Telerik Academy

Validation Controls

Data Validation, Data Validators, Validation Groups

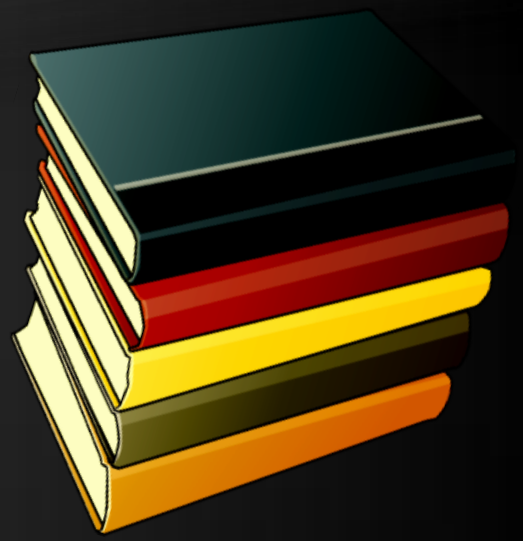
ASP.NET Web Forms

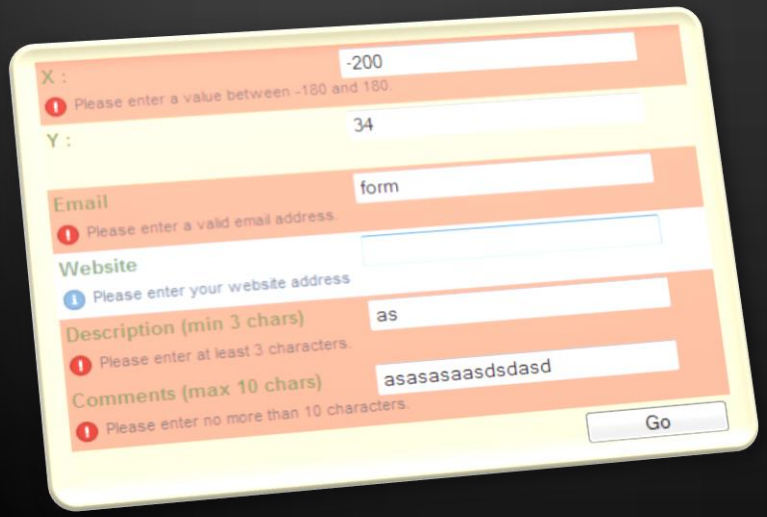
Telerik Software Academy

http://academy.telerik.com

Telerik Academy

Table of Contents

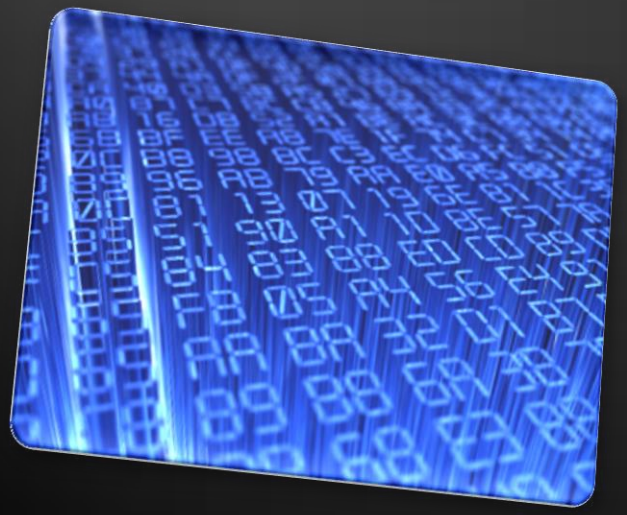




Telerik Academy

Data Validation

Validating Data in Web Applications



Data Validation

- When a user enters data in a form the programmer expects a certain data type
- Invalid input can break the application logic
- Input data validation means
  - Comparing a field to specific values or a range of acceptable values
  - Checking for required interdependencies between the fields

Telerik Academy

Validating Input Data (2)

- Predefined input formats usually follows some rules defining the acceptable set of values
  - Limitation about the length of a field
    - Allow empty value or not
  - Accepting digits only or not
  - A range of acceptable values
    - E.g. regular expression
  - Mathematical formula
    - E.g. field A > field B

Telerik Academy

Where Validation is Performed?

