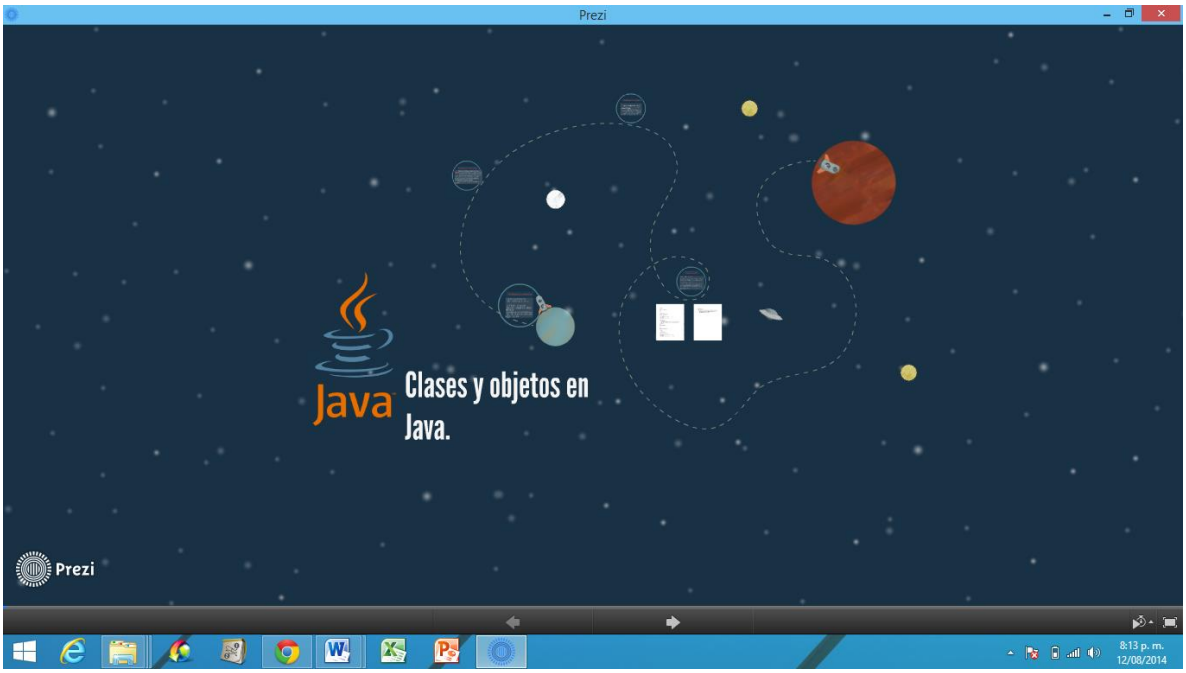
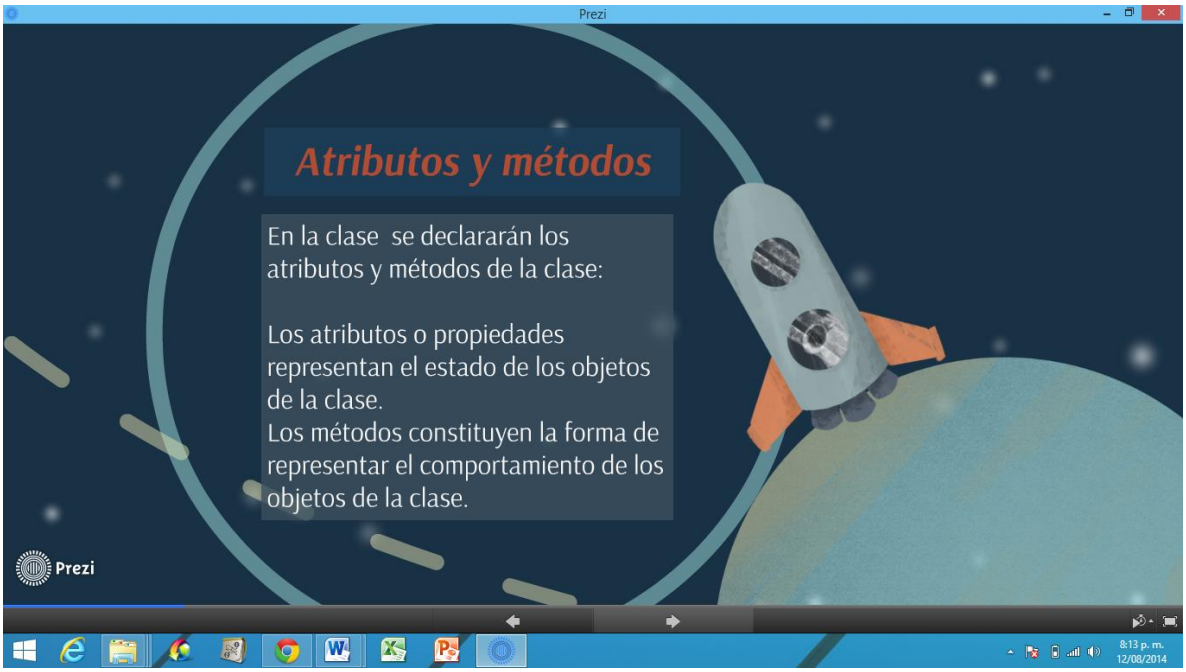
