HOCHSCHULE DER MEDIEN

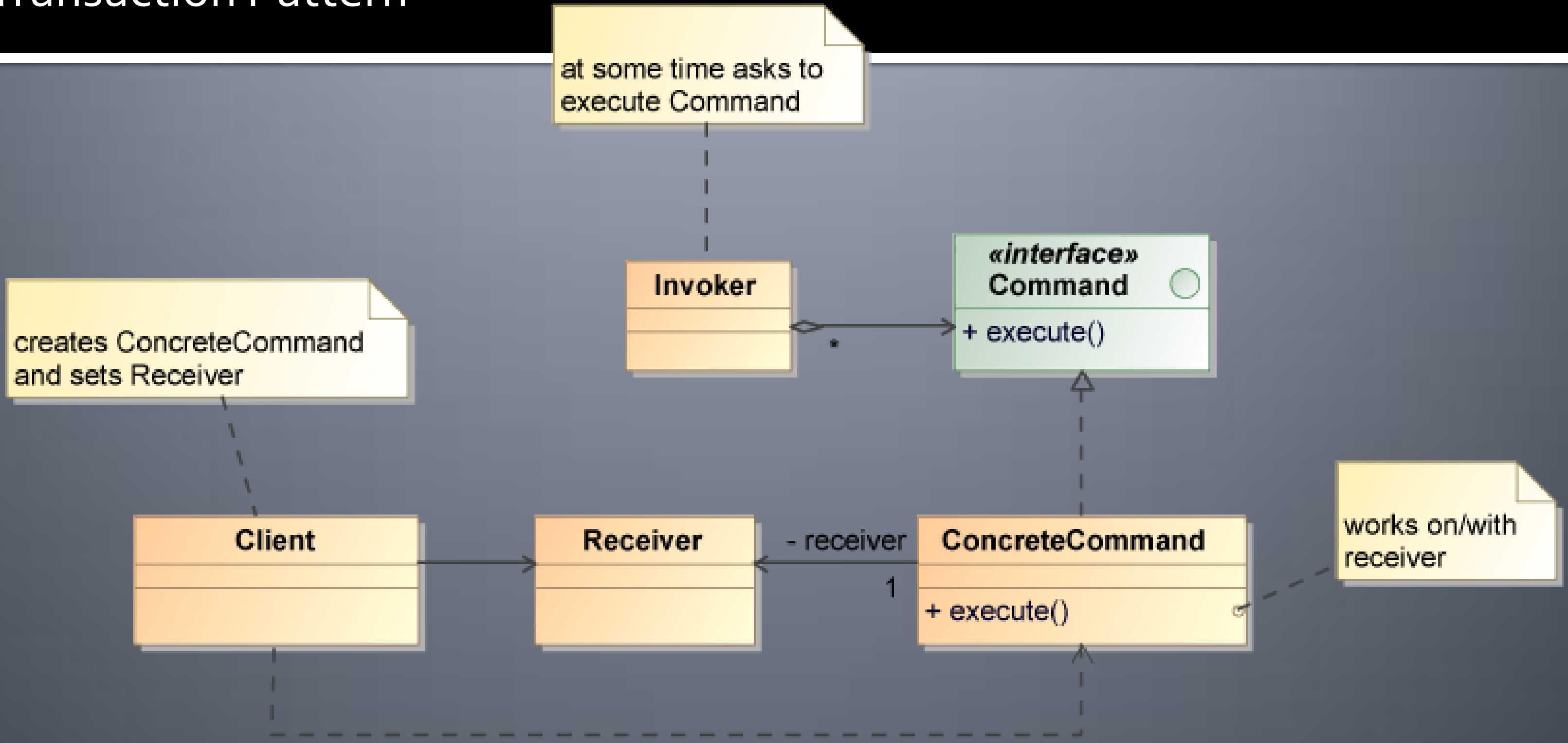
Marc Walter

Behavioral Design Patterns: Command and Memento

19.04.2013

Behavioral Design Patterns: Command Pattern

a.k.a. Action Pattern,
Transaction Pattern



Command Pattern – Intent

“Encapsulate a request as an object, thereby letting you parameterize clients with different requests […] ”

