به نام او

آموزش برنامه‌نویسی

PYTHON 3.7

راهنمای مبتدی‌ها در برنامه‌نویسی، علم داده، و توسعه وب با پایتون ۳.۷
(ویرایش دوم)

فابریزیو رومانو

ترجمه: مهندس حسین یعسوبی
انتشارات پندار پارس

سرشناسه	:	رومانو، فابریتسیو، ۱۹۷۵- م. Romano, Fabrizio, 1975-
عنوان و نام پدیدآور	:	آموزش برنامه نویسی PYTHON ۳.۷: راهنمای مبتدی‌ها در برنامه‌نویسی، علم داده، و توسعه وب با پایتون ۳.۷ (ویرایش دوم)/ فابریزیو رومانو؛ ترجمه حسین یعسوبی.
مشخصات نشر	:	تهران : پندار پارس، ۱۳۹۹.
مشخصات ظاهری	:	۴۹۰ ص.: مصور.
شابک	:	978-600-820197-7
وضعیت فهرست نویسی	:	فیپا
یادداشت	:	عنوان اصلی: Learn Python programming, second edition : the no-nonsense, beginner's guide to programming, data science, and web development with Python 3.7,2 nd ed,2018.
عنوان دیگر	:	راهنمای مبتدی‌ها در برنامه‌نویسی، علم داده، و توسعه وب با پایتون ۳.۷ (ویرایش دوم).
موضوع	:	پایتون (زبان برنامه‌نویسی کامپیوتر)
موضوع	:	Computer program language (Python)
شناسه افزوده	:	یعسوبی، حسین، ۱۳۵۲ -، مترجم
رده بندی کنگره	:	QA۷۶/۷۳
رده بندی دیویی	:	۱۳۳/۰۰۵
شماره کتابشناسی ملی	:	۷۴۰۵۴۷۴
وضعیت رکورد	:	فیپا

نام کتاب : **آموزش برنامه‌نویسی Python 3.7**

ناشر : **انتشارات پندار پارس**

تالیف : **فابریزیو رومانو (Fabrizio Romano)**

ترجمه : **حسین یعسوبی**

چاپ نخست : **دی‌ماه ۹۹**

شمارگان : **۲۰۰ نسخه**

طرح جلد : **رامین شکرالهی**

چاپ، صحافی : **روز**

قیمت : **۱۳۰.۰۰۰ تومان** **شابک : ۷-۹۷-۸۲۰۱-۶۰۰-۹۷۸**

تقدیم به

بنیان کنگره ۶۰

اندیشمند فرزانه، جناب آقای مهندس حسین دژاکام

و

معلم خوبی‌ها

جناب آقای مهندس آرش قربانی

سخن مترجم (ناشر)

هر بار که کتابی را برای ترجمه یا تألیف به دست می‌گیرم، در میانه راه، به خود می‌گویم این آخرین باری است که این کار را می‌کنم. چراکه یکی از سخت‌ترین کارهای موجود است. به‌ویژه اگر هدف، ارائه کاری با کیفیت درخور باشد. اما پس از به چاپ رسیدن اثر، گویی فرزند نداشته‌ات را در آغوش می‌گیری و در چشمانت می‌نگرد و تو هم نوازشش می‌کنی. نکته‌اش در این است که این نوزاد نابکار، زیر لب به تو می‌گوید که از هم اینک در اندیشه خلق خواهر و برادرم باش و به هیچ وجه، به مشکلات این کار نمی‌اندیشد!

به هر روی، سال ۹۹ که دنیا درگیر بیماری کرونا بود، و کسب و کارهای زیادی همچون ناشران، با کاهش تولید روبه‌رو بودند، دست‌کم برای من فرصتی شد که تا این لحظه، دو کتاب را ترجمه کنم و به چاپ برسانم.

کتابی که در دست دارید، اگر بگویم ترجمه بهترین کتاب آموزش برنامه‌نویسی پایتون نسخه ۳.۷ موجود است، گزاف نگفته‌ام. با کمی گشت و گذار در وب، به این خواهید رسید.

پایتون زبانی است که جای خود را به‌خوبی در همه زمینه‌های فنآورانه، دانشگاهی، کسب‌وکار و فضای مجازی باز کرده است و کمتر کسی است که این را نداند. بر خود لازم دانستم این اثر را به بهترین شکل ممکن ترجمه کنم تا شاید بتواند راهگشای برنامه‌نویسان و دانشجویان و حتی دانش‌آموزانی که قصد فراگیری یک زبان همه‌کاره را دارند باشد. هرچند، هیچ اثری به‌دور از اشکال نیست. کافیست بر ما منت نهید و نظرها و یا اشکال‌های احتمالی که از چشم ما به‌دور مانده را در صفحه این کتاب در سایت انتشارات پندارپارس مطرح فرمایید.

حسین یعسوبی

مدیر مسئول انتشارات پندار پارس

آذر ۹۹

فهرست

پیش‌گفتار

هنگامی که نوشتن نخستین ویرایش این کتاب را آغاز کردم، نمی‌دانستم دقیقا چه چیزی در انتظارم است. به تدریج، آموختم چگونه هر عنوان را به یک داستان تبدیل کنم. می‌خواستم با ارائه مثال‌های مفید، ساده، و قابل درک، درباره پایتون صحبت کنم، اما هم‌زمان، می‌خواستم با تزریق تجربه شخصی خودم به صفحات کتاب، هرچیزی که طی سال‌ها آموخته بودم و فکر می‌کردم می‌تواند برای خوانندگان با ارزش باشد را ارائه دهم. شاید خوانندگان موافق نباشند و این کارها را به روش دیگری انجام دهند اما امیدوارم روش من، روش بهتری باشد.

هدفم این بود که این کتاب، تنها درباره زبان نباشد بلکه درباره برنامه‌نویسی باشد. در واقع، هنر برنامه‌نویسی، چند دیدگاه را دربر می‌گیرد و زبان، تنها یکی از آنهاست.

جنبه قاطع دیگر برنامه‌نویسی، استقلال است. توانایی متوقف نکردن خودتان وقتی به دیوار بلندی برخورد می‌کنید و نمی‌دانید برای حل مشکل موجود، چه کنید. کتابی برای آموزش آن وجود ندارد و فکر کردم به‌جای اینکه سعی کنم این جنبه را آموزش دهم، با آزمایش کردن، خوانندگان را در این باره آموزش دهم.

در نهایت می‌خواستم کتابی تقریبا متفاوت بنویسم. پس تصمیم گرفتم با ویراستارم، نخستین بخش را به روش تئوری بنویسم و عناوینی که مشخصات پایتون را توضیح می‌دهد ارائه دهم و در بخش دوم، انواع پروژه‌های واقعی را ارائه دهم تا خوانندگان ببینند با این زبان، چه کارها که نمی‌توان انجام داد!

