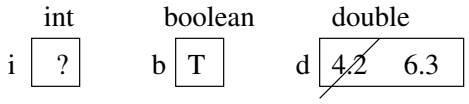
1 Memory

There are three kinds of memory in Java: primitives, references, and arays. They each have a different layout in memory.

1.1 Primitives

int, double, byte, char, and boolean are all primitive types. Their memory is draw as a box with a name, type, and data. If uninitialized the data is "0" or a "?". When the data is changed, we re-use the same box of memory.

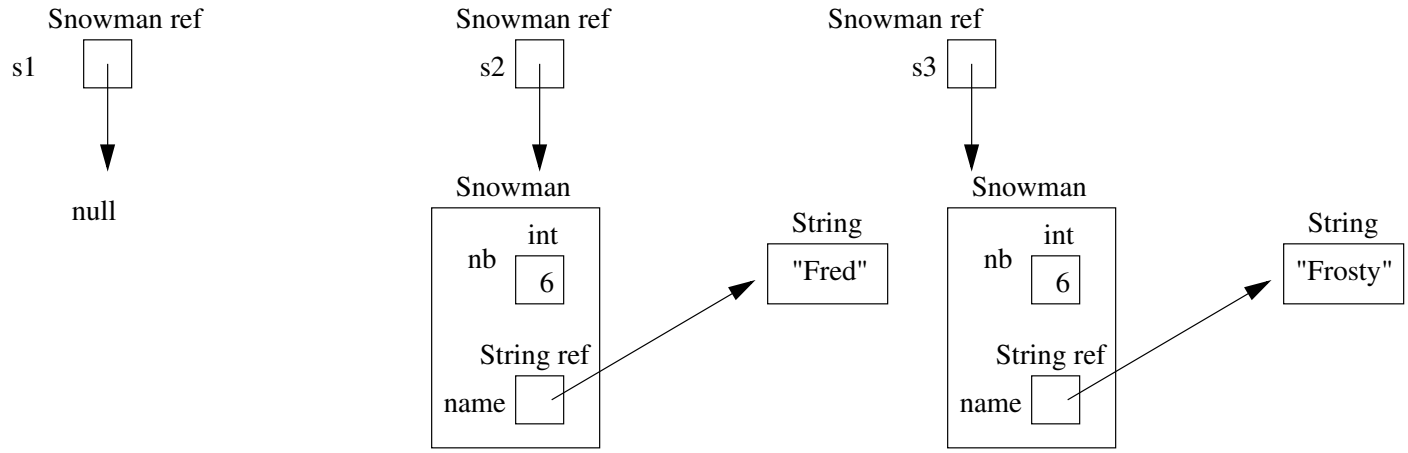
```
int i;
boolean b = true;
double d = 4.2;
d = 6.3;
```



1.2 References

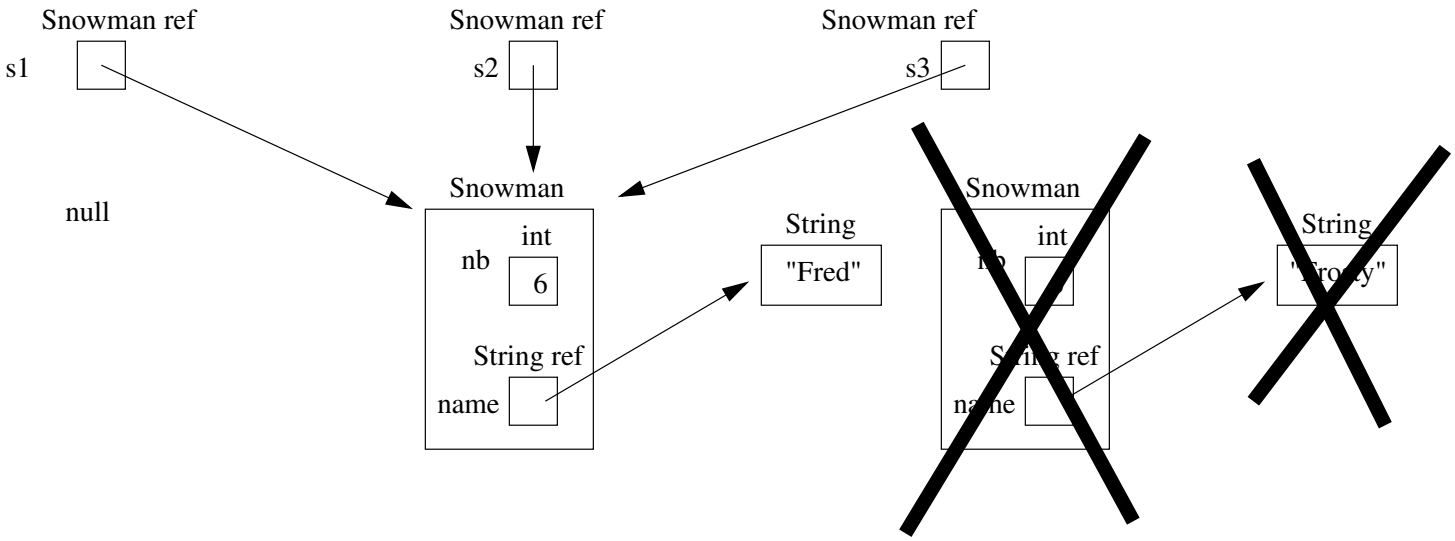
Classes in java are reference types. Strings are also classes even though they can be set differently. Their memory is drawn as a box with a name and a type. The data in the box is always a reference (arrow, pointer, arm) which points somewhere else. When these are uninitialized, they point to "null". When they are initialized, they point to an actual object of their type which contains boxes for all member data. (Be careful some member data may be reference types).

```
Snowman s1;  // The snowman class has an int and a String as member data
Snowman s2 = new Snowman("Fred");
Snowman s3 = new Snowman("Frosty");
```



