Object-Orientation programming

Object

An object is something you can touch and feel, like Ali, a school, a house, or a car. It can also be something you understand in your mind, like time or a date. So, objects can be things you physically interact with or things you think about.

ایک شے ایک ایسی چیز ہے جسے آپ چھو سکتے ہیں اور محسوس کرسکتے ہیں ، جیسے علی ایک اسکول ، ایک گھر ، یا ایک گاڑی۔ یہ کچھ ایسا بھی ہوسکتا ہے جو آپ اپنے ذہن میں سمجھتے ، ہیں ، جیسے وقت یا تاریخ۔ لہذا ، اشیاء وہ چیزیں ہوسکتی ہیں جن کے ساتھ آپ جسمانی طور پر بات چیت کرتے ہیں یا وہ چیزیں جن کے بارے میں آپ سوچتے ہیں۔

An Object has some attributes (properties) and behavior (function)

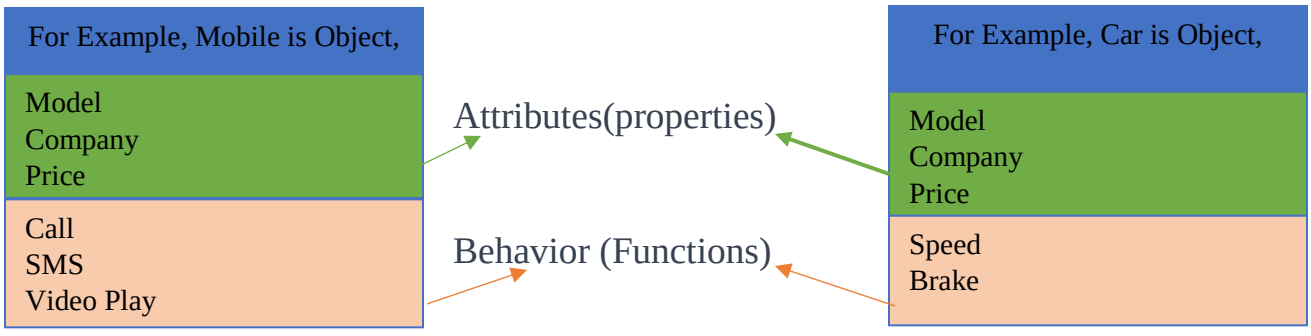
آبجیکٹ میں کچھ خصوصیات)خصوصیات (اور طرز عمل)فنکشن (ہیں

State (attributes): This is like describing what the object is or what it has. For example, if the object is a car, its color, model, and speed are part of its state.

حالت)خصوصیات :(یہ بیان کرنے کی طرح ہے کہ شے کیا ہے یا اس میں کیا ہے .مثال کے طور پر ، اگر آبجیکٹ کار ہے تو ، اس کا رنگ ، ماڈل ، اور رفتار اس کی حالت کا حصہ ہیں۔

Well-defined behavior (operations): This is like saying what the object can do. Using the car example again, the actions it can perform, like accelerating, braking, are part of its behavior. So, an object not only has characteristics (state/attributes/properties) but also can do certain things (behavior / function).

اچھی طرح سے بیان کردہ طرز عمل)آپریشن :(یہ کہنے کی طرح ہے کہ آبجیکٹ کیا کرسکتا ہے۔ گاڑی کی مثال کو دوبارہ استعمال کرتے ہوئے ، وہ عمل جو وہ انجام دے سکتا ہے ، جیسے تیز کرنا ، بریکنگ / اس کے طرز عمل کا حصہ ہیں۔ لہذا ، کسی شے میں نہ صرف خصوصیات)حالت /صفات ، خصوصیات (ہیں بلکہ کچھ چیزیں)طرز عمل /فنکشن (بھی کرسکتے ہیں۔



Object-oriented programming (OOP

It is a technique in which we visualize our programming problems in the form of objects and their interactions as happen in real life.

یہ ایک ایسی تکنیک ہے جس میں ہم اپنے پروگرامنگ کے مسائل کو اشیاء کی شکل میں دیکھتے ہیں اور ان کے تعامل ات حقیقی زندگی میں ہوتے ہیں۔

OR

Object-oriented programming (OOP) is a technique that encourages developers to model software after real-world entities, where these entities are represented as objects with attributes and behaviors. This approach aims to make the software design "easier to understand" or "more user-friendly.", modular, and "Solves real-world issues" or "addresses practical problems.".

آبجیکٹ اورینٹڈ پروگرامنگ)او او پی (ایک ایسی تکنیک ہے جو ڈویلپرز کو حقیقی دنیا کے اداروں کے بعد سافٹ ویئر ماڈل کرنے کی ترغیب دیتی ہے ، جہاں ان اداروں کو خصوصیات اور طرز عمل کے ساتھ اشیاء کے طور پر پیش کیا جاتا ہے۔ اس نقطہ نظر کا مقصد سافٹ ویئر ڈیزائن کو "سمجھنے میں آسان "یا "زیادہ صارف دوست"، ماڈیولر، اور "حقیقی دنیا کے مسائل کو حل کرتا ہے "یا "عملی مسائل کو حل کرتا ہے".

