**Class diagram**

* used in designing and modeling software to describe classes and their relationships.
* Describes the structure of an object but not its functionality.

Reading class diagrams using Unified Modeling Language (UML)

* + public
* - private
* → show the direction of the connection
* - \* a star pinpointing to a certain class means that the reference of another class as a collection set and the one without a star means that the reference of that class is only called once.
* \*-\* two stars pinpointing to each classes means that the reference of each other classes as a collection set
* ↑ the arrowhead signifies the super/parent class, and the tail signifies the sub/child class
  + <<abstract>>
* > dashed arrow with a triangle arrowhead signifies a class implements an interface (arrowhead)
  + <<interface>>

|  |  |
| --- | --- |
| Source Code of Person | Unified Modeling Language |
| public class Person {  private String name;  private int age;   public Person(String initialName)  {  this.name = initialName;  this.age = 0;  }   public void printPerson()  {  System.*out*.println(this.name + ", age " + this.age + " years");  }    public String getName()  {  return this.name;  } } | |  | | --- | | Person | | -name:String  -age:int | | +Person(initialName:String)  +printPerson():void  +getName():String | |

**Connections between classes**

|  |  |
| --- | --- |
| Source Code of Book only | Unified Modeling Language |
| public class Book {  private String name;  private String publisher;  private ArrayList<Person> authors;   // constructor   public ArrayList<Person> getAuthors() {  return this.authors;  }   public void addAuthor(Person author) {  this.authors.add(author);  } } | [Person|-name:String;-age:int|+Person(initialName:String);+printPerson():void;+getName():String][Book|-name:String;-publisher:String|+getAuthors():ArrayList;+addAuthor(author:Person)][Book]-*->[Person] |

Explanation:

Star – ArrayList<Person> authors (Book class)

Without star – Book book (Person class)

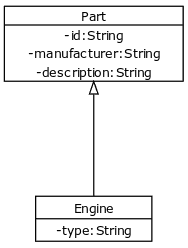
|  |  |
| --- | --- |
| Source Code both of Person and Book | Unified Modeling Language |
| public class Person {  private String name;  private int age;  private ArrayList<Book> books;   // ... }  public class Book {  private String name;  private String publisher;  private ArrayList<Person> authors;   // .. } | [Person|-name:String;-age:int|+Person(initialName:String);+printPerson():void;+getName():String][Book|-name:String;-publisher:String|+getAuthors():ArrayList;+addAuthor(author:Person)][Book]*-*[Person] |

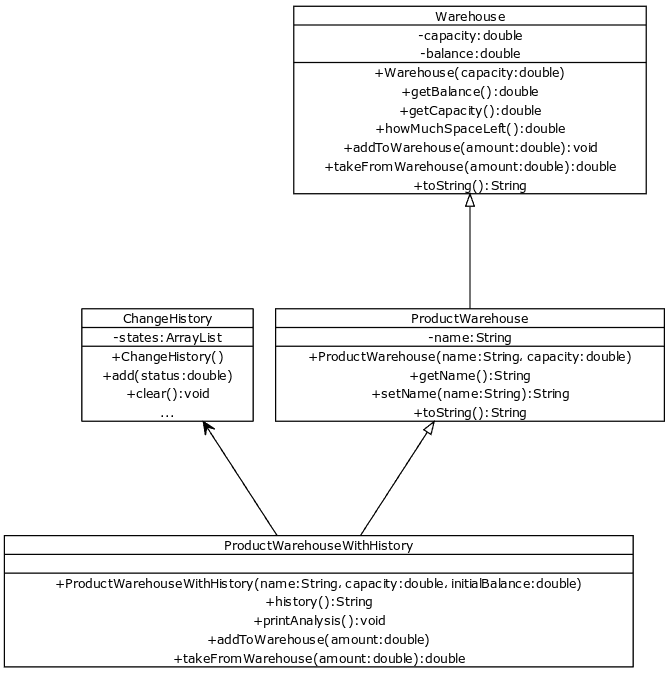
Explanation:

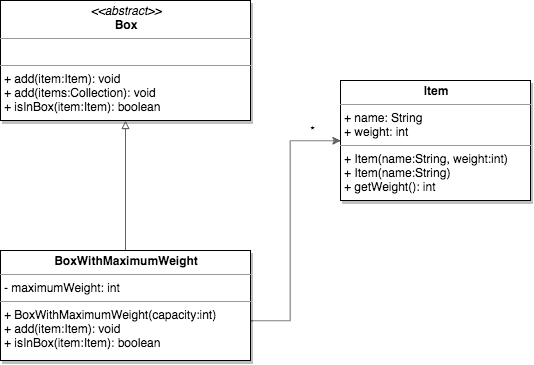
Star – ArrayList<Person> authors (Book class)

Star – ArrayList<Book> books (Person class)

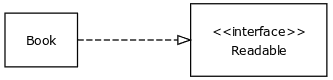
**Describing Inheritance**



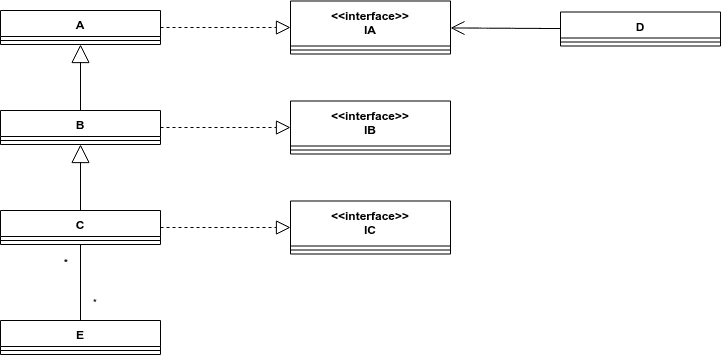




**Interface**



**Relationship of Classes and Interfaces**



public class A implements IA  
{  
}

public class B extends A implements IB{  
}

import java.util.\*;  
public class C extends B implements IC  
{  
 private List<E> e;  
}

public class D  
{  
 private IA ia;  
}

import java.util.\*;  
public class E {  
 private List<C> c;  
}

public interface IA {  
}

public interface IB {  
}

public interface IC {  
}

YUML

<http://yuml.me/preview/75f7a566>

A white rectangular sign with black text

Description automatically generated

public class Teacher{

private String name;

public Teacher(String name) {

this.name = name;

}

public String toString() {

return this.name;

}

}

public class Class{

private Teacher teacher;

private String course;

public Class(Teacher teacher, String course) {

this.teacher = teacher;

this.course= course;

}

public void printInformation() {

System.out.println(this.course + ", teacher: " + this.teacher);

}

}

import java.util.ArrayList;

public class LearningInstitution {

private ArrayList<Teacher> teachers;

public LearningInstitution() {

this.teachers = new ArrayList<>();

}

public void addTeacher(Teacher teacher) {

this.teachers.add(teacher);

}

public String toString() {

return "Number of teachers in the learning institution: " + this.teachers.size();

}

}