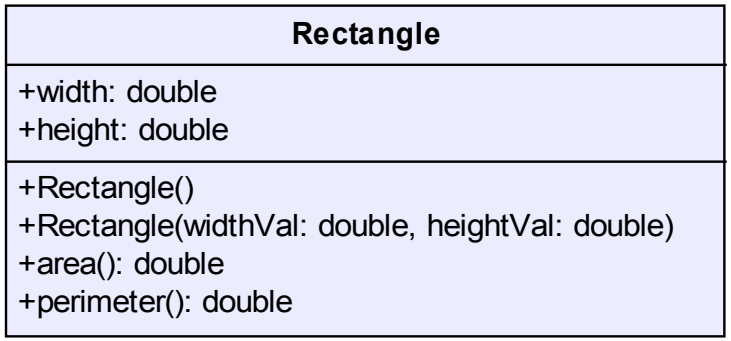
Encapsulation

What is Encapsulation?

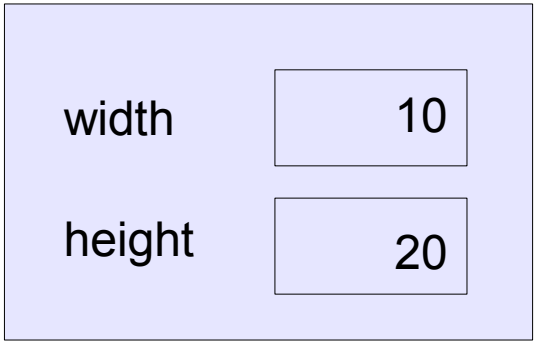
- Encapsulation, one of the most important OOP principles, is the mechanism that groups together the data and the operations.
- While in procedural programming language calling a function must include the values on which the function should perform its calculation, in object oriented programming languages 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

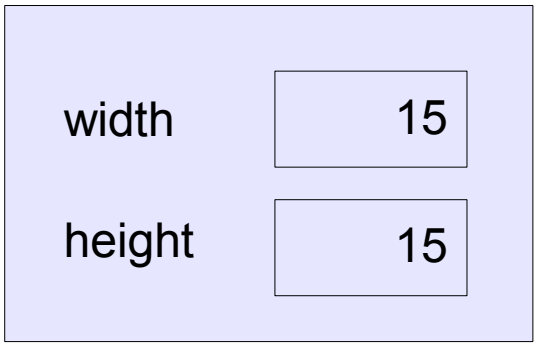


FE122




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

F8122



recA	FE122	areaA	200
recB	F8122	areaB	255

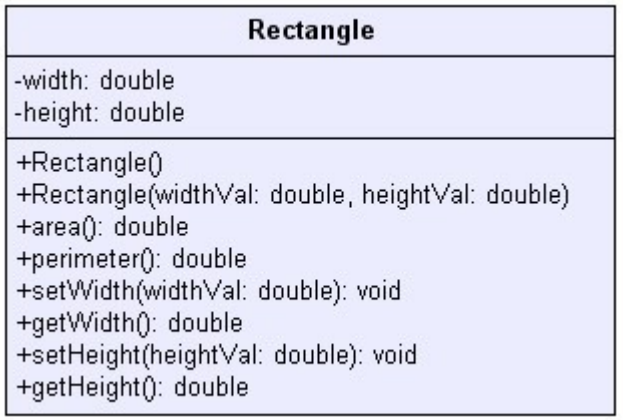
Encapsulation Benefits

- 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




```
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();
```