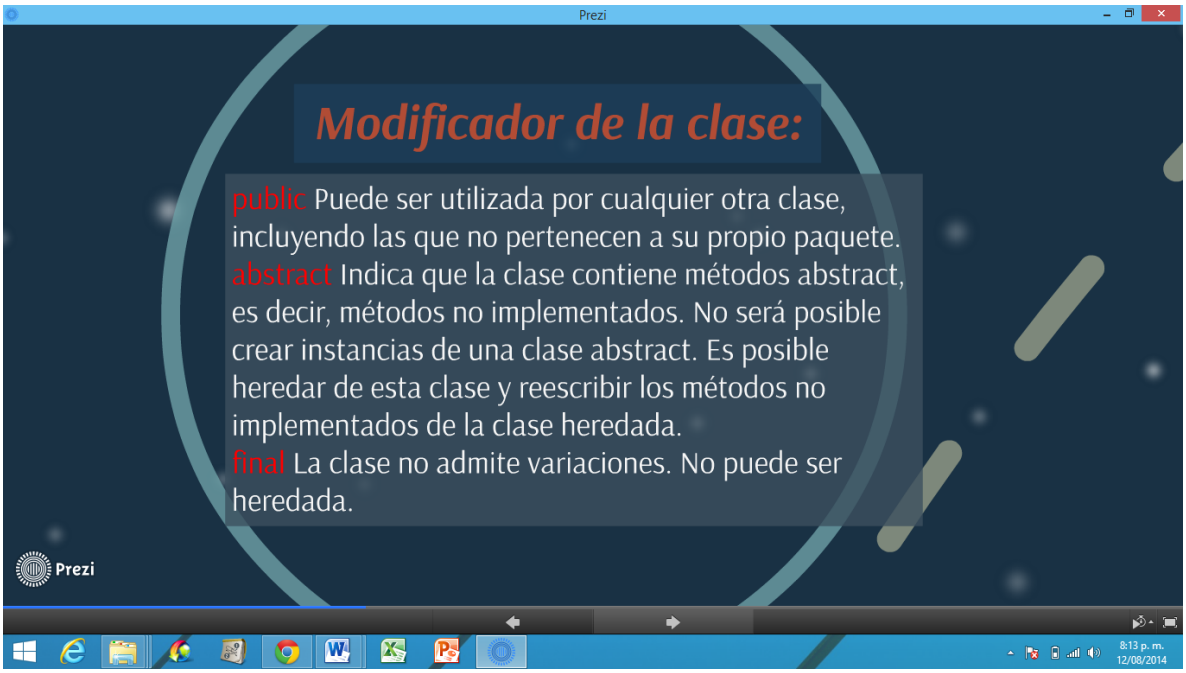
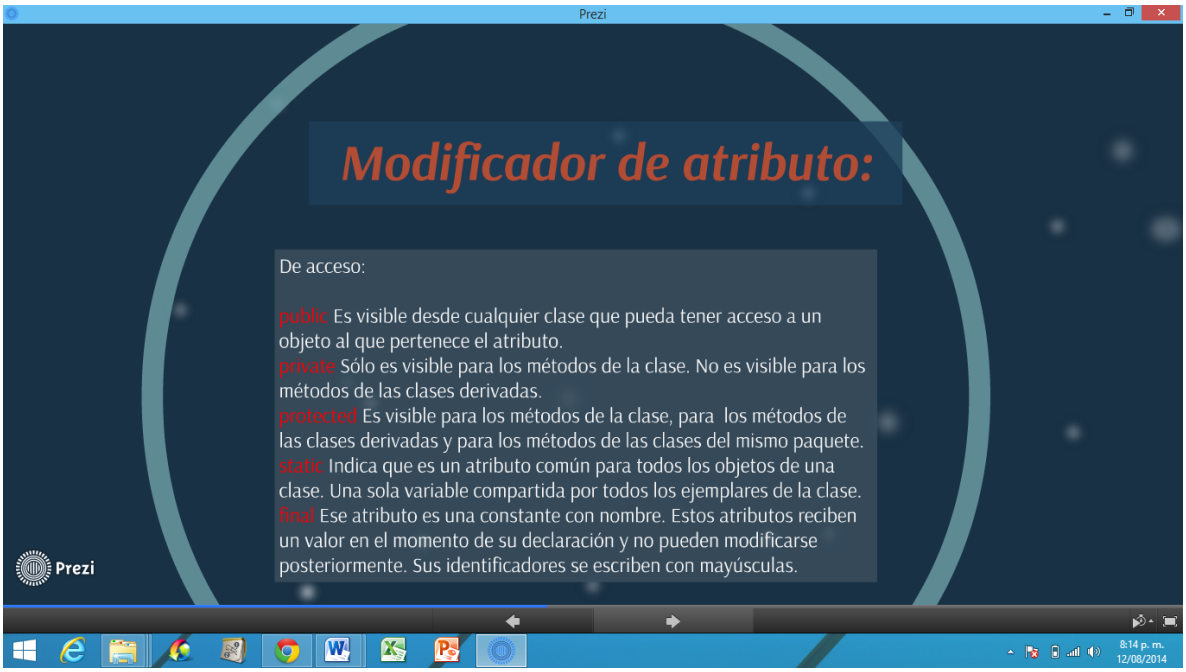
