Introduction to Object-Oriented Programming with Java Examples

Duration: 3 Days Course Code: WD150G

Overview:

In this 3-day instructor-led course, students learn how to adopt an object-oriented (OO) approach to software development. The course is designed for experienced developers coming to Java from non object-oriented languages such as COBOL, RPG, or C. It is also appropriate for those who are new to programming.
Through a combination of instructor-led lectures and hands-on exercises, students take a case study through the stages of object-oriented requirements gathering, analysis, and design using the Unified Modeling Language (UML). Students learn how the Java language supports object-oriented programming, and how object-oriented designs can be implemented in Java. Numerous hands-on exercises and demonstrations provide practical experience with OO development from analysis and design to implementation.
This course includes topics such as interpreting UML diagrams, recognizing Java constructs that enable object-orientation, and how design patterns can improve the implementation of applications. The course also provides an overview of different software development methodologies that can be applied to the development of object-oriented applications.
This course prepares students for further training in the Java programming language by providing a sound foundation in OO principles.

Target Audience:

This basic course is designed for architects, designers, analysts, developers, testers, administrators, managers, and project managers who will use object-oriented technology to build applications.

Objectives:

- State the advantages of an object-oriented approach to software development
- Describe essential object-oriented concepts and terminology
- Perform OO requirements gathering, analysis, and design
- Describe the role of Unified Modeling Language (UML) in object-oriented analysis and design
- Read the most commonly used types of UML diagrams
- Create UML use case, class, and sequence diagrams
- Describe the impact of designing an application that can accommodate changes and the approaches to support such designs
- Create Java classes that implement an object-oriented design
- Apply Java language constructs that enable and enforce OO-related concepts such as data encapsulation, strict typing and type conversion, inheritance, and polymorphism
- Explain how design patterns can improve the implementation of OO designs
- Describe the incremental and iterative process for developing applications using object technology and how it differs from traditional approaches (for example, waterfall) to application development
- Compare the Rational Unified Process (RUP) and Agile approach as software development methodologies

Prerequisites:

There are no prerequisites for this course.

Content:

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining

- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study

- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)

- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview

- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and

- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Identifying candidate objects in a case study
- Object concepts
- Exercise: Identifying classes and methods in a case study
- Key principles of object-oriented programming
- Exercise: Identifying classes and associations in a case study
- Introduction to UML
- Development project life cycle
- Requirements and use cases
- Exercise: Identifying actors and use cases
- Java technology overview
- Demonstration: Programming Java with the SDK
- Introduction to the Java language
- Demonstration: Using the software development platform
- Exercise: Java programming - Defining some classes
- OO analysis - Static UML diagrams
- Exercise: Finding candidate objects and creating a class diagram
- OO analysis - Dynamic UML diagrams
- Exercise: Developing sequence diagrams
- OO design for implementation - Associations
- Exercise: Refining the design for a case study (optional)
- OO design for implementation - Inheritance
- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

- Exercise: Java programming - Implementing a design
- Designing for change
- Exercise: Java programming - Improving the implementation
- Methodologies
- Course summary

Further Information:

For More information, or to book your course, please call us on 00 20 (0) 2 2269 1982 or 16142

training@globalknowledge.com.eg

www.globalknowledge.com.eg

Global Knowledge, 16 Moustafa Refaat St. Block 1137, Sheraton Buildings, Heliopolis, Cairo