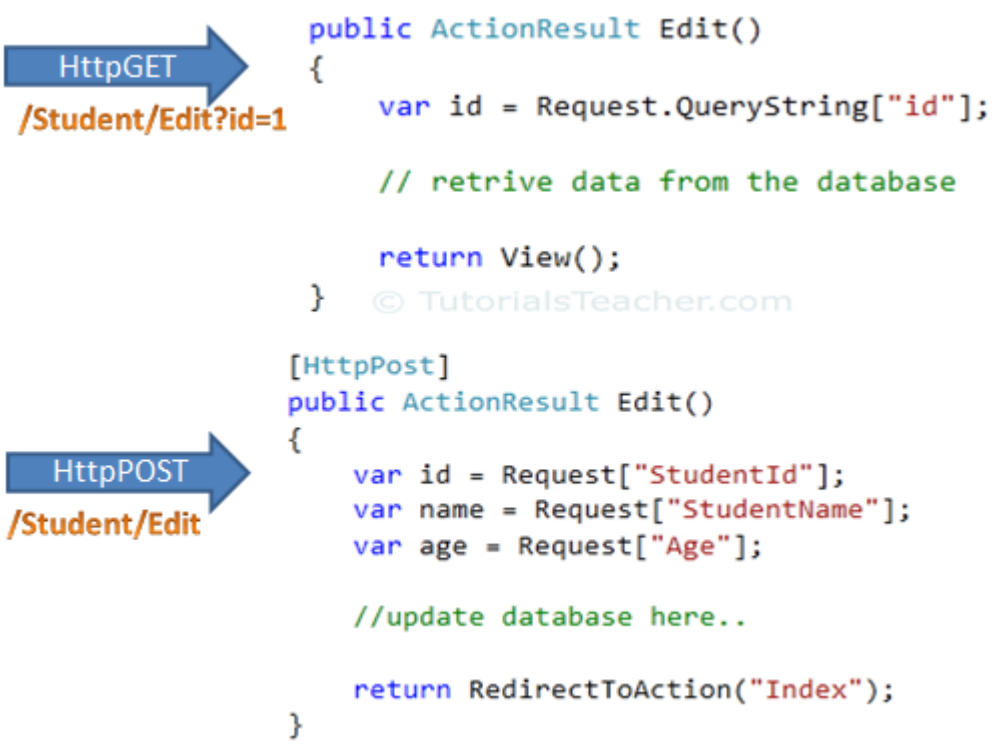
Previous Next

Model Binding

In this section, you will learn about model binding in MVC framework.

To understand the model binding in MVC, first let's see how you can get the http request values in the action method using traditional ASP.NET style. The following figure shows how you can get the values from HttpGET and HttpPOST request by using the Request object directly in the action method.



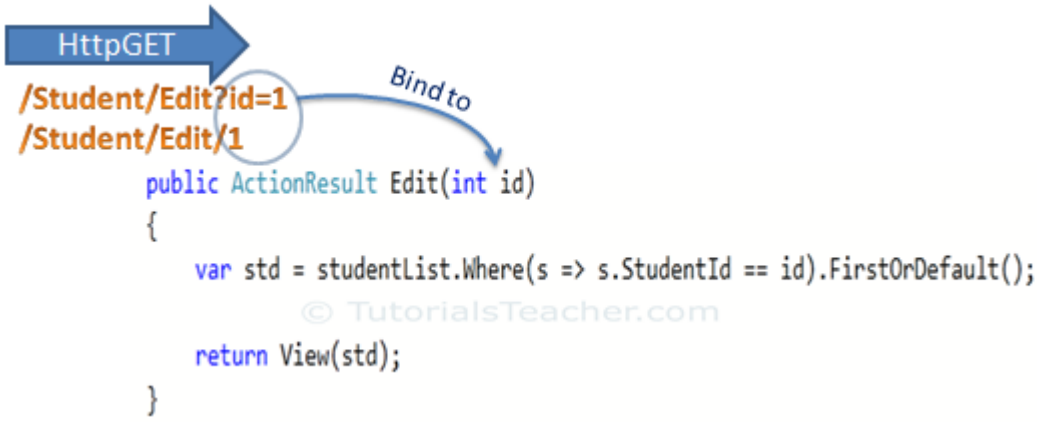
Accessing Request Data

As you can see in the above figure, we use the Request.QueryString and Request (Request.Form) object to get the value from HttpGet and HttpPOST request. Accessing request values using the Request object is a cumbersome and time wasting activity.

With model binding, MVC framework converts the http request values (from query string or form collection) to action method parameters. These parameters can be of primitive type or complex type.

Binding to Primitive type

HttpGET request embeds data into a query string. MVC framework automatically converts a query string to the action method parameters. For example, the query string "id" in the following GET request would automatically be mapped to the id parameter of the Edit() action method.



Model Binding

This binding is case insensitive. So "id" parameter can be "ID" or "Id".

You can also have multiple parameters in the action method with different data types. Query string values will be converted into paramters based on matching name.

For example, *http://localhost/Student/Edit?id=1&name=John* would map to id and name parameter of the following Edit action method.

Example: Convert QueryString to Action Method Parameters

```
public ActionResult Edit(int id, string name)
{
    // do something here
```

```
        return View();
    }
```

Binding to Complex type

Model binding also works on complex types. Model binding in MVC framework automatically converts form field data of HttpPOST request to the properties of a complex type parameter of an action method.

Consider the following model classes.

