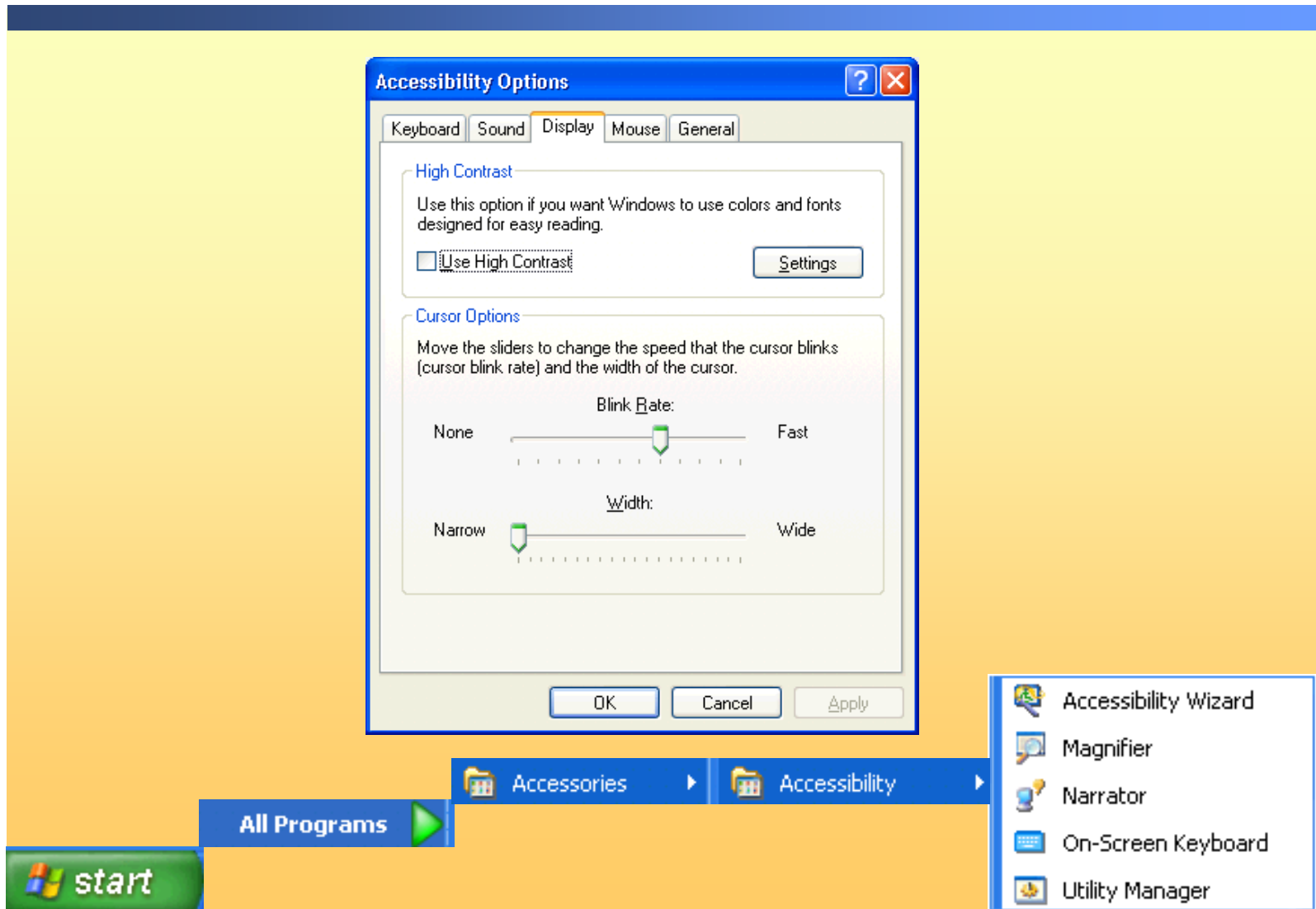


**Module 8:  
Enhancing the Usability  
of Applications**

# Overview

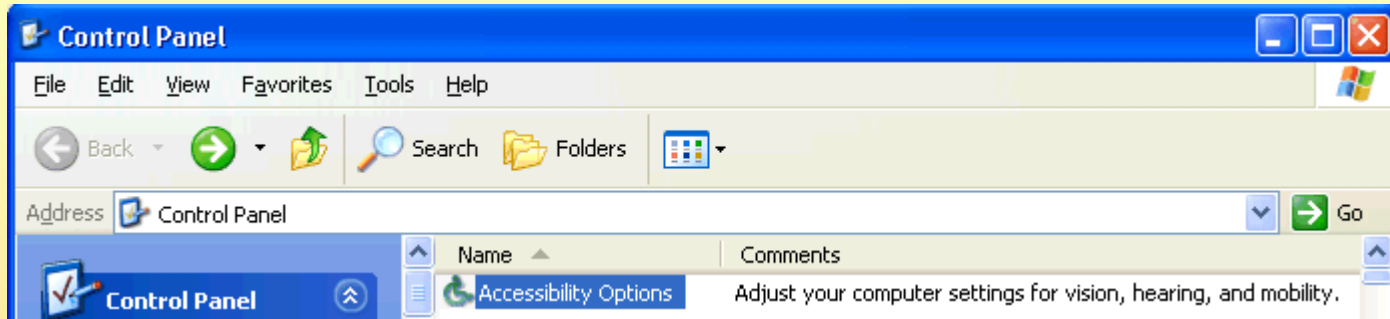
- Adding Accessibility Features
- Adding Help to an Application
- Localizing an Application

# Lesson: Adding Accessibility Features



# Accessibility Support in the .NET Framework

- Accessibility options



- Microsoft accessibility aids

- Narrator
- Magnifier
- On-Screen Keyboard

- Developers can provide accessibility support by setting properties on forms and controls in their applications

# How to Make Forms and Controls Accessible

		Control Property	Description
<b>1</b>	Set standard properties to values that support accessibility  Text, Font Size, Forecolor, Backcolor, BackgroundImage	AccessibleName	Briefly describes and identifies the object. Examples: button or menu item text
		AccessibleDescription	Provides greater context for low-vision or blind users
<b>2</b>	Set Accessibility properties  At design time or programmatically	AccessibleRole	Describes the use of the element in the user interface

```
Me.AppExitButton = New System.Windows.Forms.PushButton()  
Me.AppExitButton.Text = "E&xit"  
AppExitButton.AccessibleRole = _  
    System.Windows.Forms.AccessibleRole.PushButton  
AppExitButton.AccessibleName = "Exit"  
AppExitButton.AccessibleDescription = _  
    "Use this button to exit the application"  
Me.Controls.Add(Me.AppExitButton)
```