GoF – Design Patterns

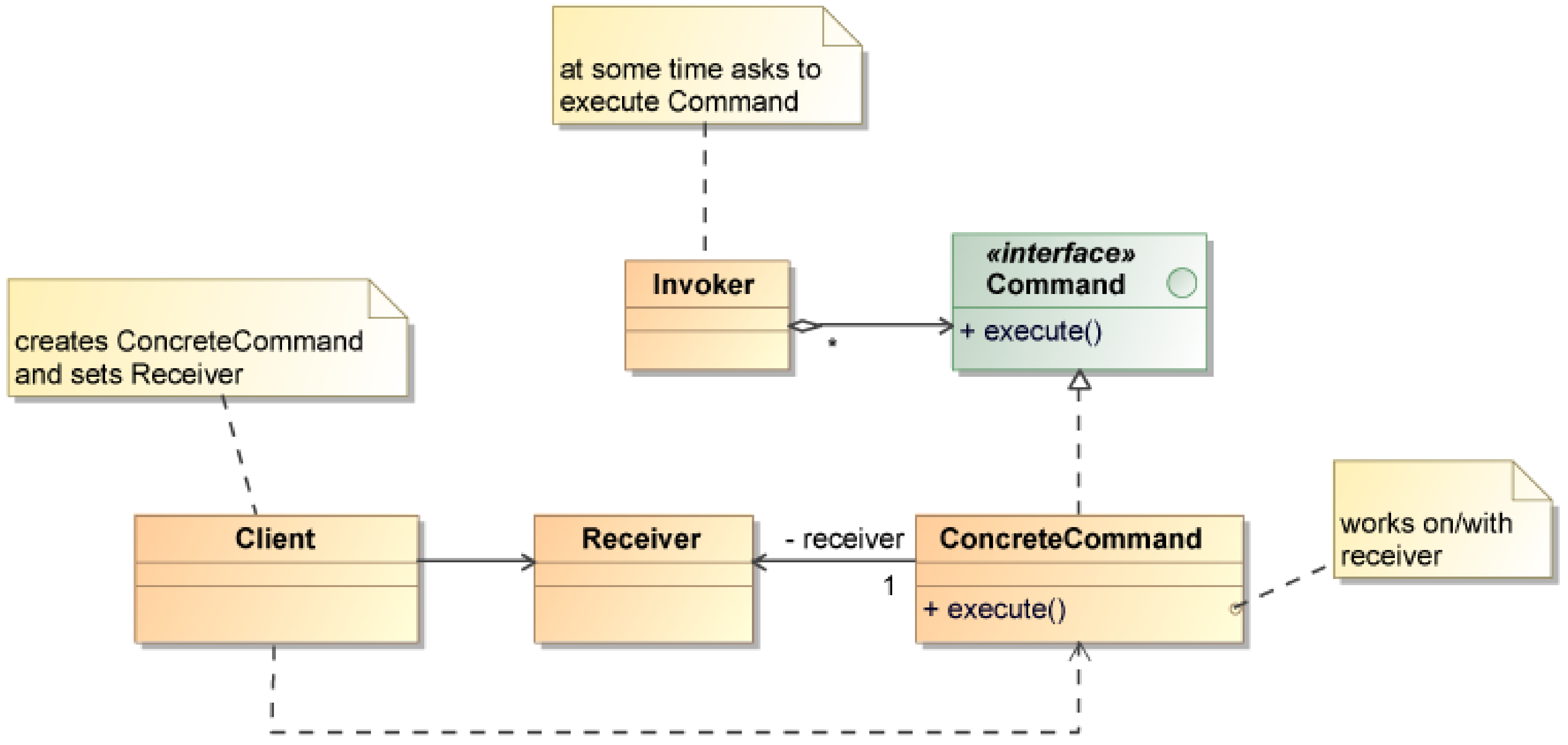
What is it about?

problem: most object oriented languages don‘t allow first-class functions

workaround needed for passing functions/methods to another object/function

solution: encapsulate method into an object, creating a function object that keeps state and a specific function

UML Class Diagram



How does it work?

Client: creates Concrete Command and sets Receiver

Receiver: Concrete Command works with/on Receiver

Invoker: invokes Command

Command: common interface for Concrete Commands

Concrete Command: executes code

Example: Command Pattern

Wolfram|Lambda

asynchronous equation solver

Used where/what for?

- asynchronous code execution
  - GUI Buttons and Menu Items (Swing, Delphi → Action object which contains image, text, code)
  - abstract Thread Pools → execute different kinds of commands
- wizard → instantiate command object, change its state and execute commands
- macro recorder
- undo machine (history of commands is kept for un- and re-doing)

Advantages and Drawbacks

- easily extendable
- combine atom commands to create macro commands
- passing methods is possible in OOP
- asynchronous execution of code
- callback functions

