

PYTHON

NAUKA PROGRAMOWANIA DLA KAŻDEGO

OPANUJ PYTHONA OD PODSTAW:

od procesu instalacji
po korzystanie z biblioteki
standardowej

ALTERNATYWNE ŚRODOWISKA PROGRAMISTYCZNE

KLUCZOWE UMIEJĘTNOŚCI: napisy i sekwencje

Własny **CHATGPT**
w Pythonie

Tworzenie gier w Pythonie:
PONG, TETRIS
ISPACE INVADERS!

Generowanie grafiki AI
z poziomu **PYTHONA**

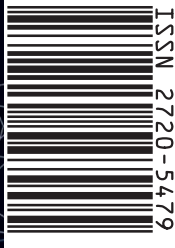
PYXEL: Kombajn
do tworzenia gier retro

Aplikacja mobilna
z backendem
w **PYTHONIE**

FULL STACK:
Kompletna aplikacja
webowa krok po kroku



Numer 2/2023
49,90 zł (8% vat)



DLA POCZĄTKUJĄCYCH:

Sterowanie
przebiegiem
programu:

**INSTRUKCJE
WARUNKOWE
I PĘTLE**

Efektywne
zarządzanie
kodem:

**FUNKCJE
I WYJĄTKI**

Operacje
na plikach:
**ODCZYT
I ZAPIS
DANYCH**

Pierwsze wyzwanie
programistyczne:
FIZZBUZZ

**STRUKTURY
DANYCH:**
krotki, słowniki
i zbiory

Pracujesz w środowisku linuksowym?

- ✓ Chcesz być na bieżąco z nowościami?
Potrzebujesz wsparcia?
- ✓ A może szukasz łatwego i stałego
dostępu do aktualnych treści związanych
z Linuksem i technologiami open source?

Zamów już dziś prenumeratę światowego bestsellera „Linux Magazine” i miej pewność, że najnowszy numer trafi prosto do Ciebie.



Z nami poszerzysz swoje zawodowe kompetencje:

✓ Skonfigurujesz.

Skonfigurujesz serwery udostępniające różnego rodzaju usługi

✓ Będziesz na bieżąco.

Będziesz na bieżąco z zagrożeniami i metodami zabezpieczania się przed nimi.

✓ Poznasz.

Poznasz nowoczesne rozwiązania służące do monitorowania systemów i sieci.

✓ Dowiesz się.

Dowiesz się jak zoptymalizować funkcjonowanie systemu pod kątem wydajności.

✓ Lepiej zrozumiesz.

Lepiej zrozumiesz działanie Systemd i innych kluczowych komponentów systemu.

✓ Nauczysz się.

Nauczysz się jak działają łańcuchy Markowa i w jaki sposób uczy się sztuczna inteligencja.

✓ Zapoznasz się.

Zapoznasz się z najciekawszymi i najnowszymi rozwiązaniami open source.

Dołącz do grona prenumeratorów Linux Magazine i korzystaj z atrakcyjnych zniżek i promocji dostępnych dla Czytelników.

Skontaktuj się z naszym specjalistą, który zaproponuje Ci najlepszą ofertę prenumeraty: **+48 22 518 29 29** lub zajrzyj na naszą stronę: **<https://linux-magazine.pl/>**

LINUX
MAGAZINE

Co w numerze?

Zapraszamy do drugiego wydania naszego magazynu poświęconego Pythonowi – potężnemu i uniwersalnemu językowi programowania.

Kiedyś, kiedy każdy mikrokomputer był wyposażony w interpreter języka programowania, listingi zaś mozolnie przepisywało się z magazynów komputerowych znak po znaku, nierzadko klnąc pod nosem z powodu drobnych błędów, programowanie było czymś, z czym chcąc nie chcąc, stykał się każdy użytkownik komputera – choćby po to, by móc załadować grę. Nierzadko jednak zainteresowanie grami przekształcało się w chęć dowiedzenia się, jak to wszystko działa – np. po to, by móc stworzyć własne gry. Nie było to jednak proste, trzeba było bowiem całą pracę wykonać samemu, dostępne zaś źródła były dość skąpe.

