Learning Objective

Click to continue

We will learn about OOPS Fundamentals:

- Inheritance
- Encapsulation
- Polymorphism

What is Encapsulation?

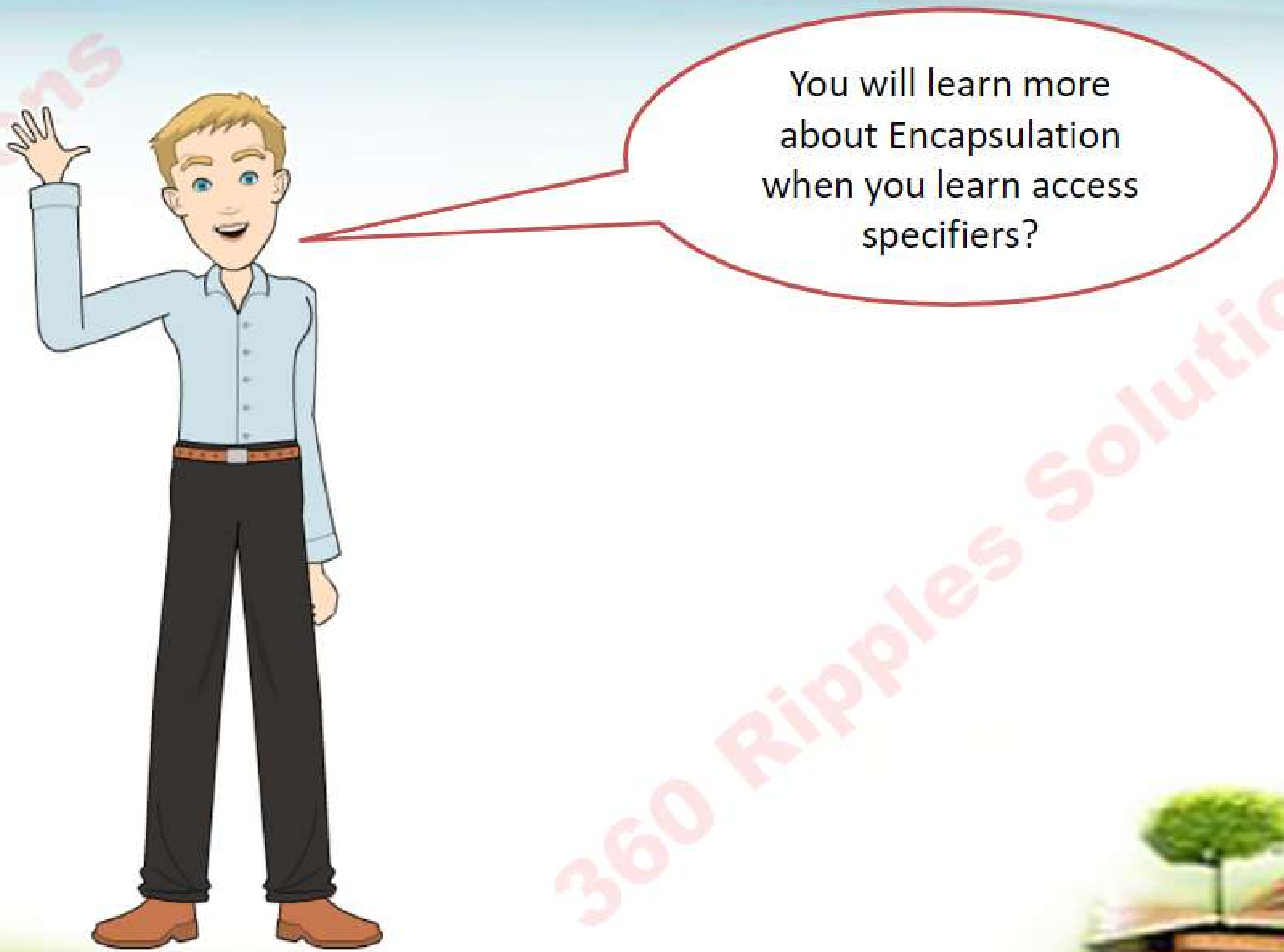
Click to continue

Hiding internal state of an object is known as ***data encapsulation*** *(aka **Data hiding**).*

Example :

In a retail application the product availability (quantity) in stock is hidden from customers. The quantity will be deduced when customers buy products.

