# Computer Science 22: Object Oriented Programming

Lecture #2: OOP: Motivation and History

## About This Lecture

- Definition of Object Oriented Programming
  - As well as Object Oriented Analysis and Design
- History of Object Oriented Programming
  - Motivations for OOP
  - OOP languages and their creators

# Object Oriented Programming

- Programming using **objects**
- **Method of implementation** in which programs are organized as cooperative collections of **objects**, each of which represents an instance of some **class**, and whose classes are all members of a hierarchy of classes united via **inheritance** relationships.

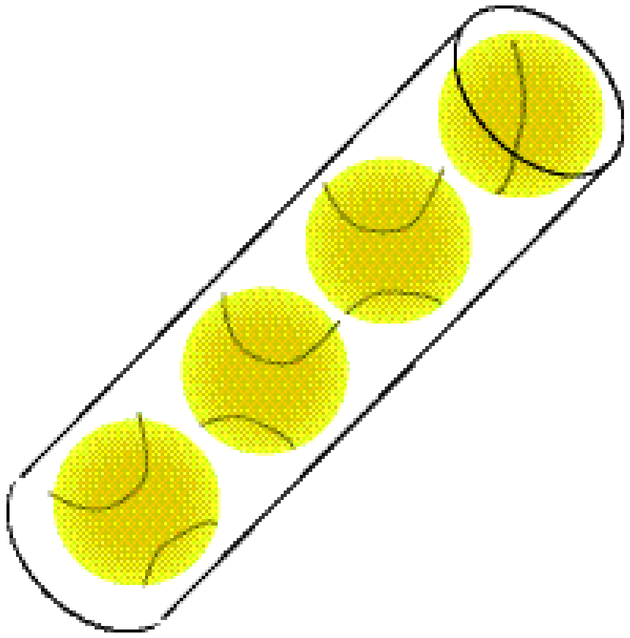
## What makes an Object?

- Descartes (the 17th century philosopher) observed that humans view the world in object-oriented terms. The human brain wants to think about objects, and our thoughts and memories are organized into objects and their relationships.
- One of the ideas of object-oriented programming is to organize the software in a way that matches the thinking style of our object-oriented brains.

## Characteristics of Objects

- An object has identity
- An object has state
- An object has behavior

# Characteristics of Objects



## Software Objects

- Many programs are written to do things concerning the real world. It is convenient to have “software objects” that are similar to “real world objects” making the program and its computation easier to think about.
- Software objects will have identity, state, and behavior just as do real world objects.

