8

Introduction to Object Oriented Programming

What is OOP

- Object Oriented Programming.
- Programming Model for Scalable Applications.
- Used in most popular languages such as Java, Python, JavaScript, C++ etc.
- Goal of OOP is to group-up some data and its operations as a single unit called "Object".

Objects

- Object is a small unit (entity) in the program that represents a real-world person or thing.
- Ex: You, Your laptop
- Any physical thing can be considered as object.
- Object is instance (example) of "class".
- Object stores a set of fields (details about object).

regNo: MH123
carModel: Honda City
carYear: 2020

regNo: TS456
carModel: Duster
carYear: 2021

regNo: TS789
carModel: Swift
carYear: 2019

Classes

- Class is a model of objects.

- Class (a.k.a "type") represents structure (list of fields and methods) of data that you want to store in similar objects.
- Class isn't collection of objects.
- Objects are created based on "Class".

```
class   Car
{
        string  regNo;
        string  carModel;
        int       carYear;
}
```

Methods

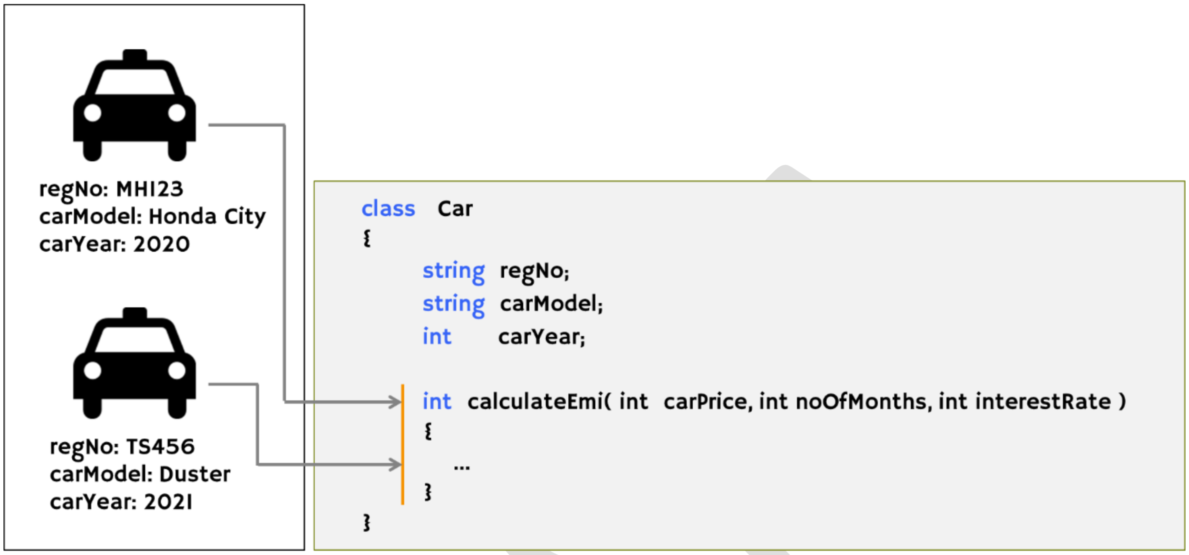
- Method is a collection of statements to perform certain operation (process or work), such as performing some calculation, displaying some output, checking some conditions etc.
- Method should be a member (part) of class.
- The code statements are not allowed outside the class; they are allowed inside the method only.

```
class   Car
{
        int  calculateEmi( int  carPrice, int noOfMonths, int interestRate )
        {
            //do calculation here
            return (emi);
        }
}
```

Object & Class Association

- Object stores fields.