با همه این اهداف ذهنی، سخت‌ترین چالش هم پیش رویم بود: همه آنچه می‌خواستم بنویسم، محدود به تعداد صفحاتی مشخص (از سوی ناشر) بود که کار دشواری بود و فداکاری‌هایی انجام شد.

این تلاش‌ها تا امروز ۳ سال به درازا کشیده شد و من هنوز نظر خوانندگان را دریافت می‌کنم و انرژی می‌گیرم و تلاش می‌کنم که دانش خود را با آنها به اشتراک گذارم.

اینک که **ویرایش دوم** کتاب را می‌نویسم چند نظر دیگر هم دارم. تصمیم داشتم یک فصل درباره `IO` به کتاب بیافزایم که به شدت به آن نیاز است، و حتی فرصت افزودن دو فصل دیگر را هم داشتم، یکی درباره رمزها و یکی درباره اجرای همزمان. قطعا در آینده، چالشی‌ترین فصل در کل کتاب است و هدف آن، شبیه‌سازی خوانندگان برای رسیدن به سطحی است که بتوانند به آسانی، کدهای آنرا هضم کنند و مفهومش را درک کنند.

در ویرایش جدید کتاب، همه فصل‌های اصلی حفظ شده، به‌جز آخرین فصل که کمی به آن اضافه شده است. همه فصل‌ها مطابق با آخرین نسخه پایتون، یعنی **`Python 3.7`** به‌روز رسانی شده است.

وقتی به این کتاب می‌نگرم محصول تکامل یافته‌ای را می‌بینم. دارای چندین فصل است و محتویاتش برای فهم بهتر داستان، دوباره سازماندهی شده است، اما روح کتاب هنوز آنجاست. مهمترین و اصلی‌ترین نکته، توانمند کردن خوانندگان است که هنوز دست‌نخورده باقی مانده است.

امیدوارم این ویرایش بتواند حتی موفق‌تر از نسخه پیشین باشد و به تبدیل خوانندگان به برنامه‌نویسان عالی کمک کند و به توسعه تفکر آنها و رشد مهارت‌های آنها یاری رساند.

کتاب برای چه کسانی مفید است

پایتون، مشهورترین زبان آموزشی مقدماتی در دانشگاه‌های تراز بالای علوم رایانه در ایالات متحده است، پس اگر تازه وارد دنیای برنامه‌نویسی شده‌اید یا اگر تجربه اندکی دارید و دوست دارید در این راه گام بگذارید، این زبان و این کتاب، آن چیزی است که به آن نیاز دارید. طراحی و قابل حمل بودن شگفت‌انگیز آن، به پربار شدن شما کمک خواهد کرد، بدون توجه به محیطی که برای کار با آن انتخاب می‌کنید.

اگر پیش‌تر با پایتون یا زبان دیگری کار کرده‌اید، باز هم این کتاب برایتان مفید است، هم به‌عنوان یک مرجع بنیادی و اصلی و هم برای ارائه تجربه‌ها و پیشنهادهای بسیار گسترده‌ای که در دو دهه اخیر، به‌دست آمده و در این کتاب در اختیارتان می‌گذاریم.

محتویات کتاب

فصل ۱، معرفی کوتاه پایتون؛ با مفاهیم بنیادی برنامه‌نویسی پایتون آشنا می‌شوید. شما را تا بالا آوردن و اجرای پایتون روی سیستم‌تان راهنمایی می‌کنیم و با چند دستور آن آشنا خواهید شد.

فصل ۲، `Data Type`های درون–ساخت؛ انواع داده‌های پایتون که به‌شکل پیش‌ساخته در آن موجود است را معرفی می‌کند. پایتون دارای مجموعه بسیار گسترده‌ای از انواع داده‌های بومی و محلی است و این فصل، با یک مثال کوتاه، هر یک از آنها را توصیف می‌کند.

فصل ۳، تکرار و ساخت تصمیم‌ها؛ می‌آموزد با بازرسی شرط‌ها، اعمال منطق، و پیاده‌سازی حلقه‌ها، چگونه جریان کد را کنترل کنیم.

فصل ۴، توابع، ساخت بلوک‌های کد؛ شیوه نوشتن توابع را آموزش می‌دهد. توابع، کلیدهایی برای استفاده دوباره از کدها و کاهش زمان دیباگ کردن است و عموما برای نوشتن بهتر کد است.

فصل ۵، ذخیره زمان و حافظه؛ دیدگاه‌های عملیاتی برنامه‌نویسی پایتون را به شما معرفی می‌کند. این فصل، شیوه نوشتن درک مطالب و ژنراتورها را می‌آموزد که ابزارهای مفیدی است که می‌توان برای افزایش سرعت کد و ذخیره حافظه، از آنها استفاده کرد.

فصل ۶، `OOP`، `Decorators`، و `Iterators`؛ مبانی برنامه‌نویسی شئ‌گرا با پایتون را آموزش می‌دهد. مفاهیم کلیدی و همه پتانسیل‌های این پارادایم را نشان می‌دهد. همچنین، یکی از محبوب‌ترین ویژگی‌های

پایتون، یعنی دکوراتورها را نشان می‌دهد. در آخر نیز مفاهیم itratorها یا همان تکرارکننده‌ها را پوشش می‌دهد.

فصل ۸، آزمایش، پروفایل‌کردن، و کار با استثناءها؛ شیوه ایجاد کدهای قوی‌تر، سریع‌تر، و ماناتر با استفاده از فنونی همچون آزمایش و پروفایل کردن را می‌آموزد. همچنین، مفهوم استثناء را به‌شکل رسمی تعریف می‌کند.

فصل ۹، رمزنگاری و توکن‌ها؛ به مفاهیم امنیت، هش‌ها، رمزنگاری، و توکن‌ها می‌پردازد که بخشی از برنامه‌نویسی روزانه‌ای است که ارائه می‌شود.

فصل ۱۰، اجرای همزمان یا Concurrent Execution؛ فصل چالشی است که نحوه کار چند چیز را در یک زمان توضیح می‌دهد. این فصل، مقدمه‌ای از جنبه‌های تئوری این موضوع را بیان می‌دارد و سپس سه تمرین زیبا ارائه می‌دهد که با تکنیک‌های متفاوتی توسعه یافته و در نتیجه، امکان تشخیص تفاوت میان نمونه‌های ارائه شده را به خواننده می‌دهد.

فصل ۱۱، دیباگ‌کردن و رفع اشکال؛ روش‌های اصلی دیباگ‌کردن کد را با ارائه مثال‌هایی از نحوه پیاده‌سازی آن نشان می‌دهد.

فصل ۱۲، GUIها و اسکریپت‌ها؛ با ارائه مثالی از دو جنبه متفاوت نما، شما را راهنمایی می‌کند. اینها برخلاف پایان‌های طیف می‌باشند: یک اجرا، یک اسکریپت است و دیگری، یک برنامه کاربردی رابط کاربری گرافیکی مناسب است.