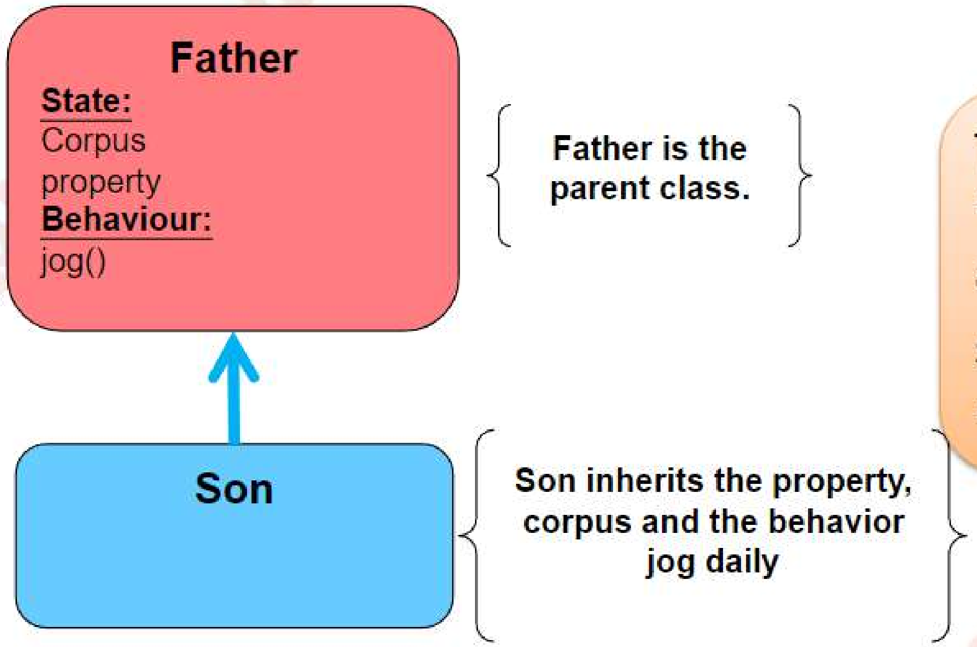
Click to continue


You will learn more about Encapsulation when you learn access specifiers?

What is Inheritance?

Click to continue

Object-oriented programming allows classes to *inherit* the state and behavior from other classes.



Father
State:
Corpus
property
Behaviour:
jog()
Son
Father is the parent class.
Son inherits the property, corpus and the behavior jog daily

The father is called the "*Super*" (or) "*base*"class and the

son is called the

"*Sub*"*class.*

Deep Dive into Inheritance

Click to continue

- Child class inherits all the behavior and states of the parent class.
- All the common methods which needs to be reused across many classes are placed in the parent class
- The sub class can also override the definition of existing methods by providing its own implementation.

You will learn about overriding soon.

What is Polymorphism?

Click to continue

Polymorphism per dictionary refers to a principle in biology in which an organism or species can have many different forms or stages.

OOPS Polymorphism:

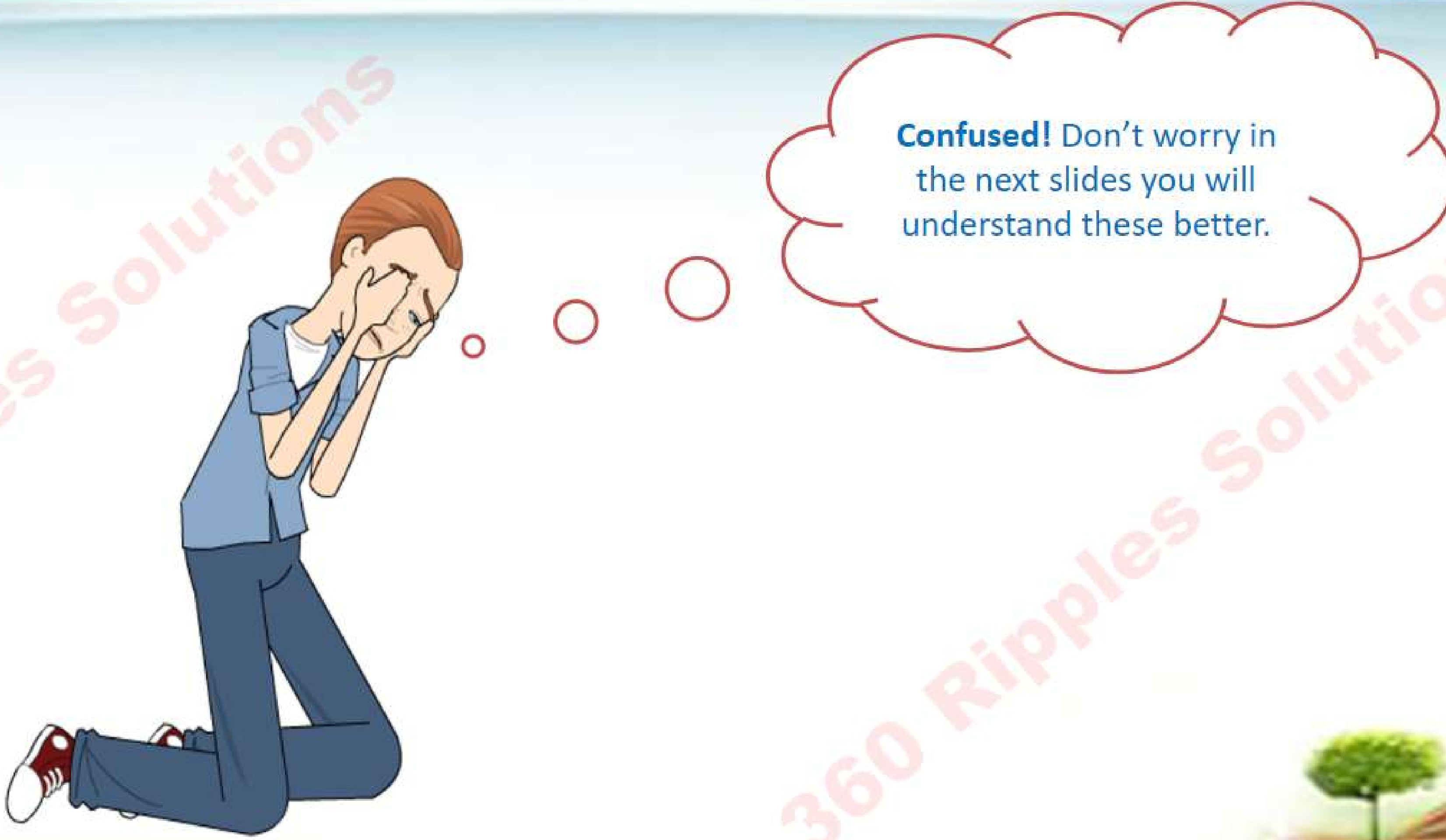
Subclasses of a class can define their own unique behaviors and yet share some of the same functionality of the parent class.