Dziś sytuacja wygląda diametralnie inaczej. Do dyspozycji mamy potężne narzędzia, takie jak Unity czy Unreal Engine, które umożliwiają tworzenie zaawansowanych gier przewyższających grafiką to, co można było sobie wyobrazić w latach 80. czy nawet 90. ubiegłego wieku. A mimo to obserwujemy wzrost popularności gier 2D takich jak Deltarune czy Celeste, które wręcz epatują swoją surową „pikselowością”, udowadniając, że hiperrealizm to nie jedyny kierunek rozwoju współczesnych gier.

Niniejsze wydanie „Linux Magazine Poleca”, podobnie jak poprzednie, składa się z dwóch części. Pierwsza to wprowadzenie do języka Python – niezbędne, by móc zrozumieć kolejne artykuły. Omawiamy poszczególne zagadnienia krok po kroku, ilustrując każde przykładami w taki sposób, by wyeliminować wszystkie możliwe wątpliwości. Jednocześnie zachęcamy Czytelników do przetestowania zdobytej wiedzy na różnych platformach online,

takich jak Codewars, by stale pogłębiać swoją znajomość języka jako takiego oraz algorytmiki. Dotyczy to zwłaszcza tych wszystkich, którzy myślą o karierze programisty.

Druga część magazynu poświęcona jest zagadnieniom praktycznym i składa się z 3 sekcji. Zaczynamy od tworzenia gier, wykorzystując proste w użyciu biblioteki, takie jak Pygame czy Pyxel. Pokażemy, jak o własnych siłach stworzyć kultowe gry, które towarzyszyły pokoleniu młodych graczy kilkadziesiąt lat temu – Pong, Tetris i Space Invaders.

W drugiej sekcji zajmiemy się programowaniem backendu. Ponieważ Python nie istnieje w próżni, pokażemy, jak w praktyce odbywa się interakcja między pythonowym backendem (Flask) a javascriptowym frontendem (React). Następnie pójdziemy o krok dalej i rozwiniemy ten sam paradygmat na aplikacje mobilne, wykorzystując Flaska do zbudowania backendu aplikacji mobilnej.

Wreszcie w ostatniej sekcji skupiamy się na coraz bardziej popularnej ostatnio sztucznej inteligencji, a zwłaszcza na jej dwóch aspektach: generowaniu tekstu i obrazków. Pokażemy między innymi, jak z poziomu Pythona komunikować się z API OpenAI, tworząc własnego kлона ChatGPT oraz w jaki sposób wygenerować obrazy powojennej Warszawy pędzlem wielkich malarzy.

Mamy nadzieję, że w tym numerze każdy Czytelnik znajdzie coś dla siebie, a zamieszczone tu przykłady posłużą za inspirację do stworzenia własnych, bardziej zaawansowanych i rozbudowanych projektów. ■■■

Uwaga

Ponieważ ze względów edukacyjnych przykłady przedstawione są w najprostszej i najkrótszej wersji, w wielu miejscach musieliśmy odejść od reguł sztuki. Po pierwsze, w kodzie stosowanym w praktyce wszystkie nazwy funkcji, zmiennych, klas itd. powinny być znaczące. Używanie jednoliterowych nazw zmiennych – poza specyficznymi przypadkami – to komplikowanie życia i sobie, i innym, którzy będą czytali nasz kod. Po drugie, pisząc programy w Pythonie, powinniśmy stosować się do specyfikacji przedstawionej w PEP8 [2] odnoszącej się do formatowania kodu. Ze względu na ograniczenia dotyczące miejsca w wielu przykładach

byliśmy zmuszeni do złamania zawartych w nich reguł, nie ma jednak powodu, by się do nich nie stosować, pisząc własny kod. I po trzecie, jak krótko wspomnieliśmy przy okazji opisu błędów i wyjątków, pisząc prawdziwy kod, musimy zawsze sprawdzać wszelkie możliwe scenariusze niepowodzeń; jest to obowiązkowe zwłaszcza podczas operacji wejścia/wyjścia. Brak kontroli w tym przypadku to poważny błąd, który prędzej czy później doprowadzi do awarii programu. Jednak w przykładach musieliśmy zrezygnować z kontroli błędów, ponieważ kod musiałby kilkukrotnie zwiększyć swoją objętość.

