# Server.Transfer Vs. Response.Redirect

Asked 15 years, 10 months ago Modified 5 years, 9 months ago Viewed 251k times

276

What is difference between `Server.Transfer` and `Response.Redirect` ?

- What are advantages and disadvantages of each?
- When is one appropriate over the other?
- When is one not appropriate?

asp.net http-redirect server.transfer

Share Edit Follow Flag

edited Sep 24, 2013 at 2:25
Ian Boyd
254k 262 895 1.3k

asked Oct 22, 2008 at 5:34
kedar kamthe
8,148 11 36 46

3 The advantages and disadvantages have been stated within the site below. developer.com/net/asp/article.php/3299641 One interesting point in the article is that Server.Transfer consumes more server power in comparison to Server.Redirect. – Ray Lu Oct 22, 2008 at 5:49

1 Server.Transfer reduces page requests, so I suppose it's "better" in that respect. However, Response.Redirect can send the user to an external site while Server.Transfer can't. – codeConcussion Feb 6, 2009 at 18:32

1 If you're running on IIS 7 Integrated mode, you might consider using `Server.TransferRequest` instead of `Server.Transfer` . – Haacked May 12, 2010 at 16:11

@Haacked should have read that in the beginning, Server.TransferRequest solved my issues iwth web matrix and iis7. Gracias. They should put that up here. – King Friday May 15, 2012 at 1:54

## 16 Answers

Sorted by: Highest score (default)

249

`Response.Redirect` simply sends a message (HTTP 302) down to the browser.

`Server.Transfer` happens without the browser knowing anything, the browser request a page, but the server returns the content of another.

Share Edit Follow Flag

edited Feb 4, 2015 at 16:12
Mark Rucker
7,814 4 42 66

answered Oct 22, 2008 at 5:41
Christian C. Salvadó
822k 184 924 840

Does this work with CSHTML pages with web matrix? I can't seem to get it to work when doing a Server.Transfer to a CSHTML page such as Server.Transfer("~/somepage.cshtml",true) but seems to work for other types of pages. Yes I have razor installed and pages work as expected otherwise. – King Friday May 15, 2012 at 0:24

11 Hey found out. You have to use Server.TransferRequest for cshtml web matrix pages. – King Friday May 15, 2012 at 1:53

does Server.Transfer() only transfers to physical pages? for eg. if I transfer to Server.Transfer("default/category1.aspx") then is it requred to have a default folder and a category1,aspx page in it? – ihimv Oct 8, 2015 at 7:38

---

104

`Response.Redirect()` will send you to a new page, update the address bar and add it to the Browser History. On your browser you can click back.

`Server.Transfer()` does not change the address bar. You cannot hit back.

I use `Server.Transfer()` when I don't want the user to see where I am going. Sometimes on a "loading" type page.

Otherwise I'll always use `Response.Redirect()` .

Share Edit Follow Flag

edited Feb 4, 2015 at 16:13
Mark Rucker
7,814 4 42 66

answered Oct 22, 2008 at 5:44
Christian Payne
7,169 5 40 59

---

77

To be Short: `Response.Redirect` simply tells the browser to visit another page. `Server.Transfer` helps reduce server requests, keeps the URL the same and, with a little bug-bashing, allows you to transfer the query string and form variables.

Something I found and agree with (source):

> `Server.Transfer` is similar in that it sends the user to another page with a statement such as `Server.Transfer("WebForm2.aspx")` . However, the statement has a number of distinct advantages and disadvantages.
>
> Firstly, transferring to another page using `Server.Transfer` conserves server resources. Instead of telling the browser to redirect, it simply changes the "focus" on the Web server and transfers the request. This means you don't get quite as many HTTP requests coming through, which therefore eases the pressure on your Web server and makes your applications run faster.

But watch out: because the "transfer" process can work on only those sites running on the server; you can't use `Server.Transfer` to send the user to an external site. Only `Response.Redirect` can do that.