# Software Objects

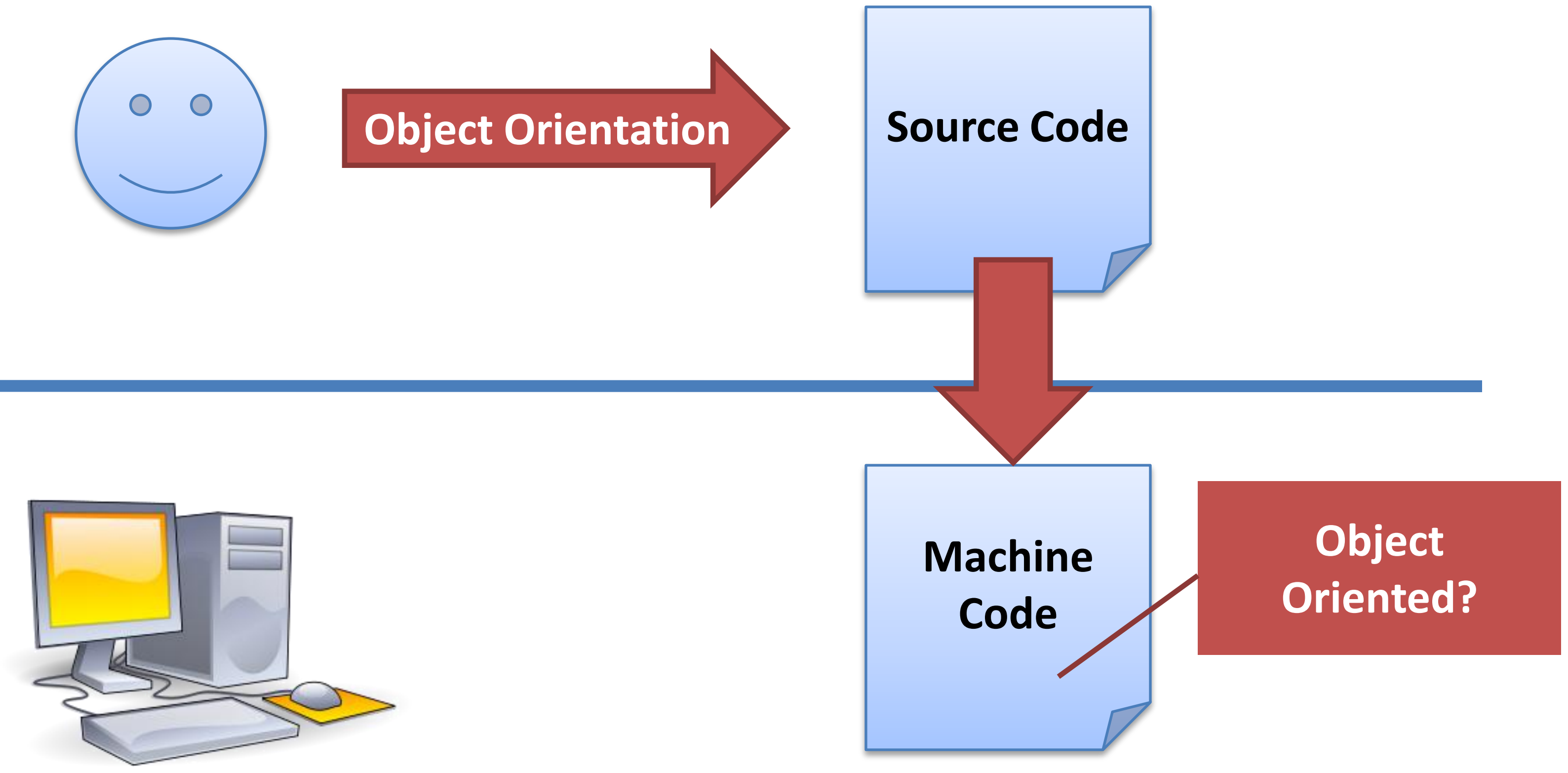
- Objects (real world and software) have **identity**, **state**, and **behavior**
- Software objects have **identity** because each is a **separate chunk of memory**
- Software objects have **state** because some of the memory that makes a software object is used for **variables which contains values**
- Software objects have **behavior** because some of the memory that makes a software object is used to **contain programs** that enable the object to “do things”. The object does something when one of its methods runs.

## Software Objects

- A self-contained programming unit packaged with an enclosed data/data structure and methods that operate on the data.

# The Object Oriented Paradigm

- It is all in the mind of the programmer



# The Object Oriented Paradigm

- A certain arrangement of the source code into logical units called classes to represent objects.
- The programmer/designer/analyst must have an object-oriented view
  - That is, any object is composed of smaller objects that interact with each other to achieve the larger object’s purpose and/or functions

## The Object Oriented Problem Solving Process

- **Object Oriented Analysis**
  - Method of analysis that examines requires from the perspective of classes and objects found in the vocabulary of the problem domain
- **Object Oriented Design**
  - Method of design encompassing the process of object-oriented decomposition and a notation for depicting models of the system under design

## Main Concepts in Object Orientation

- **Abstraction**
- **Encapsulation**
- **Modularity**
- **Hierarchy**
- **Typing, Concurrency, and Persistence**

