What is a variable of class type?

Suppose we have this class:

```
class Ex {
  public int field;
}
```

and now, in main( ):

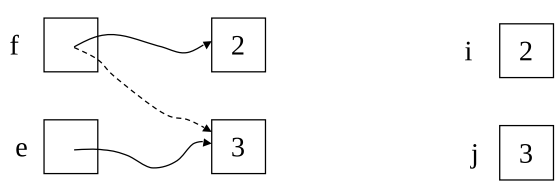
```
Ex e = new Ex();
Ex f = new Ex();
e.field = 2;
f.field = 3;
System.out.println(e.field);
e = f;
System.out.println(e.field);
f.field = 4;
System.out.println(e.field);
```

Compare:

```
int i;
int j;
i = 2;
j = 3;
System.out.println(i);
i = j;
System.out.println(i);
j = 4;
System.out.println(i);
```

Why?

So, what is a variable of class type?



i = j copies the ***value*** of j.

e = f copies the value of f too, but the value is a reference.

Often you can ignore the difference between an object and a reference to an object. But …

Method parameters

What does this print?

```
private static void incr (int i) {
  System.out.println(i);
  i += 1;
  System.out.println(i);
}
public static void main (String[] args) {
  int i = 3;
  incr(i);
  System.out.println(i);
}
```

The value of i in main is not changed!

The **value** of the parameter used in a method is a ***copy*** of the **value** of the corresponding argument in the method call.

This is always true, both for parameters of a primitive type and for parameters of a class type. But it means different things for the two kinds of types.

Parameters of class type

What does this print?

```
private static void incrEx (Ex e) {
  System.out.println(e.field);
  e.field += 1;
  System.out.println(e.field);
}
public static void main (String[] args) {
  Ex e = new Ex();
  e.field = 3;
  incrEx(e);
  System.out.println(e.field);
}
```

incrEx ***does*** change e! Why? (How could it?)

Changing a class-type parameter

What about this?

```
private static void makeEx (Ex e) {
  e = new Ex();
  e.field = 4;
}
public static void main (String[] args) {
  Ex e = new Ex();
  makeEx(e);
  System.out.println(e.field);
}
```