Areas in ASP.NET MVC

Dhananjay Kumar / Wednesday, November 25, 2015

What is an Area in ASP.NET MVC?

Areas are some of the most important components of ASP.NET MVC projects. The main use of Areas are to physically partition web project in separate units. If you look into an ASP.NET MVC project, logical components like Model, Controller, and the View are kept physically in different folders, and ASP.NET MVC uses naming conventions to create the relationship between these components. Problems start when you have a relatively big application to implement. For instance, if you are implementing an E-Commerce application with multiple business units, such as Checkout, Billing, and Search etc. Each of these units have their own logical components views, controllers, and models. In this scenario, you can use ASP.NET MVC Areas to physically partition the business components in the same project.

In short, an area can be defined as: **Smaller functional units in an ASP.NET MVC project with its own set of controllers, views, and models.**

> Areas are smaller functional units in an ASP.NET MVC project. It allows us to divide the big web application project into smaller functional units. Each Area has its own controllers, models, and views.

A single MVC application may have any number of Areas. Some of the characteristics of Areas are:

- An MVC application can have any number of Areas.
- Each Area has its own controllers, models, and views.

CHAT

- Physically, Areas are put under separate folders.
- Areas are useful in managing big web applications.
- A web application project can also use Areas from different projects.
- Using Areas, multiple developers can work on the same web application project.

Creating Areas

Creating Areas in an MVC project is very simple. Simply right click on the **project-> Add->Area** as shown in the image below:
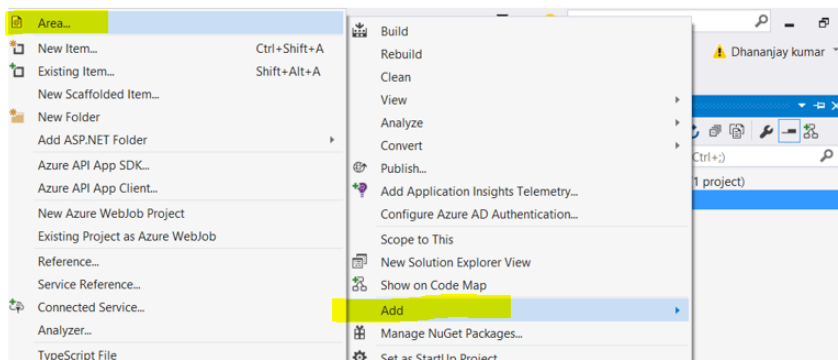
Here you will be asked to provide the Area name. In this example, let's name the Area "Blogs" and click on Add.
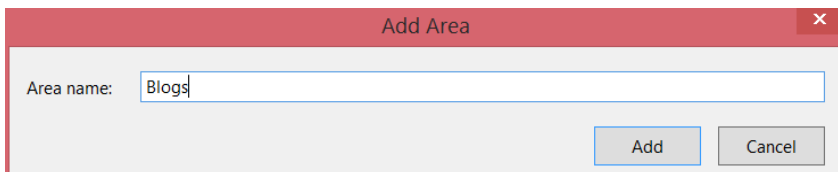

