Encapsulation

© 2008 Haim Michael. All Rights Reserved.

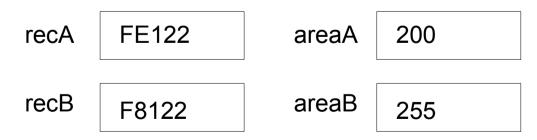
What is Encapsulation?

- Encapsulation, one of the most important OOP principles, is the mechanism that groups together the data and the operations.
- While in procedural programming language calling a function must include the values on which the function should perform its calculation, in object oriented programming languages calling a function doesn't necessarily include sending it the values it requires for its calculations.
 - Each time a method is called it executes on a specific object and capable of accessing each one of the attributes in that specific object.

Encapsulation in Practice

Rectangle		
+width: double +height: double		
+Rectangle() +Rectangle(widthVal: double, heightVal: double) +area(): double		
+perimeter(): double		

Rectangle recA, recB; recA = new Rectangle(10,20); recB = new Rectangle(15,15); areaA = recA.area(); areaB = recB.area();



FE122

width	10
height	20

F8122

width	15
height	15
height	15

© 2008 Haim Michael. All Rights Reserved.

Encapsulation Benefits

- Using the encapsulation mechanism the data can be kept safe from outside interference.
- The encapsulation enables us having the code and the data of each object isolated from the rest of the program.
- Thanks to the encapsulation mechanism we can have a well defined interface to access each one of the objects.

The 'private' Access Modifier

- Setting the 'private' access level for any of the class members will limit the possibility to use it directly from within the scope of the class only.
- Doing so we will limit any code outside of the class that wants to access a 'private' member to do it indirectly by calling one of the class 'public' operations.
- Using the 'private' access modifier we strength the class encapsulation.

The 'private' Access Modifier

FF122 width 10 FF122 recA height 20 recB F8122

Rectangle recA, recB; recA = new Rectangle(10, 20);recB = new Rectangle(15, 15);recA.setHeight(10); recB.setWidth(30); areaA = recA.area(); areaB = recB.area();

Rectangle

+Rectangle(widthVal: double, heightVal: double)

-width: double -height: double

+Rectangle()

+area(): double

+perimeter(): double

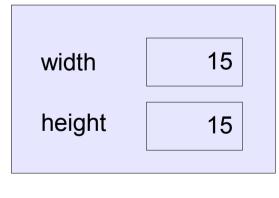
+getWidth(): double

+getHeight(): double

+setWidth(widthVal: double): void

+setHeight(heightVal: double): void

F8122



areaB

450

100 areaA

© 2008 Haim Michael. All Rights Reserved.