Linux Magazine jest miesięcznikiem specjalistycznym wydawanym na licencji Linux New Media USA, LLC, we współpracy z Computec Media GmbH, Fürth, Niemcy.

Wydawca Wiedza i Praktyka Sp. z o.o.

Redaktor Naczelny: Artur Skura, askura@linux-magazine.pl

Wydawca: Patrycja Oleksiejuk

Kierownik grupy tematycznej: Szymon Danowski

Korespondenci i współpracownicy:

Erik Bärwaldt, Chris Binnie, Zack Brown, Bruce Byfield, Karsten Günther, Marcel Hilzinger, Klaus Knopper, Christoph Langner, Jeff Layton, Martin Loschwitz, Patrick Neef, Dimitri Popov, Thorsten Scherf, Ferdinand Thommes

Opracowanie graficzne, skład i przygotowanie do druku

Raster studio, Norbert Bogajczyk, studio@rasterstudio.pl

Projekt okładki: Magdalena Huta

Reklama: reklama@linux-magazine.pl

Licencje korporacyjne, rozszerzone i niestandardowe

tel.: +48 22 429 43 05

e-mail: prenumerata@linux-magazine.pl

Zamówienia i obsługa prenumeraty:

tel.: +48 22 518 29 29

faks: +48 22 617 60 10

prenumerata@linux-magazine.pl

Linux Magazine

ul. Łotewska 9a, 03-918 Warszawa

www.linux-magazine.pl,

tel.: +48 22 429 43 05, faks: +48 22 617 60 10

Wydawca dokłada wszelkich starań, aby publikowane w piśmie i na towarzyszących mu nośnikach informacje i oprogramowanie były poprawne i przydatne, jednakże Wydawca nie ponosi odpowiedzialności za efekty wykorzystania ich, w tym nie gwarantuje poprawnego działania programów.

Żaden z materiałów opublikowanych w Linux Magazine Poleca nie może być powielany w jakiegokolwiek formie bez zgody Wydawcy. Właścicielem znaku towarowego Linux jest Linus Torvalds.

ISSN 1732-1263; Nakład 6000 egz.

Nr rejestrowy BDO: 000008579

PODSTAWY

6 Instalacja

Aby móc zacząć programować w Pythonie, musimy najpierw pobrać i zainstalować interpreter tego języka, co zajmuje nie więcej niż kilka minut.

8 Środowisko programistyczne

Domyślny wiersz poleceń Pythona i środowisko IDLE mogą się wydawać nieco spartańskie i nieprzystające do wymagań współczesnych użytkowników. Powstało więc wiele projektów z alternatywnymi środowiskami programistycznymi (IDE) umożliwiającymi łatwiejsze i bardziej wydajne tworzenie kodu w tym języku.

10 Zamiast kalkulatora

Jednym z najprostszych sposobów korzystania z Pythona jest... użycie go w roli kalkulatora. Tak, ten potężny język programowania jest również szybką maszyną obliczeniową – w dodatku bardzo wygodną w obsłudze. Niektórzy zawsze mają otwarte w tle okno z Pythonem na wypadek, gdyby trzeba było coś szybko obliczyć.

13 Wyświetlanie na ekranie

Jedną z najbardziej podstawowych instrukcji Pythona jest `print()`, dzięki której wyświetlimy na terminalu dowolne dane. Instrukcja ta jest używana między innymi do wyświetlania informacji diagnostycznych.

