JAVA METHOD CALLING

Methods don't do anything until you call them into action. It is recommended that methods you create are being stored in a separate class file and then called, although you can create it in the main method of our java file or application.

Start a new Java Application project. Give your project a name and rename the Main method to something else. Then click finish. In our example; we've called our project `prjmethods`, and the class `TestMethods`:

```
package prjmethods;
public class TestMethods {
      public static void main (String [] args)
{
            //code her
      }
}
```

To add a new class to your project, click *file* from the NetBeans menu. From the file menu, select *New File*. You'll see a dialogue box appear. In the *categories*

section, select *Java Class*. Then click the *Next Button* at the bottom. In step two, type a name for your new class. We've called ours MyMethods. You can leave everything else as defaults. Click *finish*.

We've just created a second class called MyMethods which will be in the project prjmethods. A new tab, which is the new class we just created, will appear in the NetBeans software with some default comments. If we delete the comments we would be left with:

```
package prjmethods;
public class MyMethods {    }
```

Notice that there is not Main method this time–just a blank class and a pair of curly brackets. We can now add our methods.

```
public class MyMethods {
      int total () {
            int a_Value = 10 + 10;
            return a_Value;
      }
}
```

We've just set up our own method "total" which would be returning the sum of 10 and 10 in an int variable "a_Value."

To call the total method, select your TestMethods tab in NetBeans, the one with your Main method. We're going to call the total method from the main method:

```
public static void main (String [] args) {
     MyMethods test1 = new MyMethods ();
}
```

To create a `new Object` from a class, you start with the name of the class, MyMethods in our case. After a space, you type a name for your new MyMethods variable. We've called ours `test1`. An equals sign comes next, followed by the keyword "`new`," type a space followed by your class name again. The class name needs a pair of round brackets after it, this time. End the line with a semicolon. In essence, we've just created a variable `test1` which is not an ordinary one as it holds a class. As it holds a class it is an object, which possesses

methods and properties. To call the total method into action we do this:

```
MyMethods test1= new MyMethods ();
int aVal = test1.total();
System.out.println (aVal);
```

Since our `total ()` method returns a value we ought to store that value in a variable called '`aVal`.' Thereafter we print the '`aVal`' to the screen.