**2. Sự khác nhau giữa static readonly và const?**

- Static readonly là hằng số được khởi tạo khi [thực thi chương trình.](http://love4all1080.blogspot.com/)   
- Const là hằng số được khởi tạo khi biên dịch chương trình.

**3. Sự khác nhau giữa String và StringBuilder?**

- String là lớp không thay đổi. Mọi phương thức sẽ tạo ra một biến String mới.   
- StringBuilder là lớp thay đổi. Các phương thức của biến sẽ ảnh hưởng đến đến biến [StringBuilder](http://love4all1080.blogspot.com/) hiện tại.

**4. Sự khác nhau giữa “” và String.Empty?**

- “” sẽ tạo thêm 1 đối tượng khi làm việc với nó (so sánh, copy…)   
- String.Empty không tạo đối tượng nào cả. String.Emtpy nằm trong mscorlib.   
Khi kiểm tra 1 String có rỗng hay không nên dùng String.Length == 0 để có tốc độ nhanh nhất.

**5. Có thể gán một struct bằng null hay không?**

- Có thể nếu bạn lập trình với C# 2.0. Ví dụ: int? var = 1; var = null;

**6. Sự khác nhau giữa == và phương thức Equals?**

- == so sánh theo kiểu giá trị.   
- Equals so sánh theo kiểu tham chiếu.

**10. Sự khác nhau giữa abstract class và interface?**

- Có thể thừa kế từ nhiều interface nhưng chỉ có thể thừa kế từ 1 abstract class.   
- Có thể viết sẵn các thực thi trong Abstract class, nhưng interface thì không.   
- Interface dùng để gom các hành động cần được hiện thực , các khả năng của một đối tượng (ví dụ con người và xe cộ cùng có khả năng đi lại, nhưng con người không cùng tính chất với xe cộ), còn abstract class cho các lớp thừa kế cùng 1 loại, tính chất hay trạng thái (ví dụ đàn ông, phụ nữ, trẻ con, người già đều là con người).   
- Abstract class có tốc độ thực thi nhanh hơn interface.   
- Thêm 1 tính năng mới vào interface sẽ phá vỡ toàn bộ các lớp hiện thực, còn abstract thì không.

**12. Sự khác nhau giữa DataSet và DataReader?**

- DataSet có thể đọc và ghi trong khi DataReader chỉ có thể đọc.

- DataSet có thể chứa nhiều bảng còn DataReader chỉ có thể chứa 1 bảng.

- DataSet làm việc ở chế độ không kết nối còn DataReader làm việc ở chế độ kết nối.

- DataSet có thể di chuyển tới và trở lại còn DataReader thì chỉ có thể di chuyển tới.

- DataSet truy cập chậm hơn DataReader.

**18. Sự khác nhau giữa class, object, struct và interface?**

- Class là cấu trúc dữ liệu định nghĩa bởi người dùng chứa các tính chất và phương thức. Object là một trường hợp cụ thể của class. Class không chiếm bộ nhớ, còn Object được tạo trong bộ nhớ. - Class là cấu trúc dữ liệu định nghĩa bởi người dùng chứa các tính chất và phương thức. Struct là kiểu giá trị được lưu trữ trong stack còn class là kiểu tham chiếu (các object) được lưu trữ trong heap. - Khi struct được khởi tạo không bằng hàm khởi tạo người dùng sẽ phải khởi tạo tất cả các Fields trước khi sử dụng nó.   
- Interface là một hợp đồng gồm một nhóm các phương thức cần được hiện thực.

**34. Sự khác nhau giữa error và exception?**

- Error là lỗi của chương trình còn exception là lỗi chương trình được phát hiện và có thể xử lý.

**Phân biệt HttpApplication, Session, ViewSate và HttpContext?**

- Cả 4 thành phần trên đều dùng để lưu dữ liệu của một Web Application.

- [HttpApplication](http://love4all1080.blogspot.com/) có tác dụng trong toàn bộ quá trình thực thi của Web Application.

- Session có tác dụng trong 1 lần ghé thăm (visit) của người dùng. Nó kết thúc khi gặp timeout.

- ViewState có tác dụng trong trong 1 trang web.

- HttpContext có tác dụng trong 1 request.

