

So You Wanna Be A Tech Speaker



About me

- Data Science Educator
- PyData organizer
- Tech Speaker

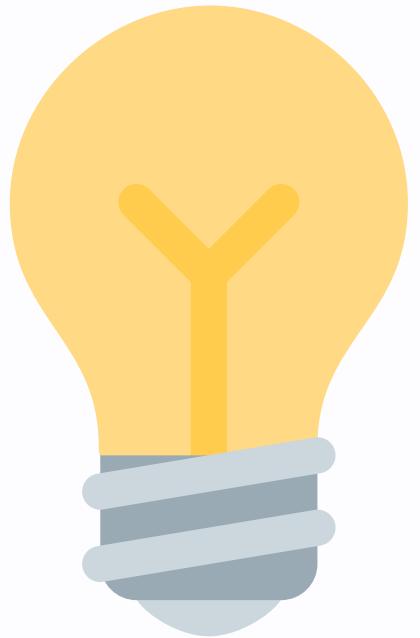




Outline

- Idea
- CFP
- Structure
- Delivery





idea

What is a tech talk?

What *isn't* a tech talk?

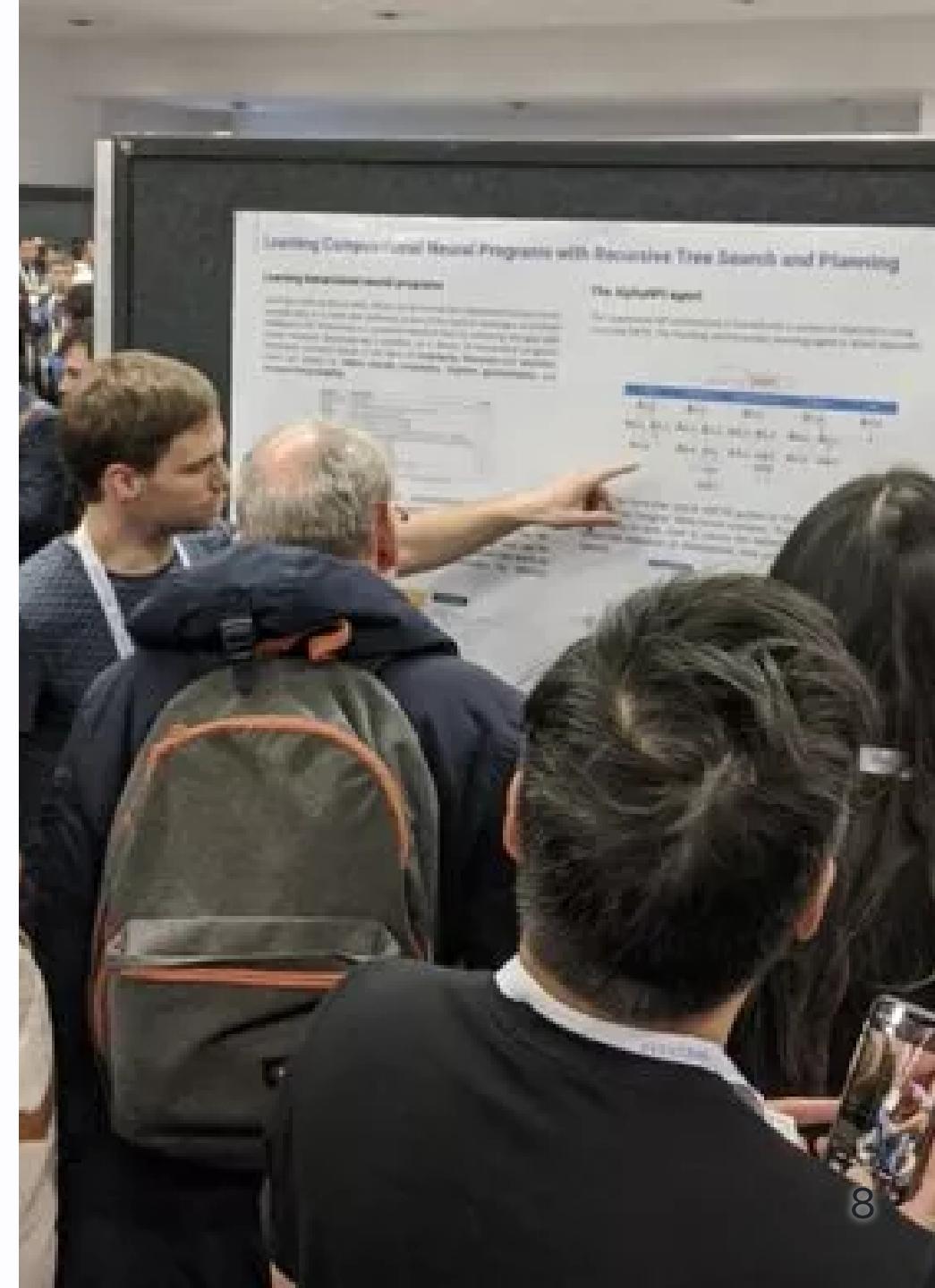
X What *isn't* a tech talk?

