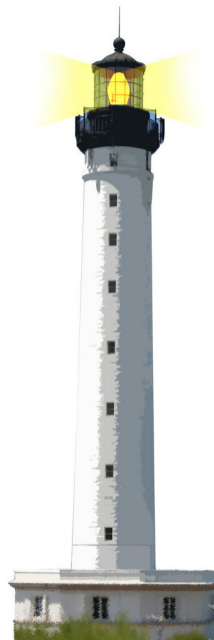


# Fat classes are bad

a.k.a. a large class vs. a class hierarchy

S. Ducasse



# Goal

- Remember the *Implementing Not* Lectures
- Transform Self Type Check Reengineering pattern



## Example: BibTeX

```
@inproceedings{Scha03a,  
  author = {N. Schaerli and S. Ducasse and O. Nierstrasz and A. P. Black},  
  title = {Traits: Composable Units of Behavior},  
  booktitle = {ECOOP},  
  publisher = {Springer Verlag},  
  year = {2003}}
```

```
@article{Teso20b,  
  author = {P. Tesone and S. Ducasse and G. Polito and L. Fabresse ...},  
  title = {A new modular implementation for Stateful Traits},  
  journal = {Science of Computer Programming}, ...}
```

```
@book{Duca22a,  
  author = {S. Ducasse and G. Rakic and ...},  
  title = {Pharo by Example},  
  publisher = {Keepers of the lighthouse},  
  url = {http://books.pharo.org}}
```

# Example: BibTeX

- Different entries: inproceedings, book, article, techreport, phd....
- Different fields: key, year, institution, booktitle, title...



## A single class in Citezen :(

```
CZScoped << #CZEntry
  slots: { #type . #key . #fields };
  sharedVariables: { #ExtendedFieldKeys . #FieldKeysToRemove };
  sharedPools: { CZFieldPool };
  package: 'Citezen-Model'
```



# isSomething methods

```
isArticle  
  ^ self type = #article
```

```
isBook  
  ^ self type = #book
```

- It is always worth to have a look at users of these isSomething methods.



## Another example: CZField

```
CZScoped << #CZField  
  slots: { #key . #value };  
  package: 'Citezen-Model'
```

```
CZField >> isDoi  
  ^ key = #doi
```



# Consequence

```
CZField >> visitField: aField
```

```
self outputStream nextPutAll: '<span class=""', aField key, '>'.  
aField isPDF
```

```
ifTrue: [outputStream nextPutAll: '<a href=""'.  
aField isURL
```

```
ifTrue: [outputStream nextPutAll: '<a href=""'.  
aField isDoi
```

```
ifTrue: [outputStream nextPutAll: '<a href="https://doi.org/'.  
aField dispatchVisitor: self.  
aField isURL
```

```
ifTrue: [outputStream nextPutAll: "">URL</a>'].  
aField isDoi
```

```
ifTrue: [outputStream nextPutAll: "">DOI</a>'].  
aField isPDF
```

```
ifTrue: [outputStream nextPutAll: "">PDF</a>'].  
outputStream nextPutAll: '</span>'.  
aField isPDF
```





# When using a Fat class

Not having a class hierarchy:

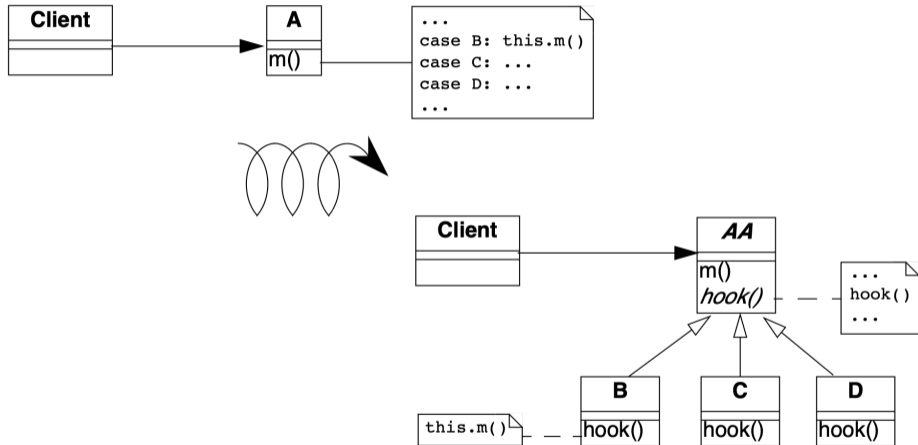
- Not **possible to dispatch** logic because there is **only one** class
- Simple extensions require **many changes** to the conditional code
- Difficult subclassing **without duplicating/adapting** the methods containing the conditional code
- Adding a new behavior always **results in changes** to the same set of methods and always results in adding a new case to the conditional code.
- Remember the Implementing Not Lectures



# Two kinds of type checking conditionals

- Self type checks
  - check some internal values to invoke self methods
  - not that bad
- Client type checks
  - violate **do not ask, tell**
  - defeat late binding and inverse the flow of control
  - not necessary in a large class

# Transform self type transformation



# Conclusion

- Favor dispatch
- Create classes to encapsulate knowledge
- **Do not ask, tell**



A course by

S. Ducasse, G. Polito, and Pablo 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/>