



Object References

James Brucker

Variables

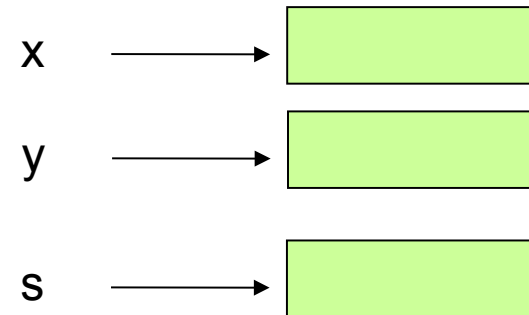
A variable is a name we use to *refer* to a memory location.

What is **stored in the memory location**?

```
/* define two variables */  
int x;  
int y;  
String s;
```

Program:

Memory:



We will see that the answer is *different* for variables of primitive data types and variables of *object* data types.

This is important -- know it!

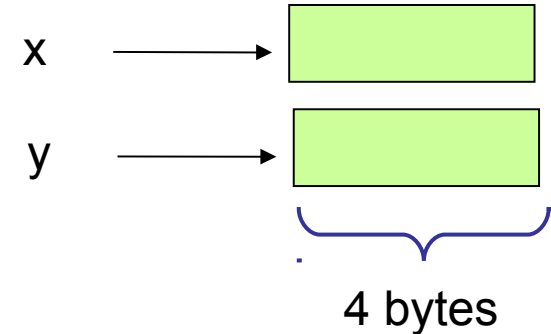
Primitive Data Types are stored as **values**

For **primitive data types**, the variable's memory location holds its **value**.
Data types for which this is true are called **value data types**.

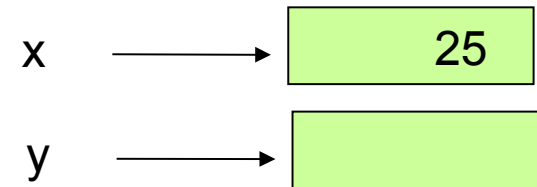
```
/* define two "int"
   variables */
int x;
int y;
```

Program:

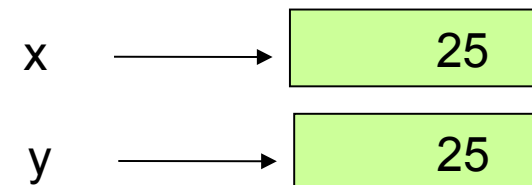
Memory:



```
/* assign value to x */
x = 25;
```



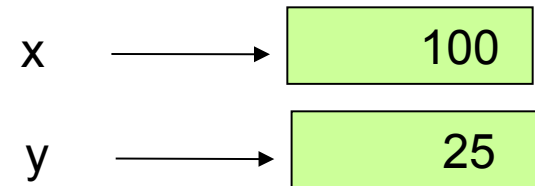
```
/* assign value to y */
y = x;
```



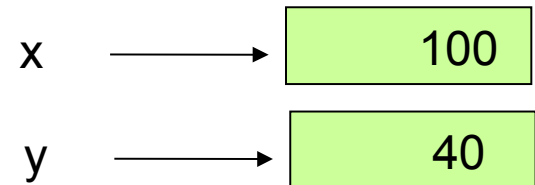
Values are copied on assignment

Changing the value of x does not affect the value of y.

```
x = 100;
```



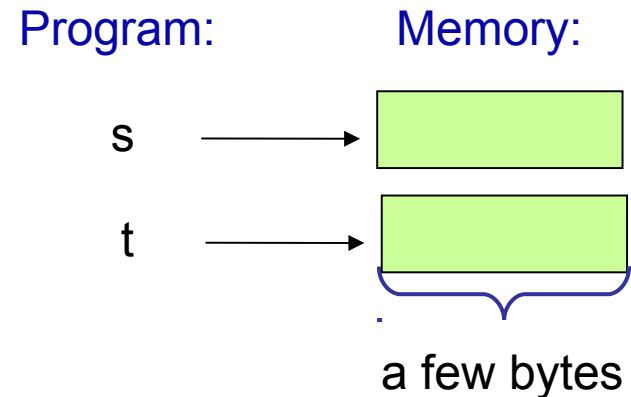
```
/* assign value to y */  
y = 40;
```



Variables for Object Data Types

For object data types, a variable is a **reference** to the object, but does not store the object !!

```
/* define two String  
   variables */  
String s;  
String t;
```



Variables **refer** to the Object's location

To *create* an object you must use the "**new**" keyword.

