

# Today

# Today

- What it means to be a multi-lingual or full-stack developer, based on my experience and observations

# Today

- What it means to be a multi-lingual or full-stack developer, based on my experience and observations
- Some insight about why you may want to follow different paths, including the pros/cons, as well as how you might do it.

# Who

# Who

- me: Made a career out of not wanting to pigeon-hole myself

# Who

- me: Made a career out of not wanting to pigeon-hole myself
  - [https://twitter.com/mike\\_moran](https://twitter.com/mike_moran)

# Who

- me: Made a career out of not wanting to pigeon-hole myself
  - [https://twitter.com/mike\\_moran](https://twitter.com/mike_moran)
  - <https://www.houseofmoran.com/>





# What: Spotting Patterns

# What: Spotting Patterns

- An illustrative example to get us started

# What: XPath Example

```
<ul>  
  <li id="2">Bloop</li>  
  <li id="1">Feep</li>  
</ul>
```

```
//li[@id = 2]//text() -> Bloop
```

```
//li[position() = last()] -> <li id="1">Feep</li>
```

# What: SQL Example

```
SELECT CustomerName, Country  
FROM Customers  
WHERE City = 'Paris'
```

| CustomerName         | Country |
|----------------------|---------|
| Paris spécialités    | France  |
| Spécialités du monde | France  |

# What: Rust Example

```
fn main() {  
    let a = [1, 0, 3];  
  
    let iterated : Vec<_> =  
        a.iter()  
        .filter(|&i| *i != 0 )  
        .collect();  
  
    dbg!(iterated);  
}
```

iterated = [1, 3]

# What

# What

- XPath  $\leftrightarrow$  SQL  $\leftrightarrow$  Rust?

# What

- XPath  $\leftrightarrow$  SQL  $\leftrightarrow$  Rust?
- What's different / similar?



**Any Questions?**

**Why: lower-level**

# Why: lower-level

- Spotting and re-using Patterns

# Why: lower-level

- Spotting and re-using Patterns
  - know the layout of the room

# Why: lower-level

- Spotting and re-using Patterns
  - know the layout of the room
- Problem-focussed

# Why: lower-level

- Spotting and re-using Patterns
  - know the layout of the room
- Problem-focussed
- Choosing the right language/library for the job

# Why: lower-level

- Spotting and re-using Patterns
  - know the layout of the room
- Problem-focussed
- Choosing the right language/library for the job
  - "Library-first Programming"

**Why: higher-level**



# Why: higher-level

- Gives Perspective: "All Computers are Shit" ( :- )

# Why: higher-level

- Gives Perspective: "All Computers are Shit" ( :- )
- Bootstrap in X, Scale out in Y

# Why: higher-level

- Gives Perspective: "All Computers are Shit" ( :- )
- Bootstrap in X, Scale out in Y
- Building a Robust Company

**Any Questions?**

# How

# How

- Try experiments

# How

- Try experiments
  - example: Svelte, React

# How

- Try experiments
  - example: Svelte, React
- Characterise options



# How

- Try experiments
  - example: Svelte, React
- Characterise options
  - "Pigeon-hole test": power vs culture

# How

- Try experiments
  - example: Svelte, React
- Characterise options
  - "Pigeon-hole test": power vs culture
- Try to "feel the grain"

# How

- Try experiments
  - example: Svelte, React
- Characterise options
  - "Pigeon-hole test": power vs culture
- Try to "feel the grain"
  - "this would be easier / better in Language X / Layer Y"

# How

- Try experiments
  - example: Svelte, React
- Characterise options
  - "Pigeon-hole test": power vs culture
- Try to "feel the grain"
  - "this would be easier / better in Language X / Layer Y"
- Be a librarian / router for others, learn on the way

# Why Not

# Why Not

- Never the "best" at anything

# Why Not

- Never the "best" at anything
- Hard for other people to understand what you do

# Why Not

- Never the "best" at anything
- Hard for other people to understand what you do
  - you are the Elephant surrounded by the Blind men



# Why Not

- Never the "best" at anything
- Hard for other people to understand what you do
  - you are the Elephant surrounded by the Blind men
- Can be easy to get distracted by too many options

# Why Not

- Never the "best" at anything
- Hard for other people to understand what you do
  - you are the Elephant surrounded by the Blind men
- Can be easy to get distracted by too many options
- Can get confusing technically

# Why Not

- Never the "best" at anything
- Hard for other people to understand what you do
  - you are the Elephant surrounded by the Blind men
- Can be easy to get distracted by too many options
- Can get confusing technically
  - example: C# `yield` != Python generator

**Any Questions?**

# Closing

# Closing

- This is an option, and not a recommendation for everyone.

# Closing

- This is an option, and not a recommendation for everyone.
  - Being a specialist is still totally fine

# Closing

- This is an option, and not a recommendation for everyone.
  - Being a specialist is still totally fine
- However:



# Closing

- This is an option, and not a recommendation for everyone.
  - Being a specialist is still totally fine
- However:
  - please be proficient in at least 2 languages; 2 is very different to 1

# Closing

- This is an option, and not a recommendation for everyone.
  - Being a specialist is still totally fine
- However:
  - please be proficient in at least 2 languages; 2 is very different to 1
  - Avoid being a "Foo" programmer; drop your prejudices (something I still struggle with)

# Contact details

# Contact details

- For any questions / suggestions:

# Contact details

- For any questions / suggestions:
  - [https://twitter.com/mike\\_moran](https://twitter.com/mike_moran)

# Contact details

- For any questions / suggestions:
  - [https://twitter.com/mike\\_moran](https://twitter.com/mike_moran)
  - [mike@houseofmoran.com](mailto:mike@houseofmoran.com)

# Contact details

- For any questions / suggestions:
  - [https://twitter.com/mike\\_moran](https://twitter.com/mike_moran)
  - [mike@houseofmoran.com](mailto:mike@houseofmoran.com)
  - <https://www.houseofmoran.com/>