UML: Uniform Modeling Language

UML is a standardized design language for object-oriented programming in various languages. The website for UML is http://www.uml.org. UML diagrams can be classified into four types.

- Class Diagram. Shows relationships between various classes in a project.
- Object Diagram. Shows interactions between various objects in your project.
- Collaboration Diagram. Shows associations between various objects. Similar to relationships between classes but differs since we usually show the methods being called and values being returned.
- Sequence Diagram. Shows interactions between various objects based on a time-line.

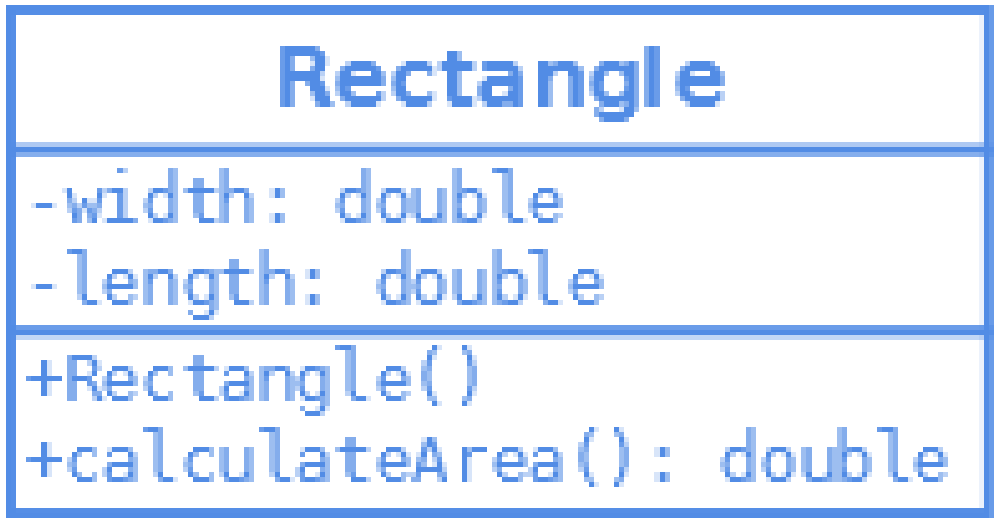
Class Diagrams

Classes are drawn as rectangles, which may be divided into 1, 2 or 3 partitions. The top partition is for the class name, the second one for the class variables, and the third one for the methods or operations. Each variable/method is preceded by a *visibility indicator*.

- $+$ indicates `public`
- - indicates `private`
- $\#$ indicates `protected`

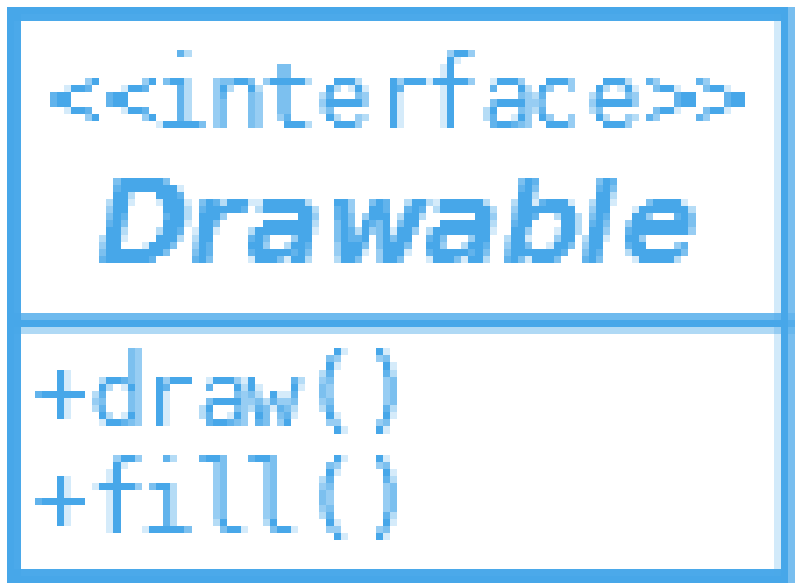
Interfaces are shown as classes except it has only two partitions. A interface name is preceded by a stereotype tag to show that it is a special kind of a class.
Methods/classes that are `abstract` are shown italicized.

UML Class Diagram



We are using the Dia program to generate the UML diagrams. It is available for free for Linux, MS Windows, MacOS X from http://projects.gnome.org/dia/