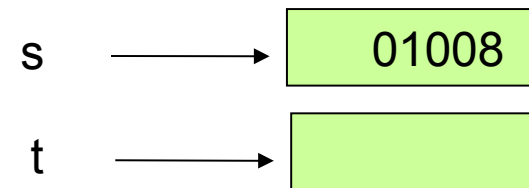
This allocates new storage in a memory region called **the heap**.

```
String s;  
/* create an object */  
s = new String("Help,  
I'm trapped in a  
Computer.");
```

The **new** command creates a new object and returns a *reference* to its memory location

The object also contains other information, such as:

- length of string
- the Class it belongs to



Address: Memory:

01000	48AC00FB
01008	Help, I'
01010	m trappe
01018	d in a
01020	Computer
01028	.00000000
01030	00000000
01038	

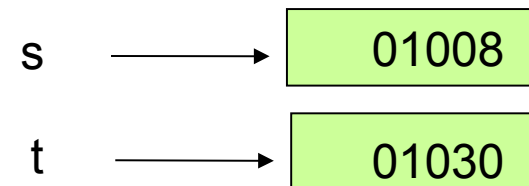
Variables refer to an Object's location

Each **new** object gets its own storage space on the heap.

The size of the object can be anything.

```
/* create an object */  
  
t = new String("Hello");
```

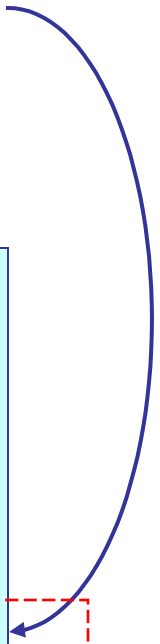
The **new** command finds some more free "heap" space large enough for the String "Hello". It creates a String object and returns a *reference* to it.



Address: Memory:

01000	48AC00FB
01008	Help, I'
01010	m trappe
01018	d in a
01020	Computer
01028	.0000000
01030	Hello000
01038	

new String



Variables for Object Data Types (4)

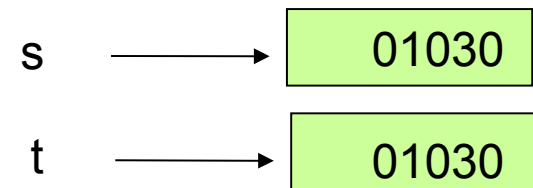
When you *assign a value* to a reference variable, you are assigning the address of the object -- not the object's value!

```
// copy object reference  
s = t;
```

Now `s` refers to the same object as `t` ("Hello").

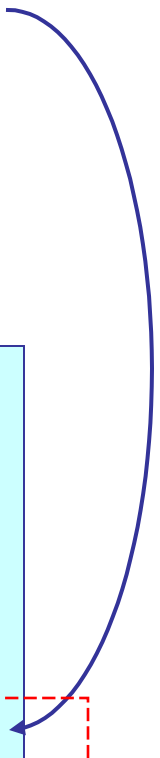
The old String object has no reference ... it is *garbage*.

Eventually Java will reclaim the storage space for re-use.



Address: Memory:

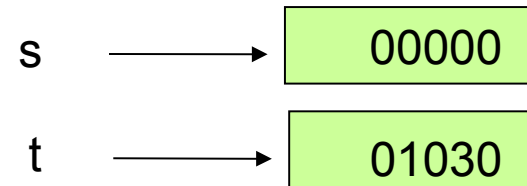
01000	48AC00FB
01008	Help, I'
01010	m trappe
01018	d in a
01020	Computer
01028	.00000000
01030	Hello000
01038	



The null value

If an object variable (object reference) doesn't refer to any object, it is assigned a value of null. You can use this to "clear" a reference.

```
/* discard old value */  
s = null;
```



Now **s** does not refer to anything.