فصل ۱۳، علم داده؛ چند مفهوم کلیدی و یک ابزار بسیار ویژه را معرفی می‌کند، Jupyter Notebook.

فصل ۱۴، توسعه وب؛ اصول بنیادی توسعه وب را معرفی می‌کند و یک پروژه را با استفاده از فریمورک وب Django تحویل می‌دهد. این مثال برپایه عبارات باقاعده است.

بهره‌برداری بیشتر از کتاب

با دنبال کردن مثال‌های کتاب، آموزش بهتری می‌گیرید. به این منظور، به یک رایانه، یک ارتباط اینترنت، و یک مرورگر نیاز دارید. **کتاب برای نسخه ۳.۷ پایتون نوشته شده، اما بیشتر بخش‌های آن با هر نسخه جدید Python 3.* باید کار کند.** راهنمای نصب پایتون روی سیستم‌عامل هم ارائه شده است. رویه‌های نصب همیشه درحال تغییر است، پس لازم است به به‌روزترین راهنمای نصب که در وب یافت می‌شود مراجعه شود. همچنین شیوه نصب کتابخانه‌های اضافه به‌کار رفته در انواع مثال‌ها را توضیح داده‌ایم و درصورت برخورد با هر مشکلی حین نصب آنها، پیشنهادهایی نیز ارائه کرده‌ایم. برای تایپ کد، نیاز به هیچ ویرایشگر خاصی نیست؛ هرچند، پیشنهاد می‌شود آنهایی که در مثال‌های پیش رو معرفی شده است را مدنظر داشته باشید تا متناسب با محیط کدنویسی ما باشد. در فصل یک پیشنهادهایی برای این موضوع ارائه شده است.

دانلود فایل‌های کد مثال‌ها

فایل‌های کد مثال‌های کتاب را از آدرس www.packtpub.com یا سایت انتشارات پندار پارس (صفحه این کتاب) بردارید. برای دانلود فایل‌ها از سایت پکت، به این شکل عمل کنید:

۱. در سایت www.packtpub.com لاگین یا رجیستر کنید.
۲. وارد برگه SUPPORT شوید.
۳. روی Code Downloads & Errata کلیک کنید.
۴. نام کتاب لاتین را در کادر Search وارد و دستورهای روی صفحه را دنبال کنید.

به محض دانلود شدن فایل، آنرا با استفاده از آخرین نسخه برنامه‌های زیر، از حالت فشرده خارج سازید:

- WinRAR/7-Zip for Windows
- Zipeg/iZip/UnRarX for Mac
- 7-Zip/PeaZip for Linux

بسته کد این کتاب در سایت GitHub به نشانی زیر نیز موجود است. در اینجا یک به‌روزرسانی برای کد در انبار GitHub موجود است:

https://github.com/PacktPublishig/Learn-Python-Programming-Second-Edition

همچنین بسته کد دیگری از کاتالوگ غنی کتاب‌ها و ویدئوهای موجود را در آدرس زیر در اختیارتان گذاشته‌ایم:

https:/ / github. com/ PacktPublishing/

ضمنا توجه داشته باشید که ورودی و خروجی‌های خط فرمان (کامندلاین) را به‌صورت زیر، یعنی فونت ضخیم با سه فلش مقابلشان، آورده‌ایم:

```
>>> import sys
>>> print(sys.version)
```

فصل ۱

معرفی کوتاه پایتون

به اختصار، کدنویسی به رایانه می‌گوید با استفاده از یک زبانی که می‌شناسد، کاری را انجام دهد. رایانه‌ها ابزارهای بسیار مفیدی هستند، اما شوربختانه نمی‌توانند برای خودشان فکر کنند. آنها نیاز دارند هر چیزی به آنها گفته شود: چگونه یک کار انجام شود، چگونه برای تصمیم‌گیری برای مسیری که باید دنبال شود، شرطی را بررسی کند، چگونه داده‌هایی که از یک وسیله (دیوایس) می‌آید رسیدگی شود، مانند شبکه یا یک دیسک، و وقتی مورد پیش‌بینی نشده‌ای رخ دهد چه واکنشی نشان داده شود؛ مثلا عملیاتی شکست بخورد یا گم شود.

به سبک‌ها و زبان‌های گوناگونی می‌توان کدنویسی کرد. چه بسیاری از آنها سخت و پیچیده هستند. هر کسی می‌تواند چگونگی کدنویسی را بیاموزد و شما نیز می‌توانید. اما، اگر می‌خواستید شاعر شوید آیا نوشتن به تنهایی کافی بود؟ مجبور بودید یک سری مهارت کسب کنید و تلاش بیشتری انجام دهید و طبع شعر هم که زیربنای کار است.

در آخر، همه چیز بستگی به این دارد که قصد دارید در این جاده حرکت کنید. کدنویسی، کنار هم چیدن یک سری دستور که کار کند نیست. چیزی خیلی بیشتر از این است.

کد خوب، باید کوتاه، سریع، زیبا و باسلیقه باشد، خواندن و فهم آن آسان باشد، بسط و تغییرش ساده باشد، پیمایش و تعمیرش ساده باشد، و به آسانی آزمایش شود. نوشتن کدی که هم‌زمان، همه این موارد کیفی را داشته باشد نیاز به زمان و کسب تجربه بالا دارد، اما خبر خوب این است که با خواندن این کتاب، نخستین گام به سمت این ایده‌آل‌ها را خواهید پیمود و تردیدی ندارم که از پس آن بر می‌آیید. در حقیقت، هرکسی می‌تواند، همه ما همیشه درحال برنامه‌نویسی هستیم، تنها از آن آگاه نیستیم. به مثال زیر توجه کنید:

فرض کنید می‌خواهید یک فنجان قهوه دم کنید. برای این کار به یک فنجان، یک ظرف قهوه‌دان، یک قاشق، آب، و کتری نیاز دارید. حتی اگر این کار را بلد نباشید، سعی می‌کنید مقداری داده را ارزیابی کنید. مطمئن می‌شوید آب در کتری هست و زیر کتری روشن است، فنجان تمیز است، و قهوه کافی در قهوه‌دان موجود است. سپس، آب را جوش می‌آورید و شاید در این حین، مقداری قهوه در فنجان بریزید. وقتی آب جوش آماده شد، آنرا در فنجان می‌ریزید و آنرا با قاشق به‌هم می‌زنید.

برنامه‌نویسی این مثال چگونه است؟

ما منابع را گردآوری کردیم (کتری، قهوه، آب، قاشق، و فنجان) و برخی شرایط مربوط به آنها را فراهم کردیم (زیر کتری را روشن کردیم، تمیز بودن فنجان، و وجود داشتن قهوه). سپس دو کار را آغاز کردیم (جوش آوردن آب و ریختن قهوه در فنجان)، و هنگامی که هر دوی آنها انجام شد، با ریختن آب در فنجان و هم زدن، رویه را به پایان رساندیم.