The main parts or principles of OOP are:

Class: -

A class is a blueprint or template for creating objects.
It defines the properties (attributes) and behaviors (methods) that the objects based on the class will have.
Acts as a user-defined data type

Blueprint or Template:

In Simple Terms: Imagine you want to build a house. Before you start building, you create a detailed plan that shows the layout, number of rooms, where the doors and windows are, and what materials to use. This plan is like a blueprint.

In Programming Terms: In programming, a class is like a blueprint for a house. It defines the structure and behavior of the house. It specifies what features (attributes) the house will have, such as the number of rooms, color, and type of roof. It also describes what the house can do, like having doors that can be opened or windows that can be closed.

نقشہ یا ڈیزائن

سادہ الفاظ میں: تصور کریں آپ نے ایک گھر بنانا ہے۔ جب آپ بنانا شروع کرتے ہیں، آپ ایک تفصیلی منصوبہ بناتے ہیں جو چھت، کمرے کی تعداد، دروازے اور کھڑکیوں کا وضع، اور استعمال ہونے والے مواد کو دکھاتا ہے۔ یہ منصوبہ ایک نقشہ کی طرح ہوتا ہے۔

پروگرامنگ میں: پروگرامنگ میں، ایک کلاس ایک گھر کے لئے ایک نقشہ کی طرح ہوتی ہے۔ یہ گھر کی ساخت اور رویہ کو تعین کرتی ہے۔ یہ متعین کرتی ہے کہ گھر کے کیا خصوصیات ہوں گی، جیسے کہ کمروں کی تعداد، رنگ، اور چھت کی قسم۔ یہ بھی بیان کرتی ہے کہ گھر کیا کر سکتا ہے، جیسے کہ دروازے کو کھولنا یا کھڑکیاں بند کرنا

Object: -

An object is an instance of a class.
It represents a real-world entity and encapsulates both data (attributes) and the methods (functions) that operate on the data.
Objects can interact with each other through their methods.

Creating Objects:

In Simple Terms: Now, think about the blueprint of the house. Once you have the plan, you can start building multiple houses based on that plan. Each house can have its own unique paint color, furniture, and decorations.

In Programming Terms: In programming, when we have a class (the blueprint for the house), we can create many instances of that class, and each instance is called an object. Each house object created from the same class can have its own specific details, like a red door, blue walls, or a wooden floor. Each house can also perform actions, such as turning on lights or opening windows, just like how each real house can be unique and functional.

اشیاء بنانا

سادہ الفاظ میں :اب گھر کے منصوبے کا خیال کریں۔ جب آپ کا منصوبہ ہوتا ہے، آپ اس منصوبے کے بنیاد پر متعدد گھروں کی تعمیر کرنا شروع کرتے ہیں۔ ہر گھر مختلف رنگ، فرنیچر، اور سجاوٹ کے ساتھ ہوتا ہے۔

پروگرامنگ میں: پروگرامنگ میں، جب ہمارے پاس ایک کلاس ہوتی ہے گھر کے لئے نقشہ، ہم اس کی بہت سی مثالیں بنا سکتے ہیں اور ہر ایک مثال کو ایک اہم شے اشیاء کہا جاتا ہے۔ ہر گھر جو ایک ہی کلاس سے بنتا ہے اس کی خاص تفصیلات ہوتی ہیں، جیسے کہ سرخ دروازہ، نیلے دیوار، یا لکڑی کی فرش۔ ہر گھر بھی کچھ کر سکتا ہے، جیسے کہ روشنی چالو کرنا یا کھڑکیاں کھولنا، بالکل ایسا ہی جیسے ہر ایک اصل گھر مخصوص اور کارروائی پذیر ہوتا ہے

So, in this example, a class is like the blueprint or plan for a house, and creating objects means building actual houses based on that plan, each with its own distinct features and capabilities.

اس مثال میں، ایک کلاس گھر کے لئے نقشہ یا منصوبہ کی طرح ہوتی ہے، اور اشیاء بنانا یعنی اس منصوبے پر مبنی حقیقی گھروں کی تعمیر کرنا، ہر ایک کو اپنی خصوصیتوں اور صلاحیتوں کے ساتھ۔

Code

```
// Define a class named House
class House {
   // Properties or attributes of the house
    numberOfRooms: number;
    color: string;

   // Constructor to initialize the properties when creating an instance
    constructor(numberOfRooms: number, color: string) {
      this.numberOfRooms = numberOfRooms;
      this.color = color;
    }

    // Method to describe what the house can do
    openDoor(): void {
      console.log('Door is open.');
    }
  }
```

```
  // Creating objects (instances) of the House class
  const myHouse = new House(3, 'Blue');
  const neighborHouse = new House(4, 'Red');

  // Using the describeHouse method to get information about each house
  console.log(`My house has ${myHouse.numberOfRooms} rooms and is
${myHouse.color}.`);
  myHouse.openDoor();

  console.log(`Neighbor's house has ${neighborHouse.numberOfRooms} rooms and is
${neighborHouse.color}.`);
  neighborHouse.openDoor();
```