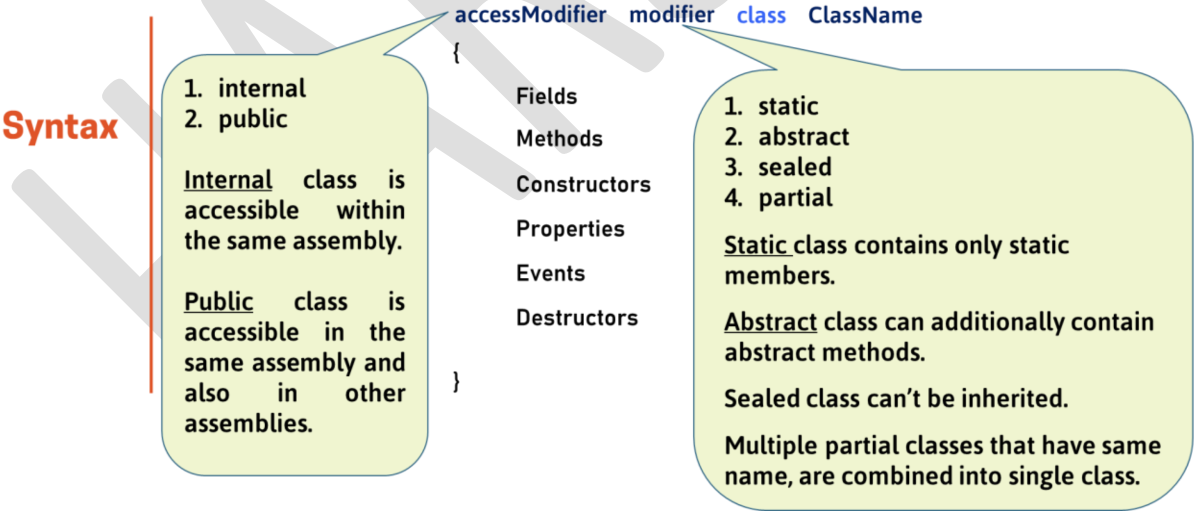
- Object associates with all methods of its class. Means, object can call methods of its class.
- Class declares list of fields; defines list of methods.



```
class   Car
{
        string  regNo;
        string  carModel;
        int     carYear;

        int  calculateEmi( int  carPrice, int noOfMonths, int interestRate )
        {
            ...
        }
}
```

Creating Class



Syntax

```
accessModifier   modifier   class   ClassName
{
        Fields
        Methods
        Constructors
        Properties
        Events
        Destructors
}
```

1. internal
2. public

Internal class is accessible within the same assembly.

Public class is accessible in the same assembly and also in other assemblies.

1. static
2. abstract
3. sealed
4. partial

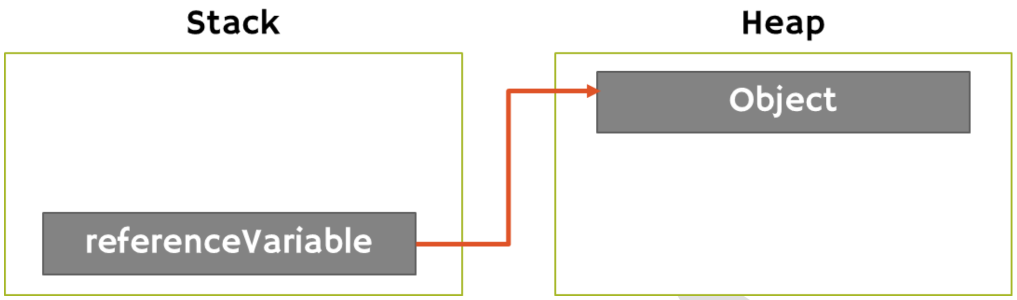
Static class contains only static members.

Abstract class can additionally contain abstract methods.

Sealed class can't be inherited.

Multiple partial classes that have same name, are combined into single class.

Creating Object



1. Creating Reference Variable

```
ClassName referenceVariable;
```

2. Create Object and Store its reference into the Reference Variable

```
referenceVariableName = new  ClassName( );
```

- Object is a programmatic representation of a person or thing.
- All objects are created based on classes; stored in 'heap'.
- For each application execution, a new heap will be created (and only one).
- All reference variables (local variables of methods) are stored in stack. For each method call, a new stack will be created.
- Method is a collection of statements to perform some operation / calculation.
- Class supports two access modifiers: 'internal' and 'public'.
- Class supports four modifiers: 'static', 'abstract', 'sealed' and 'partial'.
- Objects stores actual data (group of fields) & can access methods of class.
- A reference variable stores address of an (only one) object.