



Data Validation and Data Passing

Data Validation

- Helps to ensure that the received data is correct and valid.
- Data should be validated at client side and server side.
- Prevent unwanted attack.

Data Validation Techniques



Server Side

Client Side

Server Side Validation

- Helps to protect your data from dirty input.
- As a best practice you must use.
- Server Side Validation Ways:
 - Explicit Model Validation (Traditional)
 - Data Annotations Model Validation

Client Side Validation

- Make web page highly interactive and user friendly.
- Save a number of request to server.
- Client Side Validation Ways:
 - JavaScript
 - Third Party JS Library/framework

Model State

- A collection of name and value pairs that are submitted to the server during a POST.
- Also contains error messages about each name-value pair, if such errors exist, during a POST action.
- ModelState is a property of a Controller instance that inherits `Microsoft.AspNetCore.Mvc.Controller` class.

Data Passing Techniques

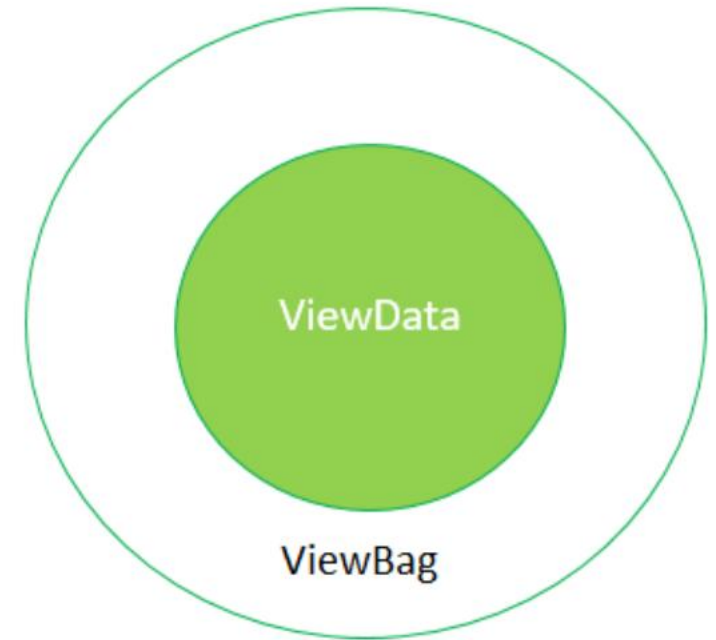
- ViewData
- ViewBag
- TempData
- Session
- QueryString
- Cookies

ViewData

- A dictionary object that stores value in key-value pair.
- Used to pass data from controller to view.
- Need typecasting for getting data in C#.
- Value becomes null If redirection occurs.

ViewBag

- A dynamic property which wraps ViewData.
- Used to pass data from controller to view.
- Don't need typecasting for getting data in C#.
- Value becomes null If redirection occurs.



ViewData and ViewBag

```
public ActionResult StudentIndex()
{
    List<string>lstStudent = new List<string>();
    lstStudent.Add("Debasis");
    lstStudent.Add("Samrat");
    lstStudent.Add("Rahul");
    ViewData["StudentData"] = lstStudent;
    return View();
}
```

```
<ul>
@foreach (var data in ViewData["StudentData"] as
List<string>)
{
    <li>@data</li>
}
</ul>
```

```
public ActionResult Index()
{
    List<string> lstStudent = new
List<string>();
    lstStudent.Add("Debasis");
    lstStudent.Add("Samrat");
    lstStudent.Add("Rahul");
    ViewBag.StudentData = Student;
    return View();
}
```

```
<ul>
@foreach (var student in
ViewBag.StudentData)
{
    <li>@student</li>
}
</ul>
```

TempData

- A dictionary object that stores value in key-value pair.
- Used to pass data from current request to subsequent request. Need typecasting for getting data in C#.
- Internally use session for storing values till ASP.NET Core 1.x.
- From ASP.NET Core 2.0, the cookie TempData provider is now the default TempData provider means you no longer need to setup session support to make use of the TempData features

```
public ActionResult Index()
{
    List<string> lstStudent = new List<string>();
    lstStudent.Add("Debasis");
    lstStudent.Add("Samrat");
    lstStudent.Add("Rahul");
    TempData["StudentData"] = Student;
    return View();
}
```

```
<ul>
    @foreach (var data in TempData["StudentData"]
as List<string>)
    {
        <li>@data</li>
    };
</ul>
```


Data Passing