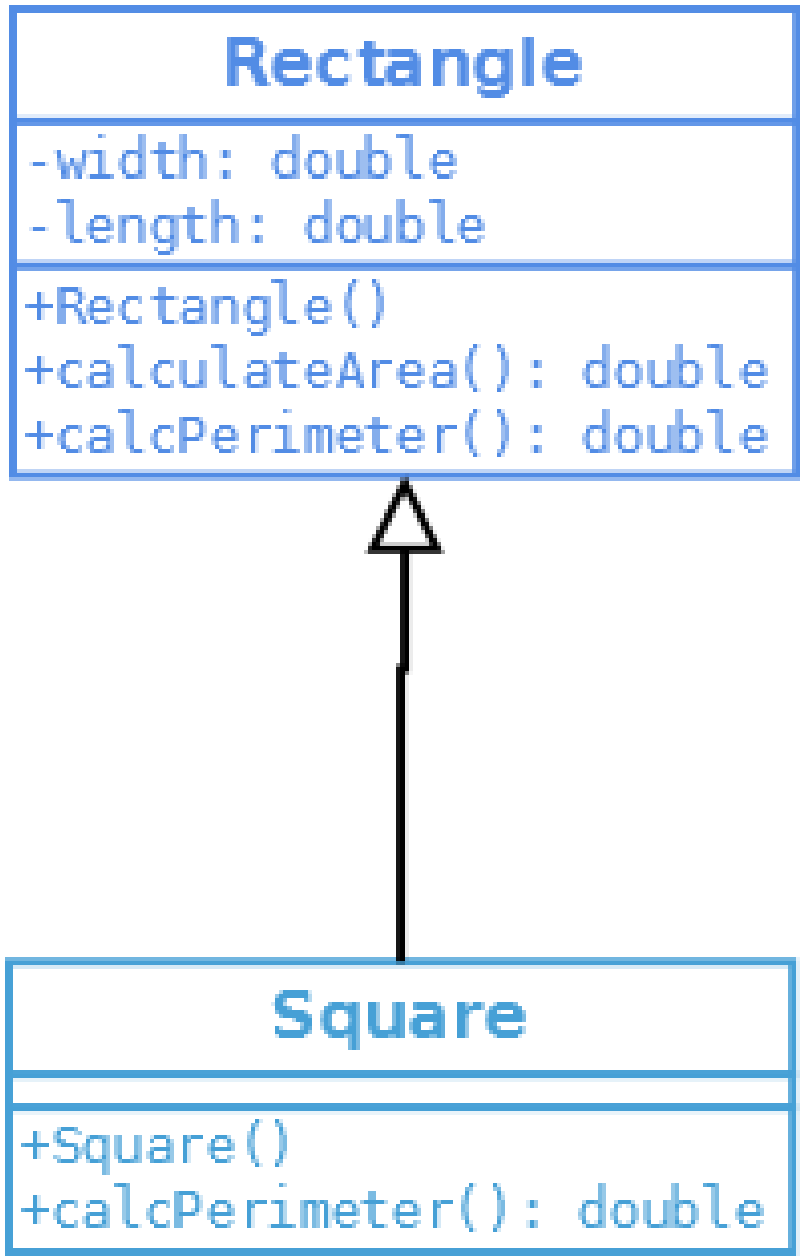
UML Interface Diagram



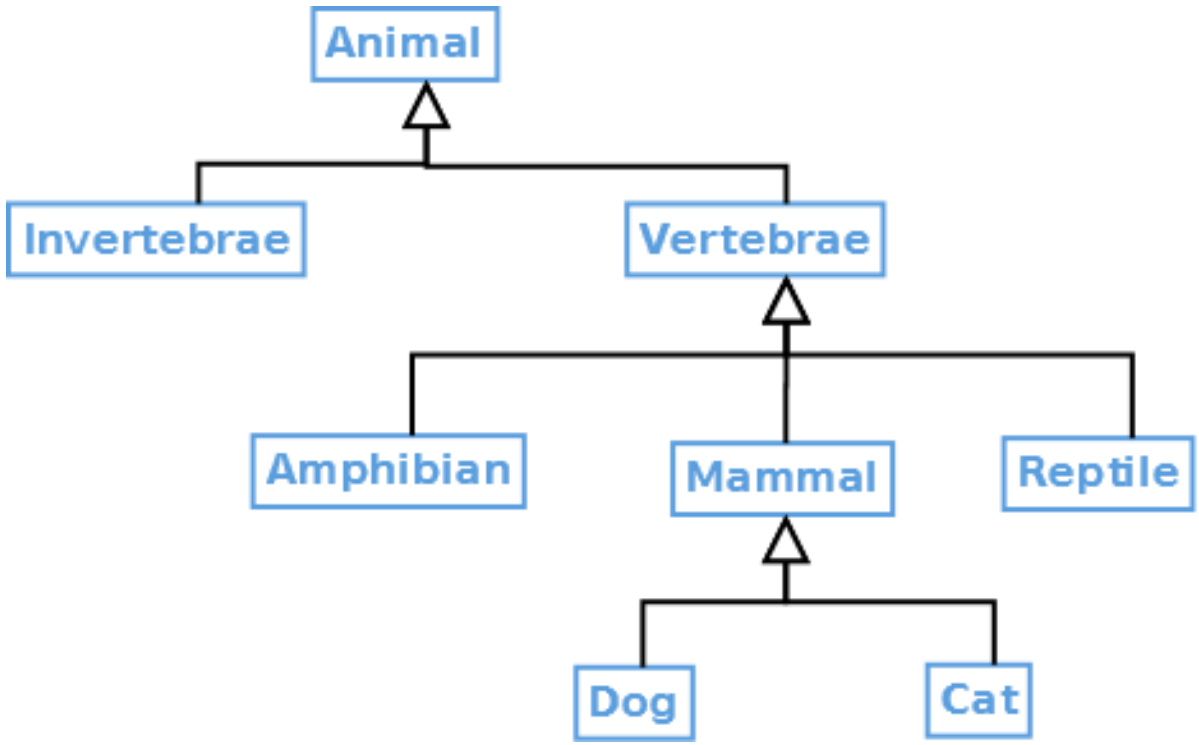
Inheritance and Associations

- Inheritance. A solid line with a closed arrowhead from the subclass to the superclass.
- Implements. A dashed line with a closed arrowhead from the implementing class to the interface.

