CODER FOUNDRY

MVC

Blog Project: Using Slugs

Slugs are a useful means of providing both a smaller introductory display piece for a blog entry, as well as an optional navigation property for an MVC blog application.

Create a class named StringUtilites. This can go in any folder in your application. One strategy is to create a new folder, “Helpers”, for any extra helper classes. Note that the static method URLFriendly accepts a parameter ‘title’ and returns a hyphenated string minus any special characters.

```
public class StringUtilities
    {

        /// <summary>
        /// Produces optional, URL-friendly version of a title, "like-this-one".
        /// hand-tuned for speed, reflects performance refactoring contributed
        /// by John Gietzen (user otac0n)
        /// </summary>
        public static string URLFriendly(string title)
        {
            if (title == null) return "";
            const int maxlen = 80;
            int len = title.Length;
            bool prevdash = false;
            var sb = new StringBuilder(len);
            char c;
            for (int i = 0; i < len; i++)
            {
                c = title[i];
                if ((c >= 'a' && c <= 'z') || (c >= '0' && c <= '9'))
                {
                    sb.Append(c);
                    prevdash = false;
                }
                else if (c >= 'A' && c <= 'Z')
                {
                    // tricky way to convert to lowercase
                    sb.Append((char)(c | 32));
                    prevdash = false;
                }
                else if (c == ' ' || c == ',' || c == '.' || c == '/' ||
                c == '\\' || c == '-' || c == '_' || c == '=')
        {
            if (!prevdash && sb.Length > 0)
            {
                sb.Append('-');
                prevdash = true;
            }
        }
        else if (c == '#')
        {
            if (i > 0)
                if (title[i - 1] == 'C' || title[i - 1] == 'F')
                    sb.Append("-sharp");
        }
        else if (c == '+')
        {
            sb.Append("-plus");
        }
        else if ((int)c >= 128)
        {
            int prevlen = sb.Length;
            sb.Append(RemapInternationalCharToAscii(c));
            if (prevlen != sb.Length) prevdash = false;
        }
        if (sb.Length == maxlen) break;
    }
    if (prevdash)
        return sb.ToString().Substring(0, sb.Length - 1);
    else
        return sb.ToString();
}
public static string RemapInternationalCharToAscii(char c)
{
    string s = c.ToString().ToLowerInvariant();
    if ("àåáâäãåą".Contains(s))
    {
        return "a";
    }
    else if ("èéêëę".Contains(s))
    {
        return "e";
    }
    else if ("ìíîïı".Contains(s))
    {
        return "i";
    }
    else if ("òóôõöøőð".Contains(s))
    {
        return "o";
    }
    else if ("ùúûüŭů".Contains(s))
    {
        return "u";
    }
    else if ("çćčĉ".Contains(s))
    {
        return "c";
    }
    else if ("żźž".Contains(s))
```

```
        {
            return "z";
        }
        else if ("śşšŝ".Contains(s))
        {
            return "s";
        }
        else if ("ñń".Contains(s))
        {
            return "n";
        }
        else if ("ýÿ".Contains(s))
        {
            return "y";
        }
        else if ("ğĝ".Contains(s))
        {
            return "g";
        }
        else if (c == 'ř')
        {
            return "r";
        }
        else if (c == 'ł')
        {
            return "l";
        }
        else if (c == 'đ')
        {
            return "d";
        }
        else if (c == 'ß')
        {
            return "ss";
        }
        else if (c == 'Þ')
        {
            return "th";
        }
        else if (c == 'ĥ')
        {
            return "h";
        }
        else if (c == 'ĵ')
        {
            return "j";
        }
        else
        {
            return "";
        }
    }

}
```

2. In the Posts Controller - Create action (HttpPost), change the following code to create the slug and check for errors.

```
public ActionResult Create([Bind(Include = "Id,Title,Body,MediaURL,Published")] BlogPost
blogPost)
{
    if (ModelState.IsValid)
    {
        var Slug = StringUtilities.URLFriendly(blogPost.Title);
        if (String.IsNullOrWhiteSpace(Slug))
        {
            ModelState.AddModelError("Title", "Invalid title");
            return View(blogPost);
        }
        if(db.Posts.Any(p => p.Slug == Slug))
        {
            ModelState.AddModelError("Title", "The title must be unique");
            return View(blogPost);
        }

        blogPost.Slug = Slug;
        blogPost.Created = DateTimeOffset.Now;
        db.Posts.Add(blogPost);
        db.SaveChanges();
        return RedirectToAction("Index");
    }

    return View(blogPost);
}
```

3. In the App_Start folder, open the RouteConfig.cs file and add this route above the default route.

```
routes.MapRoute(
    name: "NewSlug",
    url: "Blog/{slug}",
    defaults: new {
        controller = "BlogPosts", action="Details",
        slug = UrlParameter.Optional
    });
```

4. To view the Details page of a post, the Details action in the Posts controller should possibly look like this.

```
public ActionResult Details(string Slug)
{
    if (String.IsNullOrWhiteSpace(Slug))
    {
        return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
    }
    BlogPost blogPost = db.Posts.FirstOrDefault(p => p.Slug == Slug);
    if (blogPost == null)
    {
        return HttpNotFound();
    }
    return View(blogPost);
}
```

5. This is how you can create an action link to this details action.

```
@Html.ActionLink("Details", "Details", new { slug = item.Slug })
```