Kojo Idrissa @Transition

The Python Use Spectrum
PyGotham
Oct. 2019

- Now: Jr. Developer, Pythonista/Djangonaut
- Then: Accountant & uni instructor w/ an MBA

Kojo Idrissa @Transition

Who Else Am I?

- DjangoCon US Organizer & DevRel
- DEFNA North American Ambassador
  - Official Hello and Group Hug to PyGotham
- @DEFNADO/defna.org for more info

Define DEFNA has grants for Django-related events in your area.

I do a lot of community organizing stuff, and I try to help add new contributors to the community. This talk focuses on increasing contributions from our community.

Kojo Idrissa @Transition

Motivation

I gave this talk in January 2019 and the main point got lost in a debate about how people SHOULD write code.
The POINT of this talk is to get people thinking about the fact that Python is an Open Source project and what that REALLY means.
Python is growing in USE, but I don’t think it’s growing at a similar rate in CONTRIBUTORS.

Kojo Idrissa @Transition

Wanted: More Help

Python is an Open Source project. What DOES that mean?
- Making Python is almost entirely a volunteer effort
- People who make, maintain and distribute Python do it in their free time
    - I'll focus mostly on the core Python language in this talk, but this is also true for everything else in the Python ecosystem.

Kojo Idrissa @Transition

Contributions: Not Just Code

- Documentation/Translation
- Conference Organization
- Legal/Logistical
- Fundraising
- Unknown Unknowns

A project this size is more like a business than a hobby. And it needs more than just updates to the codebase to survive & grow. And as it grows (which it continues to do), the needs of the project expand. These topics are covered in most CS programs or boot camps. So, most “traditional” developers aren’t familiar with them.

Kojo Idrissa @Transition

Needed: More Help

But this isn’t just a want. The year has brought a LOT of change to Python. Guido van Rossum, the Python creator and former project head stepped down rather abruptly in July 2018. He left governance decisions to the core developers. In response, the core developers selected a new “Steering Council” method of guiding the project in December 2018. The 5 member steering council was selected in February 2019. This council will lead the project going forward.

Kojo Idrissa @Transition

PEP 13
Python Language Governance

Google this for more details. Here’s the problem: the people who figured out this solution AND implemented it were the core maintainers. That’s the same set of people doing MOST of the ‘heavy lifting’ behind the technical work to make sure that Python continues to exist and get updated. Again, mostly volunteer. And now they’re having to figure out governance models.

Kojo Idrissa @Transition

More Help Needed:

Because Burnout is real

These volunteers often have to take breaks from their Open Source commitments. I’m not going to name names, because the point of this talk isn’t to draw attention to individuals. It’s to draw attention to the fact that people need to take breaks from their volunteer activities when they’re overwhelmed by the workload. This is happening with other high-profile Python projects as well. A larger number of contributors would ease that load, we need. But who would those people be?

Who might help?

Two Necessary Conditions

+ Use Python to solve problems (see some value in it)
+ Feel included in the community; like they're welcome and valued enough to contribute.
  * If the word "feel" puts you off: how much volunteer effort are you putting in to help people you DON'T want to spend time with?
  * It's also the one existing community members have the most influence over.

Kojo Idrissa @Transition

Emphasis on "*might*"

- Note: Meeting those two conditions doesn't GUARANTEE contribution. It just puts you in the universe of potential contributors
- Where do these people come from? We'll start with the first criteria: who USES Python to solve problems.

Kojo Idrissa @Transition

Found:
More (potential) Help

The good news is: Python is growing VERY quickly, and is now one of the most popular programming languages. This means LOTS more people are using Python to solve their problems and thus see it’s value. And we have some actual data about that.

Kojo Idrissa @Transition

Python's Popularity:
TIOBE Language of the Year
2007, 2010, 2018

TIOBE index ranks language popularity. It's not a "perfect" metric, but it's a decent indicator.

LotY has the largest increase in ratings for that year.
These new members are helping our community grow and they COULD be our future contributors. And while growth is great, everyone isn't doing the same thing with Python.

Kojo Idrissa @Transition

The Python Use Spectrum

- There’s more than one way to write code. How you write code depends on things like why you’re writing code, who’s going to use it, and how it’s going to be run. I refer to these differing styles of writing code as “Programming” and “Software Engineering”.

