EPFL

Domain Modeling

Effective Programming in Scala

Levels of Abstraction

Consider the last version of the program that computes the area of a house that needs to be painted:

```
val facade = 5 * 3
val door   = 1 * 2
val window = 1 * 1
facade - door - window - window
```

Each part of the house (facade, door and windows) is defined as a surface, which is itself defined in terms of two numbers, a width and a height.

I argue that we—as humans—have to make some cognitive effort to map these numbers to the abstract concept they relate to—a shape.

We can say that numbers are a **low-level** concept, whereas shapes are a **high-level** concept.

Shapes and Areas

Alternatively, here is how we could express our program in terms of *shapes* and *areas*:

```
val facade = Rectangle(width = 5, height = 3)
val door   = Rectangle(width = 1, height = 2)
val window = Rectangle(width = 1, height = 1)
facade.area - door.area - window.area - window.area
```

I argue that this version of the program is more readable and less error-prone. We will see in the following lessons how to define a concept of “Rectangle”.

Finding the *Right* Level of Abstraction

One could object that doors or windows are more complex than rectangles (e.g., a door is made of some material, it has a handle, etc.).

However, **for the purpose of our program** a rectangle seems to carry enough information to model a door.

Finding the *Right* Level of Abstraction

One could object that doors or windows are more complex than rectangles (e.g., a door is made of some material, it has a handle, etc.).

However, **for the purpose of our program** a rectangle seems to carry enough information to model a door.

> *The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise*

Edsger W. Dijkstra

Summary

The act of defining, in a program, the concepts of a domain that map closely our human reasoning is called **modeling**.

In the following lectures we will see how to model abstract concepts in Scala.