Secondly, `Server.Transfer` maintains the original URL in the browser. This can really help streamline data entry techniques, although it may make for confusion when debugging.

That's not all: The `Server.Transfer` method also has a second parameter —"preserveForm". If you set this to `True`, using a statement such as `Server.Transfer("WebForm2.aspx", True)`, the existing query string and any form variables will still be available to the page you are transferring to.

For example, if your WebForm1.aspx has a TextBox control called TextBox1 and you transferred to WebForm2.aspx with the preserveForm parameter set to True, you'd be able to retrieve the value of the original page TextBox control by referencing `Request.Form("TextBox1")`.

Share Edit Follow Flag

edited Aug 6, 2013 at 12:50
Josh Darnell
11.4k 9 39 66

answered Feb 6, 2009 at 18:39
TStamper
30.3k 10 67 73

10 +1 for comment but this seems copied verbatim from developer.com/net/asp/article.php/3299641. If it is from another source you should at lease cite it. – Johnno Nolan Feb 25, 2009 at 9:58

... but they have copied it they should cite you. – Johnno Nolan Feb 25, 2009 at 10:03

7 I said: Something I found and agree with; – TStamper Feb 25, 2009 at 15:10

6 Should link to source and use quote formatting/highlighting for the copied parts. – Chris W. Rea Jun 18, 2010 at 19:05

1 How can `maintaining the original URL... ...really help streamline data entry techniques` ? – JohnB Dec 1, 2010 at 17:24

39

`Response.Redirect()` should be used when:

- we want to redirect the request to some plain HTML pages on our server or to some other web server
- we don't care about causing additional roundtrips to the server on each request
- we do not need to preserve Query String and Form Variables from the original request
- we want our users to be able to see the new redirected URL where he is redirected in his browser (and be able to bookmark it if its necessary)

`Server.Transfer()` should be used when:

- we want to transfer current page request to another .aspx page on the same server
- we want to preserve server resources and avoid the unnecessary roundtrips to the server
- we want to preserve Query String and Form Variables (optionally)
- we don't need to show the real URL where we redirected the request in the users Web Browser

Share Edit Follow Flag edited Sep 29, 2015 at 15:43 Brad Larson 170k 45 398 572 answered Nov 11, 2011 at 8:24 SoftDev 1,094 9 13

2 Much clearer, for me this is better as an accepted answer. – Baljeetsingh Sucharia Oct 28, 2018 at 6:23

28

Response.Redirect redirects page to another page **after** first page arrives to client. So client knows the redirection.

Server.Transfer quits current execution of the page. Client does not know the redirection. It allows you to transfer the query string and form variables.

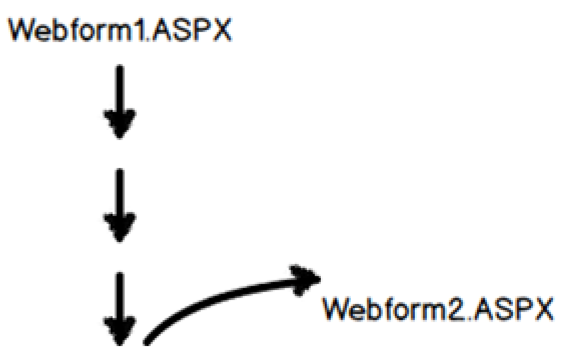
So it depends to your needs to choose which is better.

Share Edit Follow Flag edited Feb 6, 2009 at 18:36 answered Feb 6, 2009 at 18:29 Canavar 48k 17 92 123

1 Can a malicious user bypass `Response.Redirect` so as to load the original page even though I have called `Response.Redirect` ? – northben Apr 30, 2013 at 14:14

@northben - It's never easy to say "no" when it comes to technology as almost anything can be compromised - but in this case how could they - I would say no they could not...but I've been proven wrong many times in life. – JonH Nov 8, 2016 at 19:58

23



"response.redirect" and "server.transfer" helps to transfer user from one page to other page while the page is executing. But the way they do this transfer / redirect is very different.