# How to Test Accessibility

**1** Build the application

**2** Turn on an accessibility aid, such as Narrator

**3** Run the application

# Practice: Adding Accessibility Support to an Application



In this practice, you will

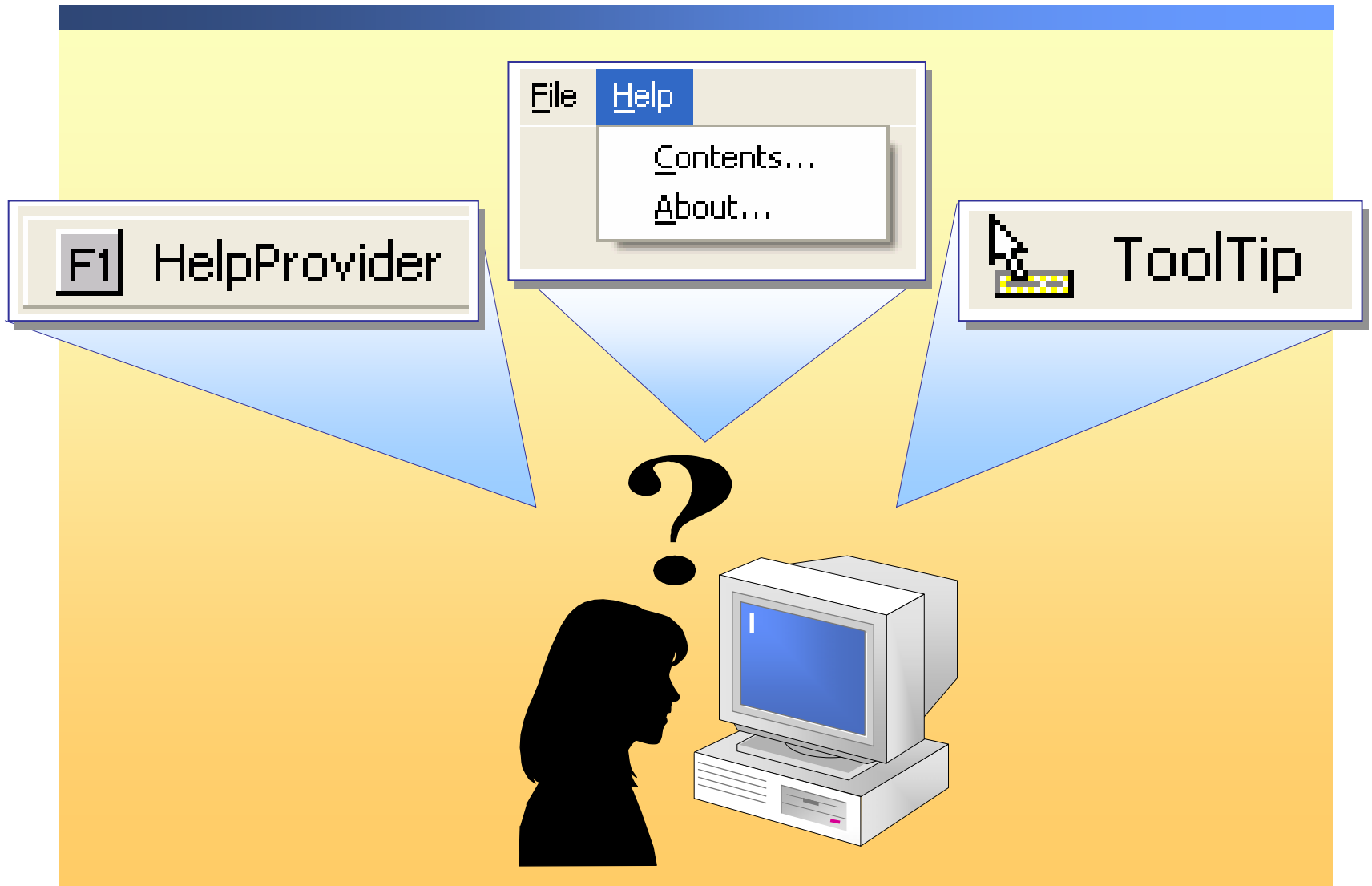
- Set the AccessibleName property for a control
- Enable the Narrator
- Run the application to see the results

Begin reviewing the objectives for  
this practice activity

10 min



# Lesson: Adding Help to an Application



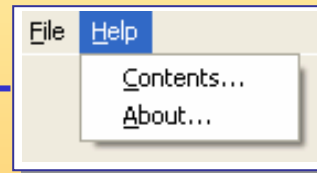
# Help in the .NET Framework

## ■ Context-sensitive Help

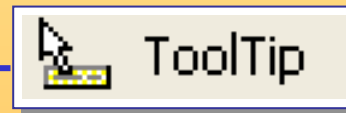
- HelpProvider Control
- HelpButton Property



## ■ Help menu support



## ■ The ToolTip control



# How to Add Context-Sensitive Help for Forms and Controls

**1** Add the HelpProvider control  
Set the HelpNamespace property

**2** Add a HelpButton to the form

**3** For each control that you want to add Help information set the following properties

- HelpKeyword
- HelpNavigator
- HelpString

**4** Build and test the application  
Give a control focus and press F1

# How to Link Help Topics to a Menu

**1**

Set the `HelpNamespace` property to point to a file or URL, such as `http://localhost/InternalBusinessAppHelp.htm`

**2**

Add a `MainMenu` control to the form

- Add Help menu item and subitems
- Implement **Help** menu subitems click event procedures to open the `HelpNamespace`

