ASP.NET MVC offers us three options - ViewData, ViewBag and TempData for passing data from controller to view and in next request. ViewData and ViewBag are almost similar and TempData performs additional responsibility. Let's discuss or get key points on those three objects:

Similarities between ViewBag & ViewData:

1. Helps to maintain data when you move from controller to view.
2. Used to pass data from controller to corresponding view.
3. Short life means value becomes null when redirection occurs. This is because their goal is to provide a way to communicate between controllers and views. It’s a communication mechanism within the server call.

Difference between ViewBag & ViewData:

1. ViewData is a dictionary of objects that is derived from ViewDataDictionary class and is accessible using strings as keys.
2. ViewBag is a dynamic property that takes advantage of the new dynamic features in C# 4.0.
3. ViewData requires typecasting for complex data type and check for null values to avoid error.
4. ViewBag doesn’t require typecasting for complex data type.

ViewBag & ViewData Example:

Hide   Copy Code

public ActionResult Index()

{

ViewBag.Name = "Monjurul Habib";

return View();

}

Hide   Copy Code

public ActionResult Index()

{

ViewData["Name"] = "Monjurul Habib";

return View();

}

In View:

Hide   Copy Code

@ViewBag.Name

@ViewData["Name"]

TempData:

TempData is also a dictionary derived from TempDataDictionary class and stored in short lives session and it is a string key and object value. The difference is the life cycle of the object. TempData keeps the information for the time of an HTTP Request. This mean only from one page to another. This also works with a 302/303 redirection because it’s in the same HTTP Request. It helps to maintain data when you move from one controller to other controller or from one action to other action. In other words, when you redirect, “Tempdata” helps to maintain data between those redirects. It internally uses session variables. Temp data use during the current and subsequent request only means it is used when you are sure that next request will be redirecting to next view. It requires typecasting for complex data type and check for null values to avoid error. It is generally used to store only one time messages like error messages, validation messages.

Hide   Copy Code

public ActionResult Index()

{

var model = new Review()

{

Body = "Start",

Rating=5

};

TempData["ModelName"] = model;

return RedirectToAction("About");

}

Hide   Copy Code

public ActionResult About()

{

var model= TempData["ModelName"];

return View(model);

}

The last mechanism is the Session which works like the ViewData, like a Dictionary that takes a string for key and object for value. This one is stored into the client Cookie and can be used for a much more long time. It also needs more verification to never have any confidential information. Regarding ViewData or ViewBag, you should use it intelligently for application performance. Because each action goes through the whole life cycle of regular ASP.NET MVC request. You can use ViewData/ViewBag in your child action, but be careful that you are not using it to populate the unrelated data which can pollute your controller.