неё окажется bin. Далее внутри bin будет очень много разных названий. Загрузка этой папки может занять даже какое-то время. Как только загрузка закончилась, можно искать, например firefox. Тыкъ на нужное название и «OK». Всё – путь указан. Но иногда всё бывает немного сложнее. Например, в этом месте нет нашего любимого PyCharm.

A picture containing indoor, table

Description automatically generatedЗдесь у нас есть варианты. Например, мы качаем приложение, разархивируем его:

* tar xvzf file.tar.gz

Но терминал можно и не использовать, так как в большинстве современных Linux есть Archive Manager, который всё красиво откроет.

В папке приложения будет папка bin, в которой будет файл вида  что-то-там.sh. Чтобы не особо напрягаться, прописывая путь, можно просто перетянуть файл в терминал. Внимание!!! Если файл от терминала отскакивает, как горох от стенки, значит, вы не раскрыли архив, а просто просматриваете содержимое архива через Archive Manager.

Перетащили этот замечательный файл в терминал, нажали на кнопку Enter, откроется приложение и станет работать. На этом этапе можно взять и скопировать путь из терминала или перетащить его в наш Alacarte и всё будет работать. В целом, это вариант. Но у него есть минусы. Во-первых, скорее всего сейчас скачанное приложение «лежит» в папке загрузок, которую придётся чистить и не раз. Во-вторых, можно сделать лучше.

Для того, чтобы сделать лучше и красивее, много усилий не понадобится. Первое, перетащите или разархивируйте приложение в одну из папок. Лучше всего это сделать в папку /opt. Снова перейдите в bin и найдите там .sh файл. Укажите этот путь в Alacarte. Не пытайтесь перенести .sh файл в /usr/bin в приложении может быть слишком много файлов, путь к которым прописан локально – через окружающие папки. Последнее, создайте картинку-лого-ярлык, для этого кликните разок в картинку с замком или множеством точек в диалоговом окне Alacarte. Лого находится в той же bin-папке, где и находится .sh файл.

Фух, с Alacarte мы разобрались. С закачкой PyCharm и установкой через Alacarte больше проблем быть не должно.

Но нас ещё ждёт Anaconda. Поехали!

Перво-наперво, в Linux поддерживаются два Python, если вы наберёте в терминале что-то типа:

* python -V

Вы получите версию Python 2.X

Чтобы работать с Python 3, который используют все, который поддерживается и на котором надо учиться, вам надо набрать следующее:

* python3 -V

Менеджер пакетов pip, скорее всего уже установлен. Чтобы проверить его версию, надо набрать в терминале:

* pip3 -V

Anaconda можно скачать непосредственно с сайта или используя wget:

* wget <https://repo.continuum.io/archive/Anaconda3-XXXX-Linux-x86_64.sh>

Вместо XXXX должны быть цифры, обозначающие версию Anaconda. Ссылку на файл можно скопировать тут (ниже на странице будут версии Anaconda): <https://www.anaconda.com/distribution/> Здесь же можно просто скачать.

Не забудьте решить в какую папку вы будете качать Anaconda, это может быть папка загрузок или новая папка, которую вы сделали сами:

* cd path-to-folder

Так мы переходим в нужную папку.

* mkdir my-new-folder
* cd my-new-folder

Так мы создаём новую папку и переходим в неё. Если вы теперь запустите wget, Anaconda скачается в ту папку, в которой вы находитесь. Если вы воспользуетесь простой закачкой, Anaconda окажется в папке загрузок. Запустите файл:

* bash path-to-anaconda.sh

Установка займёт некоторое время в конце установки будет предложено добавить Anaconda в .bashrc. Пишите yes.