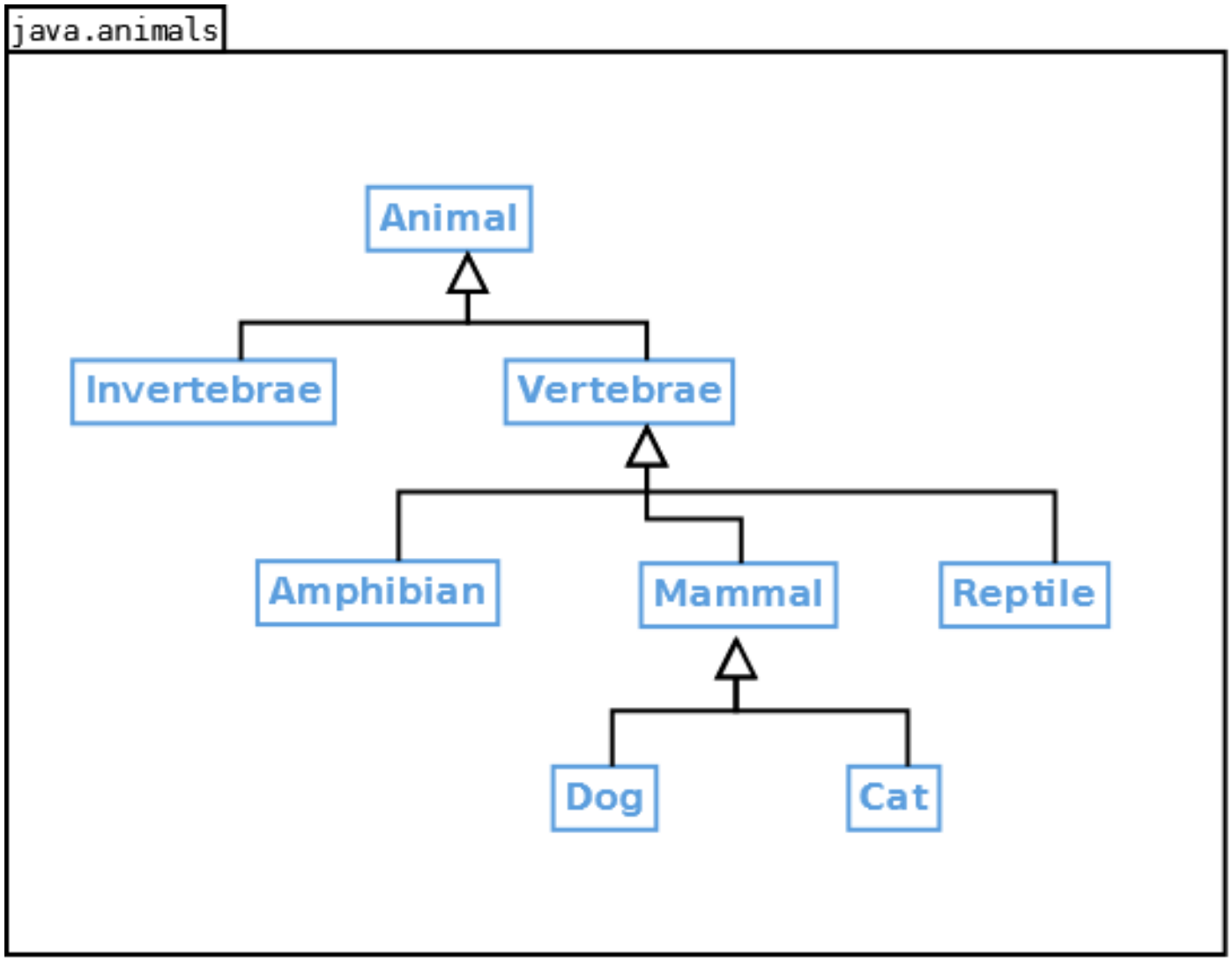
Inheritance



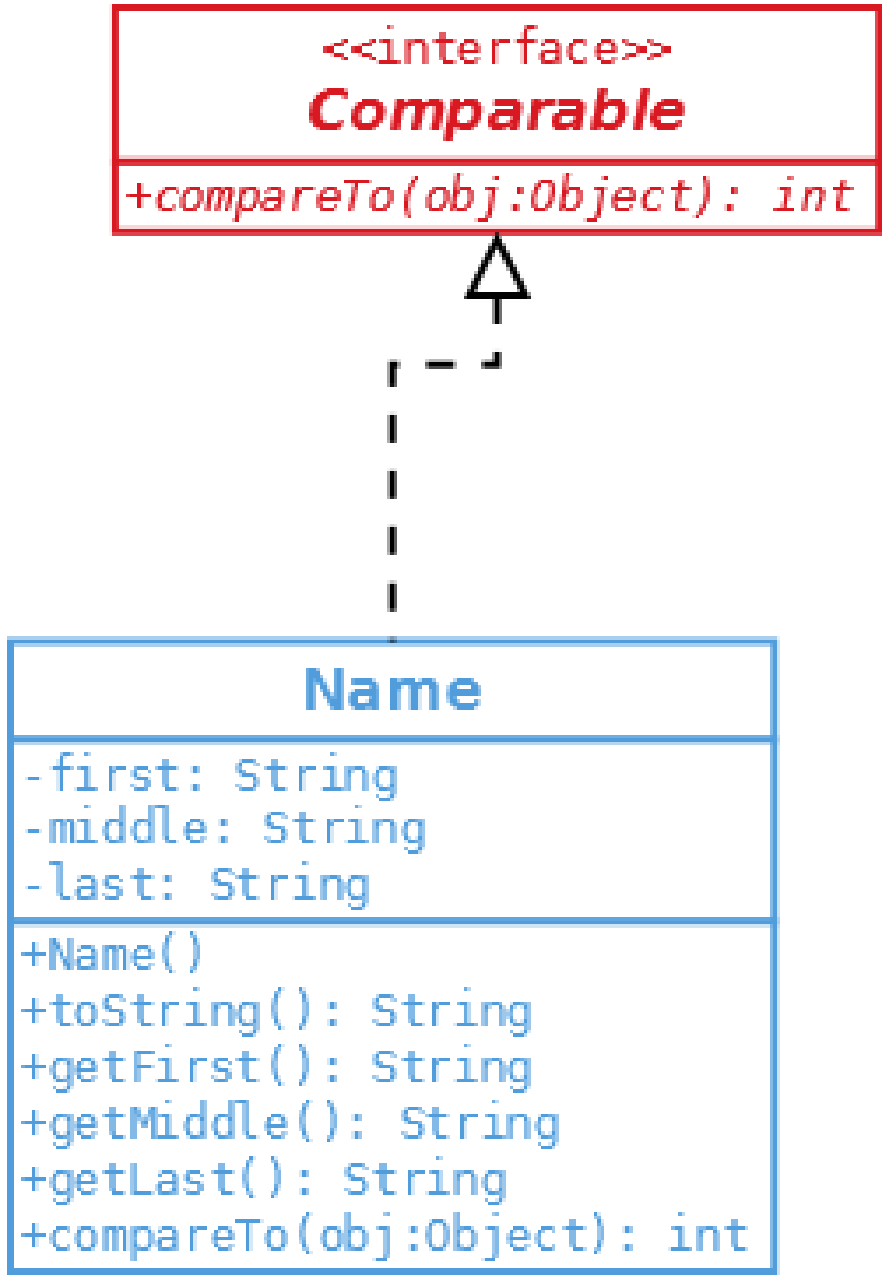
Inheritance Hierarchy



A Package


java.animals
Animal
Invertebrae
Vertebrae
Amphibian
Mammal
Reptile
Dog
Cat

Implements

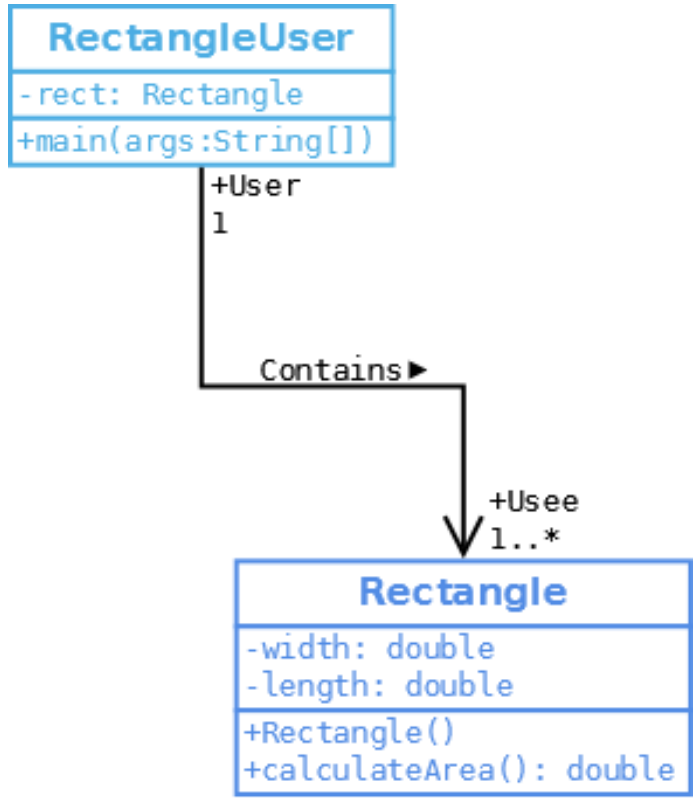

<<interface>>
Comparable
+compareTo(obj:Object): int
Name
-first: String
-middle: String
-last: String
+Name()
+toString(): String
+getFirst(): String
+getMiddle(): String
+getLast(): String
+compareTo(obj:Object): int

Associations

- Associations. A solid labeled line shows that two classes are associated. A small solid triangle (right next to the label) shows the direction of the association. An optional open arrowhead can be used to denote the direction of the association.
  Numbers at either end indicate are the *multiplicity indicators*.
  Here are are some examples of multiplicity indicators.

