Сообщить об ошибке.

Когда следует переходить на новую версию Python

Короткий ответ на вопрос "- Стоит ли переключаться на новую версии Python после ее выпуска?" - это конечно НЕТ. Чтобы понять, почему, необходимо рассмотреть процесс упаковки Python, разработки программного обеспечения и взглянуть на историю прошлых выпусков.

Проблемы с новым основным выпуском Python.

Как и во многих проектах с открытым исходным кодом, интерпретатор Python, библиотеки языка Python и большинство наборов инструментов и пакетов, основаны на добровольном труде. Ничто из нижесказанного не должно восприниматься как жалоба на людей, занимающихся обслуживанием, они делают чрезвычайно ценную работу бесплатно, и на все требуется время. Кроме того, выпуск обновлений занимают еще больше времени, когда в них участвует очень много разных групп.

Имея это в виду, рассмотрим проблемы с использованием новых версии Python в последующие дни за релизом, а так же в последующие месяцы.

Отсутствующие образы Docker

Как оказалось, для финальной версии нового Python не всегда присутствует образ Docker.

```
# символ `x` в версии Python означает номер новой версии
$ docker run -it python:3.x
# Unable to find image 'python:3.x' locally
# docker: Error response from daemon:
#         manifest for python:3.x not found:
#         manifest unknown.
```

Если хотите просто попробовать новый функционал только-что вышедшего Python, то можно использовать образ кандидата на выпуск, например Python 3.x-rc.

Конечно-же, образ Docker новой версии Python будет доступен в ближайшие несколько дней, но на раннем этапе даже установка новой версии может Python быть затруднена. И это только ПЕРВАЯ ПРОБЛЕМА.

Отсутствующие бинарные пакеты.

Еще одна проблема заключается в том, что еще не все сторонние модули/пакеты, от которых зависит проект, поддерживают новую версию Python. Например, psycopg2-binary обеспечивает привязку к базе данных PostgreSQL и если пытаться установить его на новую, только-что вышедшую версию Python, то все пойдет не так:

```
# символ `x` в версии Python означает номер новой версии
$ docker run python:3.x-rc-slim pip install psycopg2-binary
# ...
#    Error: pg_config executable not found.
#     pg_config is required to build psycopg2 from source.
# ...
```

Что происходит?

Как правило, сопровождающие пакетов Python загружают в PyPI скомпилированные версии своих пакетов, известные как "wheels". Когда устанавливается пакет, то происходит загрузка бинарного файла "wheel", который не нужно собирать и компилировать (если, при этом, не используется Alpine Linux).

Загрузка wheels-пакетов происходит не так скоро и многие пакеты, на ранней стадии, не имеют скомпилированных бинарных файлов для нового Python.

Следует отметить, что дела на этом фронте, улучшаются: в выпуске Python 3.9 у NumPy и Pandas не было доступных бинарных файлов вскоре после выпуска. Но с выходом Python 3.10, пакет NumPy выпустили на следующий день, при этом он был ограничен Linux и в основном предназначался для целей тестирования. Поскольку Pandas зависит от NumPy, то он не будет работать ни в Windows, ни в macOS, пока не будет выпущен пакет NumPy для этих операционных систем.

Есть много других проектов, в которых будут отсутствуют скомпилированные файлы. Другой случайный пример, matplotlib (27 миллионов загрузок в месяц из PyPI) также не имеет wheels-файла на ранней стадии.

Важно: убедитесь, что pip перед установкой пакетов обновлен, иначе есть возможность не получить последние бинарные файлы в Linux, а менеджер пакетов pip будет пытаться собрать его из исходного кода.

Несовместимые пакеты.

Как говорилось выше, достаточно просто самостоятельно перекомпилировать код нужного модуля или пакета, что бы он заработал с новым Python. Однако в других случаях, исходный код стороннего модуля/пакета может быть несовместим, например, ввиду окончания поддержки новым Python устаревших функций/модулей, которые используются в пакете, и следовательно, такой пакет требует некоторой доработки.

Учитывая, что многие пакеты Python поддерживаются добровольцами с ограниченным свободным временем, требуется время, чтобы обновления проникли в систему.

Ошибки в новых версиях Python на ранней стадии.

Например, версия Python 3.x.1 обычно выпускается двумя месяцами позже с длинным списком исправлений. Конечно, в будущем всегда будет еще не один выпуск с исправлением ошибок, но, учитывая 12-месячное окно между основными выпусками, будет написано много нового кода, но он не будет так широко использоваться.

Другими словами, ошибки или нестандартное поведение широко используемого функционала языка, тестируется и исправляется с выходом версии Python 3.x.1. После чего можно попробовать переключить свой проект на новую версию Python.

Отсутствие поддержки набора инструментов.

Новые версии Python часто имеют новый синтаксис, например, это относится к Python 3.10: можно использовать сопоставление структурных шаблонов. Однако другие инструменты (IDE) также должны поддерживать новый синтаксис, например автоформатеры, линтеры и т. д. Пока они этого не сделают, нельзя использовать новый синтаксис без, например, подсказок/подсветки синтаксиса в IDE.

Какую версию использовать?

Учитывая, что для обновления требуется работа, связанная с дополнительным тестированием, некоторой настройкой кода, может возникнуть соблазн отложить обновление версий Python на неопределенный срок. Зачем беспокоиться о несовместимости, новых версиях и многом другом, когда можно тупо придерживаться своей текущей версии на неопределенный срок?

Проблема в том, что Python определенной версии не поддерживается бесконечно, и библиотеки не поддерживают все версии Python бесконечно. Поэтому, если работать с версией Python пятилетней давности, то переключение становится большой проблемой, так как изменения будут более значительными как между версиями как в Python, так и в библиотеках одновременно. Что делает обновление пугающим. Но рано или поздно придется переключиться на новую версию, из за возможной эксплуатации дыр в безопасности неподдерживаемой, устаревшей версии Python.

К тому же дистрибутивы Linux, перестанут поставлять старую версию Python в качестве системной по умолчанию, что затруднит переносимость приложения между устройствами.

Вместо одного страшного масштабного обновления каждые несколько лет, гораздо безопаснее иметь непрерывный процесс небольших обновлений. Каждый раз, когда выходит новая основная версия Python или новая основная версия библиотеки, подождите выхода версии 3.x.1, а затем переключитесь.

Содержание раздела:

- ОБЗОРНАЯ СТРАНИЦА РАЗДЕЛА
- Выбираем разрядность Python3 для Windows

- Установка Python3 на ОС Windows
- Установка Python3 на Ubuntu (Debian) из репозиториев
- Выбираем разрядность Python3 для Linux
- Установка Python 3.x из исходников на ОС Linux
- Запуск и использование интерпретатора Python
- Установка и использование PyPy3, совместимость с Python3
- Когда следует переходить на новую версию Python