OBJECT TYPE CASTING

We can use parent reference to hold child object.

Example: Object o = new String(“durga”);

We can use interface reference to hold implemented class object.

Example: Runnable r = new Thread();

A b = (C) d;

A & C – class or interface

Compiler is going to 2 things

Is this conversion is legal or not

Is this assignment is legal or not

JVM having 1 checking

Compiler Rule 1:

The type of ‘d’ and ‘C’ must have some relation either child to parent or parent to child or same type. Otherwise we will get compile time error saying incovertible types found d type required C

Object o = new String(“durga”);

StringBuffer sb = (StringBuffer) o; // valid during compile time

String s = new String(“durga”);

StringBuffer sb = (StringBuffer) s; //invalid

Compiler Rule 2:

‘C’ must be either same or derived type of ‘A’. otherwise we will get ‘incompatible types’ error.

JVM(Runtime Checking)-Runtime Object type of ‘d’ must be either same or derived type of ‘C’. Otherwise we will get runtime exception saying classcast

Object o = new String(“durga”);

StringBuffer sb = (StringBuffer) o; // invalid during run time

Object o = new String(“durga”);

String str = (String) o; // valid

Strictly speaking thru type casting we are not creating any new object for the existing object we are providing another type of reference variable. i.e. we are performing type casting but not object casting.

String s = new String(“Suganthan”);

Object o = (Object)s;

Type casting:

Object obj = **new** Integer(10);

Integer i = **new** Integer(10);

Number n = i;

Object obj1 = n;

Note:

A -> B -> C

(A)((B)C)

which is short form of

B b = new C()

A a = new C()

Note:

Be careful while doing method calls on type casting

P --> m1();

C --> m1();

((P)C).m1(); //valid

((P)C).m2(); //pakka invalid

Parent refernce can be used to hold to child object but not child object method.

A --> m1();

B --> m1();

C --> m1();

C c = new C();

c.m1(); --> c

((B)C).m1(); --> c

(A(B(C))).m1(); -- c

In overriding method resolution always based on run-time object and based on run-time object.

Example 3:

A --> static m1();

B --> static m1();

C --> static m1();

C c = new C();

c.m1(); --> c

((B)C).m1(); --> b

(A(B(C))).m1(); -- a

In method hiding method resolution was taken care by compiler and based on reference type.

Same goes for variable types as well.