FARABI UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Object-Oriented Analysis and Design

Lecture 2
Object-Oriented Thinking



Dr. Bazargul Matkerim, Professor-Researcher

Learning Outcomes

Recognize	Describe	Model
Identify objects and their responsibilities.	Distinguish identity, state and behaviour.	Connect classes and show an interaction.

By the end: explain a small classroom system using objects.

From a Room to a Software Model



- Goal: support classroom equipment control.
- Candidate objects: Room, Projector, Session.
- Include only concepts relevant to that goal.

A software object represents a concept needed by the system.

Objects Have Identity, State and Behaviour

Identity	State	Behaviour
Which object? projectorId = P101	What is true now? power = off input = HDMI	What can it do? turnOn() selectInput(source)

One object combines a distinct identity with state and behaviour.

Identity Distinguishes Objects



chairA : Chair	chairB : Chair
chairId = C01 colour = blue occupied = false	chairId = C02 colour = blue occupied = false

Equal attribute values do not necessarily mean the same object.

State Describes the Current Situation



projector101

power = off
input = HDMI
brightness = 70

Attributes record state. The model chooses which attributes matter.

Behaviour Changes or Reveals State



Before	After
power = off	power = on

turnOn() requests a state change.
getPowerState() reads the state.

Behaviour includes commands that change state and queries that return data.

A Class Describes Similar Objects

Projector

Attributes
projectorId, power, input

Operations
turnOn(), turnOff()



projector101 : Projector

A specific instance
ID: P101
State: off

A class defines shared structure and behaviour. An object is an instance.

One Class, Several Objects

Projector class

Shared definition: identity, power, input and operations

projector101

Room A
power = on

projector102

Room B
power = off

projector103

Room C
power = on

Instances share a class definition but keep their own state.

Abstraction Selects Relevant Details



Seat-booking model

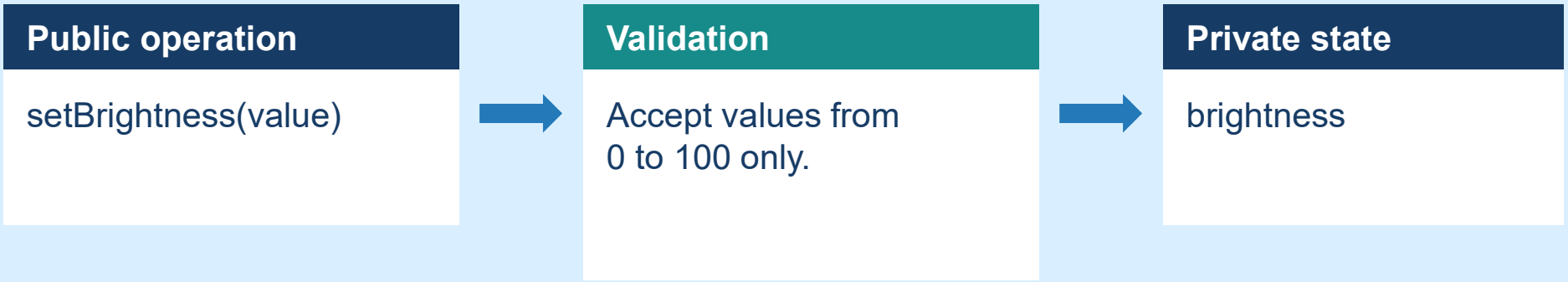
Keep: seat ID, location, occupancy status

Details outside this scope

Fabric weave and paint brand are unnecessary here.

Abstraction preserves useful detail and leaves out irrelevant detail.

Encapsulation Protects Valid State



Example: setBrightness(150) is rejected; the previous value remains.

Encapsulation controls access so operations can preserve valid state.

Responsibilities Depend on the Problem



Teaching system

Instructor leads a session
and explains a topic.

Access-control system

User has credentials
and permission to enter.

The same real person may have different software roles in different systems.

Objects Collaborate Through Requests

ClassSession

Coordinates the start
of a teaching session.