16 Napisy

Napisy, czyli łańcuchy znakowe, to jeden z podstawowych typów używanych w Pythonie, zatem umiejętność postępowania się nimi jest absolutnie niezbędna dla każdego programisty Pythona.

19 Sekwencje

Napisy to tzw. typy sekwencyjne. Składają się one z elementów ułożonych w określonym porządku – kolejność ma znaczenie. Na wszystkich typach sekwencyjnych można przeprowadzać te same operacje. Warto je zapamiętać, ponieważ są one bardzo często używane w codziennej pracy.

21 Warunki

Praktycznie każdy nietrywialny program w Pythonie zawiera instrukcje sterujące. Ich szczególnym przypadkiem są instrukcje warunkowe takie jak `if` i `else`.

23 Listy

Listy to uniwersalny typ danych składający się z elementów jednego lub różnych typów, przy czym kolejność ich ułożenia ma znaczenie. Praca z listami w Pythonie to czyta przyjemność.

27 Pętla `for`

Pętla `for` jest jednym z podstawowych narzędzi każdego programisty Pythona. Niektórzy twierdzą, że gdyby się uprzeć, każdy program można by zapisać w postaci kombinacji pętli `for`.

29 Krotki

Krotka to sekwencyjny typ danych, przypominający nieco listę. Z matematycznego punktu widzenia krotka przypomina zbiór, istnieją jednak dwie zasadnicze różnice: w krotkach elementy mogą się powtarzać i ich kolejność ma znaczenie.

31 Słowniki

Kolejnym podstawowym typem danych używanym w Pythonie są słowniki, znane w innych językach jako „tablice asocjacyjne”. Składają się z dwuelementowych par, z których jedna jest nazywana kluczem, a druga – wartością. W danym słowniku wszystkie klucze muszą mieć unikatowe wartości.

33 Zbiory

Zbiór to typ danych składający się z elementów, które mogą być różnego typu, przy czym elementy te nie mogą się powtarzać. W przeciwieństwie do list kolejność nie ma znaczenia.

35 Funkcje

Funkcje umożliwiają wydzielenie logicznych fragmentów kodu do ponownego wykorzystania. Do tej pory korzystaliśmy z wielu funkcji wbudowanych, napisanych przez twórców Pythona – czas nauczyć się tworzyć własne.

38 Fizzbuzz

Pierwotne znaczenie słowa „fizzbuzz” to rodzaj zabawy matematycznej dla dzieci, która polega na tym, że wszyscy uczestnicy wymieniają kolejne liczby naturalne z trzema zastrzeżeniami: (1) jeśli dana liczba jest podzielna przez 3, uczestnik zabawy zamiast jej nazwy mówi „fizz”, (2) jeśli jest podzielna przez 5, uczestnik mówi „buzz”, (3) jeśli jest podzielna i przez 3, i przez 5, uczestnik mówi „fizzbuzz”.

40 Wyjątki

Jeśli coś może pójść źle, to prędzej czy później to nastąpi. W Pythonie rozróżniamy dwa rodzaje takich sytuacji: błędy składniowe oraz wyjątki. Te pierwsze musimy naprawić, te drugie – obsłużyć.

42 Operacje na plikach

Odczytywanie danych z plików i zapisywanie ich to podstawowa umiejętność programisty. Przydaje się też w codziennej pracy w zawodach niezwiązanych z programowaniem, zwłaszcza w sytuacjach, kiedy mamy do czynienia z większą liczbą plików bądź operacji zapisu/odczytu.

44 Moduły

Python zawiera wiele bardzo przydatnych funkcji wbudowanych. Ale to nie wszystko: możemy korzystać z wielu innych funkcji znajdujących się w modułach, czyli bibliotekach funkcji. Wraz z Pythonem dostarczana jest biblioteka standardowa (Python Standard Library) zawierająca całe moduły z wieloma przydatnymi funkcjami.

GRY**46 Pierwsza gra w Pythonie**

Pygame to bodaj najpopularniejsza platforma do tworzenia gier w Pythonie. Pokażemy, jak w prosty sposób wykorzystywać ją do napisania kłona kultowej gry Space Invaders i przećwiczyć korzystanie z klas Pythona.