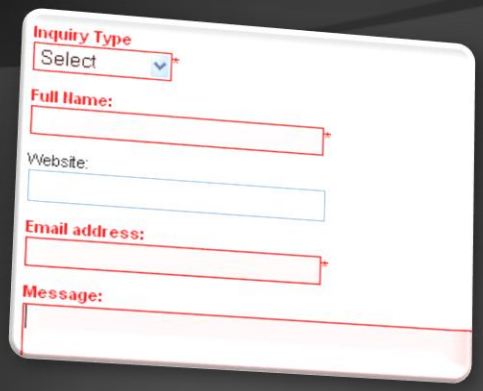
- The validation is always done on the server
  - When the client request is processed
- Additionally some validation rules could be performed at the client side
- In all cases the main part of the validation should be done at the server
  - A mischievous user can alter the validation on the client

Telerik Academy

Why Validation is Important?

- Data validation is important!
- Invalid data could cause many problems
  - Crashes of the application
  - Execution of SQL queries (SQL-injection)
  - Revealing protected data
- Example:
  - A user can enter very long text into a Web page
  - This could cause exceptions at the server side

Telerik Academy



Validation Controls

ASP.NET Controls for Data Validation

Telerik Academy

Validation Controls

- ASP.NET simplifies the process of validation by supplying a set of validation controls
- `RequiredFieldValidator`
  - Ensures the field is not left empty
- `CompareValidator`
  - Compares the input data from a control to the data in another control or to a constant value
  - Example:
    - When a password needs to be entered twice

Validation Controls (2)

- `RangeValidator`
  - Checks if the input is in given range of values
    - Between two constant values, e.g. [5..10]
    - Between two values of other controls
- `CustomValidator`
  - Provides a standard way for writing your own validation logic
  - The programmer needs to implement the logic

Telerik Academy

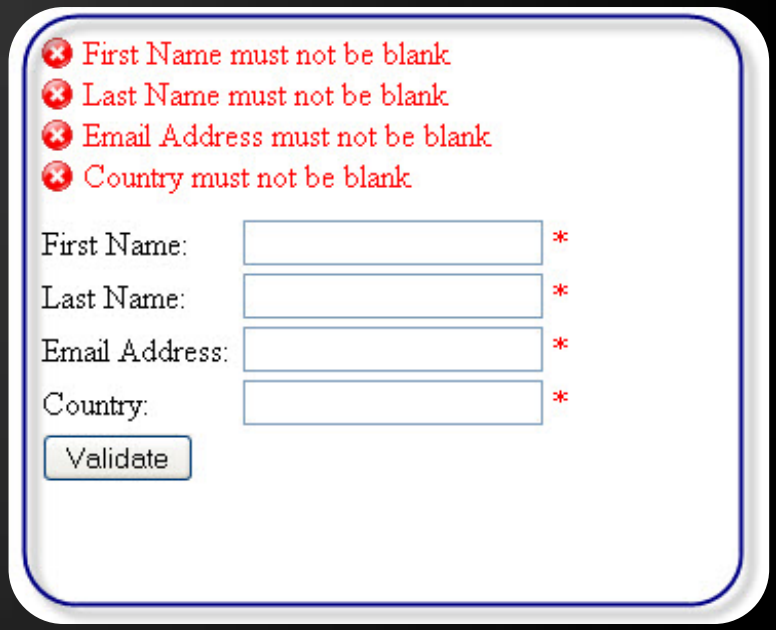
Validation Controls (3)

- `RegularExpressionValidator`
  - Checks the input against a regular expression pattern, e.g. `[1-9][0-9]+`
  - Examples:
    - Checking SSN, e-mail addresses, phones, etc.
  - Visual Studio provides many predefined patterns for the most common validations

