OOP

OBJECT ORIENTED PROGRAMMING

PROGRAMMING MODELS

- There are two models in programming.
- Process Oriented Model or ProcedureOrientedModel

  Eg: Pascal, C
- ObjectOrientedModel

  Eg: C++, Java, CSharp

SCENARIO

- *Sam is 35 years old ,he works for ABC Ltd as a general manager.*
- *He travels to office in his red alto car.*

Let us analyse this in a process model.

- There are 2 actions in this.
- works()
- travels()
- Which tells us what actions are done here, but who or which object is doing the action is not told.

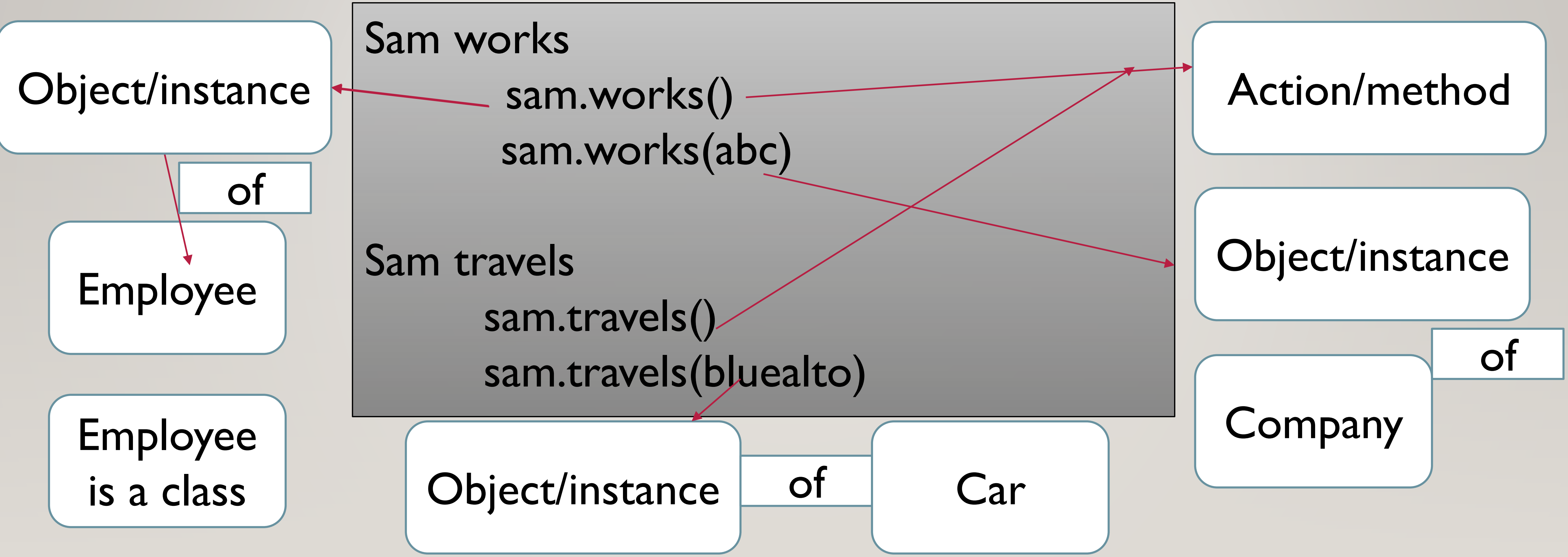
Technically,
Works()
Travels() are written as procedures or functions.
Eg:

```
void works()
 {
     //works in ABC Ltd.

}

void travels()
{
     //travels by alto car
}
```