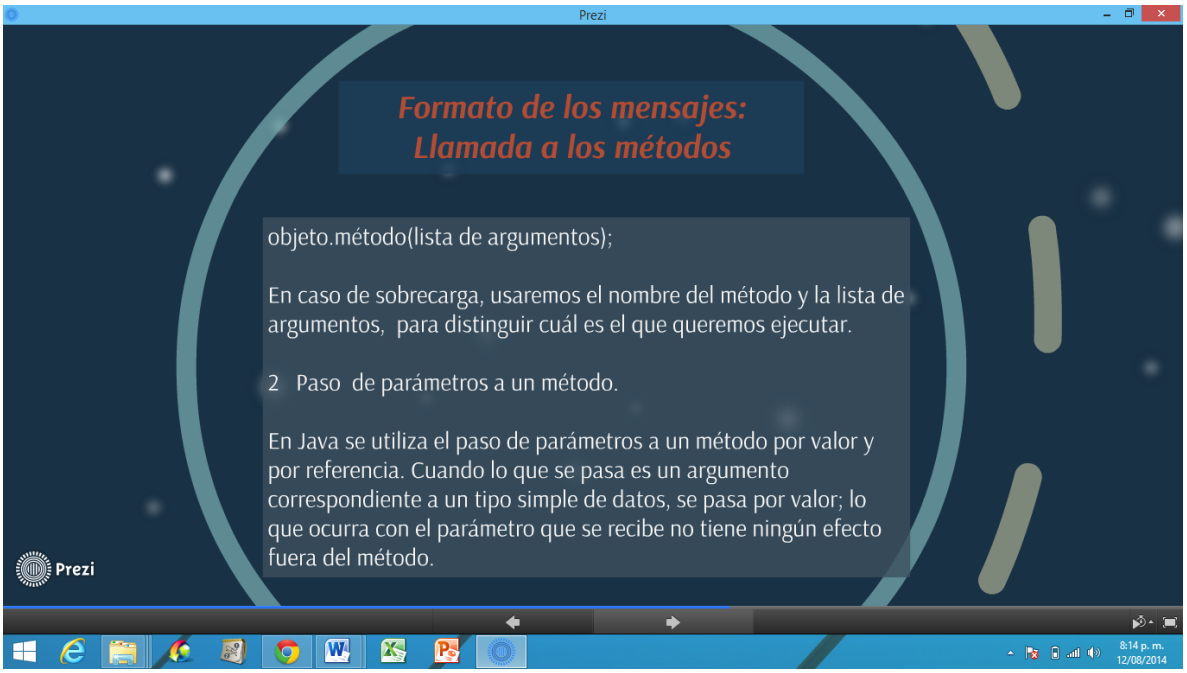
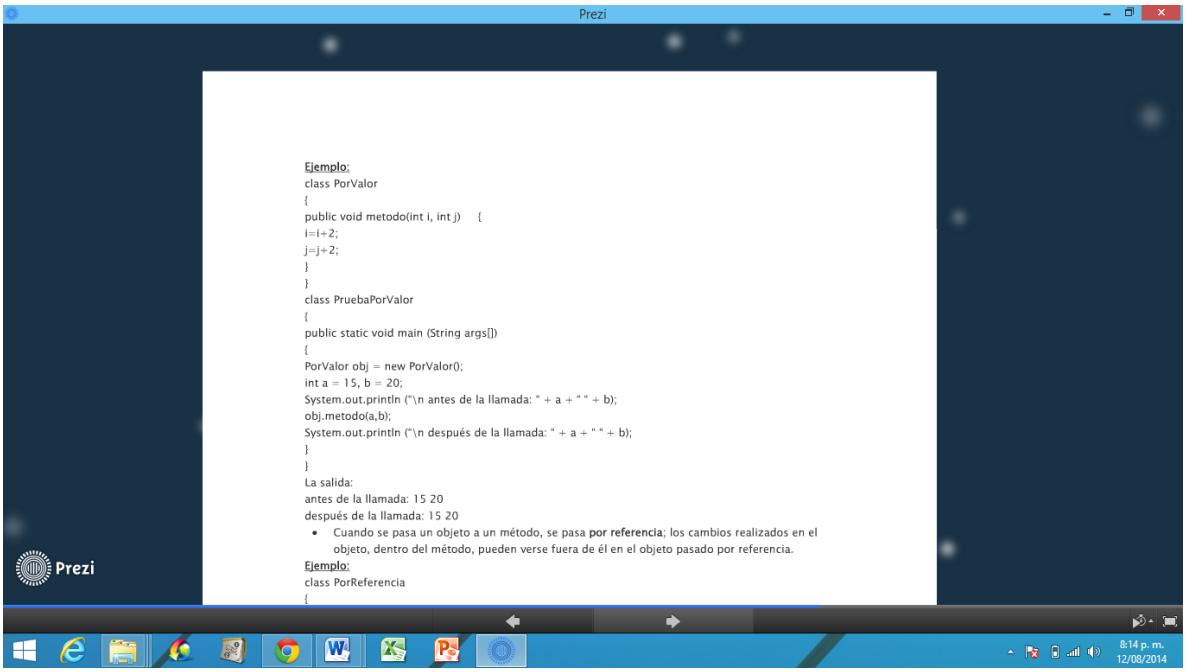

Prezi
Java
Clases y objetos en Java.
Prezi
8:13 p. m.