But, the old **value** may still be in memory (in the heap).

Address: Memory:

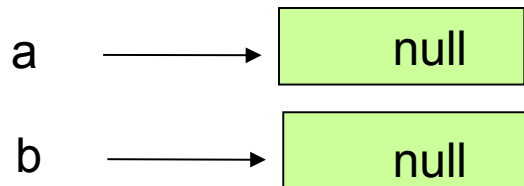
01000	48AC00FB
01008	Help, I'
01010	m trappe
01018	d in a
01020	Computer
01028	.0000000
01030	Hello000
01038	

Another Example: BankAccount (1)

```
BankAccount a;  
BankAccount b;
```

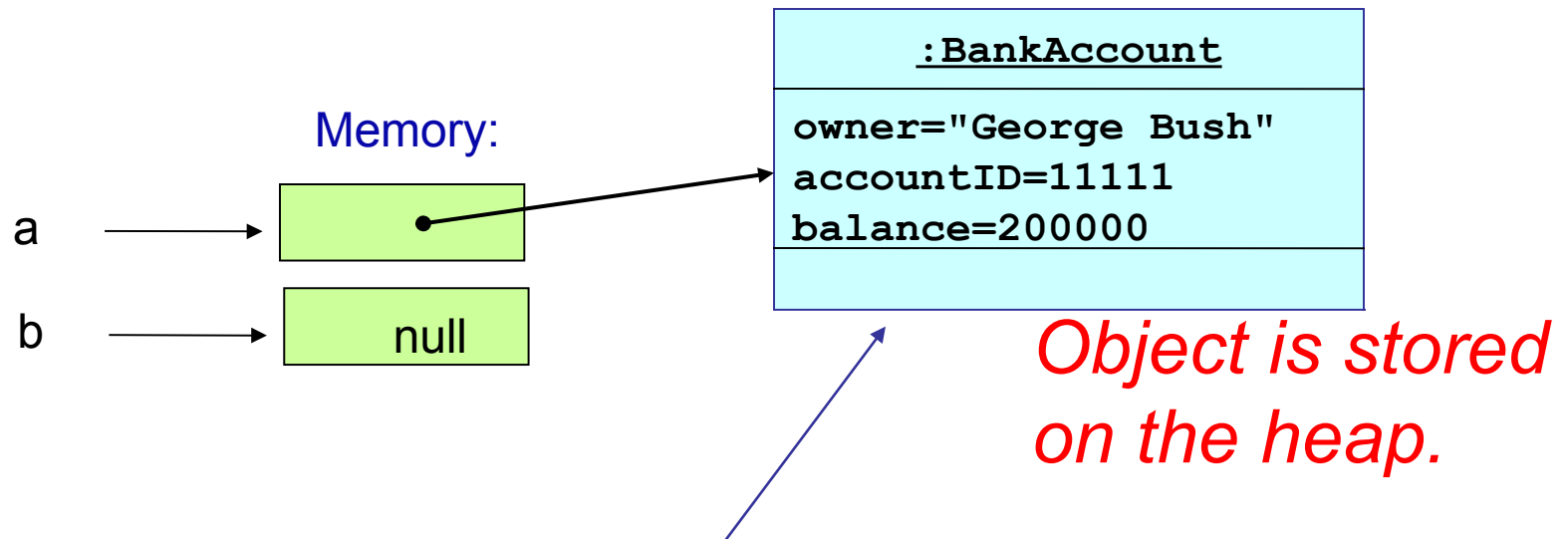
This creates BankAccount *references*, but doesn't create any BankAccount *objects*.