Ways of achieving Polymorphism

Click to continue

- **Method Overloading –** Two different implementations of the same method available in the same class.
- **Method overriding** – Methods of a subclass override the methods of a super class.
- **Dynamic Method Binding –** At run time the interpreter will find out which objects method needs to be executed.

Click to continue


Confused! Don't worry in the next slides you will understand these better.

Method Overloading

Click to continue

Method Overloading – Two different implementation of the same method available in the same class.

- Changing the input parameter data type.
- Changing the number of input parameters.

Lets assume there is a object “Boy” who can “run” (behavior). Now the boy runs casually during playing but in a race he will run fast. So he has two implementations of running.

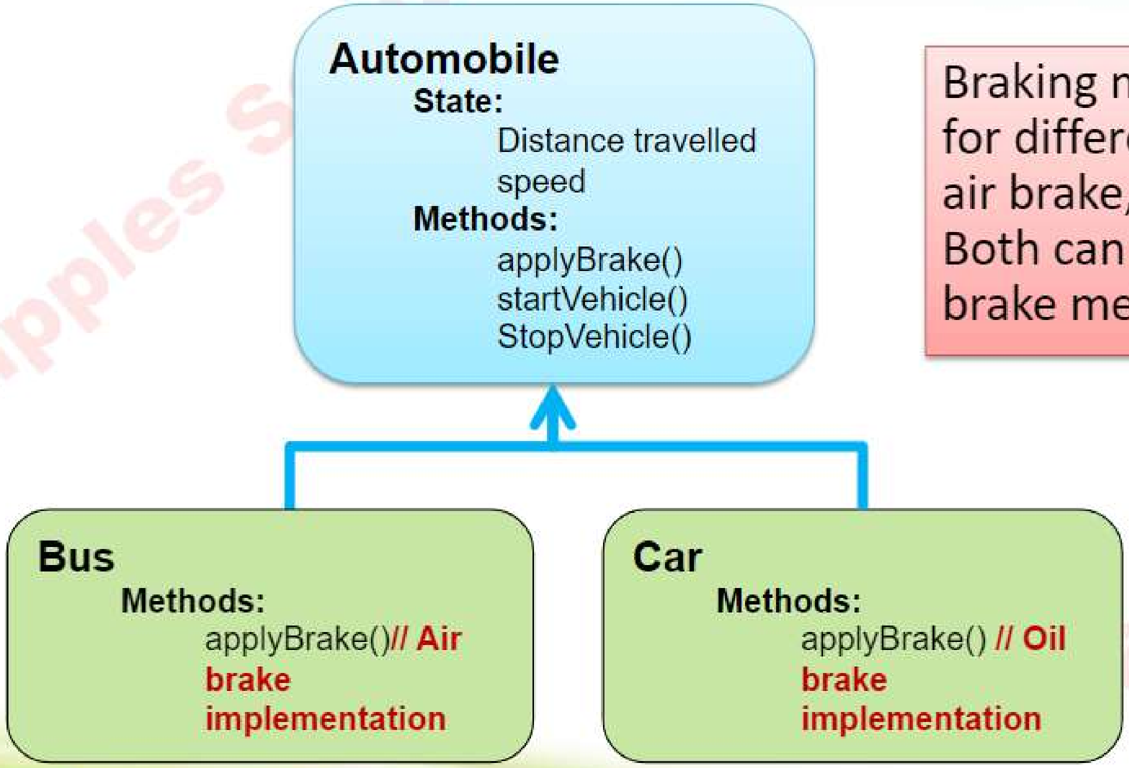
Implementation 1:
```
run()
{
        Run casually
}
```

Implementation 2:
```
run(race)  // Added  input parameter, to
overload  method.
{
        Run fast.
}
```

Method Overriding

Click to continue

Method overriding – Methods of a base class override the methods of a super class.



Braking methods are different for different vehicle. Bus uses air brake, car uses oil brake. Both can override the apply brake method accordingly.