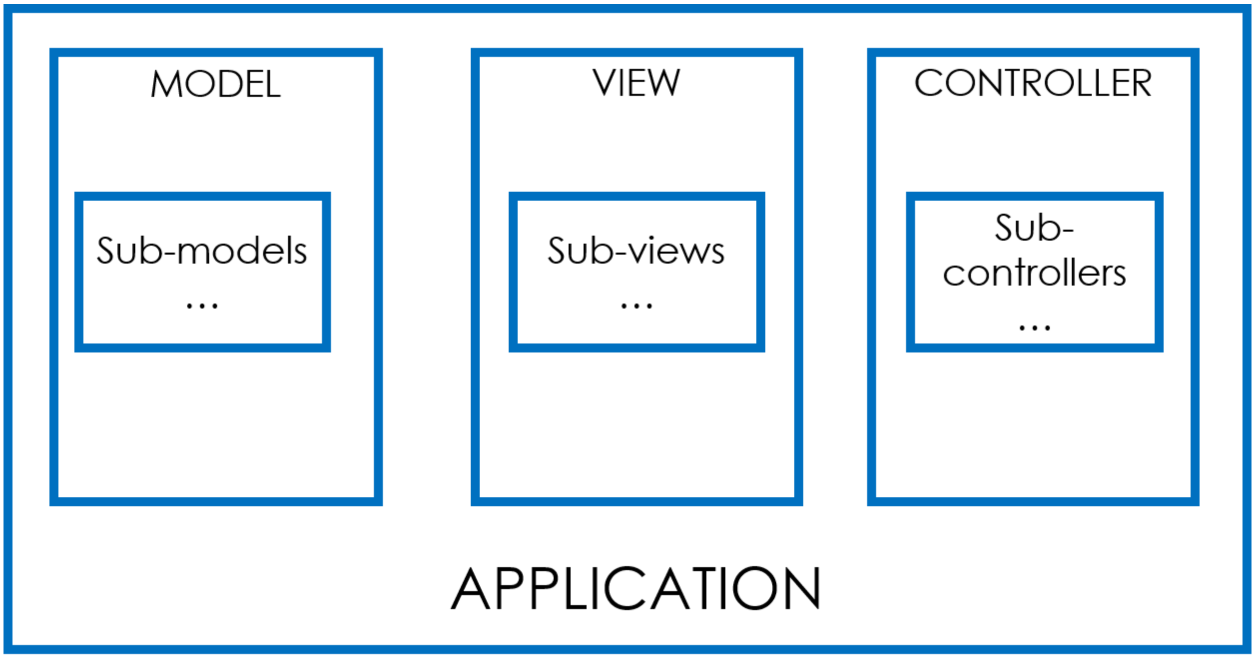
AMVC C# Framework Quick Tutorial

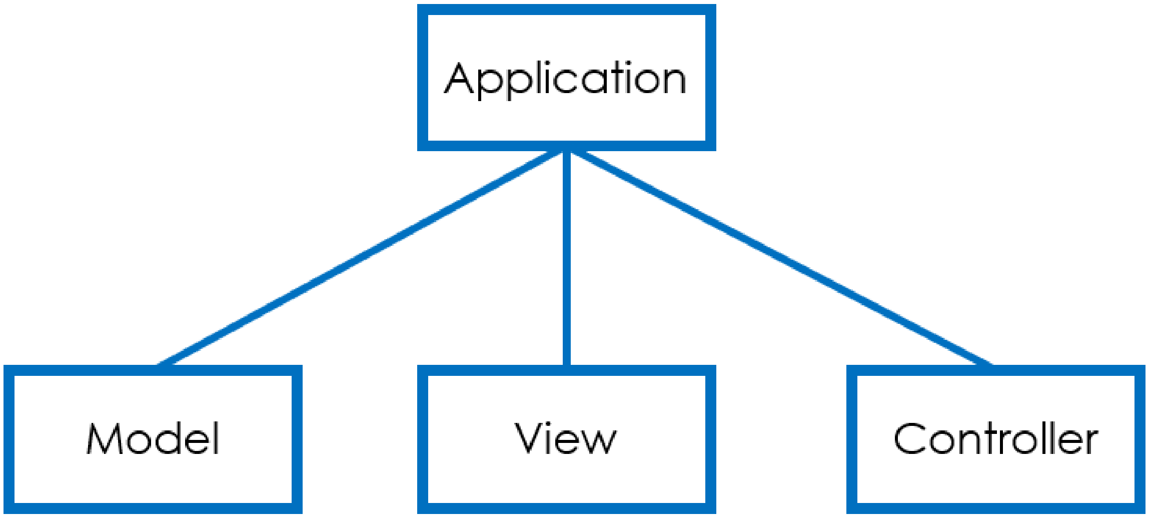
What is AMVC?

