How we draw an object

Below is a definition of classes C and S. S is a subclass of C and C is a superclass of S.

To the right of the class definitions we draw an object of class S. It has three *partitions*: one for subclass S, one above it for class C, and one above that for class Object, the *superest* class of them all. Class Object is the superclass of any class that does not explicitly extend a class. At the top-right of a partition, we draw a box with the name of the class.

To avoid clutter, since we know partition Object is always at the top and we know something about what it contains, we often do not draw it.

In the partition for any class (like C), we draw the fields and write the signatures of the methods that are declared in that class. Thus, the partition for C contains field b, the constructor, and function getB. The signature for a method is just an abbreviation for the whole method, including its body. We work with the concept that the whole instance method (like getB) resides in the object.

The tab at the top of the object contains the name of the object. It consists of the class of the object (S), the @ sign, and a hexademical number giving the address in memory where the object resides. When *we* create or draw an object, we can put any number we want there, making sure that different ojects have different addresses.

When the following assignment is executed, first the new-expression is evaluated, creating the object shown and yielding as value of the new-expression the name S@6dfe, then that value is stored in s. Thus, s contains a pointer to the new object, not the new object itself.

```
S s= new S(1);
```

```
public class C{
   private int b;

   public C(int x) {
      b= x;
   }

   public int getB() {
      return b;
   }
}

public class S extends C{
   private int f= 2;

   public S(int y) {
      super(y);
   }

   public void setF(int x) {
      f= x;
   }
}
```