Kojo Idrissa @Transition

Two Ends of a Spectrum

Programming	Software Engineering
• Individuals	• Teams
• Author	• Others
• With supervision	• In isolation
• Code output	• Code itself
• Utility	• Profession

Who writes the code?
Who's the runner or subsequent reader?
How do we expect the code to run?
What do we want: the output of the code, or the code itself?
Why are you writing code?
Software Engineering
• Higher level of robustness required; the user isn't intended to FIX things that break. This is true with open source projects as well. The software is expected to WORK.

Kojo Idrissa @Transition

The Big Takeaway

Different uses cases require different tools

The prior slide is not about making a distinction b/w who is or isn't a "real" coder. It's meant to highlight two different styles of building software. Almost everyone starts with the FIRST style, and some stay there, based on their goals/needs. Those trying to get hired as SEs or make FLOSS contributions find they need to move towards the SECOND style. Data scientists often live in the middle.

Toolsets will also differ based on your approach to writing code. Let's look at those tools.

Kojo Idrissa @Transition

Software Engineering Tools/Practices

- Version Control
- Documentation
- (Automated) Software Testing
- Dependency Management
- Software Deployment
- Knowing Your Development Environment

Software Engineers need to be familiar with all of these tools. It’s also worth noting that there’s no language mentioned here. These tools are in addition to and used with your language of choice.

Kojo Idrissa @Transition

The Big Problem

Our community culture is biased towards “software engineering”.

On the surface, this SEEMS like a good thing. But let’s look at some history.

Kojo Idrissa @Transition

Old Man Story Interlude

The Bronze Age of Python:

~1998-2000

admittedly subjective

Python was first released in 1990. I first saw it in 1998 or 1999 with 1.5.2. I recall seeing books in Barnes & Noble calling it a “scripting” language. Some people even called using Python “scripting” instead of ‘programming’. “Software engineers” didn’t use Python to do “real programming”. That was reserved for languages like C/C++. You could use Python to write scripts to connect other pieces of REAL software. But, it was gaining traction. There WERE books and it WAS being discussed. And 2.0 was released.

Kojo Idrissa @Transition

Old Man Story Interlude

The Golden Age of Python:~2010

admittedly subjective

Python 3 is out and getting traction. Universities are starting to use it as their CS 101 language. Django & Flask exist. Python became a language used for ‘real programming’ or ‘software engineering’. People are building web applications and serious tools with it. And the community culture and norms moved towards ‘software engineering’ as the One True Way to write Python. Anything ELSE was nonsense. We forgot our own history.

We went from being laughed at by Software Engineers to **being** the laughing Software Engineers.

Kojo Idrissa @Transition

Why Does This Matter?

- A bias *toward* “Software Engineering” often manifests as a bias *against* “Programming”
- Python is growing fast in Data Science and other non-Software Engineering areas

Then along came Data Science. They don’t always NEED all the ‘software engineering’ tools. But when they ask a more experienced Python developer (often a software engineer) questions, too many of us tell them they’re ‘doing it wrong’ if they don’t use our full toolset. So, large numbers of new Python developers stay at the fringes of our community. But not understanding Docker or CI doesn’t mean you can’t be part of our community.

While DS is a large, obvious example, there are plenty of other uses for Python that don’t require all of the Software Engineering tooling. Python’s accessibility, readability & clean syntax make it perfect for folks trying to solve a problem for themselves, w/o having to change professions. Python lets you cook w/o forcing you to become a chef.