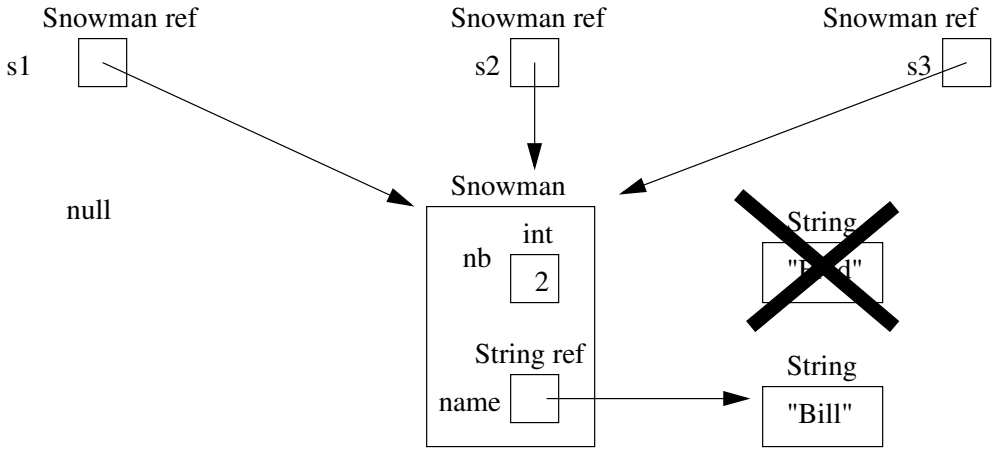
When the data in a reference changes, the reference will point to a new place. Any class objects with no pointers to them will be collected by the garbage collector and I've drawn an X through them.

```
s1 = s2;
s3 = s2;
```



The same thing happens if the data in a class instance is changed.

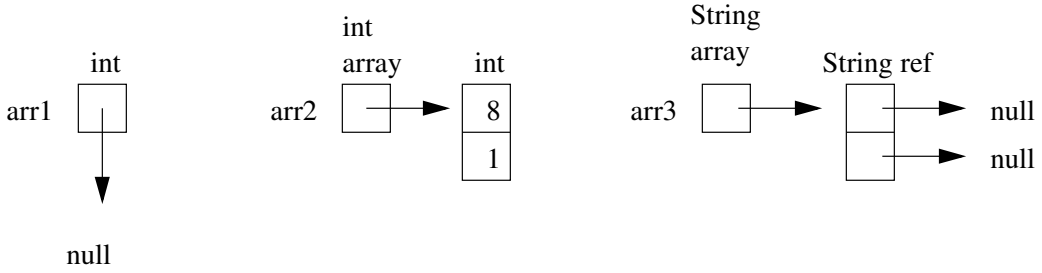
```
s1.setButtons(2);
s2.setName("Bill");
```



1.3 Arrays

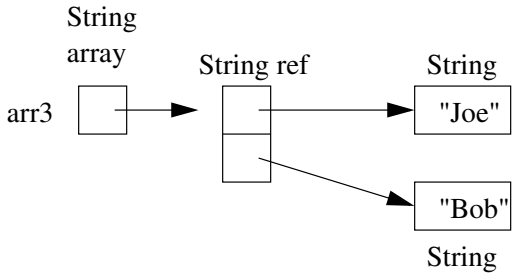
Arrays are a special case. They are a reference or a primitive depending on what they contain. Think of them as a bookshelf with either primitive data or references on each sheld. The shelves are named 0..n when the array has n+1 items. Really, arrays are a reference and the bookshelf doesnt appear until the array constructor is called.

```
int arr1[];               // A null array
int arr2[] = new int[2]; // Calling the array constructor
arr2[0] = 8;
arr2[1] = 1;
String arr3[] = new String[2]; // Not calling the String constructor
```



You have to actually call the object constructor for an array of objects to instantiate an object.

```
arr3[0] = new String("Joe"); // calling the String constructor
arr3[1] = "Bob";             // Also calling the String constructor
```



Note: For any array that has already called the array constructor, I frequently take a short cut and don't show the reference for the array, just the bookshelf.