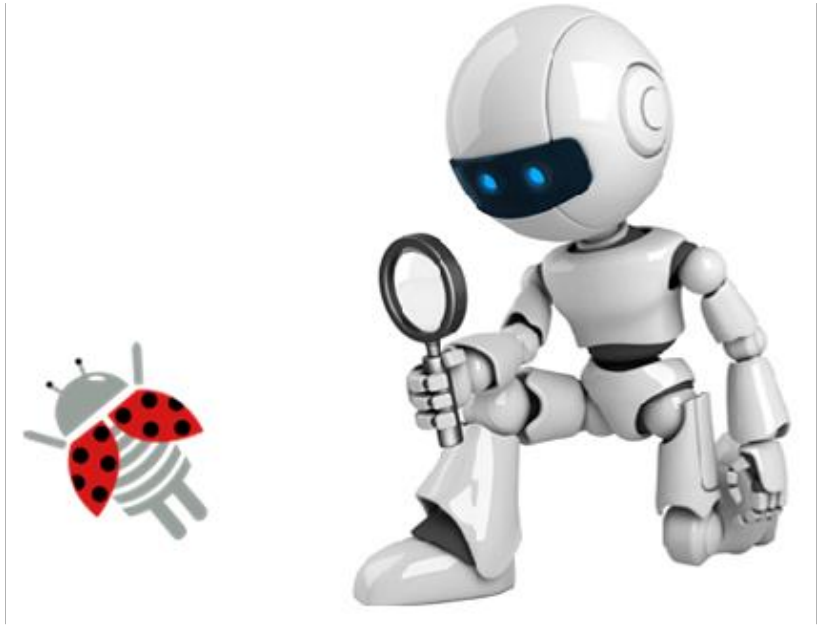
QATestify.com

Test automation course



Lesson 2

QATestify.com

Table of contents

QATestify.com

Basic terms (reminder and some new)

Everything is an object.

Think of an object as a fancy variable; it stores data, but you can "make requests" to that object, asking it to perform operations on itself.

In theory, you can take any conceptual component in the problem you're trying to solve (dogs, buildings, services, etc.) and represent it as an object in your program.

QATestify.com

Basic terms (reminder and some new)

A program is a bunch of objects telling each other what to do by sending messages.

To make a request of an object, you “send a message” to that object. More concretely, you can think of a message as a request to call a method that belongs to a particular object.

QATestify.com

Basic terms (reminder and some new)

Each object has its own memory made up of other objects.

Put another way, you create a new kind of object by making a package containing existing objects. Thus, you can build complexity into a program while hiding it behind the simplicity of objects.

QATestify.com

Basic terms (reminder and some new)

Every object has a type.

Using the parlance, each object is an instance of a class, in which “class” is synonymous with “type.” The most important distinguishing characteristic of a class is “What messages can you send to it?”

QATestify.com

Basic terms (reminder and some new)

All objects of a particular type can receive the same messages.

This is actually a loaded statement, as you will see later. Because an object of type “circle” is also an object of type “shape,” a circle is guaranteed to accept shape messages.

This means you can write code that talks to shapes and automatically handle anything that fits the description of a shape. This substitutability is one of the powerful concepts in OOP.

QATestify.com

Basic terms (reminder and some new)

So, an object has state, behavior and identity.

This means that an object can have internal data (which gives it state), methods (to produce behavior), and each object can be uniquely distinguished from every other object—to put this in a concrete sense, each object has a unique address in memory

QATestify.com

Basic terms (reminder and some new)

In the real world, you'll often find many individual objects all of the same kind. There may be thousands of other bicycles in existence, all of the same make and model.

Each bicycle was built from the same set of blueprints and therefore contains the same components.

In object-oriented terms, we say that your bicycle is an instance of the class of objects known as bicycles.

A class is the blueprint from which individual objects are created.

QATestify.com

Basic terms (reminder and some new)

Class diagrams are widely used to describe the types of objects in a system and their relationships. Class diagrams model class structure and contents using design elements such as classes, packages and objects.

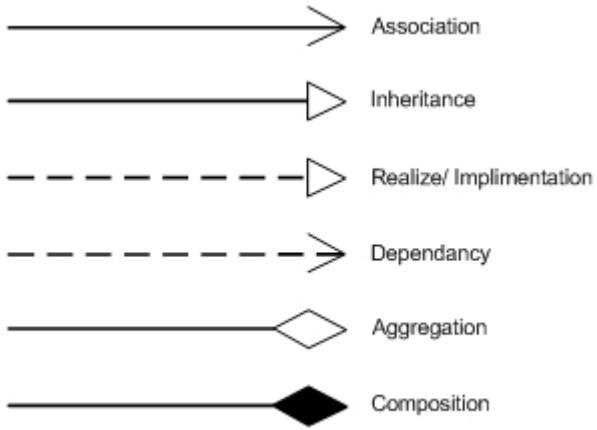
Class diagrams describe three different perspectives when designing a system, conceptual, specification, and implementation. These perspectives become evident as the diagram is created and help solidify the design.

QATestify.com

Basic terms (reminder and some new)

The Class diagrams, physical data models, along with the system overview diagram are the most important diagrams that suite the current day rapid application development requirements.

Appropriate UML notations:



Homework

1) Read about UML notations from previous slide here:
http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep

2) Read about basic data types here
http://www.tutorialspoint.com/java/java_basic_datatypes.htm

3) Read about access modifiers here
http://www.tutorialspoint.com/java/java_access_modifiers.htm

4) Read about non-access modifiers here
http://www.tutorialspoint.com/java/java_nonaccess_modifiers.htm

5) Read about Java constructor
http://www.javabeginner.com/learn-java/java-constructors/

Homework

6) Read about **this** keyword in Java
https://docs.oracle.com/javase/tutorial/java/javaOO/thiskey.html

7) Read about **void** keyword in Java https://ru.wikipedia.org/wiki/Void

8) Read about **return** keyword in Java
http://www.tutorialspoint.com/java/java_access_modifiers.htm

9) Read about creation of class instances (objects)
http://www.tutorialspoint.com/java/java_object_classes.htm

Homework

10) Create 2 classes which will interact with classes created in Lesson 1. Use UML Notation http://www.tutorialspoint.com/uml/uml_class_diagram.htm to describe these structures and Microsoft Vision to draw. Store these classes diagrams to Homework\Lesson2 folder. Give full name using javaCamelNotation (https://en.wikipedia.org/wiki/CamelCase) on your choice

11) Create detailed presentation of Lesson2 with pictures and links based on style and template of Lesson1. 1-9 items from homework to serve as Table of Contents