Kojo Idrissa @Transition

Required vs. Helpful Tools/Practices

- Version Control
- Documentation
- (Automated) Software Testing
- Dependency Management
- Software Deployment
- Know Your Development Environment

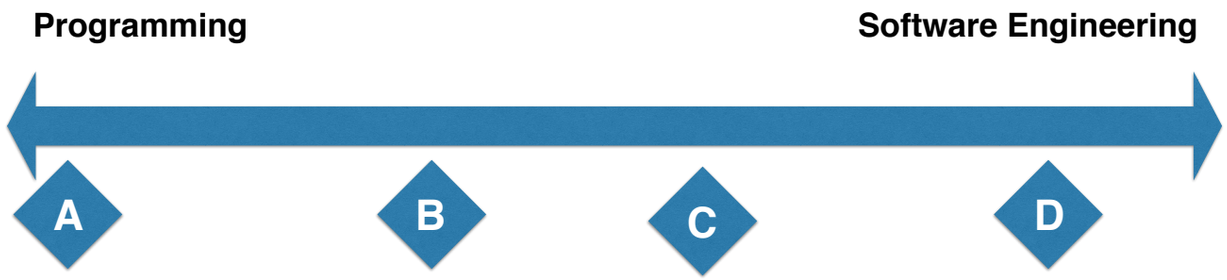
Continuing the cook vs. chef analogy:
To “cook” with Python, you HAVE to know **where** you’re expecting your code to run, and **what** you need for it to run properly.

That MAY only be having Python installed and knowing how to get it to run

Let’s take a look at some examples

Kojo Idrissa @Transition

Kojo's Personal Code Journey

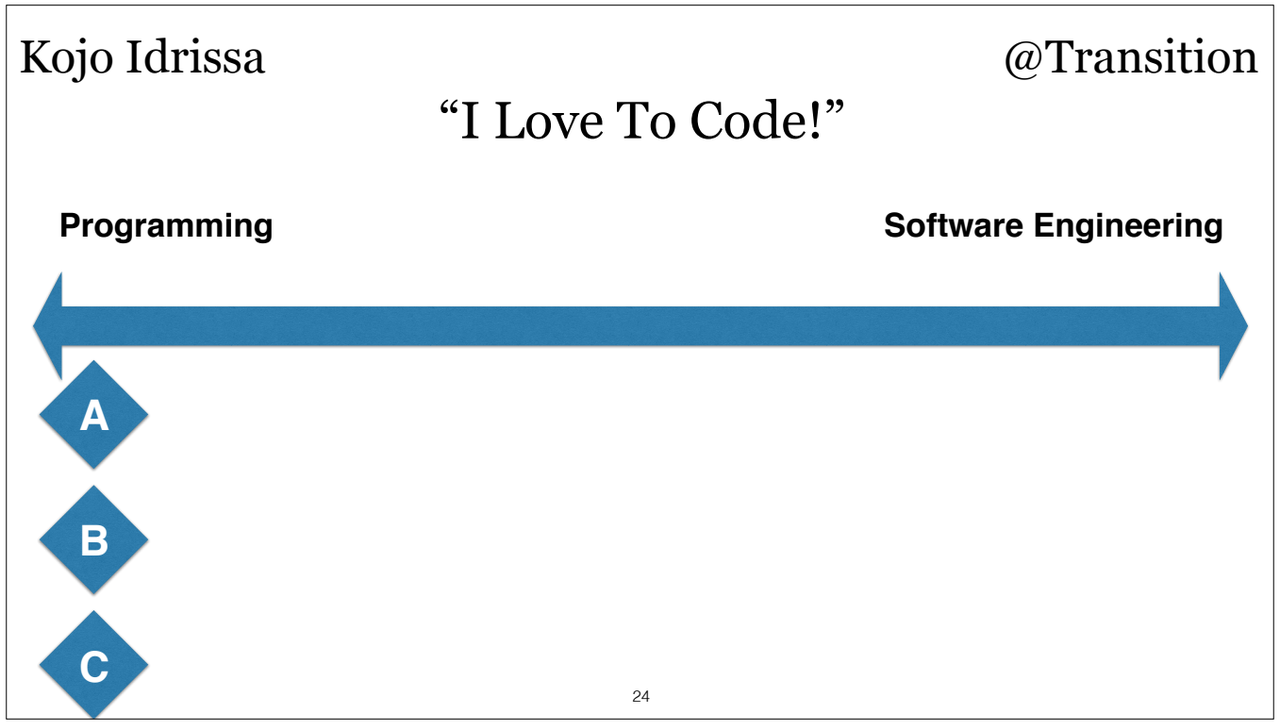


- A: My 1st Spreadsheet project: programming, to SUPPORT current profession; writing code to solve a work problem; I only NEED the required tools.
- B: My 1st Spreadsheet project: adding Git & docs, to SUPPORT my profession; learning SE tools
- C: Twilio project: learning software engineering (Git, Twilio API, tests, deploy to Heroku), but for fun
- D: Current spreadsheet project: Software Engineering AS my profession, internal tool; OUTPUT is the motivation

