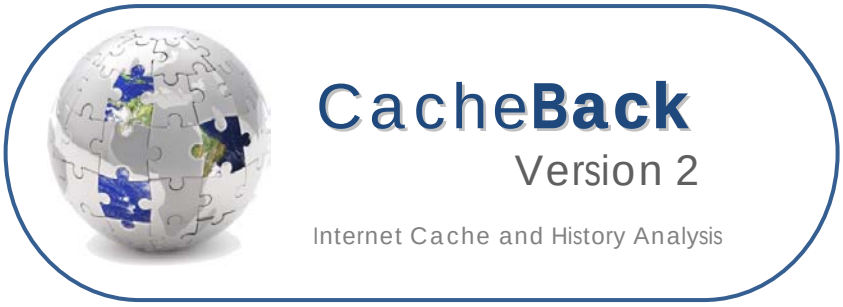
BitQuest Corporation
16715 – 12 Yonge Street, Suite 1079
Newmarket, ON L3X 1X4
Canada

Support: (905) 686-6801
Sales: (905) 836-0393

e-mail: training@cacheback.ca
web: www.cacheback.ca



User Reference Manual

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

We would like to take this time to express our sincerest appreciation for those individuals that assisted us in the review of this document. Despite busy schedules and commitments of their own, they found the time to read over this 200 page manual and provide us with their thoughts and opinions. The feedback that we receive from our users has been instrumental in helping us shape the content, format and delivery of this first, comprehensive user reference manual.

It is therefore, indeed, a pleasure to offer our very special thanks to:

- Brett Shavers, President
  e3Discovery, LLC
  Email: brett@e3discovery.com
  Website: www.e3discovery.com

  AND

- Louis M. Schlesinger, CEO
  CyForensics, LLC
  Address: 4741 Oxford Road, Macon, GA 31210, United States
  Tel: 478.731.0752 • Email: cyforensics@cyforensics.com • Website: www.cyforensics.com

Your assistance, gentlemen, has been greatly appreciated and we sincerely look forward to your continued support and interest in CacheBack!

Thank you!

INTRODUCTION

Welcome To CacheBack

Since the mid-1990's, the population of Internet users has grown exponentially. What used to be used only by only a select few, and even then many via dial-up connections, has now become an everyday necessity for both personal and business purposes. The Internet is the single most important communications vehicle we have today and in order to access this information, we rely on browser applications to make that connection. What has also evolved during this time period is the availability of "choice" by the consumer about which browsers they can use to navigate the World Wide Web. By this measure, digital forensic examiners have faced a dramatic rise in the quantity and quality of information afforded by a user's Internet history, and their browser's cache. Although there have been some efforts by vendors to provide support for the examination of Internet history, few if any have provided the much needed support to examine Internet cache as well.

CacheBack was created to fill this void in the forensic landscape by supporting law enforcement in their examination and analysis of Internet history and cache artifacts. CacheBack now provides an unrivaled support for today's top five (5) Internet browsers: Microsoft's Internet Explorer, Mozilla Firefox, Opera, Safari by Apple, and the latest addition with Google's Chrome.

CacheBack is a Windows based standalone application that was designed for use by computer forensic examiners charged with the investigation and analysis of Internet browser related artifacts. Although a large percentage of users of this software come from international, federal and state law enforcement and government agencies, it is equally in demand by private corporations, educational institutes, corporate security professionals, and private investigators.

When contemplating the idea of having to present evidence to a jury which involves visited Internet websites, it begs the question: "Is a picture not worth a thousand words?" Prior to CacheBack, Internet history evidence was typically disclosed to prosecutors in a spreadsheet (table) format which was rather unintuitive, and verbose. In addition, what was missing from this disclosure was the related visual data offered by cached web pages on a user's system. Unfortunately, there have been very little progress made in the forensic community to reconstruct these pages, let alone in a forensic context. Moreover, any visual appreciation of the original web pages visited is lost and a lesser weight is quite often afforded the evidence. CacheBack proposed to change that perception.

Today, with CacheBack, judges and jurors can immediately appreciate the *facts in issue* of a case (eg: Child Pornnography) when they can visually identify with the evidence. Seeing a web page "as it was viewed by the user" provides a much greater understanding by the viewer. Typically, web pages that are stored by browsers, in their discoverable format, are usually missing graphics, special formatting, custom colors and fonts. Without these ingredients, the result is a poor arrangement of text, usually unformatted and in some cases, missing entirely. While the option to rebuild a web page *by hand (manually)* exists, provided of course the user has the required knowledge to do so, this process could take anywhere from hours to days to finish. This is obviously not a logical or smart use of someone's time, especially considering that cases may uncover several hundred web pages of an evidentiary nature. This inherent time delay (in rebuilding web pages manually) is due to the manual decoding processes and re-construction tasks that need to be exercised.

Fortunately, CacheBack solves this problem in a just the click of a mouse! CacheBack starts by importing and parsing all user-defined Internet cache and history artifacts. Users can then quickly peruse the imported data in either a Table (tabular) or Gallery (thumbnail) view format. By simply clicking on a record and then choosing the "Rebuild" button on the toolbar, the web page is rebuilt in seconds! What was once an unintelligible jumble of text is now a rich, colorful piece of art that not only tells a story, but shouts it out! CacheBack completely removes the user from the complexities of having to decipher each respective browser's cryptic indexing system. Everything from a forensic context is managed, documented and provided to the user for reporting purposes. No understanding of how to decode or decipher browser artifacts is required. Having said that, of course, having that understanding only completes the forensic understanding of how CacheBack works. In any event, this User Reference Manual will reveal these hidden secrets and provide you with the skills and training necessary to validate the rebuilding process (if necessary), and speak to the technical details in any report or testimony.

If you are new to CacheBack, then what you should know first and foremost is that this software provides very powerful features that require you to know some essentials before getting started. CacheBack was designed to be very granular in its approach to data processing, data analyzing, and reporting. As such, many of the display and analyzing functions usually involve more than one step. Failing, for instance, to simply "checkmark" certain rows in the Table may prevent the associated URL Records from being included in a report. On the contrary however, CacheBack was designed visually in a manner that closely resembles other popular and well used programs, such as Outlook and Windows Explorer.

Apart from *this* User Reference Manual, we have expanded our support offerings by providing a series of online video tutorials which are available at our website (www.cacheback.ca). It may even be a good idea to review the Introduction video first to get a quick idea of what this manual will be covering.

History of CacheBack

CacheBack's roots go back to the year 2001, long before the arrival of other third party History only tools, and at or about a time when Guidance Software released Version 2.0 of its EnCase software which touted a wonderful new feature called "EnScripts". This new release of EnCase promised new investigative opportunities using their "EnScript" technology, especially for Internet related investigations. What was significant about the release of Version 2 was the fact that EnCase came with an Internet History EnScript which was the first step towards automating the discovery of Internet histories, in a robust and efficient manner. The initial results however were confined to dealing with just Internet histories, and there was little in the way of options to query, format, filter or analyze the results.

Consequently, due to the lack of third party tools available to examine Internet history data at that time, BitQuest's lead developer: John Bradley, (a police officer at the time) created a Windows standalone tool called "NetDigger". It was a Windows based application in combination with a Microsoft Access database that was used to parse and analyze Internet Explorer and Netscape history files. NetDigger was given away for free to law enforcement and government agencies over the next few years that followed. Unfortunately, continued development for NetDigger went by the wayside due to the limited changes in browsers at that time.

In 2004, John began working on an entirely new product called "CacheBack". The program was developed slowly over the next few years and was released publicly in early 2006. In the summer of 2007, Version 2 of the software was made available which provided a greatly enhanced user interface. It also marked the introduction of a dongle-based licensing schema and showed a commitment to continued growth and support for the product.

By mid-year 2008, CacheBack had a growing user-base and a series of major improvements were distributed throughout the year through a series of dot releases (revisions). These features included a new horizontally split, tabbed interface with multiple viewer tabs and a Gallery option to view thumbnails of HTML files (web pages). This made working with Internet artifacts much more granular and the visual element in particular, provided by the Gallery feature, was greatly welcomed.

From late 2008 through to May 2009, an intensive research and development initiative was underway to build in support for all five (5) top Internet browsers, namely: Internet Explorer, Firefox, Safari, Opera and Google Chrome. On May 1st, 2009, CacheBack and it's copyright owner: Digital Investigations Group Inc. were acquired by BitQuest Corporation. John joined BitQuest as President and Co-Founder and presently leads the development team as their Chief Software Architect.

Since that time, BitQuest has, and will continue to devote, *significant* resources in expanding the functionality of the software, including support for new features, customer support and user training.

FEATURES LIST

CacheBack Key Features

CacheBack is a powerful Internet cache and history analysis tool that comes with a variety of features that simplify the forensic examination process. The following are highlights of the core features available with the software.

- Support for today's top five (5) Internet browsers: Internet Explorer (5-8), Firefox (2-3), Opera (9), Safari (3) and Google Chrome (1).
- Rebuild cached web pages instantly by the simple click of a mouse.
- Gallery displays thumbnails of web pages and picture files for visual analysis of cache files.
- Rich HTML reports for rebuilt web pages include thumbnail hyperlinks to each rebuilt page.
- Rebuild web pages from Unallocated Clusters using new Templates feature (coming).
- Imports older NetAnalysis case files and leverage the superior reporting features in CacheBack.
- Imports results from the NetAnalysis "HistEx" tool (eg: URLs recovered from unallocated space).
- CacheGrab® V5/V6 EnScripts to expedite the recovery of cache and history artifacts within EnCase.
- Comes with the CacheGrab® standalone tool to quickly collect cache and history artifacts from a local drive or mounted evidence file, or virtual file system. Results are readied for import to CacheBack.
- Displays timestamps in any time zone using Coordinated Universal Time (UTC).
- UTC option takes into account new DST rules and Eastern hemisphere differences (eg: Australia).
- Powerful Link Analysis technology to identify relationships between web pages and history URLs.
- Produce compelling HTML reports, complete with thumbnails and metadata.
- Familiar user interface similar to Windows Explorer or Microsoft Outlook.
- Each URL is extracted and broken down into multiple artifacts (metadata) which can be defined in custom queries.
- Filter results based on file type, file extension, domain/host name.
- Create user-defined SQL queries and save them for future use.
- Pre-defined queries now available for common type investigations.
- Built-in offline browser to view web pages.
- Rebuilds web pages on the fly without altering the original files.
- CacheBack Project (.CBP) files (*aka: case files*) use Microsoft Access database engine thereby extending CacheBack's own internal querying and reporting capabilities.
- Export Table columns as plain text files (eg: comma separated values).

- Built-in picture viewer with new Zoom In/Out, Copy and Print functionality.
- Multiple tabbed views of the same evidence (Browser, Text, Hex, Picture, Audit and Links).
- *** Free technical support and unlimited upgrades to all dot releases.

Advanced Features

The following are some of the more advanced features available with CacheBack.

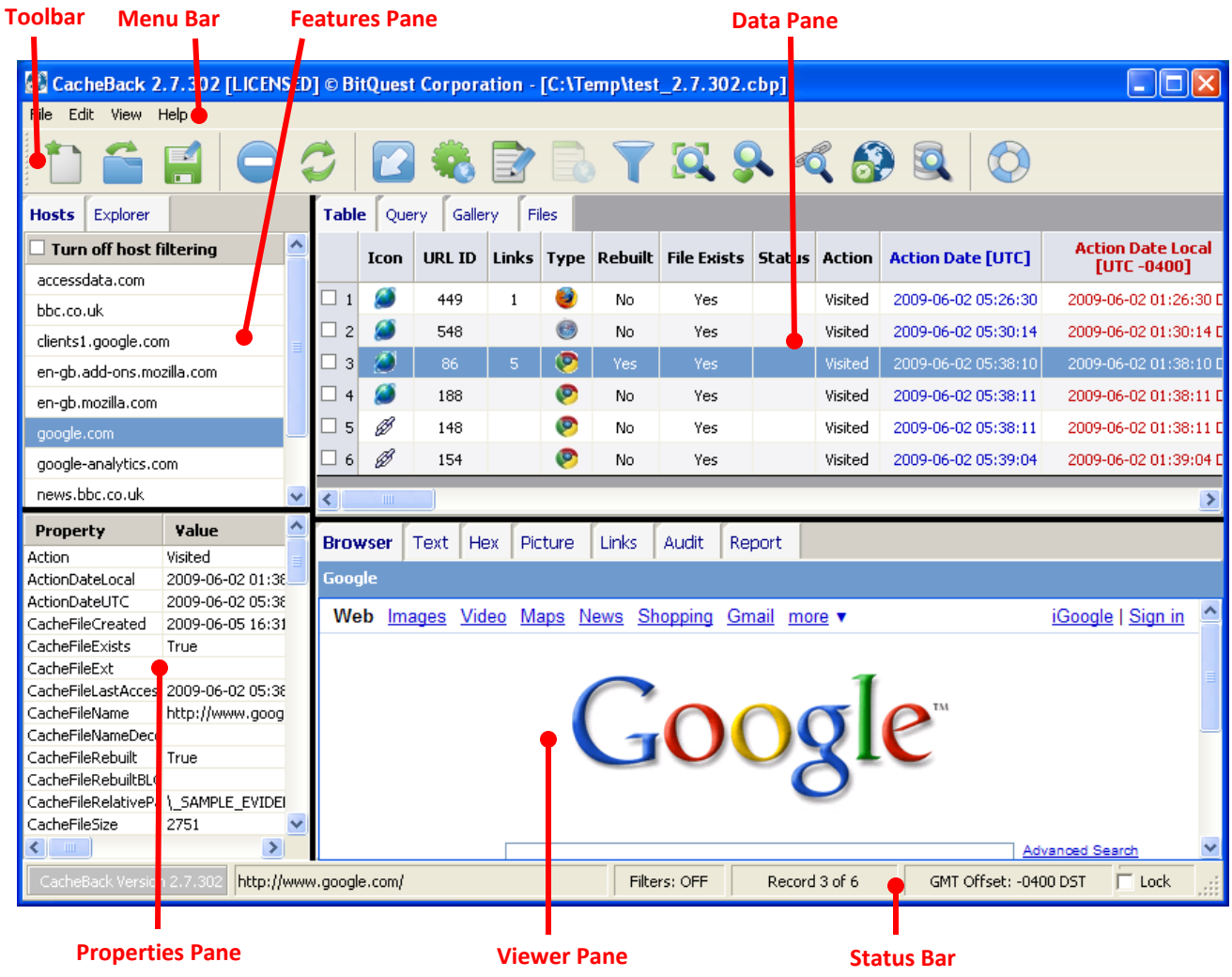
- Thumbnails of cached web pages and pictures are now stored inside the project file for increased performance.
- Rebuilt web pages are now compiled into proprietary cabinet files, encrypted, and stored inside the CacheBack project file. This translates into a huge performance increase with the added security of keeping the evidence together.
- SQLite3 cache and history database support (for all browsers where applicable).
- New consolidated HTML (web page) and Picture Gallery.
- Introduction of the PIART algorithm to *greatly* narrow search results for Picture related investigations (PIART is the Photographic Image Aspect Ratio Theory).
- New Properties Pane in the GUI to more readily display attributes about the currently selected URL, including new Exif metadata for photos.
- Performance Optimization option increases data processing/importing by at least 50%.
- System-clock-independent, multiple Time Zone and customizable timestamp format for accurate timeline analysis.
- New Query tab makes it easier than ever to create/save simple and complex search queries.
- Query SQL syntax is now pre-validated before being saved to the database – much faster execution!
- Enhanced filter options to drill down on large datasets (making them smaller).
- Improved forensic metadata collection and presentation (more info for Table view, easier user ability to manually validate artifacts).

UNDERSTANDING THE GRAPHICAL USER INTERFACE

Introduction

With the development of Version 2.7, CacheBack underwent a significant design change with the way it managed and displayed data. The following is a screenshot of the current user interface. As you can see, there are now four (4) distinct user panes: the Features Pane (top left, visible now by default), the Data Pane (top right), the Properties Pane (bottom left, new), and the Viewer Pane (bottom right).

The Toolbar has also undergone some changes with the redesign and resizing of toolbar button graphics. The lower Status Bar has also had some minor updates with an increase in the number of status labels.

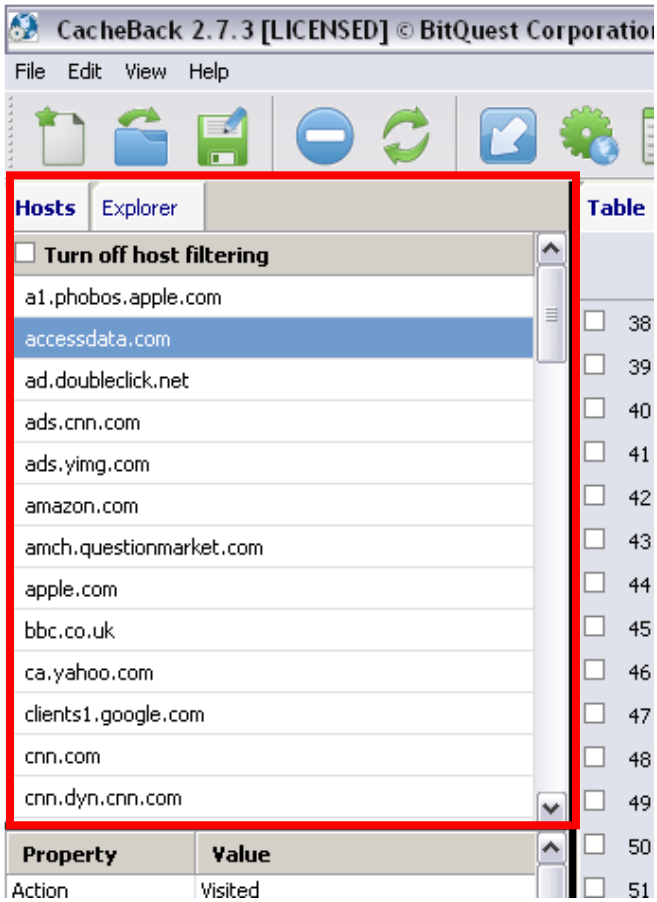


User Panes

CacheBack now uses a total of four (4) user panes to manage and display data. It is important to understand the role each pane takes in the use of the software, and how users should interact with their contents.

Features Pane

This pane is located in the top left hand corner of the main window and contains the Host Tab and the Explorer Tab.



Host Tab

The Host tab contains a single column of distinct host (domain) names that have been extracted from all imported cache and history URLs. The purpose of this list is to add a "2nd level" query overtop of the current, *underlying query*. The *underlying query* is what controls the records or recordset displayed on the Table and Gallery tabs.

The *underlying query* is the rock bottom query that controls what data is returned from the database.

The Host filtering feature *adds a query on top of the underlying query* and this has been hard coded internally. What this means essentially is that any selection in the Host tab (is a query) which is a query *that is called on top of the underlying query* (otherwise known as a "query calling a query").

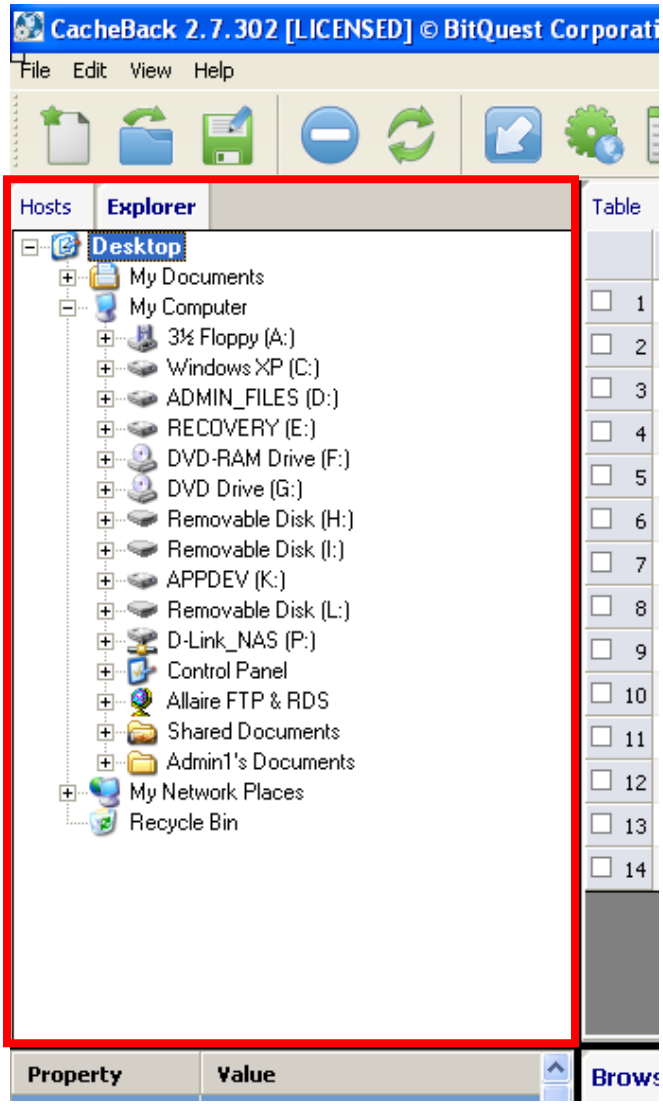
This translates into a significant performance increase in returning specific data which is a huge improvement over using just Filters. Although the Host filter option could have been implemented as a Filter VS. a Query, it was believed that Host filtering would be used so frequently that it warranted the added performance gains.

Turn off host filtering

Turn off host filtering is the only way to revert back to the *underlying query only* AFTER a Host item has been selected. Once a checkmark is placed next to "Turn off host filtering", users MUST click on the Refresh button on the Toolbar in order to refresh the current recordset using just the *underlying query*.

Explorer Tab

The Explorer tab is a simple hierarchical view of the all the drives, folder and files that are currently available to the user's system. By navigating the Explorer Tree, users can explore the contents of certain folders which will then populate the hidden Files Tab in the Data Pane. From there, items in the Files grid on the Files tab can then be selected (similar to the Table tab) to display the file's contents in one of the Viewer tabs below.



Data Pane

This pane is located in the top right hand corner of the main window and contains the following tabs: Table, Query, Gallery, and Files.

Table Tab

Of all the tabs in the GUI, this is the most important as because it manages and presents all of the evidence that is currently available to be examined. What is important to understand, first and foremost, is that the contents of the table are driven by the *current underlying query* (see Query Tab).

For example, the current project file could potentially have a total of 100,000 records in the current project (database) file. By default, ALL of these 100,000 records will be displayed in the Table. This corresponds directly with the *Default Query* that is created with each new project file.

The Default Query is "SELECT * FROM URLs ORDER BY ActionDateLocal ASC". In plain English, this query statement reads like this: **Please select ALL records (*) from the database table called *URLs* and ORDER the results using the *ActionDateLocal* column, in ASCending order.**

IMPORTANT:

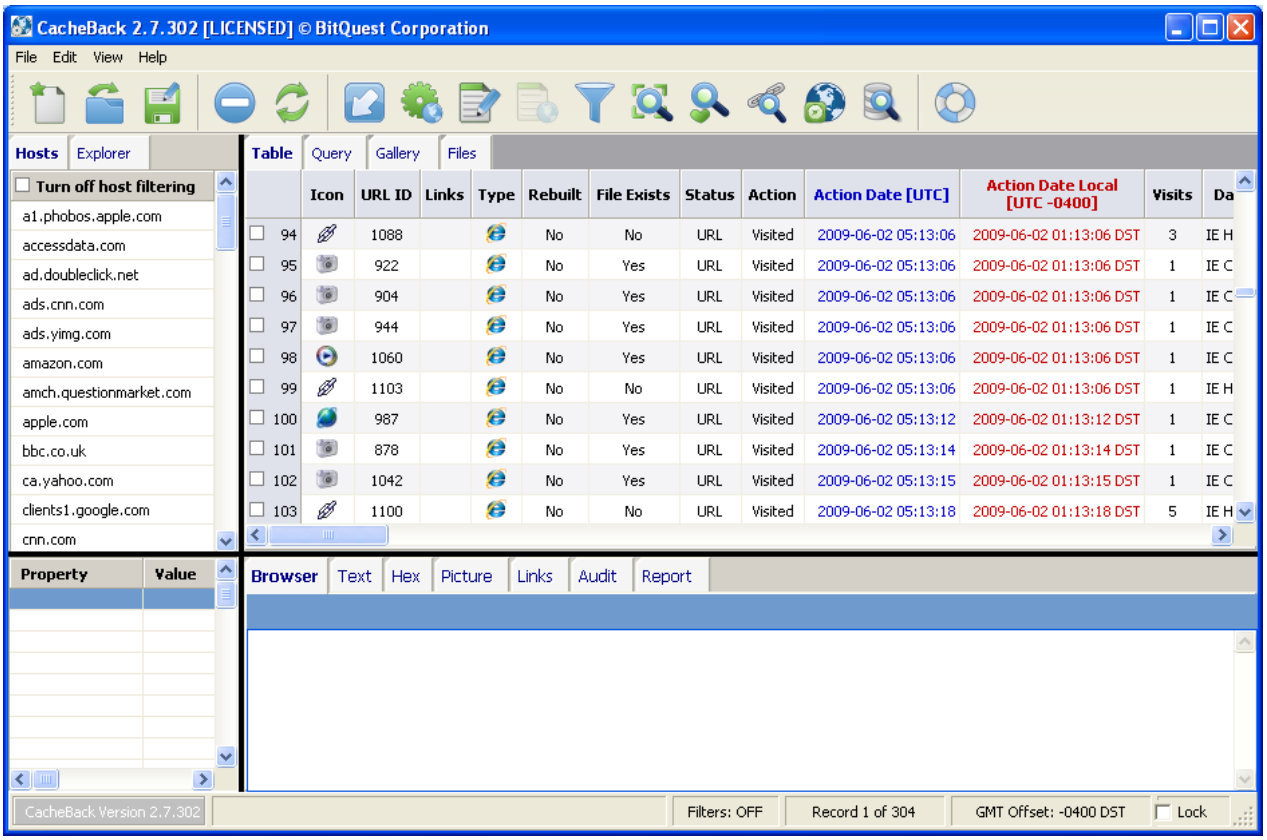
Since a subject's computer's Time Zone and Daylight Saving settings may not be set correctly (eg: defaulting to Pacific Standard Time during Windows install), CacheBack will initially convert all browser UTC times to *local time* during the IMPORT stage using the *user's (investigator's) workstation* time zone settings (eg: Project file default).

These converted timestamps are converted and stored into the *ActionDateLocal* database column. This *assignment to the ActionDateLocal column* is done ONCE and cannot be altered after import. The logic behind this approach is to allow users (investigators) to conduct custom queries using the *ActionDateLocal* time that is relative to the time zone in which the investigation is taking place. This also works around the significant possibility that the user's computer has not been changed from the Windows installation default of Pacific Standard Time which is -08:00 UTC.

All *ActionLocalUTC* times "displayed" (eg: Table view) and "reported" are on-the-fly *ActionDateUTC* conversions based on the currently selected Project-file-level time zone setting (which is selected via the Options Window). This approach ensures that times are *more true* than relying on (and having to convert) the subject's local timestamps.

CacheBack may at some point in the future require a user to manually configure the subject's Windows Registry time zone information during the initial import stages. However, at the present time, we see this is a flawed approach to analysis given the above noted argument.

The following screenshot demonstrates the Default Query results after importing all of the Sample Data (that is provided with installation of CacheBack).



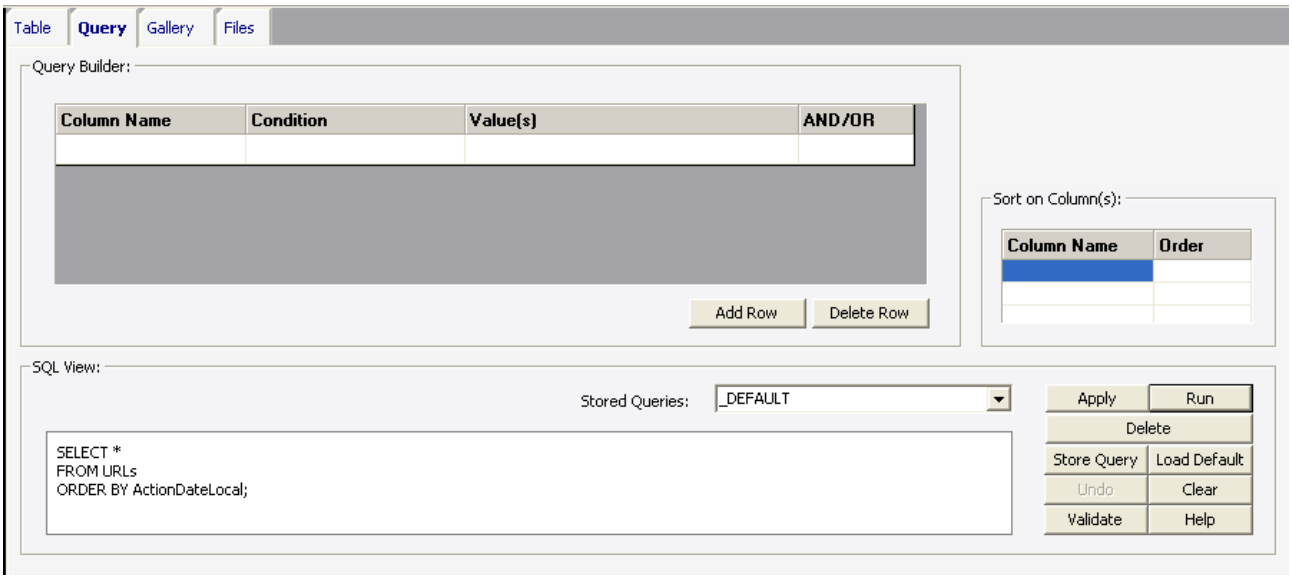
Additional information about how to navigate and use the Table effectively are discussed later on in the section entitled "User Interface: Advanced".

Query Tab

The Query tab is the 2nd tab in the Data Pane and it is here that ALL queries are managed, including and especially, the *underlying query and the Default Query*.

The new Query tab is where users can create and edit custom queries, or select and run any query from a list of pre-defined queries stored in each new CacheBack project file.

The following is a screenshot of the Query tab which showcases the new Query Builder tool and SQL Statement box. These two options are how queries are managed, which are discussed in greater detail later on in this manual.

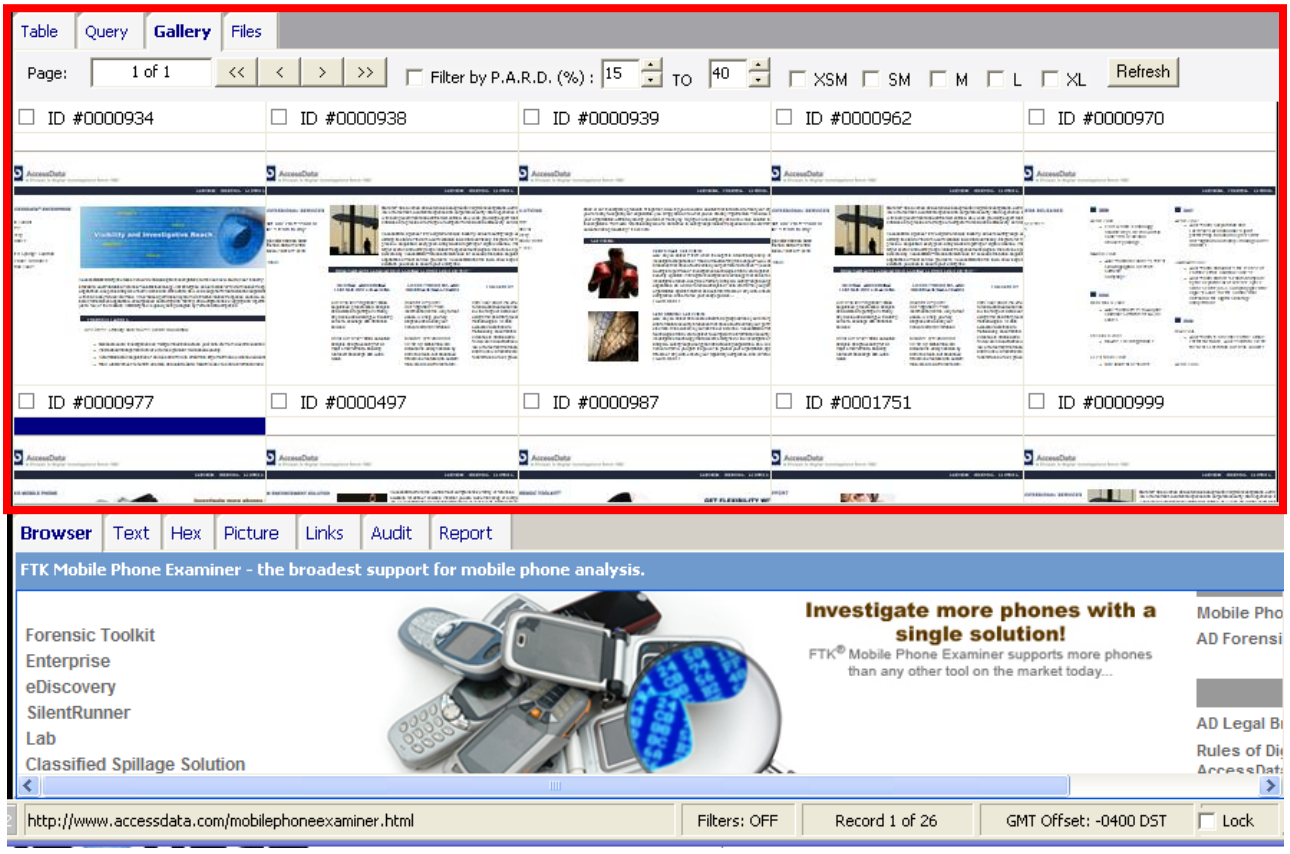


Gallery Tab

The Gallery tab is the 3rd tab in the Data Pane and it is here that cache web pages and picture files can be represented in the form of thumbnails. Having a consolidated Gallery now allows examiners to sift through cache items more thoroughly and cue in on related websites.

The ability to use thumbnails to view the data is helpful from the sense that quite often, the human eye can pick up on artifacts more easily when rendered in a picture format. For example, common websites such as Microsoft's MSN Home Page and CNN has familiar color schemes, layouts and formatting of graphics and photographs. Having the ability to peruse a cache folder based on a thumbnail version of the original file can expedite an examination where visual presentations are the familiar.

