Dr. Florian Wilhelm
Data Scientist@inovex
Contributor to Pandas, Scikit-Learn, Scipy etc.
Creator of PyScaffold

@FlorianWilhelm
FlorianWilhelm
florianwilhelm.info
Outline

1. Motivation & Concept
2. Examples
3. Math Riddle
Motivation

House-warming party with your friends
Motivation

What is the actual task?

- clean up
- find easy receipe
- put boxes in basement
- buy more beer
- invite friends
Level of Abstraction
Right level of abstraction given a task

What is needed to describe the problem?

map & reduce
Imperative vs. Declarative
How vs. What

**imperative**
*how*

- over-specification,
- detailed instructions, ...

**declarative**
*what*

- separation of concerns, single level of abstraction, ...

depending on the level of abstraction
Leaky Abstractions

Law of Leaky Abstractions by Spolsky:

“All non-trivial abstractions, to some degree, are leaky.”
Outline

1. Motivation & Concept
2. Examples
3. Math Riddle
Example 1: List Comprehensions

List of squared number from 1 to 10

**imperative:**

```
result = []
for i in range(1, 11):
    result.append(i**2)
```

**declarative:**

```
result = [i**2 for i in range(1, 11)]
```
Example 2:
Dispatching with respect to some argument

Imperative:

```python
def dispatch(arg, value):
    if arg == 'optionA':
        function_a(value)
    elif arg == 'optionB':
        function_b(value)
    elif arg == 'optionC':
        function_c(value)
    else:
        default(value)
```
Example 2: Dictionaries

Dispatching with respect to some argument

Declarative:

```python
dispatch = {'optionA': function_a,
            'optionB': function_b,
            'optionC': function_c}
dispatch.get(arg, default)(value)
```
Example 3: Sets

Find Plagiarism

How many sentences of work A are equal to my work B?

Set Theory

\[ \text{result} = A \& B \]
Example 4: Configuration Files

Python modules vs markup languages

vs

YAML
Outline

1. Motivation & Concept
2. Examples
3. Math Riddle
Math Riddle

**horizontal:**
A: digit sum of horizontal C,
C: prime number,
E: palindrome,
G: multiple of the backward number of horizontal A,
...

**vertical:**
All numbers are square numbers.

Datalog

Features

• declarative logic programming
• subset of Prolog
• query language for deductive DBs
• other use-cases: security, data integration, information extraction, networking, program analysis etc.

PyDatalog: https://sites.google.com/site/pydatalog/home
PyDatalog

Rules & Facts

Is X a square number?

\[
squared(X) \Leftarrow (\text{math.sqrt}(X).\text{is_integer}() == \text{True})
\]

Read the leftmost \(\Leftarrow\) as if

Is X divisible by Y?

\[
divisible(X, Y) \Leftarrow (\text{divmod}(X, Y)[1] == 0)
\]
Is X prime?

```
+prime(2)
+prime(3)
prime(X) <= (X > 3) & ~divisible(X, 2) & ~factor(X, 3)
factor(X, Y) <= divisible(X, Y)
factor(X, Y) <= (Y+2 < math.sqrt(X)) & factor(X, Y+2)
```
Map digits to number

\[
\begin{align*}
\text{num}[A, B] &= 10 \times A + B \\
\text{num}[A, B, C] &= 10 \times \text{num}[A, B] + C \\
\text{num}[A, B, C, D] &= 10 \times \text{num}[A, B, C] + D \\
\text{num}[A, B, C, D, E] &= 10 \times \text{num}[A, B, C, D] + E \\
\text{num}[A, B, C, D, E, F] &= 10 \times \text{num}[A, B, C, D, E] + F
\end{align*}
\]
Math Riddle

Leaky Abstraction

Keep the number of solutions low at all times
Math Riddle

Upper left corner

A2, A3 are digits from [1...9] and number A2 A3 is prime

```
A2.in_(range(1, 10)) & A3.in_(range(1, 10)) & prime(num[A2, A3])
```
Math Riddle

Solution

Querying the knowledge base:

```
print(riddle([(A0, A1, A2, A3, A4, A5), (B0, B1, B2, B3, B4, B5),
              (C0, C1, C2, C3, C4, C5), (D0, D1, D2, D3, D4, D5),
              (E0, E1, E2, E3, E4, E5), (F0, F1, F2, F3, F4, F5)]))
```

Solution:

```
A0  |  A1  |  A2  |  A3  |  A4  |  A5  |  B0  |  B1  |  B2  |  B3  |  B4  |  ...
--- | ---- | ---- | ---- | ---- | ---- | ---- | ---- | ---- | ---- | ---- | ---
1   | 1    | 4    | 7    | 2    | 2    | 4    | 2    | 9    | 5    | 5    | ...
```
Other Applications

NixOS

Lugi

TensorFlow
Summary

Advantages of Declarative Programming

- improved readability of our code
- reduced number of errors
- increased performance
- separation of concerns

“Declarative programming means finding the right abstraction level describing the problem“
Thanks for your Attention!

Questions?

Find more details under http://florianwilhelm.info/2017/07/declarative_thinking_and_programming/