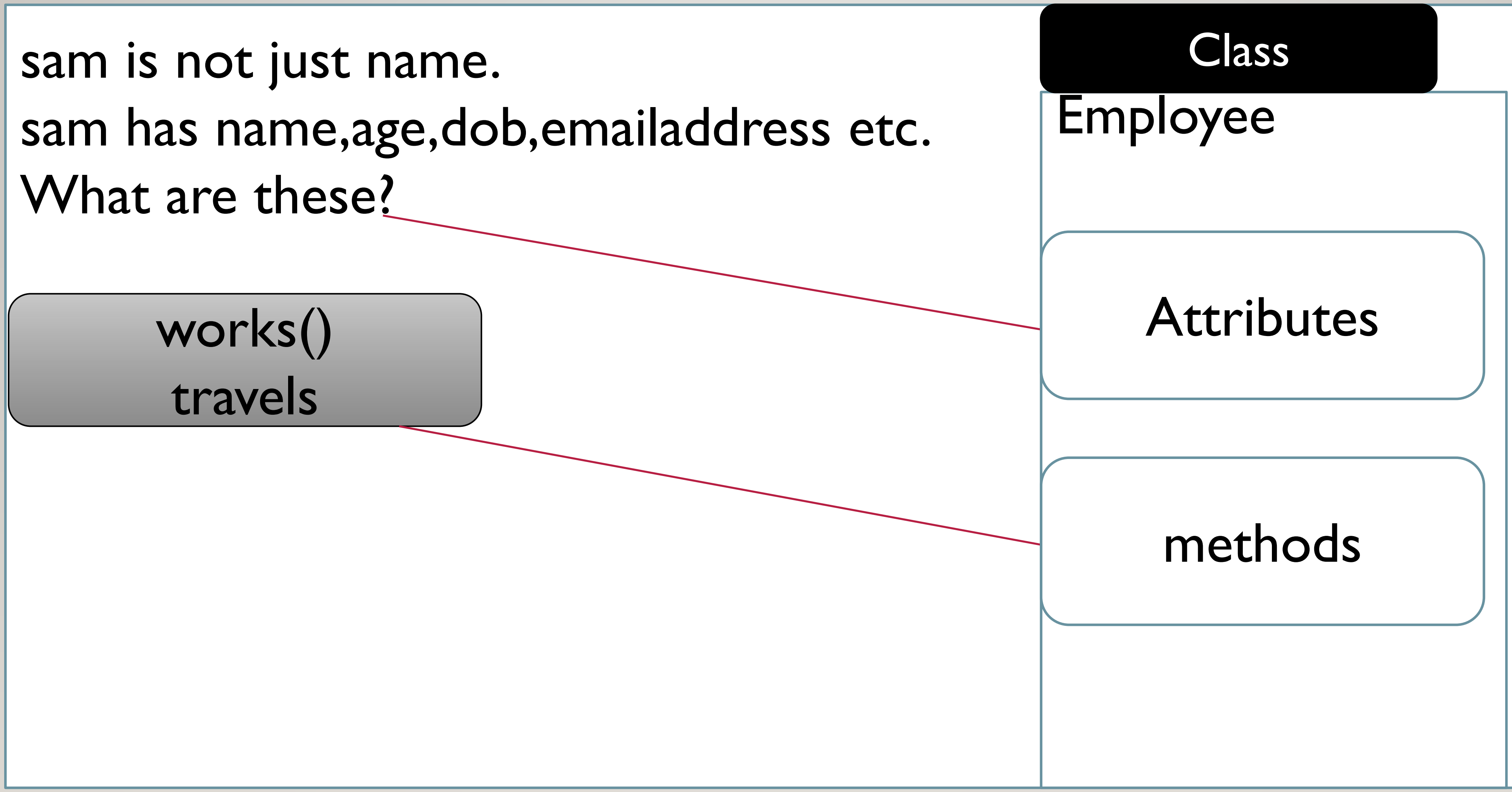
Let’s reiterate the scenario with a different perspective



```
Sam works
        sam.works()
        sam.works(abc)

Sam travels
        sam.travels()
        sam.travels(bluealto)
```

sam is not just name.
sam has name,age,dob,emailaddress etc.
What are these?



How does this look as a program?

```
public class Car
{
    private String color;
    private String model;

.......
    public moves()
    {
        //code
    }
}
```

Attributes also known as instance variables

methods

```
public class Employee
{
    private String name;
    private String dob;
    .....
    public travels(Car car)
    {
    ..
    }
}
```

Attributes also known as instance variables

ENCAPSULATION

This is ABSTRACTION.