12/08/2014

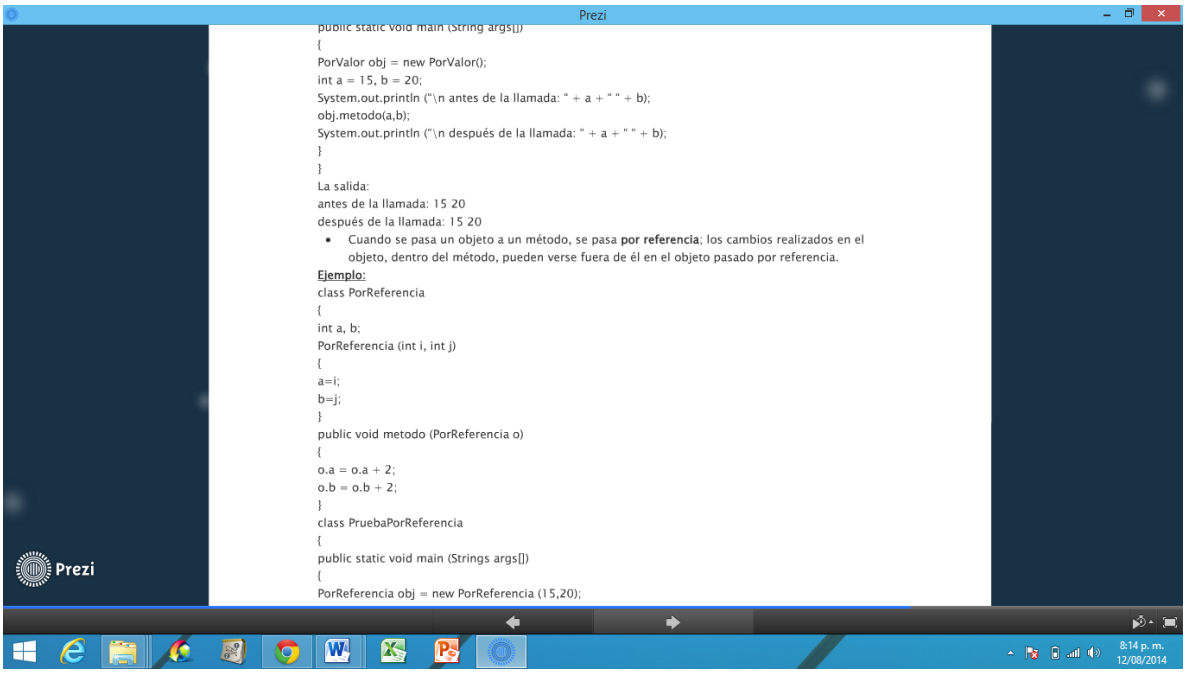

Prezi
Atributos y métodos
En la clase se declararán los atributos y métodos de la clase:
Los atributos o propiedades representan el estado de los objetos de la clase.
Los métodos constituyen la forma de representar el comportamiento de los objetos de la clase.
Prezi
8:13 p. m.
12/08/2014

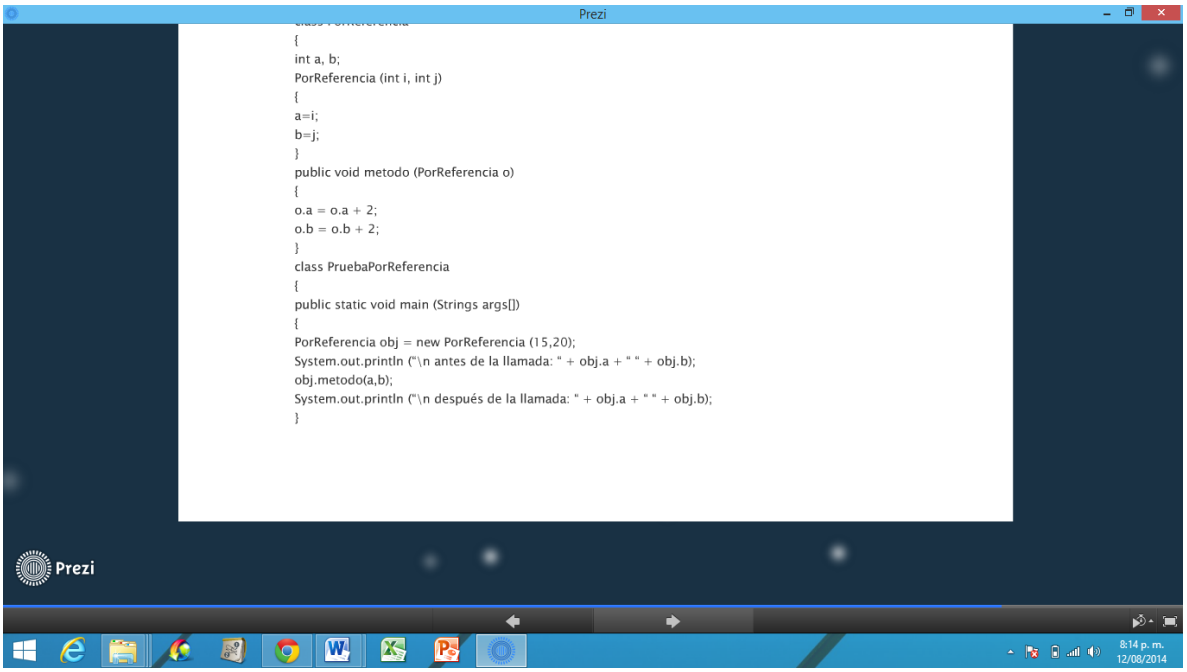

Prezi
Modificador de la clase:
public Puede ser utilizada por cualquier otra clase, incluyendo las que no pertenecen a su propio paquete.
