a Python future
Armin @mitsuhiko Ronacher
what can I tell you?
let me tell you about a future
i can't tell you what the future will be, but here is what I wish it to become
and the future will be different
Thanks for all the support (email and Twitter). I'm overwhelmed by the responses and won't be replying to most emails in person (except from core devs) but it's much appreciated. I'm still going to be around in the background!

1:07 AM - 13 Jul 2018

848 Retweets 5,413 Likes
where are we now?
and how we **CAME HERE**
people **LOVE** python
Stack overflow most **LOVED**

<table>
<thead>
<tr>
<th>Language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rust</td>
<td>78.9%</td>
</tr>
<tr>
<td>Kotlin</td>
<td>75.1%</td>
</tr>
<tr>
<td>Python</td>
<td>68.0%</td>
</tr>
</tbody>
</table>
### Most Wanted

<table>
<thead>
<tr>
<th>Language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python</td>
<td>25.1%</td>
</tr>
<tr>
<td>JavaScript</td>
<td>19.0%</td>
</tr>
<tr>
<td>Go</td>
<td>16.2%</td>
</tr>
</tbody>
</table>
but what about the **future**?
let's make a DETOUR
rust is **hard**
rust is complex
so why do people **LOVE** Rust?

(and what does it mean for Python?)
the packaging story
the DISTRIBUTION story
RELIABILITY and TRUST
a clear BACKWARDS COMPATIBILITY story
the community
rust tells us a story
rust's values are \textbf{TRANSPARENT}
the Rust programming language is fundamentally about **empowerment**: no matter what kind of code you are writing now, Rust empowers you to **reach farther**, to **program with confidence** in a **wider variety of domains** than you did before.
it's okay to create something complex
if using it is EASY
our approach so far
we succeed **DESPITE** our process
the wider community vs the core developers
the packaging story
distribution
typing
WHY?
we're afraid of complexity
we work against each other
how do we prevent this from happening?
we need to work together
we need a vision
and that vision could be bold
working groups?
pay developers?
embrace black and lint name violations?
TypeScript for Python?
py_modules and package.json?
non indexable UTF-8 strings?
phase out cabi for ffi?
load wasm modules?
strip stdlib down?
multi version import system?
simplify object model?
cross compile to wasm?