51 Pong

Każda epoka ma gdzieś swój początek. Za początek epoki gier komputerowych uważa się powstanie gry Pong, w którą grano jeszcze na tradycyjnych, czarno-białych telewizorach.

55 Tetris

Kolejną grą, którą napiszemy, będzie Tetris. Jest to druga najpopularniejsza gra wszech czasów – przez długi czas dzierżyła palmę pierwszeństwa i dopiero kilka lat temu została zdetronizowana przez Minecrafta.

61 Pyxel

Gry retro przeżywają swój renesans. Pokażemy, jak je tworzyć, używając pythonowej biblioteki Pyxel.

BACKEND**65 Pełny stos**

Popularne określenie „full stack” odnosi się do pełnego projektu obejmującego zarówno część widoczną dla użytkownika (frontend), jak i to, co jest przed nim ukryte (backend). Pokażemy, jak stworzyć prostą, lecz kompletną aplikację, wykorzystując Flask jako backend i React jako frontend.

73 Aplikacja mobilna

W poprzednim artykule napisaliśmy aplikację składającą się z backendu i frontentu. Tym razem pokażemy, jak przekształcić ją w aplikację mobilną.

SZTUCZNA INTELIGENCJA**77 Własny ChatGPT**

Korzystanie z API OpenAI z poziomu Python jest bardzo proste i daje nam możliwości, których nie oferuje zwykły interfejs webowy ChatGPT.

80 Grafika z komputera

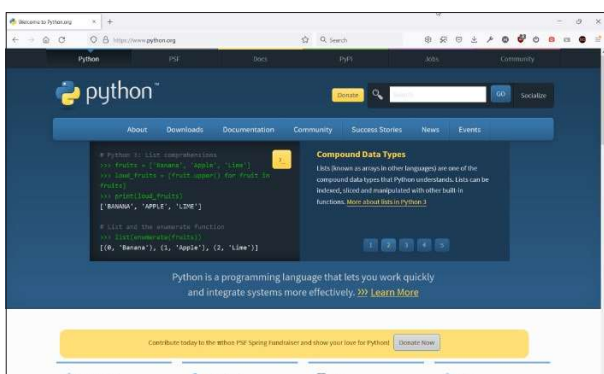
Tworzenie grafiki AI za pomocą Pythona jest prostsze niż kiedykolwiek.

Instalacja

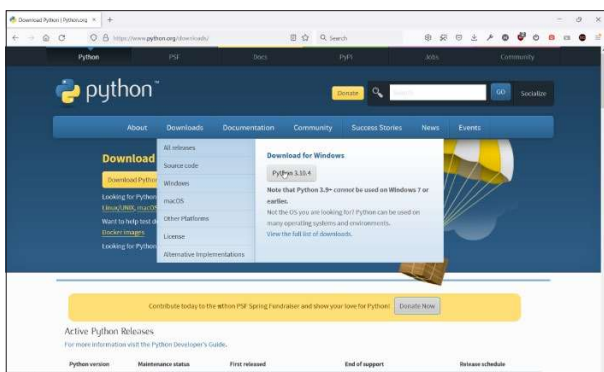
Aby móc zacząć programować w Pythonie, musimy najpierw pobrać i zainstalować interpreter tego języka, co zajmuje nie więcej niż kilka minut.

