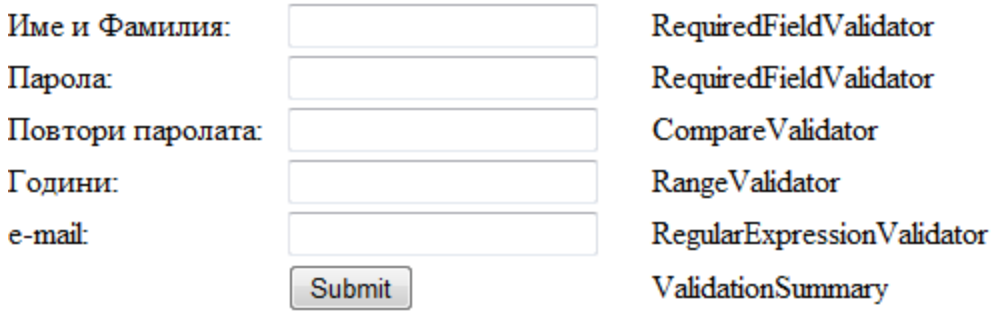
Упражнение 3 - Контроли за валидиране на входната информация

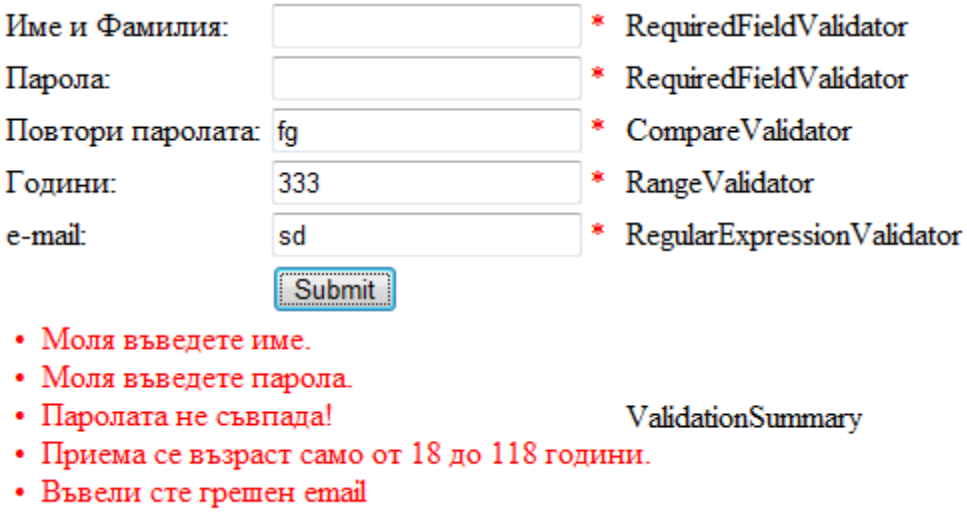
Задача

1. Създайте нова уеб страница, чрез File -> New -> Web Site (C# language).
2. Добавете необходимите контроли за регистрация на потребител. Нека страницата ви изглежда както е показано на картинката.



3. След всяка една от посочените контроли поставете съответния валидатор. Ще ги намерите в ToolBox, в раздел Validation.
4. За всеки един от валидаторите определете контролата, която валидират (ControlToValidate property).
5. За CompareValidator посочете и втори контрол за сравнение.
6. Нека се проверява възрсаст от 18 до 118 години.
7. За проверка на валидност на email адрес, изберете подходяща проверка на RegularExpressionValidator.
8. Направете свойството Text на всеки от валидаторите звездичка (*). Така ще се посочва къде има проблем, а пълната информация трябва да изведем систематизирана в ValidationSummary. Това става автоматично като се извеждат ErrorMessage полетата на всеки сработил валидатор. Променете тези съобщения,

така че потребителя да е наясно какво трябва да коригира.



Съдържание:

Фреймуъркът включва набор от сървърни контроли за валидация, които предоставят лесен за употреба, но мощен начин за проверка на входните форми за грешки и, ако е необходимо, извеждане на съобщения към потребителя. Контролите за валидация се добавят към уеб страницата по същия начин както и другите контроли. Съществуват контроли за специфични видове валидация, като проверка на диапазон или съвпадение на модела, както и **RequiredFieldValidator**, който гарантира, че потребителят не е пропуснал да попълни съответното поле. Към един входен контрол могат да се прикачат повече от една. Например, едно поле може да се определи така, че попълването му да е задължително и че стойността трябва да принадлежи на определен обхват.

Контролите за валидация работят с ограничен набор от HTML и Web server контроли. За всеки контрол има специфично свойство, което съдържа информацията, която трябва да бъде валидирана. Следващата таблица изброява контроли за въвеждане, които могат да бъдат валидирани.

Control	**Validation Property**
HtmlInputText	Value

HtmlTextArea	Value
HtmlSelect	Value
HtmlInputFile	Value
TextBox	Text
ListBox	SelectedItem.Value
DropDownList	SelectedItem.Value
RadioButtonList	SelectedItem.Value
FileUpload	FileName

Видове контроли за валидация

Най-простата форма на проверка е задължителното поле. Ако потребителят въведе някаква стойност в полето, то е валидно. Ако всички полета в страницата са валидни, страницата също е валидна. Следващият пример илюстрира употребата на **RequiredFieldValidator**.

Пример: RequiredFieldValidator

Съществуват и други специфични видове контроли за валидиране като диапазон проверка или модел на съвпадение. В долната таблица са описани контролите.