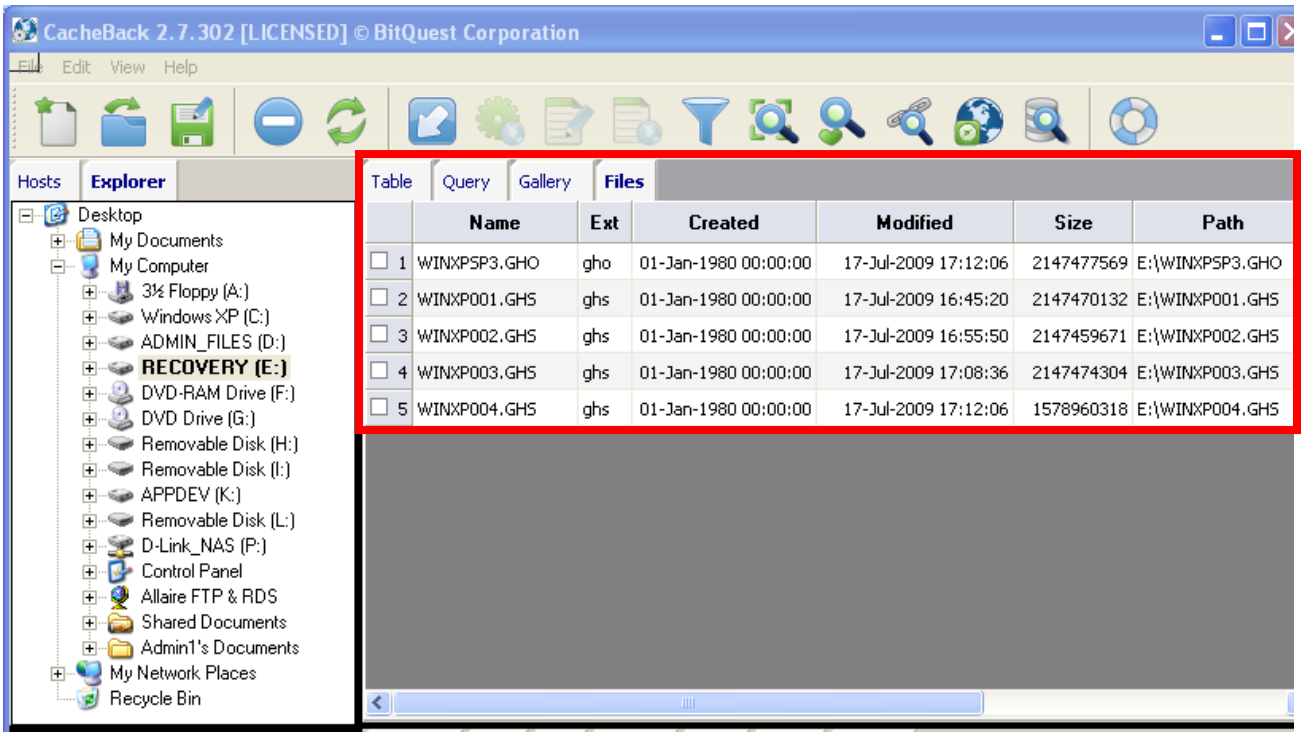
The following is a screenshot of the Gallery which was created using the Sample Data that comes with CacheBack.



Files Tab

The Files tab is the 4th and last tab in the Data Pane. The contents of the Files tab are populated based on the selections made in the Explorer tab (which is found in the Features Pane).

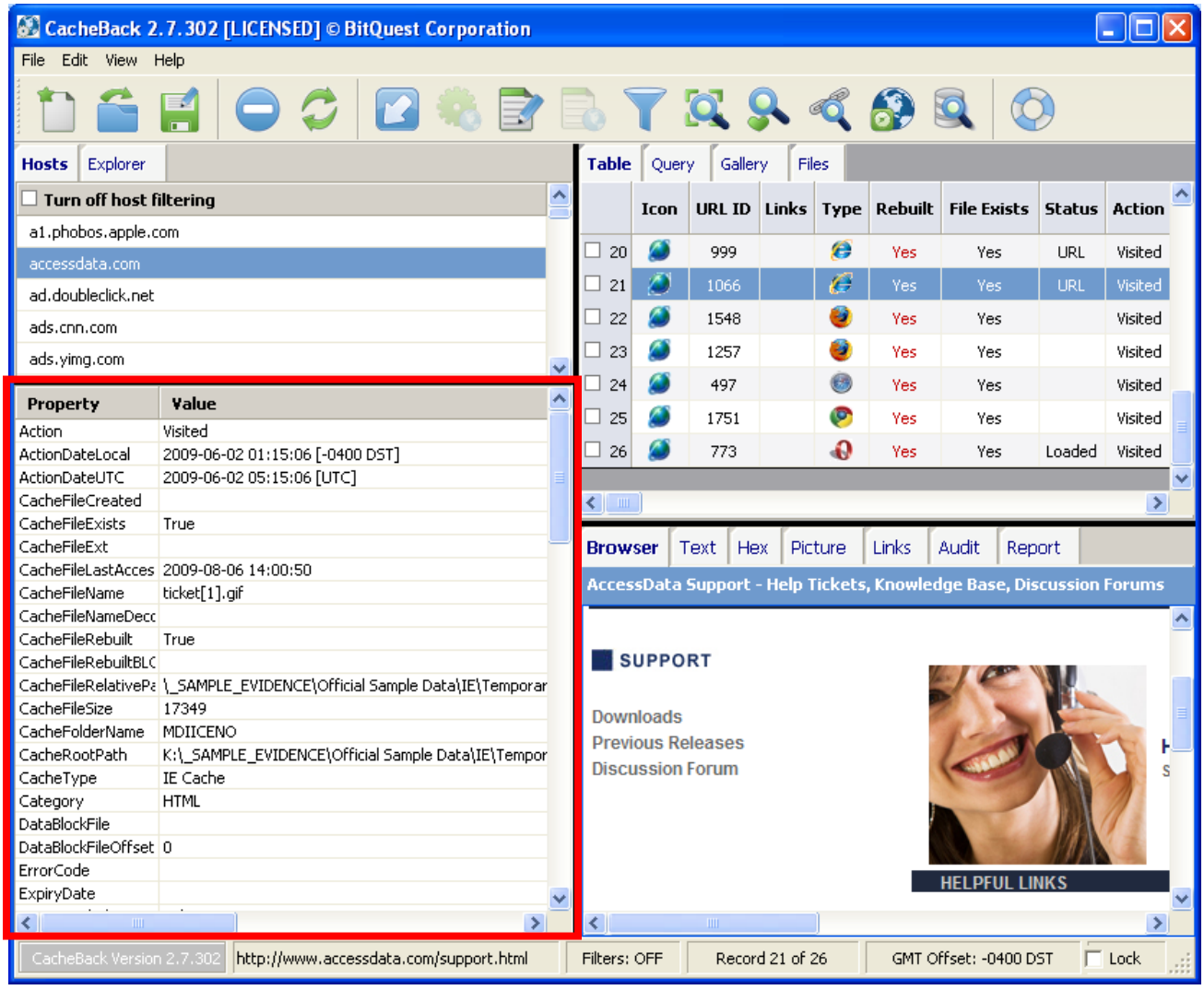
The following is a screenshot of the Files tab.



Properties Pane

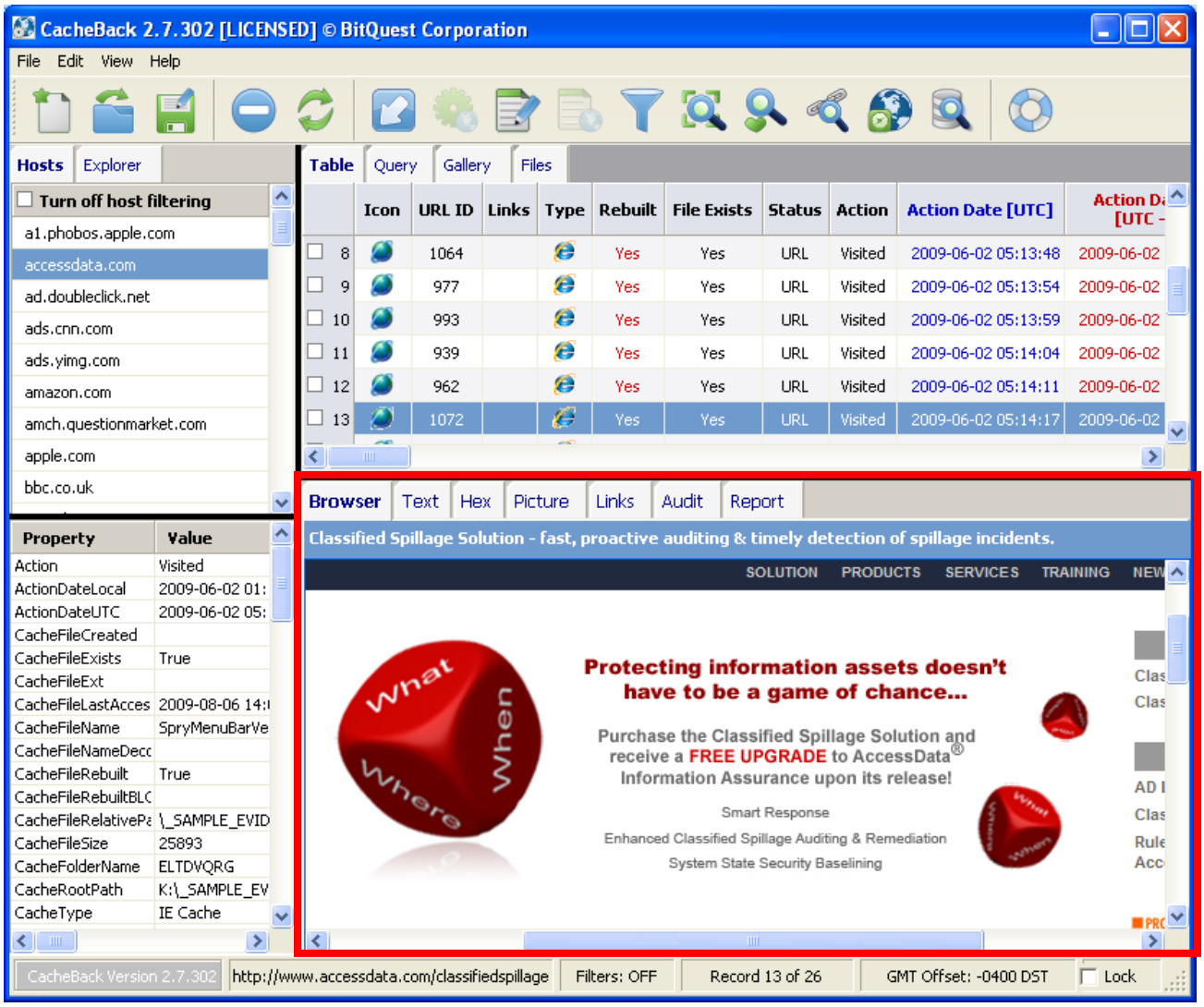
This pane is located in the bottom left hand corner of the main window and contains the Properties Table. The contents of the Properties table are loaded each time a URL record is selected, either from the Table tab, or the Gallery tab.

The following is a screenshot of the Properties pane. In this particular example, the Properties Table reveals a extensive quantity of metadata for the currently selected Table record.



Viewer Pane

This pane is located in the bottom right hand corner of the main window and contains the following Viewer tabs: Browser, Text, Hex, Picture, Links, Audit and Report. Details about what each viewer provides and how each functions is covered later on in the section entitled: “User Interface: Advanced”.



Browser Tab

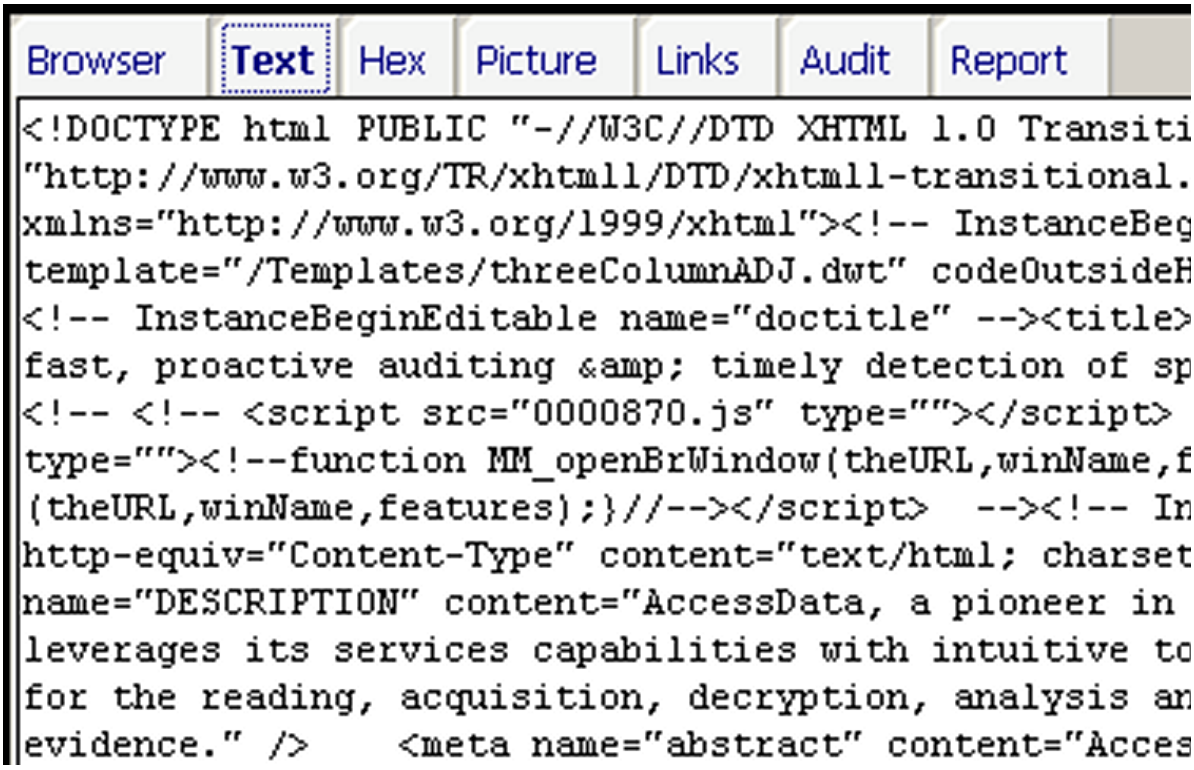
The Browser tab provides an offline browser which is used to load web based file that is associated to the currently selected URL record entry (either from the Table or the Gallery). It supports the same context menu (right-mouse-click popup menu) features offered by Microsoft Internet Explorer. Consequently, any configurations for Internet Explorer on the user's workstation will be adopted by the browser in CacheBack. This means that Advanced Internet Options for IE such as "Disable debugging..." will be adopted by the browser control in CacheBack. This provides users with the added ability to control cache content within CacheBack where content may contain potentially harmful or annoying code (eg: popup windows).

The following is a screenshot of the Browser tab.

Text Tab

The Text tab provides a “plain text” view of the contents of any file (eg: html, picture, etc.) that is associated with the currently selected URL record in Table or Gallery view. This view will take into account any carriage returns and new line feeds in the document being displayed. Users can sweep the text with their mouse and use the keyboard CTRL + C to copy the currently highlighted text to the Windows Clipboard.

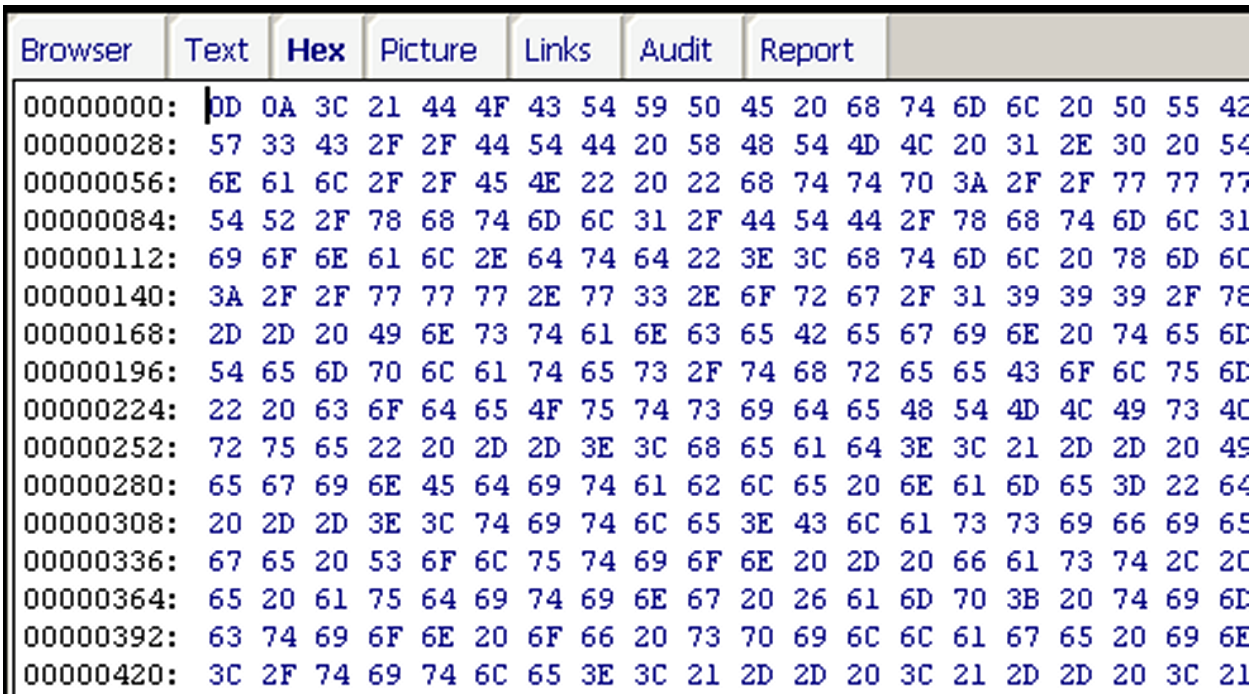
The following is a screenshot of the Text tab.



Hex Tab

The Hex tab provides a “hexadecimal” view of the contents of any file (eg: html, picture, etc.) that is associated with the currently selected URL record in Table or Gallery view. Users can sweep the text with their mouse and copy the contents of the highlighted text to the Windows Clipboard. This can be done by either using the keyboard’s CTRL + C key combination, or by simply right-mouse-clicking in the viewer area to invoke the associated context menu. Using the latter option will display an additional option to Print the current contents.

The following is a screenshot of the Hex tab.



Picture Tab

The Picture tab displays the picture file that is associated to the currently selected URL record (either from the Table or the Gallery), provided of course that the file is in fact a picture. The Picture tab contains new Zoom controls to enhance to display of the picture. These include Zoom In, Zoom Out, Rotate 90 degrees, Rotate 180 degrees, and Zoom To Fit options. Users can also Print the picture, save it to a file, or copy it to the Windows Clipboard.

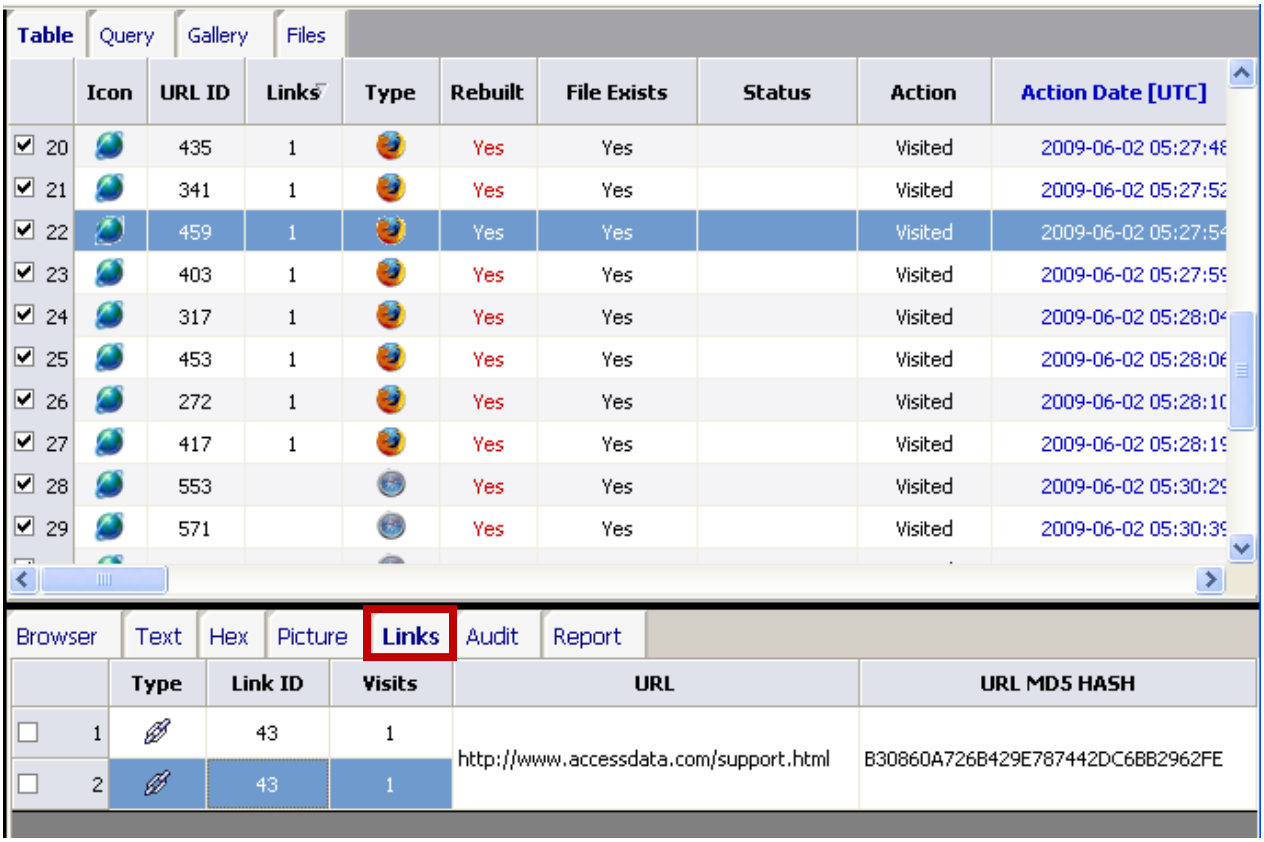
Supported file types for displaying pictures include .JPEG, .BMP, .GIF and .PNG.

The following is a screenshot of the Picture tab.

Links Tab

The Links tab provides a “table view or list” of the URLs that are associated with the currently highlighted URL record from the Table. If the URL record selected in the Table is a web page, then the URLs listed in the Links tab will contain “History” URLs only. If the URL record selected in the Table is a “History URL”, then the contents of the Links tab will be comprised of URLs from “web pages” only. Using this tab, users can quickly establish known relationships between web pages and history URLs. These relationships can help determine which if certain “links” on a web page were (possibly) selected. Any established relationships can be included in the Report by checking the appropriate option on the Advanced Options tab located on the Report window.

The following is a screenshot of the Links tab.

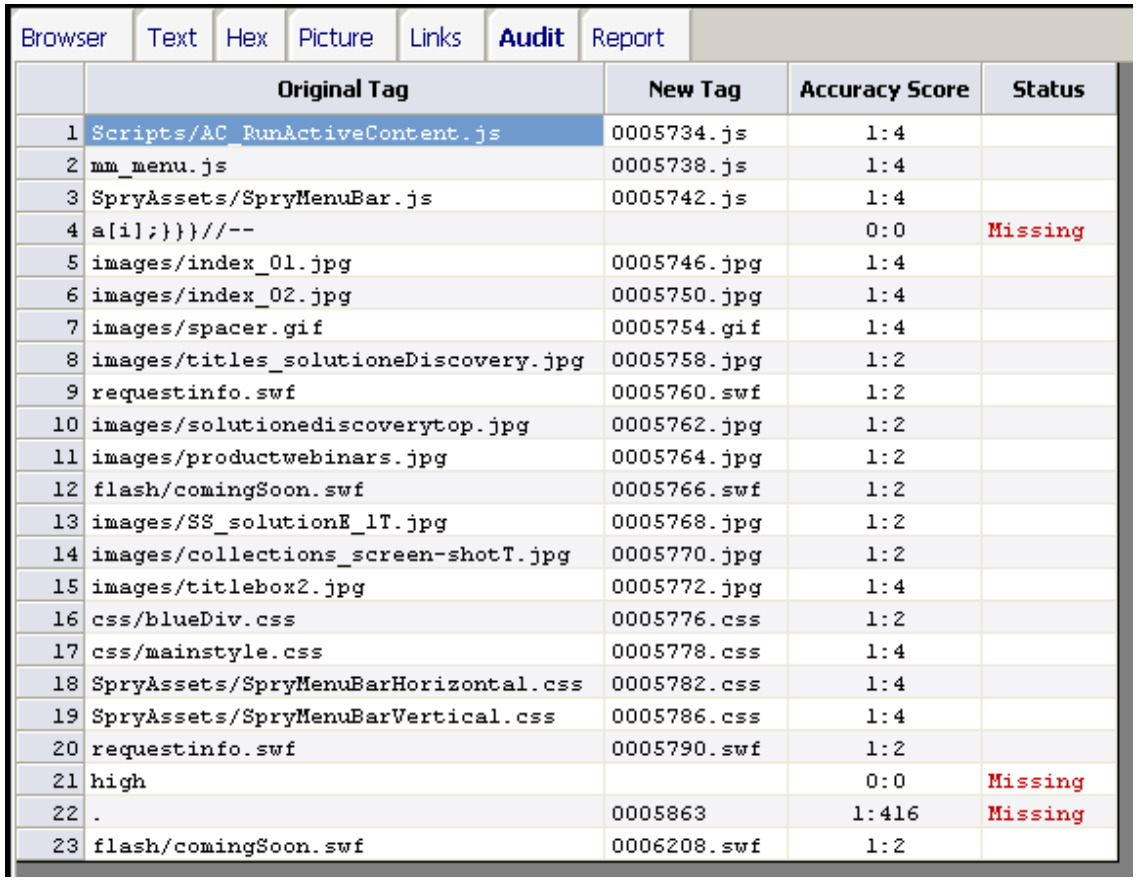


Audit Tab

The Audit tab provides a “table view or list” of the “tags” in a web page that have been “replaced”, along with the “new tag name/value”. It also provides an “Accuracy Score” (eg: “1:4”) which is an indicator of how many options CacheBack had to choose from before making its selection. Any items not replaced or “new tags not found” are indicated by the word “Missing”. The Audit details can be included in any Report by selecting the appropriate option under the Advanced Options in the Report window.

NOTE: It is common to come across “Original Tag” values that either do not exist or seem unintelligible. This is simple a result of CacheBack’s parsing engine finding a match in the web page source code. Although certain portions of the source code may be qualified as a parsed item, it will be excluded during the rebuilding process due to the added ‘type checking’ processes in place.

The following is a screenshot of the Audit tab.



	Original Tag	New Tag	Accuracy Score	Status
1	Scripts/AC_RunActiveContent.js	0005734.js	1:4	
2	mm_menu.js	0005738.js	1:4	
3	SpryAssets/SpryMenuBar.js	0005742.js	1:4	
4	a[i];}}}//--		0:0	Missing
5	images/index_01.jpg	0005746.jpg	1:4	
6	images/index_02.jpg	0005750.jpg	1:4	
7	images/spacer.gif	0005754.gif	1:4	
8	images/titles_solutioneDiscovery.jpg	0005758.jpg	1:2	
9	requestinfo.swf	0005760.swf	1:2	
10	images/solutionediscoverytop.jpg	0005762.jpg	1:2	
11	images/productwebinars.jpg	0005764.jpg	1:2	
12	flash/comingSoon.swf	0005766.swf	1:2	
13	images/SS_solutionE_lT.jpg	0005768.jpg	1:2	
14	images/collections_screen-shotT.jpg	0005770.jpg	1:2	
15	images/titlebox2.jpg	0005772.jpg	1:4	
16	css/blueDiv.css	0005776.css	1:2	
17	css/mainstyle.css	0005778.css	1:4	
18	SpryAssets/SpryMenuBarHorizontal.css	0005782.css	1:4	
19	SpryAssets/SpryMenuBarVertical.css	0005786.css	1:4	
20	requestinfo.swf	0005790.swf	1:2	
21	high		0:0	Missing
22	.	0005863	1:416	Missing
23	flash/comingSoon.swf	0006208.swf	1:2	

Report Tab

The Report tab provides two types of reports: Graphical and Tabular.

Many users will initially confuse these options with the belief that “any and all” items selected (from Table or Gallery view) can be included in a Report. While this is technically correct, NOT every item is capable of producing a “picture or thumbnail” (for Graphical reports). A case in point would be “History URLs”. In this case, “history URLs” are not associated to any actual file (eg: from a cache). Therefore, there is no means to represent this record as a thumbnail other than by using the CacheBack default .PNG thumbnail image.

As for the Tabular option, this is reserved mostly for reporting on the chronology of history and cache URLs. Therefore, the metadata provided in the report is limited to specific URL record details. In future releases of CacheBack, it may be possible to select which columns are to be displayed. But for now, the Tabular report is limited to visit times, visit count, User and URL.

The following page contains a screenshot of the Report tab displaying a Graphical report.

Table | Query | Gallery | Files

	Icon	URL ID	Links	Type	Rebuilt	File Exists	Status	Action	Action Date [UTC]	Action Date [UTC -0400]	Visits	Data So
329		1215			No	Yes		Visited	2009-06-02 05:24:14	2009-06-02 01:24:14 DST	1	Firefox Ca
330		1012			Yes	Yes		Visited	2009-06-02 05:24:16	2009-06-02 01:24:16 DST	1	Firefox Ca
331		1171			No	Yes		Visited	2009-06-02 05:24:17	2009-06-02 01:24:17 DST	1	Firefox Ca
332		1181			No	Yes		Visited	2009-06-02 05:24:17	2009-06-02 01:24:17 DST	1	Firefox Ca

Browser | Text | Hex | Picture | Links | Audit | Report

CacheBack Internet History Report

Report Date: Wednesday, July 01, 2009 12:43:05 PM
Report Prepared By: John Smith
Organization: ACME Investigations
Case Number: 2009-000123

Results are displayed in Coordinated Universal Time (UTC). The active time bias is set to: -0400 DST.
Time Zone Setting: -0400 (GMT-05:00) Bogota, Eastern Time (US & Canada), Indiana (East)

ITEM #0000001 -- WEB PAGE INFORMATION SHEET

URL ID #0001012

Title:	AccessData Lab family of solutions enables labs of all sizes, facing an array of challenges, to work more effectively.
File Name:	lab.html
Parent Folder:	
User:	
Visits:	1
Time Rebuilt:	2009-01-07 12:41:50
Action:	Visited
Action Date [Local]:	2009-06-02 01:24:16 DST
Action Date [UTC]:	2009-06-02 05:24:16 UTC
Data Source:	Firefox Cache Version 2.0
Original URL:	http://www.accessdata.com/lab.html
LOCAL EVIDENCE FILE:	0001012.CBR/0001012.htm

The following is a screenshot of the Report tab displaying a Tabular report.

NOTE: The User column is empty in this particular case because there is a no User value that has been parsed from any existing “Internet Explorer” history or cache records. Also, the Safari, Google Chrome and Opera browsers do not record the Visit Count for Cache artifacts. As a result, CacheBack uses a *negative one (-1)* to infer a “default” of “one visit”.

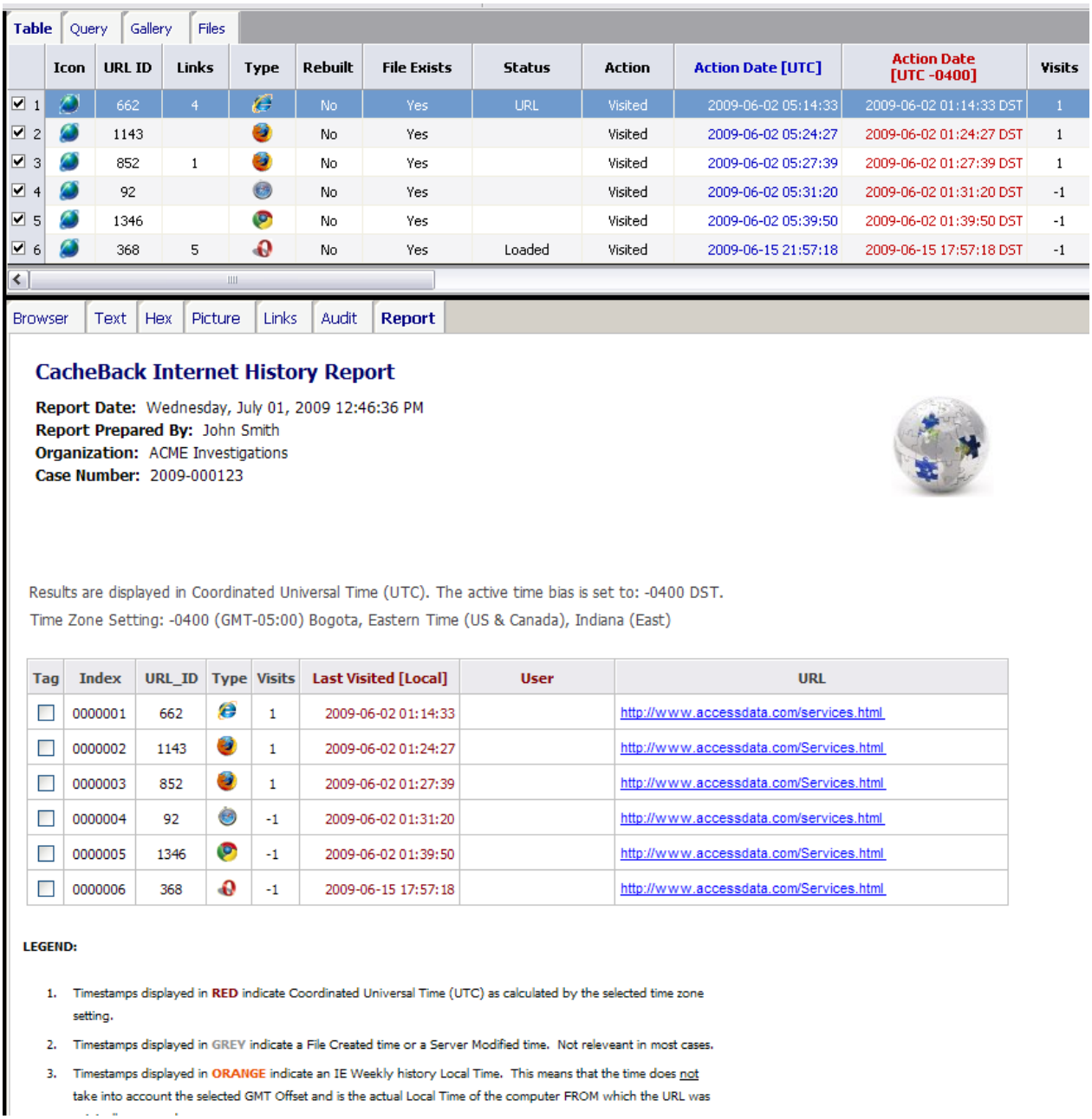


Table | Query | Gallery | Files

	Icon	URL ID	Links	Type	Rebuilt	File Exists	Status	Action	Action Date [UTC]	Action Date [UTC -0400]	Visits
1		662	4		No	Yes	URL	Visited	2009-06-02 05:14:33	2009-06-02 01:14:33 DST	1
2		1143			No	Yes		Visited	2009-06-02 05:24:27	2009-06-02 01:24:27 DST	1
3		852	1		No	Yes		Visited	2009-06-02 05:27:39	2009-06-02 01:27:39 DST	1
4		92			No	Yes		Visited	2009-06-02 05:31:20	2009-06-02 01:31:20 DST	-1
5		1346			No	Yes		Visited	2009-06-02 05:39:50	2009-06-02 01:39:50 DST	-1
6		368	5		No	Yes	Loaded	Visited	2009-06-15 21:57:18	2009-06-15 17:57:18 DST	-1

Browser | Text | Hex | Picture | Links | Audit | Report

CacheBack Internet History Report

Report Date: Wednesday, July 01, 2009 12:46:36 PM
Report Prepared By: John Smith
Organization: ACME Investigations
Case Number: 2009-000123

Results are displayed in Coordinated Universal Time (UTC). The active time bias is set to: -0400 DST.
Time Zone Setting: -0400 (GMT-05:00) Bogota, Eastern Time (US & Canada), Indiana (East)

Tag	Index	URL_ID	Type	Visits	Last Visited [Local]	User	URL
	0000001	662		1	2009-06-02 01:14:33		http://www.accessdata.com/services.html
	0000002	1143		1	2009-06-02 01:24:27		http://www.accessdata.com/Services.html
	0000003	852		1	2009-06-02 01:27:39		http://www.accessdata.com/Services.html
	0000004	92		-1	2009-06-02 01:31:20		http://www.accessdata.com/services.html
	0000005	1346		-1	2009-06-02 01:39:50		http://www.accessdata.com/Services.html
	0000006	368		-1	2009-06-15 17:57:18		http://www.accessdata.com/Services.html

LEGEND:

1. Timestamps displayed in **RED** indicate Coordinated Universal Time (UTC) as calculated by the selected time zone setting.
2. Timestamps displayed in GREY indicate a File Created time or a Server Modified time. Not releveant in most cases.
3. Timestamps displayed in **ORANGE** indicate an IE Weekly history Local Time. This means that the time does not take into account the selected GMT Offset and is the actual Local Time of the computer FROM which the URL was

Toolbar

With the release of CacheBack 2.7, the toolbar has not only undergone a cosmetic facelift, but certain new buttons and features have been added. The following is a brief overview of each toolbar button.

