Strategy Pattern

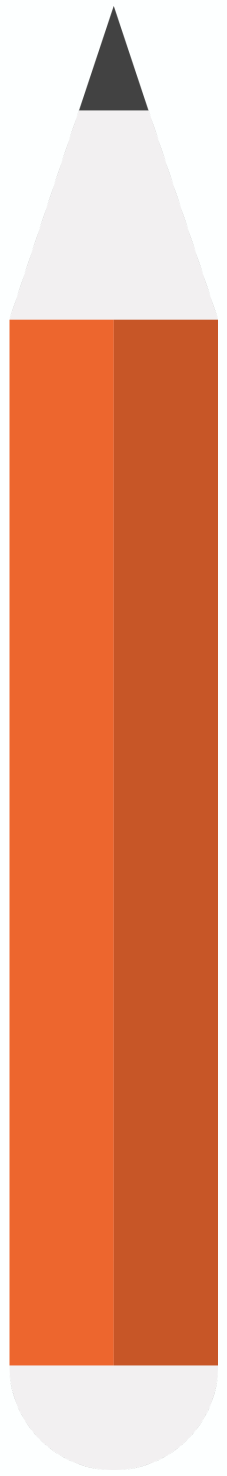
Bryan Hansen

twitter: bh5k | http://www.linkedin.com/in/hansenbryan

Concepts

- Eliminate conditional statements
- Behavior encapsulated in classes
- Difficult to add new strategies
- Client aware of strategies
- Client chooses strategy
- Examples:
  - java.util.Comparator

Design



Abstract base class

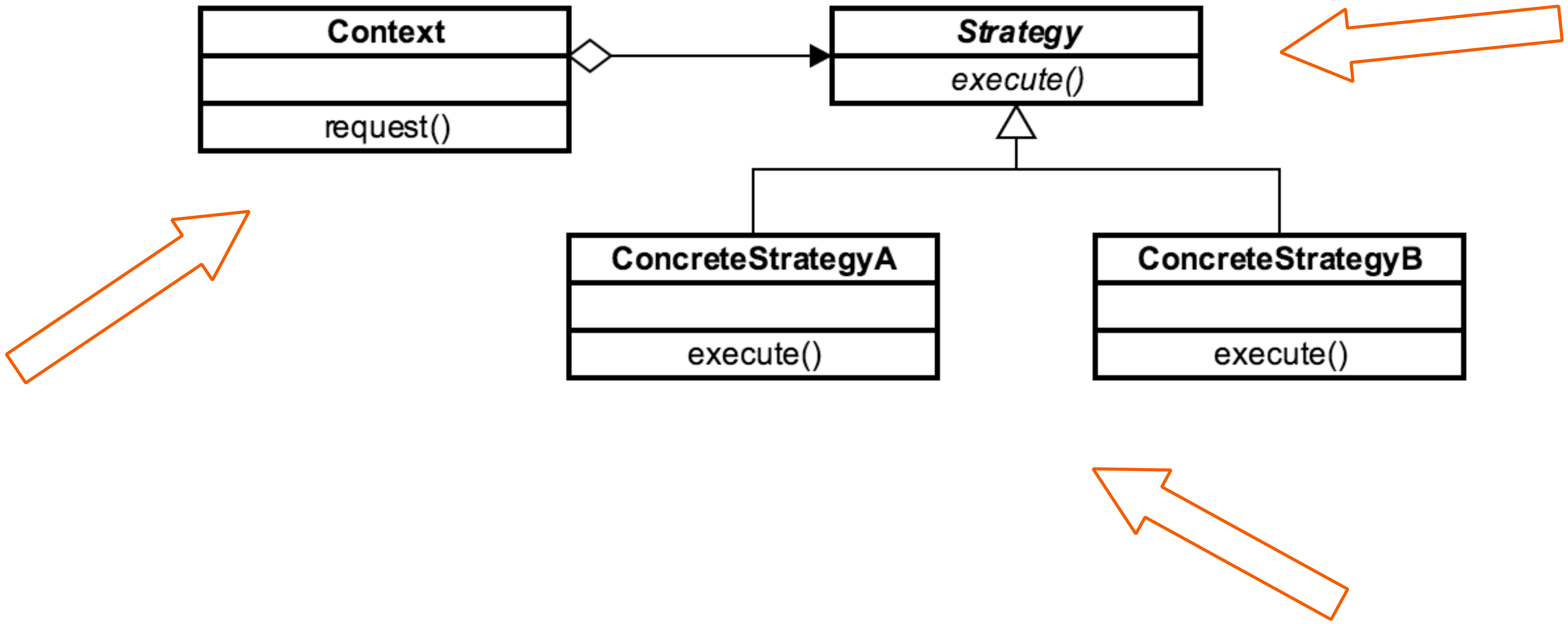
Concrete class per strategy

Removes if/else conditionals

Strategies are independent

Context, Strategy, ConcreteStrategy

UML


Context
request()
Strategy
execute()
ConcreteStrategyA
execute()
ConcreteStrategyB
execute()

Everyday Example - Comparator

```
Collections.sort(people, new Comparator<Person>() {
  @Override
  public int compare(Person o1, Person o2) {
    if(o1.getAge() > o2.getAge()) {
      return 1;
    }

    if(o1.getAge() < o2.getAge()) {
      return -1;
    }

    return 0;
  }
});
```

Exercise Strategy

Context, Strategy, ConcreteStrategy

Switch Strategy

Pitfalls

- Client aware of Strategies
- Increased number of classes

Contrast

Strategy

- Interface based
- Algorithms are Independent
- Class per Algorithm

State

- Interface based
- Transitions
- Class per State

Strategy Summary

- Externalizes algorithms
- Client knows different Strategies
- Class per Strategy
- Reduces conditional statements