In case you are visual guy and would like see demonstration rather than theory I would suggest to see the below facebook video which explains the difference in a more demonstrative way.

https://www.facebook.com/photo.php?v=762186150488997

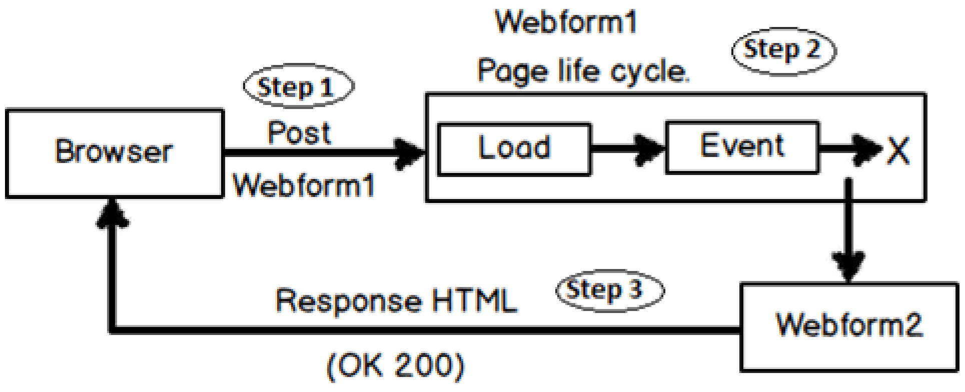
The main difference between them is who does the transfer. In "response.redirect" the transfer is done by the browser while in "server.transfer" it's done by the server. Let us try to understand this statement in a more detail manner.

In "Server.Transfer" following is the sequence of how transfer happens:-

1.User sends a request to an ASP.NET page. In the below figure the request is sent to "WebForm1" and we would like to navigate to "Webform2".

2.Server starts executing "Webform1" and the life cycle of the page starts. But before the complete life cycle of the page is completed "Server.transfer" happens to "WebForm2".

3."Webform2" page object is created, full page life cycle is executed and output HTML response is then sent to the browser.



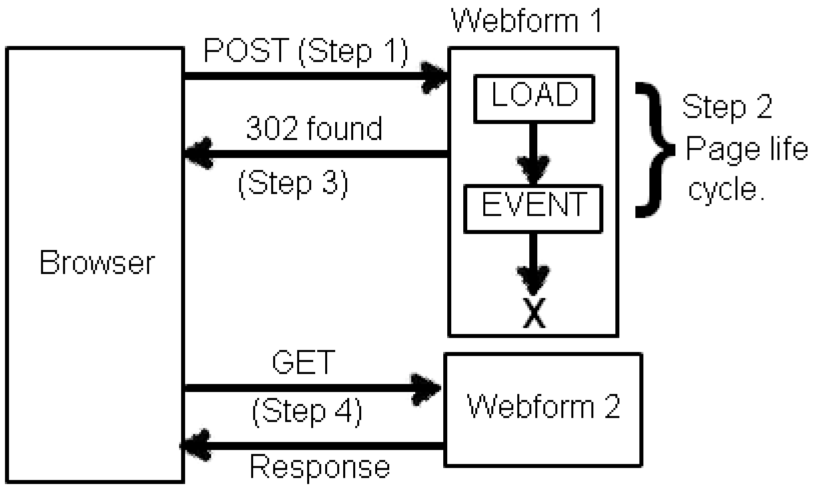
While in "Response.Redirect" following is the sequence of events for navigation:-

1.Client (browser) sends a request to a page. In the below figure the request is sent to "WebForm1" and we would like to navigate to "Webform2".

2.Life cycle of "Webform1" starts executing. But in between of the life cycle "Response.Redirect" happens.

3.Now rather than server doing a redirect , he sends a HTTP 302 command to the browser. This command tells the browser that he has to initiate a GET request to "Webform2.aspx" page.