Kojo Idrissa @Transition

Data-Heavy Professional

- A: Person (Accountant, Geologist, Engineer, Chemist) does data analysis, running code on their laptop to SUPPORT their profession; writing code to solve a work problem
- B: Person learns about Git & how branching/merging can help them, to SUPPORT their profession
- C: Python knowledge increases; no need for more SE tools
    - NO DESIRE to become an SE, even as coding skill increases
- This could be a “data scientist”. Or, it could be someone who just wants to make their job easier.



- A: Person learns to code, using “Think Python”
- B: They LOVE it, and continue learning more Python. Maybe they make games that run on their laptop?
- C: The then move on to writing code that pulls data from web APIs and does cool things with it.
- This person has NO desire to be an SE. They just like the things they can do with code.

Kojo Idrissa @Transition

Common Denominator

- All three consistently use Python as a solution
- All three are potential contributors to the Python ecosystem
- Unless we push Person 2 & 3 away

Again: the ONLY people who'd even CONSIDER contributing to the Python ecosystem are people who see some benefit from Python. This is a necessary, but not sufficient condition.

To move people from potential toward actual contributors, they have to feel like they're WELCOME & INCLUDED in our community, even if they don't want to change professions or act like software engineers.

Most of us have a friend who's a GREAT cook, but if Gordon Ramsey showed up, he'd set their kitchen on fire and call them bad names.

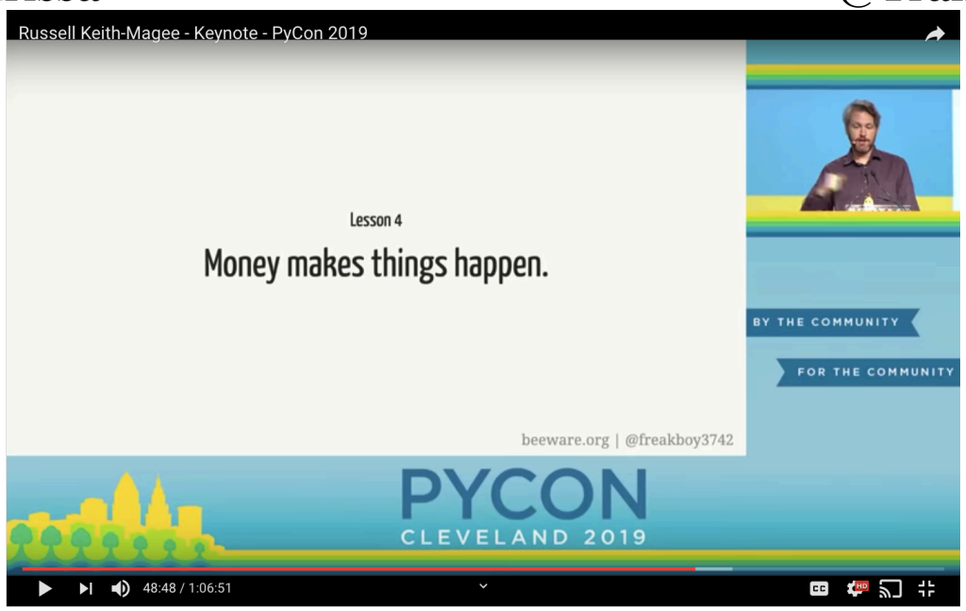
Kojo Idrissa @Transition

RK-M's PyCon 2019 Keynote

- Paying professionals to support the project
- Building a team with contribution skills beyond code

