OOSD Concepts

Dr. Jason Barron

South East Technological University

September 17, 2024

Contents

- OO Approach
- Object Definition, Attributes, Behaviour
- Encapsulation
- Class Definition, Attributes, Operations
- Class Diagram

Object-based Programming

Writing Classes and Methods, Creating Objects

- Up to now we have only used predefined classes, objects and methods...
- We have defined our own classes, but merely as placeholders for method **main()**
- Such classes are referred to as the **main class**
- Now we look at writing our own "custom" class which will **NOT** have a main method

OO Approach

- The object-oriented approach to systems development is based on the concept of **objects** that exist within a system's environment
- Objects are everywhere
- Example: look around
  - Door
  - Window
  - Room itself

OO Approach

- Object-oriented systems focus on capturing the **structure** and **behaviour** of information systems in little modules that encompass both data and process

What is an Object?

”Something that is or is capable of being seen, touched or otherwise sensed and about which users store data and associate behaviour”

Attributes

- Each object has **attributes** (**data**) that describe information about the object e.g. patient's name, date of birth, address and phone number
- Each object has **state** defined by the value of its attributes and its relationships with other objects at a particular point in time e.g. the patient may be "new", "current" or "former"

Behaviour

- The **behaviour** of an object refers to those things that the object can do and that correspond to functions that act on the object's data (or attributes) e.g. an appointment object can:
  - schedule an new appointment
  - delete an appointment
  - locate the next available appointment

Examples

- Real-world objects share two characteristics: They all have **Data attributes** and **Behaviour**
- Dogs have
  - Data attributes (name, colour, breed, hungry) and
  - Behaviour (barking, fetching, wagging tail)
- Bicycles also have
  - Data attributes (current gear, current pedal cadence, current speed) and
  - Behaviour (changing gear, turning, applying brakes)
- **METHODS** IMPLEMENT AN OBJECT'S BEHAVIOUR

Object Types

- The types of object may include:
  - A person
    - employee, customer, instructor, student
  - A place
    - Warehouse, office, building, room
  - A thing
    - Product, vehicle, computer, videotape,
  - An event
    - Order, payment, invoice, application, registration, reservation

Example: A "Rabbit" object

- You could (in a game, for example) create an object representing a rabbit
  It would have data (attributes):
  - How hungry it is
  - How frightened it is
  - Where it is
- And methods (behaviours):
  - eat, hide, run, dig

Object Types

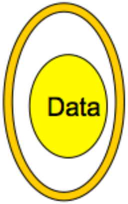
- How many objects can you identify from the following?
    - A library
    - An online store
    - A college
- For each object identified:
    - Identify state
    - Identify behaviour

Object-Oriented Approach

- The OO approach views information systems not as data and processes but as a collection of objects that **encapsulate** data and processes
- Objects can contain data attributes and methods
- Each object encapsulates the attributes and behaviour together as a single unit

Encapsulation

- The **attributes** and **behaviours** of an OBJECT are packaged together
- They are considered part of the object
- The **only way** to access or change an object's attributes is through that object's specific **behaviours (methods)**
- We call this the object's **interface**

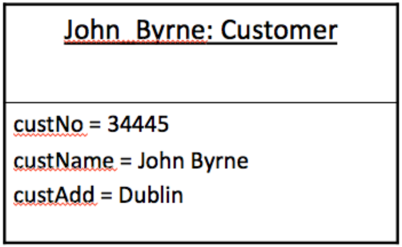


Object Data Attribute Examples

- Object called "Customer"
- Data Attributes
  - Customer number
  - Customer name
  - Customer address
- Can you think of any others?
- Each individual customer is referred to as an **object instance**