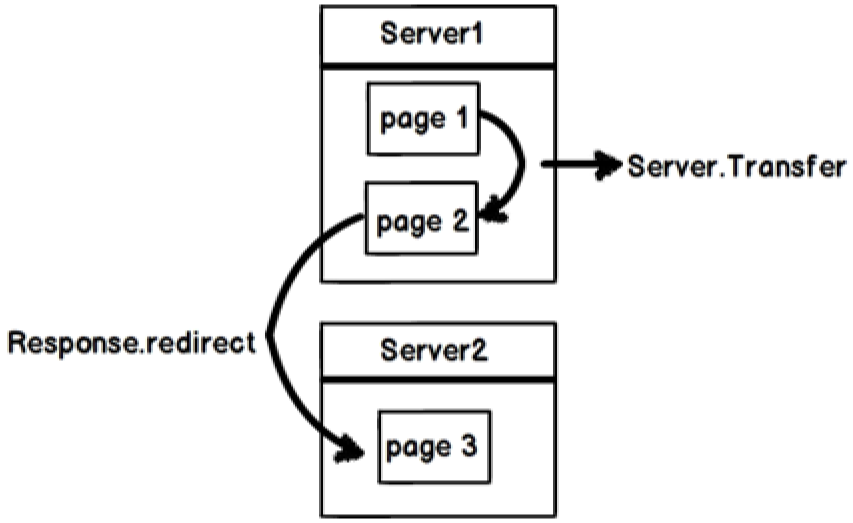
4.Browser interprets the 302 command and sends a GET request for "Webform2.aspx".



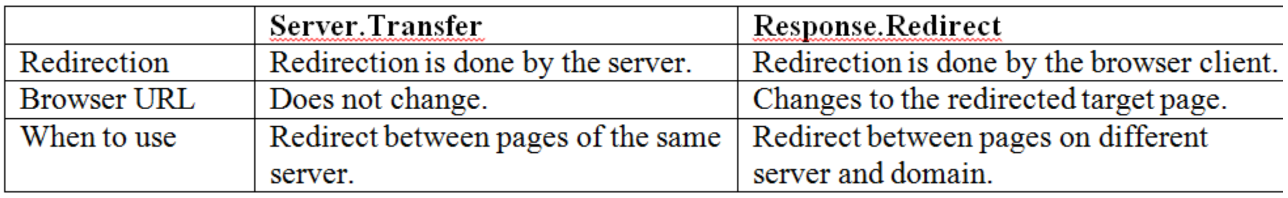
In other words "Server.Transfer" is executed by the server while "Response.Redirect" is executed by thr browser. "Response.Redirect" needs to two requests to do a redirect of the page.

**So when to use "Server.Transfer" and when to use "Response.Redirect" ?**

Use "Server.Transfer" when you want to navigate pages which reside on the same server, use "Response.Redirect" when you want to navigate between pages which resides on different server and domain.



Below is a summary table of which chalks out differences and in which scenario to use.



| | Server.Transfer | Response.Redirect |
|---|---|---|
| Redirection | Redirection is done by the server. | Redirection is done by the browser client. |
| Browser URL | Does not change. | Changes to the redirected target page. |
| When to use | Redirect between pages of the same server. | Redirect between pages on different server and domain. |

Share Edit Follow Flag

edited May 19, 2014 at 10:15

answered May 19, 2014 at 10:07

Shivprasad Koirala
**28.4k** 7 86 75

Useful when problems using ***Server.Transfer and Response.Redirect*** *stackoverflow.com/questions/1433448/thread-was-being-aborted* – Kiquenet Oct 2, 2015 at 15:56

For `Server.Transfer` : ***the same server*** or ***the same IIS web site*** ? – Kiquenet Jun 20, 2016 at 11:58

Could you please update the following paragraph because of at least 6 chars required for my edit: In other words "Server.Transfer" is executed by the server while "Response.Redirect" is executed by **thr** browser. "Response.Redirect" **needs to** two requests to do a redirect of the page. – paul cheung Nov 3, 2018 at 5:58