Telerik Academy

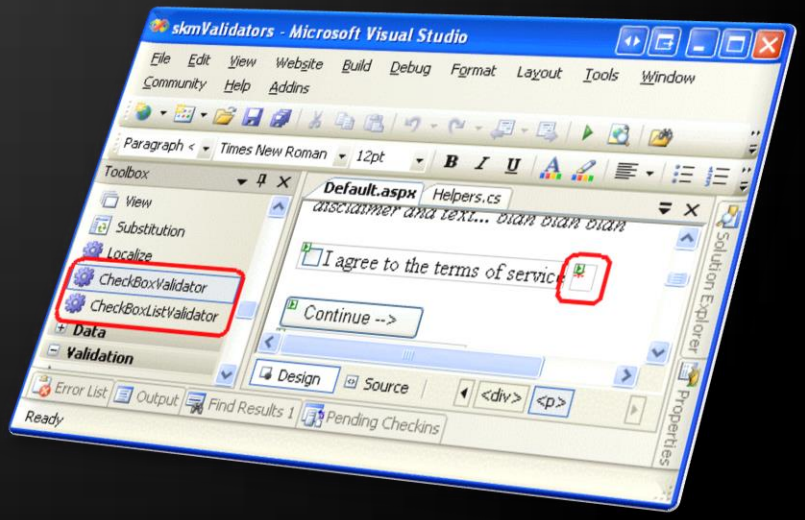
Validation Controls (4)

- `ValidationSummary`
  - A summary of all errors generated by the validation controls on the page
  - Provides an information about all errors in all page fields
  - Usually placed near the submit button



Telerik Academy

ASP.NET Validation Controls – Common Properties



Validator – Example

- After the validator is placed on the page you can assign its attributes (properties)
  - ControlToValidate, ErrorMessage, Text, …

```
<asp:type_of_validator
   ID="validator_id"
   runat="server"
   ControlToValidate="control_id"
   ErrorMessage="error_message_for_summary"
   Display="static|dynamic|none"
   Text="Text_to_display_by_input_control">
</asp:type_of_validator>
```

Common Properties

- `ControlToValidate`
  - Sets which control to be validated
  - One validator validates one particular field
- `IsValid`
  - Indicates if the input is valid
- `EnableClientScript`
  - Sets whether a client side validation is performed
  - Server side validation is always performed

The Type Property

- `RangeValidator` and `CompareValidator` have the `Type` property
  - Specifies the type of the data validated
  - The possible values are `String`, `Integer`, `Double`, `Date` and `Currency`
  - The specified range of values belongs to the same type, e.g. `[1.1.2009…31.12.2010]`
  - Visual Studio automatically sets this property when you specify `ControlToValidate`

ErrorMessage

- ErrorMessage
  - A message that is displayed if the input is found to be invalid
  - If the Text property is set it is shown instead
- When a ValidationSummary is defined
  - The value of the Text property is shown at the validation control's position
  - ErrorMessage is displayed in the ValidationSummary

Telerik Academy

The Display Property

- The `Display` property sets the layout of the error message
  - Affects messages which are displayed at the validation control's position
  - Messages shown in the `ValidationSummary` control are unaffected
  - Applied when the form use `FlowLayout` only
  - `Static` – fixed space is allocated for the error
  - `Dynamic` – space is allocated only if an error should be displayed, otherwise nothing is shown

Telerik Academy

CompareValidator – Example

```
<asp:TextBox ID="TextBoxPassword" runat="server" />
<asp:TextBox ID="TextBoxRepeatPass" runat="server" />
<asp:CompareValidator ID="CompareValidatorPassword"
  runat="server" ControlToCompare="TextBoxPassword"
  ControlToValidate="TextBoxRepeatPass"
  ValueToCompare="Text" ForeColor="Red"
  ErrorMessage="Password doesn't match!" />
<asp:Button ID="BtnSubmit" runat="server"
  Text="Submit" />
```

* Note: due to a design flaw in the `CompareValidator`, when the second control is empty or contains whitespace only, the validator is not executed at all!

