**Customize view discovery in asp.net core mvc**

**Default View Discovery in ASP.NET Core MVC**   
  
There are several overloaded versions of View() method in ASP.NET Core MVC. If we use one of the following overloaded versions of View() method, it looks for a view file with the same name as the action method.

* View()
* View(object model)

**For example,**we are returning a View() from the Details action method of the HomeController. So, by default, MVC looks for a view file with name Details.cshtml in "Views/Home" folder.

public class HomeController : Controller  
{  
    public ViewResult Details()  
    {  
        return View();  
    }  
}

**View(string viewName) method**   
  
If you do not like this default convention, you can use the overloaded version of the View(string viewName) method, that takes **viewName**as a parameter, to look for a view file with your own custom name.   
  
In the following example, MVC looks for a view file with name "Test.cshtml" instead of "Details.cshtml". Had we not specified the view name, it would have looked for "Details.cshtml".

public class HomeController : Controller  
{  
    public ViewResult Details()  
    {  
        return View("Test");  
    }  
}

**Specifying view file path**   
  
With this overloaded version, we can either specify a **view name**or a **view file path**. In the following example, we specified the absolute view file path. So in this case, MVC looks for a view file with name "Test.cshtml" in "MyViews" folder. Had we not specified the path to the view file, by default, MVC would look for "Details.cshtml" file in "Views/Home" folder. With the absolute path, the **.cshtml** extension must be specified.

public class HomeController : Controller  
{  
    public ViewResult Details()  
    {  
        return View("MyViews/Test.cshtml");  
    }  
}

**Please note :** With the absolute path, to get to the root project directory, we can use **/** or **~/**. So the following 3 lines of code does the same thing

return View("MyViews/Test.cshtml");  
return View("/MyViews/Test.cshtml");  
return View("~/MyViews/Test.cshtml");

**Relative View File Path**  
  
When specifying a view file path, we can also use a relative path. With relative path we do not specify the file extension **.cshtml**. In the following example, MVC looks for **Update.cshtml**file in **"Views/Test"** folder.

public class HomeController : Controller  
{  
    public ViewResult Details()  
    {  
        return View("../Test/Update");  
    }  
}

If you want to go back up 2 levels in the folder hierarchy, use **../** twice as shown below. In the following example, MVC looks for **"Details.cshtml"** in **"MyViews"** folder in the root project directory

public class HomeController : Controller  
{  
    public ViewResult Details()  
    {  
        return View("../../MyViews/Details");  
    }  
}

**Other view() overloaded versions**

|  |  |
| --- | --- |
| **Overloaded Method** | **Description** |
| View(object model) | Use this overloaded version to pass model data from the controller to the view. We will discuss passing data to the view from the controller in our next video |
| View(string viewName, object model) | Pass both the view name and model data. |