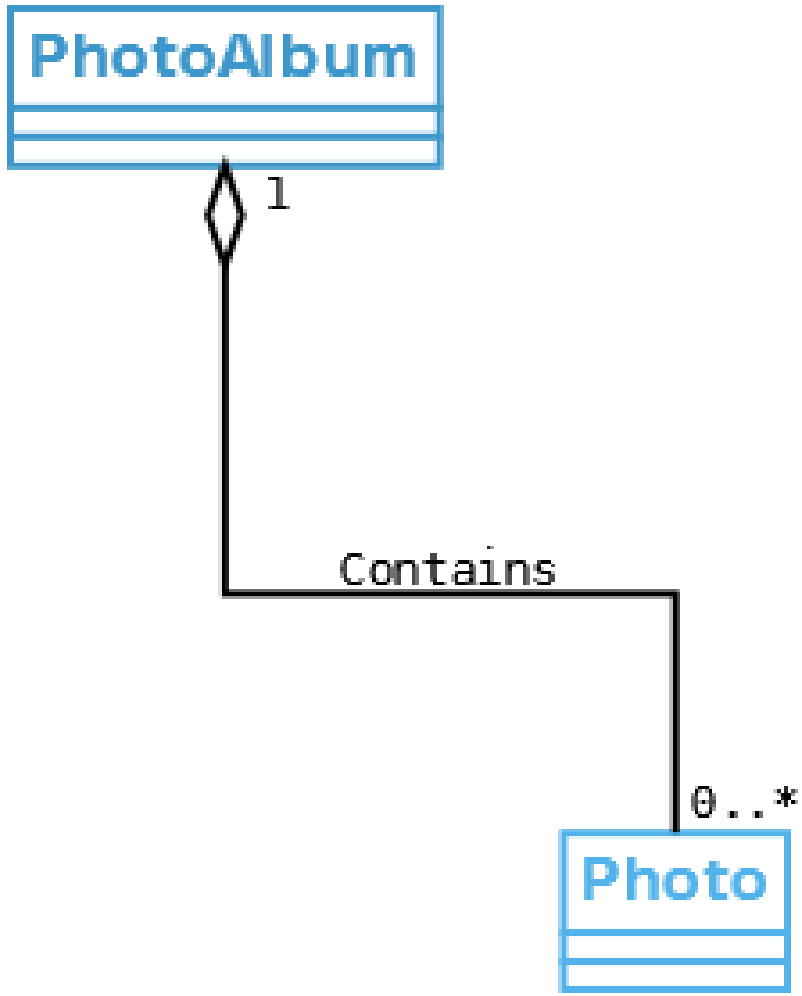
1	one
0..2	zero to two
0..*	zero or more
*	zero or more
1..*	one or more