Windows

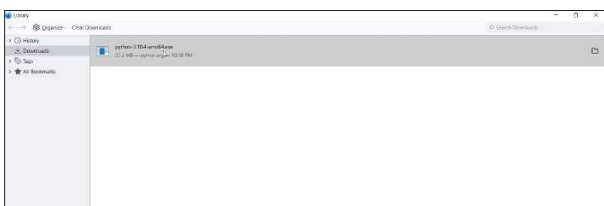
1. Wchodzimy na witrynę python.org



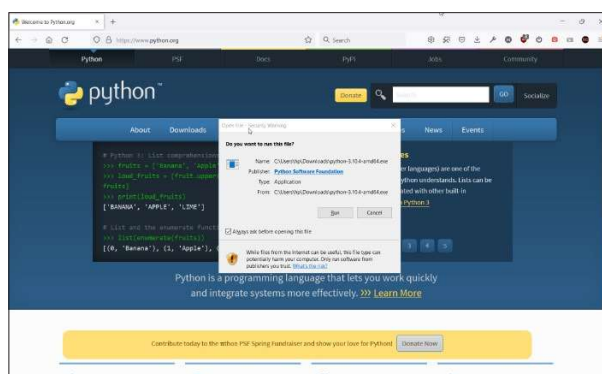
2. Klikamy zakładkę *Download* i duży przycisk z napisem *Python* oraz numerem wersji – jest to najnowsza stabilna wersja Pythona.



3. Klikamy pobrany plik wykonywalny.



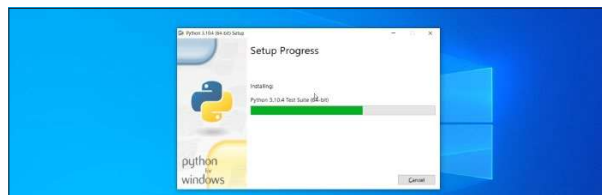
4. Windows może nas zapytać, czy na pewno chcemy go uruchomić – potwierdzamy.



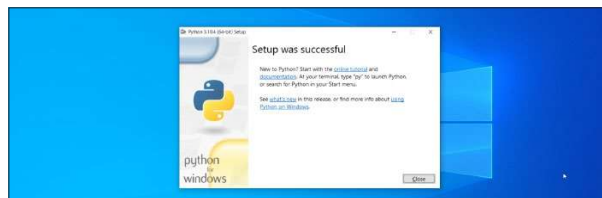
5. Pojawi się okno instalatora – zatwierdzamy domyślne ustawienia i klikamy *Install now*.



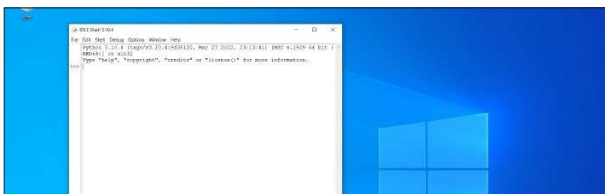
6. Instalator rozpocznie pracę, kopiując niezbędne pliki i przeprowadzając wstępną konfigurację.



7. Po zakończeniu całego procesu powinniśmy ujrzeć ekran informujący nas, że cały proces przebiegł pomyślnie.



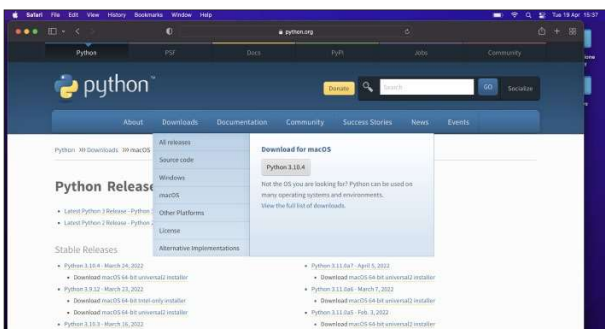
8. Python został zainstalowany – w menu startowym znajdziemy dwie pozycje: Python oraz IDLE. Ta pierwsza to wiersz poleceń Pythona, czyli okno, w którym możemy wpisywać polecenia Pythona, interpreter je od razu wykona. Natomiast IDLE to nieco bardziej rozbudowane środowisko, w którym możemy nie tylko wpisywać wiersz po wierszu kod Pythona, ale również tworzyć skrypty napisane w tym języku, edytować je i uruchamiać.