- 1 command = 1 class → high number of classes

Related Patterns

Visitor

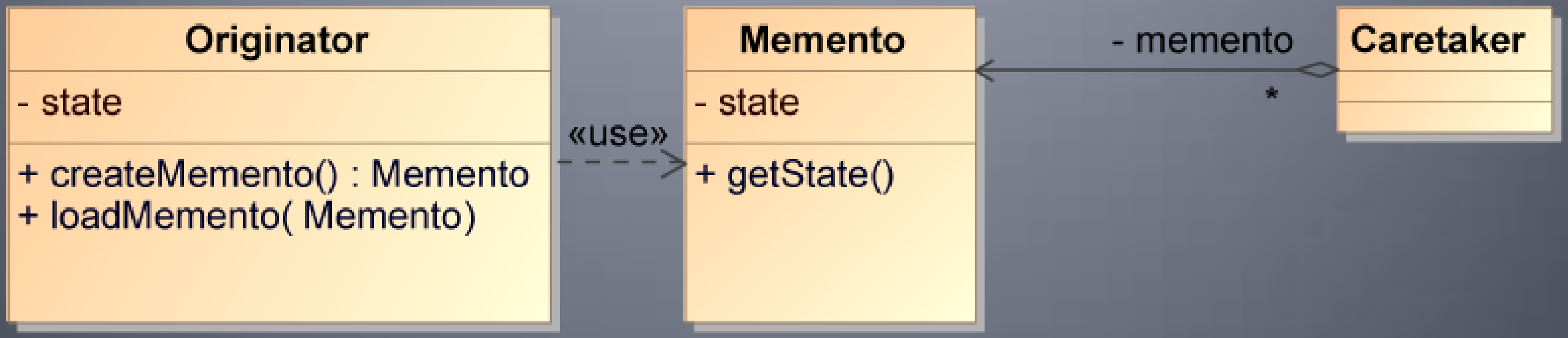
Observer

Composite

Memento

Behavioral Design Patterns: Memento Pattern

a.k.a Token Pattern



Memento Pattern – Intent

“Without violating encapsulation, capture and externalize an object's internal state so that the object can be restored to this state later.”

GoF – Design Patterns

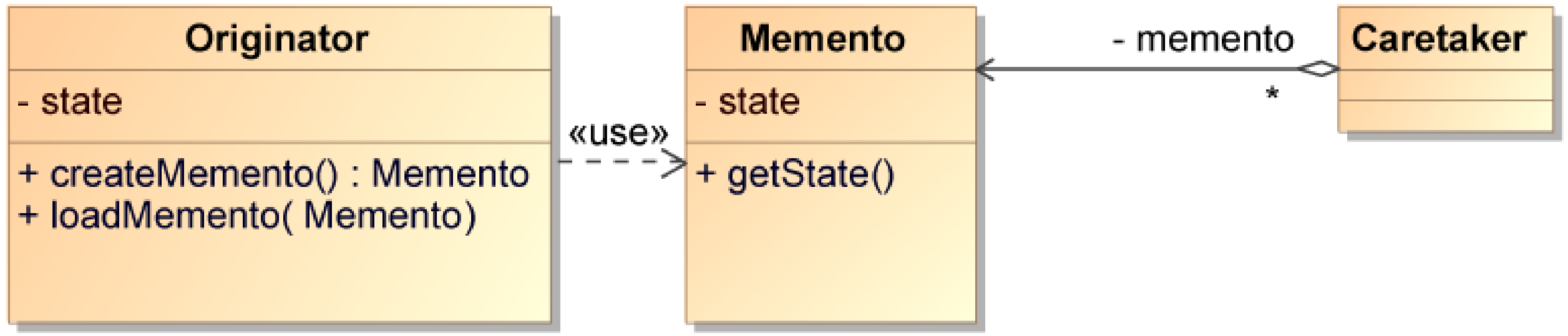
What is it about?

save and restore an object‘s state

keep the object simple

create another object to hold state

UML Class Diagram



How does it work?

Memento: saved state in an object

Originator: saves snapshot of internal state in Memento object

Caretaker: handles Mementos and supplies them back to Originator

Used where/what for?

- save object state (e.g. to disk)
- undo and redo functionality
- serialize objects/ their data

Advantages and Drawbacks

+ preserves encapsulation
+ simplifies Originator (Originator does not have to keep previous states)

- might contain a lot of overhead → expensive
- hidden costs in storing Mementos, Caretaker does not know what it is storing

Related Patterns

Command

Iterator

Example: Command and Memento Pattern

Pretty Source

prettyfy source code
with undo/redo functionality

Thanks for your time

Questions?

Sources

- http://en.wikipedia.org/wiki/Command_pattern
- http://www.philipphauer.de/study/se/design-pattern/command.php
- http://www.oodesign.com/memento-pattern.html
- http://en.wikipedia.org/wiki/Memento_pattern
- Book: Gang of Four – Design Patterns [1994]

Resources

- slide 02,05: Class Diagram Command Pattern
  http://en.wikipedia.org/wiki/File:Command_Design_Pattern_Class_Diagram.png
- slide 07: Wolfram|Alpha Logo
  slidehttp://shuisman.com/?p=179
- slide 11,14: Class Diagram Memento Pattern
  http://dofactory.com/Patterns/PatternMemento.asp