Association Example

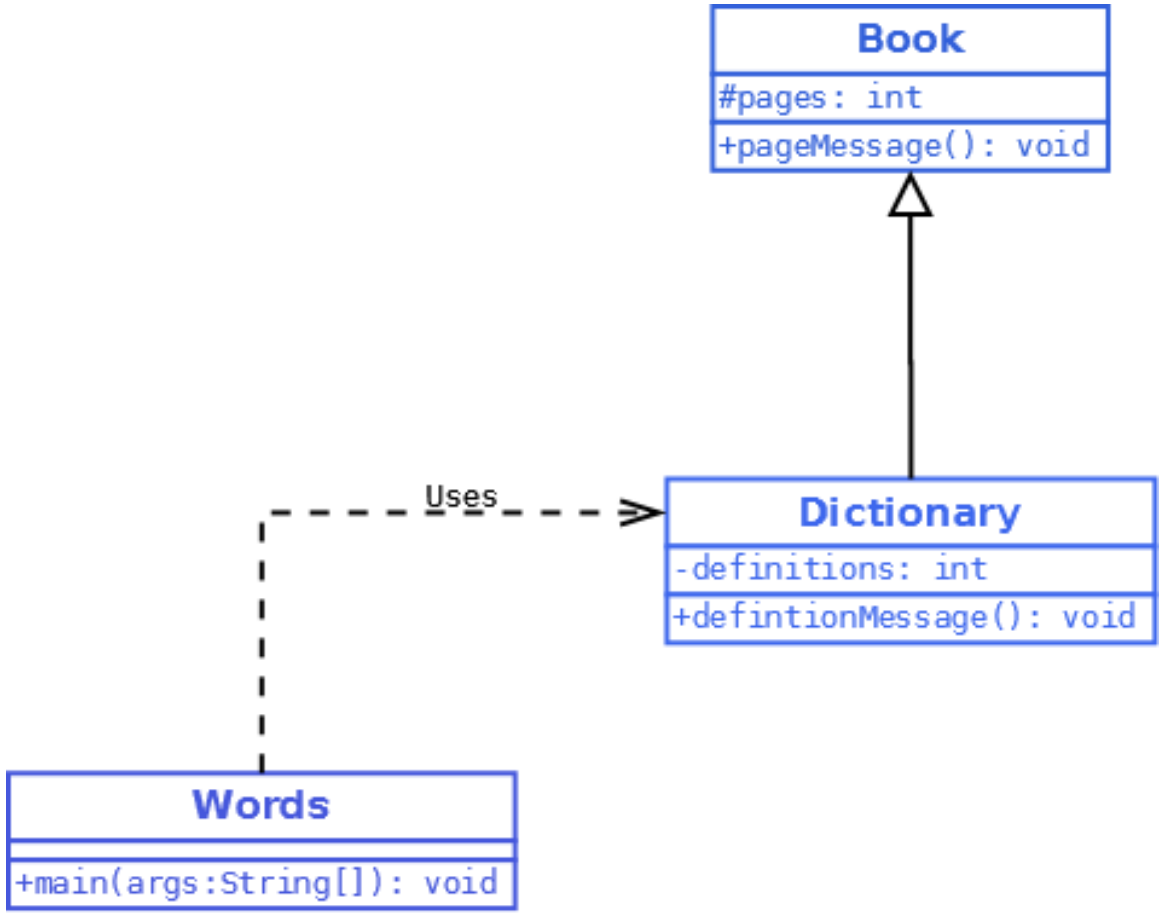


Aggregation

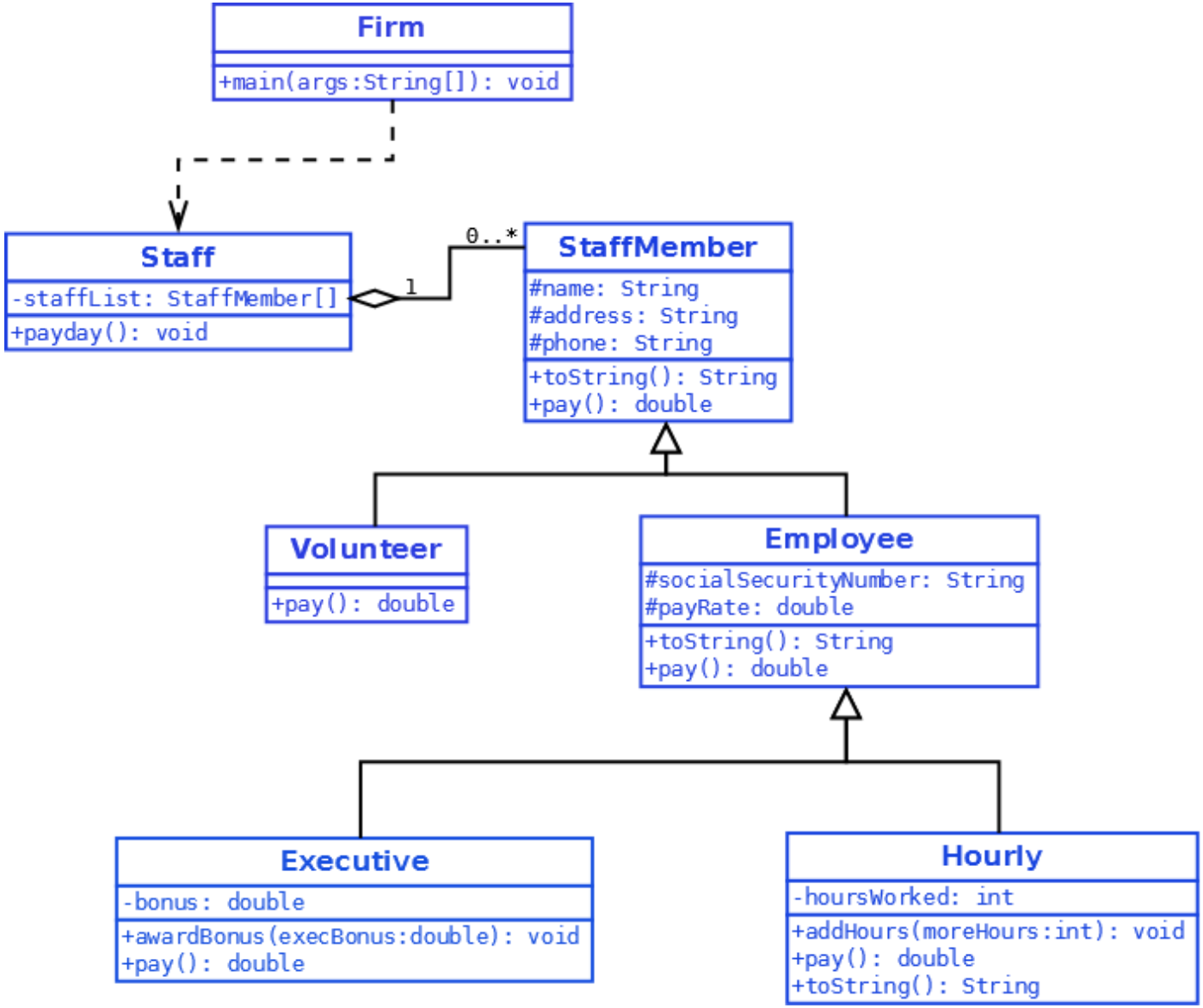
- An Aggregation is an association in which one class contains many others. The container class has a diamond on its end of the association line.