Let us stop for a moment here and explore the project. We will find a folder **Areas** has been added, and inside the Areas folder, we will find a subfolder with the name Blogs, which is the area we just created. Inside the Blogs subfolder, we will find folders for MVC components Controllers, Views, and Models.
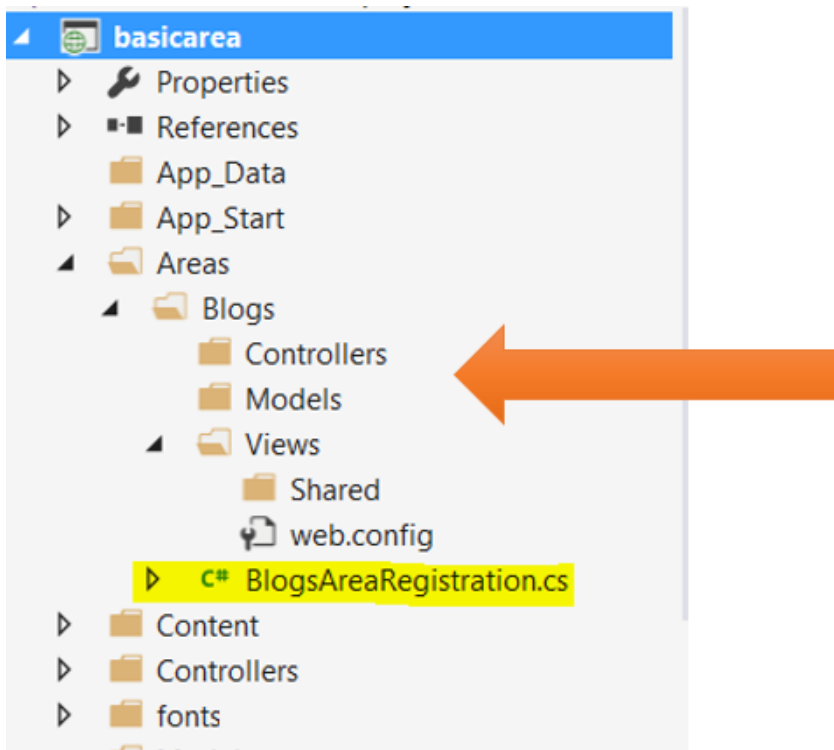
In the Area Blogs folder we will find a class BlogAreaRegistration.cs. In this class, routes for the Blog Area have been registered.

```
0 references
public override void RegisterArea(AreaRegistrationContext context)
{
    context.MapRoute(
        "Blogs_default",
        "Blogs/{controller}/{action}/{id}",
        new { action = "Index", id = UrlParameter.Optional }
    );
}
```
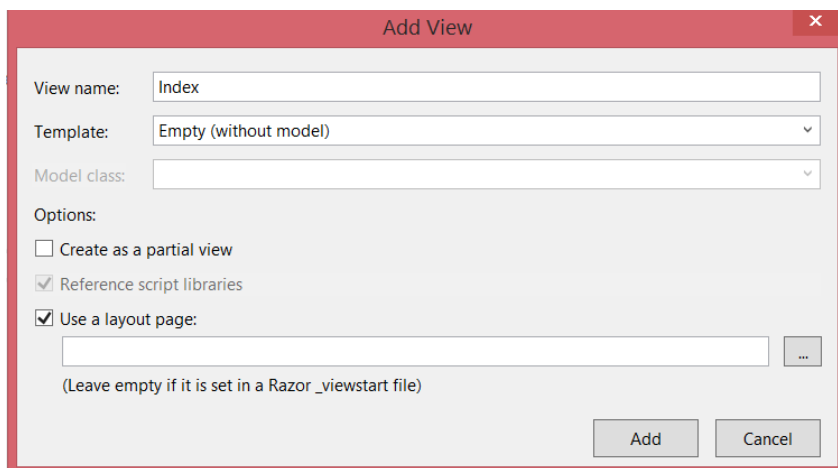
Now we can add Controllers, Models, and the Views in the Area in the same way we add them normally in an MVC project. For example, to add a controller, let's right click on the controller folder in the Blogs folder and click on Add->Controller. So let us say we have added a HomeController in the Blogs controller. You will find the added controller in the project as shown in the image below:
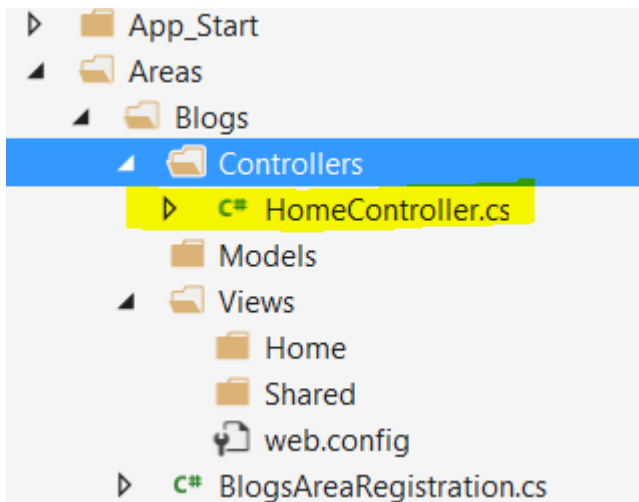
You will find that the HomeController has a method called
Index. To create a view, right click on the Index action and
select Add View as shown in the image below:
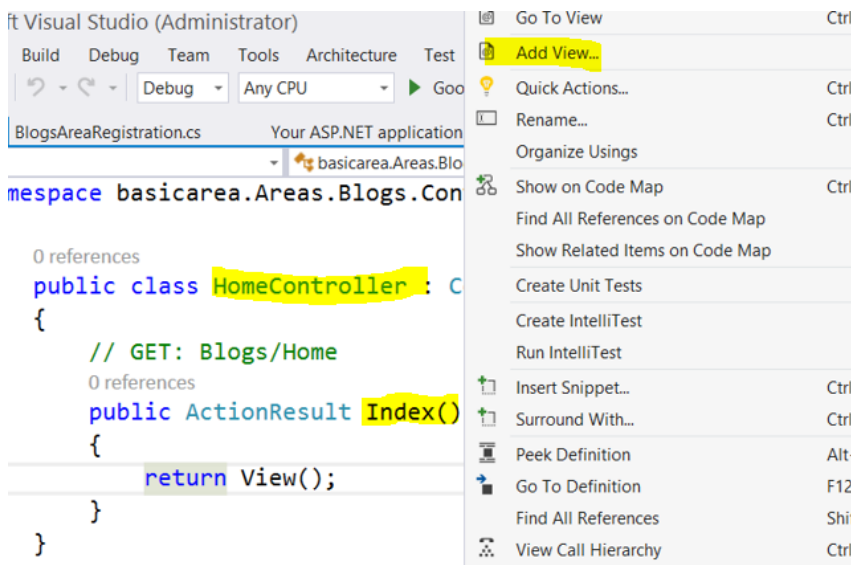
On the next screen, we need to select the view template,
model class, and others. To keep things simpler, let us
leave everything default and click on the Add button to
create a View for the Index action in the Home controller
of the Blogs Area.

