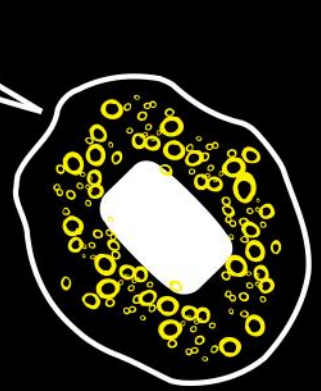
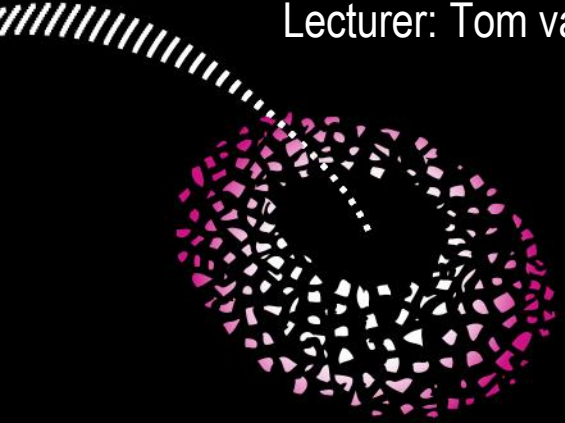
UNIVERSITY OF TWENTE.

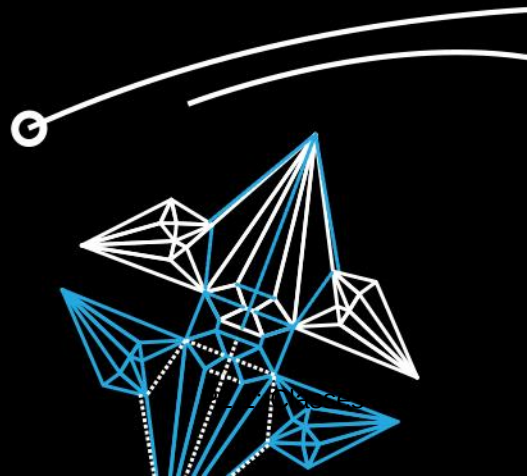


Introduction to Object-Oriented Programming

Topic of Software Systems (TCS module 2)

Lecturer: Tom van Dijk





INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING

- Object-oriented programming
- Encapsulation and abstraction
- Separation of concerns

ESSENCE OF PROGRAMMING

COMPUTERS

A program is a list of instructions operating on binary data (0s and 1s)

HUMANS

People understand concepts (ideas) and abstractions

ESSENCE OF PROGRAMMING

We need programming languages to:

- Add structure to the program
- Separate different concerns
- Reason about concepts instead of binary data
- Reason that software functions correctly

QUICK HISTORY OF PROGRAMMING

- In the beginning:
  Direct input of instructions (assembly) operating on binary data
- 1950s, 1960s: Structured and procedural programming
  Languages like FORTRAN, COBOL, ALGOL, C
  Features: variables, procedures, code flow (if, for, while), scope
- 1970s, 1980s: Object-oriented programming
  Languages like Smalltalk, C++, Java
  Features: objects, inheritance, polymorphism, encapsulation

OBJECT-ORIENTED PROGRAMMING

- A program is a collection of objects
- Objects have private internal state (variables, called fields)
- Objects interact via public methods

- This is called encapsulation: hide information in objects and only expose public methods to access/manipulate the data
- Like black boxes

EXAMPLE: LIBRARY

A library has:

- Bookcases that contain Books
- Books that have a fields such as title, author, etc.
- Customers that borrow Books

Each book is an object. Each bookcase is an object.

The customers are objects. The library is an object.

EXAMPLE: PETS

A Dog object

- Private fields: hungry, thirsty, energy, mood
- Public methods: feed, play, command
- Private methods: bark, run, sleep

ABSTRACTION

Abstraction means: hiding unnecessary details

Distinction between:

- high-level interface (the public method signatures)
- low-level implementation (method body, state, private methods)

ABSTRACTION VS ENCAPSULATION

Abstraction is at the design level

- Hide inner implementation details, only provide what is necessary

Encapsulation is at the implementation level

- Hide information, by keeping state private, and providing public methods

SEPARATION OF CONCERNS

Different parts or responsibilities go to different classes

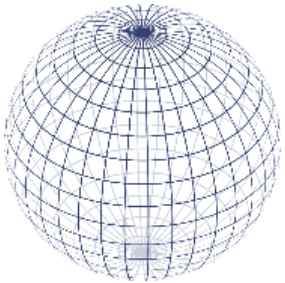
Example: a simulation game with houses, cars, factories, trucks, trains, etc.

- When working on a truck, no need to think about houses and factories

OBJECT-ORIENTED PROGRAMMING

Class

Generic description of attributes and behavior



Object

Instance of a class with specific attributes and behavior

OBJECT-ORIENTED PROGRAMMING

Every object is an instance of a class.

The class defines the fields and methods of each object.

All books in the library are instances of one class Book.

The bookcases are all instances of one class Bookcase.

FIELDS

Fields have a type

- primitive type (int, float, boolean, char)
- reference type (to an object)

```
class Box
{
    String label;
    int length;
    int height;
    int width;
}
```

METHODS

Simple getters and setters

- Manipulate the state of the object (values in the fields)
- `getName()` and `setName(String name)`

Complex operations

- `borrowBook(Customer theCustomer, Book theBook)`
- `getNumberOfBooks(Customer theCustomer)`

PACKAGES

In Java, every class is in a package

- Convenient for the programmer (more structure!)
- Example: `java.util.List`
- Example: `nl.utwente.tcs.myfirstprogram.HelloWorld`

TERMINOLOGY

- Classes are sometimes called composite types
- Fields are also called attributes or properties
- Class methods that return a value are also called functions or queries
- Void methods are sometimes called commands

CONCLUSION

- Procedural and object-oriented programming
- Encapsulation and abstraction
- Separation of concerns

- Every object is an instance of a class
- Objects have fields and methods
- Objects are in a package