AMVC stands for Application Model-View-Controller. This is a variant of MVC which decouples contexts into modules called Application.



AMVC Components Interaction

AMVC internal interaction follows the original MVC paradigm. To make convenience, all three components MVC are able to communicate through their Application (Application is root).



Controller controls Model and View like original MVC but access them through Application

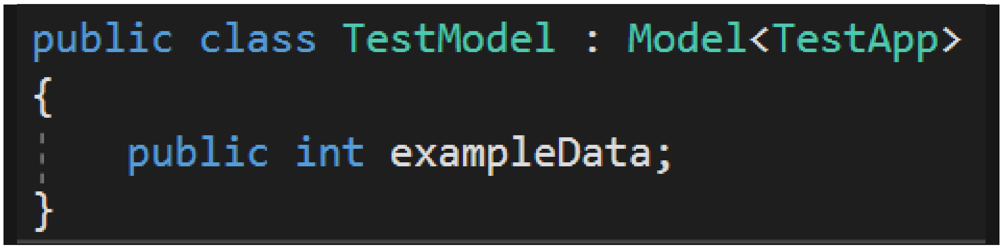
Additionally, one variant on this framework is to make all model, view and controller are able to interact together (view->controller, controller->model, model->controller, …) instead of restricting access of view and model like the original MVC.

Implementation Your Own AMVC System

You need to create your own Model, View and Controller concretely. The syntax to create them as follows:

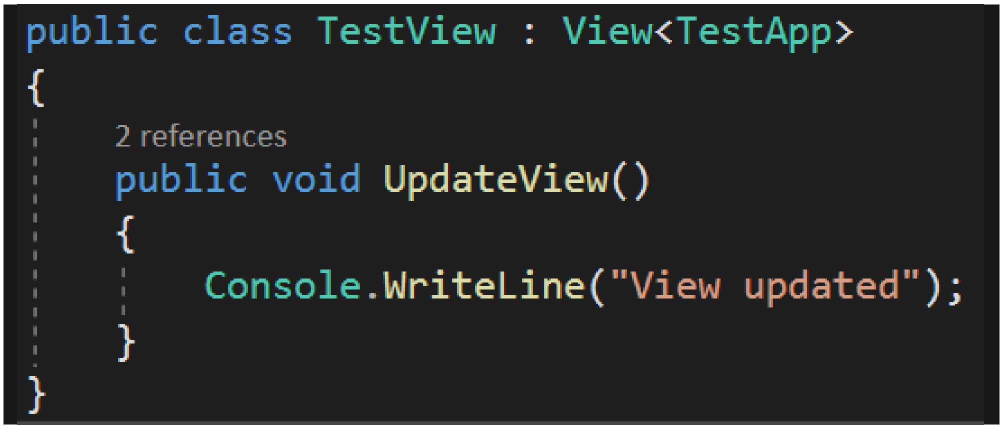
```
public class YourModel : Model<YourApp> { }
public class YourView : View<YourApp> { }
public class YourController : Controller<YourApp> { }
public class YourApp : App<YourModel, YourView, YourController> { }
```

Below is an example:



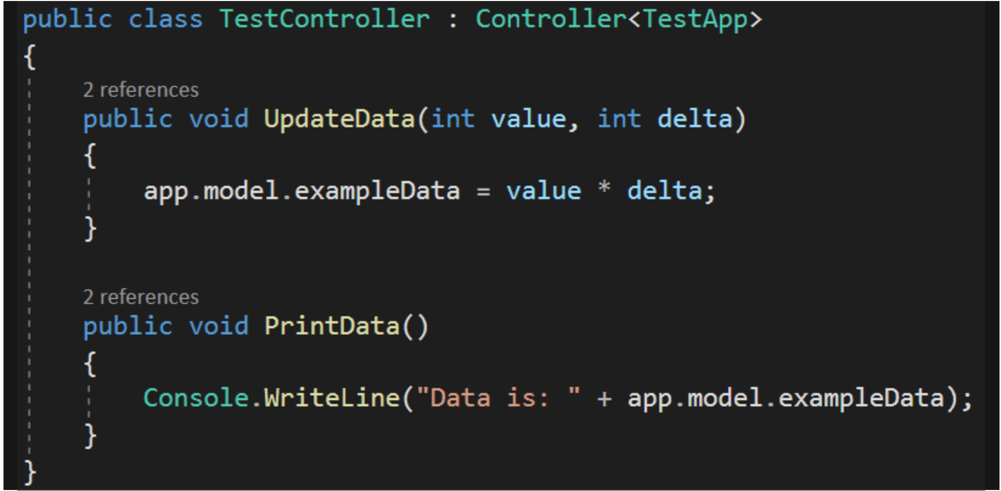
```
public class TestModel : Model<TestApp>
{
    public int exampleData;
}
```