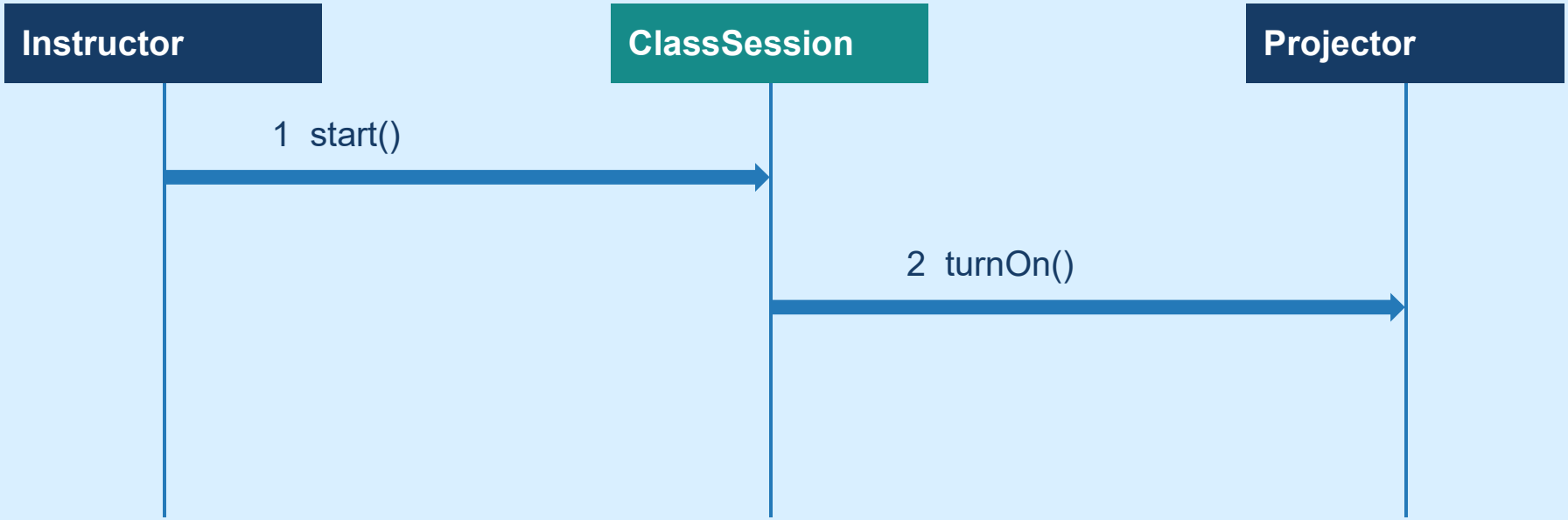
Projector

Handles turnOn() and
controls its own state.

Request: turnOn()

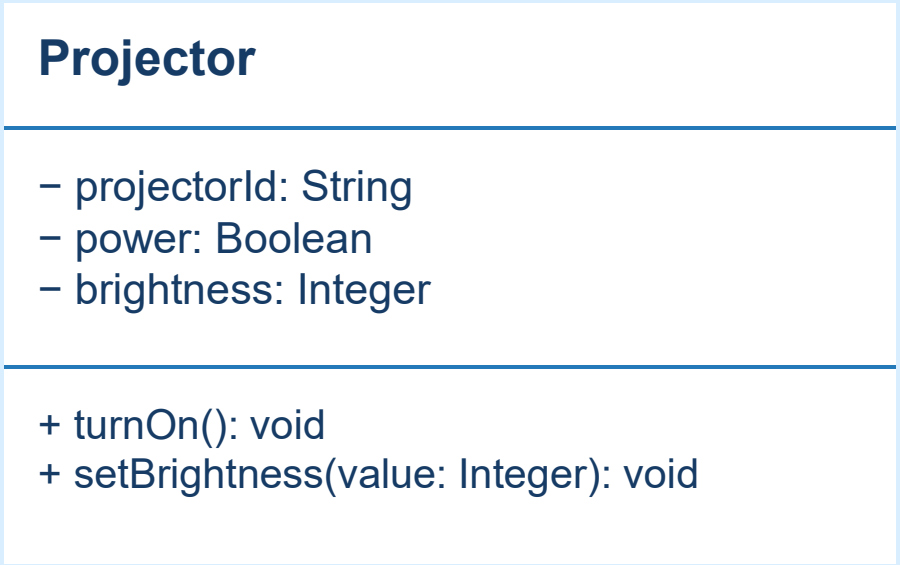
The requester expresses an intention; the receiving object performs its work.

A Classroom Interaction



Session becomes active after equipment startup succeeds.

UML Class Notation



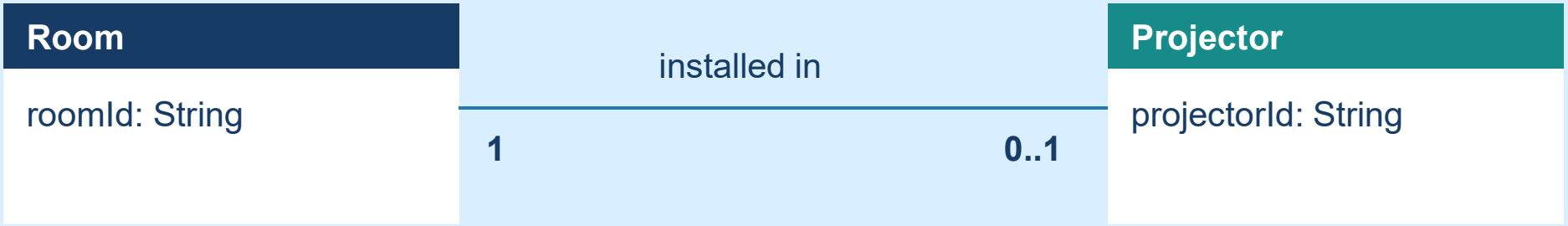
Top: class name

Middle: attributes and their types

Bottom: operations and parameters

Visibility symbols: + public − private. This is a simplified UML class.

Relationships Connect Objects



Assumption: each projector belongs to one room;
a room has zero or one installed projector.

Multiplicity records a domain rule. Confirm it with stakeholders.

From Requirements to Objects

“An instructor starts a session and turns on the room projector.”

Candidate concepts	Responsibilities	Check the model
Instructor ClassSession Projector	Request session start Coordinate the activity Control projection	Who owns each action? What state changes? What can fail?

Nouns suggest candidates; scenarios reveal whether the objects are useful.

Common Modelling Mistakes

Modelling mistake	Better approach
Turning every noun into a class	Keep concepts relevant to the goal
Exposing all attributes	Use operations to protect state
One object doing everything	Distribute coherent responsibilities

Judge a model by how clearly it explains the required behaviour.

Classroom Modelling Exercise

Scenario: a student reserves an available seat for a classroom session.

Model	Discuss
Choose the relevant classes. Give each one attributes and operations.	Show a reservation request. Explain what happens when the seat is already reserved.

Deliverable: one small class diagram and a numbered interaction.

Review and Key Ideas

An object has identity, state and behaviour.
A class defines a family of similar objects.
Abstraction chooses detail; encapsulation protects state.
Objects collaborate to achieve a system goal.

Exit question: two projectors have identical settings.
Are they the same object? Explain your answer.