Example: Model classes in C#

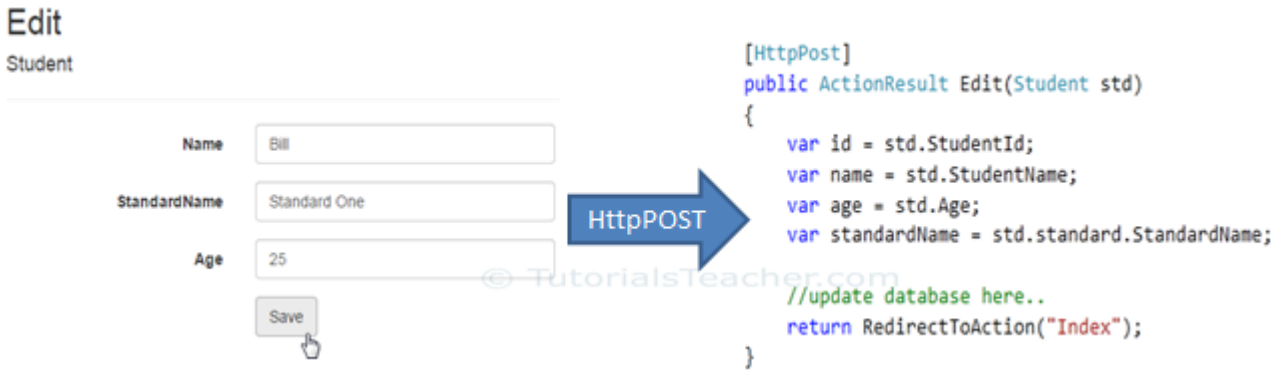
```
public class Student
{
    public int StudentId { get; set; }
    [Display(Name="Name")]
    public string StudentName { get; set; }
    public int Age { get; set; }
    public Standard standard { get; set; }
}

public class Standard
{
    public int StandardId { get; set; }
    public string StandardName { get; set; }
}
```

Now, you can create an action method which includes Student type parameter. In the following example, Edit action method (HttpPost) includes Student type parameter.

Example: Action Method with Class Type Parameter

```
[HttpPost]
public ActionResult Edit(Student std)
{
    var id = std.StudentId;
    var name = std.StudentName;
    var age = std.Age;
    var standardName = std.standard.StandardName;

    //update database here..

    return RedirectToAction("Index");
}
```

So now, MVC framework will automatically maps Form collection values to Student type parameter when the form submits http POST request to Edit action method as shown below.



Model Binding

So thus, it automatically binds form fields to the complex type parameter of action method.

FormCollection

You can also include FormCollection type parameter in the action method instead of complex type, to retrieve all the values from view form fields as shown below.

Model Binding

Bind Attribute

ASP.NET MVC framework also enables you to specify which properties of a model class you want to bind. The [Bind] attribute will let you specify the exact properties a model binder should include or exclude in binding.

In the following example, Edit action method will only bind StudentId and StudentName property of a Student model.

Example: Binding Parameters

```
[HttpPost]
```

```
public ActionResult Edit([Bind(Include = "StudentId, StudentName")] Student std)
{
    var name = std.StudentName;

    //write code to update student

    return RedirectToAction("Index");
}
```

You can also use Exclude properties as below.

Example: Exclude Properties in Binding

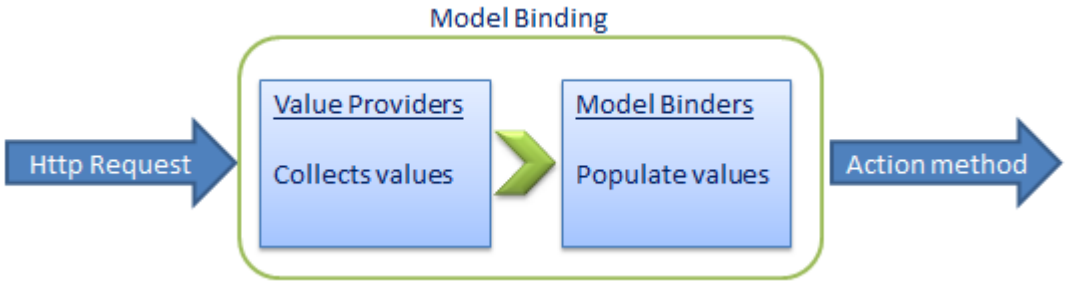
```
[HttpPost]
public ActionResult Edit([Bind(Exclude = "Age")] Student std)
{
    var name = std.StudentName;

    //write code to update student

    return RedirectToAction("Index");
}
```

The Bind attribute will improve the performance by only bind properties which you needed.

Inside Model Binding

As you have seen that Model binding automatically converts request values into a primitive or complex type object. Model binding is a two step process. First, it collects values from the incoming http request and second, populates primitive type or complex type with these values.

Value providers are responsible for collecting values from request and Model Binders are responsible for populating values.



Model Binding in MVC

Default value provider collection evaluates values from the following sources:

1. Previously bound action parameters, when the action is a child action
2. Form fields (Request.Form)
3. The property values in the JSON Request body (Request.InputStream), but only when the request is an AJAX request
4. Route data (RouteData.Values)
5. Querystring parameters (Request.QueryString)
6. Posted files (Request.Files)

MVC includes DefaultModelBinder class which effectively binds most of the model types.

Visit MSDN for detailed information on Model binding.

Previous Next

TUTORIALS

- ASP.NET Core
- ASP.NET MVC
- IoC
- Web API
- C#
- LINQ
- Entity Framework
- AngularJS 1
- Node.js
- D3.js
- JavaScript
- jQuery
- Sass
- Https

E-MAIL LIST

Subscribe to TutorialsTeacher email list and get latest updates, tips & tricks on C#, .Net, JavaScript, jQuery, AngularJS, Node.js to your inbox.

Email address GO

We respect your privacy.