My TestModel



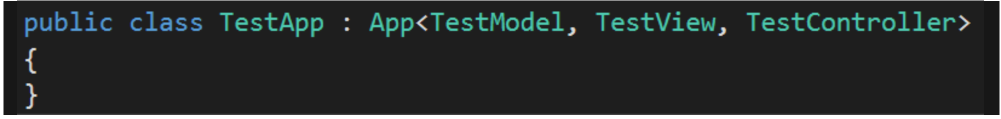
```
public class TestView : View<TestApp>
{
    2 references
    public void UpdateView()
    {
        Console.WriteLine("View updated");
    }
}
```

My TestView with some methods



```
public class TestController : Controller<TestApp>
{
    2 references
    public void UpdateData(int value, int delta)
    {
        app.model.exampleData = value * delta;
    }

    2 references
    public void PrintData()
    {
        Console.WriteLine("Data is: " + app.model.exampleData);
    }
}
```

My TestController with some methods

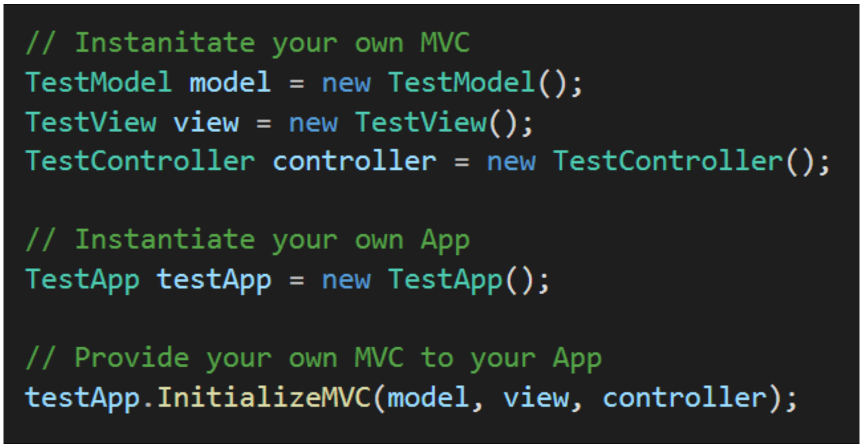


```
public class TestApp : App<TestModel, TestView, TestController>
{
}
```

And TestApp is consist all of them

Initialization of Your Own AMVC System

Use **new** keyword to instantiate your **Application** then use **InitializeMVC** method to provide your MVC to it.



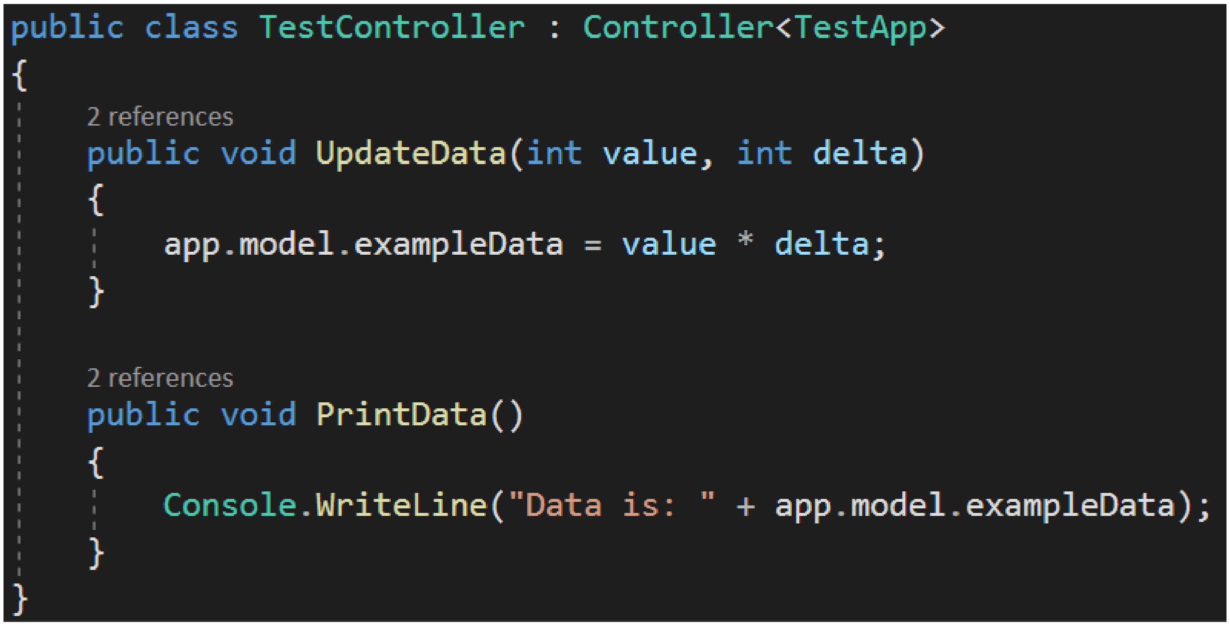
```
// Instanitate your own MVC
TestModel model = new TestModel();
TestView view = new TestView();
TestController controller = new TestController();

// Instantiate your own App
TestApp testApp = new TestApp();

// Provide your own MVC to your App
testApp.InitializeMVC(model, view, controller);
```