Sam is driving a Car
He knows he drives his car which is blue in color
And is of alto model.
Does he know or should he know the chassis no of his car?or should he know when the car was manufactured?
Not Really.
Some details can be exposed and some hidden when operating on the object.

Peter has an account in XYZ bank.
He can withdraw and deposit amount into his account.

Which are the objects ,classes,attributes and methods in this scenario?

EFFECTS OF ENCAPSULATION.

- Data Security

  - Creating private data

- Data Hiding-
- Implementation hiding

- Creation of composite types.
  - - class forms a wrapper around data and code.

OBJECTS

- Object is a runtime entity.
- It is an instance of a class
- Stored in heap
- Has state and behavior.
- State of an object is values stored in it at a particular point of time.

Eg:

- Employee sam=new Employee();

Calls a Default Constructor and initializes with default values

Reference

Operator to create an object in heap

OBJECTS

- Employee john=new Employee(“John”,”21-03-1997”);

Reference

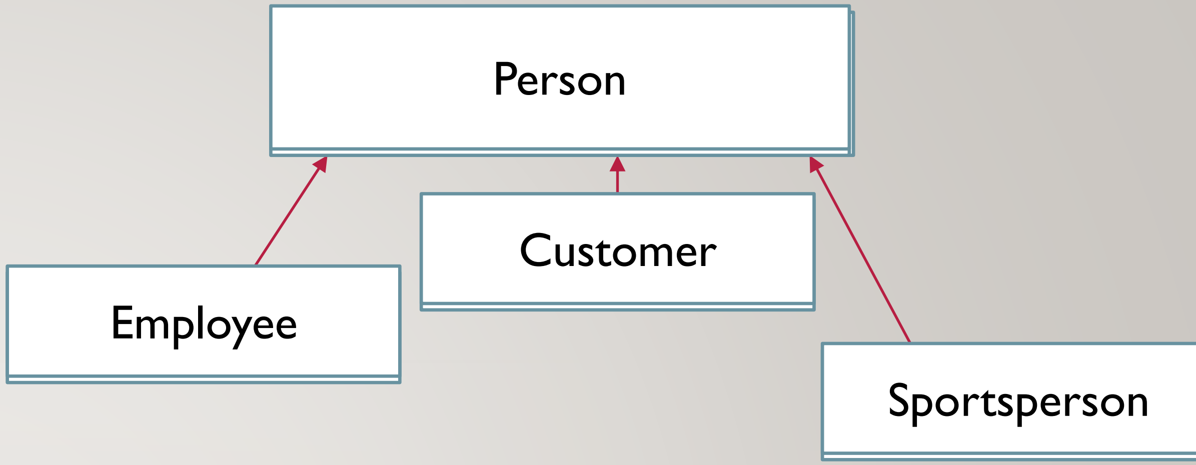
Operator to create an object in heap

Calls a parameterized Constructor and initializes with given values

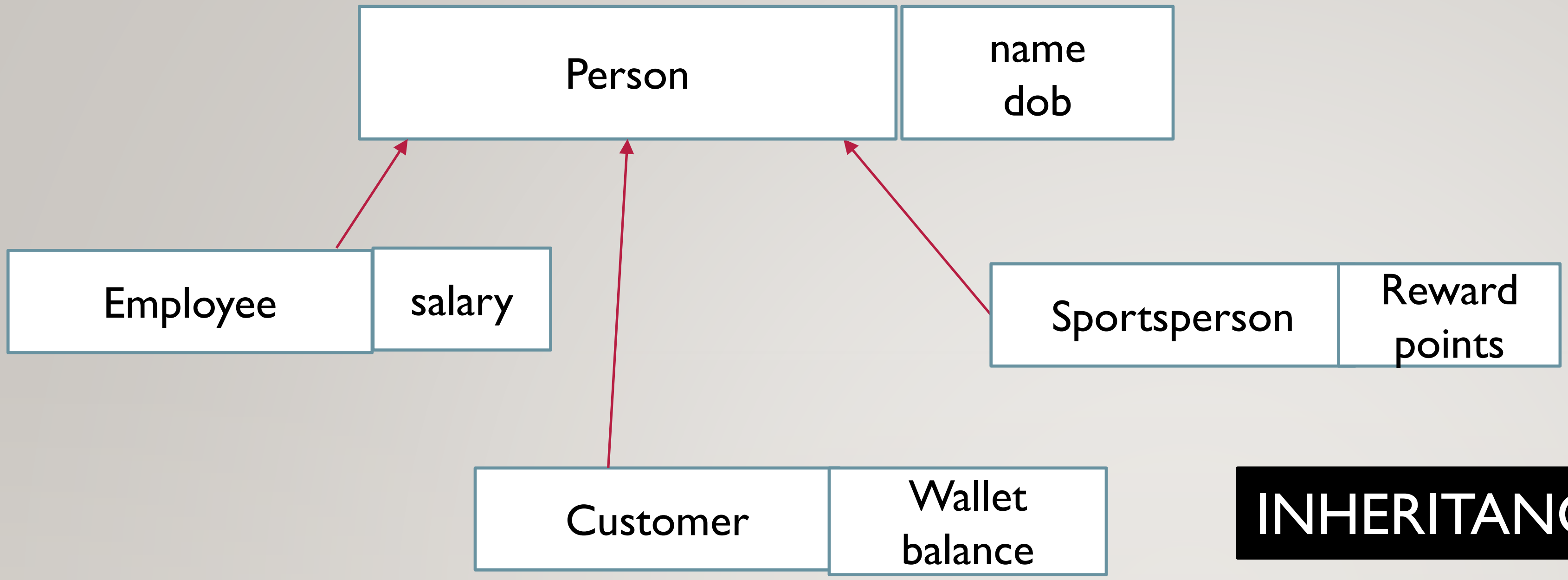
Constructor is a special method of a class that shares it’s name with the class and defined to initialize values to the object

Wait for more concepts on constructors

Sam is an employee.
Sometimes he uses cab to commute, he is a customer for uber where he uses his wallet to pay.
Evening he plays squash, he is also a sports guy he has scored some reward points.



How do we create classes for the above?



INHERITANCE

Inheritance is a feature where an existing class is enhanced to have extended capabilities with more specific behavior and attributes.
Maintains **is-a** relationship among classes.
A subclass is created without affecting the exisiting class.

INHERITANCE EXAMPLE

```
public class Customer extends Person{
