PT821: Object-Oriented Programming

Introduction to OOP with Java

Masoud Hamad

State University of Zanzibar
BITA Second Year - Semester 1

First Lecture

Outline

Welcome to Object-Oriented Programming!

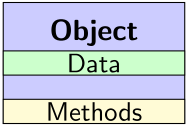
- Course Code: **PT821**
- Credit Points: **10**
- Contact Hours: **3 hours per week**
- Program: **BITA Second Year**
- Semester: **1**

Course Aim

Master the principles and practice of Application Software Development using Java and Object-Oriented Programming concepts.

Why Object-Oriented Programming?

- **Real-world modeling**
- **Code reusability**
- **Maintainability**
- **Scalability**
- **Security**



Learning Outcomes

By the end of this course, you will be able to:

1. **Master** Java programming language basics
2. **Understand** OOP concepts and Java features
3. **Design** programs using:
    - Data structures
    - Control structures
    - Inheritance
    - Interfaces
    - Abstract classes
4. **Master** exception handling techniques
5. **Write** robust, industrial-strength code
6. **Manage** object collections and serialization

What is Programming?

Definition

Programming is the process of creating a set of instructions that tell a computer how to perform a task.

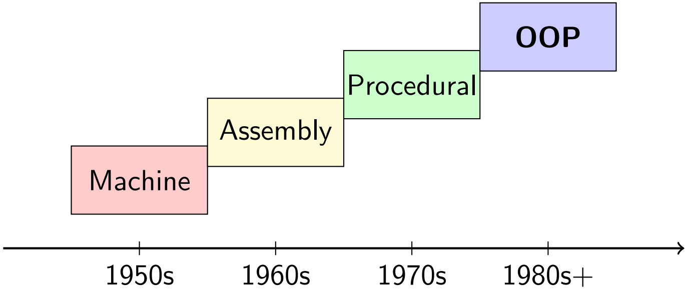
Key Components:

- Problem solving
- Algorithm design
- Code implementation
- Testing & debugging

Programming Paradigms:

- Procedural
- **Object-Oriented**
- Functional
- Logic-based

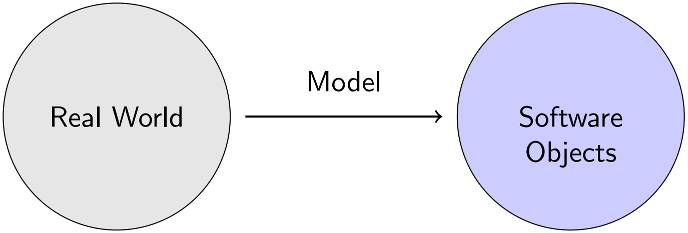
Evolution of Programming



What is Object-Oriented Programming?

Definition

OOP is a programming paradigm based on the concept of "objects" which contain data (attributes) and code (methods).



Core OOP Principles

1. Encapsulation

Bundling data and methods that operate on that data within a single unit (class)

2. Inheritance

Creating new classes based on existing classes

3. Polymorphism

Ability of objects to take multiple forms

4. Abstraction

Hiding complex implementation details and showing only essential features

Objects and Classes

Class:

- Blueprint or template
- Defines structure and behavior
- Like a cookie cutter

Object:

- Instance of a class
- Has state and behavior
- Like actual cookies

Example

Class: `Student`

Objects: `student1`, `student2`, `student3`

Why Java for OOP?

- **Pure OOP language**
- **Platform independent**
- **Simple and familiar syntax**
- **Automatic memory management**
- **Rich API and libraries**
- **Strong community support**
- **Industry standard**

JAVA

Write Once, Run Anywhere

Java History

- **1991**: Project Green started by James Gosling
- **1995**: Java 1.0 released by Sun Microsystems
- **1996**: JDK 1.0 released
- **2010**: Oracle acquired Sun Microsystems
- **Today**: Java is one of the most popular programming languages

"Write once, run anywhere" - Java's philosophy

Your First Java Program

```java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, BITA Students!");
        System.out.println("Welcome to OOP with Java!"
            );
    }
}
```

