Software Engineering: Tutorial 8

David Voigt

December 16th, 2022

Common Mistakes: 1

```
1  def getAnswer(bigPack: Pack, smallPack: Pack): ProblemAnswer = {
2      var result = ProblemAnswer.Small
3      if (bigPack.pricePerLiter() <= smallPack.pricePerLiter()) {
4          result = ProblemAnswer.Large
5      }
6      return result
7  }
```

Common Mistakes: 1

```
1  def getAnswer(bigPack: Pack, smallPack: Pack): ProblemAnswer =
2      if (bigPack.pricePerLiter() <= smallPack.pricePerLiter())
3          ProblemAnswer.Large
4      else
5          ProblemAnswer.Small
```

Common Mistakes: 2

```
try {
    throw Exception("The input has the wrong format")
} catch {
    case e: Exception => println("The input format is invalid")
}
```

Common Mistakes: 2

```
try {
    throw Exception("The input has the wrong format")
} catch {
    case e: Exception => println("The input format is invalid")
}
```

- Either the current state is really **exceptional** and we terminate the current computation or the computation may continue and we just print a warning
- Here the computation is not terminated and the exception is just used for immediately printing something to the screen
- Use exceptions if something **exceptional** occurs to prematurely exit a computation with a descriptive error message

Common Mistakes: 3

```
if c then true
else false
```

Common Mistakes: 3

```
if c then true
else false
```

but this is just

```
c
```

Revision: Interfaces

What is an interface?

Revision: Interfaces

What is an interface?

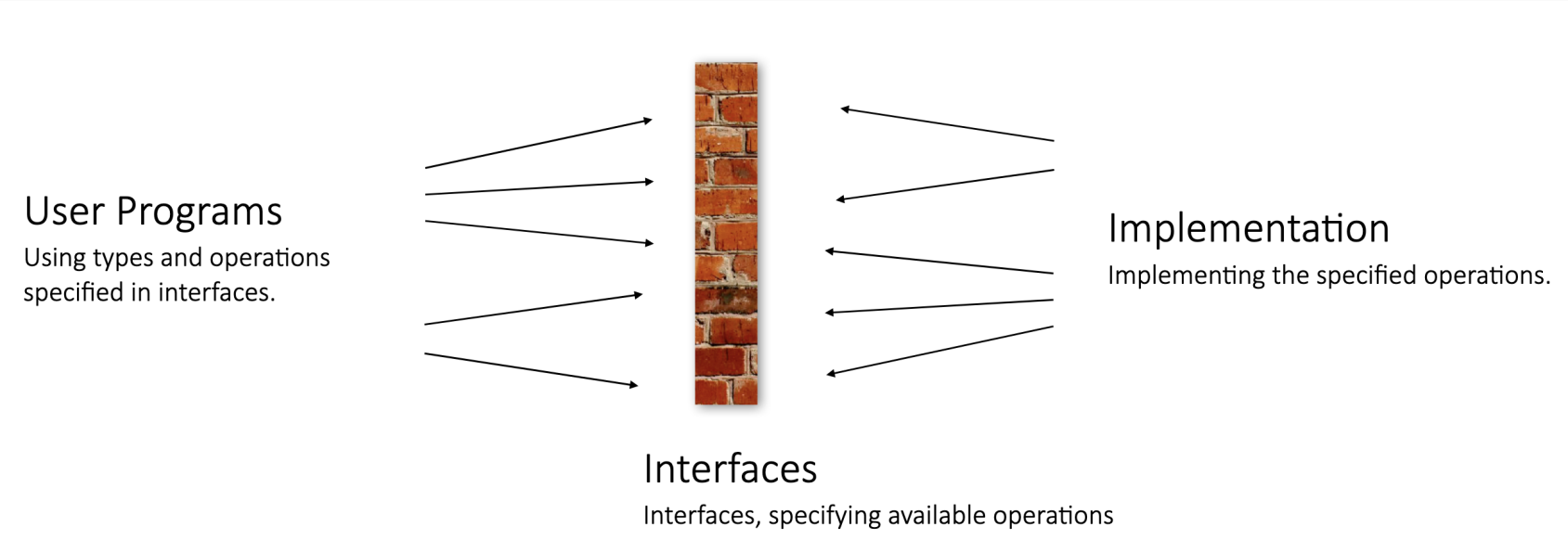

Figure 1: Brachthäuser, J. (2022). "Interface visualization". Slides Software Engineering (WS22)

- **API**: **A**pplication **P**rogramming **I**nterface describes how two services communicate
- Hides the actual implementation from the user

Design by Contract

Precondition

- Guarantees for the **input** as requirement for the user
- To be keept as weak as possible for maximum reuse

Postcondition

- Guarantees for the **output** as requirement for the implementor
- To be keept as strong as possible
- Tradeoff reusability with maintainability

Invariants

- Aspects left unchanged by usage

Design by Contract: Example

```
def head[T](xs: List[T]): T
```

Preconditions, Postconditions, Invariants?

Design by Contract: Example

```
def head[T](xs: List[T]): T
```

Preconditions, Postconditions, Invariants?

Preconditions

- The list is non-empty

Postconditions

- The return value is the first element in the list

Invariants

- The given list is unchanged