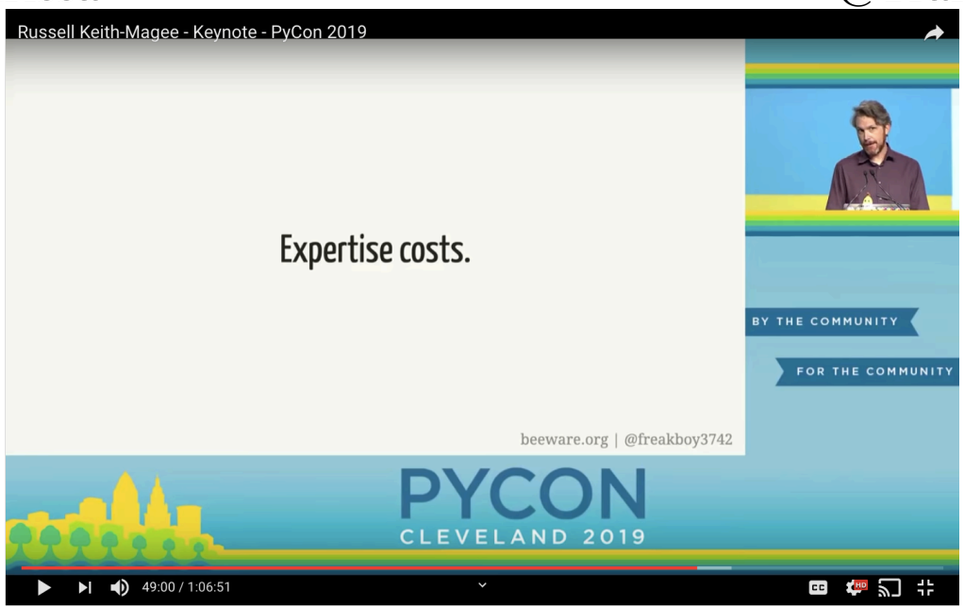
This is an EXCELLENT talk looking at the future of Python. I'm only picking out parts that are most relevant to my talk, but I highly recommend you watch the video.