می‌توانید آنرا ببینید؟ تنها یک کارکرد سطح بالای یک برنامه قهوه را توضیح دادیم. کار دشواری نبود اما این همان کاری است که مغز ما هر روز انجام می‌دهد: ارزیابی شرط‌ها، تصمیم برای انجام کارها، انجام وظایف، تکرار برخی از آنها، و توقف در برخی نقاط. اشیاء را تمیز کن، آنها را سرجایشان بگذار، و غیره.

همه آن چیزی که لازم است اکنون بیاموزید این است که همه کارهایی که به‌شکل خودکار در زندگی روزمره خود انجام می‌دهید را خرد کنید تا یک رایانه بتواند برخی از آنها را واقعا درک کند. و لازم است یک زبان را برای ساختن آن، به خوبی فراگیرید.

این کتاب برای همین منظور است. نحوه انجام این کار را خواهیم گفت و تلاش می‌کنیم با تعریف چند مثال ساده اما متمرکز (از نوع مورد علاقه‌ام)، این کار را انجام دهیم.

در این فصل، موارد زیر را پوشش می‌دهیم:

- ویژگی‌ها و زیست‌بوم پایتون
- راهنمایی‌هایی برای شیوه اجرا و کار با پایتون و محیط‌های ویژه
- شیوه اجرای برنامه‌های پایتون
- شیوه سازماندهی کد و مدل اجرایی پایتون

معرفی پایتون

هنگامی که به آموزش کدنویسی می‌پردازم دوست دارم به جهان واقعی ارجاع دهم؛ باور دارم که حقایق موجود در جهان واقعی، به شناخت بهتر مفاهیم کمک می‌کند. هرچند، اینک زمان آن رسیده که کمی جدی‌تر باشیم و کدنویسی را از زاویه فنی‌تر ببینیم.

هنگامی که کدی را می‌نویسیم، دستوری را برای انجام کارهایی به رایانه می‌دهیم. اتفاق در کجا می‌افتد؟ در خیلی جاها: حافظه رایانه، درایوهای سخت، کابل‌های شبکه، CPU و غیره. این یک جهان کامل است که بیشتر وقت‌ها بیانگر زیرمجموعه‌ای از جهان واقعی است.

اگر یک قطعه کد نرم‌افزاری بنویسید که امکان خرید آنلاین لباس را به مردم بدهد، مجبورید افراد واقعی، لباس‌های واقعی، برندهای واقعی، اندازه و غیره را درون مرزهایی از یک برنامه، ارائه کنید.

پس برای انجام این کار، نیاز به ایجاد و رسیدگی به اشیاء در برنامه‌ای که می‌نویسید دارید. هر شخص می‌تواند یک شئ باشد. هر ماشین، یک شئ است. یک جفت جوراب هم یک شئ است. خوشبختانه، پایتون اشیاء را به خوبی می‌شناسد.

دو ویژگی اصلی که هر شئ دارد، مشخصه‌ها (properties) و متدها (methods) است. اجازه دهید یک شخص را به‌عنوان مثال شئ درنظر بگیریم. نوعا در یک برنامه رایانه‌ای، افراد را به‌عنوان مشتری یا کارمند معرفی می‌کنید. مشخصه‌هایی که برای آنها نگاه می‌دارید چیزهایی مانند نام، SSN، سن، آیا دارای گواهی‌نامه رانندگی هستند، نشانی ایمیل، جنسیت، و غیره است. در یک برنامه رایانه‌ای، همه داده‌هایی که به‌منظور استفاده از یک شئ برای رسیدن به هدف خود نیاز دارید را ذخیره می‌کنید. اگر درحال کدنویسی یک وب‌سایت فروش لباس هستید، شاید بخواهید قد و وزن افراد را هم در کنار دیگر مقیاس‌های مشتریان خود ذخیره کنید تا بتوانید لباس‌های مناسب را به آنها پیشنهاد دهید. پس **مشخصه‌ها، ویژگی‌های یک شئ هستند**. همیشه از آنها استفاده می‌کنیم: می‌توانید آن قلم را به من بدهید؟___ کدام یکی؟___ قلم مشکی را. در اینجا ما از مشخصه مشکی یک قلم برای شناسایی آن استفاده کردیم.

متدها چیزهایی هستند که یک شئ می‌تواند انجام دهد. من شخصا متدهایی مانند صحبت کردن، راه رفتن، خوابیدن، خوردن، خواندن و نوشتن و غیره دارم. **هر آن چیزی که بتوانم انجام دهم می‌تواند به عنوان متدهای اشیائی دیده شود که مرا تعریف می‌کند**.

پس اکنون که می‌دانید اشیاء چیست و اینکه آنها متدهایی که می‌توانید اجرا کنید و مشخصه‌هایی که می‌توانید رسیدگی کنید را عرضه می‌کنند، آماده آغاز کدنویسی هستید. کدنویسی در حقیقت، به‌سادگی مدیریت آن اشیائی است که در زیرمجموعه جهانی که ما در حال بازتولید در نرم‌افزارمان هستیم زندگی می‌کنند. اشیاء را می‌توان به دلخواه خود، ایجاد، استفاده، استفاده مجدد و حذف کرد.

با توجه به فصل مدل داده‌ای در مستند رسمی پایتون:

"اشیاء، تجرد داده‌های پایتون هستند. همه داده‌ها در یک برنامه پایتون، توسط اشیاء یا ارتباط میان آنها بیان می‌شوند."

در فصل ۶ نگاه دقیق‌تری به اشیاء پایتون می‌اندازیم. اینک لازم است بدانید که **هر شیئی در پایتون دارای یک شناسه یا ID، یک نوع، و یک مقدار است**.

ID یک شئ پس از ایجاد هرگز تغییر نمی‌کند. این یک شناساننده یکتا برای آن است و پایتون برای بازیابی شئ وقتی می‌خواهیم از آن استفاده کنیم در پشت صحنه از آن استفاده می‌کند.

نوع یا type هم به هیچ وجه تغییر نمی‌کند. نوع می‌گوید چه عملیاتی توسط شئ پشتیبانی می‌شود و چه مقادیر ممکنی می‌تواند به آن تخصیص یابد. در فصل ۲ مهم‌ترین انواع داده پایتون را خواهیم دید.

مقدار یا value می‌تواند تغییر کند یا نکند. اگر بتواند تغییر کند، به آن شئ تغییرپذیر، ناپایدار، یا mutable می‌گوییم؛ وگرنه به آن تغییرناپذیر، پایدار، یا immutable می‌گوییم.

چگونه می‌توان از یک شئ استفاده کرد؟ البته که یک نام به آن می‌دهیم! با نام‌گذاری آن می‌توانیم هر بار برای بازیابی و استفاده از آن، از آن نام استفاده کنیم.