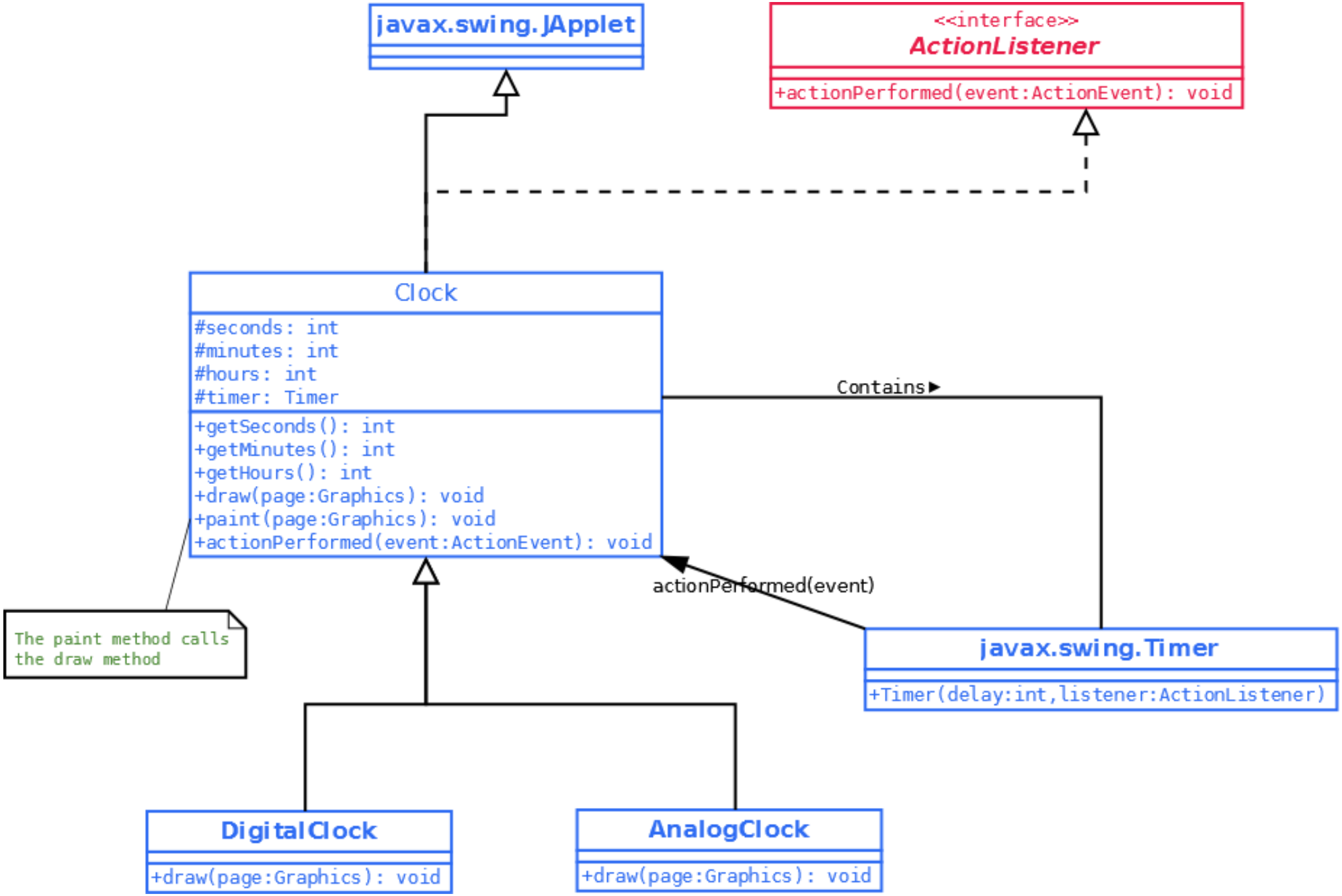
Inheritance and Dependency



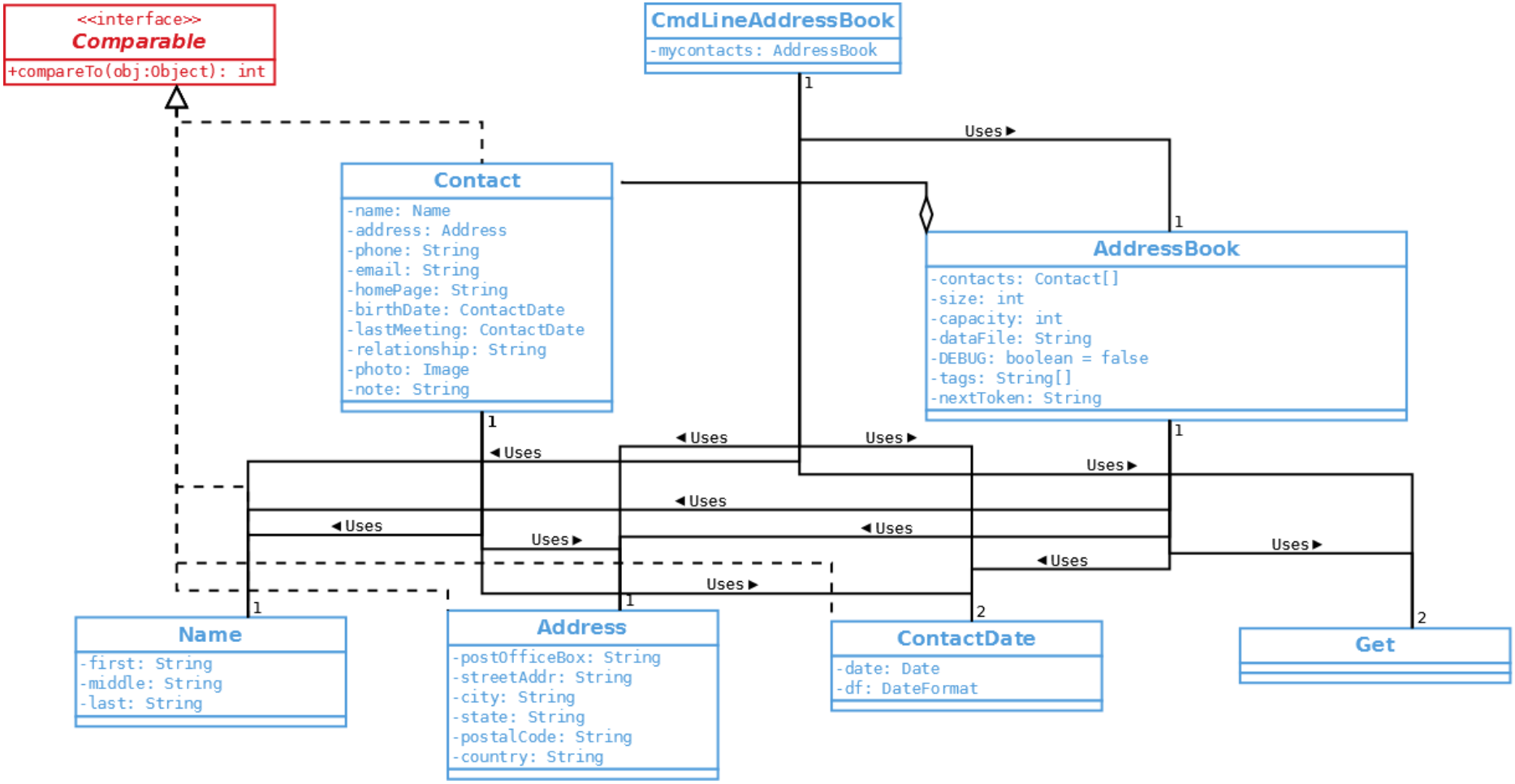
Employees Example



A Clock Heirarchy Diagram

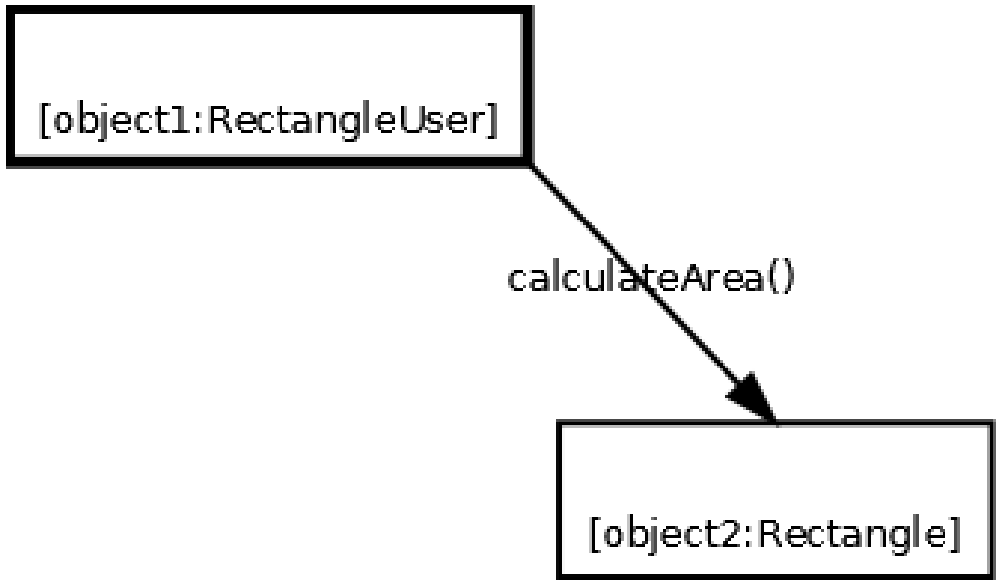


The Address Book Class Diagram



Object Diagrams

Objects are drawn the same way as classes with 1/2/3 partitions. The difference is that names for objects are underlined and consist of the name and type of the objects.
When depicting object interactions, an active object is shown in bold face.
Objects send messages to each other (through method class and return values). This are shown as labeled links with an arrow denoting the direction.



Object Interaction