11

The beauty of Server.Transfer is what you can do with it:

```
TextBox myTxt = (TextBox)this.Page.PreviousPage.FindControl("TextBoxID");
```

You can get anything from your previous page using the above method as long as you use Server.Transfer but not Response.Redirect

Share Edit Follow Flag

answered Feb 25, 2013 at 16:38
Israel Margulies
**8,874** 2 31 26

10

In addition to ScarletGarden's comment, you also need to consider the impact of search engines and your redirect. Has this page moved permanently? Temporarily? It makes a difference.

see: Response.Redirect vs. "301 Moved Permanently":

> We've all used Response.Redirect at one time or another. It's the quick and easy way to get visitors pointed in the right direction if they somehow end up in the wrong place. But did you know that Response.Redirect sends an HTTP response status code of "302 Found" when you might really want to send "301 Moved Permanently"?
>
> The distinction seems small, but in certain cases it can actually make a big difference. For example, if you use a "301 Moved Permanently" response code, most search engines will remove the outdated link from their index and replace it with the new one. If you use "302 Found", they'll continue returning to the old page...

Share Edit Follow Flag

answered Feb 6, 2009 at 18:38
Diodeus - James MacFarlane
**114k** 33 160 179

The link doesn't work. Use this web.archive.org link instead. – stomy Oct 19, 2018 at 14:40

8

There are many differences as specified above. Apart from above all, there is one more difference. `Response.Redirect()` can be used to redirect user to any page which is not part of the application but `Server.Transfer()` can only be used to redirect user within the application.

```
//This will work.
Response.Redirect("http://www.google.com");

//This will not work.
Server.Transfer("http://www.google.com");
```

Share Edit Follow Flag

edited Oct 24, 2018 at 14:09
Mikael Dúi Bolinder
2,187 2 23 49

answered Aug 9, 2013 at 11:07
Microsoft Developer
5,399 22 94 142

6

Transfer is entirely server-side. Client address bar stays constant. Some complexity about the transfer of context between requests. Flushing and restarting page handlers can be expensive so do your transfer early in the pipeline e.g. in an HttpModule during BeginRequest. Read the MSDN docs carefully, and test and understand the new values of HttpContext.Request - especially in Postback scenarios. We usually use Server.Transfer for error scenarios.

Redirect terminates the request with a 302 status and client-side roundtrip response with and internally eats an exception (minor server perf hit - depends how many you do a day) Client then navigates to new address. Browser address bar & history updates etc. Client pays the cost of an extra roundtrip - cost varies depending on latency. In our business we redirect *a lot* we wrote our own module to avoid the exception cost.

Share Edit Follow Flag

answered Oct 22, 2008 at 5:42

community wiki
stephbu

5

Response.Redirect is more costly since it adds an extra trip to the server to figure out where to go.

Server.Transfer is more efficient however it can be a little mis-leading to the user since the Url doesn't physically change.

In my experience, the difference in performance has not been significant enough to use the latter approach

Share Edit Follow Flag

answered Oct 22, 2008 at 5:41

deadbug
**4,769** 4 26 23

4

**Response.Redirect:** tells the browser that the requested page can be found at a new location. The browser then initiates another request to the new page loading its contents in the browser. This results in two requests by the browser.

**Server.Transfer:** It transfers execution from the first page to the second page on the server. As far as the browser client is concerned, it made one request and the initial page is the one responding with content. The benefit of this approach is one less round trip to the server from the client browser. Also, any posted form variables and query string parameters are available to the second page as well.

Share Edit Follow Flag

answered Jan 24, 2012 at 1:43
Nick Kahn
**20k** 93 282 414

4

Server.Transfer doesn't change the URL in the client browser, so effectively the browser does not know you changed to another server-side handler. Response.Redirect tells the browser to move to a different page, so the url in the titlebar changes.

