Abstraction

Abstraction is a process to abstract or hide the functionality and provide users or other programmers to use it only. Like for the method Console.WriteLine(), no one knows what actually is happening behind the function calling. We are just using it by calling and passing the arguments. This is the thing called Abstraction.

Encapsulation

Encapsulation means to encapsulate or put everything into one thing and provide others to use it. Like in a shaving kit, all the necessary kits are available. And also these kits are available as loose in market. But the shaving kit encapsulates every other kit into a small bag and provides a user to use it.

Hope you now have a basic idea about both of these properties. Let's see a real world example of encapsulation and abstraction.

Let's assume you have to create a method to insert users data and pass it to other developers to use. So first, create a class and add a method to insert the data into database with validation.

There will be three fields:

1. Name
2. Email
3. Phone number

So these inputs have to validate first and then insert into db.

First, create a class with all methods:

Hide   Copy Code

class User

{

public bool AddUser(string name, string email, string phone)

{

if (ValidateUser(name, email, phone))

{

if (AddtoDb(name, email, phone) > 0)

{

return true;

}

}

return false;

}

private bool ValidateUser(string name, string email, string phone)

{

*// do your validation*

return true;

}

private int AddtoDb(string name, string email, string phone)

{

*// Write the Db code to insert the data*

return 1;

}

}

As you can see, there are three methods that are written in this User class.

* AddUser: To call from outside the class. That is why the access modifier is public.
* validateUser: To validate the user's details. Can't access from outside the class. It's private.
* AddtoDb: To insert data into database table and again it is private, can't access from outside the class.

Now another user will just call AddUser method with parameters. And that user has no idea what is actually happening inside the method. I didn't write the code to validate and insert into db, as you can get it from others examples. We will discuss about it later.

To call the AddUser method, do as follows:

Hide   Copy Code

class Program

{

static void Main(string[] args)

{

User objUser = new User();

bool f = objUser.AddUser("Arka", "ark@g.com", "1234567890");

}

}

Now come back to the main discussion.

Here, we are hiding the procedure of adding data into database from other users, this is Abstraction. And putting all the three methods into one User class and providing other users to use it, that is called Encapsulation.

So procedure hiding is Abstraction and putting every necessary thing into one is Encapsulation.

**Difference between Abstraction and Encapsulation**  
  
Abstraction is a process. It is the act of identifying the relevant qualities and behaviors an object should possess. Encapsulation is the mechanism by which the abstraction is implemented.

|  |  |
| --- | --- |
| Abstraction | Encapsulation |
| Abstraction solves the problem in the design level. | Encapsulation solves the problem in the implementation level. |
| Abstraction is used for hiding the unwanted data and giving only relevant data. | Encapsulation is hiding the code and data into a single unit to protect the data from outer world. |
| Abstraction is set focus on the object instead of how it does it. | Encapsulation means hiding the internal details or mechanics of how an object does something. |
| Abstraction is outer layout in terms of design.  For Example: - Outer Look of a iPhone, like it has a display screen. | Encapsulation is inner layout in terms of implementation. For Example: - Inner Implementation detail of a iPhone, how Display Screen are connect with each other using circuits |