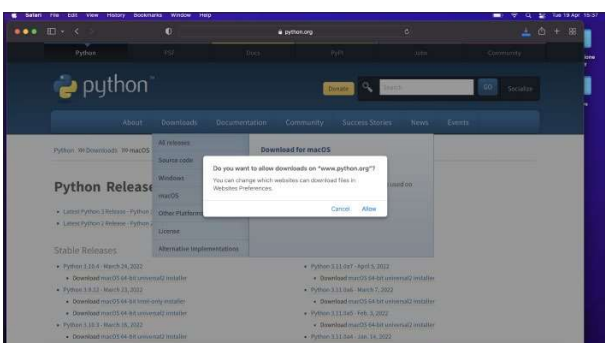
macOS

Do niedawna do macOS-a dołączana była starsza wersja Pythona (z linii 2.x), która nie jest zbyt przydatna do codziennej pracy, zaś w Monterey Apple ostatecznie usunął ją z systemu. Dlatego najprościej jest zainstalować najnowszą wersję Pythona bezpośrednio z witryny projektu.

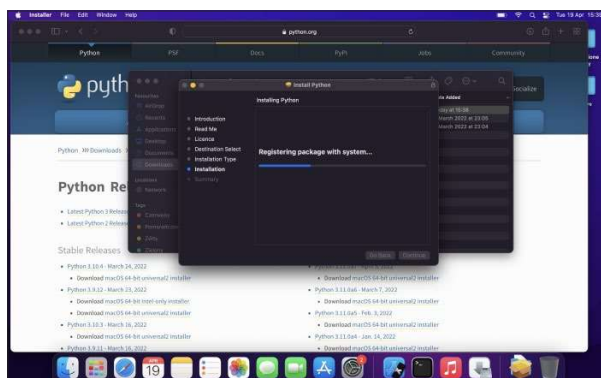
W tym celu wchodzimy na stronę Python.org i klikamy zakładkę *Downloads* – witryna powinna automatycznie rozpoznać nasz system i zaproponować pobranie najnowszego wydania Pythona.



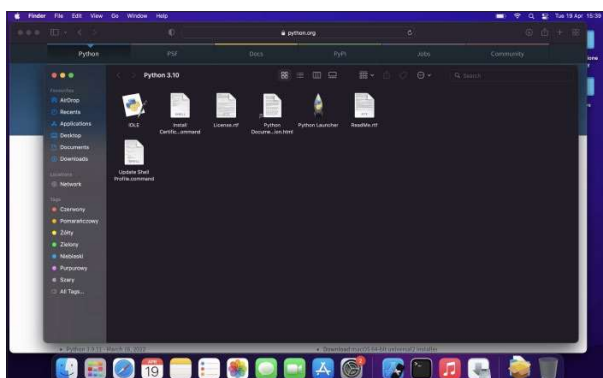
Może być konieczne zezwolenie na pobieranie plików z witryny Python.org.



Następnie instalator skopiuje wszystkie wymagane pliki w odpowiednie miejsca i przeprowadzi pozostałe wymagane czynności.



Po instalacji ujrzymy folder, w którym znajdziemy IDLE oraz Python Launcher – niewielkie narzędzie, które ułatwia konfigurację uruchamiania skryptów Pythona podwójnym kliknięciem.



Uwaga: jeśli używamy jednej z wcześniejszych wersji macOS-a lub aktualizowaliśmy go, w systemie nadal może być obecny starszy Python 2.7 instalowany przez Apple. Możemy to łatwo sprawdzić, wydając w Terminalu polecenie *python* i sprawdzając numer wersji. Jeśli jest to 2.7, powinniśmy uruchamiać Pythona poleceniem *python3*.

Linux

Dobra wiadomość: obecnie Python instalowany jest domyślnie w wielu popularnych dystrybucjach Linuksa – najprawdopodobniej więc nie musimy w ogóle nic robić. Jeśli jednak z jakiegoś powodu nie znajdziemy go w systemie, wystarczy zainstalować go za pomocą menedżera pakietów, tak jak dowolny inny program. Np. w Ubuntu zrobimy to poleceniem *sudo apt install python*.