Parent of the  
Help dialog box

Path and name of  
Help file

`Help.ShowHelp (Me, HelpProvider.HelpNamespace)`



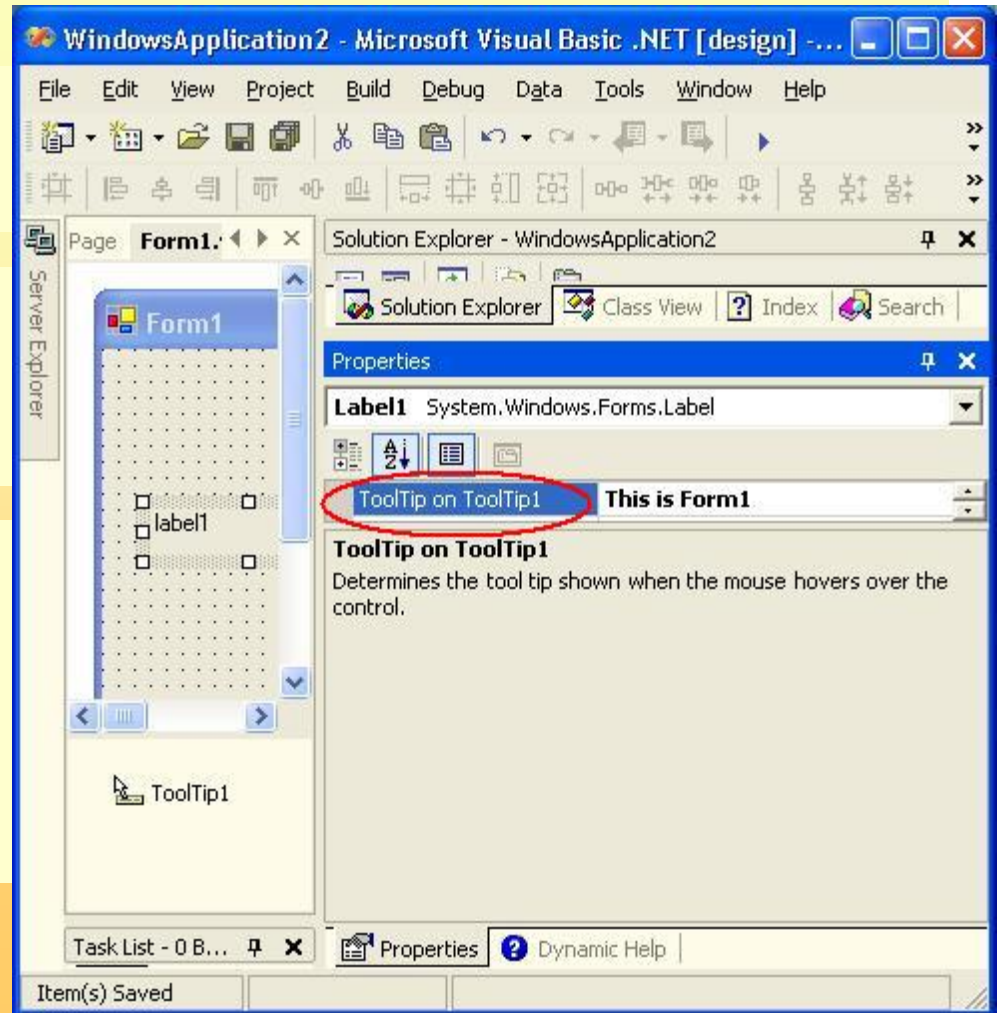
# How to Display Help with the ToolTip Control

**1** Add the ToolTip control

**2** Set the value for the ToolTip on ToolTip... property

**3** Build and test the application

Point to a control that has an associated ToolTip



# Practice: Adding ToolTips to an Application

In this practice, you will

- Add the **ToolTip** control to an application
- Set the **ToolTip** property for a control