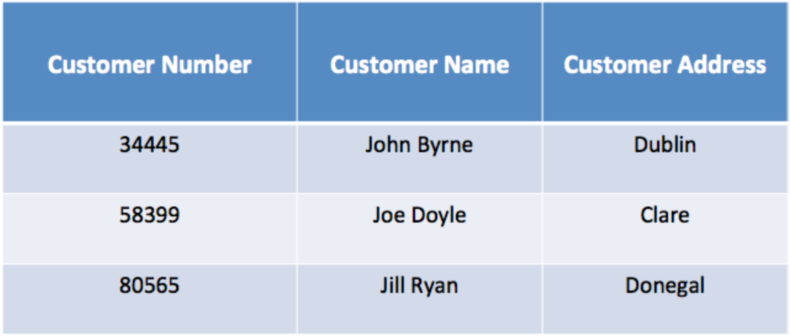
Object Diagram Example

one instance of Customer



Example

3 instances of the Customer Object



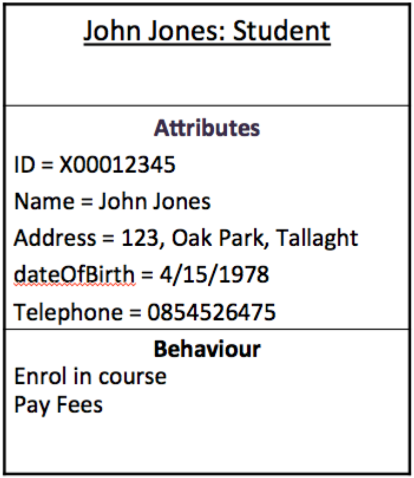
Customer Number	Customer Name	Customer Address
34445	John Byrne	Dublin
58399	Joe Doyle	Clare
80565	Jill Ryan	Donegal

Object Data Attribute Examples

- Object called "Student"
- Data Attributes
  - Student ID
  - Student Name
  - Student Address
  - Date of Birth
  - Student Telephone
- Behaviour
  - Enrol in course
  - Pay Fees

Object Diagram Example

one instance of student object



<u>John Jones: Student</u>

Attributes

ID = X00012345

Name = John Jones

Address = 123, Oak Park, Tallaght

dateOfBirth = 4/15/1978

Telephone = 0854526475

Behaviour

Enrol in course

Pay Fees

Object Behaviour

- The OO approach requires an adjustment to how we commonly perceive objects
- Some objects may be deemed motionless (no thought and no action)
- **Example: A Door**

Object Behaviour

- Example: A Door
  - In object orientation, that door can be associated with behaviour that it is assumed can be performed
- Behaviour of a door
  - The door can open, shut, lock, unlock
  - All of these behaviours are associated with the door and are accomplished by the door and no other object

Object Behaviour

- Can you identify behaviours for the following Objects?:
    - Customer
    - Employee
    - Book
    - Game
    - Order
    - Animal

Object Concepts

- Objects can be categorised into **classes**
- A **class** is a set of objects that share common attributes and behaviours
- e.g.
    - Students
    - Customers
    - Shops
    - Vehicles

Concept: Classes describe objects

- Every object belongs to (is an **instance of)** a **class**
- An object may have **fields**, or **variables**
  - The class describes those fields
- An object may have **methods**
  - The class describes those methods
- A class is like a template, or mould
  - You use the class's **constructor** to make objects

Concept: Classes are like Abstract Data Types

- An **Abstract Data Type** (ADT) bundles together:
    - some data, representing an object or "thing"
    - the operations on that data
- The operations defined by the ADT are the **only** operations permitted on its data
- Example: a **CheckingAccount**, with operations deposit, withdraw, getBalance, etc.
- Classes enforce this bundling together
    - If all data values are **private**, a class can also enforce the rule that its defined operations are the only ones permitted on the data

Example Thermometer class

```
// Implements a Thermometer class.
// Stores the current temperature in Celcius

public class Thermometer{ // begin Thermometer
   private double celsius; // celsius is accessible to all methods in this class (more
        about private later).

   public Thermometer(){ // constructor method
      setCelsius(0);
   }

   public void setCelsius(double cel){ // method to set the temperature
      celsius = cel;
   }

   public double getCelsius(){   // method to get the temperature
      return celsius;
   }
} // end class Thermometer
```

Example Thermometer class

- Now that we have our Thermometer class defined, we will need another class file that contains a **main** method
- This main method will create one or more **instances** of our Thermometer class
- That main class will be called the **driver program**

Example Thermometer Driver Program

```
// Student Name  : Oisin Cawley
// Student Id Number :
// Date          : Nov-2015
// Purpose       : My first class implementation

public class ThermTest{ // begin class ThermTest

public static void main(String args []) { // being main method

   Thermometer thermA = new Thermometer();// Create an instance of Thermometer class

   System.out. println ("Temp. of Thermometer A is " + thermA.getCelsius() );
   thermA. setCelsius (20.0) ;
   System.out. println ("Temp. of Thermometer A is " + thermA.getCelsius() );

    } // end main
} // end class ThermTest
```