Memory:



BankAccount (2)

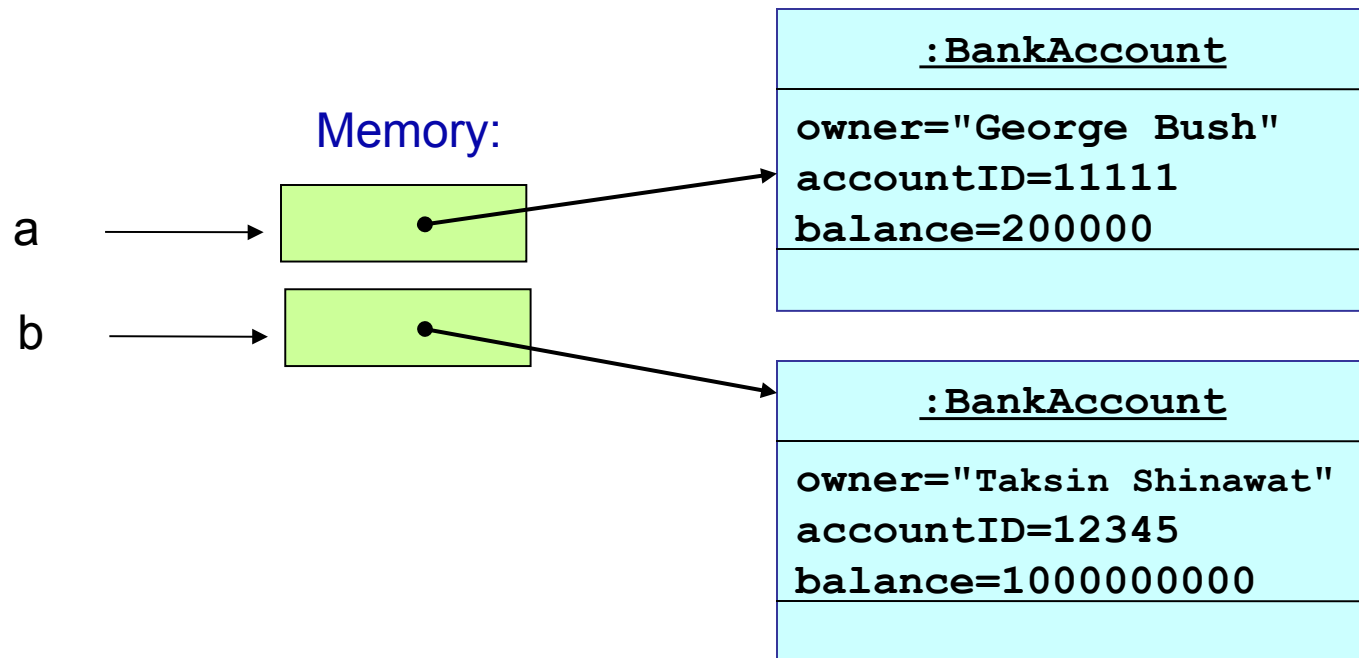
```
a = new BankAccount( "George Bush", 11111);  
a.deposit(200000);
```



UML **Object Diagram** notation, show the values of one object..

