

Type Wrappers Classes

★★.

boxing, unboxing
auto boxing, auto unboxing

Unit-1

| | | | |
|------|------------------------------|-----------------|----------|
| Char | byte short int long | float double | boolean. |
|------|------------------------------|-----------------|----------|

- There are the 8 primitive data types (PDT)
They are built-in & are implemented in a simple manner (w/o classes).
- But, sometimes we need obj representation of these PDT.
- Type Wrappers (TW) are used to represent PDT as obj & vice versa.

Q) Why do we need TW? Why do we need to represent PDT as obj?

A) 1) To call fun by reference.

- PDT are passed by value to fun. So fun is called by value when PDT are passed.
- If we need to call fun ref, then obj needs to be passed.

- So PDT is converted to obj & passed to the fun. So fun will be called by ref.

2) Collections framework (Collections classes & similar to STL) provide implementation of data struct.

- These classes work only on obj, so PDT need to be converted to obj.

3) obj is also needed to support synchronization in multi threading.

4) java.util package handles only obj.


Q) Name some type wrapper classes.

| | | | |
|-----------|---------|--------|---------|
| Character | Byte | Float | Boolean |
| | Short | Double | |
| | Integer | | |
| | Long | | |

- ^{First} ~~each~~ letter is capital
- Character & Integer (not Char & Int).
- *. TW are classes that encapsulate PDT within an obj.

Q) What is boxing? How is it done?

Ans . The process of converting PDT to obj is called boxing.

• `int a = 10` \longrightarrow 

- Here, int var is converted to an obj of Integer class.

- It can be done by 2 ways.

1) using new operator

int a=10; // P2T.

Integer obj = new Integer(a);

create an obj
of integer class

Pass PDT as
argument.

2) using value of ()

Integer obj-i = Integer. value of (a);

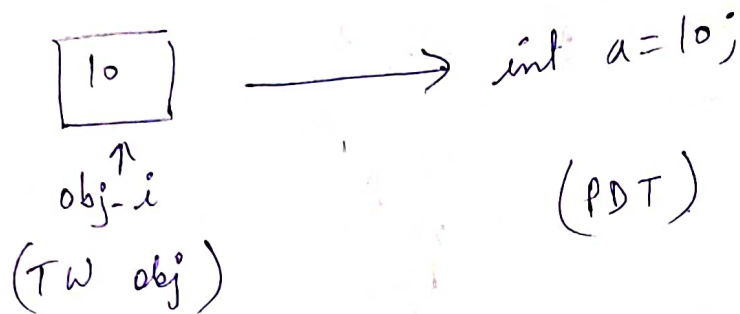
create an obj
of integer class

↓
TW class

convert 'a' to obj
using valueOf.

Q) What is unboxing? How is it done?

- The process of converting TW obj to PDT



- It is done by using `intValue()`

eg `int a = obj-i.intValue();`

Annotations:

- `int a` is labeled "PDT".
- `obj-i` is labeled "TW obj converted to PDT".
- `intValue()` is labeled "using intValue()".

Note: Use suitable suitable TW class & fun name as per the PDT

| PDT | Name of class | Name of fun |
|------|---------------|--------------------------|
| char | Character | <code>charValue()</code> |
| byte | Byte | <code>byteValue()</code> |
| int | Integer | <code>intValue()</code> |
| long | Long | <code>longValue()</code> |
| ; | ; | ; |

Q) What is auto boxing & auto unboxing?

Ans • Boxing & unboxing can be done w/o using any fun.

eg:- To convert int to obj, directly assign value.
int a = 10;

Integer obj-i = new Integer(a);
Integer obj-i = Integer.valueOf(a); } boxing
(by using fun)

Integer obj-i = a; // auto boxing (no need to use any fun)

↖
PDT converted to obj

int a = obj-i.intValue(); // unboxing is done.

int a = obj-i; // Auto-unboxing (no fun is used)

↖
obj converted to PDT

TW obj can be used similar to PDT

1) TW obj can be printed using Soln.

int a = 10;

Integer obj-i = new Integer(a);

Soln(a); // PDT

Soln(obj-i); // TW obj can be printed

Soln(obj-box); // Box is a user defined. It can't be printed directly

X // get() shd be called.

// obj-box.get()

② Tw obj & PDT can be operated w/ each other.

Integer obj-i1, obj-i2;

obj-i1 = 10;

obj-i2 = 20;

obj-i2 = obj-i2 + obj-i1; // add 2 obj

obj-i2 = obj-i2 + obj-i1 + 5;

• When PDT & obj r added then obj is unboxed to PDT, both r added, & then reboxed to obj

unbox
↓
add
↓
rebox

obj-i2++;

unbox
↓
increment
↓
rebox.

③ Two different Tw obj can be operated w/ each other.

Double obj-d = 98.6;

Integer obj-i = 100;

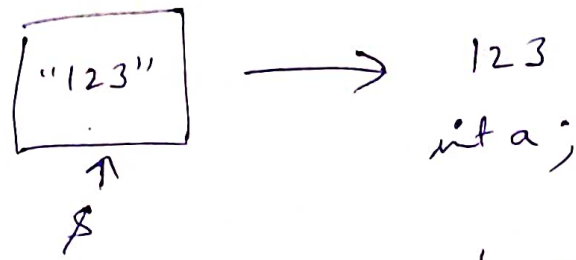
obj-d = obj-d + obj-i;

both r unboxed

↓
added

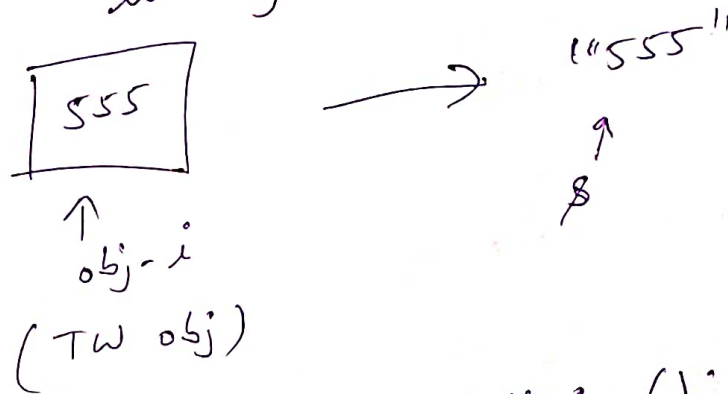
↓
reboxed.

④ To convert a string to int use
`parseInt()`



eg `int i = Integer.parseInt("1234");`

⑤ To convert an obj to string use
`toString()`



eg `String s = obj-i.toString();`