در یک وضعیت عمومی‌تر، اشیائی همچون اعداد، رشته‌ها (متن)، کلکسیون‌ها و غیره، با یک نام مرتبط هستند. معمولا می‌گوییم که این نام، نام یک متغیر است. **متغیر را مانند یک جعبه‌ای ببینید که برای نگهداری داده‌ها می‌توانید استفاده کنید.**

بنابراین، همه اشیائ مورد نیاز را دارید؛ اینک چه؟ خب باید از آنها استفاده کنیم، درست است؟ شاید بخواهیم آنها را از طریق یک ارتباط شبکه‌ای انتقال دهیم یا در یک دیتابیس ذخیره کنیم. شاید آنها را روی یک صفحه وب نمایش دهیم یا آنها را درون یک فایل بنویسیم. به منظور این کار لازم است با پر کردن یک فرم کاربری، یا فشار یک دکمه، یا بازکردن یک صفحه وب و انجام یک جست‌وجو، واکنش نشان دهیم. این کارها را با اجرای کد خود، ارزیابی شرط‌ها برای انتخاب اینکه کدام اجزا اجرا شود، چند بار، و در چه شرایطی، عملی می‌کنیم.

و برای انجام همه اینها، به‌شکل اساسی نیاز به زبان داریم. در اینجاست که پایتون به کارمان می‌آید. پایتون زبانی است که در این کتاب برای دستور دادن به رایانه در انجام کاری برایمان، استفاده خواهیم کرد.

مقدمه چینی کافیست، اجازه دهید وارد گود شویم.

ورود به پایتون

پایتون، تولید شگفت‌انگیز Guido Van Rossum، یک ریاضی‌دان و دانشمند هلندی علوم رایانه است که تصمیم گرفت با پرژه‌ای که در کریسمس ۱۹۸۹ آغاز شد به جهانیان هدیه کند. این زبان در حدود سال ۱۹۹۱ به‌شکل عمومی منتشر شد و امروز، یکی از برترین زبان‌های برنامه‌نویسی در کل دنیاست.

من وقتی ۷ سال داشتم برنامه‌نویسی را روی یک کومودور VIC-20 آغاز کردم که بعدها با برادر بزرگترش یعنی کومودور ۶۴ جایگزین شد. آن زبان، بیسیک بود. سپس به سراغ زبان پاسکال، اسمبلی، C، C++، جاوا، جاوااسکریپت، ویژوال بیسیک، PHP، ASP، ASP.NET، C# و دیگر زبان‌های کوچکی که حتی به‌خاطر نمی‌آورم رفتم، اما تنها وقتی با پایتون آشنا شدم احساس کردم که گمگشده خود را یافته‌ام.

وقتی همه اجزای بدن شما در حال نعره زدن است، و می‌گوید این را بخر! این برایمان بهترین است!

این مرا یاد روزی انداخت که خواستم از آن استفاده کنم. سینتکس آن کمی متفاوت از آنچه استفاده کرده بودم بود، اما پس از گذراندن احساس عجیب آغازین (مانند داشتن لباس‌های نو)، احساس کردم عاشقش شده‌ام. عمیقا. اجازه دهید بگویم چرا.

درباره پایتون

پیش از اینکه وارد جزئیات کار شویم به حسی بپردازم که چرا فردی می‌خواهد از پایتون استفاده کند (پیشنهاد می‌کنم صفحه پایتون در ویکیپدیا را برای جزئیات بیشتر بخوانید).

به نظر من، پایتون دارای نقاط قوت زیر است:

قابل‌حمل بودن

پایتون در هر جایی اجرا می‌شود و حمل‌ونقل یک برنامه از لینوکس به ویندوز یا مکینتاش، معمولا تنها در حد موضوع فیکس کردن مسیرها و تنظیمات است. پایتون برای قابل حمل بودن طراحی شده و در پشت رابط کاربری، مراقب تغییرات ناگهانی سیستم‌عامل مشخص می‌باشد تا مجبور نباشید سختی کدنویسی مربوط به یک پلتفرم مشخص دیگر را به جان بخرید.

انسجام

پایتون به‌شدت منطقی و منسجم است. می‌توانید ببینید که توسط یک دانشمند خبره علوم رایانه طراحی شده است. اگر متدی را نشناسید، بیشتر وقت‌ها کافیست تنها حدس بزنید چگونه آن متد فراخوانی می‌شود.

شاید اکنون به اهمیت آن پی نبرید، به‌ویژه اگر مبتدی باشید، اما این یک ویژگی اصلی آن است. یعنی کمتر موجب آشفتگی افکارتان و به‌هم ریختگی و زیرورو کردن اسنادتان می‌شود و کمتر نیاز به رجوع به مغزتان حین کدنویسی می‌شود.

بهره‌وری توسعه‌دهنده

مطابق با نظر Mark Lutz (در ویرایش پنجم کتاب Learning Python انتشارات اورلی)، یک برنامه پایتون، معمولا یک پنجم تا یک سوم از اندازه محیط کد جاوا یا C++ را می‌گیرد. یعنی اجرای سریع‌تر برنامه. و سریع‌تر بودن عالی است. سریع‌تر بودن یعنی یک پاسخ سریع‌تر به بازار. کد کمتر نه تنها به معنای نوشتن کد کمتر است بلکه کد کمتری باید خوانده شود (و کدزن‌های حرفه‌ای، بسیار بیشتر از آنکه کدنویسی کنند کدخوانی می‌کنند)، کد کمتری تعمیر و نگهداری شود و پالایش شود.

جنبه مهم دیگر این است که پایتون بدون نیاز به گردآوری زمان‌بر و طولانی و به‌هم پیوستگی گام‌ها، اجرا می‌شود، بنابراین مجبور نیستید منتظر دیدن نتایج کار خود بمانید.

یک کتابخانه وسیع

پایتون دارای یک کتابخانه استاندارد عظیم باورنکردنی است. اگر جواب کارتان را هم ندهد، جامعه پایتون در سرتاسر جهان دارای کتابخانه‌های تنومند شخص‌سومی برای نیازهای شخصی است که می‌توان آزادانه (رایگان) در Python Package Index (PyPI) به آن دسترسی داشت. هنگامی که کدهای پایتون را می‌نویسید و متوجه می‌شوید که نیاز به یک ویژگی (فیچر) مشخصی دارید، در بیشتر موارد دست‌کم یک کتابخانه موجود است که آن ویژگی، از پیش برای شما پیاده‌سازی شده است.

کیفیت نرم‌افزار

پایتون به‌شدت روی خوانا بودن، انسجام، و کیفیت متمرکز است. یکنواختی زبان، امکان خوانا بودنش را افزایش می‌دهد و امروزه، بسیار مهم است که برنامه‌نویسی، یک تلاش جمعی باشد تا یک تلاش انفرادی، و به‌شکل گروهی انجام پذیرد. جنبه مهم دیگر پایتون، طبیعت ذاتی چند الگویی آن است. از آن می‌توان به‌عنوان یک زبان اسکریپتی استفاده کرد، اما می‌توان از سبک‌های شئ‌گرایی، دستوری، و برنامه‌نویسی فانکشنال نیز بهره‌برداری کرد. در کل، استعدادش بالاست.