New Project

Creates a new CacheBack Project file (.CBP) using the name and target folder specified by the user.

Open Project

Opens an existing CacheBack Project file (.CBP) from the source folder specified by the user and closes any currently opened projects (at the consent of the user).

Save Project

The current project file is "Saved As..." a new filename as provided by the user. User will be prompted to overwrite any existing files with the same name.

Stop Process

Sometimes a web page that is being loaded into the browser control located on the Browser tab can take an unusual amount of time to finish. This option allows the user to cancel or stop the loading process.

Refresh

This option refreshes / reloads the contents of the Table or the Gallery (depending on which tab is currently selected) using the *current underlying query* (as defined on the Query tab, and in conjunction with any Host filtering option selected).

Import

Invokes the Import Wizard window and prompts user to select which cache and / or history artifacts to import into the current project file. NOTE: Files already imported (eg: with the same path) cannot be imported a second time into the same project file.

Rebuild

Rebuilds all validated web page files that are selected (checkmarked) in the Table or Gallery. All rebuilt files are then compressed, encrypted, and stored back in the project file... along with all the web page ingredient files (eg: pictures, cascading stylesheets). A new / updated thumbnail is also created and stored in the project file.

Report

Invokes the Report window and allows the user to select a Graphical or Tabular report, based on the Table or Gallery items (URL records) selected. Results are produced inside the Report viewer tab as HTML.

Publish

Prompts the user to select an "output folder" where the current Report (must be displayed in the Report tab!) will be published. This includes all ingredient files such as pictures, cascading stylesheets, etc.

Filters

Provides a series of Filters which are used to "show or hide" URL records in the Table. Filters act as an "overlay" to the *_DEFAULT query* and is not considered a best approach to reducing the visible recordset. Filters should only be used to fine-tune the displayed results. Since filters can sometimes result in unwanted or unpredictable displays or report data, users should use *queries* to reduce the current dataset, that is being displayed or reported.

Custom Search

Invokes the SQL Query Builder window which provides users with a quick and easy way to define customer search queries. For more robust query options, users should use the Query Builder available on the Query Tab.

Manage Favorites

This feature was disabled in Version 2.7 in favor of the features made available in the new Query Tab. This *Manage Favorites* option will be made available in a future dot release of CacheBack and will provide an easy-to-use interface to managing saved queries.

Link Analysis

This option scans through a project file and identifies matches (relationships) between History URLs and hyperlinks found embedded in Cache web pages. This option REQUIRES that the *underlying query* SHOW ALL URLs. Hence, using the Load Default query option on the Query Tab will prepare CacheBack for Link Analysis.

Time Zones

Invokes the Global options tab on the Options window where users can set the date and time formats for displayed timestamps. Users can also select which Time Zone which will be used for displaying and reporting timestamps from History or Cache artifacts. The Options Window is discussed later in this manual.

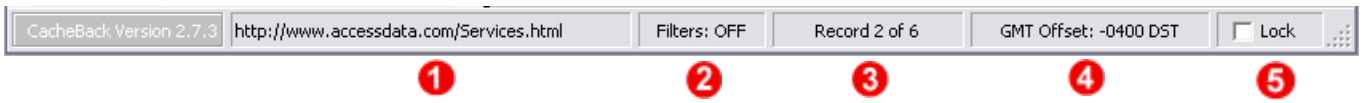
Quick Queries

Provides a context menu (popup menu) that contains a list of the most common queries to use in CacheBack which can help greatly reduce the number of visible records in the Table or Gallery view.

Help

Provides direct access to the local help file, online update options and technical support services available through the CacheBack website. It also provides a direct link to the CacheBack user message forums.

Status Bar



The Status Bar contains five (5) panels that provide feedback about the current state of the program. The following is a description of each of the panels.

1. **URL Panel**. Displays the full, original URL path for the currently selected record.

2. **Filters Panel**. Indicates if Filters are currently enabled or disabled.

3. **Record Panel.** Indicates the "ordinal position" of the currently selected record as enumerated in far left column in the Table.

4. **Time Zone Panel.** Indicates the current Time Zone setting as specified in the Global tab options of the Options window. The Options window is accessed via the View menu at the top of the main window.

5. **Lock Panel**. Use this feature to "lock" the currently selected Viewer tab in place so that any URL selected in Table or Gallery, will display its contents (if possible) in the current viewer tab.

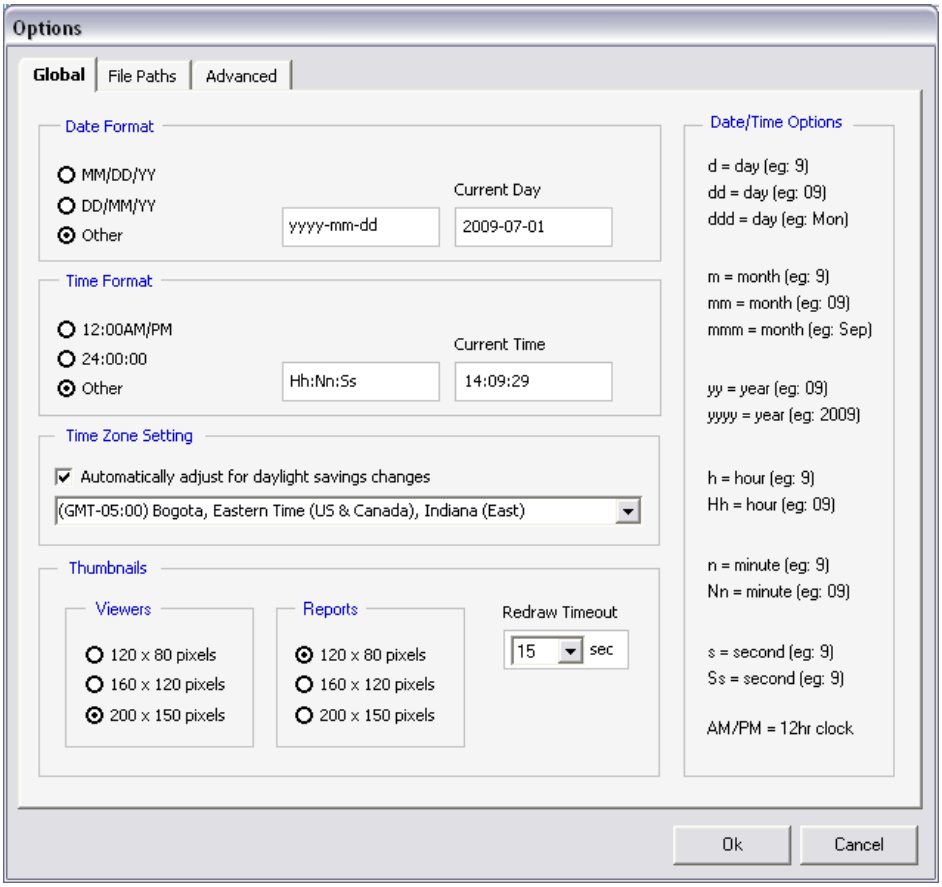
Windows

The Options Window

The Options window is where users can customize the display format and time zone settings for dates and times within CacheBack. It also allows for file paths to be defined and includes some advanced options to control how the Gallery displays its contents. This window is accessed by selecting "Options…" under the "View" menu option (above the Toolbar).

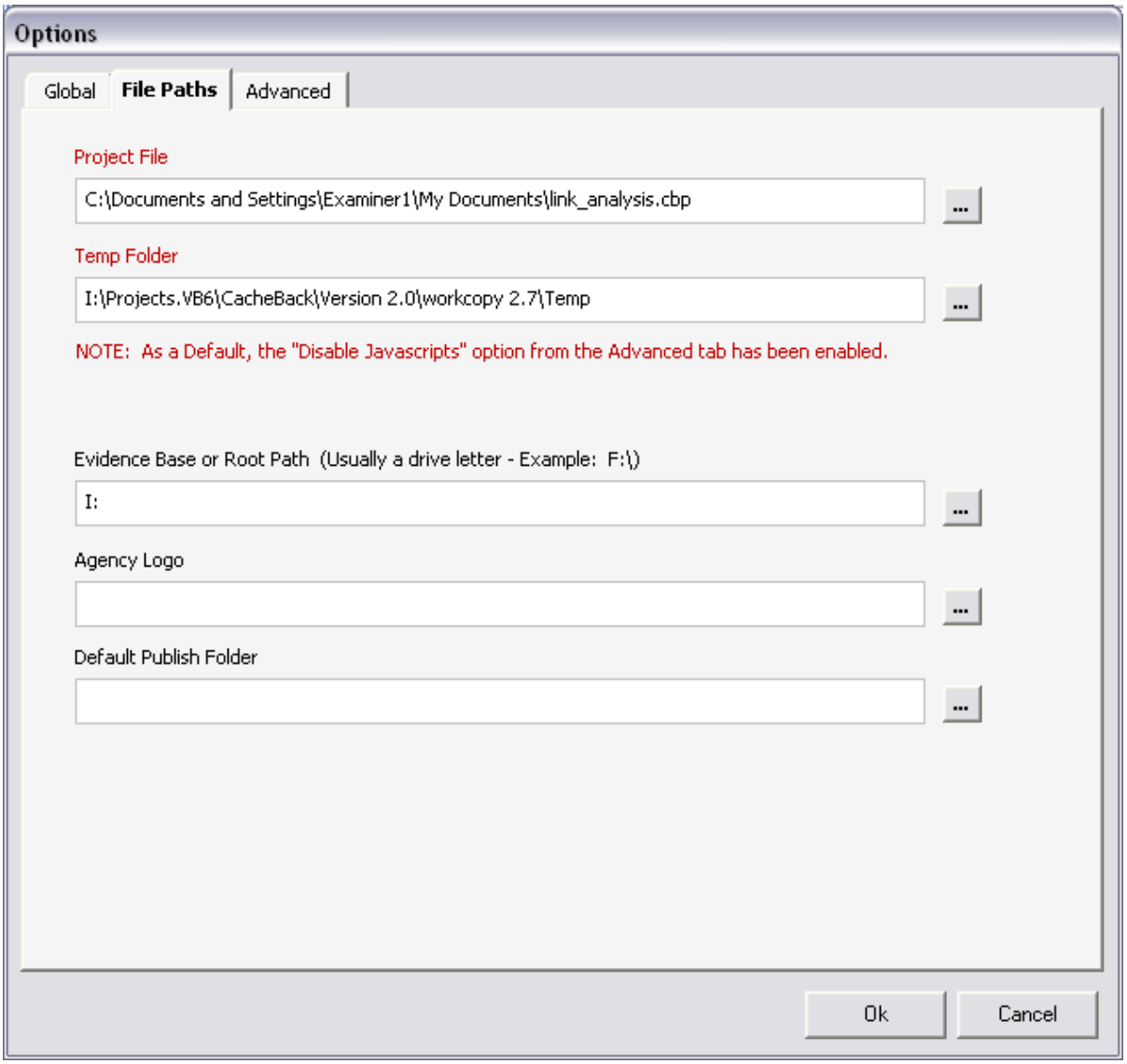
Global Tab

This is the first tab on the Options window and provides users with choices on how to format and display timestamps. It also provides choices on the size of thumbnails used in the Gallery and in Reports.



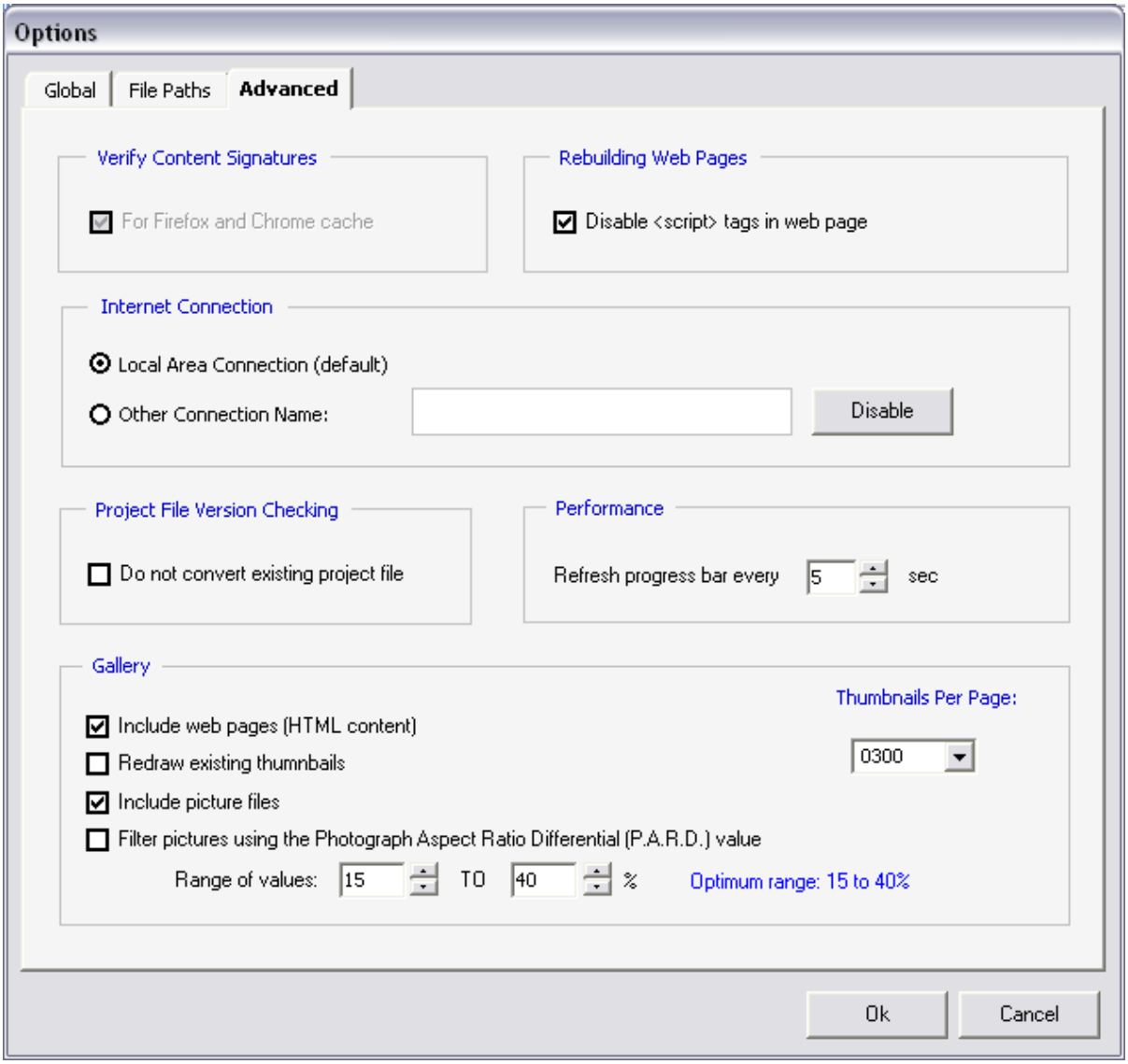
File Paths Tab

The full qualified paths for the project file, temp folder, and other options are defined here. The details or significance of these paths are discussed later on in this manual.



Advanced Tab

This tab provides options to control scripting in web pages, performance, and the display of pictures and web pages in the Gallery. It also enables the user of picture filtering using the Photograph Aspect Ratio Differential value (which is discussed in detail later on in the manual).



NOTE: The option to toggle the default network adapter on or off is provided here for simplicity via the "Internet Connection" setting. It provides users with a quick and easy way to disconnect from the Internet while conducting examinations locally using CacheBack. While CacheBack's offline browser displays cached files locally,

this option is an added measure of security to prevent web page scripts from making calls to any outside resource (eg: remote javascript files).

IMPORTANT: The "Local Area Connection" is the default name given to the first physical network adapter in a Windows system. In the event that you or your network administrator has renamed the connection (eg: via the Windows Control Panel under Network Connections), then it can be entered using the "Other Connection Name" option. This is case sensitive and MUST correspond to an adapter (connection) listed in the Network Connections window.

Forensic workstations should never be connected to the Internet or external network during an examination! This ensures that your evidence is never contaminated from outside sources.

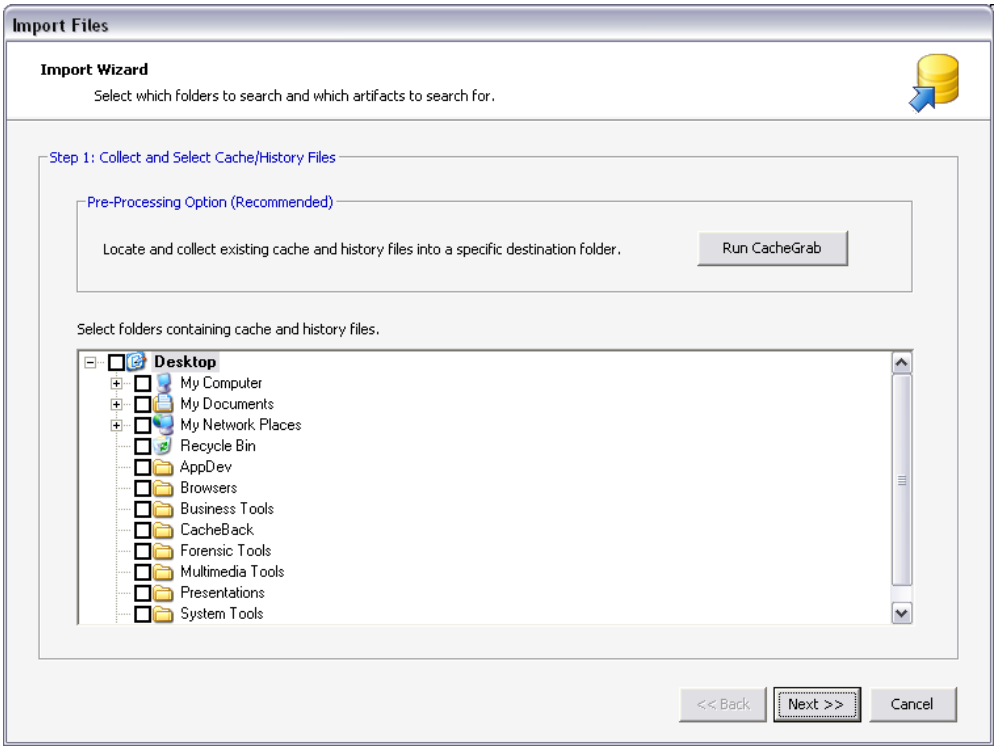
Import Wizard

The Import Wizard window provides users with a step-by-step process of selecting and importing Internet cache and history files into CacheBack. This wizard also provides some advanced parsing features, as well as, facilitates the collection of source data using the new CacheGrab tool. The following are brief descriptions of each step found in the wizard. More detail about this process is provided later on in this manual.

Step 1: Data Source and CacheGrab®

There are two ways to tell CacheBack where the Cache and History files you want examined are located. One is by manual selection (eg: using Windows Explorer to locate and collect the files), the other is by using CacheBack's automated cache and history recovery tool, called: CacheGrab. In either method, the results should be copied to, and made available, in one or more specific folders.

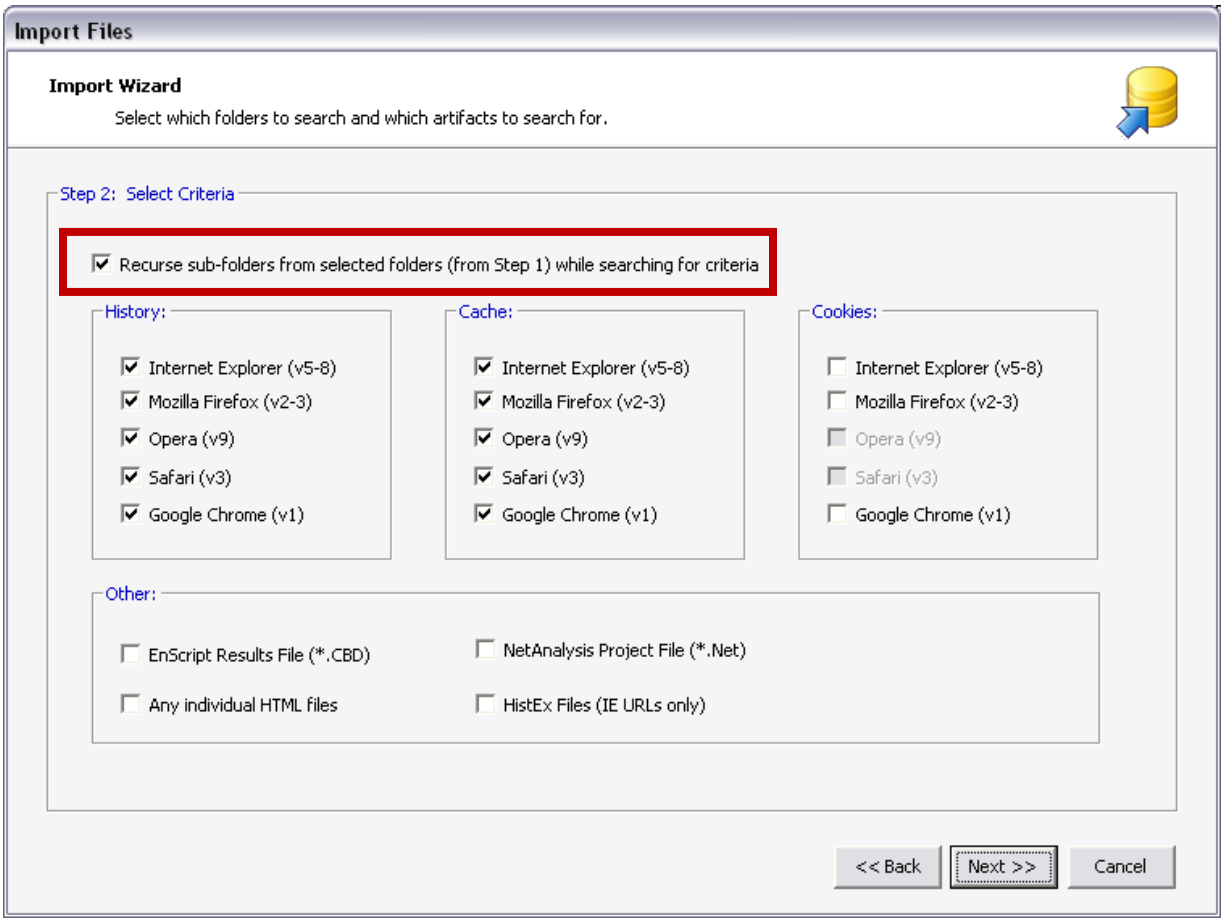
When we are ready to import files, we first need to checkmark each folder that is to be searched. If the intention is to have CacheBack recurse through children folders of any selected folder, then it is not necessary to manually "checkmark" each child folder. *Recursion* is a setting (on Step 2) that tells CacheBack to search any and all "children folders" that may reside in a particular checkmarked folder.



Step 2: Selecting Artifacts for Import

CacheBack supports History and Cache for all five (5) top browsers. With the release of 2.7, support has been added for Cookies for MSIE, Firefox and Google Chrome (support for Opera and Safari coming soon). Support is also provided for .CBD (CacheBack Data) files which are produced by the new CacheGrab EnScripts available for EnCase Version 5 and Version 6 users. HistEx IE URL results (a tool by NetAnalysis) is also supported, along with "plain HTML files" which may have been carved out from Unallocated Space.

NOTE: The "Recurse sub-folders" is checked off by default.

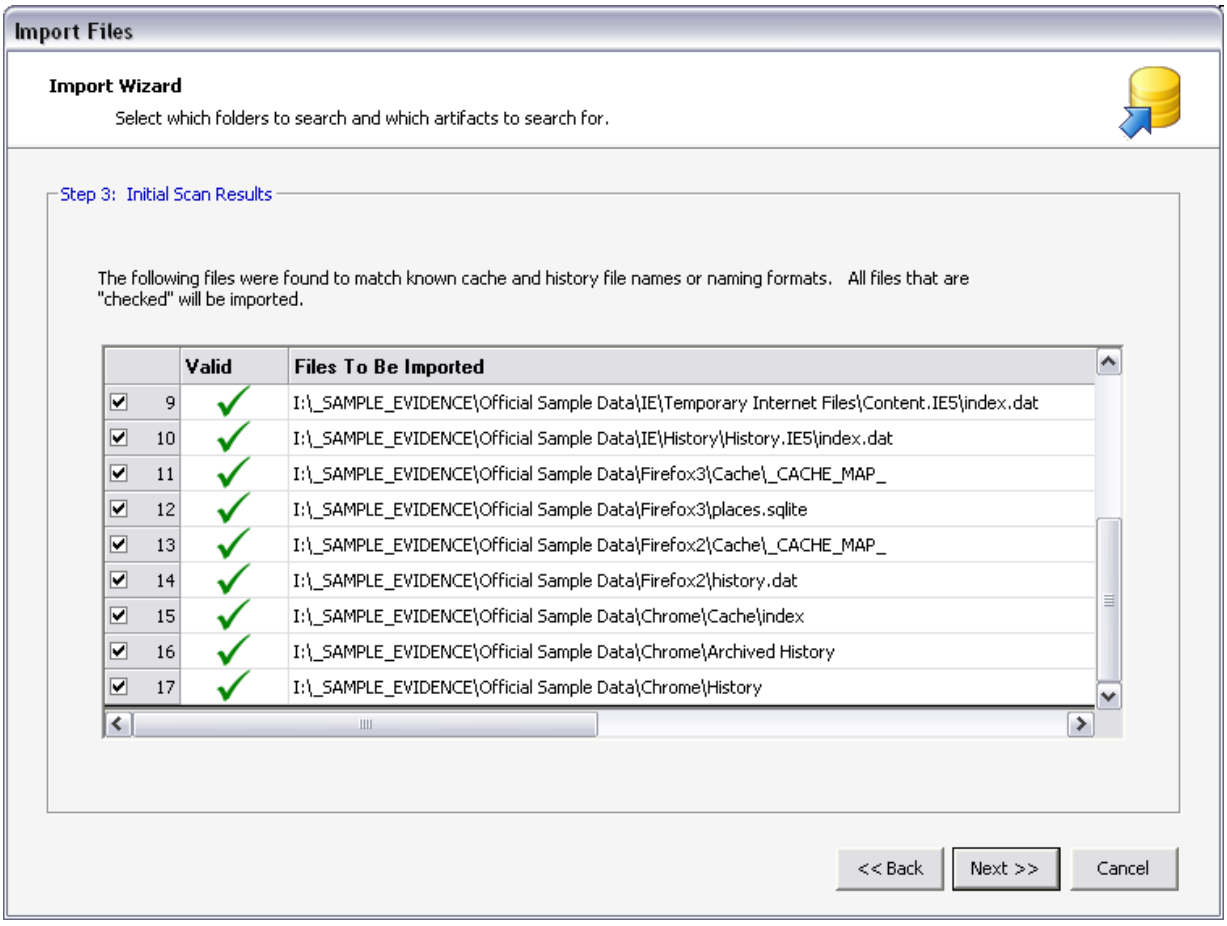


Step 3: Validation Process

Based on user selections from Step 1 and Step 2, CacheBack will then pre-validate any files that match the selected criteria. Validation is performed in two stages.

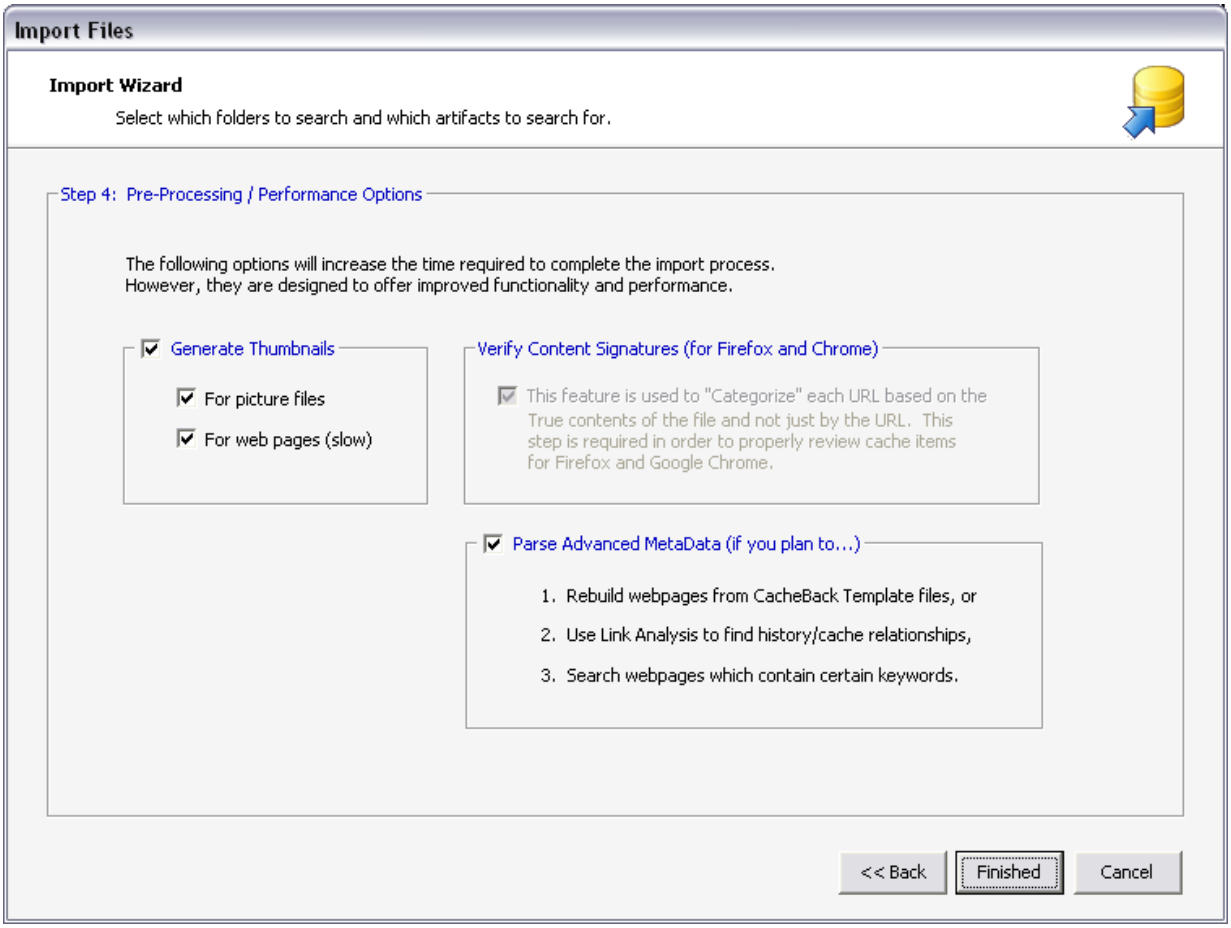
The first stage searches the selected folders looking for matches based on expected “file naming” conventions (eg: index.dat). The second stage requires (a) an actual “peek” into the file looking for a specific header match, or (b) the existence of certain keywords, or (c) that a specific parent or child folder exist, in relation to the file.

The second stage may implement two or more of the above criteria, as is the case with “index.dat” files used by Internet Explorer to manage Cache, History and Cookie data. NOTE: The second phase of the validation process will place a green checkmark in the Valid column when a positive match is made.



Step 4: Pre-Processing and Performance Options

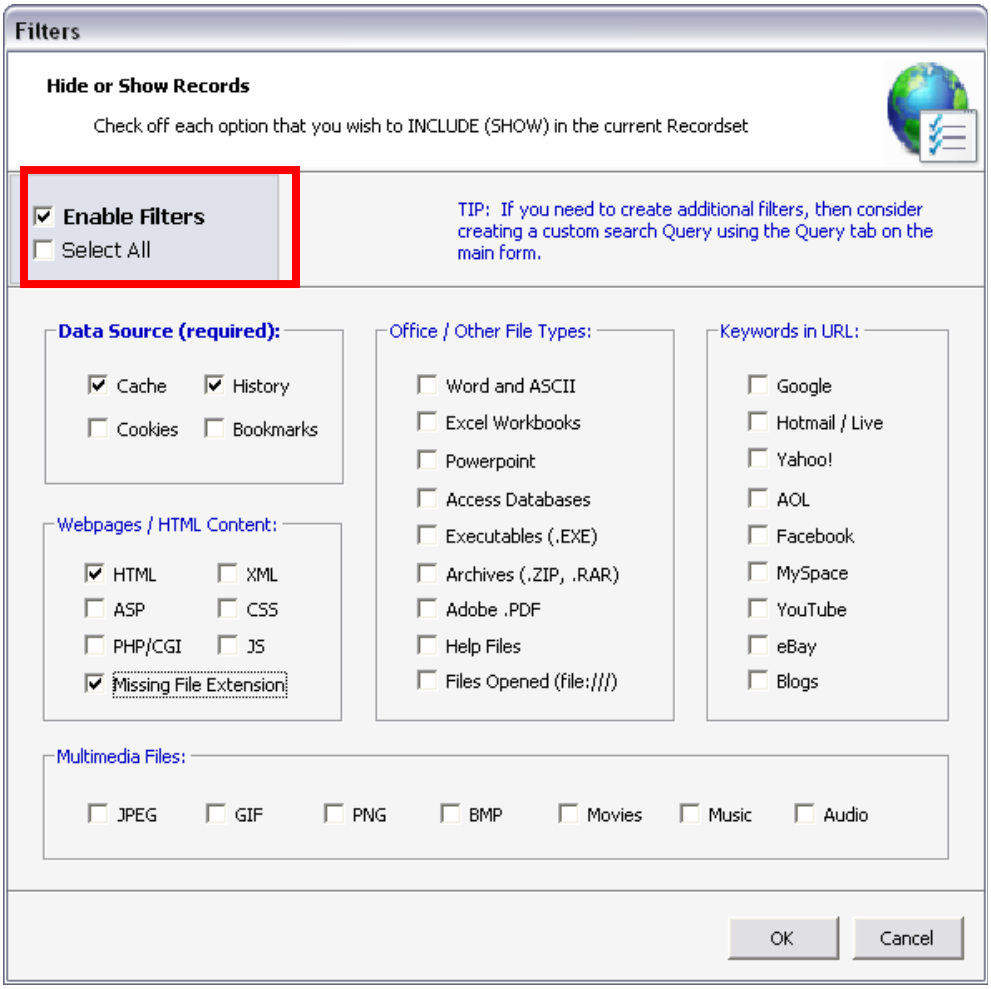
These final options control the efficiency in which thumbnails are displayed in the Gallery at run time. It also dictates if Link Analysis will be available once the import process is complete AND if custom keyword searches of "web page contents" can be made.



