
















Temporary Internet files Primer

On Windows Vista and above, Internet Explorer's Temporary Internet Files are maintained in two isolated WinINET cache containers. One cache is used for sites loaded in Protected Mode (Internet Zone and Restricted Zone) and the other cache is used for sites loaded outside of Protected Mode (Trusted Sites, Local Intranet, and Local Machine).

Each cache container consists of two components: a memory-mapped index database (index.dat) and a nested folder structure containing the response entities that have been cached. Each index holds up to 60000 entries, and each entry maps one request URL to a set of response headers, a bit of metadata, and optionally a file path to the response entity body (if one exists). The cache is cleaned (scavenged) when either the index entry limit is reached, or the disk quota ([250mb by default for IE9](#)) is exceeded. If the user deletes their browser history (Tools » Delete Browsing History) the cache index is overwritten with zeros, and all response entities are deleted from disk. If the user closes an InPrivate Browsing session, every item in the cache which was stored during the InPrivate session is removed.

When Internet Explorer asks WinINET to make a network request on its behalf, [if the request flags allow](#), WinINET will reuse a fresh response from the local cache, if one is available. If an expired response is available, WinINET will attempt to validate its freshness by making a conditional HTTP request in order to get back a **HTTP/304** if the response entity is still valid or a new copy of the response entity if the cached version is no longer up-to-date. If WinINET only has a partial response in the cache, it will issue a HTTP request with a **Range** header indicating the remaining part of the file which is not yet in the cache. In order to avoid corruption, the Range request will contain an **If-Range** header containing the **ETag** of the originally cached response, and/or a pre-RFC2616 **Unless-Modified-Since** header containing the Last-Modified time of the originally cached response. If the server's copy of the resource has changed, it will send the entire file again; if not, it will send a **HTTP/206** partial response containing only the requested range of the file.

Prior to IE6, Internet Explorer introduced a mechanism for viewing cached files; you can access this mechanism by clicking Tools » Internet Options » General » Browsing History » Settings. In the TIF and History Settings dialog, click the **View Files** button. Alternatively, you can simply type **shell:cache** into the Internet Explorer Address Bar or the Start » Run prompt. Doing this will open a Windows Explorer window to the **C:\Users\username\AppData\Local\Microsoft\Windows\Temporary Internet Files** folder.

Name	Internet Address	Type	Size	Expires
 1863_SP_renew_renew.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	5 KB	None
 1863_SP_renew_renew_now.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	5 KB	None
 1863_SP_renew_renewing_btm.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	1 KB	None
 1863_SP_renew_renewing_is_easy.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	4 KB	None
 1863_SP_renew_right.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	1 KB	None
 1863_SP_renew_stay-in-the-network.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	4 KB	None
 1863_SP_renew_stay-in-the-network_btm.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	1 KB	None
 1863_SP_renew_subscription_expires.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	3 KB	None
 1863_SP_renew_top-1.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	3 KB	None
 1863_SP_renew_top-2.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	6 KB	None
 1863_SP_renew_top-3.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	3 KB	None
 1863_SP_renew_top-4.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	4 KB	None
 1863_SP_renew_winning_together.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	2 KB	None
 1863_SP_renew_winning-together-top.jpg	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee JPG Image	4 KB	None
 1863_spacer.gif	http://image.email.microsoftemail.com/lib/fec9157976...	ACDSee GIF Image	1 KB	None

It is important to understand that what you see above is **not** exactly what is stored on disk—alert readers will observe that Explorer is showing columns, like **Internet Address** and **Expires**, that are not typically seen for other folders.

This is accomplished through the magic of a [Shell Namespace Extension](#). As explained on MSDN: *With a namespace extension, you can take any body of data and have Windows Explorer present it to the user as a virtual folder. When a user browses into this folder, your data is presented as a tree-structured hierarchy of folders and files, much like the rest of the Shell namespace.*

This folder is mapped to the Namespace Extension using the **desktop.ini** file within the folder; it contains the following text:

```
[.ShellClassInfo]
UICLSID={7BD29E00-76C1-11CF-9DD0-00A0C9034933}
```

The CLSID listed refers to a COM object implemented in IEFrame.dll. When the Namespace Extension is invoked, it generates the “friendly” view of the non-Protected Mode cache. It generates this view by making API calls into the WinINET cache code. The COM object enumerates the cache using [FindFirstUrlCacheEntry](#) / [FindNextUrlCacheEntry](#) without passing any filter; this means that files download by XDomainRequest, files temporarily cached while InPrivate, and [cached HTTP/3xx redirects](#), are not shown in this view.

The “Name” listed isn’t actually the name of the cache file on disk, but rather a filename simplistically parsed out of the URL. The HTTP expiration information and similar columns is retrieved from the metadata stored in the index.

Most importantly, this view does not show the Protected Mode cache; it only shows files that are downloaded outside of Protected Mode. In the screenshot, you'll see a number of Internet-Zone URLs; these are here because Internet Explorer's Medium Integrity "Frame Process" is downloading the Favicons; the rest of these pages are downloaded and rendered by the Low Integrity Protected Mode "Tab Process."

This view may even show files that do not exist; if the backing response entity file was deleted from the disk without calling [DeleteUrlCacheEntry](#) to delete the index entry, the Namespace Extension will still show the entry. However, if WinINET ever wants to reuse the entry, it will find the file missing and will need to re-download from the server.

To see the actual files on disk, open **shell:cache\Content.IE5**. You'll see the following view:

Name	Date modified	Type	Size
index.dat	3/19/2011 7:38 AM	DAT File	464 KB
desktop.ini	3/18/2011 7:01 AM	Configurat...	1 KB
C4ABNR0X	3/19/2011 7:35 AM	File folder	
ICSUYRXA	3/19/2011 7:35 AM	File folder	
MWT155A0	3/19/2011 7:30 AM	File folder	
Q0Y4OM79	3/19/2011 7:30 AM	File folder	

You see the index.dat database file, and four randomly named subfolders each of which, if opened, contains cached response files:

Organize ▾		Share with ▾	
Name	Date modified	Type	Size
Ad[1].tpl	3/19/2011 8:11 AM	TPL File	1 KB
answers_favicon[1].ico	3/18/2011 9:11 PM	Icon	2 KB
desktop.ini	3/18/2011 7:01 AM	Configurat...	1 KB
favicon[1].ico	3/18/2011 8:05 PM	Icon	2 KB
favicon[2].ico	3/18/2011 9:10 PM	Icon	2 KB
favicon[4].ico	3/18/2011 9:19 PM	Icon	15 KB

The folders are randomly named to mitigate certain types of attacks that involve placing malicious content at predictable file locations and then opening it using a url that uses the **file://** protocol scheme. There are four of them to similarly aid in randomness, as well as for legacy reasons (older filesystems had a limit on the number of files contained in a single folder).

The Protected Mode cache files can be viewed in a similar way, by opening **shell:cache\Low\Content.IE5** instead:

Name	Date modified	Type	Size
index.dat	3/19/2011 7:18 AM	DAT File	2,272 KB
desktop.ini	3/18/2011 7:01 AM	Configurat...	1 KB
BW79U9SN	3/19/2011 7:38 AM	File folder	
HPR49409	3/19/2011 7:38 AM	File folder	
T3KL80CF	3/19/2011 7:38 AM	File folder	
UFNQCAH8	3/19/2011 7:38 AM	File folder	

Again, you'll see the same layout, with four more randomly named subfolders.