How would you explain this?

4. In the code editor, click in the blank line below the comment, and then type the following code:

```
public event EventHandler LogonSuccess;
```

5. In the **Task List** window, double-click the **TODO: Exercise 1: Task 2b: Implement the Logon_Click event handler for the Logon button** task.
6. In the code editor, click in the blank line below the comments, and then type the following code:

```
private void Logon_Click(object sender, RoutedEventArgs e)
{
    // Save the username and role (type of user) specified on the form in the   global context
    SessionContext.UserName = username.Text;
    SessionContext.UserRole = (bool)userrole.IsChecked ? Role.Teacher : Role.Student;

    // If the role is Student, set the CurrentStudent property in the global    context to a dummy student; Eric (
    if (SessionContext.UserRole == Role.Student)
    {
        SessionContext.CurrentStudent = "Eric Gruber";
    }

    // Raise the LogonSuccess event
    if (LogonSuccess != null)
    {
        LogonSuccess(this, null);
    }
}
```

How would you explain this?

```
<Button Grid.Row="3" Grid.ColumnSpan="2"
VerticalAlignment="Center" HorizontalAlignment="Center"
Content="Log on" FontSize="24" Click="Logon_Click" />
```

How would you explain this?

```
// Handle successful logon
private void Logon_Success(object sender, EventArgs e)
{
    // Update the display and show the data for the logged on user
    logonPage.Visibility = Visibility.Collapsed;
    gridLoggedIn.Visibility = Visibility.Visible;
    Refresh();
}
```

```
<y:LogonPage x:Name="logonPage" LogonSuccess="Logon_Success" Visibility="Collapsed" />
```

What’s this? Where did it come from?

```
switch (SessionContext.UserRole)
{
    case Role.Student:
        // Display the student name in the banner at the top of the page
        txtName.Text = string.Format("Welcome {0}", SessionContext.UserName);

        // Display the details for the current student
        GotoStudentProfile();
        break;

    case Role.Teacher:
        // Display the teacher name in the banner at the top of the page
        txtName.Text = string.Format("Welcome {0}", SessionContext.UserName);

        // Display the list of students for the teacher
        GotoStudentsPage();
        break;
}
```

How would you explain this?

```
// Parse the student name into the first name and last name by using a regular expression
// The firstname is the initial string up to the first space character.
// The lastname is the string after the space character
Match matchNames = Regex.Match(SessionContext.CurrentStudent, @"([^ ]+) ([^ ]+)");

if (matchNames.Success)
{
    string firstName = matchNames.Groups[1].Value; // Indexing in the Groups collection starts at 1, not 0
    string lastName = matchNames.Groups[2].Value;

    // Display the first name and last name in the TextBlock controls in the studentName StackPanel
    ((TextBlock)studentName.Children[0]).Text = firstName;
    ((TextBlock)studentName.Children[1]).Text = lastName;
}

// If the current user is a student, hide the Back button
// (only applicable to teachers who can use the Back button to return to the list of students)
if (SessionContext.UserRole == Role.Student)
{
    btnBack.Visibility = Visibility.Hidden;
}
else
{
    btnBack.Visibility = Visibility.Visible;
}
```

How would you explain this?

```
Button itemClicked = sender as Button;
if (itemClicked != null)
{
    // Find out which student was clicked - the Tag property of the button  contains the name
    string studentName = (string)itemClicked.Tag;
    if (StudentSelected != null)
    {
        // Raise the StudentSelected event (handled by MainWindow) to display the   details for this student
        StudentSelected(sender, new StudentEventArgs(studentName));
    }
}
```

How would you explain this?

4. In the code editor, click in the blank line in the **studentsPage_StudentSelected** method, and then type the following code:

```
SessionContext.CurrentStudent = e.Child;
GotoStudentProfile();
```

How would you explain this?

```
public struct Grade
{
    public int StudentID { get; set; }
    public string AssessmentDate { get; set; }
    public string SubjectName { get; set; }
    public string Assessment { get; set; }
    public string Comments { get; set; }
}
```

6. In the **Task List** window, double-click the **TODO: Exercise 2: Task 1b: Create the Student struct** task.
7. In the code editor, click in the blank line below the comment, and then type the following code:

```
public struct Student
{
    public int StudentID { get; set; }
    public string UserName { get; set; }
    public string Password { get; set; }
    public int TeacherID { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
}
```

8. In the **Task List** window, double-click the **TODO: Exercise 2: Task 1c: Create the Teacher struct** task.
9. In the code editor, click at the end of the comment line, press Enter, and then type the following code:

```
public struct Teacher
{
    public int TeacherID { get; set; }
    public string UserName { get; set; }
    public string Password { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Class { get; set; }
}
```

How would you explain this?

```
// Find the user in the list of possible users - first check whether the user is a  teacher
var teacher = (from Teacher t in DataSource.Teachers
               where String.Compare(t.UserName, username.Text) == 0 &&
                     String.Compare(t.Password, password.Password) == 0
               select t).FirstOrDefault();

// If the UserName of the user retrieved by using LINQ is non-empty then the user is a teacher
if (!String.IsNullOrEmpty(teacher.UserName))
{
    // Save the UserID and Role (teacher or student) and UserName in the global context
    SessionContext.UserID = teacher.TeacherID;
    SessionContext.UserRole = Role.Teacher;
    SessionContext.UserName = teacher.UserName;
    SessionContext.CurrentTeacher = teacher;

    // Raise the LogonSuccess event and finish
    LogonSuccess(this, null);
    return;
}
```

How would you explain this?

```
// Find all the grades for the student
ArrayList grades = new ArrayList();

foreach (Grade grade in DataSource.Grades)
{
    if (grade.StudentID == SessionContext.CurrentStudent.StudentID)
    {
        grades.Add(grade);
    }
}

// Display the grades in the studentGrades ItemsControl by using databinding
studentGrades.ItemsSource = grades;
```