

Hands-on course , 3  
day(s)  
Ref : IPJ

## Participants

Professionals not developing and Project Managers or Business Analysts who want to understand the techniques used by their colleagues or subcontractors. Developers who want to smoothly get used to Object-Oriented Programming.

## Pre-requisites

Not any particular knowledge.

## Next sessions

# Introduction to Object/Java Programming

## OBJECTIVES

*In this workshop, the fundamental principles of Object-Oriented Programming are taught through practice. It brings the participants a clear understanding enabling them, according to their needs, to control the technical relationships with their subcontractors, to fully supervise their development teams, or even to better control the construction of object oriented business requirements. It is also often used as a redeployment springboard towards Object-Oriented Programming.*

### [1\) General Presentation](#)

### [2\) Syntax, Types and Expressions](#)

### [3\) Methods and Statements](#)

### [4\) Using Abstraction](#)

### [5\) Using Inheritance](#)

### [6\) Using Interface](#)

### [7\) Developing Classes](#)

### [8\) Developing Interfaces](#)

### [9\) Developing Derived Classes](#)

### [10\) Exceptions](#)

## Workshop

*Each day is split in three theoretical sessions, each followed by simple exercises in which the principles, the use and the development of objects are illustrated. The participants will be able to get used to Eclipse, leader of Java objet development environments. Those exercises will be performed through Windows XP workstations, the JDK 1.5 and Eclipse 3.2.*

## 1) General Presentation

- Introduction.
- Fundamental principles of Object-Oriented Programming: Abstraction/Encapsulation.
- Fundamental principles of Object-Oriented Programming: Inheritance.
- General presentation: the Language.
- The Tools and the Library.
- Releases of Java.

## 2) Syntax, Types and Expressions

- Syntactic structure of a Java application.
- Syntax example of a simplified application.
- External view of a class: Use Syntax.
- Internal view of a class: Implementation Syntax.
- Concept of Type.
- Compared use of Primitive Types and Object Types.
- Simple use of Primitive Types: Integers, Floats, Characters and Booleans.
- Concept of Expression.
- Examples of Declarations: Variables and Constants.
- Compared declarations of Primitive Types and Object Types.
- Using Operators with Objects.
- Special case of Static Fields or Class Variables.
- Conversions between Primitive Types and Object Types.
- Code Conventions.

## 3) Methods and Statements

- Syntax of Method Calls.
- Class Methods and Instance Methods.
- Definition and Use of Methods.
- Method Overriding.
- Concept of Sub-Block.
- Categories of Statements.
- Main Control Statements: `#if#`, `#while#`, `#for#`, `#return#`, `#break#`.

## 4) Using Abstraction

- Simple example of Use of an Object: Declaration, Instantiation or Creation, Delegation.
- Using Object Constructors.
- Using the Programming Interface of Objects: example of the Date class.
- A very useful class: the String class.
- Characteristics of Character Strings.
- Using the StringBuffer class: example of use of Method Overloading.

## 5) Using Inheritance

- Reminder of the Inheritance Principle and Terminology.
- Using Inheritance.
- Example of Inheritance Graph.

- The Object class and Genericity.
- Using Polymorphism.
- Specialization of a Polymorphic Reference.
- Typing of References / Typing of Objects.
- Behavior of Methods and Typing.
- Genericity of Collection Classes: example of the Vector class.
- New Features in Java 5 (Tiger): Generics.

## 6) Using Interface

- Implicit/Explicit Interface of a Class.
- Syntax of Explicit Interfaces.
- Use case of Interface References: Flexibility, Scope Reduction, Polymorphism.
- Example of Implementation of Multiple Interfaces.
- Synthesis of the benefit of Interfaces for Methods.
- Using Interfaces for Constants.
- Advanced examples of Use of Interfaces.

## 7) Developing Classes

- Methodological Approach; Static, Dynamic and Business Analysis.
- UML Notation: Class, State and Sequence Diagrams.
- Class Skeleton: Basic Components, Tools of Code Automatic Generation.
- Additional information about Access Rights.
- Organizing Packages.
- Constraints related to Packages.
- Implementing Constructors.
- Default Constructor.
- Additional information about the Implementation of Constructors.
- Self Reference #this#.
- Static Fields and Methods.
- #main# Method

## 8) Developing Interfaces

- Syntax of Interfaces, case of Constants.
- Definition of Interfaces for Methods.
- Implementing and Extending Multiple Interfaces.
- Partial Implementation of Interface.
- Examples and additional information about the Use of Interfaces.

## 9) Developing Derived Classes

- Reminder of the Principles.
- Methodological Approach for a Decomposition in Classes.
- Abstract Methods and Classes.
- Interfaces and Abstract Classes.
- Inheritance and Access Rights of Fields.
- Inheritance and Call of Constructors.
- Overriding and Overloading.

## 10) Exceptions

- Principles and Event Sequence.
- Catching and Handling an Exception.
- Throwing an Exception.
- Unchecked Exceptions.
- Simple example with Exception Handling.