Class Customer

java.lang.Object
  Customer

```
public class Customer
extends java.lang.Object
```

Defines a simple Customer class that holds some information regarding customers, this information includes: nickname and id number. When a Customer object is created the nickname and the id number can be retrievd by **getName()** and **getId()** methods. For example:

Customer cos = new Customer("blue_jet", 11135422);
cos.**getName()**; returns **"blue_jet"**
cos.**getId()**; returns **11135422**

When a Customer object is created, **the nickname and the id number can not be changed.**

Two Customer objects can be compared by using the **equals()** methods, for example:

Customer cos1 = new Customer("blue_jet", 11135422);
Customer cos2 = new Customer("blue_jet", 222356478);

If **either** of nicknames or ids are equal, or **both** are equal then the result is **true**. The result of comparison is **false** only and only when both values in both objects are different. So, the result of:

cos1.**equals**(cos2) or cos2.**equals**(cos1)

is **true**. But, the result of:

cos2 = new Customer("aaaazzz", 222356478);
cos1.equals(cos2) or cos2.equals(cos1)

is **false** .

To **print** the Customer object, you can costumize the result by getting the nickname and id number separately and print it in your own way, or you can print the object directly. Here are some examples of printing the Customer object:

Customer cost01 = new Customer("Meijad", 22204563);
System.out.println(cost01);
System.out.println("--------");
System.out.printf("{name: %s, id: %d}%n", cost01.getName(), cost02.getId());

The results are as following:

"[Meijad, 22204563]"
"--------"
"{name: Meijad, id: 22204563}"

To create a deep (actual) **copy** of the Customer object the method **clone()** can be used. The general intent is that, for any Customer object c, the expression:
c.clone() != c;
showing they are different objects with different references in the memory, but the result of comparing values as below:
x.clone().equals(x)
is **true** .

Author:
amjadm@miamioh.edu

Constructor Summary

Constructors

Constructor and Description
`Customer(Customer co)` Copy Constructor, this constructor initilizes the object with the given Customer object's information.
`Customer(java.lang.String nickname, long id)` This constructor initilizes the object with the given nickname and id number.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
`Customer`	`clone()` Creates and returns a deep copy of this Customer object.
`boolean`	`equals(java.lang.Object obj)` Compares this Customer object to the given Customer object.
`long`	`getId()` Returns a long type value which is the id number of this object.
`java.lang.String`	`getName()` Returns a String value which is the nickname of this object.
`java.lang.String`	`toString()` Returns this Customer object in the String fromat: "[nikname, id]"

Methods inherited from class java.lang.Object

`finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

Customer

```
public Customer(java.lang.String nickname,
                long id)
```

This constructor initilizes the object with the given nickname and id number.

Parameters:
`nickname – a String type name.`
`id – a long type number as id.`

Customer

```
public Customer(Customer co)
```

Copy Constructor, this constructor initilizes the object with the given Customer object's information.

Parameters:
`co – a Customer object to initialize the object with.`

Method Detail

getName

```
public java.lang.String getName()
```

Returns a String value which is the nickname of this object.

Returns:
`the nickname of this object.`

getId

```
public long getId()
```

Returns a long type value which is the id number of this object.

Returns:
`the id number of this object.`

toString

```
public java.lang.String toString()
```

Returns this Customer object in the String fromat: "[nikname, id]"

Overrides:
`toString in class java.lang.Object`

Returns:
`A String value from the nickname and the id of this object.`

equals

```
public boolean equals(java.lang.Object obj)
```

Compares this Customer object to the given Customer object.

If either names or ids are equal the result is true.
If both names and ids are equal the result is true.
The result is false when both nicknames and ids are different.

Overrides:
`equals in class java.lang.Object`

Parameters:
`obj – Customer object`

Returns:
`A boolean value, the result of comparison.`

clone

```
public Customer clone()
```

Creates and returns a deep copy of this Customer object. The general intent is that, for any Customer x, the expression:
x.clone() != x;
but the result of comparing values as below:
x.clone().equals(x)
is **true** .

Overrides:
`clone in class java.lang.Object`

Returns:
`The deep copy of the Customer object.`