abstract Indica que la clase contiene métodos abstract, es decir, métodos no implementados. No será posible crear instancias de una clase abstract. Es posible heredar de esta clase y reescribir los métodos no implementados de la clase heredada.
final La clase no admite variaciones. No puede ser heredada.
Prezi
8:13 p. m.
12/08/2014

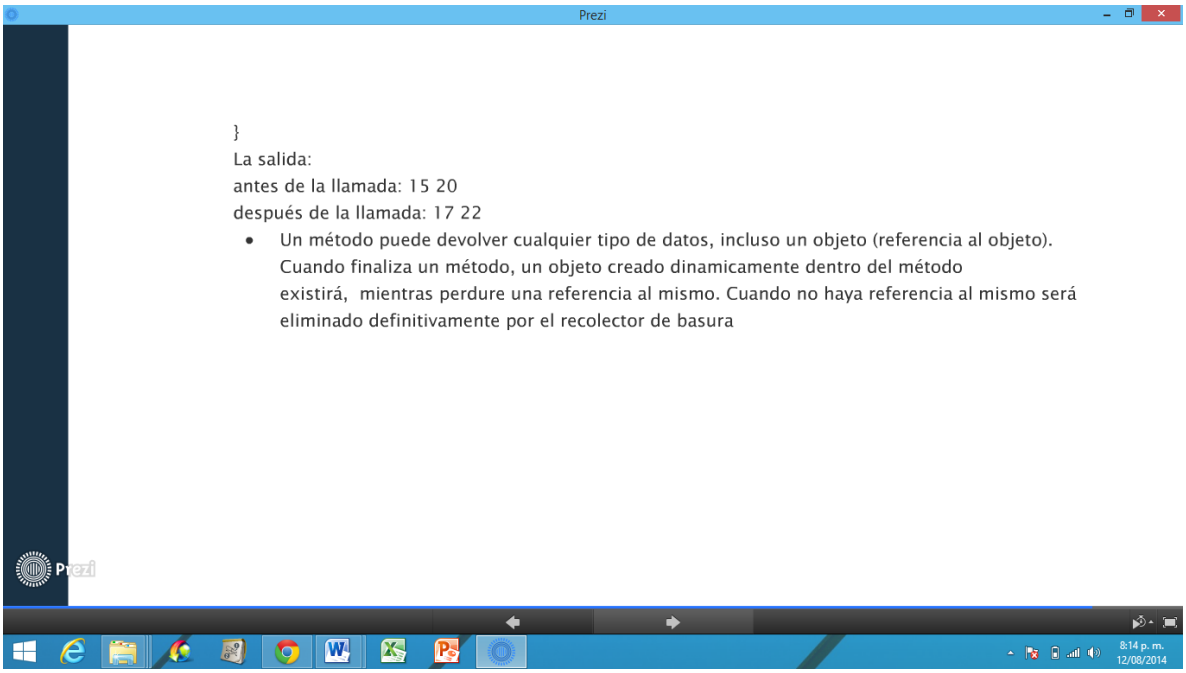

Prezi
Modificador de atributo:
De acceso:
public Es visible desde cualquier clase que pueda tener acceso a un objeto al que pertenece el atributo.