Если вы вдруг что-то пропустили, забыли, надо открыть .bashrc файл. Например так:

* atom .bashrc

Добавьте в файл (любое место в файле) следующую строку:

* export PATH=$ANACONDA\_INSTALL\_DIR/bin:$PATH

У меня, например, это выглядит так:

* export PATH=”/root/anaconda3/bin:$PATH”

Не забудьте сохранить файл. А теперь введите в терминале следующую строчку:

* source ~/.bashrc

Теперь посмотрим, как будет работать Python3:

* python3

В терминале и сразу enter. В описании в том числе будет написано Anaconda, Inc. on linux. Это победа. Теперь если открыть абсолютно новый терминал и написать:

* anaconda-navigator

У вас запустится соответствующее приложение. Здесь можно запускать и Jupyter Notebook, и Jupyter Lab. Хотя в anaconda уже установлены такие пакеты, как Pandas, NumPy и другие, просто ради теста попробуйте в новом терминале напечатать:

* conda install numpy

Чтобы проверить, какая Anaconda вам досталась, то есть какой в ней набор научных библиотек, просто напечатайте в терминале:

* conda list

На некоторое время вернёмся к PyCharm. Для корректной работы нам надо настроить интерпретатор. Прежде всего зайдите во вкладку View -> Appearance -> Toolbar. На последнее слово кликните один раз, так, чтобы рядом появилась галочка. Теперь в верхней части окна появится значок гаечного ключа. В некоторых случаях интерпретатор настроится сам. Но гаечный ключ нам всё равно понадобится для установки разных библиотек. Нажмите на этот значок, в открывшемся окне выберите название вашего проекта, выбирайте Project Interpreter. Нажмите на выезжающий список, нажмите на Show All, в новом окне нажмите на +, а затем укажите путь к интерпретатору. Чтобы понять, где лежит ваш Python, можно воспользоваться следующей хитростью:

* pip3 -V

Терминал покажет версию менеджера библиотек и пакетов, а также путь, по которому можно найти Python и pip внутри него. Я лично предпочитаю pip, но есть и фанаты conda. Некоторые пакеты можно ставить только через один менеджер, другие только через другой. Но это крайне редкие, высоко-специфичные пакеты, с которыми вы вряд ли столкнётесь. Пример такого пакета – RDKit, библиотека для химинформатики.

PyCharm обладает всеми необходимыми для разработки свойствами. Через него можно быстро и легко устанавливать пакеты, для работы с данными, типа массивов NumPy, можно использовать специальный режим среды разработки – Scientific Mode.

Далее я добавлю часть скриншотов сделанных на Mac, но большой разницы не будет:

Для установки новых библиотек/пакетов, нам нужно снова вернуться к настройкам (гаечный ключ, название проекта, интерпретатор и плюсик):

A screenshot of a computer

Description automatically generated

А теперь выбираем необходимый пакет:

A screenshot of a cell phone

Description automatically generated

Устанавливаем библиотеку/пакет. Иногда могут возникнуть ошибки. Изучите отчёт. Может не хватать какого-то компонента. Стоит установить: cython, wheel, flask; они могут быть необходимы для корректной работы NumPy. Изучите отчёт по ошибке, скопируйте его, просмотрите в текстовом редакторе. Скорее всего с такой проблемой кто-то уже сталкивался, решения зачастую можно найти на StackOverFlow. Библиотеки, которые разрабатывают отдельные энтузиасты, обычно имеют либо свою страничку (верхний скрин-шот, справа, текст в синем), либо такую же на github. В случае ошибки, можно спросить автора в соответствующем раз деле.

Почти всё. Теперь посмотрим, как работает Scientific Mode. Я пока только импортировала NumPy, но среда разработки уже предлагает использовать научный режим:

A screenshot of a computer screen

Description automatically generated

См. правый нижний угол. Научный режим позволяет мне посмотреть полученный массив (матрицу) без вывода результата в консоль:

A screenshot of a cell phone

Description automatically generated

А теперь посмотрим то же, но в обычном режиме, в консоли ничего нет:

A screenshot of a computer screen

Description automatically generated

Результат появляется только тогда, когда я добавляю print():

A screenshot of a computer

Description automatically generatedСупер коротко, как установить необходимые библиотеки через терминал. Так как Anaconda, это научный дистрибутив, большая часть необходимого находится уже там, чтобы проверить, что есть, в терминале вводим команду:

* conda list

Скорее всего там уже будут необходимые нам библиотеки: pandas, NumPy, MatPlotlib, но при необходимости установим пакеты так:

* conda install numpy
* conda install pandas
* conda install matplolib

Если используется pip, делаем так:

* pip install pandas numpy matplotlib

При одновременной установке пакетов не разделяем названия запятыми, только пробелами!