

Software Engineering – Behavioural design patterns

Adam Krechowicz

1 Behavioural design patterns

Behavioural design patterns allows to solve problems with the behaviour of the objects (of how they work).

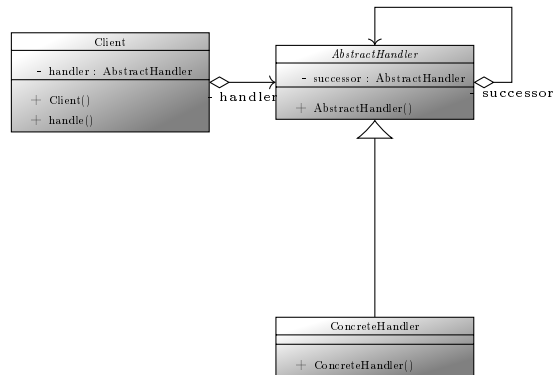
Behavioural group consists of the following patterns:

- Chain of responsibility
- Command
- Interpreter
- Iterator
- Mediator
- Memento
- Observer
- State
- Strategy
- Template method
- Visitor

1.1 Chain of responsibility

Is used when we want to:

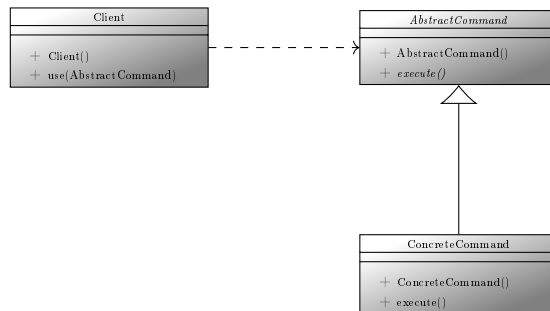
- have possibility to execute tasks in different ways
- with strictly defined order of this executions



1.2 Command

Is used when we want to:

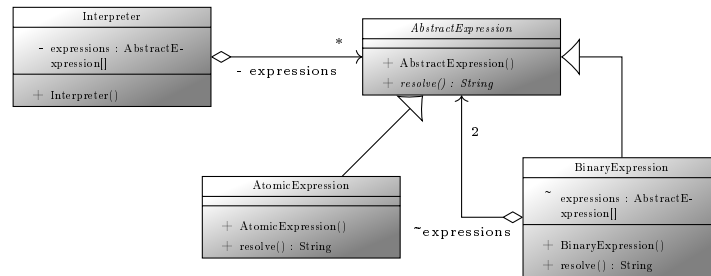
- Encapsulate the request in form of an object
- Create queue of request
- Route requests to different receivers
- Log informations about executed tasks



1.3 Interpreter

Is used when we want to:

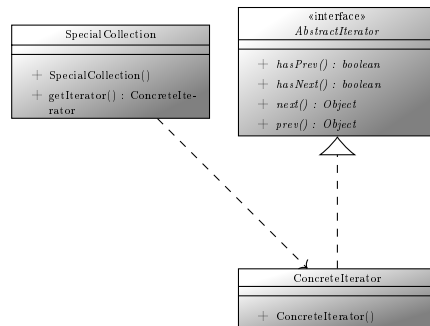
- Create the way to represent expressions
- that can be dynamically processed



1.4 Iterator

Is used when we want to:

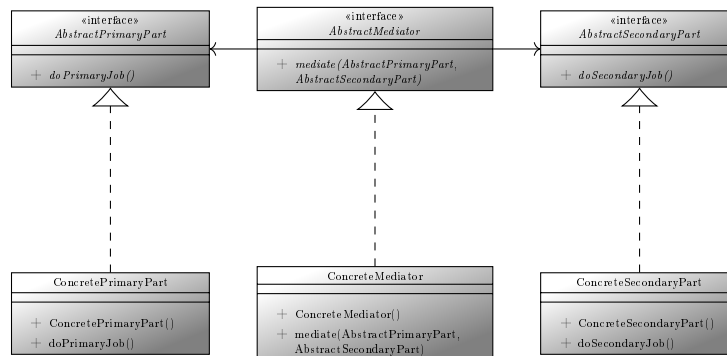
- Create uniform access to aggregated objects
- no matter of its internal structure



1.5 Mediator

Is used when we want to:

