Chapter 4

Base Classes and Inheritance

Class inheritance is a commonly used construct\(^1\) in object-oriented languages, and C# provides a full implementation.

**The Engineer Class**

The following class implements an Engineer class and methods to handle billing for that Engineer.

```csharp
using System;
class Engineer
{
    // constructor
    public Engineer(string name, float billingRate)
    {
        m_name = name;
        m_billingRate = billingRate;
    }
    // figure out the charge based on engineer's rate
    public float CalculateCharge(float hours)
    {
        return(hours * m_billingRate);
    }
    // return the name of this type
    public string TypeName()
    {
        return("Engineer");
    }
    private string m_name;
    private float m_billingRate;
}
class Test
{
    public static void Main()
```

\(^1\)Too commonly used, in my opinion, but that discussion would be another book.
{  Engineer engineer = new Engineer("Hank", 21.20F);  Console.WriteLine("Name is: {0}", engineer.TypeName());  }
}

Engineer will serve as a base class for this scenario. It contains private fields to store the name of the engineer and the engineer's billing rate, along with a member function that can be used to calculate the charge based on the number of hours of work done.

Simple Inheritance

A CivilEngineer is a type of engineer and therefore can be derived from the Engineer class:

using System;
class Engineer {
  public Engineer(string name, float billingRate) {
    m_name = name;
    m_billingRate = billingRate;
  }

  public float CalculateCharge(float hours) {
    return(hours * m_billingRate);
  }

  public string TypeName() {
    return("Engineer");
  }

  private string m_name;
  protected float m_billingRate;
}
class CivilEngineer: Engineer {
  public CivilEngineer(string name, float billingRate) : base(name, billingRate) {
  }

  // new function, because it's different than the same as base version
  public new float CalculateCharge(float hours) {
    if (hours < 1.0F) {
      hours = 1.0F;        // minimum charge.
    }
  }

}