Create another BankAccount

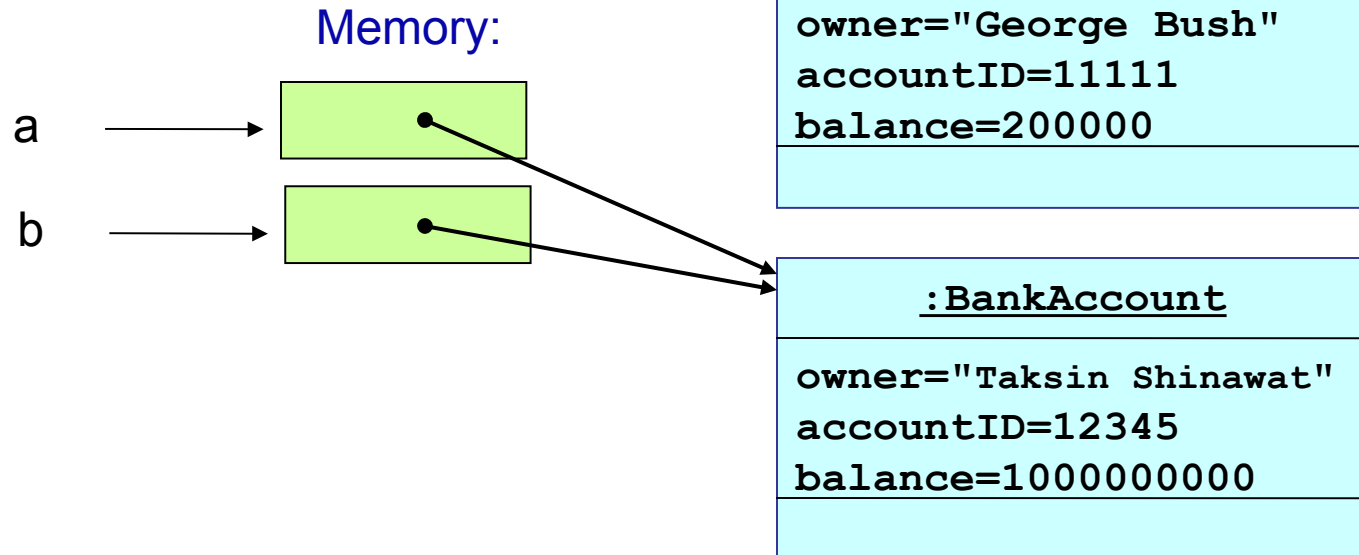
```
b = new BankAccount( "Taksin Shinawat", 12345);  
b.deposit(10000000000);
```



assign a: copy or reference?

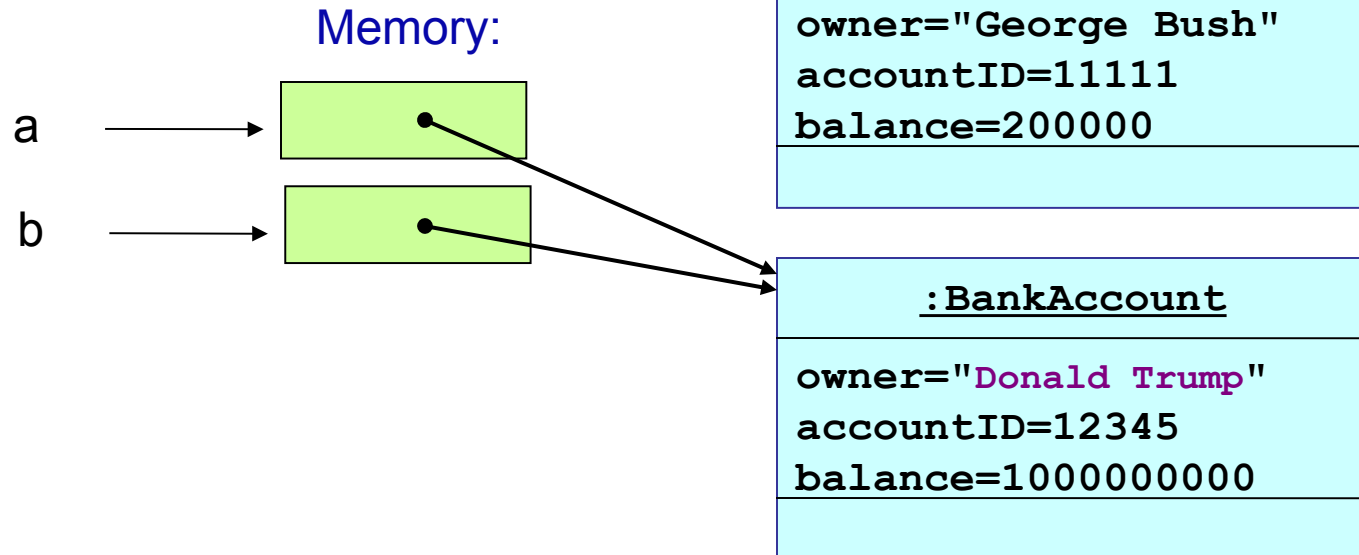
```
// copy Taksin's data into the other object?  
a = b;
```

No copy! It makes **a** "point to" the same object as **b**.



Who's Got the Money?

```
// copy Taksin's data into the other object?  
a = b;  
a.setOwner("Donald Trump");
```



Exercises