Kojo Idrissa @Transition



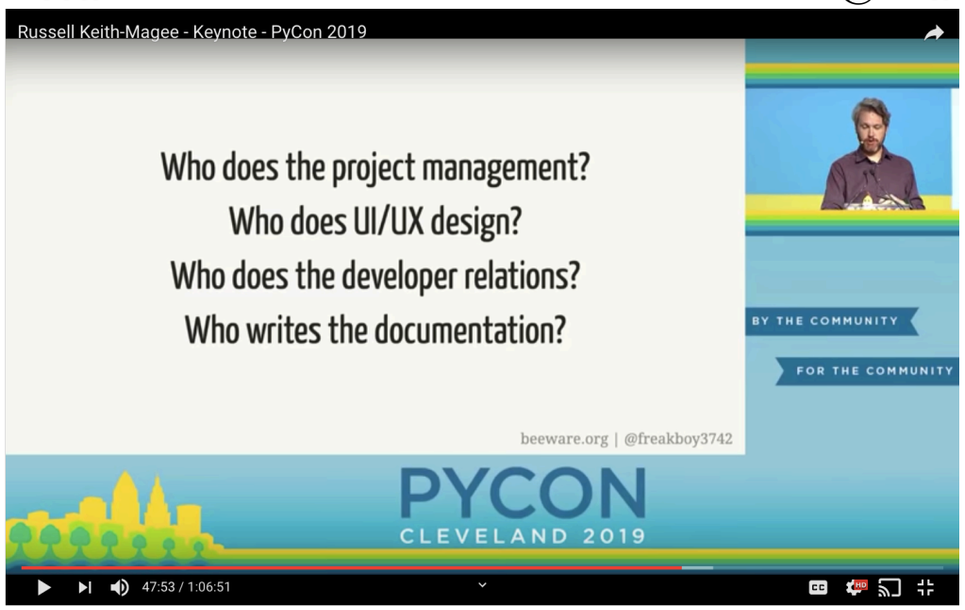
Money DOES make things happen

Kojo Idrissa @Transition



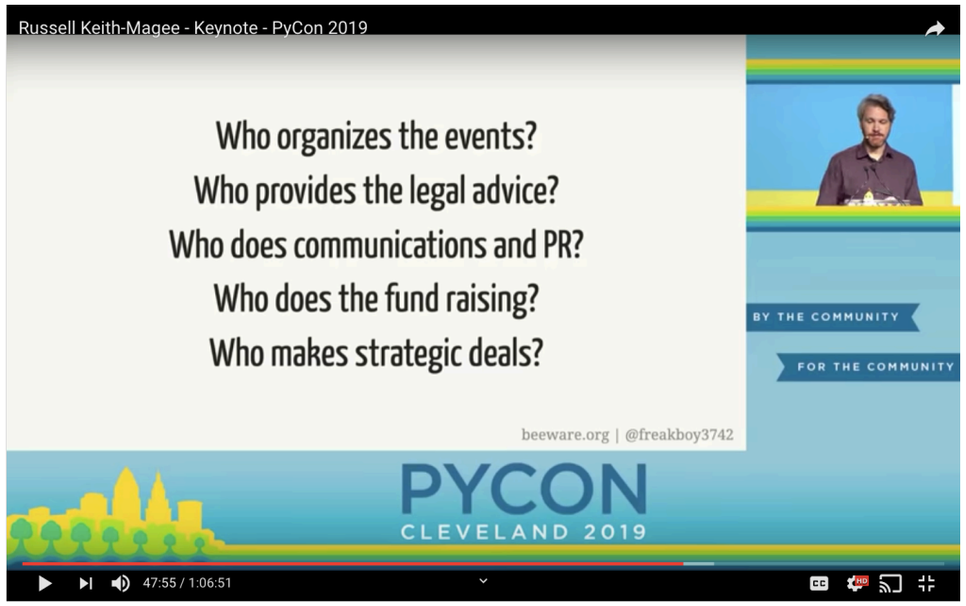
Expertise costs

Kojo Idrissa @Transition



PM, UI/UX, Dev Rel, Docs

Kojo Idrissa @Transition



A Humble Bundle fundraiser for the PSF just ended. Who made THAT deal?

Kojo Idrissa @Transition

RK-M's PyCon 2019 Keynote

Skilled professionals have options: why would they help us if we don't include them?

I've been working on this talk for quite some time. What stood out to me during Russ' talk was, "What separates the Python community from any OTHER potential client for someone with skills we need? Everyone else's money is just as green. But I doubt their doctor-friend never says anything like:
"You used a **Band-Aid** on your cut?!? Why didn't you just use surgical glue? You don't HAVE any?!? NEWB."

Call To Action

- **Recognize**: different styles of coding fit different problem sets
- **Recognize**: Everyone doesn't want to become a software engineer
- **Python Software Engineers**: Know Your Audience

So, what's the point of this talk? What do I want you to do?

If you're talking to someone who's learning to code, and they're stuck and come to you because you're "an expert", don't tell them they need to be using Git or Docker or CI UNLESS THAT'S ACTUALLY WHAT THEY NEED TO SOLVE THEIR CURRENT PROBLEM. And if that IS, explain WHY it's what they need.

Kojo Idrissa @Transition

FHC&Q

- But, everyone COULD benefit from {*insert name of software engineering tool here*}.
- “Do we want people in our community who aren’t willing to learn ‘hard’ things?”
- “So, just be nice to the newbies?”

Frequently Heard Comments
But is the cost of learning that tool TRULY less than the benefits of using it? And does the person you’re speaking to REALLY need it? Or is it a tool YOU’VE become so comfortable with, you think everyone should be using it?

Yes, b/c everyone doesn’t need to be an SE. And, they may be bringing skills we WON’T have w/ a primary software engineering focus. Also: we’re LITERALLY the beggars who can’t be choosers.

First: stop using that term. It’s something computer nerds (when we weren’t cool) came up with to FEEL cool (I’m old). Instead, use your skills and knowledge to HELP the people who come to you for advice, not to berate them.

Summary/Conclusion

- Python is a volunteer-run FLOSS project
- People can use it w/o ever contributing
- New Members! New Contributors?

All Software Engineers are Programmers, but all Programmers aren’t (and may not want or need to be) Software Engineers.

Kojo Idrissa @Transition

Questions/Comments?

- @Transition on Twitter
- kojoidrissa.com
- github.com/kojoidrissa/pygotham_2019

I’m VERY interested in people’s feedback. The first time I gave this talk, it became fairly contentious. Reach me on Twitter or here at the conference.