private Sólo es visible para los métodos de la clase. No es visible para los métodos de las clases derivadas.
protected Es visible para los métodos de la clase, para los métodos de las clases derivadas y para los métodos de las clases del mismo paquete.
static Indica que es un atributo común para todos los objetos de una clase. Una sola variable compartida por todos los ejemplares de la clase.
final Ese atributo es una constante con nombre. Estos atributos reciben un valor en el momento de su declaración y no pueden modificarse posteriormente. Sus identificadores se escriben con mayúsculas.
Prezi
8:14 p. m.
12/08/2014


Prezi
Formato de los mensajes:
Llamada a los métodos
objeto.método(lista de argumentos);
En caso de sobrecarga, usaremos el nombre del método y la lista de argumentos, para distinguir cuál es el que queremos ejecutar.
2 Paso de parámetros a un método.
En Java se utiliza el paso de parámetros a un método por valor y por referencia. Cuando lo que se pasa es un argumento correspondiente a un tipo simple de datos, se pasa por valor; lo que ocurra con el parámetro que se recibe no tiene ningún efecto fuera del método.
Prezi
8:14 p. m.
12/08/2014


Prezi
Ejemplo:
class PorValor
{
public void metodo(int i, int j) {
i=i+2;
j=j+2;
}
}
class PruebaPorValor
{
public static void main (String args[])
{
PorValor obj = new PorValor();
int a = 15, b = 20;
System.out.println ("\n antes de la llamada: " + a + " " + b);
obj.metodo(a,b);
System.out.println ("\n después de la llamada: " + a + " " + b);
}
}
La salida:
antes de la llamada: 15 20
después de la llamada: 15 20
Cuando se pasa un objeto a un método, se pasa por referencia; los cambios realizados en el objeto, dentro del método, pueden verse fuera de él en el objeto pasado por referencia.
Ejemplo:
class PorReferencia
{
Prezi
8:14 p. m.
12/08/2014


Prezi
public static void main (String args[])
{
PorValor obj = new PorValor();
int a = 15, b = 20;
System.out.println ("\n antes de la llamada: " + a + " " + b);
obj.metodo(a,b);
System.out.println ("\n después de la llamada: " + a + " " + b);
}
}
La salida:
antes de la llamada: 15 20
después de la llamada: 15 20
Cuando se pasa un objeto a un método, se pasa por referencia; los cambios realizados en el objeto, dentro del método, pueden verse fuera de él en el objeto pasado por referencia.
Ejemplo:
class PorReferencia
{
int a, b;
PorReferencia (int i, int j)
{
a=i;
b=j;
}
public void metodo (PorReferencia o)
{
o.a = o.a + 2;
o.b = o.b + 2;
}
class PruebaPorReferencia
{
public static void main (Strings args[])
{
PorReferencia obj = new PorReferencia (15,20);
8:14 p. m.
12/08/2014


Prezi
{
int a, b;
PorReferencia (int i, int j)
{
a=i;
b=j;
}
public void metodo (PorReferencia o)
{
o.a = o.a + 2;
o.b = o.b + 2;
}
class PruebaPorReferencia
{
public static void main (Strings args[])
{
PorReferencia obj = new PorReferencia (15,20);
System.out.println ("\n antes de la llamada: " + obj.a + " " + obj.b);
obj.metodo(a,b);
System.out.println ("\n después de la llamada: " + obj.a + " " + obj.b);
}
8:14 p. m.
12/08/2014


Prezi
}
La salida:
antes de la llamada: 15 20
después de la llamada: 17 22
• Un método puede devolver cualquier tipo de datos, incluso un objeto (referencia al objeto). Cuando finaliza un método, un objeto creado dinamicamente dentro del método existirá, mientras perdure una referencia al mismo. Cuando no haya referencia al mismo será eliminado definitivamente por el recolector de basura
Prezi
8:14 p. m.
12/08/2014