

Clases de envoltorio

Utilización

➤ Encapsulan tipos primitivos como objetos:

```
Integer integ=new Integer(200);  
Double db=new Double (30.4);  
:  
int k=integ.intValue();  
double d=db.doubleValue();
```

➤ Métodos estáticos para convertir String en tipo primitivo u objeto:

```
int p=Integer.parseInt("300");  
int n=Integer.parseInt("100011",2); //35  
Integer num=Integer.valueOf("100011",2);
```

Autoboxing/unboxing

➤ Se puede asignar directamente el tipo primitivo a la variable objeto (autoboxing):

```
Integer ent=200; //autoboxing
```

```
Double db=45.7; //autoboxing
```

➤ Se puede recuperar el tipo primitivo asignando directamente la variable objeto a la variable primitiva (unboxing):

```
int n=ent; //unboxing
```

```
Integer k=30; //autoboxing
```

```
k++; //unboxing más autoboxing
```

Inmutabilidad de objetos

➤ Los objetos de las clases de envoltorio son inmutables, no se pueden modificar:

```
Integer ent=200; //autoboxing
```

```
ent=ent+100; //genera un nuevo objeto, unboxing + autoboxing
```



Igualdad objetos de envoltorio

- Se aplica lo mismo que para cadenas:

Diferentes objetos

```
Integer int1=new Integer(20);  
Integer int2=new Integer(20);  
//falso  
if(int1 ==int2){  
}
```

Mismo objeto

```
Integer int1=20;  
Integer int2=20;  
//verdadero  
if(int1 ==int2){  
}
```

- Se puede utilizar el método equals() para comparar los valores envueltos por el objeto