Filters Window

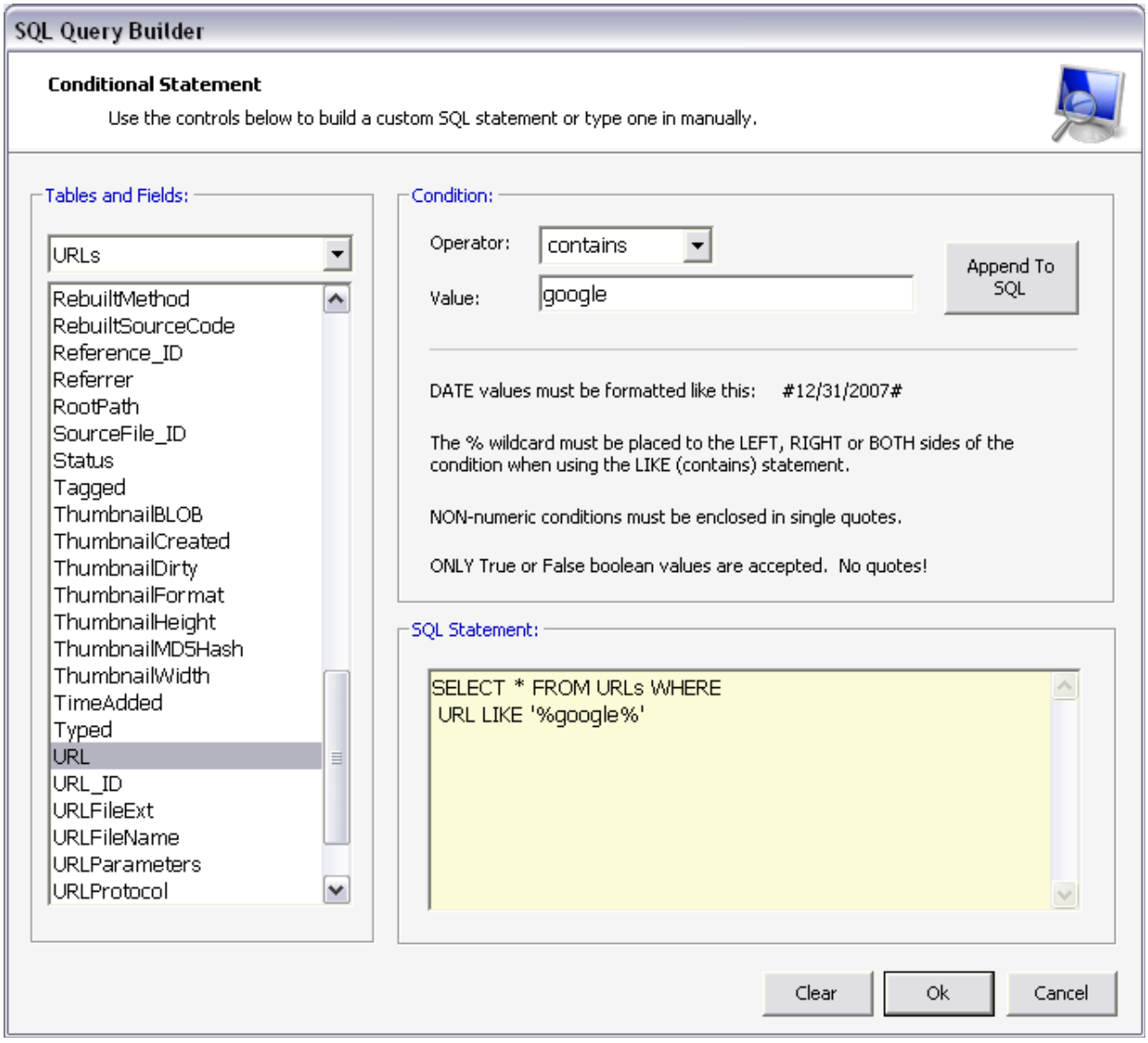
The Filters window, accessible via the Toolbar, provides users with the ability to "show" only those records (in the Table) that match the selected filter options. What is important for users to understand is that the use of Filters are more or less a way of *fine tuning* the *displayed results (URL records)* in the Table. What this means is that Filters simply control which records are "visible" and which ones are "hidden". More importantly, Filters act as an "overlay" to the *underlying query (or query statement)* which is responsible for returning the data (URL records) displayed in the Table (or Gallery).

If a particular filter does not exist, then users will have to consider a custom query using the Query Builder located on the new Query Tab. One last note... When using Filters, it is important to remember that "Enable Filters" must be "checked" in order for the Filters to be applied. Likewise, removing the associated checkmark disables the use of Filters.



Custom Query Window

The Custom Query Window is accessed via the top Toolbar and is intended to be a quick means of searching for specific data in the project file. Users simply choose a Column Name and then define the condition of the value that is being sought. By appending different values to the SQL Statement box, users can create some rather complex search queries. NOTE: CacheBack Version 2.7 introduced the Query Tab and Query Builder options which have replaced the option of saving Favorites through the Custom Query Window.

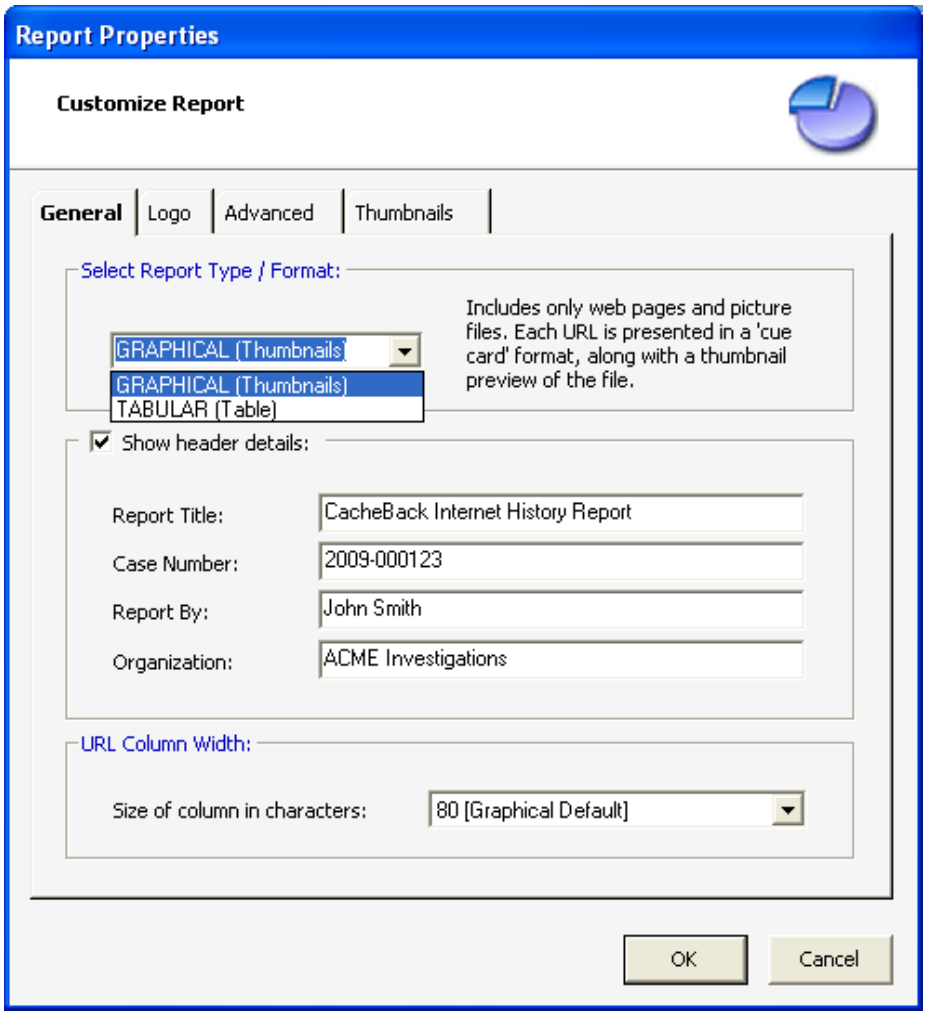


Report Window

The Report window is accessed via the Toolbar and is used to generate both Graphical and Tabular styled reports. All reports are rendered inside the browser control that is found on the Report viewer tab. Users have the option of configuring header details, inclusion options, and custom logo for their reports. All reports can also be Published to an output folder using the Publish option on the Toolbar.

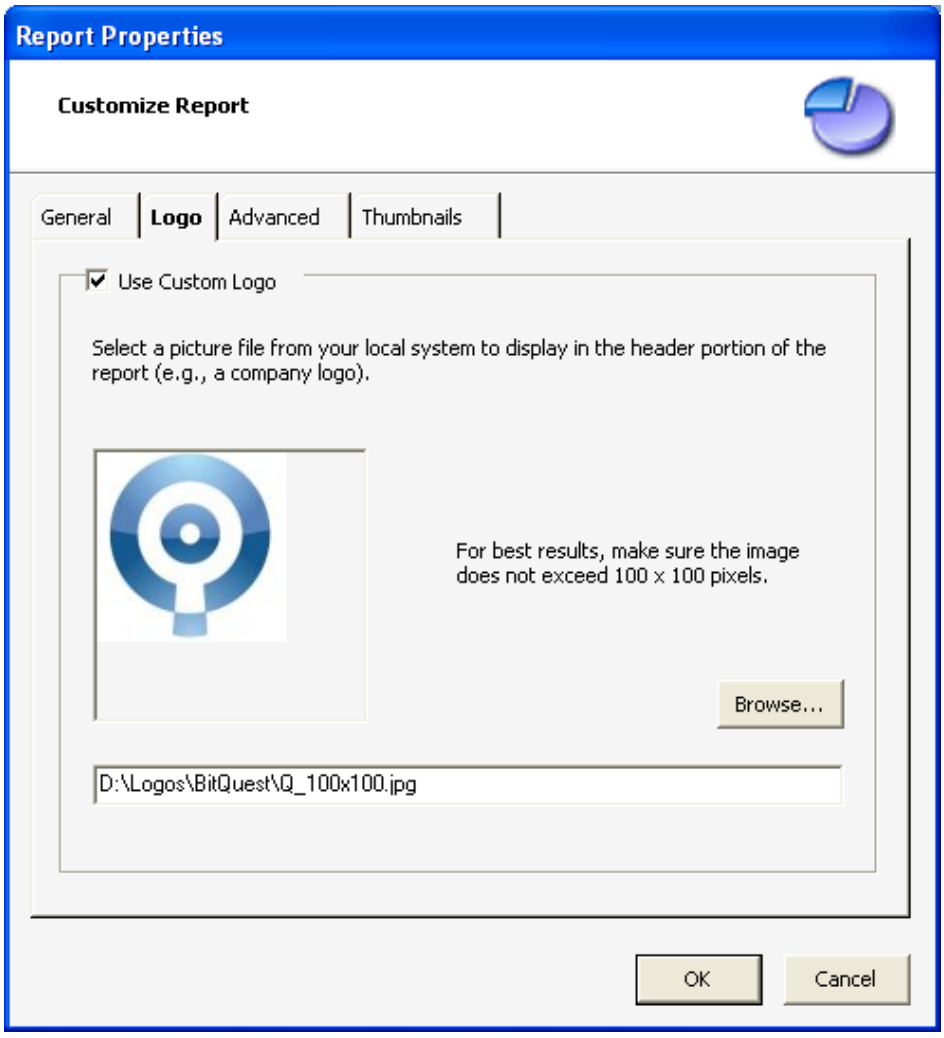
General Tab

The General tab provides the basic elements for each report, starting with the Report Type and followed by the Report Header. For reports with URL records containing unusually long URL paths, CacheBack provides the **URL Column Width** option to make sure HTML reports are easily viewed and printed in standard Portrait format.



Logo Tab

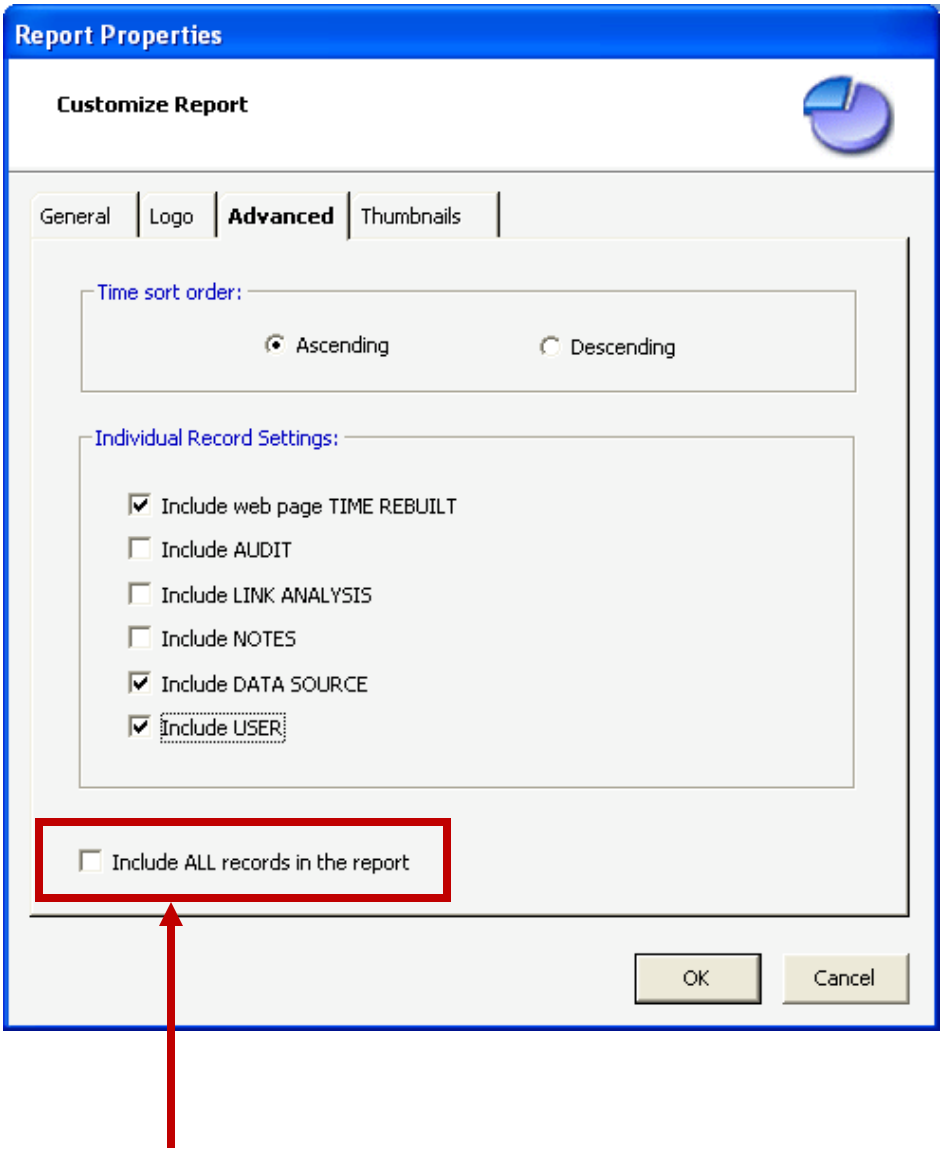
Provides users with the option of defining a file path to any valid picture file that is intended to be used as a logo in the top right corner of the report.



Advanced Tab

This tab provides several “include” options for the report, such as “Audit”, “Link Analysis” and “Notes”.

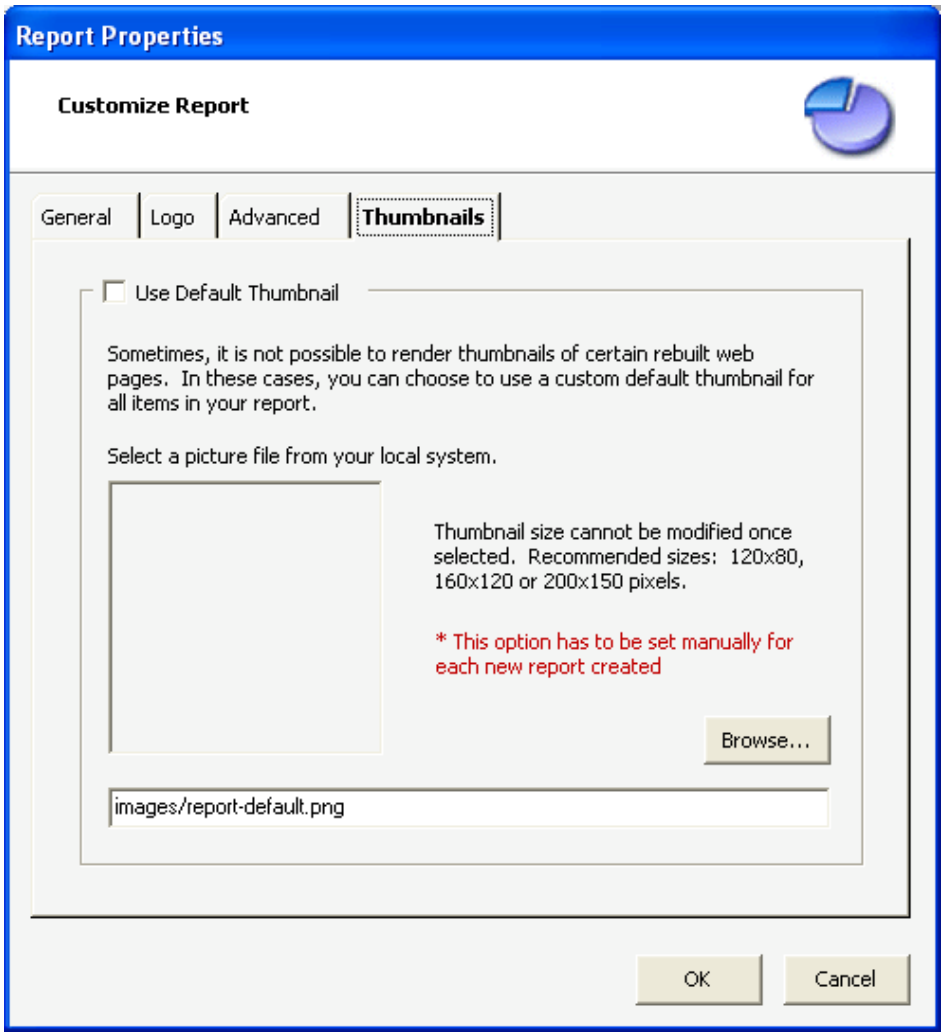
While the AUDIT, LINK ANALYSIS and NOTES options are only available for Graphical Reports, the DATA SOURCE and USER options are available on both types of reports (eg: Graphical and Tabular).



Note: Use the Include ALL records in the report option to include every record that is presently displayed in the Table, on the main window. This can override any already tagged rows in the Table and provide an easy way to print everything, without undoing your selections.

Thumbnail Tab

Provides users with the option of defining a file path to any valid picture file that is intended to be used as a "default thumbnail" for any report entries requiring a thumbnail graphic. This option was intended originally for projects involving pictures of offensive material where the user could substitute a placeholder image instead.



Keyboard Shortcuts

The following keyboard shortcuts are available to expedite certain common tasks within CacheBack.

CTRL + N Creates a new project (case) file.

CTRL + O Opens an existing project (case) file.

CTRL + S Prompts the user to Save the existing project (case) As another filename and make the new filename the active project (case) file.

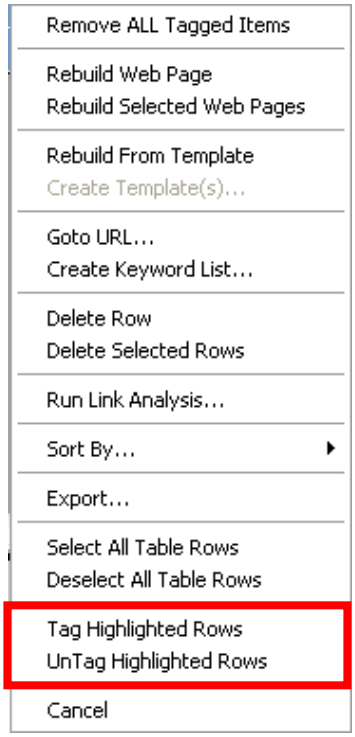
CTRL + H Invokes the Import Wizard window which prompts users to select History and cache files for import.

SELECTING RECORDS IN TABLE

Users can "highlight" rows in the Table using a combination of the **SHIFT** key and the **UP** and **DOWN** arrow keys on the keyboard. Once a range of rows have been "highlighted", users can then right-mouse-click on the Table to display the context (popup) menu. From the list of options, users can then:

a) Tag Highlighted Rows

b) UnTag Highlighted Rows



NOTE: Most, if not all, features in CacheBack that work with URL records, rely on which records in the Table (or Gallery) have been selected (eg: a single row highlighted), or tagged (checkmarked).

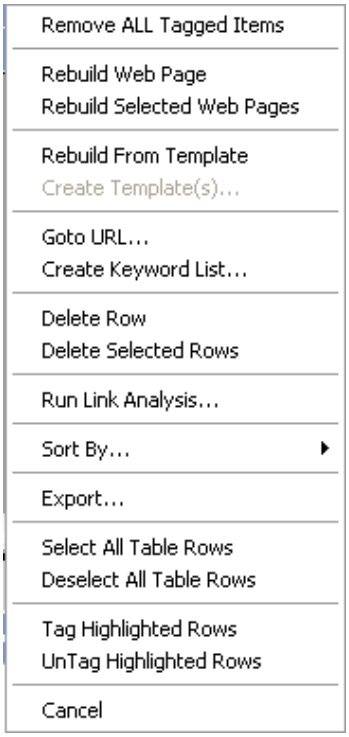
IMPORTANT: The most common problem new users will face is forgetting to select one or more records using the checkbox feature. This is extremely important to remember as failing to checkmark specific records will result in reports that contain too few, or no records.

Context (Popup) Menus

Like most Windows applications, CacheBack uses context menus for convenient access to repeated or advanced tasks. There are essentially four (4) main context menus available in the program. All context menus (except for the Quick Queries button on the toolbar) are accessed by *right-mouse-clicking* inside of an existing control or part of the user interface.

Table Context Menu

This menu is available when the Table Tab is selected in the Data Pane, and anytime there is at least one (1) record displayed in the Table. It is displayed by right-mouse-clicking anywhere inside the Table. Depending on certain conditions or features enabled or disabled at the time, one or more items on the context menu itself may be temporarily disabled.



Gallery Context Menu

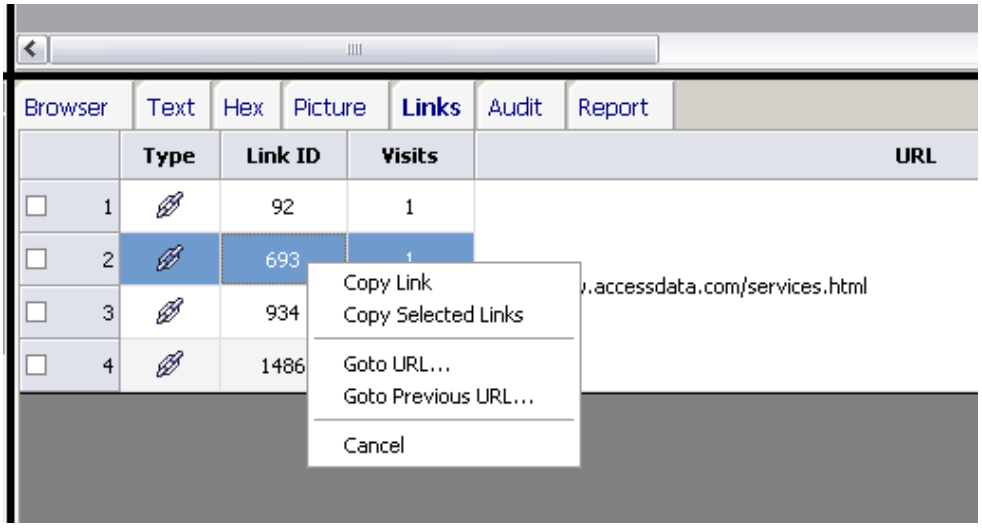
This menu is available from the Gallery, which is found on the Gallery Tab, in the Data Pane.

This context menu is the SAME menu as the one displayed in the Table (see above). The only different here is that the Gallery does not offer some of the features on the menu that are available when viewing records via the Table tab.

Links Context Menu

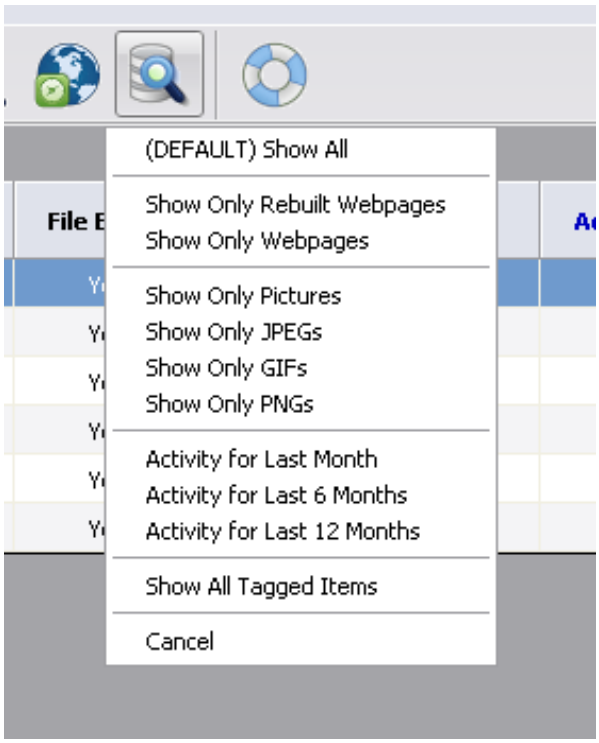
This menu is available from the Links Tab which is found in the Viewer Pane. It is available anytime there is at least one (1) record displayed in the table (grid) found on the Links tab.

This context menu allows users to move back and forth between a “cache” URL record and its associated “history” URL record(s) *within the Table only*, or vice versa.



Quick Queries Context Menu

This menu is accessed by the Quick Queries button on the Toolbar and does NOT require the use of the right mouse button. Simply clicking on the button (using the left / default mouse button) will invoke the menu. This menu provides a fast way to run *pre-validated and compiled* search queries, which are stored inside each new project file.



SETTING OPTIONS USING "CACHEBACK.INI"

CacheBack uses a file called "cacheback.ini" to maintain user preferences in relation to the use of the software. The .INI file is located in the program directory where CacheBack has been installed and can be easily viewed and/or modified using Notepad. The following is a listing of the default cacheback.ini file that comes with each new installation of the software. Any values that are missing are replaced at run-time with a pre-defined default value. Boolean values default to "False" and string values default, in most cases, to "" (an empty string).

One example of using the .INI file would be for network administrators to remotely define or re-define the "TempFolder" path or "AgencyLogoPath" as company resources or policies change.

WARNING: Removing, renaming or incorrectly defining values within the CacheBack.INI file can adversely affect the use of the CacheBack program and in some situations render is unusable. It is mentioned here for completeness and should only be edited by advanced users, or network administrators wishing to setup "like" installations on multiple workstations.

[DEFAULT]
ParseBodyTag=True
TempFolderPath=
LastProjectFilePath=
ConnectionOffline=
PublishTempFolder=
ValidPictureObjects=
ValidIncludeObjects=
ParseAdvancedMetaData=True

[OPTIONS]
NetworkConnectionName=Local Area Connection
DisableScriptTags=True
DisableAutoRebuildInHTMLGallery=True
ReportIncludeNotes=False
ReportIncludeAudit=False
ReportIncludeLinkAnalysis=False
DateFormat=yyyy-mm-dd
TimeFormat=Hh:Nn:Ss
DebugMain=False
ParseAdvancedMetaData=False
LocalFavoritesPath=
SharedFavoritesPath=

TemplatesPath=F:\Templates
TimeZoneBias=300
TimeZoneDescription=
GalleryIncludeHTML=True
GalleryIncludePictures=True
RedrawThumbIfPageRebuilt=False
StatusBarRefreshRate=5
GalleryUsePIART=False
ProgressBarRefreshRate=5
TZDescription=
TimeZoneName=
VerifyFileSignatures=False
ShowTipsAtStartup=True

[THUMBNAILS]
HTMLThumbWidth=120
HTMLThumbRedrawTimeout=15
ViewerThumbnailWidth=200
ReportThumbnailWidth=120

[FILES] **// these are the last 6 files opened under the File menu**
File1=
File2=
File3=
File4=
File5=
File6=

[REPORT]
CaseNumber=
ReportTitle=CacheBack Internet History Report
Organization=
ReportBy=
ReportedBy=
CompanyLogoPath=
ReportType=0
IncludeAudit=False
IncludeLinkAnalysis=False
IncludeNotes=False
IncludeTimeRebuilt=True
UseDefaultThumbnail=True
DefaultThumbnailPath=images/report-default.png
UseDefaultThumbnailPath=False

[FILE PATHS]
TempFolder=
LastProjectFilePath=
AgencyLogoPath=
LastProjectDBPath=

BROWSER ARTIFACTS

Introduction

Welcome to the discussion about Internet artifacts! This topic used to be much simpler in earlier days where the only real browsers in use by the general public were Microsoft Internet Explorer and Netscape Navigator. Today, we have five (5) browsers to investigate that are most commonly used and on top of this, there are some big changes between versions, in some cases. These browsers include Microsoft Internet Explorer, Mozilla Firefox, Apple Safari, Opera and Google Chrome.

What we will learn in the pages that follow is that Windows XP and Windows Vista store the different cache and history files in entirely different file path locations on a hard drive. Understanding *what* needs to be collected is the most important thing you need to know. This guide will show you not only where to look for these files, but we will introduce you to CacheGrab®, a standalone utility to make the collection process much easier.

While it would be insightful for us to discuss the *detailed* physical intricacies of how cache artifacts are managed by the different browsers, it precipitates a much larger discussion that is outside the scope of this manual. Instead, we will provide you with a higher-level understanding of the mechanisms involved for each browser. We will also show you how you can use CacheBack metadata to manually verify the source data. The advanced discussions about the actual cache constructs will be included in an upcoming course or alternate training delivery system. If you are interested in such a course or training, please contact us at info@cacheback.ca and let us know.

What we also do not cover in this manual is the various nuances or user preferences / options made available *by the actual browser programs themselves*. At this time, we have decided to confine the scope of discussion to the elements required by CacheBack. We may visit advanced topics relating to Registry settings and user preferences at a later time, but for now, we will keep our instructions “on topic”.

Chart 1 – Cache and History artifacts locations by browser and operating system.

The following chart lists all artifacts by browser and operating system. In order to save space, we have omitted the parent folder path from the actual file paths due to repetitiveness. As a reminder, user profiles on an XP system are found in the: `C:\Documents and Settings\<user profile>` folder. On a Windows Vista system, they are found in: `C:\Users\<user profile>`.

BROWSER / ARTIFACT	VER.	O/S	FOLDER PATH (*Relative* to the User Profile)	FILE(s) (if applicable)
Microsoft Internet Explorer				
Cache	5-8	XP	\Local Settings\Temporary Internet Files	*(all files & subfolders)*
		Vista	\AppData\Local\Microsoft\Windows\History	*(all files & subfolders)*
History		XP	\Local Settings\History	*(all files & subfolders)*
		Vista	\AppData\Local\Microsoft\Windows\Temporary Internet Files	*(all files & subfolders)*
Cookies		XP	\Local Settings\Cookies	*(all files & subfolders)*
		Vista	\AppData\Local\Microsoft\Windows\Temporary Internet Files	*(all files & subfolders)*
Mozilla Firefox				
Cache	2	XP	\Local Settings\Application\Data\Mozilla\Firefox\Profiles\<name>.default\Cache	*(all files & subfolders)*
	3		\Application Data\Mozilla\Firefox\Profiles\<name>.default\Cache	
	2 & 3	Vista	\AppData\Local\Mozilla\Firefox\Profiles\<name>.default\Cache	*(all files & subfolders)*
History	2	XP	\Application Data\Mozilla\Firefox\Profiles\<name>.default\	history.dat
		Vista	\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default\	
	3	XP	\Application Data\Mozilla\Firefox\Profiles\<name>.default\	places.sqlite
		Vista	\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default\	
Cookies	2	XP	\Application\Data\Mozilla\Firefox\Profiles\<name>.default\	cookies.txt
		Vista	\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default\	cookies.sqlite
	3	XP	\Mozilla\Firefox\Profiles\<name>.default\	cookies.txt
		Vista	\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default\	cookies.sqlite
Apple Safari				
History	2	XP	\Application Data\Apple Computer\Safari\	History.plist
	3	Vista	\AppData\Roaming\Apple Computer\Safari\	

BROWSER / ARTIFACT	VER.	O/S	FOLDER PATH (*Relative* to the User Profile)	FILE(s) (if applicable)
Apple Safari				
Cache	2	XP	\Local Settings\Application Data\Apple Computer\Safari\	Cache.db
	3	Vista	\AppData\Local\Apple Computer\Safari\	
Cookies	*Not supported at this time*			
Opera				
Cache	9	XP	\Local Settings\Application Data\Opera\Opera\profile\cache4\	*(all files & subfolders)*
		Vista	\AppData\Local\Opera\Opera\Profile\cache4\	
History		XP	\Application Data\Opera\Opera\profile\	typed_history.xml
			\Application Data\Opera\Opera\profile\	download.dat
			\Application Data\Opera\Opera\profile\	global.dat
		Vista	\AppData\Roaming\Opera\Opera\Profile\	typed_history.xml
			\AppData\Roaming\Opera\Opera\Profile\	download.dat
			\AppData\Roaming\Opera\Opera\Profile\	global.dat
Cookies	*Not supported at this time*			
Google Chrome				
Cache	1	XP	\Local Settings\Application Data\Google\Chrome\User Data\Default\Application\Cache	*(all files & subfolders)*
		Vista	\AppData\Local\Google\Chrome\User Data\Default\	
History		XP	\Local Settings\Application Data\Google\Chrome\User Data\Default\Application\	History
		Vista	\AppData\Local\Google\Chrome\User Data\Default\	
Cookies		XP	\Local Settings\Application Data\Google\Chrome\User Data\Default\Application\	Cookies
		Vista	\AppData\Local\Google\Chrome\User Data\Default\	Cookies

Microsoft Internet Explorer

At least since the days of Windows 98, Microsoft Internet Explorer (MSIE) has been the default Internet browser for the Windows platform, both for 32-bit and 64-bit implementations. *Cache design* versioning for this browser was last modified with Version 5.2 which has remained the default schema through to and including today's Version 8. Thankfully, and unlike other browsers, MSIE has maintained its methodology of how it stores information for History and Cache artifacts.

For a *non-authoritative* detailed history of the browser, visit: http://www.wikipedia.org. (see Glossary of Terms for Wikipedia.org)

To begin our discussion, it makes sense to first point out the logical file paths where MSIE stores its information. We begin by indicating that MSIE uses a recurring file named **index.dat** to manage the mapping of data for History, Cookies, and Cache. This file is a proprietary, binary formatted file that maps out URLs (Universal Resource Locator) and their associated data (in the case of cache or Temporary Internet Files). Copies of the **index.dat** file exist for each user profile on a computer, where Internet Explorer has been used. *NOTE: "Temporary Internet Files" is the "cache" folder name for IE (both terms can be used interchangeably).*

Example of a URL:
http://www.google.com
http://images.google.ca/imghp?hl=en&tab=wi

On a Windows XP machine, all user profiles are stored in the **C:\Documents and Settings** folder. On a Windows Vista machine, all user profiles are stored in the **C:\Users** folder. A sample user profile name might be "John Doe". Therefore, a user profile's home directory would be:
C:\Documents and Settings\John Doe (on XP) and **C:\Users\John Doe** (on Vista).

Having said that, the following discussion about specific artifacts will be based on a *relative* path to the sample user profile path (as indicated above for XP and Vista). A *relative* path is simply a path that excludes the *known* portion of the path. In our case, the user profile paths noted above are considered our *known* portions. Therefore, for completeness, the following relative paths would assume the pre-pending of the above noted paths.

