Implementing Validation

By: Mosh Hamedani

Adding Validation

Decorate properties of your model with data annotations. Then, in the controller:

```
if (!ModelState.IsValid)
    return View(…);
```

And in the view:

```
@Html.ValidationMessageFor(m => m.Name)
```

Styling Validation Errors

In site.css:

```
.input-validation-error {
    color: red;
}

.field-validation-error {
    border: 2px solid red;
}
```

Implementing Validation

By: Mosh Hamedani

Data Annotations

- [Required]
- [StringLength(255)]
- [Range(1, 10)]
- [Compare("OtherProperty")]
- [Phone]
- [EmailAddress]
- [Url]
- [RegularExpression("…")]

Custom Validation

```
public class Min18IfAMember : ValidationAttribute
{
     protected override ValidationResult IsValid(object value,
ValidationContext validationContext)
     {
          …
          if (valid) return ValidationResult.Success;
          else return new ValidationResult("error message");
     }
}
```

Implementing Validation

By: Mosh Hamedani

Validation Summary

```
@Html.ValidationSummary(true, “Please fix the following errors”);
```

Client-side Validation

```
@section scripts {
     @Scripts.Render(“~/bundles/jqueryval”)
}
```

Anti-forgery Tokens

In the view:

```
@Html.AntiForgeryToken()
```

In the controller:

```
[ValidateAntiForgeryToken]
public ActionResult Save() { }
```