Accessing in AMVC System

As description on Interaction Diagram above, components in MVC interact through application. The syntax as follows:

From inside model or view or controller to other one (access through **app** property)	
Access to Model	`app.model`
Access to View	`app.view`
Access to Controller	`app.controller`

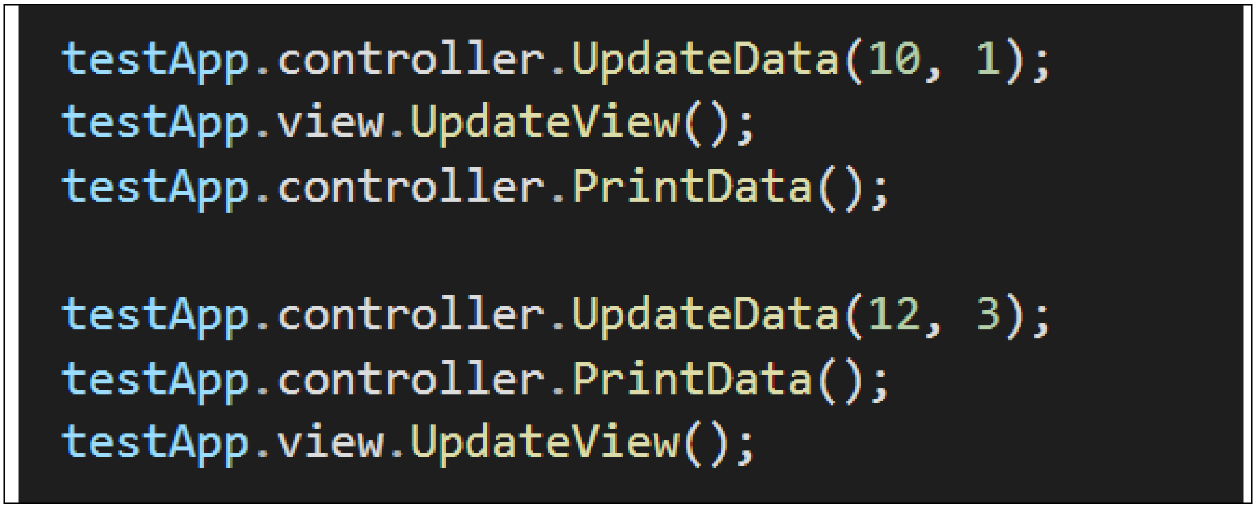
Example of accessing to model inside controller



```
public class TestController : Controller<TestApp>
{
        2 references
        public void UpdateData(int value, int delta)
        {
                app.model.exampleData = value * delta;
        }

        2 references
        public void PrintData()
        {
                Console.WriteLine("Data is: " + app.model.exampleData);
        }
}
```

From outside	
Access to Model	`<yourApp>.model`
Access to View	`<yourApp>.view`
Access to Controller	`<yourApp>.controller`

Example of accessing from outside



```
testApp.controller.UpdateData(10, 1);
testApp.view.UpdateView();
testApp.controller.PrintData();

testApp.controller.UpdateData(12, 3);
testApp.controller.PrintData();
testApp.view.UpdateView();
```

Notes

- A Controller can contain sub-controllers, so do View and Model
- Getter of application finds and returns based on **Type of Application**, thus make sure you create **one instance of one type of application**.