Summary of toString(), clone(), equals()

<table>
<tr><th>Class</th><th>toString()
(Similar to C++: operator<<)</th><th>clone()
(Similar to C++: operataor=</th></tr>
<tr><td>Purpose</td><td>

To make it easier for the users of your class to display the content of an object.
If you don't provide toString(), they would need to use "get()" to get and display the values of all the attributes.

This could be a big challenge if the class

contains a lot of attributes, or
contains member class(es), or
inherited from another class

</td><td>

To make it easier for the users of your class t
an object.
If you don't provide clone(), they would need
set()" to copy values of all the attributes.

This could be a big challenge if the cla

contains a lot of attributes, or
contains member class(es), or
inherited from another class

Assignment in Java is "call by reference". Th

<pre>class B implements Clonable {
   private int b;
   public Object clone() { ...}
}
class A {
   private int a;
   private B bObj;
   public A(int a1, B bObj1)
   {
      a=a1;
      // Bad idea: bObj shares the
      //   memory as bObj1
      bObj = bObj1;

      // Good idea: bObj and bObj1
      bObj = bObj1.clone();
   }
}</pre>

References

Clone Object and Assignment

</td></tr>
<tr><td>
<pre>---------------
| A |
| int a; |
---------------</pre>

Circle .
Cuboid
Time
Incorrect implementation of equals()

</td><td>
<pre>public String toString() {
   // Approach I
   return ("a = " + a + "\n");

   // Approach II
   // StringBuffer s = new StringBuffer();
   // s.append("a = " + a + "\n");
   // return s.toString();
}</pre>
<pre># Python sample code
def __str__(self):
    return ("a = " + self.__a + "\n");</pre>
</td><td>
<pre>public Object clone() {
   try
   { return super.clone();
   }
   catch (CloneNotSupportedException e)
   {
      // return null;
      throw new InternalError(e.getMessag
   }
}</pre>
<pre># Python sample code</pre>
</td></tr>
<tr><td>
<pre>----------------------
| Person |
| String name; |
----------------------</pre>
Sample code
</td><td>
<pre>public String toString() {
   return ("Name = " + name + "\n");
}</pre>
<pre># Python sample code
def __str__(self):
   return ("Name = " + self.__name + "\n");</pre>
</td><td>
<pre>public Object clone() {
   try
   {
      return super.clone();
   }
   catch (CloneNotSupportedException e)
   {
      // return null;
      throw new InternalError(e.getMessag
   }
}</pre>
<pre># Python sample code</pre>
</td></tr>
<tr><td>
<pre>Fixed array

| A |
| int a[]; |
-----------------</pre>
Sample code

Person

</td><td>
<pre>public String toString() {
   StringBuffer s = new StringBuffer();
   for (int i=0; i < a.length; i++)
       s.append("a [" + i + "] = " + a[i] + "\n");
   return s.toString();
}</pre>
<pre># Python sample code
def __str__(self):
   result=""
   for i in range(len(a))
       result += "a [" + i + "] = " + a[i] + "\n"
   return (result)</pre>
</td><td>
<pre>public Object clone() {
   try
   { return super.clone();
   }
   catch (CloneNotSupportedException e)
   {
      // return null;
      throw new InternalError(e.getMessag
   }
}</pre>
<pre># Python sample code</pre>
</td></tr>
</table>

```
Dynamical array
----------------
| A             |
|   Vector a;   |
----------------
```

Sample code

```
public int getA(int i) {
   return ((Integer)a.elementAt(i)).intValue();
}
public int getNumOfA() { return a.size(); }
public String toString() {
   StringBuffer s = new StringBuffer();
   for (int i=0; i < getNumOfA(); i++)
      s.append("a " + i  + " = " + getA(i) + "\n");
   return s.toString();
}
```

Python sample code

```
public Object clone() {
   try
   { A aobj = (A)super.clone();
     aobj.a = (Vector)a.clone();
     // for (int i=0; i < a.size(); i++)
     //    aobj.a.setElementAt(
     //       ((int)a.elementAt(i)).clor
     // }
     return aobj;
   }
   catch (CloneNotSupportedException e)
   {
      // return null;
      throw new InternalError(e.getMessag
   }
}
```

Python sample code

```
-------------------------
| B                     |
|    int b;             |
|  -------------        |
|  | A         |        |
|  |   int a   |        |
|  -------------        |
-------------------------
class A {
   private int a;
}
class B {
   private int b;
   A aObj;
}
```

- Sample code
- Clone Object and Assignment

```
public String toString() {
   // return ("b = " + b + "\n" + aObj.toString());
   return ("b = " + b + "\n" + aObj);

   // Approach II
   // StringBuffer s = new StringBuffer();
   // s.append("b = " + b + "\n" + aObj);
   // return s.toString();
}
```

Python sample code

```
public Object clone()
{
   try
   { B bobj = (B)super.clone();
     bobj.aObj = (A)aObj.clone();
     return bobj;
   }
   catch (CloneNotSupportedException e)
   {
      // return null;
      throw new InternalError(e.getMessag
   }
}
```

Python sample code

```
-------------------------
| B                     |
|    int b;             |
|  -------------        |
|  | A         | ..... |
|  |   int a   |       |
|  -------------        |
-------------------------
class A {
   private int a;
}
class B {
   private int b;
   Vector aobjs;
}
```

- Sample code
- Vector

```
public int getNumOfAobjs() { return aobjs.size(); }
public A getA(int i) {
   return ((A)aobjs.elementAt(i)); }
public String toString() {
   StringBuffer s = new StringBuffer();
   s.append("b = " + b + "\n");
   for (int i=0; i < getNumOfAobjs(); i++)
      s.append(getA(i).toString());
   return s.toString();
}
```

Python sample code

```
public Object clone()
{
   try
   { B bobj = (B)super.clone();
     bobj.aobjs = (Vector)aobjs.clone();
     for (int i=0; i < aobjs.size(); i++)
       bobj.aobjs.setElementAt(
          ((A)aobjs.elementAt(i)).clone()
     }
     return bobj;
   }
   catch (CloneNotSupportedException e)
   {
     // return null;
     throw new InternalError(e.getMessage
   }
}
```

Python sample code

```
-------------
| A         |
|   int a   |
-------------
      |
-------------
| B         |
|   int b   |
-------------
```

Java sample code

Python sample code

Java sample code

Python sample code

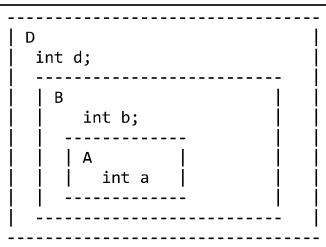
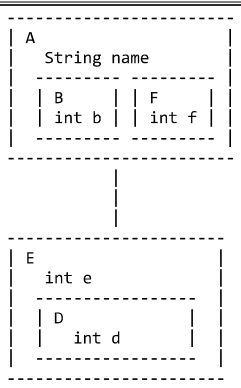
```
-------------
| A         |
|   int a   |
-------------
      |
-------------
| B         |
|   int b   |
-------------
      |
-------------
| D         |
|   int d   |
-------------
```

Java sample code

Python sample code

Java sample code

Python sample code

	Java sample code Python sample code	Java sample code Python sample code
	Java sample code	Java sample code

Notes:

- References
  - Clone
  - Vector of objects

Back | Next

Last modified on: 10/05/2023 20:22:37