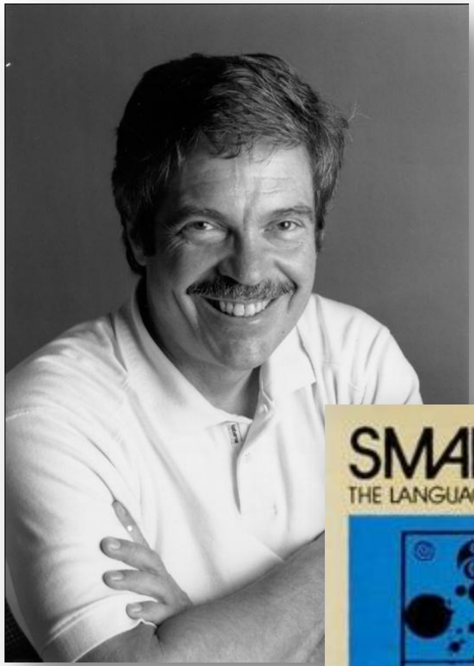
## Bit of History

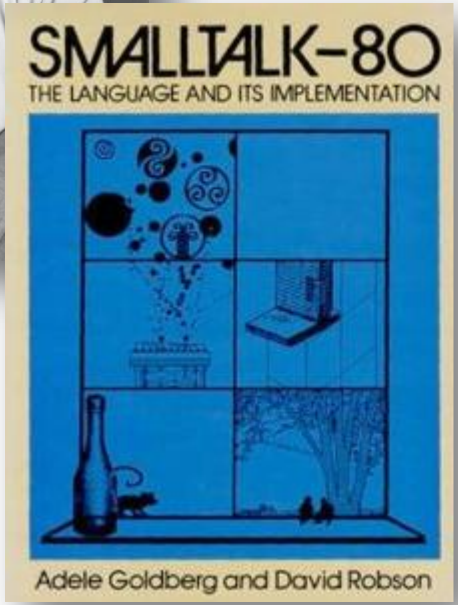
- SIMULA (Simulation Language) introduced object-oriented programming concepts such as object, class, dynamic binding, etc.

**Ole-Johann Dahl and Kristen Nygaard**
**Fathers of Object Technology**
**Creators of Simula I and Simula 67**

## Bit of History

- 1970s – Alan Kay led a tem that developed Smalltalk at Xerox PARC
  - inspired by Simula, graphics-driven, used in Dynabook
  - Coined the term "Object Oriented Programming"



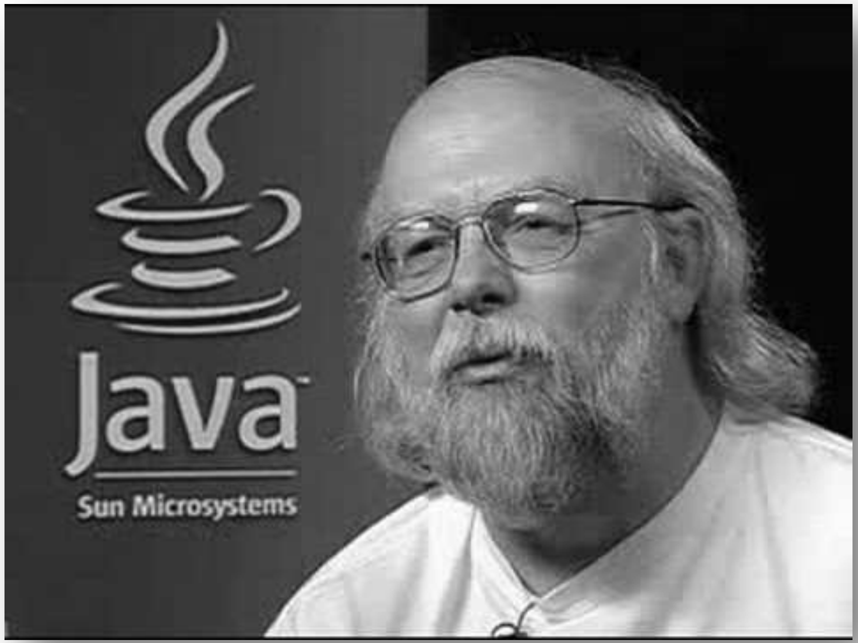

SMALLTALK-80
THE LANGUAGE AND ITS IMPLEMENTATION
Adele Goldberg and David Robson

## Bit of History

- Bjarne Stroustrup implemented “C with Classes” aka C++
- Became  the most widely used OO language

## Bit of History

- 1990s – James Gosling and his team at Sun developed Java
- Java rode with the popularity of the Internet*
- Simplified C++



# Other Notable Languages

- **Eiffel**
- **Objective-C**
- **Modula-2**
- **Oberon**
- **Python**
- **Ruby**
- **The ‘Visual’ languages**
- **CLOS**

## Trends in Programming Languages

- Since the 1990’s, Object-Oriented Analysis, Design, and Programming is the mainstream paradigm for developing software
- Previous non-object-oriented programming languages are now rewritten with OO capabilities
- New languages are usually “multi-paradigm”

## Criteria for OOPL

- Supports objects that are data abstractions with interface of named operations and hidden local state (**abstraction**)
- Objects have an associated type [class] (**encapsulation**)
- Types [classes] may inherit attributes from supertypes [superclasses] (**inheritance**)

## VS. Object Based PL

- Also uses objects
- Programmers use built-in objects and may not create new object types
- Use in specialized environments (i.e., database, game engines, etc)
- Cannot do inheritance

## Types of OOPLs

- Class-based OOPLs
  – Makes use of class to define objects
- Prototype-based OOPLs
  – Define a prototype which becomes the basis for new objects
- Pure OOPL
  – All “datatypes” are objects
- Hybrid OOPL
  – Mix of primitives and objects