- It is not a *lecture*



X What *isn't* a tech talk?

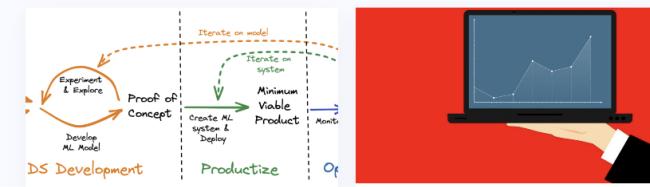
- It is not a *lecture*
- It is not an *academic talk*





✗ What *isn't* a tech talk?

- It is not a *lecture*
- It is not an *academic talk*
- It is not a *blog post*



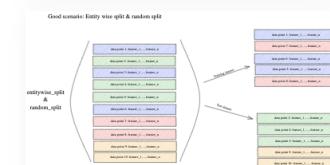
MLOPS
MLOps: why and how to build end-to-end product teams

Daniel Willemsen
on 07 November 2022 →



DropBloX: Coding Challenge at PyCon DE & PyData Berlin 2022

Yke Rusticus
on 27 July 2022 →



Optimizing TopK queries in DataFusion

Daniël Heres
on 28 September 2022 →



AZURE / DATA ENGINEERING / PYTHON
adfPy: an intuitive way to build data pipelines with Azure Data Factory

Daniel van der Ende
on 25 July 2022 →



DATA SCIENCE AND AI / MLOPS / OPEN SOURCE SOFTWARE / PYTHON

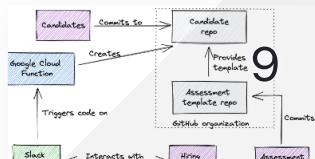
How Streamlit will help you get your machine learning products used