یکپارچگی نرم‌افزار

جنبه مهم دیگر پایتون این است که می‌تواند با بسیاری دیگر از زبان‌ها توسعه داده شده و یکپارچه شود که به معنای این است که حتی وقتی یک شرکت درحال استفاده از زبان دیگری به‌عنوان ابزار اصلی کارش است، پایتون می‌تواند وارد شود و به‌عنوان یک عامل چسب، میان برنامه‌های کاربردی پیچیده‌ای که به هر روشی نیاز به صحبت با همدیگر دارند، عمل کند. این نوعی از یک مبحث پیشرفته است اما در جهان واقعی، این ویژگی بسیار مهم است.

رضایت‌مندی و لذت

کار با پایتون، جذاب و سرگرم‌کننده است. من می‌توانم ۸ ساعت کدنویسی کنم و خوشحال و راضی، شرکت را ترک کنم و این مغایر با تقلای برنامه‌نویس‌های دیگری است که مجبورند تحمل کنند، زیرا از زبان‌هایی استفاده می‌کنند که فاقد همین مقدار ساختارهای داده‌ای و دستورهای خوش طرح هستند. تردید نکنید که پایتون، کدنویسی را بسان سرگرمی می‌کند و این، انگیزه و کیفیت کارتان را بالا می‌برد.

اینها جنبه‌های اصلی است که چرا پایتون را به هرکسی پیشنهاد می‌کنم. البته، ویژگی‌های فنی و پیشرفته بسیاری هست که می‌توان درباره آنها صحبت کرد که جای آنها در این پیش‌گفتار نیست و فصل به فصل که جلو برویم خواهید دید.

موانع چیست؟

شاید تنها مانعی که کسی بتواند در پایتون بیابد که ربطی به سلیقه‌های شخصی ندارد، سرعت اجرایش است. نوعا، پایتون آهسته‌تر از برادران کامپایل شده‌اش است. پیاده‌سازی استاندارد محصولات پایتون، وقتی یک برنامه کاربردی را اجرا می‌کنید، یک نسخه‌ی کامپایل شده از سورس کد به‌نام بایت کد است (با پسوند `.pyc`)، که سپس توسط مفسر پایتون اجرا می‌شود. مزیت این رویه در قابل حمل بودن آن است که ما بهای این کاهش سرعت ناشی از این حقیقت که پایتون مانند زبان‌های دیگر، در سطح ماشین کامپایل نمی‌شود را می‌پردازیم.

هرچند، امروزه سرعت پایتون به‌ندرت یک مشکل تقلی می‌شود، و پهنه کاربردش توجهی به این ویژگی منفی آن ندارد. اتفاقی که می‌افتد این است که در زندگی واقعی، هزینه سخت‌افزار، یک مشکل دائمی نیست، و معمولا دست‌یابی به سرعت با کارهای موازی‌سازی، به‌اندازه کافی آسان است. افزون بر این، بسیاری از برنامه‌نویسان، بخش وسیعی از زمان انتظار را صرف تکمیل عملیات `IO` می‌کنند.؛ بنابراین سرعت اجرای خام، اغلب یک عامل دوم در بازدهی کلی به‌شمار می‌رود. حتی وقتی پای خرد کردن کد به چند قطعه به میان می‌آید، یکی می‌تواند روی پیاده‌سازی‌های سریع‌تر پایتون همچون `PyPy` سوئیچ کند، که یک متوسط افزایش سرعت پنج–لا را با پیاده‌سازی تکنیک‌های کامپایل پیشرفته، ارائه می‌دهد.

هنگام پرداختن به علم داده، بیشتر تمایل به کتابخانه‌هایی دارید که با پایتون استفاده کنید، مانند `Pandas` و `NumPy`، و به سرعت محلی ناشی از روشی که آنها پیاده‌سازی می‌شوند دست‌یابید.

اگر آن استدلال به‌اندازه کافی خوب نبود می‌توانید همواره درنظر بگیرید که پایتون برای هدایت درب پشتی سرویس‌هایی همچون Spotify و Instagram به‌کار رفته است که بازدهی، امر نگران کننده‌ای است. با این حال، پایتون کار خودش را به بهترین شکل ممکن انجام داده و می‌دهد.

امروزه چه افرادی از پایتون استفاده می‌کنند؟

هنوز قانع نشده‌اید؟ اجازه دهید نگاهی به شرکت‌هایی که امروزه از پایتون استفاده می‌کنند بیاندازیم: گوگل، یوتیوب، دراپ‌باکس، یاهو!، Zope Corporation، Industrial Light & Magic، Walt Disny Feature Animation، Blender 3D، Pixar، NASA، NSA، Red Hat، Nokia، IBM، Netflix، Yelp، Intel، Cisco، HP، Qualcomm و JPMorgan Chase تنها چند مورد از آنهاست.

حتی بازی‌هایی همچون Battlefield 2، `Civilization IV` و QuArK با استفاده از پایتون پیاده‌سازی شده‌اند.

پایتون در مفاهیم پیچیده‌ای همچون برنامه‌نویسی سیستمی، برنامه‌نویسی وب، برنامه‌های کاربردی `GUI`، بازی‌سازی و روباتیک، نمونه‌سازی سرعت، تعامل و یکپارچه‌سازی سیستم، علم داده، برنامه‌های کاربردی

دیتابیس، و موارد دیگر کاربرد دارد. چندین دانشگاه معتبر نیز پایتون را به عنوان زبان اصلی در رشته علوم کامپیوتر تدریس می‌کنند.

تنظیم محیط

پیش از اینکه به نصب پایتون روی سیستم‌تان بپردازیم اجازه دهید کمی درباره نسخه‌ای از پایتون که در این کتاب استفاده می‌کنیم صحبت کنیم.

پایتون ۲ در مقابل پایتون ۳

پایتون دارای دو نسخه اصلی ۲ و ۳ است که نسخه ۳ پس از نسخه ۲ آمده است. این دو، شباهت زیادی به هم دارند اما از برخی جهات، کمی با هم ناسازگارند.

در دنیای واقعی، پایتون ۲ که در سال ۲۰۰۰ منتشر شد فاصله زیادی با قدیمی شدن دارد. کوته سخن، باوجودی‌که پایتون ۳ از سال ۲۰۰۸ بیرون آمد، فاز دگرگونی آن از نسخه ۲، هنوز به پایان نرسیده است. این تقریبا ناشی از این واقعیت است که پایتون ۲ به‌شکل گسترده در صنعت به‌کار گرفته شده است و البته، شرکت‌ها آنقدر مشتاق به به‌روزرسانی سیستم‌های خود با نسخه جدید نبودند، که دلیلش مشکل نداشتن با نسخه ۲ بود. با جست‌وجو در وب می‌توانید به تفاوت‌های اساسی این دو نسخه پی برید.