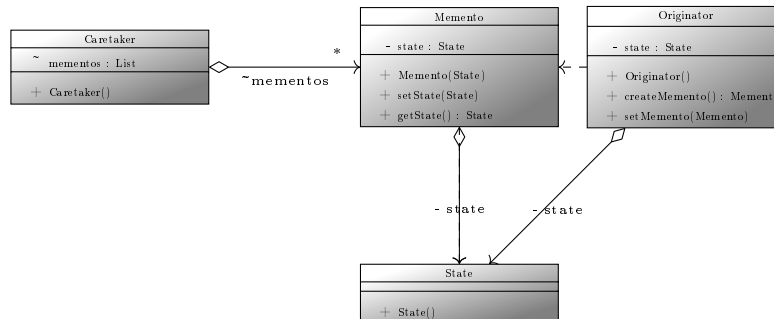
- Create universal way of communications between objects
- Allows to create loosely coupled structure
- Decrease the relations between objects



1.6 Memento

Is used when we want to:

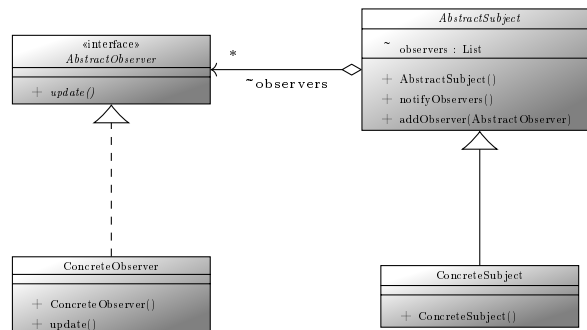
- allow to save the state of object
- to ensure that it can be reverted in the future



1.7 Observer

Is used when we want to:

- notify objects
- about some changes in other objects
- to minimize the need of pooling
- receivers can be dynamically changed

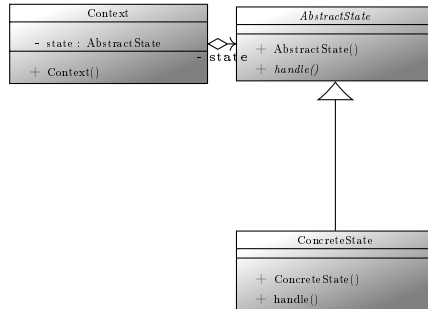


1.8 State

Is used when we want to:

- change the behaviour of object
- dynamically

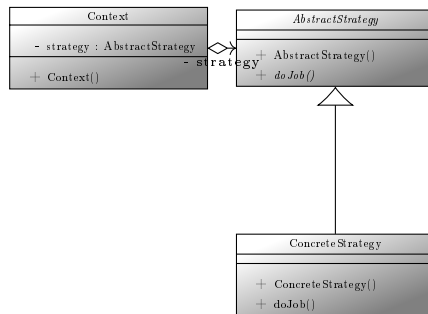
- in such a way that it seems like object changed its class



1.9 Strategy

Is used when we want to:

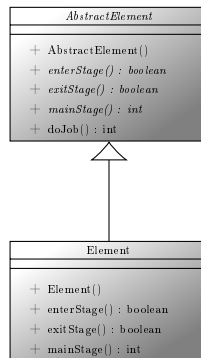
- Allows executing of one task in many different ways
- Concrete way can depend on many different things



1.10 Template method

Is used when we want to:

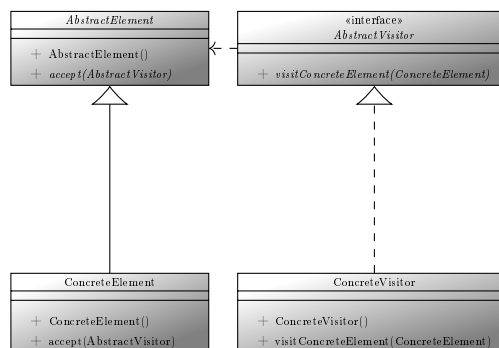
- Define the abstract structure for some algorithm
- define the steps of algorithm
- Customize algorithm without changing code structure



1.11 Visitor

Is used when we want to:

- Use operations that are executed on group of objects
- change behaviour of objects



2 Tasks to complete

1. Identify the place for behavioural design pattern in the system
2. Describe the problem that justifies the need of design pattern
3. Describe the pattern its theory and its place in the system
4. Create class diagram for pattern
5. Create source code that implements pattern
6. Create test source code for pattern

Each member of the team should pick other pattern.

Results should be placed in appropriate article each pattern for section (<section class="pattern">). The structure of the section is as follows:

- h5 – pattern name
- `<p class="author">` – author of pattern
- `<div class="pattern-problem">` – pattern problem
- `<div class="pattern-description">` – pattern description
- `<p class="uml pattern-diagram">` – pattern class diagram
- `<pre class="pattern-code"><code class="lang-java">` – source code
- `<pre class="pattern-test-code"><code class="lang-java">` – test source code