**Object Creation:**

class A  
{  
 int x;  
 public void show()

{

}

}

In C++:   
A obj;  
  
In Java:   
A obj = new A();  
Will create this object inside **Heap memory***obj = reference; A() = constructor*

obj(10231)

Heap memory

x, show()

**Object Storage:**

There are 2 types of variables:

* Primitive variables (value is directly stored in stack)
* Reference variables (value is stored in heap, **hashcode** is stored in stack)

Int I = 5;  
A obj = new A();

Hashcode

*Stack*

|  |  |
| --- | --- |
| I | 5 |
| Obj | 10231 |

**Object Deletion:**

* Garbage Collection does it (If no link is present to an object. It is eligible for garbage collection.

(Unlike C++, don’t have to delete object manually)

* obj = null; // Will delete object created for obj in this case

**Anonymous Object:**

If you are using the object only once (like if you are calling show method only once), it’s pointless to create object for it and unnecessarily wasting stack space.

Analogy: Pointless to create a primitive variable for a value that you are using only once.

A a = new A(); // will occupy space in Stack **(Referenced Object)**B obj = a.show();

**B obj = new A().show();**  // will not occupy space in Stack **(Anonymous Object)**

Advantages of using anonymous object:

* No space is occupied in stack memory
* Referenced object won’t be cleaned by garbage collection after usage, while anonymous object will be removed once it’s used.

**If you want to use an object only once, always use Anonymous object.**

**Anonymous object can’t be used when you want to use object more than once is because every time it will call new object.**