[0810] Using One or More Interfaces

Here's a simple example program that uses a bird-oriented interface I like to call "ISquawkable":

```
public interface ISquawkable
{
    void Squawk();
}

class BabyChick : ISquawkable
{
    public void Squawk()
    {
        Console.WriteLine("Peep! Peep!");
    }
}

class Eagle : ISquawkable
{
    public void Squawk()
    {
        Console.WriteLine("SCREECH!!");
    }
}

class Application
{
    public static void Main()
    {
        ISquawkable[] birds = {new BabyChick(), new Eagle(), new Eagle(), new BabyChick()};

        foreach (ISquawkable bird in birds)
        {
            bird.Squawk();
        }
    }
}
```

What output will this program produce?

ISquawkable is a single method interface that requires any class that implements it to create a void Squawk() method.

Because both BabyChick() and Eagle() implement the ISquawkable interface (and therefore contain the required Squawk() method), they can both be placed into the birds array - i.e, a collection of ISquawkable objects. And thus, when we use a foreach() loop to iterate thru that array, we can reliably call the Squawk() method on any object in that array.

Note: If the Eagle class also implemented the ICloneable interface (for instance), then it would look like this:

```
class Eagle : ISquawkable, ICloneable
{
    public void Squawk()
    {
       Console.WriteLine("SCREECH!!");
    }

    public object Clone()
    {
        return this.MemberwiseClone();
    }
}
```

So that is an example of a class implementing multiple interfaces.