Daniel Willemsen
on 01 August 2022 →



PYTHON
Protocols in Python: Why You Need Them

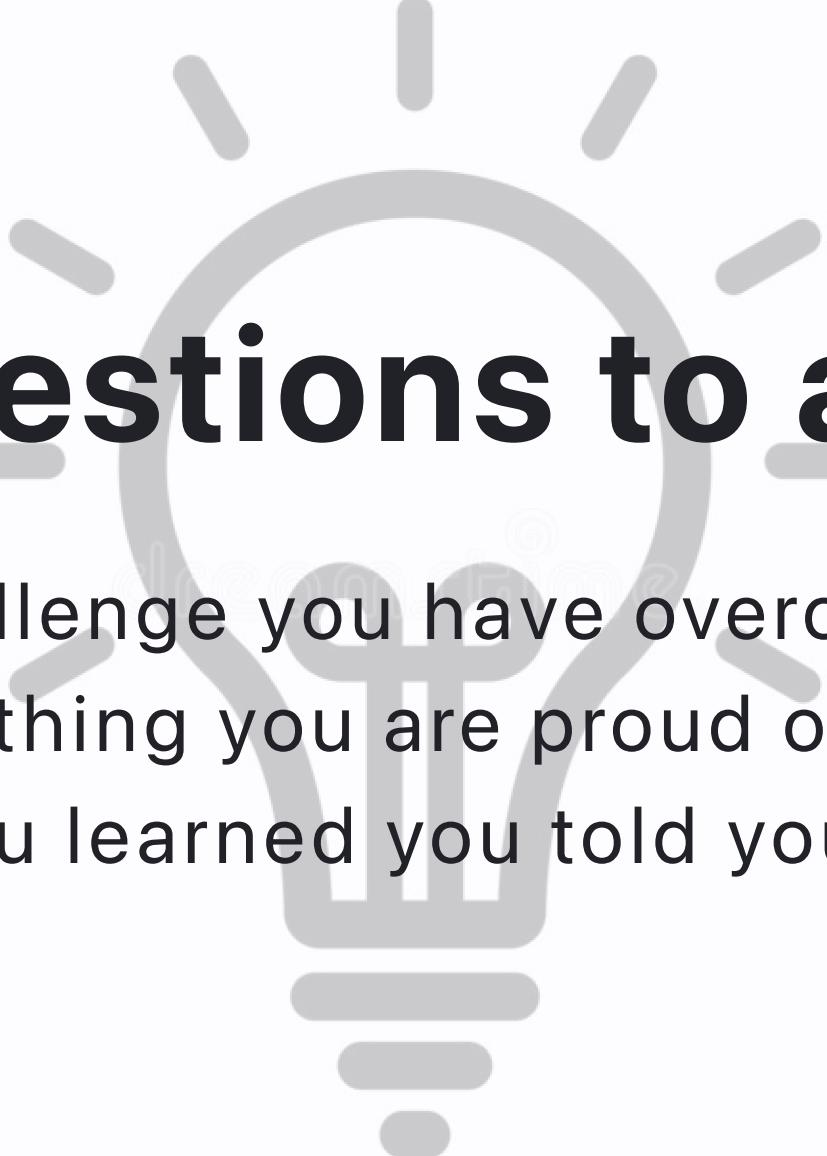
Rogier van der Geer
on 25 July 2022 →



✓ What *is* a tech talk then?

- It is *informative*
- It is a *story*
- It is *personal*





Questions to ask

- What is a challenge you have overcome?
- What is something you are proud of?
- What have you learned you told your friend about?

**What is something only you
can tell me about?**

CALL FOR PROPOSALS

CLICK THE LINK

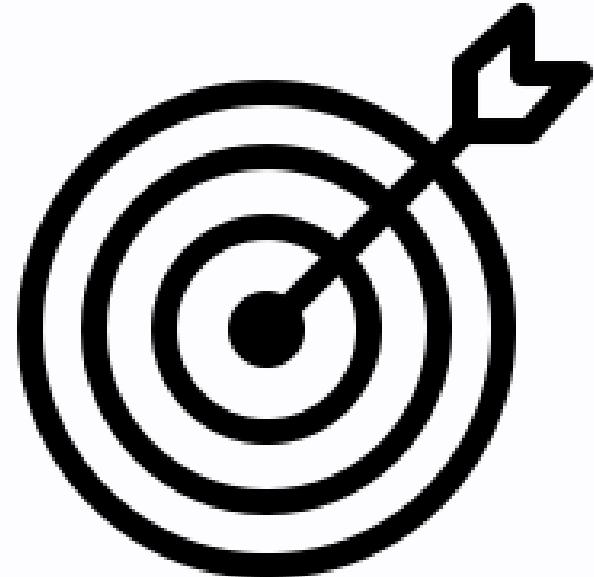


The Title



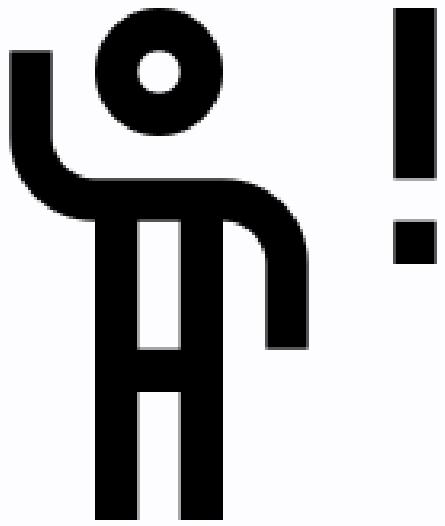
Clear titles

- Data Validation for Data Science
- Building Successful Data Science Products
- Feature Engineering for Time Series Forecasting
- Clean Architecture: How to structure your ML projects to reduce technical debt