Key Components:

- `public class` - Class declaration
- `main` method - Entry point
- `System.out.println` - Output statement

Basic OOP Example

```
public class Student {
    // Attributes (Data)
    private String name;
    private int age;

    // Constructor
    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Method (Behavior)
    public void study() {
        System.out.println(name + " is studying OOP!")
            ;
    }
}
```

Course Topics Overview

1. **Java Basics**
   - Variables, data types, operators
   - Control structures
   - Arrays and strings
2. **OOP Fundamentals**
   - Classes and objects
   - Constructors and methods
   - Access modifiers
3. **Advanced OOP**
   - Inheritance and polymorphism
   - Abstract classes and interfaces
4. **Java Applications**
   - Exception handling
   - Collections and serialization
   - File I/O operations

Assessment Methods

Continuous Assessment (40%)

- Lab exercises
- Assignments
- Quizzes
- Class participation

Final Assessment (60%)

- Final project
- Written examination

Important

Regular attendance and practice are crucial for success!

Course Resources

- **Recommended Textbooks:**
  - "Java: How to Program, 12th Edition" by Deitel & Deitel (2023)
  - "Java: The Complete Reference, 13th Edition" by Herbert Schildt (2024)
  - "Effective Java, 3rd Edition" by Joshua Bloch (2018)
  - "Clean Code: A Handbook of Agile Software Craftsmanship" by Robert Martin (2019)
- **Online Resources:**
  - Oracle Java Documentation
  - Online tutorials and coding platforms
  - Course materials on LMS
- **Software Required:**
  - Java Development Kit (JDK) 21 LTS
  - IDE: Eclipse 2023-12, NetBeans 20, or BlueJ

Setting Up Your Environment

1. **Download and Install JDK**
   - Visit Oracle website or OpenJDK
   - Download JDK 21 LTS (Long Term Support)
   - Follow installation instructions
2. **Choose an IDE**
   - Eclipse 2023-12 (recommended - no AI features)
   - NetBeans 20 or earlier
   - BlueJ (ideal for learning)
3. **Verify Installation**
   - Open command prompt
   - Type: `java -version`
   - Type: `javac -version`

Tips for Success

Do's:

- Practice coding daily
- Ask questions
- Work on projects
- Collaborate with peers
- Read documentation

Don'ts:

- Don't skip classes
- Don't copy code blindly
- Don't ignore errors
- Don't delay assignments
- Don't be afraid to fail

"The only way to learn programming is by programming!"

AI Tools: Guidelines and Precautions

DO's:

- Use AI for concept explanations
- Ask AI to explain error messages
- Use AI for code review feedback
- Learn from AI-suggested improvements
- Verify AI responses with documentation

DON'Ts:

- Don't copy AI code without understanding
- Don't submit AI-generated assignments
- Don't rely solely on AI for learning
- Don't skip manual debugging practice
- Don't use AI during exams

Important Academic Policy

All submitted work must be your own. AI assistance must be declared and used appropriately for learning purposes only.

Class Exercise

Think-Pair-Share Activity:

1. **Think** (2 min): List 3 real-world objects around you
2. **Identify**: What are their attributes and behaviors?
3. **Share**: Discuss with your neighbor
4. **Present**: Share one example with the class

Example

Object: **Mobile Phone**
Attributes: brand, model, color, battery level
Behaviors: makeCall(), sendMessage(), takePicture()

Summary

- OOP is a powerful programming paradigm
- Java is an excellent language for learning OOP
- Four pillars: Encapsulation, Inheritance, Polymorphism, Abstraction
- This course will prepare you for real-world software development
- Success requires consistent practice and engagement

Next Class

We'll dive deeper into Java basics and write our first programs!

Questions?

Any Questions?

Contact Information:
Instructor: Masoud Hamad
Email: massoud.hamad@suza.ac.tz
Office Hours: Thursday 08:00AM-12:00PM

"The journey of a thousand programs begins with a single line of code."

Thank You!

See you in the next class

Don't forget to install Java and an IDE before next class!