Telerik Academy

The Almost Working CompareValidator

Live Demo

Telerik Academy

Validation Controls Used in a Combination

```
<asp:TextBox ID="TextBoxEmail" runat="server" />
<asp:RequiredFieldValidator
  ID="RequiredFieldValidatorEmail"
  runat="server" ForeColor="Red" Display="Dynamic"
  ErrorMessage="An email address is required!"
  ControlToValidate="TextBoxEmail" />
<asp:RegularExpressionValidator
  ID="RegularExpressionValidatorEmail"
  runat="server" ForeColor="Red" Display="Dynamic"
  ErrorMessage="Email address is incorrect!"
  ControlToValidate="TextBoxEmail"
  ValidationExpression="[a-zA-Z][a-zA-Z0-9\-\.]+[a-zA-
Z]@[a-zA-Z][a-zA-Z0-9\-\.]+[a-zA-Z]+\.[a-zA-Z]{2,4}" />
<asp:Button ID="ButtonSubmit" runat="server" />
```

Telerik Academy

A Web Page Validation

- The `Page.IsValid` property is evaluated as a logical "AND" of all validation controls on the page

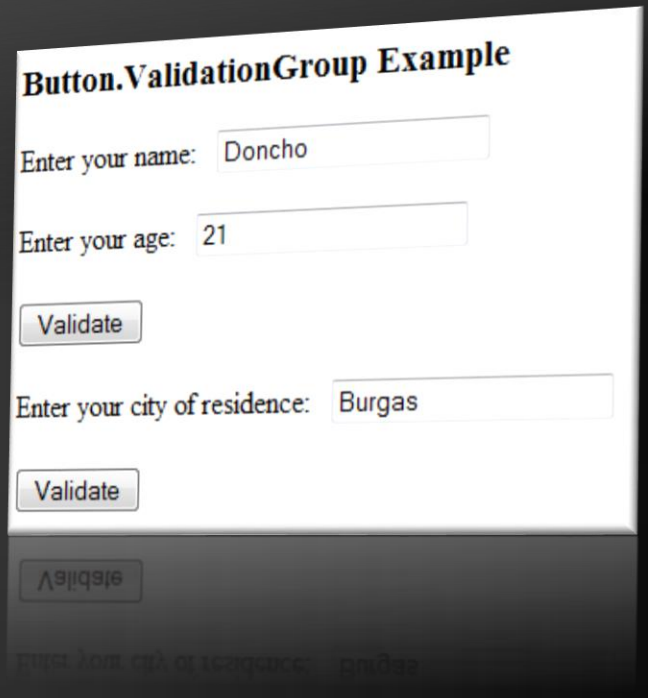
```
private void ButtonSubmit_Click(
  object sender, EventArgs e)
{
  if (Page.IsValid)
  {
     LabelMessage.Text = "The page is valid!";
     // Perform some logic here
  }
  // An else clause is unneeded – the page will
  // be returned to the user and all error
  // messages will be displayed
}
```

Telerik Academy

Page Validation

Live Demo

Telerik Academy



Validation Groups

Validation Group

Telerik Academy

The ValidationGroup Property

- You can group validation controls
- Scenario
  - A PostBack is initiated from a control (button) that has a validation group set
  - Only validation controls of that group are validated
- You can explicitly cause validation from code-behind by calling the `Page.Validate("groupName")` method

Telerik Academy

The ValidationGroup Property (2)

- `Page.IsValid` reflects the validity of all groups validated so far
- This includes
  - Validation as a result of PostBack
  - Validation as a result of call to `Validate()`

Telerik Academy

Validating Dynamic Forms

Live Demo

Telerik Academy

Validation Controls

Questions?

http://academy.telerik.com

Exercises

1. Create a form to register users with fields for preferred user name, password, repeat password, first name, last name, age, email, local address, phone and an "I accept" option. All fields are required. Valid age is between 18 and 81. Display error messages in a `ValidationSummary`. Use a regular expression for the email and phone fields.
2. Separate the fields in groups and validate them using Validation Groups. The Validation Groups should be at least three – Logon Info, Personal Info, Address Info.

Exercises (2)

3. * Add a radio button to choose the gender (male / female). If the user is male, dynamically display a list of check boxes for choosing his favourite cars (e.g. BMW, Toyota, etc.). If the user is female display a drop-down list to allow her select her favourite coffee (e.g. Lavazza, New Brazil, etc.). Note that selecting a coffee is optional for the female users. Implement this by server PostBacks.
4. * Implement the previous with client-side JavaScript.