

instance names the class type. This is referred to as *strong typing*, and allows the compiler to cross-check to make sure that references to the other classes are valid at compile time.

```

/*-----
File      : SampleClassRunner.p
Purpose   : Wrapper procedure to run SampleClass.cls
-----*/

DEFINE VARIABLE miEmployeeCount AS INTEGER NO-UNDO.
DEFINE VARIABLE miCountryCount AS INTEGER NO-UNDO.
DEFINE VARIABLE moSampleClass AS OOSamples.SampleClass NO-UNDO.

moSampleClass = NEW OOSamples.SampleClass().

moSampleClass:InitializeNotes(INPUT "Admin").

miEmployeeCount = moSampleClass:EmployeeCount.

miCountryCount = moSampleClass:AssignCountry (INPUT "France").

DELETE OBJECT moSampleClass.

MESSAGE "There are" miEmployeeCount "employees, of which" SKIP
miCountryCount "have just been assigned to France."
VIEW-AS ALERT-BOX.

```

The presentation on ***Using Interfaces as a Programming Contract*** creates an interface for a set of classes that do data management.

```

/*-----
File      : IDataManager
-----*/

USING Progress.Lang.*.

INTERFACE OOSamples.IDataManager:
    METHOD PUBLIC HANDLE FetchData (INPUT pcFilter AS CHARACTER ).
    DEFINE PUBLIC PROPERTY RowCount AS INTEGER GET.
END INTERFACE.

```

The interface acts as a contract. A class that **IMPLEMENTS** the interface must adhere to the contract, and this is verified by the compiler.

```

/*-----
File      : EmployeeManager
-----*/

USING Progress.Lang.*.
USING OOSamples.IDataManager.

CLASS OOSamples.EmployeeManager IMPLEMENTS IDataManager:

```

EmployeeManager.cls implements the **PUBLIC** property defined in the interface, and extends it to specify that the property cannot be set by another class.

```

DEFINE PUBLIC PROPERTY RowCount AS INTEGER
GET.
PRIVATE SET.

```

It defines a temp-table and a query specific to employees.

```

DEFINE TEMP-TABLE ttEmployee
  FIELD EmployeeFirstName AS CHARACTER
  FIELD EmployeeLastName AS CHARACTER
  FIELD EmployeePosition AS CHARACTER.

DEFINE QUERY qEmployee FOR AutoEdge.Employee.

```

The default constructor runs a constructor in this class's super class, if there is one.

```

CONSTRUCTOR PUBLIC EmployeeManager ( ):
  SUPER ().

END CONSTRUCTOR.

```

EmployeeManager implements **FetchData**, populating its temp-table with rows from the **Employee** table, and setting the **RowCount** property.

```

METHOD PUBLIC HANDLE FetchData( INPUT pcFilter AS CHARACTER ):

  DEFINE VARIABLE hQuery AS HANDLE NO-UNDO.

  hQuery = QUERY qEmployee:HANDLE.
  hQuery:QUERY-PREPARE("FOR EACH AutoEdge.Employee WHERE " + pcFilter).
  hQuery:QUERY-OPEN ().
  hQuery:GET-FIRST ().
  DO WHILE NOT hQuery:QUERY-OFF-END:
    CREATE ttEmployee.
    BUFFER-COPY AutoEdge.Employee TO ttEmployee.
    hQuery:GET-NEXT ().
  END.
  RowCount = hQuery:NUM-RESULTS.
  hQuery:QUERY-CLOSE ().
  RETURN TEMP-TABLE ttEmployee:HANDLE.

END METHOD.

```

EmployeeManager can also have additional methods or properties that are not specified by the interface, such as this one:

```

METHOD PUBLIC VOID InitializeNotes ( INPUT pcPosition AS CHARACTER ):

/*-----
  Purpose:      Assign a value to all empty EmployeeNotes.
  Notes:        Done for a specific EmployeePosition.
-----*/

  FOR EACH AutoEdge.Employee WHERE Employee.EmployeePosition = pcPosition:
    Employee.EmployeeNotes = "Initial note for this " + pcPosition.
  END.

END METHOD.

```