We will see that a view was created inside the Views subfolder of the Blogs folder as shown in the image below:
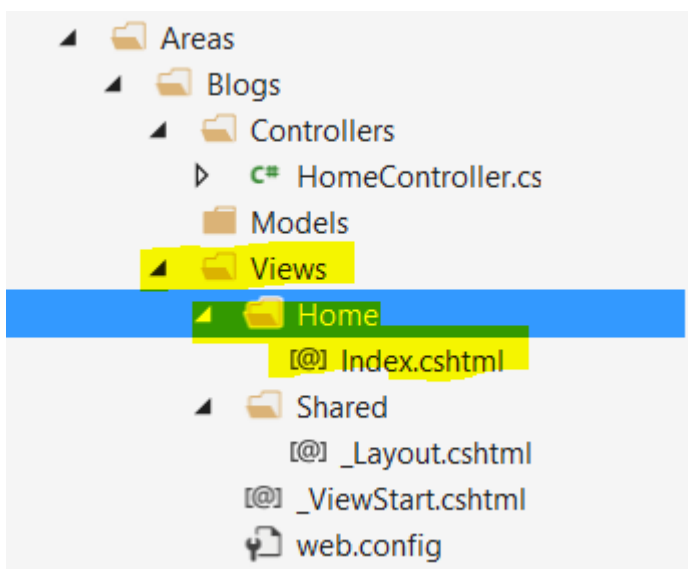
To verify, let us go ahead and change the title of the view as shown in the image below:

```
@{
    ViewBag.Title = "Blogs Area Home Index";
}

<h2>Index</h2>
```
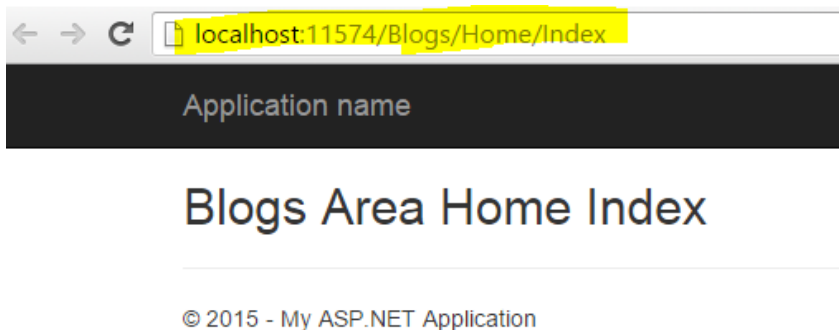
So far we have created:

1. An Area with the name Blogs
2. A controller inside that, named Home

3. A view for the Index action for the Home controller

4. Change the title of the view

As a last step to work with Areas, we need to verify whether the Areas are registered in the App_Start of the project or not. To do this, open global.asax and add the highlighted line of code below (if it's not there already):

```
public class MvcApplication : System.Web.HttpApplication
{
    0 references
    protected void Application_Start()
    {
        AreaRegistration.RegisterAllAreas();
        FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
        RouteConfig.RegisterRoutes(RouteTable.Routes);
        BundleConfig.RegisterBundles(BundleTable.Bundles);
    }
}
```

Now that we have created the Areas, let us go ahead and run the application and notice the URL.

← → C 🗋 localhost:11574/Blogs/Home/Index

Application name

Blogs Area Home Index

© 2015 - My ASP.NET Application

As highlighted, to invoke the controller of the area, we need to use:

baseurl/areaname/controllername/{actionname}

In this case Home controller of Blog area is invoked by:

Baseurl/Blogs/Home/Index

Summary

As you've seen here, Areas are some of the most important components of ASP.NET MVC, allowing us to break big projects into smaller, more manageable units, as demonstrated in this blog's example. I hope you find the post useful, and thanks for reading!

Have you already tried Infragistics jQuery/HTML5 toolset, an industry-leading native web development solution that enables you to create highly-responsive web design on any device? [Download a Free Trial today](#).

👍
[Log in](#) to like this post!

🏷 [ASP.NET MVC](#), [Areas in ASP.NET MVC](#), [News](#)