Class vs. Object-Oriented Programming

S. Ducasse, L. Fabresse, G. Polito, and P. Tesone

http://www.pharo.org
Goals

- Think about object-oriented programming
- Understand that class programming is not object-oriented programming
- Favor objects!
Sometimes we get class-based programming design:

- Classes are used as data holder
- Instances of such class would share the same data
- Require a new class to represent a new instance or configuration of data
- No real instance specific state
Studying a class hierarchy

- **Changer**
  - commandClass
  - order
  - multiFormClass

- **BackgroundChanger**
  - commandClass
  - order
  - multiFormClass

- **PositionChanger**
  - commandClass
  - order
  - multiFormClass

- **IdentifierChanger**
  - commandClass
  - order
  - multiFormClass

- **CommandBackground**
  - commandClass
  - ^ CommandBackground
  - multiFormClass
  - ^ FormColor

- **CommandIdentifier**
  - commandClass
  - ^ CommandIdentifier
  - multiFormClass
  - ^ FormText

- **CommandPosition**
  - commandClass
  - ^ CommandPosition
  - multiFormClass
  - ^ FormPoint

- **CommandChanger**
  - commandClass
  - order
  - multiFormClass

- **Order**
  - order
  - ^ 10
Analysis

- Data-oriented classes
- Static: We **have to create** a new class for each new changer
- A class **represents one** instance! Fishy
- A class state should describe instance **shape** not instance values
- Each instance can have a different state
Compare with instance-based design
Analysis

Pros:

- Just create instances
- Can represent multiple and different configurations

Changer new
command: CommandPosition;
multiFormClass: PropertyDualInput ;
....
yourself
With subclasses

```plaintext
initialize
  order := 10.
  multiFormClass := DualInputPresenter.
  commandClass := CommandPosition

PositionChanger
  initialize

Changer
  commandClass
  order
  multiFormClass

aChanger (position)
aChanger (identifier)
aChanger (background)
```
Need a discovery mechanism

- Class-based
  - Annotation, hierarchy query, explicit registration
- Instance-based
  - Need to store instances somewhere
  - Explicit registration
Conclusion

- When you need a new class to represent a new instance, this is fishy
- A class describes the shape of instance not their values
- **Favor** instances over classes
Advanced Object-Oriented Design and Development with Pharo

A course by
S. Ducasse, L. Fabresse, G. Polito, and P. Tesone

Except where otherwise noted, this work is licensed under CC BY-NC-ND 3.0 France
https://creativecommons.org/licenses/by-nc-nd/3.0/fr/