//attributes
// methods
}
```

Keyword

Inheritance is used for reusability and extendibility

An Online Library lends both ebooks ,podcasts and videos on subscription.
Ebook is identified by an title and author.
Podcast by a topic.
Video by a title.

Observe this
12+23 => 35
“12”+23 =>1223
“hello”+”world”=> “helloworld”

Operator + is both an addition operator and a concatenator
, but at different times,behaves differently at different times.

POLYMORPHISM
Single interface multiple Forms

Static

Dynamic

The above example is static polymorphism.
Compiler knows what action need to be done based on the types

```
Number n1=new Number();
n1.add(12,23); //(int,int)
n1.add(12.5,25.8); //(double,double)
n1.add("hello","world")
// String,String
```

- Static Polymorphism
- Compiler binds the method add with the object n1 based on the datatypes which are different.
- Method add has three different signatures.
- This is also called as **OverLoading.**

DYNAMIC POLYMORPHISM OR LATE BINDING

- If there is a behavior in a class called makePayment() then I am not giving complete information about how I makePayment(), but during execution I dispatch a required implementation of the behavior.
- Eg:
- Payment pref=new CashPayment();
- pref.makePayment();
- pref=new ChequePayment();
- pref.makePayment();

OOP RECAP TEST

1 ________________ is a runtime entity for a class.

2 An Object has _______________ and _______________

3 In a class behavior is depicted by defining ____________________

4 _______________ represents the value that is stored in an object at a given point of time.

5 ____________________ is the oop concept where objects are operated upon with only relevant details.

6 ________________ is an encapsulated unit of data and methods.

7 _____________ members are confined only to the class in which they are defined.

8______________ members are exposed to the client.

9 _________ relationship exists between a base class and a child class.