File locations on Windows XP:

MASTER History File

\Local Settings\History\History.IE5\index.dat

DAILY History File

\Local Settings\History\History.IE5\MSIE0**20090701**20090**702**\index.dat

WEEKLY History File

\Local Settings\History\History.IE5\MSIE0**20090701**20090**708**\index.dat

COOKIES File

\Local Settings\Cookies\index.dat

\Local Settings\Cookies*.txt (all cookie files)

CACHE (Temporary Internet Files)

\Local Settings\Temporary Internet Files\Content.IE5\index.dat

\Local Settings\Temporary Internet Files\Content.IE5*.* *(all files & subfolders)*

File locations on Windows Vista:

MASTER History File

\AppData\Local\Microsoft\Windows\History\History.IE5\index.dat

DAILY History File

\AppData\Local\Microsoft\Windows\History\History.IE5\MSIE0**20090701**20090**702**\index.dat

WEEKLY History File

\AppData\Local\Microsoft\Windows\History\History.IE5\MSIE0**20090701**20090**708**\index.dat

COOKIES File

\AppData\Local\Microsoft\Windows\Cookies\index.dat

\AppData\Local\Microsoft\Windows\Cookies*.txt (all cookie files)

CACHE (Temporary Internet Files)

\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\index.dat

\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5*.* (all sub-folders)

Mozilla Firefox

The history of the Firefox browser began with its original release of Version 1.5 on November 30, 2005. Version 2.0 followed on October 24, 2006 and Internet histories were then being formatted using the non-standardized Mork Database (plain text) format. This presented a challenge for third party developers in their attempts at understanding the format and at writing supporting tools. Documentation was never formalized until many months later, at the response of much criticism by the developer community.

Version 3.0 was released on December 8, 2008 and introduced the SQLite3 open-source database file format to manage Internet histories. This marked a significant change in the development processes behind Firefox.

CacheBack today supports both Version 2 and Version 3 cache and history file formats. Interestingly, both versions of the browser occupy *generally,* the same file locations on Windows XP/Vista and "share" the same folder locations for history and cache files. This makes it initially difficult to discern which version is the active version based solely on the file and folder hierarchy. However, the presence of the "history.dat" (Version 2 history) file and/or the "places.sqlite" (Version 3 history) files could be the indicators about which is present, or current.

For a *non-authoritative* detailed history of the browser, visit: http://www.wikipedia.org. (see Glossary of Terms for Wikipedia.org)

As discussed earlier in the section on Microsoft Internet Explorer, the History and the Cache files for Mozilla Firefox are also stored separately for "each user profile" on a Windows XP or Vista system. That having been said, the following are the expected file and folder locations for the different Firefox artifacts, *relative to the current user profile folder location*.

File locations on Windows XP:

History

\Application Data\Mozilla\Firefox\Profiles\<name>.default**history.dat** (Version 2)

\Application Data\Mozilla\Firefox\Profiles\<name>.default**places.sqlite** (Version 3)

Cache

\Local Settings\Application\Data\Mozilla\Firefox\Profiles\<name>.default**Cache**

Cookies

\Application\Data\Mozilla\Firefox\Profiles\<name>.default**cookies.txt** (Version 2)

\Mozilla\Firefox\Profiles\<name>.default**cookies.sqlite** (Version 3)

File locations on Windows Vista:

History

\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default**history.dat** (Version 2)

\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default**places.sqlite** (Version 3)

Cache

\AppData\Local\Mozilla\Firefox\Profiles\<name>.default**Cache** (Version 2 and 3)

Cookies

\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default**cookies.txt** (Version 2)

\AppData\Roaming\Mozilla\Firefox\Profiles\<name>.default**cookies.sqlite** (Version 3)

Apple Safari

Unlike the other major browsers, the Safari browser by Apple originated as *the* default browser for the Macintosh operating system. For many years, Safari has been a Mac-only browser. However, in June 2007, the first Windows version was introduced as a Beta. From that point forward, Apple continued to provide both Macintosh and Windows versions of their *growing* popular browser.

In June 2009, the Safari browser was released as Version 4 *jointly* for both operating systems. CacheBack presently supports Windows Version 3 of the browser with developer currently underway for Version 4 – and with support for *both* platforms. NOTE: The Safari browser (Version 3) uses an XML file format to manage its cache, history and cookie data. The respective files all use the file extention: **plist**.

For a detailed history of the browser, visit: http://en.wikipedia.org/wiki/Safari_(web_browser)

As discussed earlier in the section on Microsoft Internet Explorer, the History and the Cache files for Apple Safari are also stored separately for "each user profile" on a Windows XP or Vista system. That having been said, the following are the expected file and folder locations for the different Safari artifacts, *relative to the current user profile folder location*.

File locations on Windows XP:

History

\Application Data\Apple Computer\Safari**History.plist** (Version 3)

Cache

\Local Settings\Application Data\Apple Computer\Safari**Cache.db**

Cookies

\Application Data\Apple Computer\Safari\Cookies**Cookies.plist**

File locations on Windows Visa:

History

\AppData\Roaming\Apple Computer\Safari**History.plist** (Version 3)

Cache

\AppData\Local\Apple Computer\Safari**Cache.db**

Cookies

not support at this time

Opera

While the Opera browser is a lesser used program among the other favored browsers, it still has a faithful following.

Unlike the conventional use of more popular and structured file formats such as SQLite3 and XML, Opera stores its data in a series of proprietary binary files. There is however some similarities in the way Opera stores and manages its cache data using an "index" file system.

CacheBack presently supports Opera Version 9.

For a *non-authoritative* detailed history of the browser, visit:
http://en.wikipedia.org/wiki/History_of_the_Opera_web_browser

As discussed earlier in the section on Microsoft Internet Explorer, the History and the Cache files for Opera are also stored separately for "each user profile" on a Windows XP or Vista system. That having been said, the following are the expected file and folder locations for the different Opera artifacts, *relative to the current user profile folder location*.

File locations on Windows XP:

History

\Application Data\Opera\Opera\profile**typed_history.xml**

\Application Data\Opera\Opera\profile**download.dat**

\Application Data\Opera\Opera\profile**global.dat**

Cache

\Local Settings\Application Data\Opera\Opera\profile**cache4**

Cookies

\Application Data\Opera\Opera\profile**cookies4.dat**

File locations on Windows Vista:

History

\AppData\Roaming\Opera\Opera\Profile**typed_history.xml**

\AppData\Roaming\Opera\Opera\Profile**download.dat**

\AppData\Roaming\Opera\Opera\Profile**global.dat**

Cache

\AppData\Local\Opera\Opera\Profile**cache4**

Cookies

not support at this time

Google Chrome

On September 2, 2008, Google debuted its first browser called “Chrome” (short for “Chromium”) with the promise of new features, unlike competitors. Google’s Chrome boasted superior speed performance through a new Javascript engine as well as features such as DNS pre-fetching.

By mid-2009, Google had released Version 2 of its browser with a Version 3 currently in Beta.

While CacheBack presently supports Version 1 to date, it will very soon have support for the other versions, not already inherently supported by Version 2.7.4.

For a *non-authoritative* detailed history of the browser, visit: http://en.wikipedia.org/wiki/Google_Chrome

As discussed earlier in the section on Microsoft Internet Explorer, the History and the Cache files for Goolge Chrome are also stored separately for “each user profile” on a Windows XP or Vista system. That having been said, the following are the expected file and folder locations for the different Google Chrome artifacts, *relative to the current user profile folder location*.

File locations on Windows XP:

History

\Local Settings\Application Data\Google\Chrome\User Data\Default\Application**History**

Cache

\Local Settings\Application Data\Google\Chrome\User Data\Default\Application**Cache**

Cookies

\Local Settings\Application Data\Google\Chrome\User Data\Default\Application**Cookies**

File locations on Windows Vista:

History

\AppData\Local\Google\Chrome\User Data\Default**History**

Cache

\AppData\Local\Google\Chrome\User Data\Default**Cache**

Cookies

\AppData\Local\Google\Chrome\User Data\Default**Cookies**

DATES AND TIMES

Introduction

When it comes to Internet forensics and browser artifacts, one of the most important components to any analysis is the proper interpretation, conversion, and representation of dates and times. Understanding how each browser stores timestamps is "critical" to an investigation, especially considering the variations imposed by "local time", "time zones" and "Daylights Savings". This issue is even further compounded by the fact that different browsers store dates and times differently. Therefore, it is very important for examiners to be able to differentiate between the different formats, and more importantly, be able to manually validate their original binary values.

Prior to CacheBack Version 2.7, timestamps (for when website URLs were "Last Visited") were recorded strictly in Greenwich Mean Time (GMT). Furthermore, not all browsers used the same terminology to describe the timestamps that were commonly associated to the action of "visiting" or "accessing". While storing timestamps in Universal Coordinated Time (UTC) (also known as GMT time) relieved the concerns of having to account for Daylight Savings, it presented a challenge in allowing CacheBack users to search for items based on their own local time. For instance, if an activity took place at -0500 EST (Eastern Standard Time), CacheBack would store the time as "0000" UTC. Therefore, users would have to take into account various time zone offsets as possibilities in their search, thereby widening their search unnecessarily. More importantly, this behavior of storing time in UTC might not have seemed clear to many users.

With Version 2.7, CacheBack has done away with the "Last Visited" database column. Instead, it has replaced it with a much more meaningful implementation using the descriptors: **Action**, **ActionDateUTC** and **ActionDateLocal** *as column names*. For instance, if we use the same example noted above, our Action value might be "Visited", the ActionDateUTC would equal to "0000", and the ActionDateLocal would logically be "-0500". This now allows users the option to search by either time zone offset as well as specific "Actions" (NOTE: Actions is discussed later on this manual).

What is also very interesting to know is that with the rising popularity of the 64-bit architecture for processors and software, data types, used to store timestamps, previously consisted of 4 bytes and 8 bytes, depending on the browser and meaning of the timestamp. Today, most if not all timestamps utilize a full 16-bytes to store their values. This makes it easier now to store time in *seconds, milliseconds, microseconds and nanoseconds*. Being able to convert these

values into *meaningful and accurate* timestamps is therefore very critical for any investigation relying on precise time analysis.

From a forensic context, most examiners have a general or common understanding of the Windows Registry artifacts relating to Time Zone information, and in particular, a computer's "time" and "time zone settings". While the detailed discussion about Registry artifacts is outside the scope of this document, it is simply important to understand that there are two distinct values maintained in the Registry for the purpose of calculating time. The first is the "Active Bias" setting and the "Bias" setting. Both values correspond to the computer's Time Zone or GMT offset from UTC time (eg: Toronto is 5 hours ahead of UTC or better known as "-05:00" hours).

A detailed discussion about how these values impact the calculation of timestamps in CacheBack is discussed later on.

NOTE:

The following sections discuss advanced time related issues that are essential for proper date and time analysis from a forensic context. To ensure that CacheBack users are provided with *accurate and qualified* information, we have referenced materials made available by the Naval Oceanography Portal, copyrighted by the Naval Meteorology and Oceanography Command, situated at 1100 Balch Blvd, Stennis Space Center, MS 39529, United States. Passages that are derived from the Naval Oceanography Portal are indicated by the acronym **USNO**.

Understanding Coordinated Universal Time (UTC)

[USNO] The times of various events, particularly astronomical and weather phenomena, are often given in "Universal Time" (abbreviated UT) which is sometimes referred to, now colloquially, as "Greenwich Mean Time" (abbreviated GMT). The two terms are often used loosely to refer to time kept on the Greenwich meridian (longitude zero), five hours ahead of Eastern Standard Time. Times given in UT are almost always given in terms of a 24-hour clock. Thus, 14:42 (often written simply 1442) is 2:42 p.m., and 21:17 (2117) is 9:17 p.m. Sometimes a Z is appended to a time to indicate UT, as in 0935Z.

When a precision of one second or better is needed, however, it is necessary to be more specific about the exact meaning of UT. For that purpose different designations of Universal Time have been adopted. In astronomical and navigational usage, UT often refers to a specific time called UT1, which is a measure of the rotation angle of the Earth as observed astronomically. It is affected by small variations in the rotation of the Earth, and can differ slightly from the civil time on the Greenwich meridian. Times which may be labeled "Universal Time" or "UT" in data provided by the U.S. Naval Observatory (for example, in the annual almanacs) conform to this definition.

However, in the most common civil usage, UT refers to a time scale called "Coordinated Universal Time" (abbreviated **UTC**), which is the basis for the worldwide system of civil time. This time scale is kept by time laboratories around the world, including the U.S. Naval Observatory, and is determined using highly precise atomic clocks. The International Bureau of Weights and Measures makes use of data from the timing laboratories to provide the international standard UTC which is accurate to approximately a nanosecond (billionth of a second) per day. The length of a UTC second is defined in terms of an atomic transition of the element cesium under specific conditions, and is not directly related to any astronomical phenomena.

UTC is the time distributed by standard radio stations that broadcast time, such as WWV and WWVH. It can also be obtained readily from the Global Positioning System (GPS) satellites. The difference between UTC and UT1 is made available electronically and broadcast so that navigators can obtain UT1. UTC is the basis for civil standard time in the U.S. and its territories. Standard time within U.S. time zones is an integral number of hours offset from UTC.

UTC is equivalent to the civil time for Iceland, Liberia, Morocco, Senegal, Ghana, Mali, Mauritania, and several other countries. During the winter months, UTC is also the civil time scale for the United Kingdom and Ireland.

One can think of UT1 as being a time determined by the rotation of the Earth, over which we have no control, whereas UTC is a human invention. It is relatively easy to manufacture highly precise clocks that keep UTC, while the only "clock" keeping UT1 precisely is the Earth itself. Nevertheless, it is desirable that our civil time scale not be very different from

the Earth's time, so, by international agreement, UTC is not permitted to differ from UT1 by more than 0.9 second. When it appears that the difference between the two kinds of time may approach this limit, a one-second change called a "leap second" is introduced into UTC. This occurs on average about once every year to a year and a half.

The International Date Line

Time Zones are essential in calculating *local time* for different parts of the world. We use time zones and adjust our local time based on *offsets* from UTC (0000). Because the United Kingdom observes UTC throughout the year, it will be our reference hereinafter for the purposes of UTC discussions.

The following are common time zone offsets for different countries. These values do NOT take into account Daylight Savings:

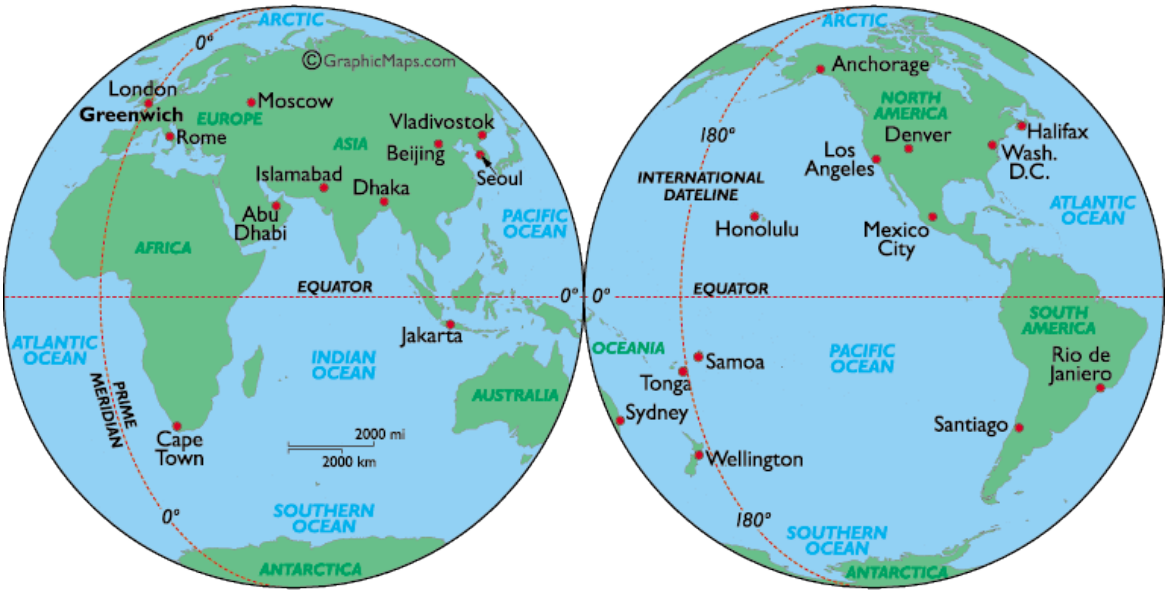
1. Toronto, Ontario (Canada): -0500 UTC
2. Los Angeles, California (USA): -0800 UTC
3. Sydney, New South Wales (Australia): +1000 UTC

You will notice that the above discussion about offsets was prefaced with the fact that the above examples do NOT take into account Daylight Savings. This is quite important to understand because it precipitates a discussion about "Hemispheres" which is often not discussed in forensic contexts. Hemispheres are critical to the calculation of time zone offsets in conjunction with Daylight Savings for different parts of the world, during special times of the year.

[USNO] The International Date Line is the imaginary line on the Earth that separates two consecutive calendar days. That is the date in the Eastern hemisphere, to the left of the line, is always one day ahead of the date in the Western hemisphere. It has been recognized as a matter of convenience and has no force in international law.

Without the International Date Line travelers going westward would discover that when they returned home, one day more than they thought had passed, even though they had kept careful tally of the days. This first happened to Magellan's crew after the first circumnavigation of the globe. Likewise, a person traveling eastward would find that one fewer days had elapsed than he had recorded, as happened to Phileas Fogg in "Around the World in Eighty Days" by Jules Verne.

The International Date Line can be anywhere on the globe. But it is most convenient to be 180° away from the defining meridian that goes through Greenwich, England. It also is fortunate that this area is covered, mainly, by empty ocean. However, there have always been zigs and zags in it to allow for local circumstances.



Eastern Hemisphere

Tonga: 8 Jul 2009 02:12:48 UTC

Western Hemisphere

Samoa: 9 Jul 2009 02:12:48 UTC

Daylight Time

[USNO] Starting in 2007, daylight time begins in the United States on the second Sunday in March and ends on the first Sunday in November. On the second Sunday in March, clocks are set ahead one hour at 2:00 a.m. local standard time, which becomes 3:00 a.m. local daylight time. On the first Sunday in November, clocks are set back one hour at 2:00 a.m. local daylight time, which becomes 1:00 a.m. local standard time. These dates were established by Congress in the Energy Policy Act of 2005, Pub. L. no. 109-58, 119 Stat 594 (2005).

Not all places in the U.S. observe daylight time. In particular, Hawaii and most of Arizona do not use it. Indiana adopted its use beginning in 2006.

- In 2006, daylight time begins on April 2 and ends on October 29.
- In 2007, daylight time begins on March 11 and ends on November 4. [New law goes into effect.]
- In 2008, daylight time begins on March 9 and ends on November 2.
- In 2009, daylight time begins on March 8 and ends on November 1.

Many other countries observe some form of "summer time", but they do not necessarily change their clocks on the same dates as the U.S.

History of Daylight Time in the U.S.

[USNO] Although standard time in time zones was instituted in the U.S. and Canada by the railroads in 1883, it was not established in U.S. law until the Act of March 19, 1918, sometimes called the Standard Time Act. The act also established daylight saving time, a contentious idea then. Daylight saving time was repealed in 1919, but standard time in time zones remained in law. Daylight time became a local matter. It was re-established nationally early in World War II, and was continuously observed from 9 February 1942 to 30 September 1945. After the war its use varied among states and localities. The Uniform Time Act of 1966 provided standardization in the dates of beginning and end of daylight time in the U.S. but allowed for local exemptions from its observance. The act provided that daylight time begin on the last Sunday in April and end on the last Sunday in October, with the changeover to occur at 2 a.m. local time.

During the "energy crisis" years, Congress enacted earlier starting dates for daylight time. In 1974, daylight time began on 6 January and in 1975 it began on 23 February. After those two years the starting date reverted back to the last Sunday in April. In 1986, a law was passed that shifted the starting date of daylight time to the first Sunday in April, beginning in 1987. The ending date of daylight time was not subject to such changes, and remained the last Sunday in October. The Energy Policy Act of 2005 changed both the starting and ending dates. Beginning in 2007, daylight time starts on the second Sunday in March and ends on the first Sunday in November.

Summer Time (Northern and Southern Hemispheres)

The following is a list of territories that observe Summer Time from March to October/November:

- Europe
- North America
- Central America / Carribean
- Asia
- Africa (Egypt, Morocco, Tunisia)

The following is a short list of territories that observe daylight savings opposite to the Northern Hemisphere countries:

- Australia / Oceania
- South America
- Africa
- Antarctica

Hemispheres and Daylight Saving Time Issues

Now that we have an understanding of UTC, the International Date Line (aka: the Prime Meridian), and Daylight Time, it sets the groundwork for discussion about Daylight Saving and how different hemispheres impact date and time analysis from a forensic context.

The following scenario introduces some interesting facts about a make-believe investigation, which real-life investigators are likely to encounter in multi-jurisdictional case.

FACT #1

You are an F.B.I. agent working in New York City, New York, USA. New York City has a standard UTC offset of -0500.

FACT #2

The time now is **8-July-2009 07:05 AM**. Since this time falls within the U.S. observed Daylight Time, the offset is now **-0400 UTC.** Daylight Saving (Summer Time) **is** in effect.

FACT #3

It just so happens that the computer you are examining was shipped to you by a federal law enforcement contact in Sydney, Australia. Apparently the offence took place in Australia and the subject computer was also configured properly for Australian time (with Daylight Savings configured).

FACT #4

Evidence in this case (eg: witness statements, events) suggest that the alleged offence date was **February 5, 2009** and that Internet activities occurred in or about **09:43 AM**.

FACT #5

During your forensic analysis of the Internet activity for the subject computer, you come across several browser cache and history URL records for the date of February, 5, 2009. You are using a combination of tools for your analysis, including CacheBack.

ISSUE

In the above scenario, some third party tools (other than CacheBack) might erroneously represent timestamps as **08:43 AM** because the examiner's workstation is configured to calculate Daylight Savings, which is based on his location in the

Northern Hemisphere. It is an instinctive approach by developers to reference time properties of the *local machine* and not the *evidence origin* itself.

As a result, this approach would provide a terribly inaccurate time value because the offence time is calculated using the wrong hemisphere as a variable to the equation.

CacheBack 2.7 has taken this anomaly into account when reporting / displaying timestamps from different time zones AND from different hemispheres. Using the selected Time Zone and Daylight Saving option within the program, CacheBack will not only properly calculate the time zone offset, but take into account the *actual daylight saving* value for *the selected time zone* and not *the time zone of the examiner's workstation*.

Hence, the timestamp for the above example would be reported accurately by CacheBack as **09:43 AM**. Lastly, as an added validation tool, CacheBack reports all timestamps in UTC time as a separate column entitled: "Action Time UTC".

***SPECIAL NOTE**: The initial implementation of hemisphere calculations introduced into CacheBack 2.7 using Eastern and Western Hemispheres and remained in effect with version 2.7.3. It distinguished dates using negative offsets (eg: North America) and positive offsets (eg: Australia) from the Prime Meridian (Longitude zero: Greenwich) as opposed to latitudinal locations using Northern and Southern hemispheres. While the former approach was effective in most circumstances (based on the demographics of international users of CacheBack at that time), it has been expanded upon more accurately 2.7.4.*

This revision of CacheBack now calculates the hemisphere attribute for an individual country based on their Northern or Southern hemisphere location, and not their UTC offset (or relevant position to the Prime Meridian). Based on an examiner's Time Zone selection within CacheBack, datestamps are now more properly calculated for all countries, in all time zones. This is reflected in any "displayed" or "reported" time value by the use of the "DST" or "STD" suffix.

Daylight Time (Northern and Southern Hemispheres)

The following section summarizes the discussion about dates and times in a manner that is easier to remember and understand.

We showed you earlier a map of the globe, divided by Eastern and Western hemispheres. We also discussed how time is accurately calculated based on THREE main ingredients:

1. The *UTC (GMT) offset* for the specific region,
2. The *hemisphere* in which the specific region is situated, and lastly,
3. Whether or not the specific region *observes daylight savings*.

Let's take a quick look at the global map one more time, this time, noting the *Northern and Southern Hemispheres* which are divided by the Earth's Equator.

Diagram -1 – Northern Hemisphere

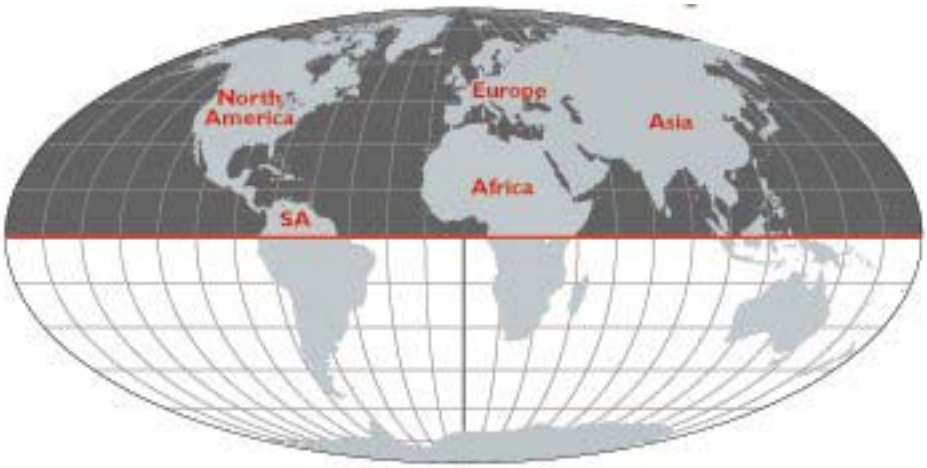


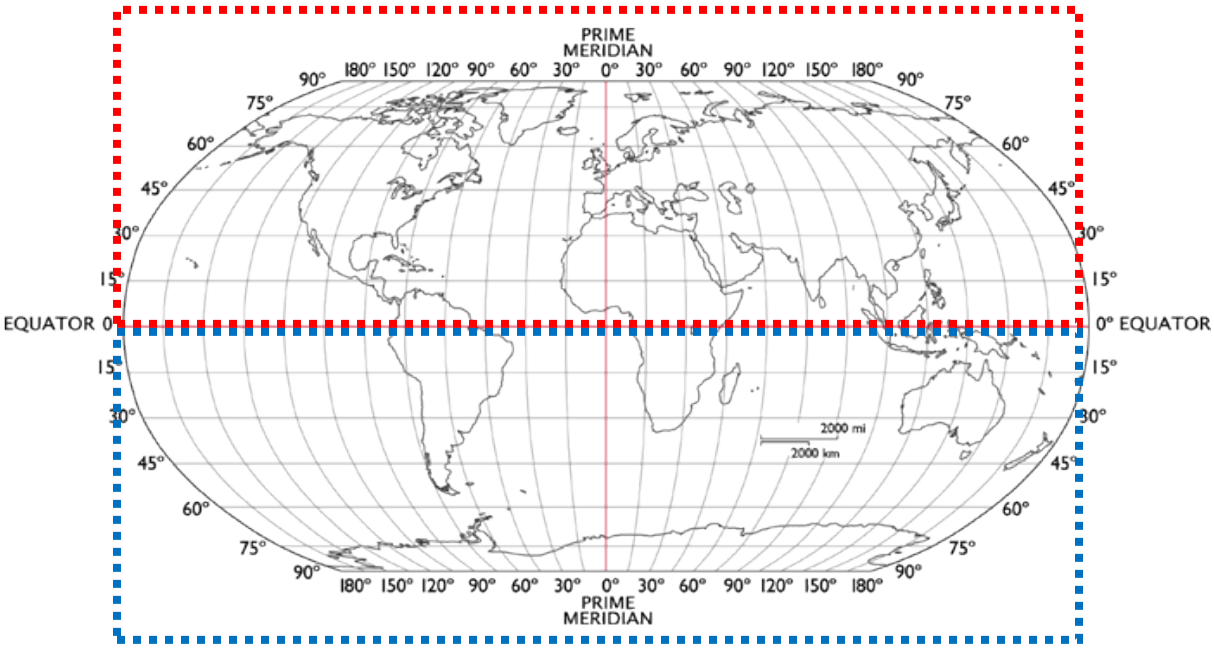
Diagram -1 – Southern Hemisphere

For those countries situated South of the Equator, *that do observe daylight savings*, the start and end of Daylight Savings (season) will be *opposite* of those countries to the North. Let's see what that really means…

In the following diagram, RED (Northern Hemisphere) outlines where Daylight Saving Time would be in effect on **July 28, 2009**. The color BLUE (Southern Hemisphere) indicates where Summer Time would be observed on the same date. The trick to remember is that as we move closer to the East where the sun rises, our UTC offsets move as well!

EXAMPLE:

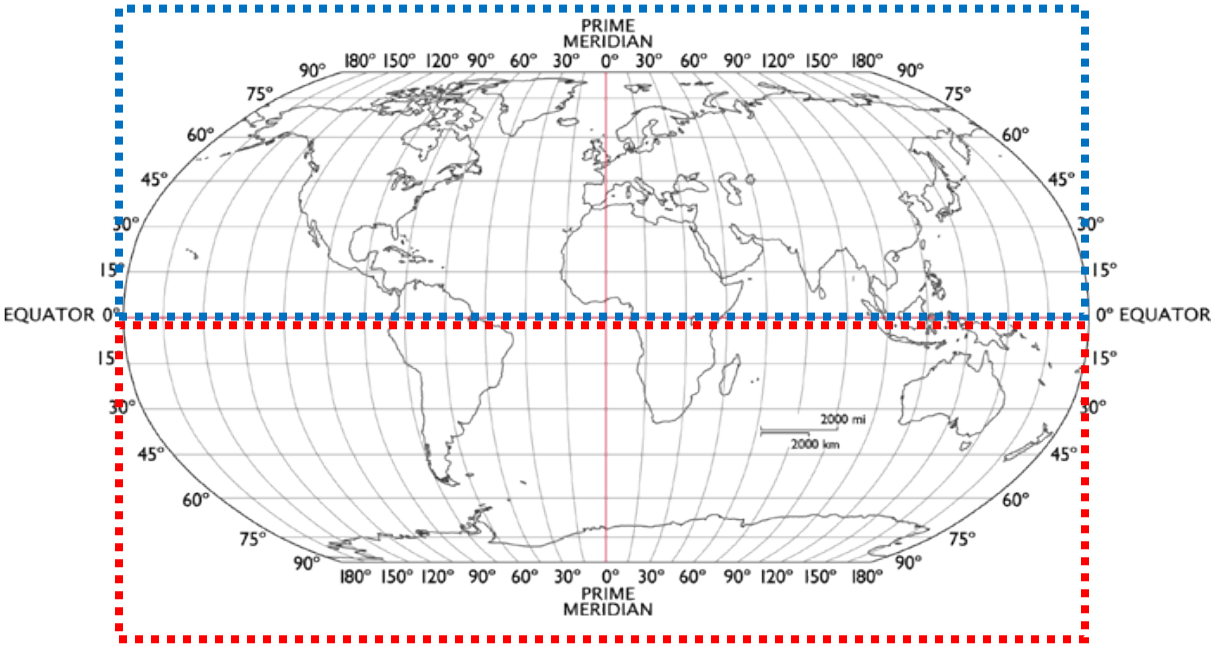
Toronto, Canada:	Eastern **Daylight** Time (EDT) **-04:00** UTC
London, UK:	British **Summer** Time (BST) **+01:00** UTC
Sydney, Australia:	Australian Eastern **Standard** Time (EAST) **+10:00** UTC



If on the other hand, the date was ***December 31, 2009,*** then Daylight Savings would be observed differently:

EXAMPLE:

Toronto, Canada:	Eastern **Standard** Time (EST) **-05:00** UTC
London, UK:	Greenwich Mean Time (GMT) **+00:00** UTC
Sydney, Australia:	Australian Eastern **Daylight** Time (EADT) **+11:00** UTC



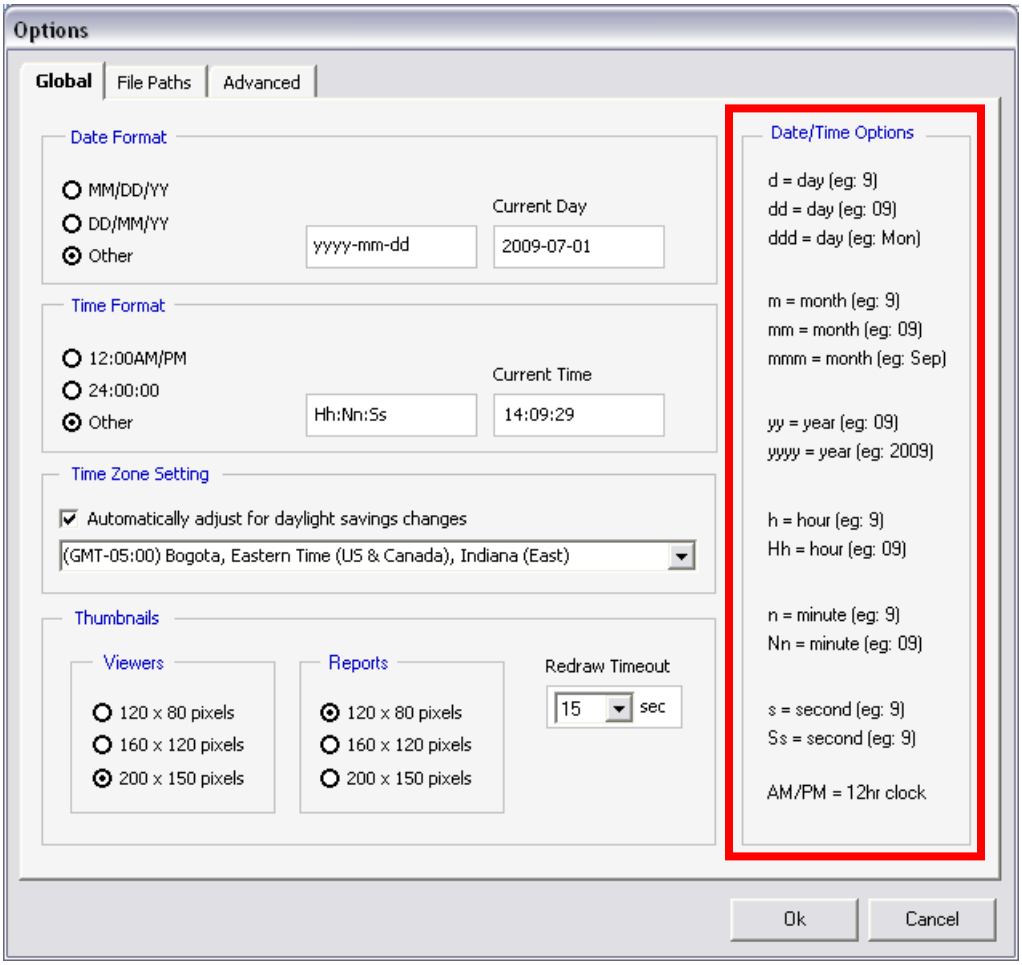
Formatting Displayed Times and Dates in CacheBack

With CacheBack 2.7, users can now configure the way dates and times are displayed and reported using the new Options Window, which is available from the View menu.

Users are encouraged to use the default display format (see below) as the value is easily sorted in chronological order, whether as a true *date* data type, or as a simple *string* data type. The default format also makes use of the 24 hour clock for added clarity.

Example: **`YYYY-MM-DD Hh:Nn:Ss`** will produce today as **`2009-07-06 11:17:34`**.