مشکل دیگری که مانع این تغییر است، توانایی کتابخانه‌های شخص سومی است. معمولا، یک پروژه پایتون به ده‌ها کتابخانه خارجی تکیه دارد و البته وقتی پروژه جدیدی را آغاز می‌کنید نیاز دارید از پیش، برای هر الزام تجاری که شاید به دنبال آن بیاید، از موجود بودن کتابخانه سازگار با نسخه ۳ مطمئن شوید. اگر این مورد نباشد آغاز یک پروژه شاخه–جدید در پایتون ۳ به معنای تولید یک ریسک نهفته است، که یعنی شرکت‌ها از برداشتن آن خشنود نیستند.

هرچند، هنگام نوشتن این کتاب، شمار بسیار زیادی از کتابخانه‌های استفاده شده، به پایتون ۳ پورت شده‌اند و آغاز یک پروژه در پایتون ۳ در بیشتر موارد، امن است. بسیاری از کتابخانه‌ها از نو نوشته شده‌اند تا بیشتر تحت فشار قدرت کتابخانه `six` (این نام، از ضرب ۲ در ۳ می‌آید که ناشی از پورت کردن از نسخه ۲ به ۳ است)، با هر دو نسخه سازگار باشند، که به درون‌گرایی و وفق دادن رفتار متناسب با نسخه به‌کار رفته، کمک می‌کند. با توجه به PEP 373[1]، پایان عمر[2] پایتون ۲.۷ در سال ۲۰۲۰ تنظیم شده و نسخه ۲.۸ آن وجود نخواهد داشت، پس این

[1] https://legacy.python.org/dev/peps/pep-0373
[2] End of life (EOL)

زمانی است که شرکت‌هایی که دارای پروژه‌های اجرایی در پایتون ۲ هستند لازم است تا دیر نشده، به‌فکر تدبیر مناسب برای رفتن به پایتون ۳ باشند.

در باکس من (MacBook Pro)، این آخرین نسخه پایتونی است که دارم:

```
>>> import sys
>>> print(sys.version)
3.7.0a3 (default, Jan 27 2018, 00:46:45)
[Clang 9.0.0 (clang-900.0.39.2)]
```

همان‌گونه که می‌بینید یک نسخه آلفای منتشر شده از پایتون ۳.۷ است که در ژوئن ۲۰۱۸ انتشار یافت. متن آغازین بالا کمی کد پایتون است که در کنسول خود تایپ کرده‌ام. در مورد این به وقتش صحبت خواهیم کرد.

همه مثال‌های این کتاب با استفاده از پایتون ۳.۷ اجرا خواهد شد. حتی اگر هنگامی که آخرین نسخه باز هم کمی متفاوت از آنچه دارم باشد، مطمئن خواهم شد که همه کدها و مثال‌ها در زمان انتشار کتاب، با نسخه ۳.۷ به‌روز می‌شود.

برخی کدها نیز می‌تواند در پایتون ۲.۷ اجرا شود، یا به‌شکلی که هست یا با تغییر جزئی، اما در این هنگام فکر می‌کنم بهتر است ابتدا پایتون ۳ را بیاموزید و سپس اگر لازم شد، به‌جای اینکه پایتون ۲ را از ابتدا فراگیرید، تفاوت پایتون ۲ با ۳ را بیاموزید.

نیازی نیست نگران این جزئیات باشید چون در عمل، مشکل بزرگی نیست.

نصب پایتون

واقعا هرگز به داشتن یک بخش setup در یک کتاب متقاعد نشده‌ام، باوجودی که مجبورید حتما فرایند نصب را انجام دهید. بیشتر وقت‌ها، میان زمانی که نویسنده، دستورها را می‌نویسد و زمانی که شما آنها را پیاده‌سازی می‌کنید ماه‌ها می‌گذرد. و این بستگی به شانس شما دارد. یک نسخه تغییر می‌کند و شاید مواردی از آن با روشی که در کتاب آمده، کار نکند. خوشبختانه ما وب را داریم، پس برای کمک به نصب و اجرای پایتون، تنها به اهداف و نکات مهم فرایند نصب می‌پردازم.

می‌دانم که بیشتر خوانندگان ممکن است ترجیح دهند کتاب به‌شکل راهنمای گام به گام باشد. شک دارم این روش، کارشان را آسان‌تر کند، پس به‌شدت باور دارم که اگر می‌خواهید کار با پایتون را آغاز کنید باید نخستین تلاشتان در جهت آشنایی با زیست‌بوم آن باشد. این بسیار مهم است و موجب افزایش اعتماد به نفس شما در فصل‌های پیش رو خواهد شد. هر کجا هم که گیر کردید، دوست خوبمان گوگل را فراموش نکنید.

تنظیم مفسر پایتون

پیش از هر چیز، اجازه دهید درباره سیستم‌عامل شما صحبت کنیم. پایتون به احتمال زیاد، از پیش در تقریبا هر توزیع لینوکسی به‌شکل پایه‌ای نصب و به‌شکل کامل، با آن یکپارچه شده است. اگر دارای سیستم‌عامل مک هستید، احتمال دارد پایتون از پیش روی آن باشد (هرچند، ممکن است تنها پایتون ۲.۷ باشد)، اما اگر از ویندوز استفاده می‌کنید به احتمال زیاد، نیاز به نصب آن دارید.

دریافت پایتون و کتابخانه‌های مورد نیاز و اجرای آن نیاز به کمی کار دستی دارد. به‌نظر می‌رسد لینوکس و مک، سیستم‌عامل‌های کاربرپسندتری برای برنامه‌نویسان پایتون باشند؛ اما ویندوز، بیشترین تلاش را می‌طلبد.

سیستم‌عامل کنونی من، یک `MacBook Pro` است و تا آخر کتاب از این با پایتون ۳.۷ استفاده می‌کنم.

کار را از وب‌سایت رسمی پایتون به نشانی `https://www.python.org` آغاز کنید که میزبان مستندات رسمی پایتون و بسیاری منابع مفید دیگر است. زمانی را به بررسی آن تخصیص دهید.

وب‌سایت دیگری که پر از منابع عالی پایتون و زیست بوم آن است `http://docs.python-guide.org` است. راهنمایی‌های نصب پایتون در دیگر سیستم‌عامل‌ها با روش‌های مختلف را در آن خواهید دید.

بخش دانلود را بیابید و نصب کننده سیستم‌عامل خود را انتخاب کنید. اگر روی ویندوز هستید، مطمئن شوید وقتی نصب کننده را اجرا می‌کنید، کنار گزینه `install pip` تیک زده باشید (واقعا پیشنهاد می‌کنم برای امنیت بیشتر، یک نصب کامل از همه کامپوننت‌هایی که نصب کننده ارائه می‌دهد را انجام دهید). درباره `pip` در ادامه خواهیم گفت.

اکنون که پایتون را روی سیستم خود نصب کردید، باید بتوانید یک کنسول را باز کنید و با تایپ `python`، پوسته تعاملی پایتون را اجرا کنید.

