THINKING OUTSIDE THE BOX
Hello, I'm Armin!
I do Computers - with Python.
Currently at Fireteam / Splash Damage.

We do Internet for Pointy Shooty Games.
the box is comfortable
the comfort is dangerous
the 9 dots puzzle
(and things of similar nature)
connect the dots

using exactly 4 straight lines, without retracing or removing one's pen from the paper
connect the dots using exactly 4 straight lines, without retracing or removing one's pen from the paper.
i know it's cliché
the six thinking hats
the six thinking hats

This guy
“the majority is always wrong:"

congratulations;
you're now a cynical asshole
1 a change of environment
computer games
this graph is not very scientific

- online infrastructure
- Game code
* this graph is not very scientific

- online infrastructure
- Game code

Python!!!11
community influences your thinking
C++ good;
Scripting Languages Bad
... if it takes you 30 minutes to do a one line change then you obviously would not want to have runtime type checks ...
it's too easy to dismiss something on fringe or outdated experience alone
never underestimate how much your environment/community influences you
never underestimate how much your environment/community influences you

(unfortunately that also includes things like “the GIL is not a problem”)
asking the right questions
the wrong questions are easy to spot on others.
how do I do something after return render_template(...) if I don't want to register teardown_request for all requests.
he was looking for celery / message queue
first expectation was that the user wanted to change the HTTP response
* `teardown_request` would not have worked anyways
better:

“I don't want my user to wait while I do some processing on his data”
the box was Flask
how to ask the right questions
* assume you already started out wrong
* describe the *actual* problem
"How do I use Websockets with Flask?"
better:

“How do I notify my user about changes with low latency.”
question leaves room for the answer
(a) Server-sent events via WSGI

(b) application <-> redis <-> persistent connection server

bit.ly/pypush
Jump on IRC
help out other people
questioning the right things
the worst parts in my libraries are the ones where I took the design from elsewhere
it’s not because I know better...
• most things have some design behind
• as people copy it, the original design gets obscured and forgotten
• the original design might no longer apply
• starting something new?
• question everything!
not with the intention of proving existing design wrong;

with the intention of understanding it.
paradigm shifts
Many times we don't even realize that things were an example of thinking outside the box.

(a) "echo" -> Request/Response objects

(b) Interactive Interpreters
That's also what makes it hard to find examples now ...
every idea is a rehash

don't get too excited when you feel all "obviously ..."
sometimes all that's necessary is transposing a concept from one industry to the other.
5 interesting examples
Mill Processor:

- Basic Block: One entry, one exit.
- Break instruction bundles in half
- Two decoders, one moves left from EBB entry point, one moves to the right
Mill Processor:

- Basic Block: One entry, one exit.
- Break instruction bundles in half
- Two decoders, one moves left from EBB entry point, one moves to the right

two independent units, two separate caches
High-level Queues for Request Handling

Stateful Frontend Servers

Stateless Workers
* Sending around signed data
* Cookies
* Access/Refresh Tokens
* Activation Links
* Offers
• The Rust Programming Language
• Memory ownership tracking

(and otherwise just steal from C++, C, Python, Ruby, Haskell and Scheme)
* Spotify's Native/Web Bridge
* spawns HTTP server on localhost:XXXXX
* provides OAuth bridge
* JavaScript authenticates with local server, sends commands and retrieves updates.
That's it;
Now ask questions

@mitsuhiko
lucumr.pocoo.org/talks
gittip.com/mitsuhiko
calibration slide