

Class vs. Object-Oriented Programming

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



Goals

- Think about object-oriented programming
- Understand that class programming is not object-oriented programming
- Favor objects!



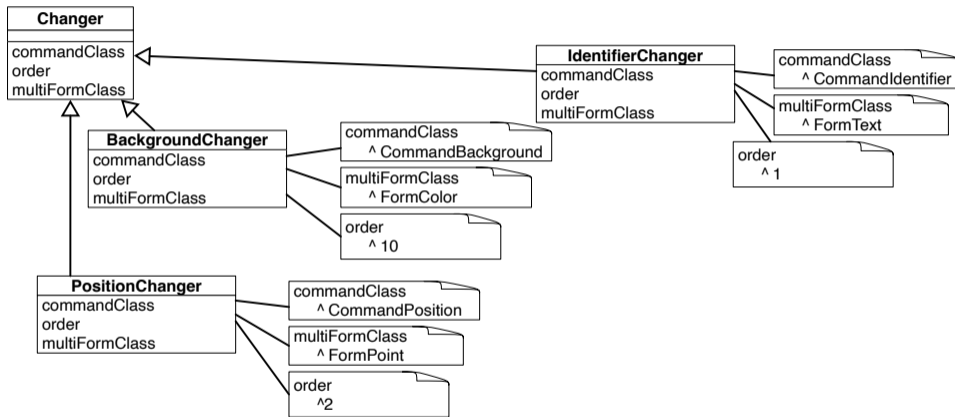
Class-based programming design

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

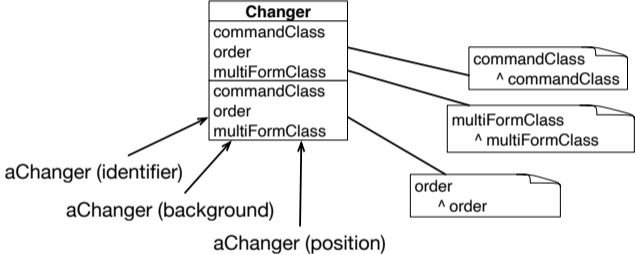


Analysis

- 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

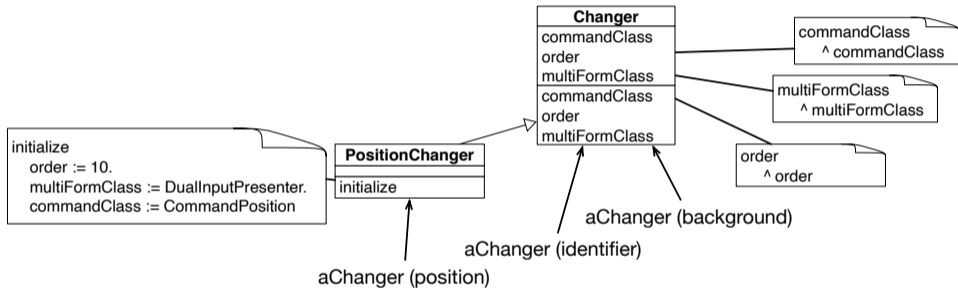
Pros:

- Just create instances
- Can represent multiple and different configurations

```
Changer new  
  command: CommandPosition;  
  multiFormClass: PropertyDualInput ;  
  ....  
  yourself
```



With subclasses



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



Produced as part of the course on <http://www.fun-mooc.fr>

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/>