Clickbait titles

- Do we really need Data Scientists?
- Stupid Things I've Done with Python
- Why do I need to know Python? I'm a pandas user..
- "Off with their I/Os!" - or how to contain madness by isolating your code



Content





Proposal content

- **What** is the talk about?
- **Why** is the topic interesting?
- **Who** is it for?
- **Type** of talk and **tone**?
- **Key takeaways**



Example Introduction

“

Engulfed in a tedious refactoring of your code, you're adding the 7th layer of mocks to a test when you realise something must have gone wrong somewhere, but what? You've written readable code, split into functions and classes to avoid long chunks of code, and yet, every time, you end up with hardly testable code, a test suite that runs for hours, functions with seventeen arguments, and you wonder if it's you mocking the code or the code mocking you.

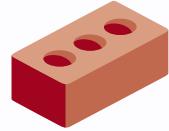
”

The intended audience is intermediate to senior data scientists, who have already, or will soon encounter problems with testing, maintaining or expanding a growing codebase.

This talk will help you understand the benefits of good architecture, with a focus on isolating your I/O (inputs/ outputs) and other third-party dependencies, and guide through how to achieve it in practice, from simpler to more complex cases. I will present good practices coming from software engineering, with a focus on applying them to a data science context.

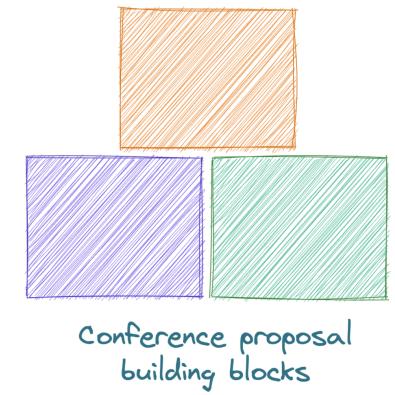
Outline: - (2 min) Intro - (4 min) Functional programming ideas - (6 min) Isolating I/O with a clean architecture (“onion” architecture) - (5 min) Benefits in terms of testing and maintainability - (8 min) How to isolate third-party dependencies using dependency injection and abstraction layers - (5 min) QA

I cannot promise that the Liskov substitution principle won’t be mentioned, but I will do my best to make it clear and understandable.



Proposal elements

- Engaging abstract
- *“Show, don’t tell”* target audience
- Clear takeaway
- A (rough) intended outline

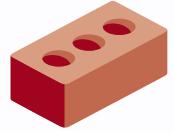


Look up previous editions of the conference you're applying for and read the descriptions of the talks that got accepted!

*Why are you qualified to
give this talk?*

Talk structure





Talk elements

- Technical part
- Context (*story!*)
- Lessons learnt





Talk structure

- Beginning: challenge
- Middle: technical part
- End: lessons learnt



The Problem



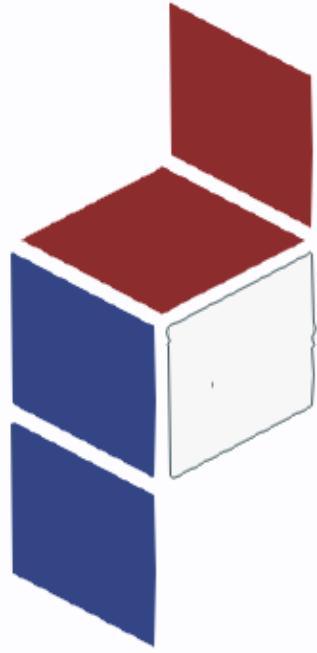
End with Call to Action





Delivery

- It's okay to be nervous
- Practice out loud
- But not too much!
- Show as you tell
- Don't be afraid of questions
- Have fun :)



PyData^{xx}
Amsterdam

Feel free to reach out!

-  Marysia Winkels
-  marysia.nl
-  hello@marysia.nl