**Cách truyền dữ liệu từ Controller về View có những cách nào? ViewBag, ViewData, TempData và  Session? Sự khác nhau giữa chúng như thế nào? Khi nào thì dùng cái nào?**

ASP.NET MVC cung cấp 3 tùy chọn ViewData, ViewBag và TempData để truyền dữ liệu từ controller vào view và trong request kế tiếp. ViewData và ViewBag tương tự như nhau, TempData thực hiện thêm nhiệm vụ. Chúng ta sẽ nói về điểm chính của 3 đối tượng:

**Sự giống nhau giữa ViewBag và ViewData:**

1. Giúp duy trì dữ liệu khi bạn di chuyển từ view vào controller

2. Được dùng để truyền dữ liệu từ controller vào view tương ứng

3. Chu kỳ tồn tại ngắn, giá trị sẽ thành null khi redirect xuất hiện. Đây là bởi vì mục tiêu thiết kế các đối tượng này là để giao tiếp giữa controller và view

**Sự khác nhau giữa ViewBag và ViewData:**

1. ViewData là 1 từ điển các đối tượng (dictionary) kế thừa từ lớp ViewDataDictionary và có thể truy cập dùng chuỗi key.

2. ViewBag là thuộc tính động (dynamic) mang đến thuận lợi từ đặc tính mới của C# 4.0

3. ViewData yêu cầu chuyển kiểu với các kiểu dữ liệu phức tạp và kiểm tra giá trị null để tránh lỗi

4. ViewBag không yêu cầu chuyển kiểu với các kiểu dữ liệu phức tạp

**Ví dụ về ViewBag và ViewData:**

public ActionResult Index()

{

ViewBag.Name = "Ngoc Dinh NGUYEN";

return View();

}

public ActionResult Index()

{

ViewData["Name"] = "Ngoc Dinh NGUYEN";

return View();

}

Trong View:

@ViewBag.Name

@ViewData["Name"]

**TempData:**

TempData cũng là 1 từ điển đối tượng (dictionary) kế thừa từ lớp TempDataDictionary, được lưu trong 1 chu kỳ rất ngắn. Điểm khác biệt chính là chu kỳ sống của đối tượng. TempData giữ thông tin trong khoảng thời gian HTTP Request. Điều này có nghĩa là chỉ từ 1 trang đến 1 trang khác. Điều này cũng hoạt động với 302/303 redirection bởi vì nó cùng HTTP Request. Giúp duy trì dữ liệu khi bạn di chuyển từ 1 controller sang 1 controller khác hoặc từ hành động này sang hành động khác. Nói cách khác, khi bạn redirect, TempData giúp duy trì dữ liệu giữa các redirect. Nó là biến cục bộ dùng trong session. Nó yêu cầu chuyển kiểu cho các kiểu dữ liệu phức tạp, và kiểm tra giá trị null để tránh lỗi. Thông thường nó dùng để lưu error message hoặc validation message.

public ActionResult Index()

{

var model = new Review()

{

Body = "Start",

Rating = 5

};

TempData["ModelName"] = model;

return RedirectToAction("About");

}

public ActionResult About()

{

var model = TempData["ModelName"];

return View(model);

}

## ViewBag-ViewData

* Giống nhau về cách sử dụng của ViewBag và ViewData:
  + Giúp bạn có thể chuyển dữ liệu từ [Controller](http://codehay365.com/hd-lap-trinh-web-asp-net-mvc-5-phan-2/) xuống [View](http://codehay365.com/hd-lap-trinh-web-asp-net-mvc-5-phan-4/)
  + Truyền dữ liệu từ Controller xuống View tương ứng
  + Chỉ sử dụng 1 lần duy nhất, sau khi dùng xong dữ liệu ở hai thằng này được giải phóng.
* Khác nhau giữa ViewBag và ViewData:
  + ViewData là một dạng Dictionary, để truy cập vào biến bạn cần khai báo một key dạng string: ViewData[“Danhmuc”].
  + ViewBag: là một điểm mới trong Net Framework 4.0. Cho phép bạn định nghĩa các biến dạng Dynamic: ViewBag.DanhMuc.
  + ViewData: đòi hỏi bạn cần kiểm tra dữ liệu Null và chuyển đổi kiểu trước khi sử dụng
  + ViewBag: không đòi hỏi kiểm tra hay chuyển đổi kiểu (ép kiểu)

**Cách làm việc Routing của ASP.NET MVC**

* Tất cả cấu trúc URL đều được khai báo trong Global.ascx, chịu trách nhiệm ánh xạ đường dẫn trên trình duyệt.

Trình duyệt yêu cầu một địa chỉ từ Controller Acction được gọi là định tuyến URL (URL routing).

URL routing sẽ chỉ định request tới Controller Action.

URL routing sử dụng một bảng định tuyến để điều khiển các yêu cầu (request).

Bảng định tuyến được tạo khi ứng dụng được chạy lần đầu tiên.

Bảng định tuyến được thiết lập trong file Global.asax