<http://learnwithharsha.com/my-first-java-lesson-is-a-and-has-a-relationships/>

You are here: [Home](http://learnwithharsha.com/) / *My First java lesson “IS-A and HAS-A relationships”*

[**My First java lesson “IS-A and HAS-A relationships”**](http://learnwithharsha.com/my-first-java-lesson-is-a-and-has-a-relationships/)

By [SriHarsha Bolisetti](http://learnwithharsha.com/author/sriharsha/" \o "SriHarsha Bolisetti) on February 8, 2013

I am assuming you all guys have basic idea about oops and java, as all my lessons would be targeted upon nailing the OCJP 6 certification.

**IS-A Relationship : Interface to class relationship**

* This refers to inheritance or implementation.
* Expressed using keyword “extends”.
* Main advantage is code reusability.

Lets look an example to understand about inheritance better.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | **class** P    {    **public** **void** m1() {}    }    **class** C **extends** P    {    **public** **void** m2() {}    }    **class** Test    {    **public** **static** **void** main (String [] args)    {   * *//case 1* What ever the child class has, by default is not available to parent, hence on the parent class reference we can only call parent class methods but not child specific methods.     P p = **new** P();    p.m1(); *// compiles fine*    p.m2(); *// Will get a compile time error as "Cannot find m2()"*     * *//case 2* Whatever the parent class has, is by default available to the child. Hence by using child reference, we can call both parent and child class methods.   C c = **new** C();    c.m1(); *// compiles fine*    c.m2(); *// compiles fine*    *//case 3*   * //3rd point Parent class reference can be used to hold child class objects , but by using that reference we can call only parent class methods but not child specific methods. * P p = **new** C(); * p.m1(); *// compiles fine*     p.m2(); *// Will get a compile time error as "Cannot find m2()"*    *//case 4* We cant use child class reference to hold parent class objects    C c = **new** P(); *// Compiler error as "incompatible types"*    }    } |

OBSERVATIONS from above example:

* Whatever the parent class has, is by default available to the child. Hence by using child reference, we can call both parent and child class methods.
* What ever the child class has, by default is not available to parent, hence on the parent class reference we can only call parent class methods but not child specific methods.
* Parent class reference can be used to hold child class objects , but by using that reference we can call only parent class methods but not child specific methods.
* We cant use child class reference to hold parent class objects

Entire java API is implemented based on *inheritance*. Every java class extends from Object class which has most common and basic methods required for all java classes. Hence we can say “Object ” class is root class of all java methods.

A point to remember on inheritance ..  A java class cannot extend more than one class at a time so it wont provide support for multiple inheritance in classes, but it can extend more than one interface at a time so we can say java provides support for multiple inheritance w.r.t. interfaces.

**HAS-A Relationship -- Class to class relationship**

* Has-A means an instance of one class “has a” reference to an instance of another class or another instance of same class.
* It is also known as “composition” or “aggregation”.
* There is no specific keyword to implement HAS-A relationship but mostly we are depended upon “new” keyword.

**Composition :**

* Without existence of container object, if there is no chance of existence of contained objects then container and contained objects are said to be strongly associated and this strong association is known as *composition*.

Eg: A “university” has several “departments”. Without existence of “university” there is no chance for the “departments” to exist. Hence “university” and “departments” are strongly associated and this strong association is known as *composition*.

**Aggregation**

* Without existence of container object, if there is a chance of existence of contained objects then container and contained objects are said to be loosely associated and this strong association is known as *aggregation.*

Eg: A  “department” has several “professors”. Without existence of “departments” there is good chance for the “professors” to exist. Hence “professors” and “department” are loosely associated and this loose association is known as *Aggregation*.

Thats the end of the my first lesson. I know that’s not much but from now on wards i will first give you an example and then explain topics based on that example

Next lesson *“Overloading” vs “Overriding”.*