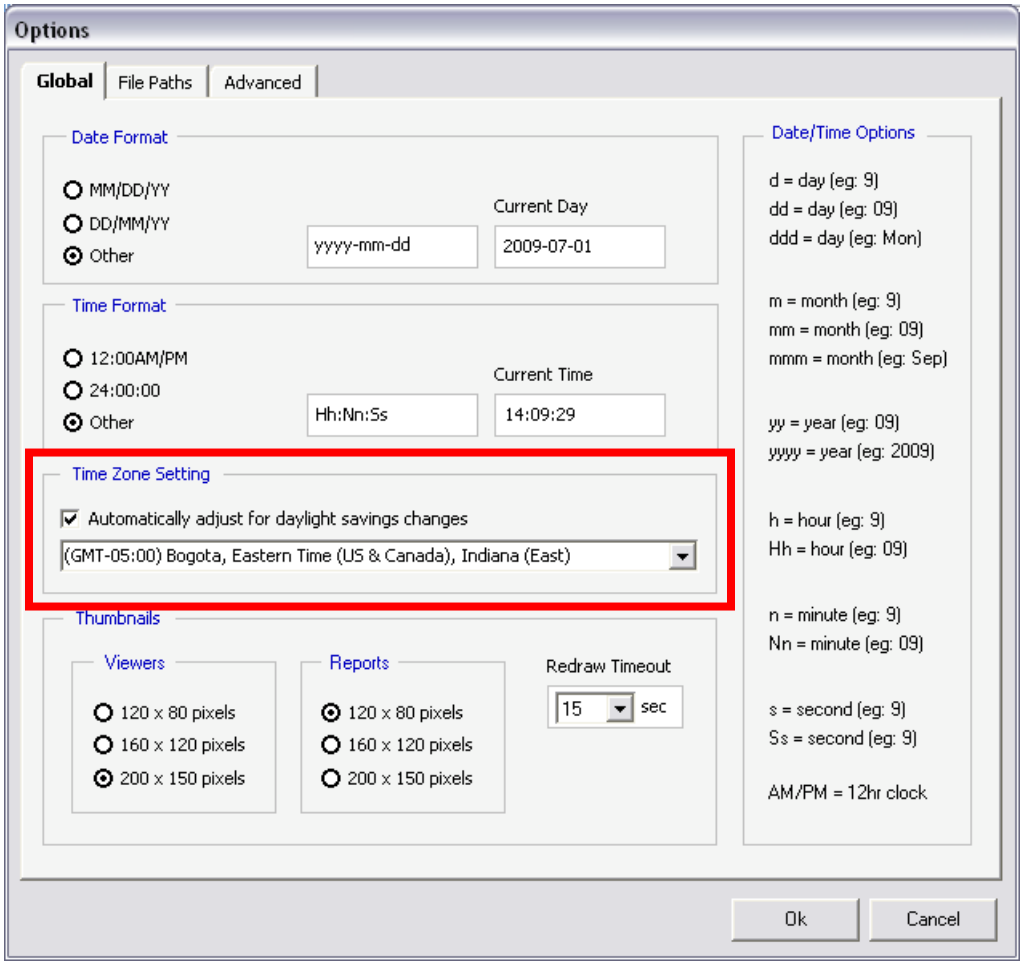
NOTE: The permitted masking characters are displayed on the right hand side of the window.



Setting Time Zone and Daylight Savings Options

By default, CacheBack will configure the Time Zone and Daylight Saving preferences to the values recorded in the *examiner's (CacheBack user's)* Windows system. This will be reflected inside the Global Tab of the Options Window under the Time Zone Setting area, of the dialogue box. The Options Window is accessed via the View menu on the top menu bar (above the Toolbar). As indicated below, this area allows users to select a specific Time Zone in which all dates and times are to be converted, for displaying and reporting purposes. The "**Automatically adjust for daylight savings changes**" option is provided in order to address time artifacts that may be influenced by this feature.

Some third party forensic tools base their reporting of timestamps using the Daylight Saving setting of the "examiner's computer", and not that of the *origin of the timestamp* (eg: a computer seized in an opposing Hemisphere as discussed earlier). This would result in timestamps being inaccurately reported as either *plus or minus one hour*.



ActionDateLocal and ActionDateUTC

These two database columns store URL date related artifacts in a Date format.

The **ActionDateLocal** column holds a converted version of the *ActionDateUTC* value. The storage and conversion of this timestamp is performed *only once during the initial Import process*, using the Time Zone Settings found in the Options Window. If a user changes the current Time Zone settings to another time zone, the existing *stored values for ActionDateLocal* (as it is stored within the project file) will remain unchanged. However, the *displayed and reported* dates and times for *ActionDateLocal*, are *converted at run-time using the current Time Zone Setting* from the Options Window.

NOTE: The above distinction about when the database column ActionDateLocal is assigned a value is very important for users who will be conducting custom queries. It is therefore recommended that date-based queries, whenever possible, be conducted using ActionDateUTC *instead.*

The **ActionDateUTC** column holds the actual (or calculated) UTC version of the primary timestamp associated to a URL. If the URL record being examined (or parsed by CacheBack) relates to an activity such as "Visiting" a website, then this timestamp will be used. In addition, the "activity" will be described (stored) in the **Action** column.

NOTE: The use of the Time Zone Settings feature in the Options Window will have no affect on the value stored in the ActionDateUTC as this column is assigned a value only once. This is done at the initial importing and parsing of browser artifacts.

WEEKLY Timestamps in Internet Explorer

Timestamps for WEEKLY *History* records (URLs) are stored inside the INDEX.DAT file as the "Local Time" from the computer where the actual URL record entry was created (eg: Australia).

Examiners who are importing an IE Weekly index.dat file into CacheBack MUST have the CacheBack Time Zone option set to the examiner's own time zone. This is NECESSARY in order for CacheBack to properly determine the true UTC time.

Once the importing is complete (for any given WEEKLY index.dat file), then the examiner can feel free to re-configure the CacheBack Time Zone option as required.

IMPORTANT:

Since a subject's computer's Time Zone and Daylight Saving settings may NOT be set correctly (eg: defaulting to Pacific Standard Time during Windows install), CacheBack will initially convert all browser UTC times to local time during the IMPORT stage using the examiner's workstation time zone settings (eg: Project file default).

These UTC timestamps are converted and stored into the ActionDateLocal database column. As of Version 2.7.4, this assignment to the ActionDateLocal column is done ONCE and cannot be altered after import. The logic behind this approach is to allow examiners to conduct custom queries using the ActionDateLocal time that is relative to the time zone in which the investigation is taking place. This also works around the significant possibility that the subject's computer has not been changed from the Windows installation default of Pacific Standard Time which is -08:00 UTC.

All ActionDateLocal times "displayed" (eg: Table view) and "reported" are on-the-fly ActionDateUTC conversions based on the currently selected Project-file-level time zone setting (which is selected via the Options Window). This approach ensures that times are more true than relying on (and having to convert) the subject's local timestamps. This also allows for the examiner to view the timestamps in any time zone, as often as required, during the course of an investigation.

"DST" and "STD" Suffixes

Timestamps that are *reported and displayed* within CacheBack can represent any date of a given year.

Countries that observe Daylight Savings throughout the year will typically commence Daylight Savings either in the Spring (for countries in the Northern Hemisphere), or the Fall (for countries in the Southern Hemisphere).

As a result, examiners have to be cognizant of this fact when reviewing evidence from two different *daylight periods*. The *correct* UTC offset has to be taken into account for the *daylight period of the evidence*, and NOT the *daylight period of the examiner*!

EXAMPLE:

John Doe of New York City is working a case in July. New York City observes daylight savings in accordance with USA Standards and therefore, John is currently observing "Daylight Savings" (Spring to Fall). Therefore, his UTC offset in July would be -0400. The Standard UTC offset for his time zone is -0500.

During John's examination of a case, he comes across a key piece of evidence that apparently occurred at 11:00 PM UTC on February 1st. John began his investigation with the CacheBack Time Zone option set to his own time zone using "(UTC-0500) Eastern Time (US & Canada)" with the "*Automatically adjust for daylight savings changes*" checkbox enabled (selected).

John's initial (manual) interpretation of the 11:00PM UTC timestamp would be to convert the time to 7:00PM using his existing -0400 UTC offset, since he is currently observing daylight savings. *THIS IS INCORRECT!!!* Why? In the month of February, John's *daylight period* does not observe daylight savings. This is the time that is of importance, not the time of the examination. Since the actual UTC offset in February would be -0500, then the 11:00PM UTC timestamp should be properly converted to 6:00PM.

CacheBack 2.7.4 makes this distinction very clear by appending the letters "DST" or "STD" to each and every timestamp that is being displayed or reported.

THE BASICS: BEFORE GETTING STARTED

WHAT Does CacheBack Do?

CacheBack was originally designed as a standalone tool to rebuild web pages from an Internet browser's cache. In Version 2 and through a series of dot releases, CacheBack was expanded to include Internet "history" as part of the analysis offerings.

As of Version 2.7, CacheBack performs the following general functions:

- Locates browser-specific files and folders using a tool called CacheGrab
- Imports and parses History and Cache files into a CacheBack Project (.CBP) file. This is a relational database file that can be opened using MS Access.
- Thumbnails are generated during the import process for the Gallery view option.
- URL records are displayed within the Table or Gallery view options for examination by the user.
- Records can be hidden or show using a combination of queries or filters.
- Table records that originate from a browser's *cache* can be "checkmarked" and then "Rebuilt".
- Rebuilt webpages and any other selected URL records (from the Table or Gallery) can be combined into a single HTML report that is displayed in the Report viewer tab.
- Reports can be Published to a user-defined output folder.
- Dates and times of displayed and reported URL records can be modified at run-time using the Time Zone Settings features on the Options Window.
- Provides a graphical viewer interface using thumbnail views of web pages and pictures thereby making it easier for users to zero in on particular websites.

The WHY and HOW it Works

The WHY

CacheBack was designed to do what it does, and do it in the way that it does it, due largely in part of the investigative requirements of most law enforcements agencies.

Since about the year 2001, there has been virtually no support whatsoever in the *rebuilding of web pages* for Internet browser cache. Considering the fact that most child exploitation (CE) and child pornography (CP) related investigations revolve around "images" and "websites", it is curious as to why no other products (that rebuild web pages) are or have been in existence.

CacheBack was designed to fill this void in the computer forensic landscape.

As CacheBack has evolved since it's original support for Internet Explorer and Firefox 2 browser cache, it became only logical that CacheBack also incorporate the analysis of Internet "history" as well. While there has existed less than a handful of third party products available to parse Internet histories, they all seemed to cater to only a select number of the five (5) available and widely used browsers. For the very reason that CacheBack is constantly being updated to address cache for the different browsers, it only made sense to likewise incorporate support for histories.

As a *sidebar* to the other sections in this manual that address topics in depth, this section highlights a simple number of features that often find themselves in frequently asked questions. The following serves no other purpose than to provide a more personalized context about the development and use of the CacheBack program.

The Graphical User Interface. The approach to designing the interface for CacheBack was relatively straight forward. It was engineered to look and feel like the bigger forensic tools that most forensic examiners had come to know and appreciate.

The Project File. Given the quantity of artifacts likely to be examined in any particular case, and the need to be able to analyze the data quickly, effectively and intelligently, a database file system was essential. For the purposes of scalability and compatibility, a Microsoft Access database schema was chosen for the backend. Future implementations of a storage system will contemplate enterprise databases and open-source options.

The Importing and Parsing. The CacheBack Research and Development Team has spent considerable time reverse engineering the binary files used to store both cache and history files for the top browsers. Extensive custom class libraries have been designed and compartmentalized for easy migration to newer builds in the future. As browsers change the way information is stored and maintained, CacheBack's internal libraries will be expanded to meet new parsing requirements.

Architecture. CacheBack 2.7 is a 32-bit application that is supported on the following Windows platforms: XP, Vista, NT/2000, 2003. Preliminary tests have confirmed that CacheBack will also run on Windows 7 which is due to be released in 2010.

Creation of Rebuilt Web Pages. Select cache files are "read and copied" from their (imported) source location to a **~tmp** file situated in the user-defined Temp folder on their system. CacheBack closely examines the contents of the **~tmp** file and finds the best-match replacement artifacts (eg: pictures, stylesheets). The **~tmp** file is modified, the matching artifacts are harvested to the Temp folder, and the new file is then displayed inside CacheBack. The specific details of how this process is done is discussed later on in the manual.

With the release of CacheBack 2.7, each rebuilt web page is now (a) packaged into a proprietary cabinet file along with all the ingredient files that make up the page (eg: pictures, etc)., (b) compressed, (c) encrypted, and (d) stored back into the database using the *CacheRebuiltFileBLOB* column. While this may add some considerable bloat to the database file, this approach is effective in increasing the performance of the application AND maintaining the integrity and continuity of the evidence.

NOTE: Future releases of CacheBack may attempt to streamline this process in order to reduce the file size. Given the possibility of large scale investigations, this approach is logical. However, in contrast to this idea, it is also possible that an enterprise database solution (backend) could negate the issues about file size.

Thumbnails. Thumbnails are generated initially during the import process of cache and history artifacts. Web pages and pictures are initially used to generate thumbnails. The "BLOB" (binary) data for the thumbnails are stored in the *ThumbnailBLOB* column for the specific URL record entry. As web pages are rebuilt, new thumbnails are created and stored back in the database.

Browser (tab). The Browser viewer tab contains an Internet browser control that is based on the Internet Explorer APIs and provides a default offline view mode. Web pages, that are associated to a URL entry that is selected from the Table or Gallery within CacheBack, are displayed inside the browser control. In this process, some web pages

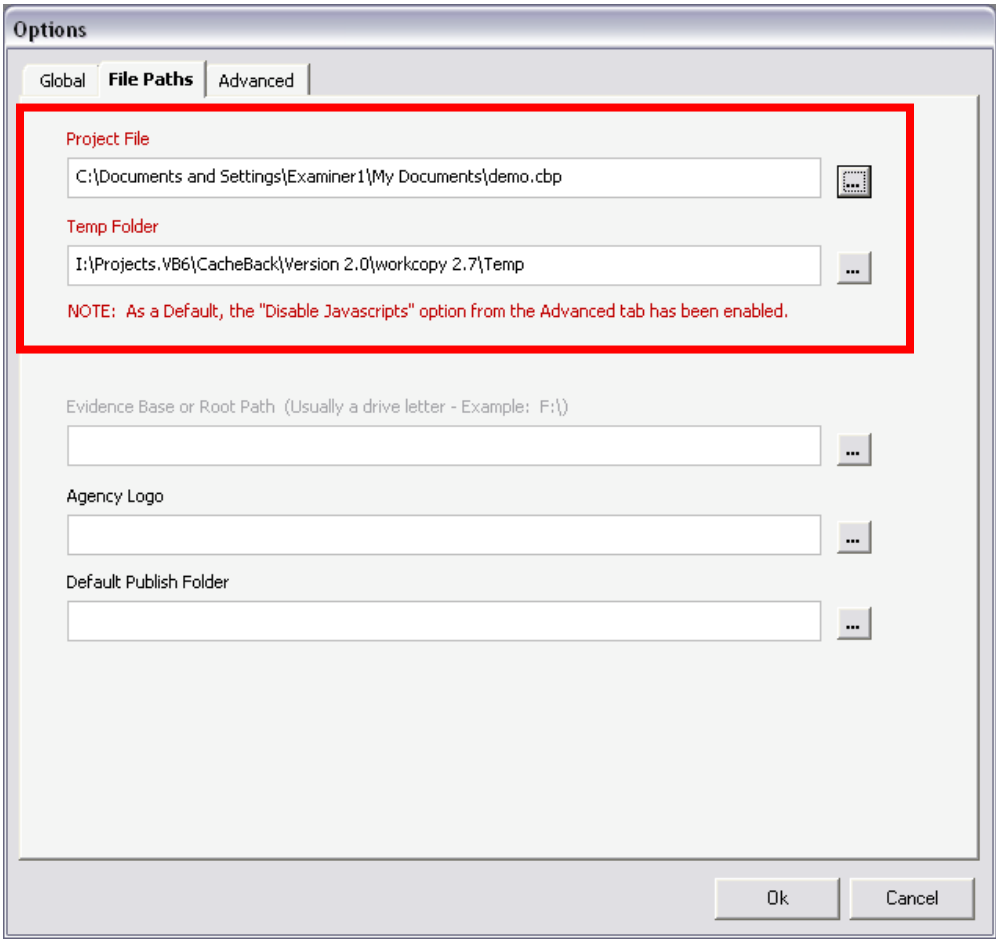
may not display properly OR may not display at all. This is most likely due to the presence of complex javascript(s) inside the source code for the file. While some pages may not display because an outside resource (eg: Internet connection) is not available, it may also be due to CacheBack's default "**Disable <script> tags in web page**" option being in effect (checked). This feature is found in the Options Window under the Advanced tab and has been proven to be an effective tool at preventing "popups" within CacheBack. It also serves to prevent scripts from making calls to outside resources which would normally cause annoying delays while the browser reaches a timeout.

STARTING A NEW PROJECT

Creating a New Project File

After launching CacheBack, the user interface will be blank. To start a new case or project, select the New ( ) button on the Toolbar or alternatively the File menu...followed by the New menu option. The Options Window will then appear and default to the File Paths tab. The only items required at this point is the "Project File" path (to the project file that is to be created) and the "Temp Folder" that will be used to store and manage temporary files.

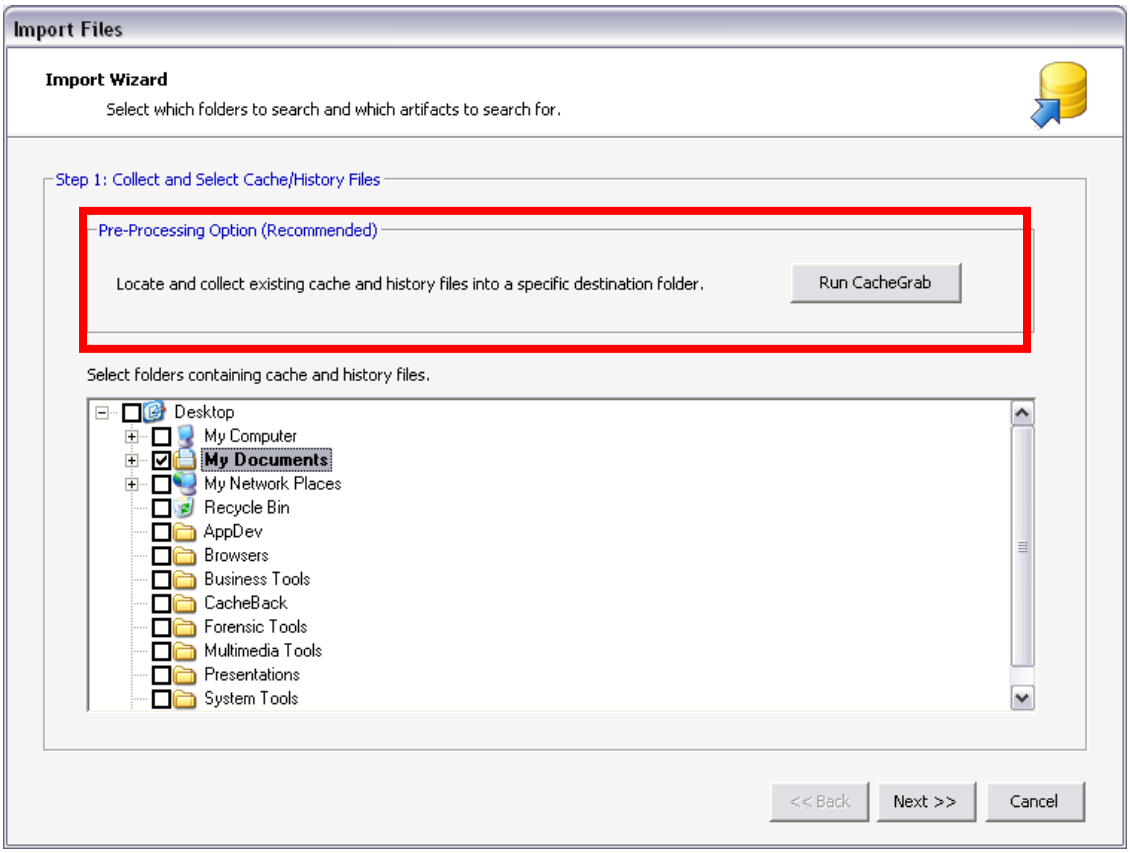
NOTE: The actual project file (.cbp) will be created ONLY after you select the Ok button. From there, the file will remain OPEN and IN USE until you either create a new project, open an existing project, or close the program.



Importing Data

Once the new project file has been created, CacheBack will automatically display Step 1 of 4 of the Import Wizard window. The purpose of Step 1 is to inform CacheBack about the folder location(s) of cache and history files that are to be imported into the new project.

There are two options that can be selected here. The first is a Pre-Processing Option that uses the CacheGrab data collection tool. It is a standalone program designed to compliment and assist CacheBack in identifying valid files for import. As of the writing of his manual, CacheGrab Version 1.2.3 scans any target drive for (a) Windows user profiles, then (b) the selected browser artifacts for each selected profile.



Using CacheGrab®

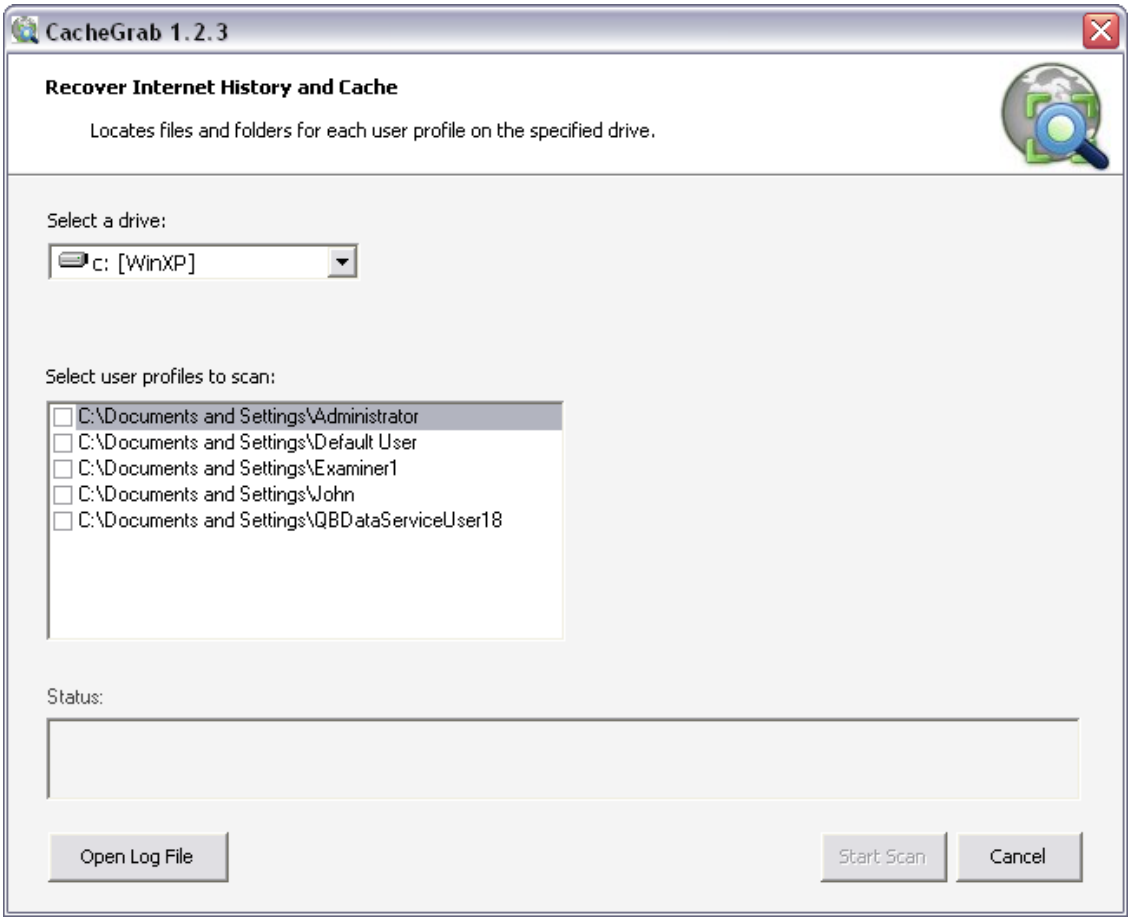
CacheGrab is a standalone program designed to compliment CacheBack by facilitating the collection of browser cache and history files from user profiles on a given Windows XP or Vista operating system.

As of the writing of this manual, Version 1.2.3 is the latest build which not only scans local drive (logical) volumes, but has been successfully tested with mounted volumes (image files) and virtual file systems. Ideally, CacheGrab should be used to scan any volume *other than* the user's own Drive C:. The reason for this is simple. Windows' inherent security measures *may* prevent CacheGrab from accessing restricted folders. This is certainly the case when attempting to recover Internet Explorer History or Temporary Internet Files folders.

"Permission Denied" issues can sometimes occur and therefore it is recommended that drives or volumes that are the subject of a CacheGrab scan be anything other than the Drive C:. This ensures that the operating system will not necessarily prevent CacheGrab from accessing the entire *logical* contents of the drive. While permission errors can sometimes still get in the way of a scan, CacheGrab circumvents this issue by simply copying the topmost folder that is not restricted. For example, it will copy out a user's entire "Local Settings" folder (if it has to) to a user-defined output folder. From there, CacheGrab will use the **SetAttrib** API to remove any Hidden and System attributes thereby (a) revealing the contents of each folder, and (b) disabling any Read Only settings.

CacheGrab comes packaged with the CacheBack software and is installed in the same program directory. A separate shortcut is also made available for CacheGrab in the CacheBack program group. It is also available as a separate download from the CacheBack website (www.cacheback.ca) and through the Download message board found at the CacheBack Message Forums site: http://forums.cacheback.ca).

The following is a screenshot of CacheGrab as it first appears, in this case while running on an XP system. Notice how the drive defaults to “Drive C:” and the discovered user profiles have been loaded into a list box.

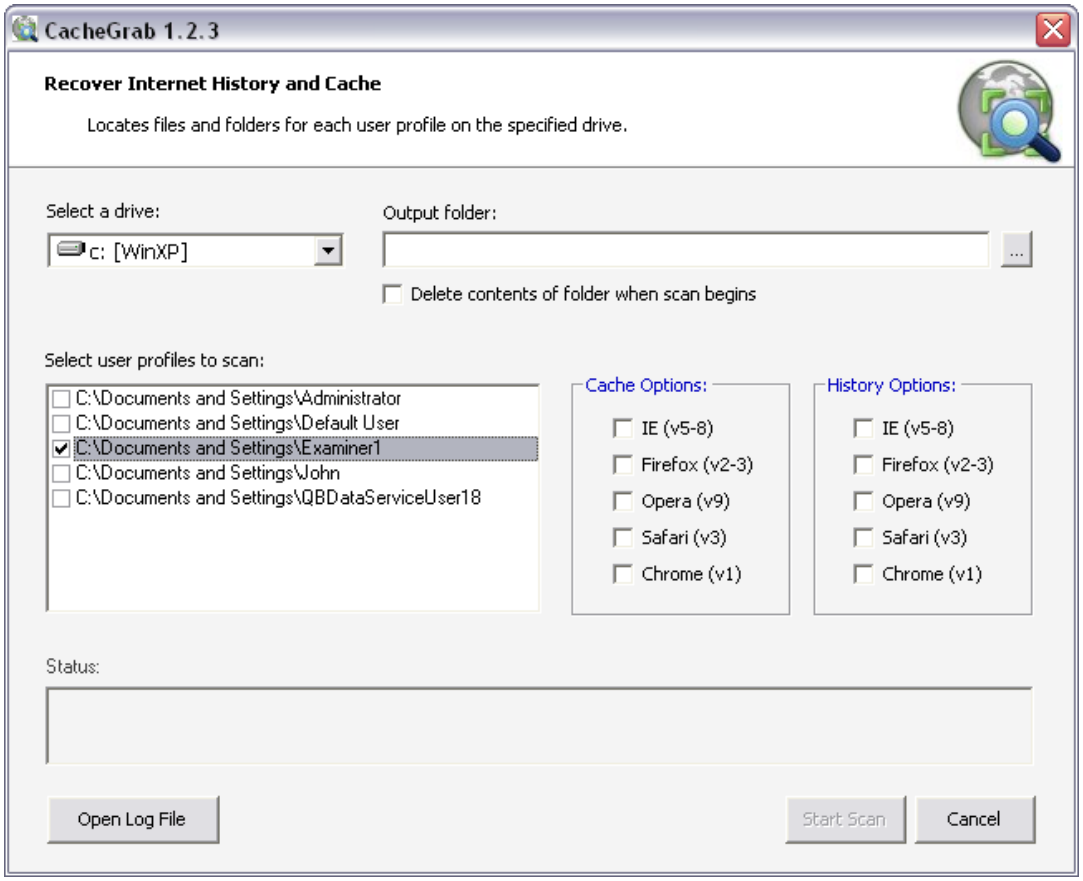


How CacheGrab Works

CacheGrab provides a very simple user interface. Each option becomes available only after the previous, required option has been selected or defined. In the above case, the first step of selecting a drive has been done for us. As a direct result, CacheGrab has scanned the contents of the drive looking for valid XP or VISTA user profiles. Any found profiles are then loaded below, inside a list box.

You will notice that there is a Status area, an Open Log File button, and a Start Scan button which has been disabled. The Open Log File button opens the current, actively running *log file* using Notepad. This log file contains the date and time of each activity that takes place within the program which can be used for disclosure or reporting purposes.

The next logical step is to tell the program what user profiles you want scanned. In the following example, we've selected the "Examiner1" user profile. Notice how other elements have suddenly appeared in the interface.



Output Folder

This box requires a valid output folder path where CacheGrab will deposit *copies* of any found cache and history files. You will notice a checkbox beneath it that provides the option to "Delete contents of folder when scan begins".

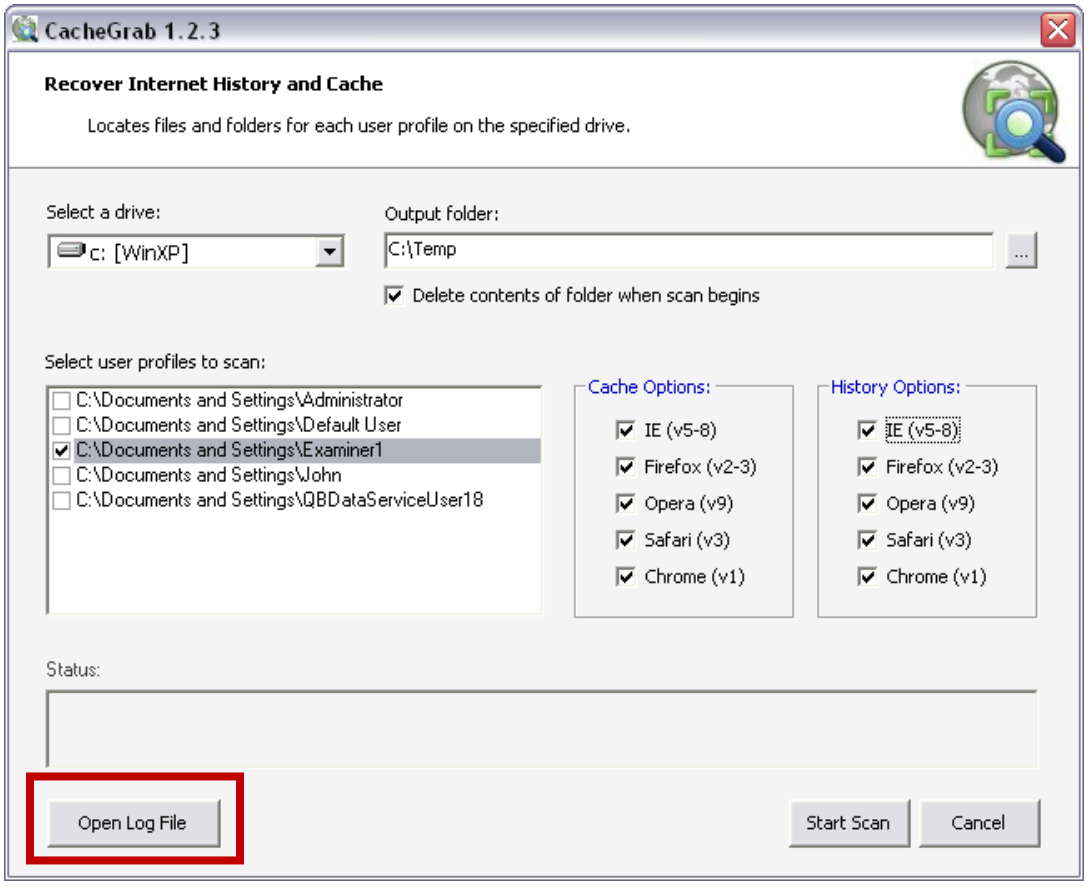
CAUTION: If the "Delete contents of folder when scan begins" is selected, then all files and folders residing inside the specified Output folder will be deleted, once the "Start Scan" button has been selected.

Cache Options and History Options

As easy as placing a checkmark in a box, users can tell CacheGrab what browser artifacts are to be discovered and copied to the output folder. Future releases of CacheGrab may provide additional artifacts such as cookies, bookmarks, and Unallocated Clusters.

Start Scan

Once all the required options have been either inputted or selected, the Start Scan button will become enabled and CacheGrab will then be ready to do its thing.



Open Log File

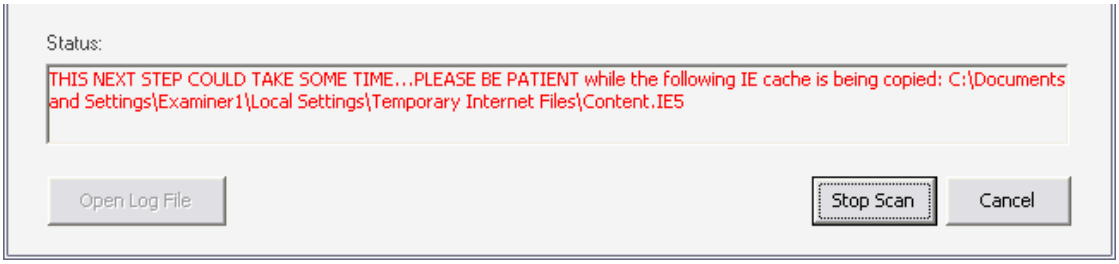
There is a log file that is created for each CacheGrab "Scan". Every action taken by the scan is documented carefully and is itemized chronologically by date and time, to the second. Clicking the "Open Log File" will open up the log file using Windows' Notepad. The contents can then be copied into any forensic report if required.

The following is a sample Status message that users will encounter when Internet Explorer artifacts are being copied out to the Temp folder. This message is to advise users that the process currently underway may take some time to complete.

Do not become impatient and believe that the program has suddenly stopped working. Even if the title bar of CacheGrab changes and displays "[Not Responding]", this is again NOT the case.

