RoR vs ASP.NET MVC

**stylesheet/javascript links**

**Asp.Net MVC**

<link rel="stylesheet" type="text/css"   
 href="<%=Url.Content("~/public/stylesheets/style.css") %>"  
/>  
<script type="text/javascript"   
 src="<%=Url.Content("~/public/javascripts/jquery.js") %>"  
></ script>

**RoR**

<%= stylesheet\_link\_tag 'style' %>  
<%= javascript\_include\_tag 'jquery' %>

**hyperlinks**

**Asp.Net MVC**

<%= Html.ActionLink("View Widgets", "Index", "Widget") %>  
<%= Html.ActionLink("Widget 5", "Show", "Widget",  
  new {id = 5}) %>

**RoR**

<%= link\_to 'View Widgets', widgets\_path %>  
<%= link\_to 'Widget 5', widget\_path(5) %>

**forms**

**Asp.Net MVC**

<% using(Html.BeginForm()) { %>  
 <%= Html.LabelFor(x => x.Name) %>  
 <%= Html.TextBoxFor(x => x.Name) %>  
 <input type="submit" value="submit" />  
<% } >

**RoR**

<% form\_for(@widget) do |f| %>  
 <%= f.label :name %>  
 <%= f.text\_field :name %>  
 <%= f.submit 'submit' %>  
<% end %>

**Packaging**

**Asp.Net MVC**

Nugget

**RoR**

**Gems**

**Packaging**

**Available in both of them**

**rendering partials**

**Asp.Net MVC**

<%= Html.Partial("form") %>

**RoR**

<%= render :partial => 'form' %>

**Looping**

**Asp.Net MVC**

<% foreach(var widget in Model.Widgets) { %>  
  <%= widget.Name %>  
<% } %>

**RoR**

<% for widget in @widgets %>  
  <%= widget.name %>  
<% end %>

**validation**

**Asp.Net MVC**

public partial class Widget

{

  [Required(ErrorMessage = "Name is Required")]  
  [StringLength(30, ErrorMessage = "Max 30 chars"

  public string Name { get; set; }

}

**RoR**

class Widget < ActiveRecord::Base  
  validates\_presence\_of :name  
  validates\_length\_of :name, :in => 5..30  
end

**orm data access**

**Asp.Net MVC (Linq-to-Sql)**

var db = new MyDataContext();  
var widgets = db.Widgets;  
var widget = db.Widgets.SingleOrDefault(x => x.Id == 5);  
var components = widgets.Components.OrderBy(x => x.name);

**RoR (ActiveRecord)**

@widgets = Widget.all  
@widget = Widget.find(5)  
@components = @widget.components.sort\_by{|w| w.name}

**saving a model**

**Asp.Net MVC (Linq-to-Sql)**

public ActionResult Create(Widget widget)  
{  
  db.InsertOnSubmit(widget);  
  db.SubmitChanges();  
}

**RoR (ActiveRecord)**

def create  
  @widget = Widget.new(params[:widget])  
  @widget.save  
end

**helper methods**

**Asp.Net MVC**

// define it:  
public static string DivThis(this HtmlHelper html, string text)  
{  
  return "<div>" + text + "</div>";  
}  
  
<!-- call it: -->  
<= Html.DivThis("hello") %>

**RoR**

# define it  
def div\_this(text)  
  '<div>' + text + '</div>'  
end  
  
<!-- call it: -->  
<%= div\_this('hello') %>

**temporary data**

**Asp.Net MVC**

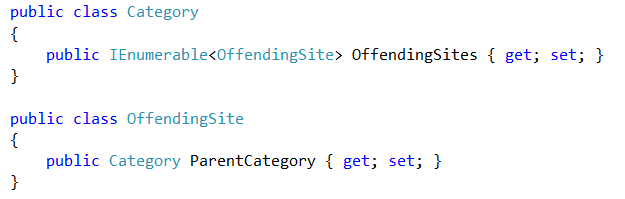
// set it  
TempData["someKey"] = "hello"  
  
<!-- use it -->  
<%= TempData["someKey"].ToString() %>

**RoR**

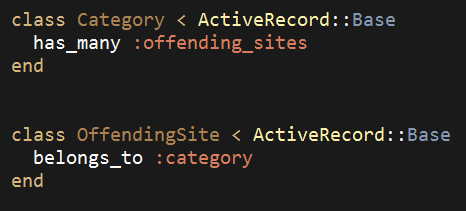
# set it  
flash[:some\_key] = 'hello'  
  
<!-- use it -->  
<%= flash[:some\_key] %>

**Model: associations**

**Asp.NET MVC**

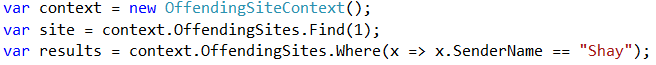


RoR(Active Record)

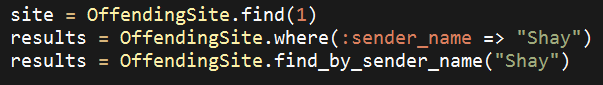


Model: data access

ASP.NET MVC



RoR



**Migrations**

Natively available in RoR

There is some EntityFramework feature in asp.net mvc

RoR dev. environment – completely free

ASP.NET MVC dev. environment – not almost free

**View Helpers**

RoR – helpers

ASP.NET MVC – helpers, editor view/edit templates

**Deployment/Scripting**

RoR – rake

ASP.NET MVC - custom