**Building the Model in MVC – Displaying Recordings for a Genre**

|  |  |  |
| --- | --- | --- |
| **Introduction**  This tutorial will build on the practices and the concepts you were introduced to in the lecture. | | **Required Software**  For this tutorial, you will need access to the following software applications:  MS Explorer  Visual Studio  SQL SERVER |
| **Reference**  Walther et al, ‘ASP.NET MVC Framework Unleashed’, SAMS | **Breakdown of Tutorial**  This tutorial consists of the following tasks:  You will extend *'Forest'* to include new functionalities. You will be given a series of practical exercises, along with selected questions to consolidate your learning. | | |

**Review questions**

|  |  |
| --- | --- |
| **Refreshers:** Recall from memory as much as possible to answer the following refresher questions. Then open the Forest MVC projects you built during ‘Building the Model in *MVC 1*’ tutorial to help you develop and extend your answers. | |
|  | Explain the application in terms of functionality. What did it do? |
|  | |
|  | What was the architecture of the application like? Draw a diagram to explain the architecture. For example you could describe the rooting of the URL. |
|  | |
|  | How many projects did you create? What were the names of the projects? Why did we create a number of projects as oppose to one? |
|  | |
|  | You were introduced to the concept of interface-based development. What is interface-based development? What are the benefits of this development approach? |
|  | |

**Tutorial objectives**

In the previous tutorial, we built the *Forest* MVC application. The object of this exercise is to extend the application to increase its functionalities. We will learn how MVC's modular development approach helps us build extensible systems.

**Extending the Forest MVC application**

|  |
| --- |
| Open the *Forest* solution. Build the solution and run the application to make sure all is well. |

|  |  |
| --- | --- |
| **About the Application**  Here we introduce an application flow to facilitate the exercise. | |
| C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg | The application opens in *GetMusicCategories.cshtm* View which is *ActionResult* of *GetMusicCategories()* Action of *MusicController*. We get a list of *Genres*. |
| *A genre is click. GetMusicRecordings(Genre)* Action returns a list of *Recordings* in that *Genre*. The list could be limited to few of the fields, for example *Artist* and *Title*. |
|  | Sketch the application's work flow. |
|  | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **About the Architecture** | | | | |
| Capture  Always from right to left ... | | | | |
| C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg | Data Tire | | Our application tends to work with a variety of data sources including various servers such as SQL Server, Oracle Database, MySQL, XML databases and local database files. In our case, we will make use of a SQL Server Local Database file. This database is integrated into the Integration tier. |
| Model | Integration Tier | The integration tier facilitates access to data sources. Looking at code in this layer we expect to see SQL, LINQ, XPath, etc for querying the data provider and data manipulator objects. The integration tier also facilitates access to 3rd party data services such as Web Services. |
| Service Tier | The service tier is concerned with business logic and data manipulation. We implement all the business rules that the application performs in this layer. Looking at code in this layer, we would expect to see references to integration tier objects as well as implementation of mathematical calculations, conditional decisions, etc. |
| MVC | | MVC consists of the controllers, views, JScripts and the various components that deal with the user interaction. MVC is the presentation layer of the architecture. |

|  |  |
| --- | --- |
| **Building the Application**  Objective;  \_ Display *music recordings*. Having displayed *Genres*, we would like to hyperlink Genres to display music recordings in that *Genre*. | |
| **C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg** | MVC facilitates extensibility of the application. That is, you will add additional functionality to the application with ease and without affecting the existing code. As part of this, we have an established regime for coding.  **We always code from right to left. We always code starting with Data and finishing with the controller, the action(s) and the view(s).** |
| **C:\Users\mo\Desktop\Capture.PNG Working on *Forest.Data* project** | |
| \_ Ensure that you have a valid connection to your database.  \_ Ensure that you have created the Forest database. Forest database consists of;   * Music\_Category table. * Music\_Recording table. | |

|  |  |  |
| --- | --- | --- |
| \_Ensure that you have the *ADO.NET Entity Data Model* in your project, named *ForestDataModel.edmx*. We built this in a previous tutorial.  On inspection; |  | |
| The *DbContext* class is the class that provides us with access to the data tables encapsulated within the *ADO.Net Entity Data Model* class. It provides facilities for querying and working with entity data as objects. Explore Your *ADO.Net Entity Data Model* class to ascertain and make a note of the name of your DbContext class.  **DbContext class name:** | | |
| \_ Extend the *IMusicDAO* Interface; | |  |
| \_ Extend the *MusicDAO* class to implement the new method definition in the interface; | |  |
| Save and build *Forest.Data* project. | | |

|  |  |
| --- | --- |
| **C:\Users\mo\Desktop\Capture1.PNG Working on *Forest.Services* project** | |
| \_ Extend the *IMusicService* Interface; |  |
| \_ Extend the *MusicDAO* class to implement the new method definition in the interface; |  |
| Save and build *Forest.Services* project. | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Capture Working on *Forest* project**  Objective;  Note that we created the *MusicController* in a previous tutorial. We also built the *Categories* action that returned a list of *Music\_Category*. We also configured the routing module of the application to open, firing the *Categories* action, displaying the genres of the music. Object of this exercise is to add a new action to the MusicController called Recordings. This action would display recordings in a particular genre. To achieve this we will do the following;  \_ Create the controller action.  \_ Create the *View* for the action.  \_ Create the navigation through linking the genres to the *Recordings* action. | | | | |
| Working on the *MusicController*; | | | | |
| \_ Create a new controller action named *Recordings*. | | | |  |
| \_ Create a *View* for the action. | |  | | |
| Optional;  \_ We can further work on the *Recordings* view to limit the fields that are displayed; | | | |  |
| \_ Save and close the controller. | | | | |
|  | \_ Comment out the *Html.DisplayFor* helper that prints *Genre*.  \_ Create a *Html.ActionLink* helper to display and hyperlink *Genre* to *Recordings* action, passing *Genre* as parameter. | |  | |
| \_ Save all, build the project and press *F5* to run the application. | | | | |

|  |  |
| --- | --- |
| C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg | Note: You must build and run your application very often, almost after each small step in the development.  Build and run your application to make sure it runs to this point. |

|  |  |
| --- | --- |
| Your application should open the *Categories* view for the *Categories* action of *MusicController*. You should see two navigation links: ‘*Music*’ and ‘*Video*’ | |
|  | Hover the mouse over the *Genre* links and you should see the URL on the status bar. What is the URL? What is the link pointing to? Is it passing anything? |
|  | |
|  | Having clicked one of the *Genre*s, you should be looking at the list of recordings. Hover the mouse over the *Edit*, *Details* and *Delete* links. What are the URLs? What are the links pointing to? Are they passing anything? |
|  | |

**Student-centred exercise**

You are to work on your own to extend the application for Video;

|  |  |
| --- | --- |
| * Create the Video Data Accessor (VideoDAO) | C:\Users\cmsmr2\Desktop\Capture.JPG |
| * Create the Video Search Service (VideoService) | C:\Users\cmsmr2\Desktop\2.JPG |