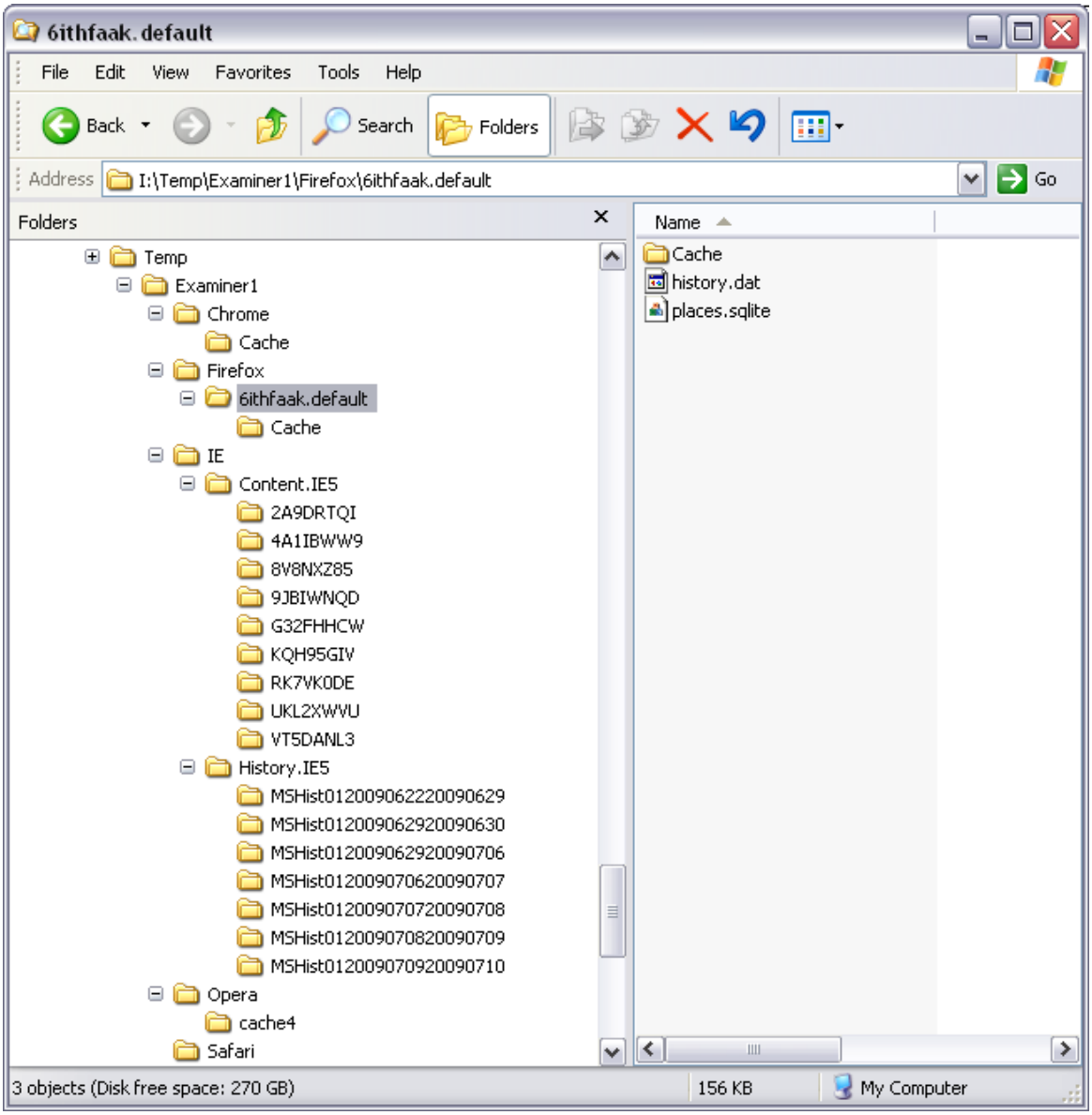
CacheGrab will simply execute a *single processing thread* that will not be able to be interrupted until the "folder copy" routine is complete. Remember, due to Windows permission issues, it is not possible to simply copy the individual files from the cache and history folders.

Therefore, CacheGrab can only "copy the entire folder". As a result, Windows sees this as a single thread or a single task. When the copy process does not complete in a relatively short period of time, which is to be expected, you WILL see the following message.



Outputted Results

Once CacheGrab has completed its Scan, the results can then be easily reviewed using Windows Explorer. The following is a screenshot that demonstrates how CacheGrab organizes the collected files into logical folder names, based on the browser type and the artifact type. The following example shows all of the sub-folders created under a user profile called “Examiner1”.



Using ATTRIB To Unhide Folders

Users also have the option of collecting the different browser files and folders *manually* by locating them on the source hard drive, and COPYing them to a specific output folder.

NOTE: Windows will prevent the access to certain folders (or sub-folders) as is the case when dealing with Internet Explorer's "History" and "Temporary Internet Files" folder contents. A sample MSIE cache can be found at the following path for a user profile called "Examiner1":

`C:\Documents and Settings\Examiner1\Local Settings\Temporary Internet Files\`**`Content.IE5`**

You will notice in particular the sub-folder called "Content.IE5". Normally, this folder is *Hidden* by Windows thereby preventing accessing using tools such as Windows Explorer. Third party software tools will also run into "Permission Denied" errors. There is however a simple workaround.

First, let's assume that you have successfully COPied out the folder **Temporary Internet Files** to the temporary folder: **`C:\Temp`**.

USING THE DOS "ATTRIB" COMMAND

What we now need to do is use the DOS command tool **ATTRIB** to remove the "Hidden" attribute from the folder. Here's how we do this:

1. First, we need to open up a fresh DOS Command Prompt. Start by going to the Start Menu in windows, goto the Run command line option, and type in **cmd** then press OK. A new command window will appear.

2. At the DOS prompt, type in "**cd**" (without the quotes). This will return you to the root of the drive C:.

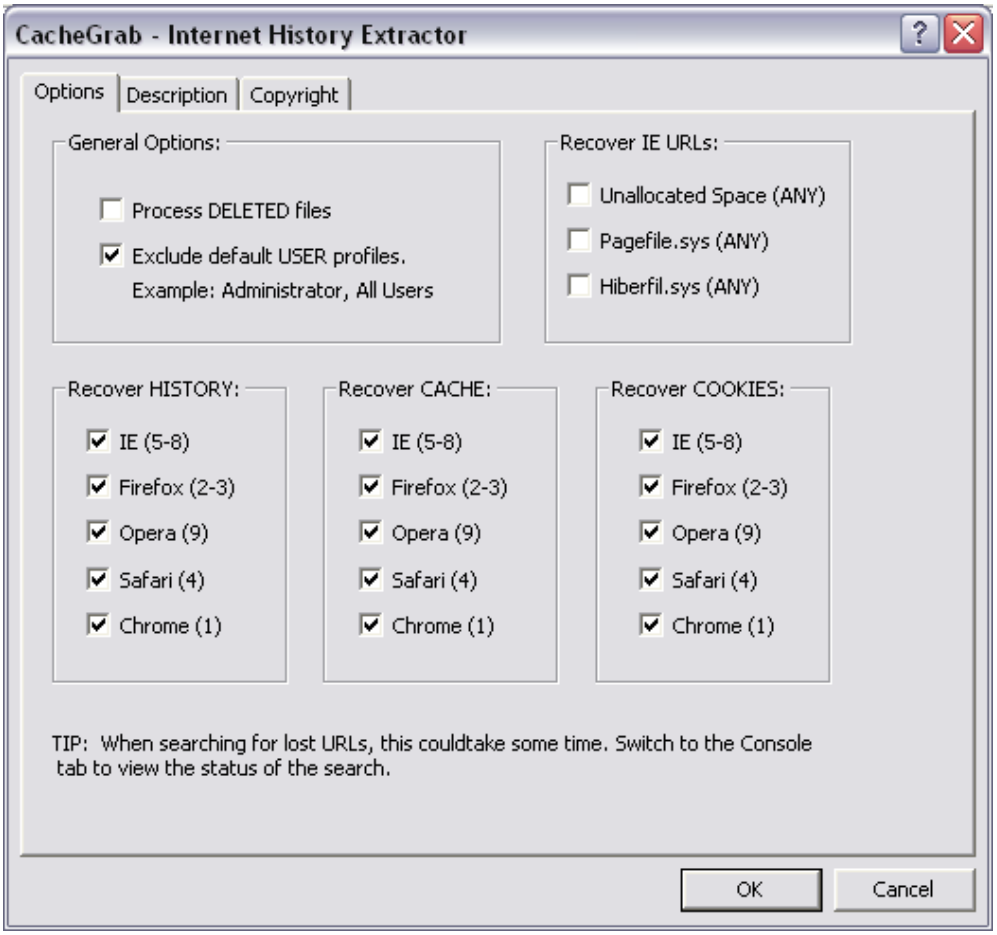
3. Now change directories to the temp folder using: **cd C:\Temp** and press Enter.

4. Now, type in **ATTRIB –a –s –h C:\Temp /S /D**. This will conduct a recursive search through the Temp folder and sub-folders and will "remove" the **a**rchive, **s**ystem and **h**idden attributes. This will now make the Temporary Internet Files and all of its present sub-folders accessible.

Using EnScripts

Many users of CacheBack are computer forensic examiners who also use EnCase by Guidance Software Inc. as their primary analysis tool. As such, CacheBack comes with a CacheGrab EnScript that supports both Version 5 and Version 6 of EnCase. The EnScript comes packaged with the CacheBack software and is also available as a separate download from the CacheBack website (www.cacheback.ca).

Using the EnScript is really quite easy. The following screenshot demonstrates some of the intuitive options that makes using the EnScript relatively straight forward. The results of the EnScript are similar to the way CacheGrab (the Windows version) collects browser artifacts. The only difference is that the EnScript can also search Unallocated Space for lost / deleted Internet Explorer URLs. In this case, the found URLs are assembled into a series of proprietary binary data (*.CBD) files which can then be imported directly by CacheBack.



NOTE: The EnScripts were updated in July 2009 so that examiners, who are processing multiple hard drive images in a single case, can search "Unallocated Clusters, Pagefile.sys and/or Hiberfil.sys" that have been

randomly selected. This feature enhancement is controlled by simply "checkmarking" those files, from select drives, that examiners are wanting searched.

*DO NOT confuse this new enhancement with the section on the form entitled "*Recover IE URLs*" option which is an exclusive "search all (or any)" feature. This setting will ignore the fact that some Unallocated Clusters (etc) may not have been selected (checkmarked) and proceed to search them anyway.*

Support for HistEx

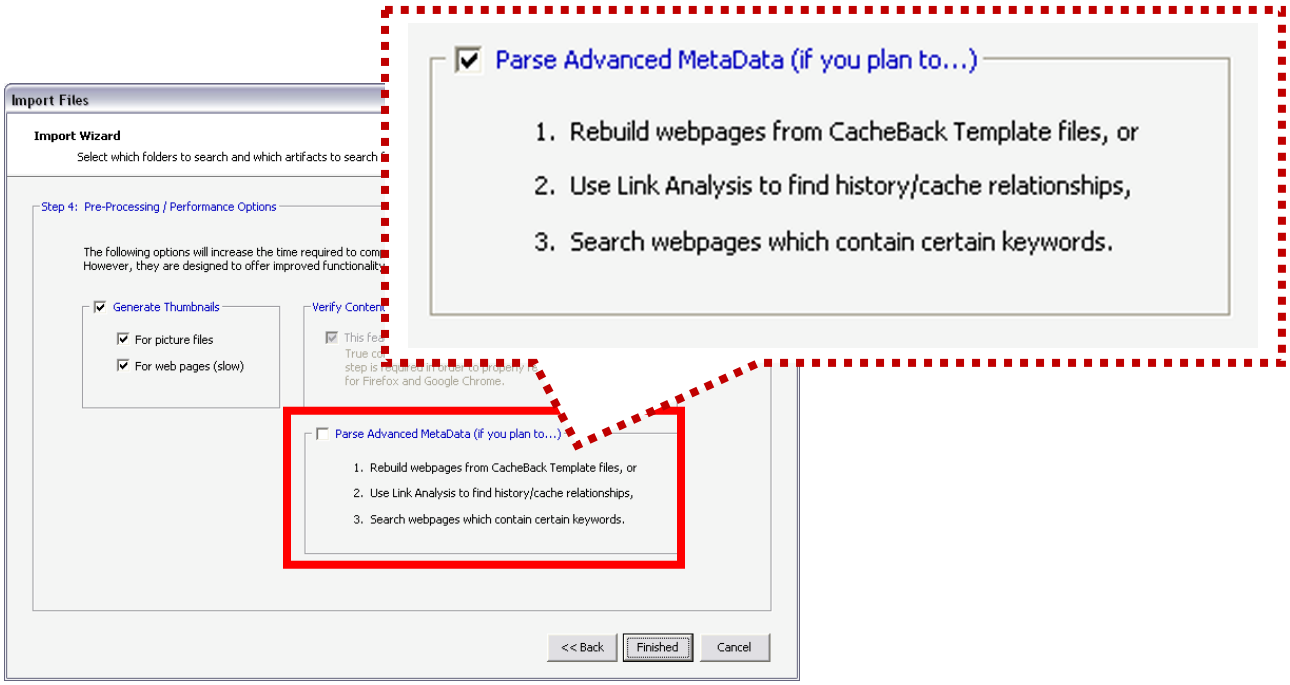
HistEx is a product developed by Craig Wilson of Digital-Detective.co.uk and is a tool that has become a favorite IE URL mining tool when dealing with Unallocated Space. In order to support the many users of HistEx, CacheBack can now import HistEx results (for IE) directly without any difficulties.

NOTE: This is just one step to supporting the discovery of artifacts in Unallocated Space. At the time of the writing of this manual, CacheGrab Version 2 is underway which will specifically address Unallocated Space. It will also harvest browser and cache files from allocated space as it presently does, however, there will be a MAJOR increase in speed through code optimization.

CacheGrab Version 2 is expected to be released by September 2009.

Pre-Processing: Parse Advanced Metadata (for Link Analysis)

The Import Wizard is a step by step guide to importing cache and history data into a new CacheBack project file. The last step (Step 4) of this wizard deserves a bit of attention because it has an impact on the available of using one of the advanced features called "**Link Analysis**".



Link Analysis is the identification of relationships (matches) between URL records from History files / databases for the different browsers AND the hyperlinks (anchor tags) found inside the HTML source code for web pages that originate from Cache files. The purpose of this feature is to help investigators quickly ascertain what hyperlinks on a given web page *may have been selected* (eg: clicked on) by a user. Most often than not, this *association* of URLs can be helpful in telling a story.

Link Analysis is an easy way for CacheBack users to zero-in on specific websites where subjects may have spent time looking or accessing different links on one or more pages. Using the Links column in the Table and sorting in DESCending order, the higher links count can be the better indicator of what has been happening on a computer.

Please note that this option is ONLY available at this stage of the program. Failing to use this feature at this point simply means that Link Analysis will not be available using the imported data.

Here's How It Works

Step 1. Once the Link Analysis option on Step 4 of the Import Wizard has been selected (see above screenshot) and CacheBack begins importing cache and history files (records), each web page encountered will automatically have its *contents* (source code inside the Body tag) copied into the database.

Step 2. Once the importing process has completed AND the parsed URL records have been loaded into the Table, you must choose the Link Analysis  option from the Toolbar OR use the shortcut option form the context (popup) menu (which is accessed in the Table by right-mouse-clicking).

Step 3. As the Link Analysis routine is running, CacheBack will make matches between history and cache URLs and flag them accordingly.

Step 4. Once the Link Analysis process has completed, it's now simply a matter of sorting the Table on the Links column, in DESCending order. The quantity of links found in any given web page (which there can be many) or any history URL (which there will always only be one), will be displayed in the Links column. The higher number of Links for any given entry, suggests more potential activity at one given web page (or website).

Step 5. By clicking on a Table row that contains a Links value, and by switching to the Links Tab in the Viewer Pane, you will now see the associated History URLs. The opposite is true as well. Selecting a Table row that contains 1 Links value (eg: a History URL) will reveal the associated web page URL(s) in the Links tab.

Figure 1 – Link Analysis Underway

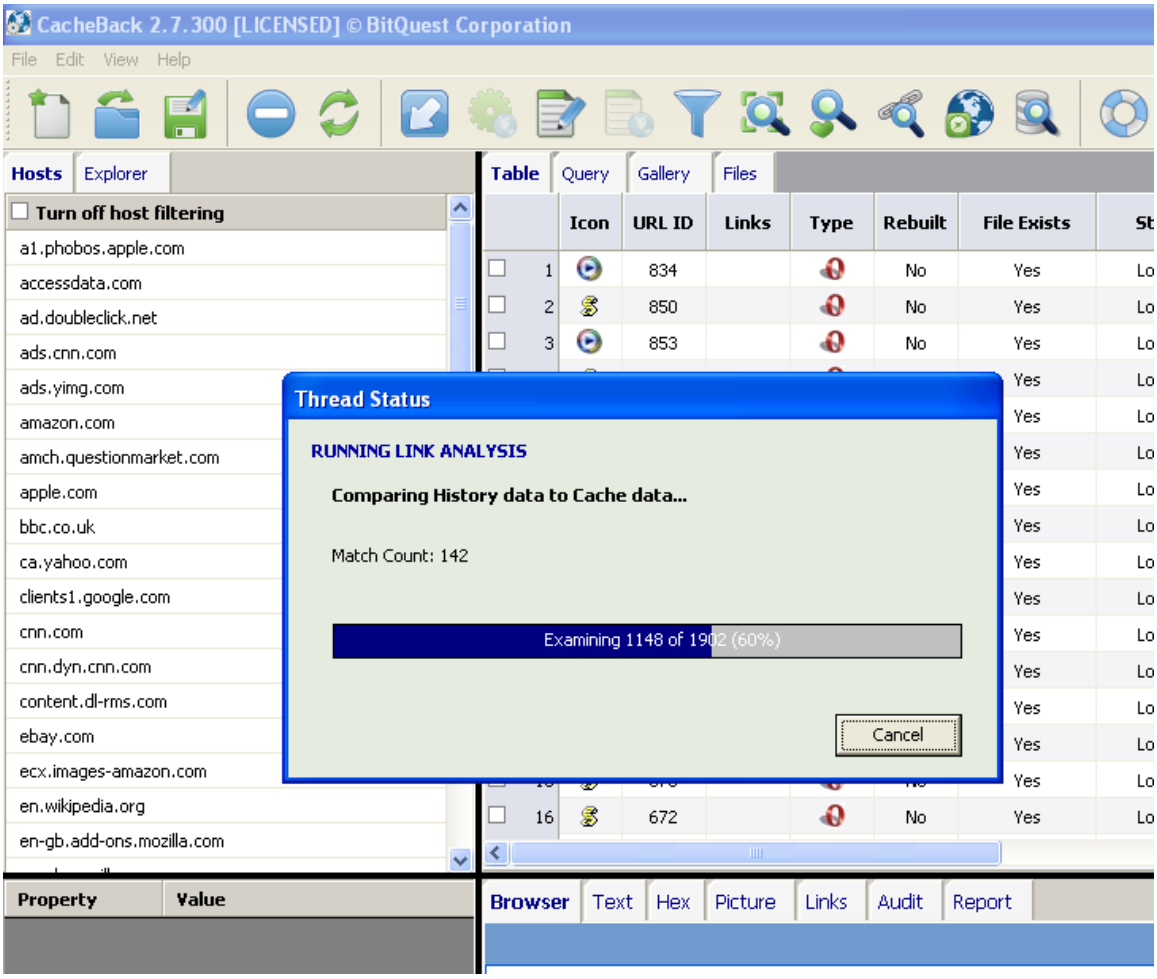


Figure 2 – Link Analysis Completed

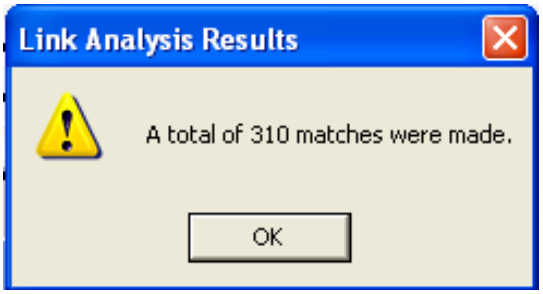


Figure 3 – We need to refresh the recordset first!

Once Link Analysis is complete, you Refresh the Table contents by clicking on the toolbar Refresh () button.

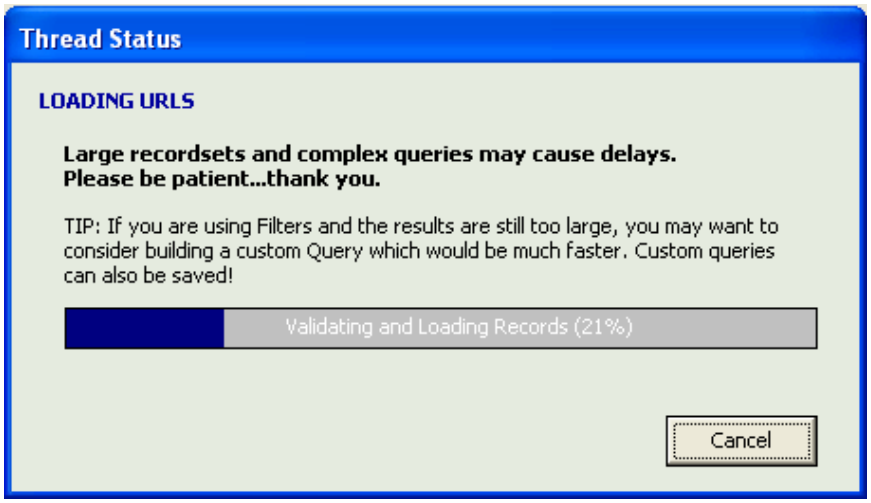


Figure 4 – Once the Table has been refreshed, we sort on the Links column.

In this example, we have sorted the Table in descending order. We can see that the first few records contain as many as 10 identified matches / relationships between History URLs and the contents of certain cache web pages.

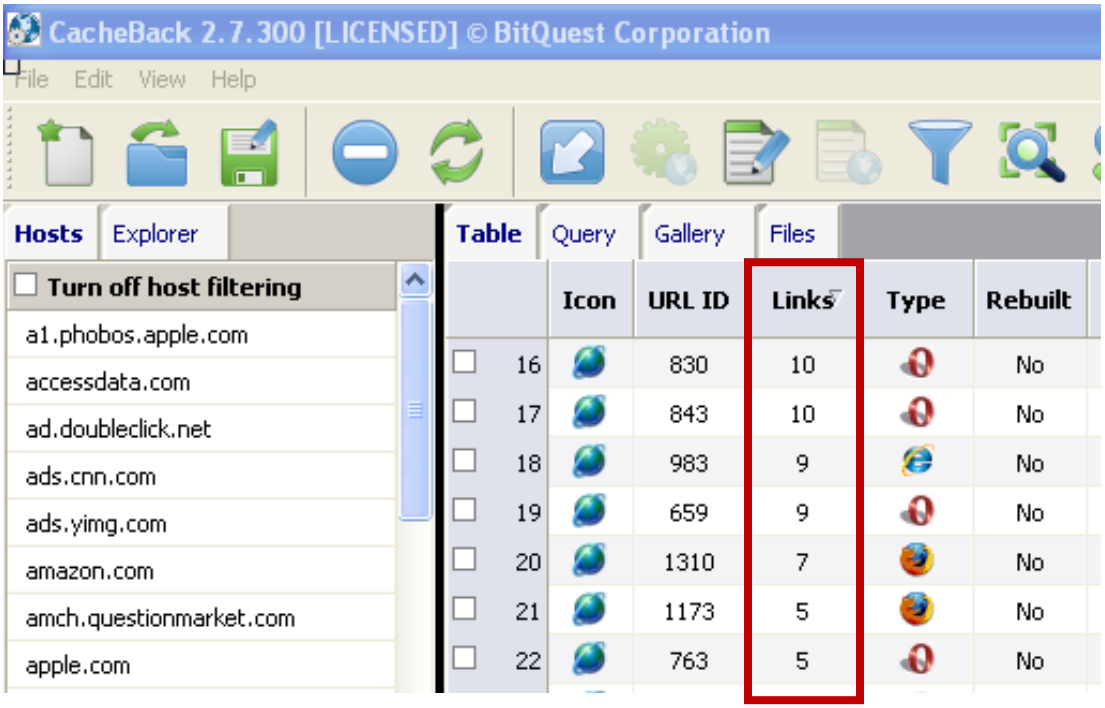


Figure 5 – Displaying the related URLs on the Links viewer tab.

By selecting a row from the Table (in this case, a web page URL), we can see all of the "History" related URLs that match the same URL as that of the web page.

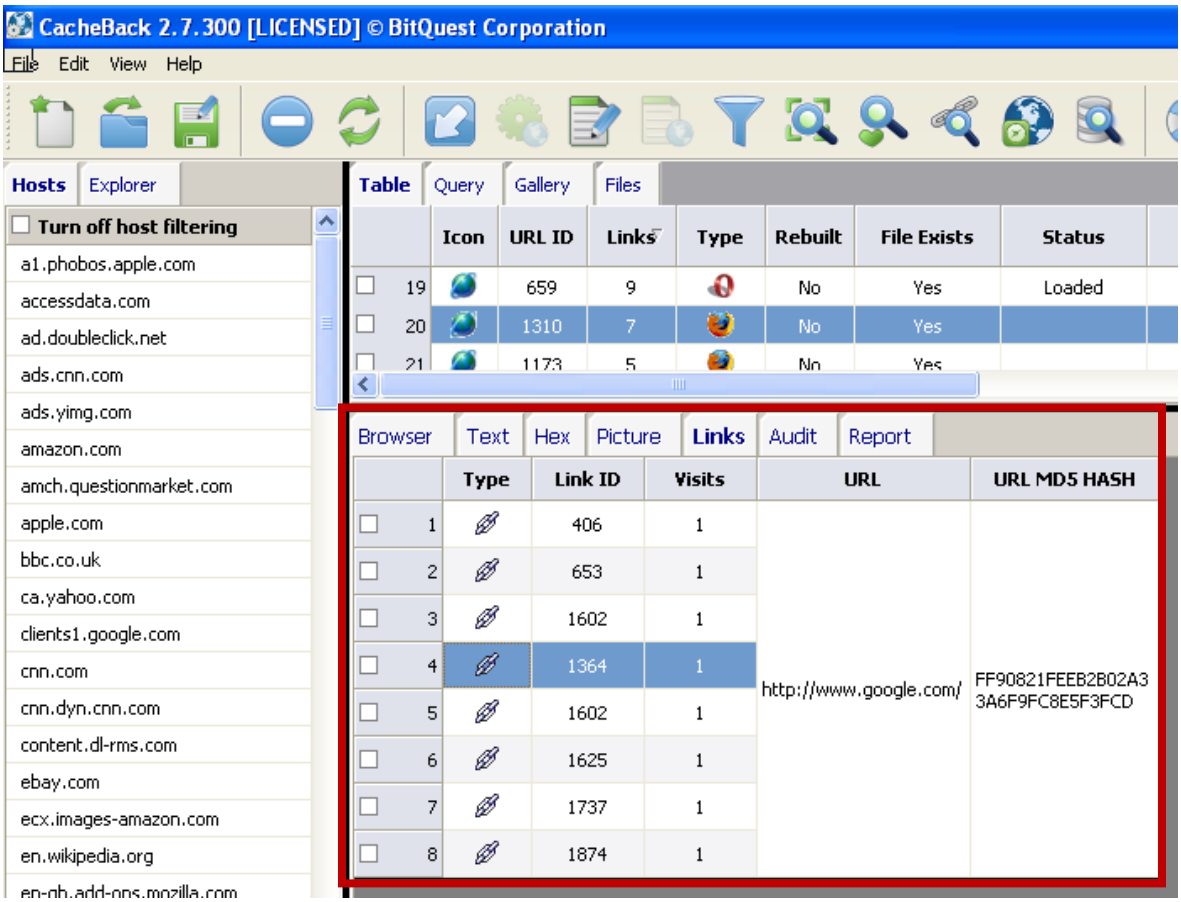


Figure 6 – Let's see what Link ID #1364 can tell us…

By right-mouse-clicking overtop of the selected row in the Links viewer table, a context menu appears with the option to "Goto URL…". Choosing this option will JUMP to the associated URL record entry in the Table above.

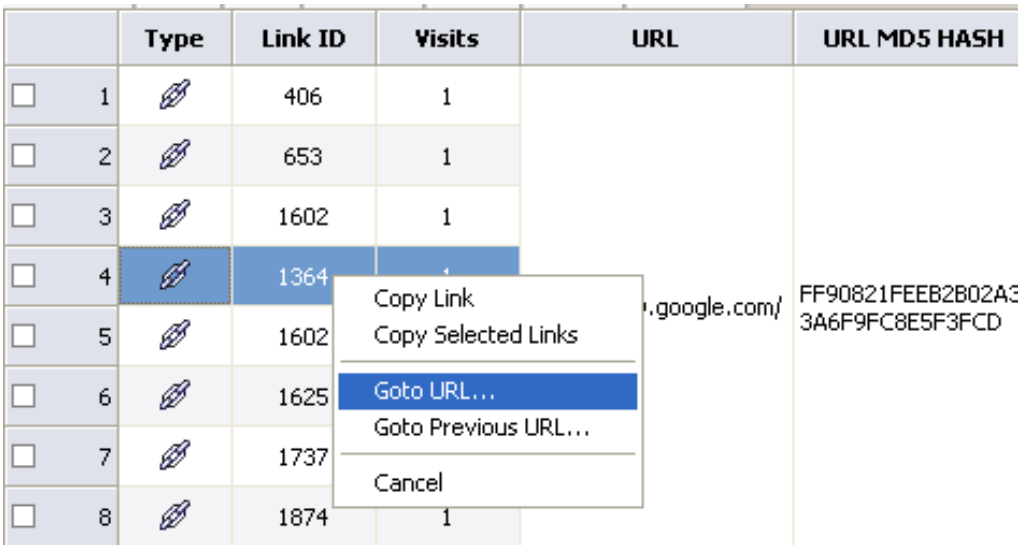


Figure 7 – Notice how the matching URL becomes the topmost record in the Table.

Because the URL is from a History, the Text viewer tab becomes the default tab selected.

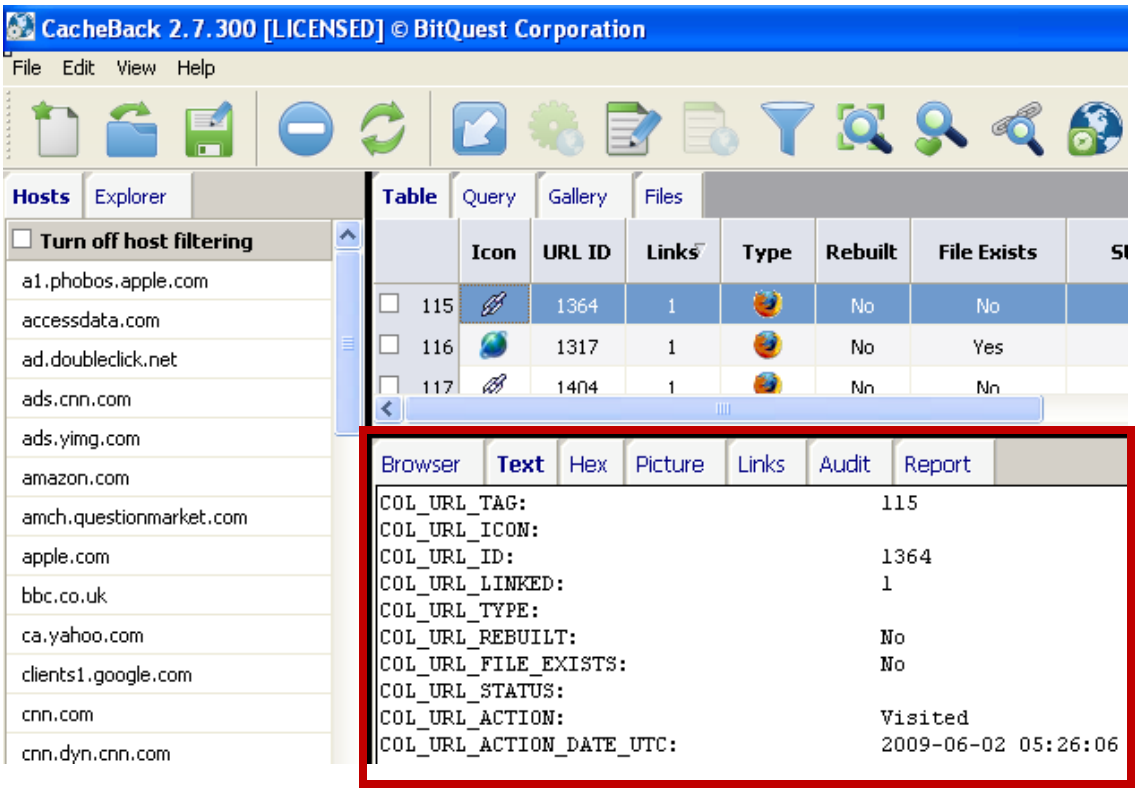


Figure 8 – The Properties Pane displays the URL data as well .

Notice how much easier it is to read the same metadata offered by the Text viewer.

Property	Value
Action	Visited
ActionDateLocal	2009-06-02 01:26:06 [-0400 DST]
ActionDateUTC	2009-06-02 05:26:06 [UTC]
CacheFileCreated	2009-06-05 16:31:19
CacheFileExists	False
CacheFileExt	
CacheFileLastAcces	
CacheFileName	
CacheFileNameDecc	
CacheFileRebuilt	False
CacheFileRebuiltBLC	
CacheFileRelativePa	
CacheFileSize	0
CacheFolderName	
CacheRootPath	
CacheType	Firefox3 History
Category	
DataBlockFile	
DataBlockFileOffset	0
ErrorCode	
ExpiryDate	
GZipEncoded	False
HashedURL	FF90821FEEB2B02A33A6F9FC8E5F3FCD
Hidden	False
HistoryFilePath	K:_SAMPLE_EVIDENCE\Official Sample Data\Firefox3\places.sqlite
HistoryVersion	3.0
Host	google.com
HTMLBody	
IsBookmark	False
IsCookie	False
IsHistory	True
IsJavascript	False
IsPicture	False
IsWebPage	False
LastVisited	2009-06-02 05:26:06
Links	
MapFileOffset	0

Working with the Data

Introduction

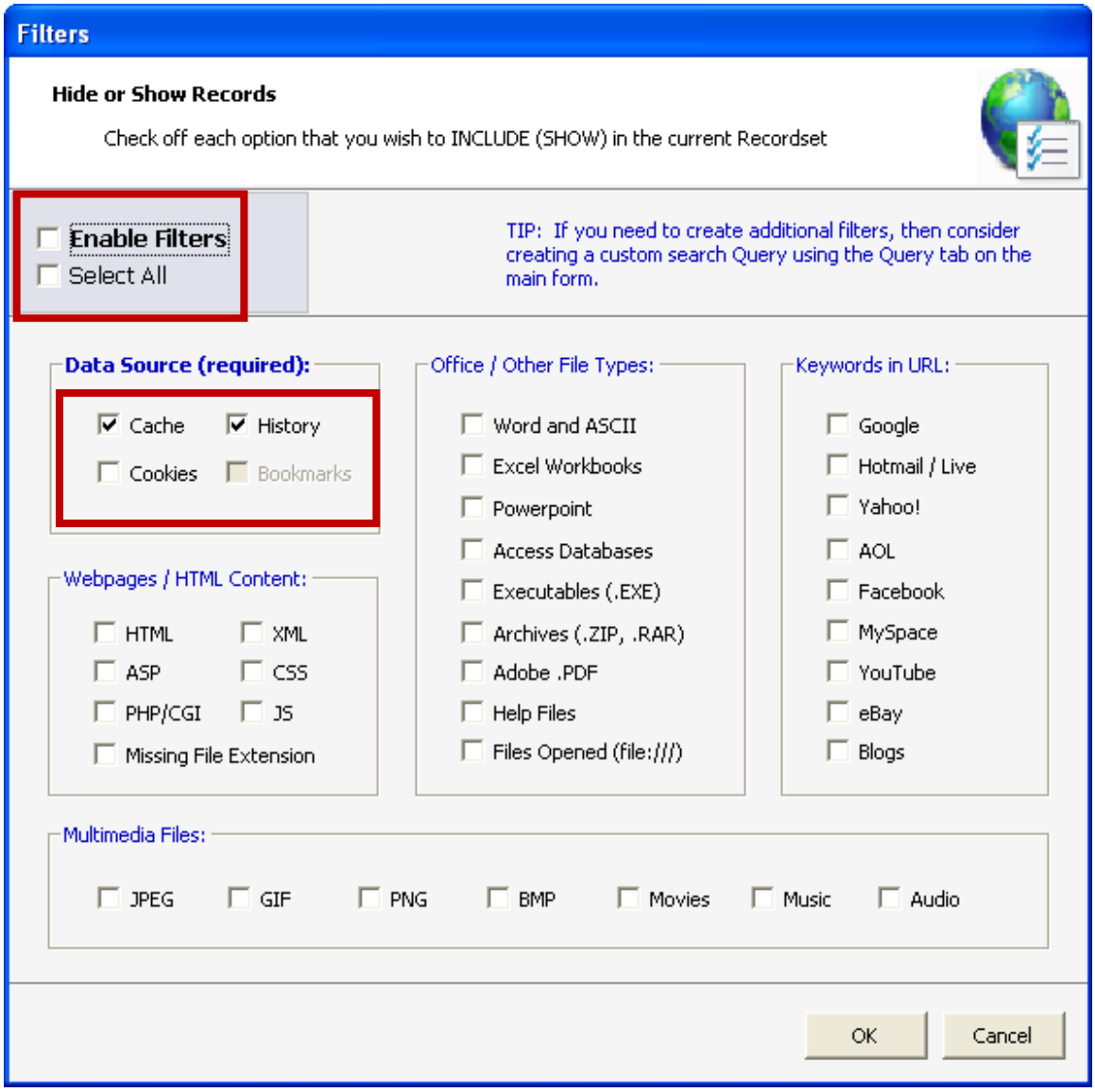
CacheBack has the ability to process hundreds of thousands of URL records at a time. These records can originate from History and Cache data sources, and can relate to a variety of file types and content. Having mechanisms to weed through these large datasets is essential, and CacheBack delivers this requirement through the use of *Filters, Queries and Host Filtering* options.