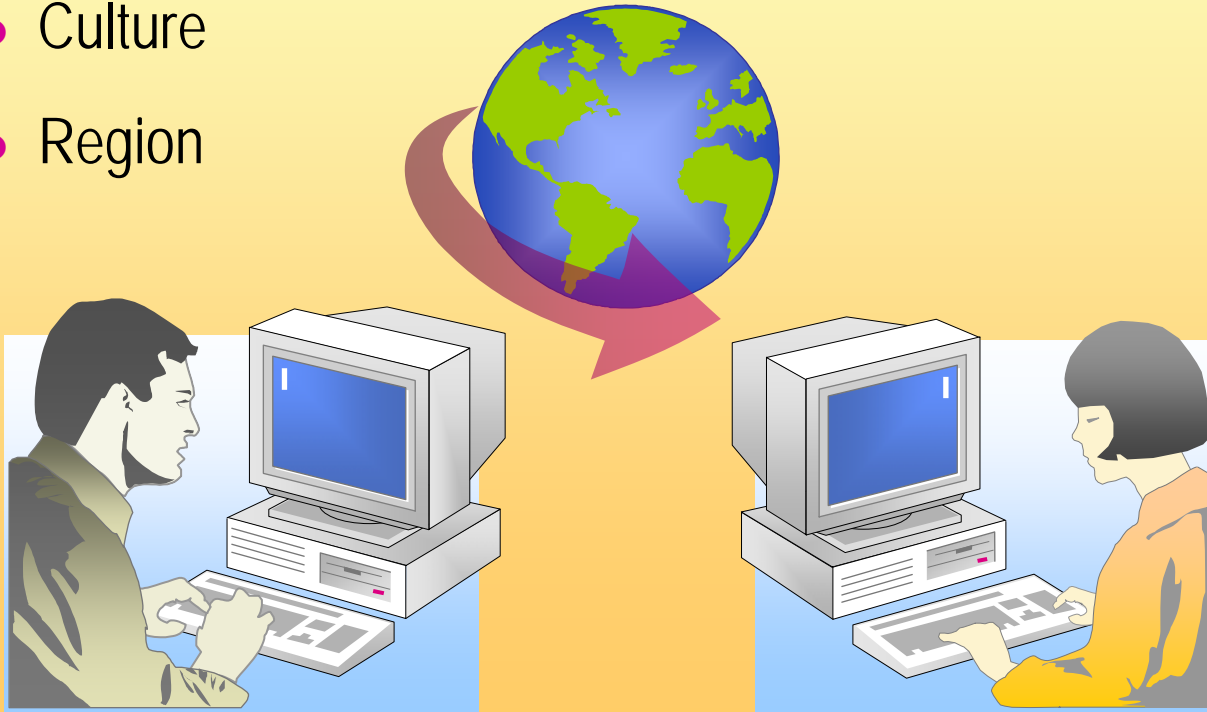
Begin reviewing the objectives for  
this practice activity

15 min



# Lesson: Localizing an Application

- Globalization
- Localization
  - Culture
  - Region



# Localization in the .NET Framework

- Localizing the user interface elements
- Localizing other resources
  - Strings
  - Bitmaps
  - Other objects, such as audio files

# How to Set Localization Properties

- 1** Create the default culture version of the form
- 2** Set the Localizable property of the form to True
- 3** Set the appropriate value for the Language property of the form
- 4** Modify the Text property values for the form and controls into the appropriate language
- 5** Resize and/or reposition each control as needed
- 6** Build the application