Име на контрол	**Описание**
RequiredFieldValidator	Гарантира, че потребителят не пропуска поле.
CompareValidator	Сравнява попълненото от потребителя с постоянна стойност или стойността на друг контрол с помощта на оператор за сравнение (по-малко, равно, по-голямо и т.н.).
RangeValidator	Проверява, дали въведеното от потребителя е между определена долна и горна граници. Обхватите могат да са двойки от числа, букви от азбуките или дати. Границите могат да бъдат описани като константи.
RegularExpressionValidator	Проверява дали въведеното съвпада с модела определен от регулярен израз. Този тип проверка ви позволява да проверявате за предсказуема последователност от символи, като тези в социално-осигурителни номера, имейл адреси, телефонни номера, пощенски кодове и т.н.
CustomValidator	Проверява въведеното от потребителя, използвайки програмирана от потребителя логика. Този тип проверка ви позволява да проверявате за стойности, получени по време на изпълнението.
ValidationSummary	Показва в обобщен вид грешките за всички валидатори на

една страница.

Валидация при страната на клиента

Контролите за валидация винаги изпълняват проверката в сървърен код. Въпреки това, ако потребителят работи с браузър, поддържащ DHTML, контролите за валидация могат също да извършват проверка посредством клиентски скрипт. При проверката при клиента, всички грешки се откриват, когато формулярът се подава към сървъра. Ако някой от валидаторите установи грешка, изпращането на формата към сървъра се прекратява и се показва **Text** полето на валидатора. Това позволява на потребителя да коригира входните данни преди да подаде формуляра към сървъра. Стойностите на полетата са ревалидират веднага след като съответното поле загуби фокус, като по този начин предоставя на потребителя богата и интерактивна проверка.

Имайте предвид, че валидация винаги се извършва на сървъра, дори и ако вече е направена такава при клиента. Това помага да се предотврати възможността потребителите да заобиколят валидацията чрез представяне за друг потребител или предварително одобрени транзакции.

Валидацията при клиента е активирана по подразбиране. Ако клиентът е в състояние, uplevel проверката ще се извърши автоматично. За да се забрани на валидацията при клиента, стойността на свойството **ClientTarget** трябва да бъде "Downlevel" ("Uplevel" включва клиентската проверка). Другия начин е да се зададе стойност „false” на свойството **EnableClientScript**, за да забраните клиентската проверка за определен контрол.

Пример: Валидация при страната на клиента

Показване на грешките при валидация

Когато потребителските данни се обработват (например, когато формата се изпраща), Web Forms framework предава данните към свързаните валидационен контрол или контроли. Контролите за валидация тестват въведените от потребителя данни и определят свойство, чиято стойност да посочи дали е преминат теста за проверка. След като всички проверки са били обработени се определя стойността на **IsValid** свойството на страницата: ако някоя от проверките показва грешка, цялата страница се определя като невалидна.

Ако контрол за валидация показва грешка, съобщение за грешка може да се изведе на страницата от самия контрол или в **ValidationSummary** контрола на друго място в страницата. **ValidationSummary** контролът се показва, когато IsValid свойството на страницата е false. Той проверява всеки контрол за валидация върху страницата и събира текстовите им съобщения. Следващият пример илюстрира извеждането на грешки с ValidationSummary контрол.

Пример: Validation Summary

Работа с CompareValidator

CompareValidator сървърна контрола сравнява стойностите на две контроли. **CompareValidator** използва три ключови свойства за да изпълни валидацията си. **ControlToValidate** и **ControlToCompare** съдържат стойностите за сравнение. **Operator** определя типа на сравнение, което ще се извърши – например, равно или не равно. **CompareValidator** осъществява проверката чрез оценка на тези свойства при следния израз:

```
( ControlToValidate ControlToCompare )
```

Ако в резултат се получи true, резултатът е правилна валидация. **ValueToCompare** стойността може да се настрои да се сравнява със статична стойност вместо ControlToCompare.

CompareValidator контролът може да бъде използван за проверка на типа данни. Например, ако се изисква данни за рождена дата в регистрационна страница, **CompareValidator** контролът може да бъде използван за да се осигури това, че датата е в правилен формат преди да бъде изпратена към базата данни.

Пример: CompareValidator

Работа с RangeValidator

RangeValidator контролът проверява дали входната стойност е в рамките на даден интервал. **RangeValidator** използва три ключови свойства за да изпълни проверката си. **ControlToValidate** съдържа стойността за проверка. **MinimumValue** и **MaximumValue** определят долната и горната граница на валидния интервал.

Пример: RangeValidator

Извършване на собствена верификация

CustomValidator контролът извиква дефинирана от потребителя функция, която да извърши проверки извън възможностите на стандартните валидатори. Потребителската пункция може да се изпъни на сървъра или в клиентски скрипт, като JScript или VBScript. При валидация при клиента, името на функцията трябва да бъде посочено в **ClientValidationFunction** свойството. Собствената функция трябва да име вида

`function myvalidator(source, arguments)`. Като **source** е обект **CustomValidator** при клиента, а **arguments** е обект с две свойства, **Value** и **IsValid**. Свойството **Value** е стойността за проверка, а **IsValid** е булева стойност отчитаща резултата от валидацията.

При валидация при сървъра, собствената проверка се поставя в делегата **OnServerValidate** на валидатора.

Пример: Custom Validator

Типична валидационна форма

Следващият пример демонстрира типична регистрационна форма, използваща различни контроли за валидация.

Пример: Validation Form