توجه: لطفا دقت کنید که معمولا به‌جای `Python interactive shell` (پوسته تعاملی پایتون)، از عبارت "کنسول پایتون" استفاده می‌کنیم.

برای باز کردن این کنسول در ویندوز، به منوی `Start` ویندوز بروید و در کادر `Run`، عبارت `cmd` را تایپ کنید. اگر حین کار با مثال‌های کتاب، با هرچیزی که شبیه یک مشکل مجوز است روبرو شدید، لطفا مطمئن شوید که کنسول را با مدیر سیستم یا همان `administrator` درستی اجرا کرده‌اید.

در `macOS X`، می‌توانید با رفتن به `Application | Utilities| Terminal`، یک `Terminal` را آغاز کنید.

اگر هم روی لینوکس هستید، همه چیز را درباره کنسول می‌دانید.

از واژه کنسول به‌شکل مبادله‌ای برای اشاره به کنسول لینوکس، خط فرمان ویندوز، و ترمینال مکینتاش استفاده خواهم کرد. همچنین با فرمت پیش‌فرض لینوکس به کامند–لاین پرامپت (یا همان خط–فرمان سریع) اشاره خواهم کرد، مانند این:

$ sudo apt-get update

اگر با این دستور آشنا نیستید لطفا کمی وقت برای یادگیری مبانی کارکرد کنسول بگذارید. به اختصار، پس از نماد $ معمولا دستوری می‌آید که باید تایپ کنید. به بزرگی و کوچکی واژگان و فاصله‌ها دقت کنید که بسیار مهم است.

پس از باز شدن کنسول، `python` را در پرامپت تایپ کنید و مطمئن شوید پوسته تعاملی پایتون نمایش می‌یابد. برای خروج، `exit ( )` را تایپ کنید. فراموش نکنید که اگر پایتون `2.*` از پیش روی سیستم‌عامل نصب باشد، شاید مجبور شوید پایتون ۳ را مشخص کنید.

پس از اجرای پایتون، باید عبارات زیر را ببینید (برخی جزئیات برپایه نسخه و OS تغییر می‌کند):

```
$ python3.7
Python 3.7.0a3 (default, Jan 27 2018, 00:46:45)
[Clang 9.0.0 (clang-900.0.39.2)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

اینک که پایتون نصب شده و می‌توانید آنرا اجرا کنید زمان آن است که مطمئن شوید ابزار دیگری دارید که با آن مثال‌های این کتاب را دنبال کنید: `virtualenv`.

درباره `virtualenv`

همان‌گونه که احتمالا از نامش حدس زده‌اید، `virtualenv` کلا درباره محیط‌های مجازی است. اجازه دهید کمی توضیح دهیم که اینها چیست و چرا به اینها نیاز داریم. با یک مثال ساده، توضیح می‌دهیم.

پایتون را روی سیستم نصب می‌کنید و کار با یک وب‌سایت برای `Client X` را آغاز می‌کنید. یک پوشه پروژه می‌سازید و شروع به کدنویسی می‌کنید. در کنار آن، برخی کتابخانه‌ها را نیز نصب می‌کنید؛ مانند فریمورک `Django` که در فصل ۱۴ به آن خواهیم پرداخت. فرض کنید نسخه `Django 1.7.1` را برای `Project X` نصب می‌کنید.

اکنون وب‌سایت شما آنقدر خوب است که مشتری دیگری به‌نام `Y` می‌گیرید. او می‌خواهد وب‌سایت دیگری بسازید، پس `Project Y` را آغاز می‌کنید و به همان روش، دوباره نیاز به نصب جانگو دارید. تنها مشکلی که

وجود دارد این است که نسخه جانگو اکنون به 1.8 ارتقاء یافته است و نمی‌توانید آن را روی سیستم خود نصب کنید زیرا جایگزین نسخه پیشین آن که برای Project X نصب کرده بودید خواهد شد. نمی‌خواهید ریسک معرفی مشکلات ناسازگاری را بکنید، بنابراین دو انتخاب دارید: یا درگیر نسخه‌ای شوید که هم‌اکنون روی سیستم دارید، یا آنرا به‌روز کنید و مطمئن شوید نخستین پروژه، باز به‌شکل کامل و صحیح با نسخه جدید کار می‌کند.

اگر صادق باشیم، هیچ کدام از این انتخاب‌ها چندان خوشایند نیست. قطعا نیست. اینجاست که راه‌حل مجازی‌سازی به میان می‌آید: virtualenv!

Virtualenv ابزاری است که امکان ایجاد یک محیط مجازی را می‌دهد. به دیگر سخن، ابزاری برای ساخت محیط‌های ایزوله شده پایتون است که هر کدام، پوشه‌ای است حاوی همه اجرا شدنی‌های لازم برای استفاده از پکیج‌هایی که یک پروژه پایتون لازم دارد (فعلا پکیج‌ها را مانند کتابخانه درنظر بگیرید).

پس برای ایجاد یک محیط مجازی برای Project X، همه وابستگی‌ها را نصب کنید و سپس یک محیط مجازی برای پروژه Y بسازید و همه وابستگی‌هایش را بدون کوچکترین نگرانی نصب کنید، زیرا هر کتابخانه‌ای که بعدا نصب می‌کنید، در انتهای مرزهای محیط مجازی مناسب می‌نشیند. در این مثال، پروژه X کتابخانه Django 1.7.1 را نگه می‌دارد، درحالی که پروژه Y، کتابخانه Django 1.8 را نگه می‌دارد.

توجه: بسیار مهم است که هرگز کتابخانه‌ها را مستقیما در سطح سیستم نصب نکنید. مثلا لینوکس برای وظایف و عملیات گوناگونی به پایتون تکیه دارد و اگر با سیستم نصب پایتون بخواهید آزارش دهید، ریسک به‌هم ریختگی تعامل کل سیستم را بالا خواهید برد (حدس بزنید برای این فرد چه اتفاقی می‌افتد...). پس این را به‌عنوان یک قانون مدنظر داشته باشید که پیش از رفتن به تخت‌خواب، مسواک بزنید: *همیشه، همیشه هنگام آغاز یک پروژه جدید، یک محیط مجازی بسازید.*

برای نصب virtualenv روی سیستم، چند روش وجود دارد. برای نمونه، در یک توزیع لینوکس Debian-محور، می‌توان آنرا با فرمان زیر نصب کرد:

```
$ sudo apt-get install python-virtualenv
```

شاید ساده‌ترین راه، دستورهایی باشد که در وب‌سایت رسمی virtualenv ارائه شده است:

```
https://virtualenv.pypa.io
```

خواهید دید که یکی از مرسوم‌ترین روش‌های نصب آن به‌کمک pip است که یک سیستم مدیریت پکیج به‌کار رفته برای نصب و مدیریت پکیج‌های نرم‌افزاری نوشته شده در پایتون است.