In C#, it is possible to inherit fields and methods from one class to another. We group the "inheritance concept" into two categories:

- **Derived Class** (child) - the class that inherits from another class
- **Base Class** (parent) - the class being inherited from

To inherit from a class, use the `:` symbol.

In the example below, the `Car` class (child) inherits the fields and methods from the `Vehicle` class (parent):

Example

```
class Vehicle  // base class (parent)
{
  public string brand = "Ford";  // Vehicle field
  public void honk()             // Vehicle method
  {
    Console.WriteLine("Tuut, tuut!");
  }
}
```

```
class Car : Vehicle  // derived class (child)
{
  public string modelName = "Mustang";  // Car field
}


class Program
{
  static void Main(string[] args)
  {
    // Create a myCar object
    Car myCar = new Car();


    // Call the honk() method (From the Vehicle class) on the myCar object
    myCar.honk();
    // Display the value of the brand field (from the Vehicle class) and the value of the
modelName from the Car class
    Console.WriteLine(myCar.brand + " " + myCar.modelName);
  }
}
```

Why and When to Use "Inheritance"?

- It is useful for code reusability: reuse fields and methods of an existing class when you create a new class.

Tip: Also take a look at the next chapter, Polymorphism, which uses inherited methods to perform different tasks.

The sealed Keyword

If you don't want other classes to inherit from a class, use the `sealed` keyword:

If you try to access a `sealed` class, C# will generate an error:

```
sealed class Vehicle
{
  ...
}


class Car : Vehicle
{
  ...
}
```

The error message will be something like this:

```
'Car': cannot derive from sealed type 'Vehicle'
```