Server.Transfer is slightly faster since it avoids one roundtrip to the server, but the non-change of url may be either good or bad for you, depending on what you're trying to do.

Share Edit Follow Flag

answered Feb 6, 2009 at 18:38
krosenvold
**76.6k** 33 155 208

3

Just more details about Transfer(), it's actually is Server.Execute() + Response.End(), its source code is below (from Mono/.net 4.0):

```
public void Transfer (string path, bool preserveForm)
{
    this.Execute (path, null, preserveForm, true);
    this.context.Response.End ();
}
```

and for Execute(), what it is to run is the **handler** of the given path, see

> ASP.NET does not verify that the current user is authorized to view the resource delivered by the *Execute* method. Although the ASP.NET authorization and authentication logic runs before the original resource handler is called, ASP.NET directly calls the handler indicated by the *Execute* method and does not rerun authentication and

authorization logic for the new resource. If your application's security policy requires clients to have appropriate authorization to access the resource, the application should force reauthorization or provide a custom access-control mechanism.

You can force reauthorization by using the *Redirect* method instead of the *Execute* method. *Redirect* performs a client-side redirect in which the browser requests the new resource. Because this redirect is a new request entering the system, it is subjected to all the authentication and authorization logic of both Internet Information Services (IIS) and ASP.NET security policy.

-from MSDN

Share Edit Follow Flag

edited Jun 20, 2020 at 9:12
Community Bot
1 1

answered Sep 12, 2013 at 5:43
rockXrock
3,443 1 26 19

---

2

Response.Redirect involves an extra round trip and updates the address bar.

Server.Transfer does not cause the address bar to change, the server responds to the request with content from another page

e.g.

**Response.Redirect:-**

1. On the client the browser requests a page http://InitiallyRequestedPage.aspx
2. On the server responds to the request with 302 passing the redirect address http://AnotherPage.aspx.
3. On the client the browser makes a second request to the address http://AnotherPage.aspx.
4. On the server responds with content from http://AnotherPage.aspx

**Server.Transfer:-**

1. On the client browser requests a page http://InitiallyRequestedPage.aspx
2. On the server Server.Transfer to http://AnotherPage.aspx
3. On the server the response is made to the request for http://InitiallyRequestedPage.aspx passing back content from http://AnotherPage.aspx

**Response.Redirect**

*Pros:*- RESTful - It changes the address bar, the address can be used to record changes of state inbetween requests.

*Cons:*- Slow - There is an extra round-trip between the client and server. This can be expensive when there is substantial latency between the client and the server.

**Server.Transfer**

*Pros:*- Quick.

*Cons:*- State lost - If you're using Server.Transfer to change the state of the application in response to post backs, if the page is then reloaded that state will be lost, as the address bar will be the same as it was on the first request.

Share Edit Follow Flag

answered May 14, 2014 at 1:23
Mick
**6,764** 4 53 71

0

**Response.Redirect** Response.Redirect() will send you to a new page, update the address bar and add it to the Browser History. On your browser you can click back. It redirects the request to some plain HTML pages on our server or to some other web server. It causes additional roundtrips to the server on each request. It doesn’t preserve Query String and Form Variables from the original request. It enables to see the new redirected URL where it is redirected in the browser (and be able to bookmark it if it’s necessary). Response. Redirect simply sends a message down to the (HTTP 302) browser.

**Server.Transfer** Server.Transfer() does not change the address bar, we cannot hit back.One should use Server.Transfer() when he/she doesn’t want the user to see where he is going. Sometime on a "loading" type page. It transfers current page request to another .aspx page on the same server. It preserves server resources and avoids the unnecessary roundtrips to the server. It preserves Query String and Form Variables (optionally). It doesn’t show the real URL where it redirects the request in the users Web Browser. Server.Transfer happens without the browser knowing anything, the browser request a page, but the server returns the content of another.

Share Edit Follow Flag

answered Jul 20, 2015 at 13:33
Shailendra Mishra
**71** 10