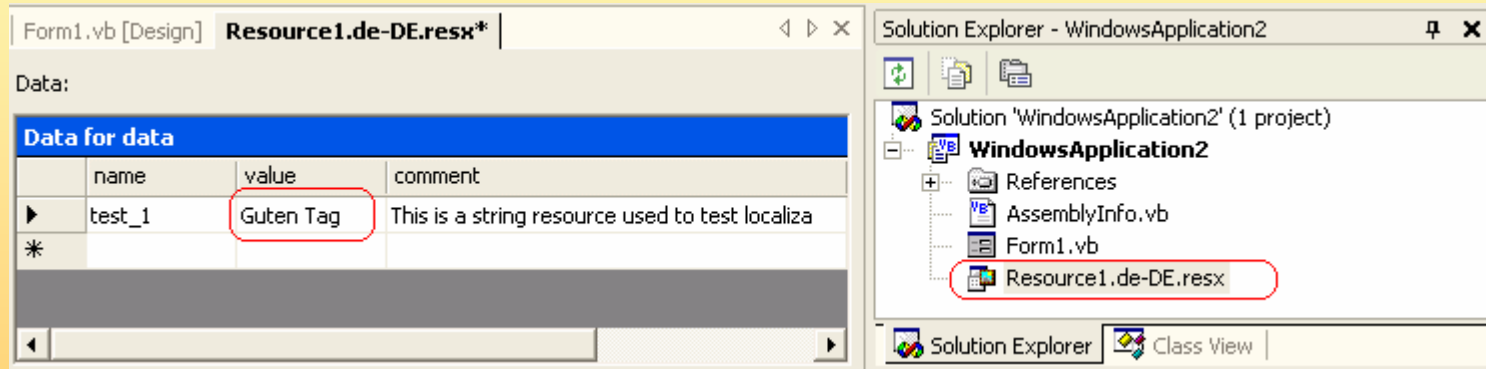
# How to Create Localized Resource Files

1

Open an existing project and add an assembly resource file for the appropriate culture

2

Add entries to the resource file with values in the appropriate language for the culture



The screenshot shows the Visual Studio IDE. On the left, the 'Data' window for 'Resource1.de-DE.resx\*' is open, displaying a table with the following content:

Data for data			
	name	value	comment
▶	test_1	Guten Tag	This is a string resource used to test localiza
*			

On the right, the 'Solution Explorer' shows the project 'WindowsApplication2' with the following files listed: References, AssemblyInfo.vb, Form1.vb, and Resource1.de-DE.resx. The 'Resource1.de-DE.resx' file is circled in red.

3

Save the file

4

Build the application

# How to Change the Locale

The user can change the regional and language options from Control Panel

- 1 Add code to an application to programmatically set the culture and UI Culture for an application to the new value
- 2 Add code to the application to use a resource manager to extract the elements from the resource file

```
Imports System.Globalization
Imports System.Resources
Imports System.Threading
```

```
...
Thread.CurrentThread.CurrentUICulture = _
    Thread.CurrentThread.CurrentCulture
```

Root name of the  
resource file

```
...
Dim rm As New ResourceManager ("MyNamespace.Resource1", _
    Assembly.GetExecutingAssembly())
```

Main assembly for  
the resources

```
MessageBox.Show(rm.GetString("test_1"))
```

# Practice: Localizing an Application



In this practice, you will

- Localize the user interface of an application
- Add localized string resources to an application

Begin reviewing the objectives for  
this practice activity

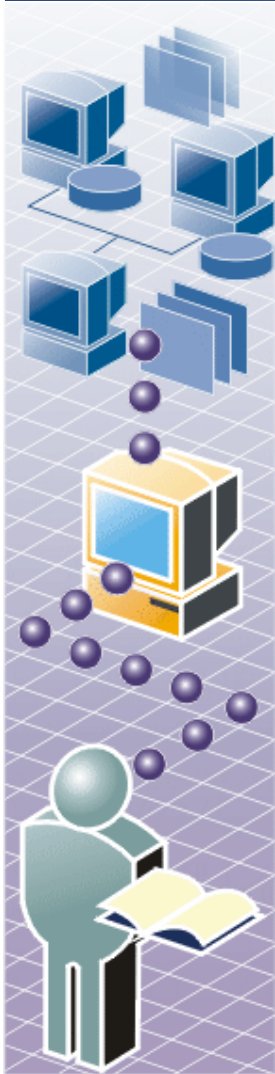
15 min



# Review

- Adding Accessibility Features
- Adding Help to an Application
- Localizing an Application

# Lab 8.1: Enhancing the Usability of an Application



- Exercise 1: Adding Support for Accessibility
- Exercise 2: Adding Help to an Application
- Exercise 3: Adding ToolTips to an Application
- Exercise 4: Localizing the User Interface of an Application
- Exercise 5: Localizing Resources in an Application

# Course Evaluation