Managing Records Using Filters

CacheBack provides filters to "reveal" (not hide) URLs that "match" the selected filter options. What is also very important to understand is that filters act as an *overlay* to the query that defines the complete contents of the Table (or Gallery). Filters can be thought of as a means to *fine-tune* what URL records are *visible* and which *are not*.

CAUTION: Filters, while helpful, may yield unexpected results if not used correctly. Certain records can "slip through" because the filtering terms are either too narrow, or too wide. Symptoms may include limited or too many records being displayed in the Table, or being reported. If filters cannot yield the correct results, you may want to consider using queries.

In order to use Filters, simply select the Filters ()button from the toolbar to load the Filters Window. There are 2 *mandatory* options that need to be selected. "Enable Filters" turns this feature ON or OFF, and the "Data Source" indicates which types of URL records to apply any filters to.



Using filters is as simple as checking off those items that you want to *appear* in the Table.

URLs that do not match the selected filter criteria will be hidden from view. If, after applying filters, you are still working with large quantities of URLs, then you should consider using a custom query instead of a filter.

How To Use Filters

Before you can do anything with Filters, you must first "enable" filters using the Enable Filters option. This checkbox is used to turn filters ON or OFF at any time during the use of the program, regardless of whether or not different filters have already been selected.

Data Source (required)

Filters can be applied to up to four (4) different *sources* of URLs. By default, filters will be applied to URLs that originated from Cache and History files. The third option is from Cookies and the fourth option is from Bookmarks (to be implemented).

Webpages and HTML Content

These filters look at URLs in two different ways.

First, if the URL is from a History file, then the filter checks the URL's file extension for a known *web document* file extension. If the URL is from a Cache file, then the filter checks both the URL file extension, and the actual contents of the cache file (if present). The physical inspection of any file is simply to look for the **<HTML** or the **<BODY** tags.

The most common file extensions associated to HTML content include, but are not limited to, the following: .HTM, .HTML, .ASP, .JS, .XML, .PHP and .CGI. In some cases, a cached web page may contain NO file extension. For this reason, we have included a checkbox for "Missing File Extension".

Office / Other File Types

For this option, CacheBack will filter URLs strictly by certain keywords or file extensions that must be present in the URL (eg: .DOC, .PPT, file:///).

Keywords in URLs

As the above caption indicates, this filter requires that certain keywords be present in the URL (eg: hotmail.com, facebook.com).

Multimedia Files

These filters look at URLs in two different ways.

First, if the URL is from a History file, then the filter checks the URL's file extension for a known *picture or movie* file extension. If the URL is from a Cache file, then the filter checks both the URL file extension, and the actual contents of the cache file (if present).

The most common file extensions associated to multimedia files include, but are not limited to, the following: .JPG, .BMP, .GIF, .MPG, .MOV, .AVI and .WMV.

Managing Records Using Queries

Introduction

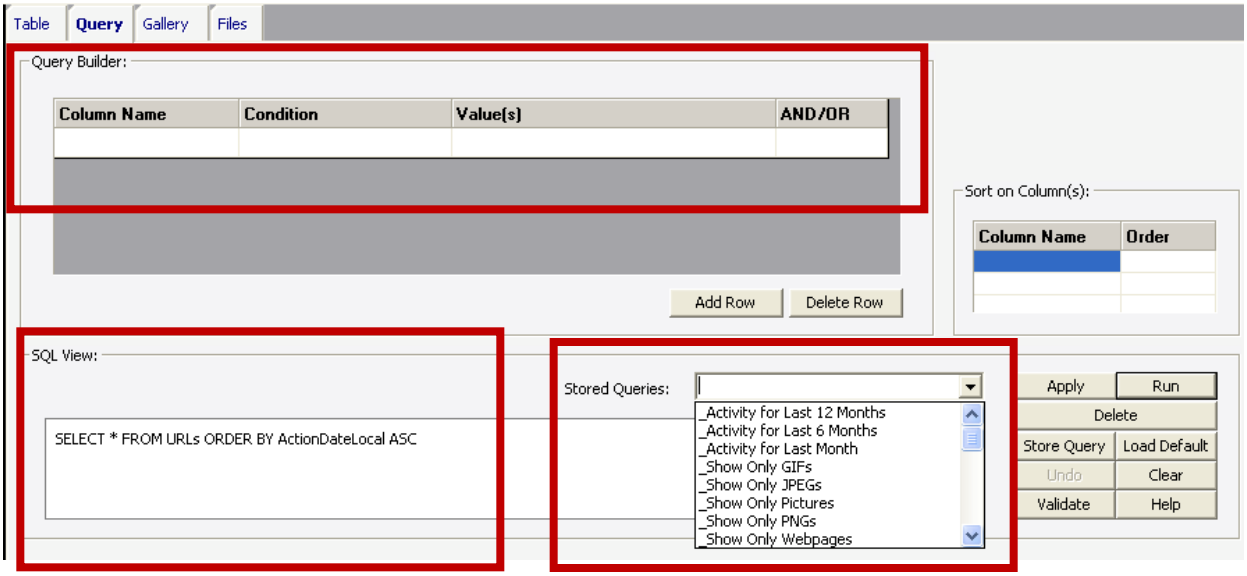
CacheBack 2.7 has completely re-worked the way it handles data and now provides greatly improved performance through the use of *stored queries*.

Each new project file now comes pre-loaded with a number of investigation-type-specific queries. Queries are accessed in one of three different locations: the Query Tab, the Quick Queries () button from the toolbar, and the SQL Query Builder Window () which is also available from the toolbar.

PLEASE NOTE: While the term "query builder" is used to refer to two types or two different locations of query builders, it should be noted that the builder on the Query Tab is the recommended tool for building queries. The SQL Query Builder Window is provided for backwards compatibility and may be deprecated in a future dot release of the software.

Figure 1 – The Query Tab found inside the Data Pane.

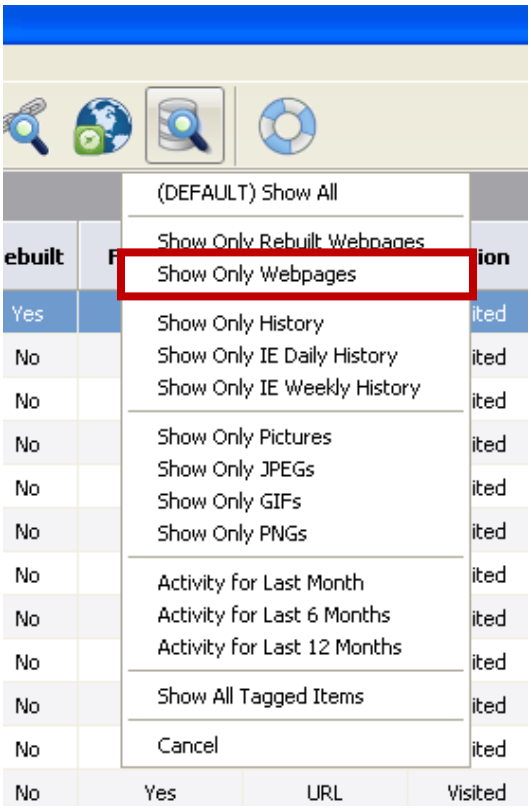
The Query Tab features three areas to help define queries: *the Query Builder, Stored Queries and the SQL View*.



The "_" (underscore) that prefixes any query name in the Stored Queries box indicates that it is a RESERVED query and should not be deleted or renamed. Doing so may prevent these queries from being called by other parts of the program (eg: the Quick Queries menu).

Figure 2 – The Quick Queries menu.

The Quick Queries context (popup) menu provides the fastest (and recommended) means of running pre-defined queries. This particular list contains queries that are most commonly used at the onset of an examination.

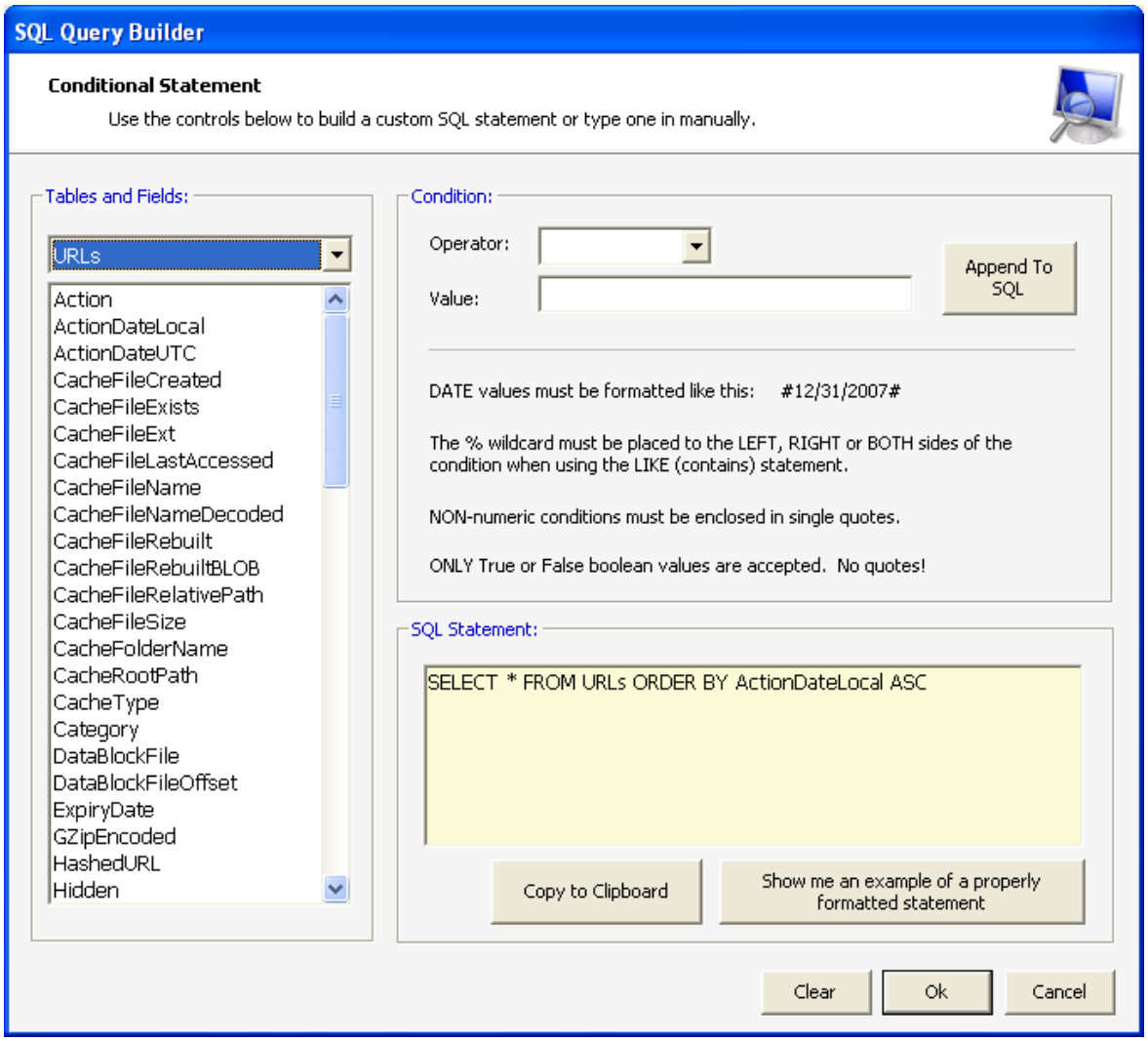


In most cases, at the start of any new project file AND after importing any cache and history files, users wanting to rebuild web pages will want to run the "**Show Only Webpages**" query first! This will not only reduce the number of displayed records in the Table, but it will identify all URLs that are web pages (or contain HTML tags consistent with web page content).

Figure 3 – The SQL Query Builder Window.

Users define the "SQL Statement" by using a combination of (1) Tables and Fields, (2) Conditions, and (3) Values which then get appended to the box below. All queries MUST begin with "SELECT * FROM URLs" which is the pseudocoded meaning for "Return all fields".

When a new condition is appended to the SQL Statement, you will see the keyword "WHERE" appear followed by one (1) or more conditions.



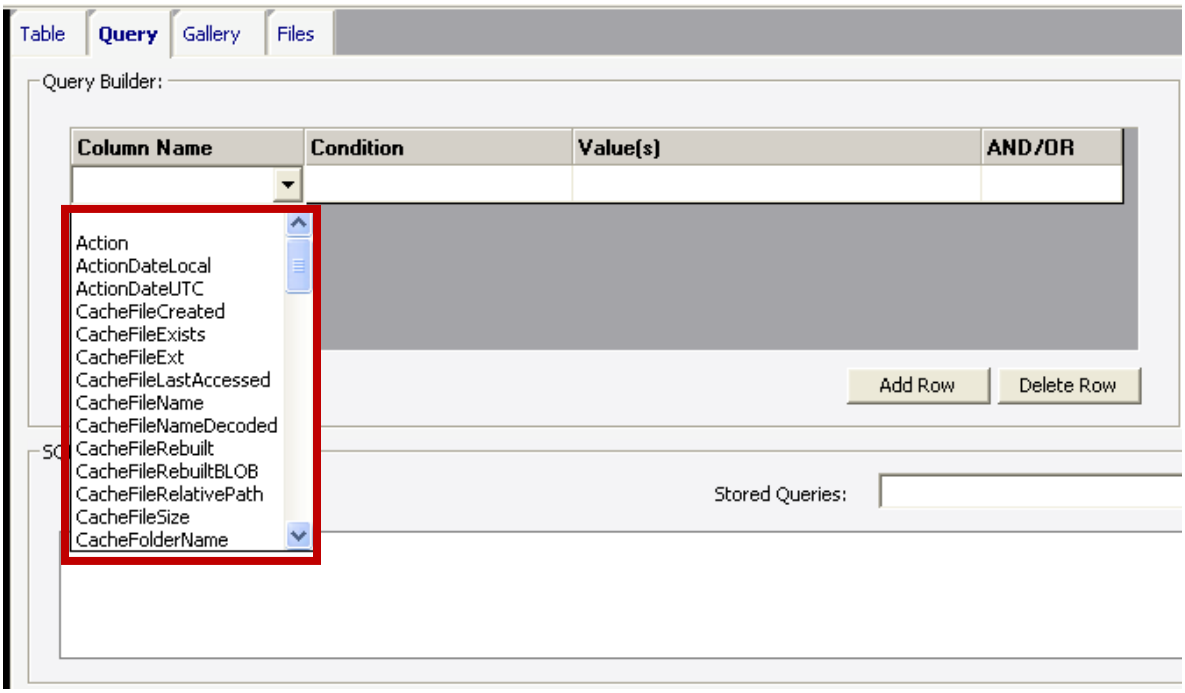
This option was provided in CacheBack 2.7 as a form of convenience for those users wishing to throw together a quick query. However, we recommend that users familiarize themselves with the Query Tab option as the preferred means of creating and saving queries.

Building Queries with the Query Builder

Building complex and simple custom queries is a pretty easy task in CacheBack 2.7!

Using the new Query Builder tool on the Query Tab forces users to think one step (or one condition) at a time. Adding conditions to the query is what a query is really all about. The Query Builder therefore provides two simple functions to make this possible. The first is by “Adding Rows” and the second is by “Removing Rows”.

To begin a new query, simple start with the first cell of the first row. When we click inside the cell, a list of available choices appear as a dropdown list. This list contains all available columns (aka: field names) in the project file (database).



You will notice that the list is sorted alphabetically. There are a total of 74 columns to choose from which enables the creation of some very comprehensive and specific query statements.

The following diagrams will walk through the creation of a sample query that is designed to identify all cached Hotmail or Yahoo mail web pages that were accessed within the last month.

Figure 1 – Condition #1: Hotmail

In this first condition, we selected the column "URL" and jumped ahead to enter the Value "hotmail". We then clicked inside the Condition cell and a dropdown list appears as noted below.

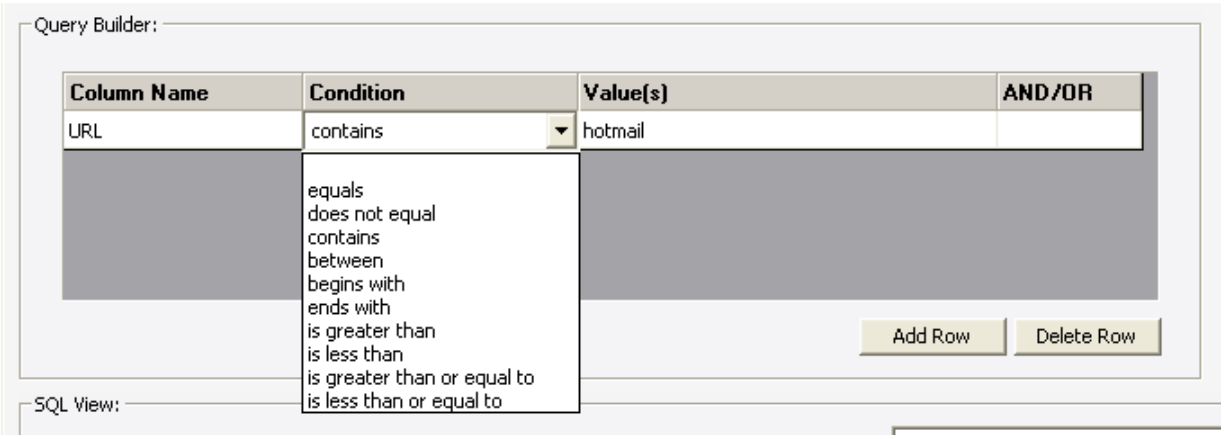
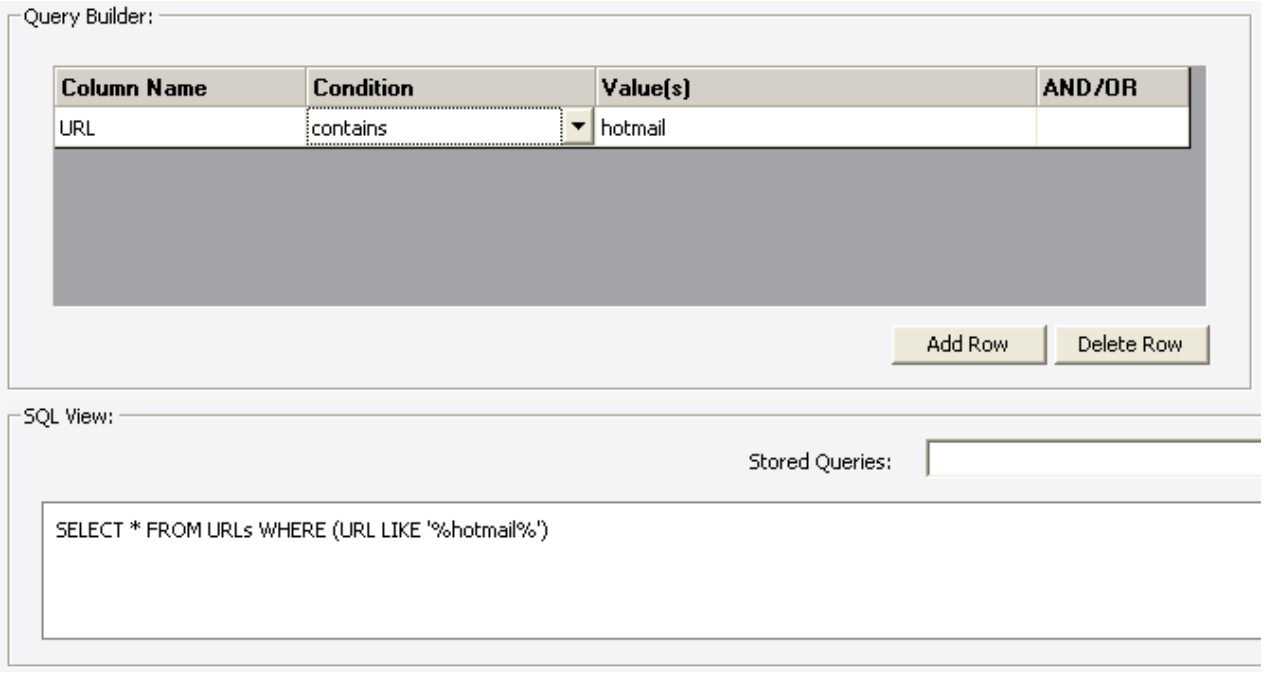


Figure 2 – Completing the first condition.

After choosing the condition of “contains”, our first row is completed and the Query Builder automatically updates the SQL View below.



Notice how the condition “hotmail” is surrounded by single quotes and a percent sign (wildcard: zero or more). This tells us that the term “hotmail” can appear anywhere in the URL.

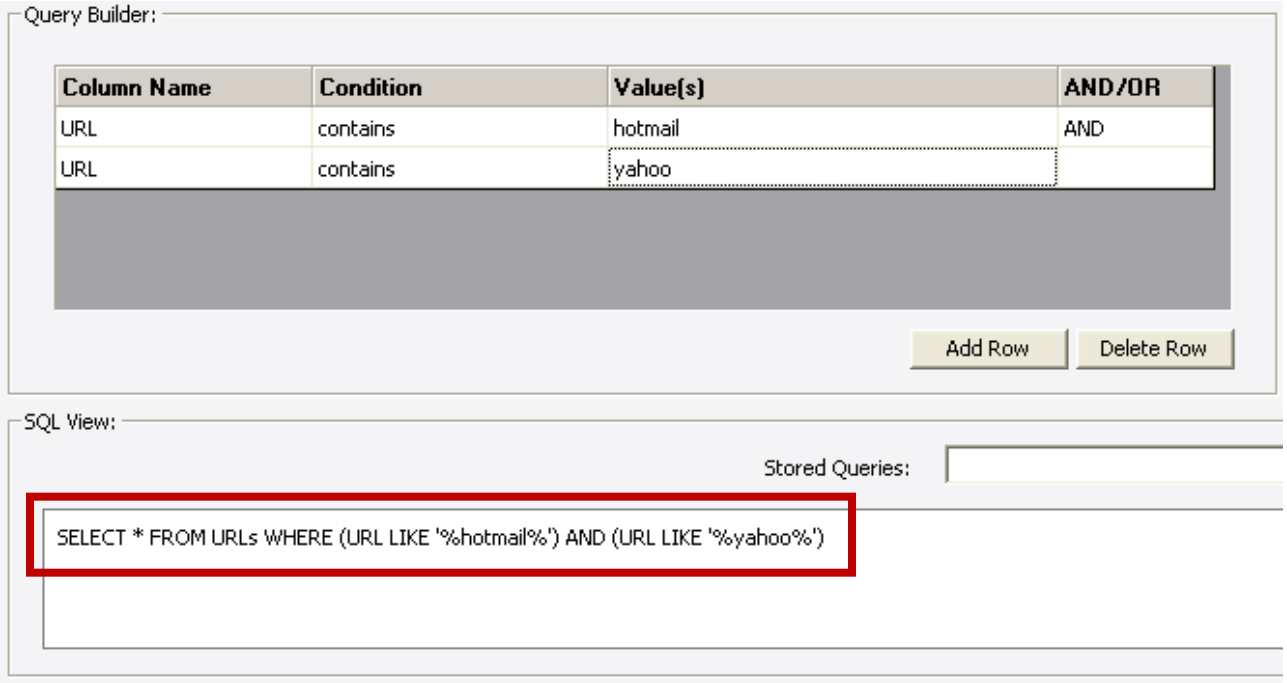
Our next step is to add the yahoo condition…

Figure 3 – Adding the condition for Yahoo mail.

Because we want to add a 2nd condition to our query, we need to use the "Add Row" button. When we do this, a couple of things occur. First, you will notice that the last column of the Query Builder is automatically assigned the continuation operator "AND". Second, a new row will appear.

So for this next step, we want to select the same Column and similar condition. This time however we want to use the value "yahoo".

Here's what our 2nd condition should now look like.



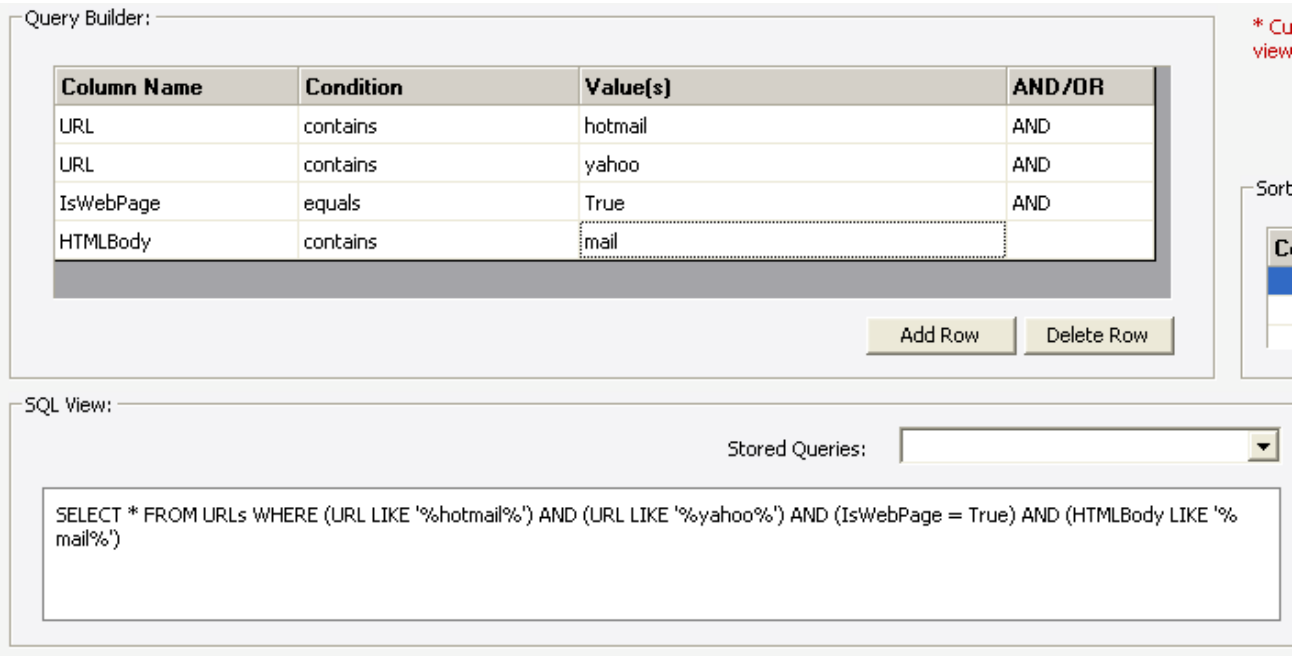
And, just like before, we can see that our changes have been automatically formatted into a proper SQL statement.

NOTE: Parentheses are added for clarity purposes and are processed as entire logical units. This is "key" to defining complex queries. Much in the same way that we use parentheses to group mathematical conditions into orders of precedence, SQL syntax treats these logical units as orders of precedence.

We now need to add one or more conditions that check to make sure that we find “web pages” instead of History records AND that will likely reveal those web pages that contain some reference to mail.

Figure 4 – Two new conditions to check for web pages and mail content.

The following diagrams shows two extra conditions added, using the same steps discussed above. Notice again how the SQL View udpates itself to a properly formatted definition.



Notice this time that we have added the column “IsWebPage” which stores a Boolean data type of either True or False. In this case, we are testing for the value of True. Our 4th condition verifies that the word “mail” appears somewhere in the “HTMLBody” of the page.

NOTE: If we want to be really sure that our query returns only those records where the cache file still exists somewhere (eg: in order for us to rebuilt it later on), then we should add a 5th condition:

Column Name	Condition	Value(s)	AND/OR
URL	contains	hotmail	AND
URL	contains	yahoo	AND
IsWebPage	equals	True	AND
HTMLBody	contains	mail	AND
CacheFileExists	equals	True	

The SQL View will now reformat our query as follows:

```
SELECT * FROM URLs WHERE (URL LIKE '%hotmail%') AND (URL LIKE '%yahoo%') AND (IsWebPage = True) AND (HTMLBody LIKE '%
mail%') AND (CacheFileExists = True)
```

TIP: SQL statements can contain new linefeed characters (carriage returns) to make it easier to read (display) queries. The Query Build does not inherently do this and it is not required. However, if we were typing this query by hand (which you can do by the way!), it could look a lot better if it was formatted like this:

```
SELECT * FROM URLs
WHERE (URL LIKE '%hotmail%') AND (URL LIKE '%yahoo%')
AND (IsWebPage = True)
AND (HTMLBody LIKE '%mail%') AND (CacheFileExists = True)
```

To finish off our example, we now need to add a condition that tests for time of activity (eg: within the last month).

When we use dates and times as conditional statements within a query, we need to surround the date/time values within “#” symbols. This tells the application that the value between these symbols should be converted and validated as “a date”.

FORMATTING DATE CONDITIONS

The following is a sample of a condition that tests for a known date:

```
ActionDateUTC = #12/31/2008#
```

The following is a sample condition that tests for a range of dates:

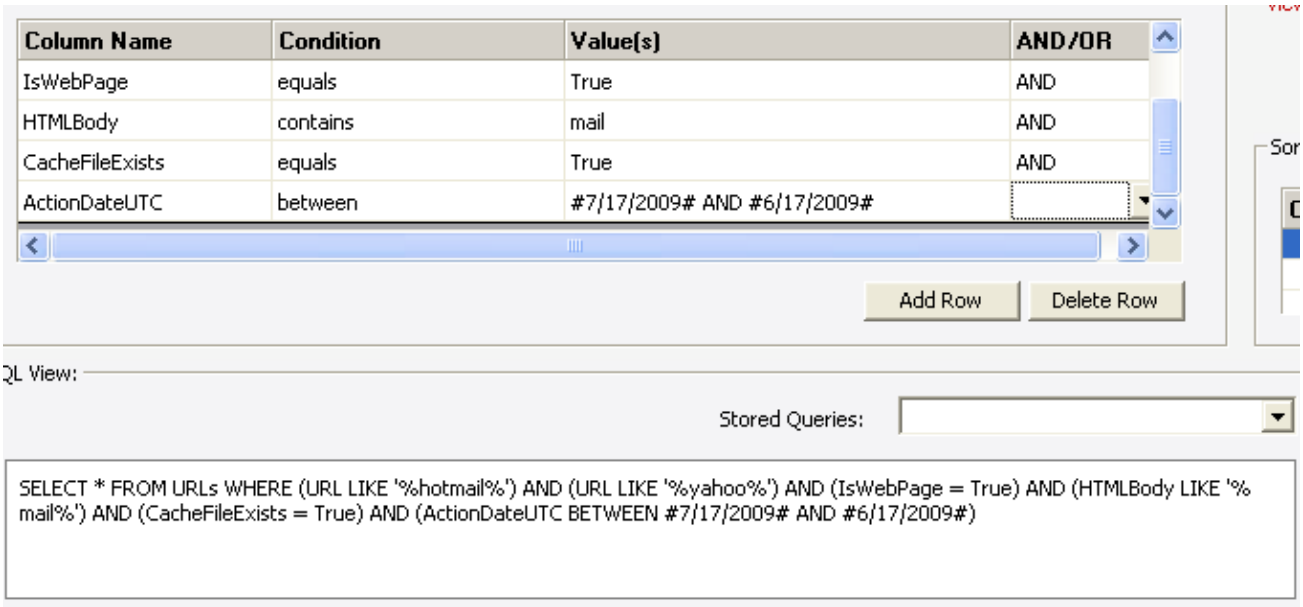
```
ActionDateUTC BETWEEN #12/31/2008# AND #1/30/2009#
```

NOTE: The keyword "between" is a much better way of testing for date ranges. A less common, but equally valid, way of testing for this same date range would be as follows:

```
ActionDateUTC > #12/31/2008" AND ActionDateUTC < #1/30/2009#
```

NOTE: If we wanted to include the start and end dates within our search results, then the expressions above would need to be changed to #12/30/2008# and #2/1/2009# respectively.

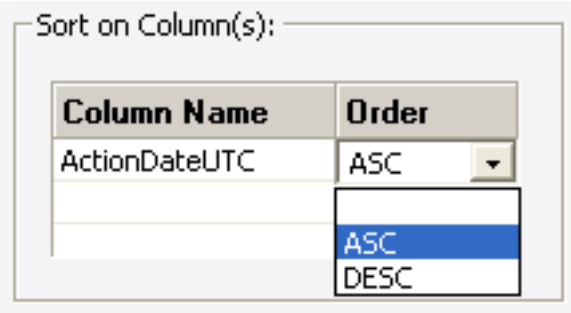
Figure 5 – We now add the condition to test for entries from the last month.



The last thing we need to do is SORT the results to make it easier for us to work with. This is where we define the "ORDER BY" statement using the "Sort on Columns" feature in the top right corner of the Query Tab. Different from the way we used the Query Builder, the Sort On Columns table has only 3 fixed rows to work with. This is because it is rare to sort on more than three columns. This table uses two dropdown lists: one of the Column Name, and one for the Order. There are only two values available for the Order column: ASC (for ascending) or DESC (for descending).

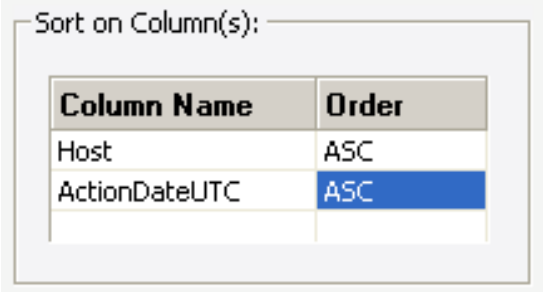
Figure 6 – Sorting our results on one (1) column.

In this example, we would like to sort our results strictly on the last time an action took place in relation to the record. Usually this will be the last time the URL was last visited. In 2.7, CacheBack has replaced the column name "Last Visited" with two columns: "ActionDateUTC" and "ActionDateMeaning". It suffices for this example to simply indicate that we want to sort on the ActionDateUTC column.



Sometimes, however, it might make more sense, or be of more use, to the user to sort (or group) the results based on the website domain (eg: google.com, hotmail.com, yahoo.com). For this reason, we may want to add a second column and change the sort order to more accurately organize our data.

Figure 7 – Sorting our results on two (2) columns.



NOTE: Logically speaking, it would make no sense to sort on the ActionDateUTC column first, and then by the Host (aka: domain name). This should be obvious in the fact that it is entirely possible to jump to and from different websites within one or more browsers, and/or one or more tabs within the same browser. This is something to keep in mind when creating more complex queries.

Figure 8 – Our new SQL statement.

Now that we have defined the conditions and sorting columns, we can see how CacheBack has automatically re-formatted these values into a syntactically correct SQL Statement as show below.

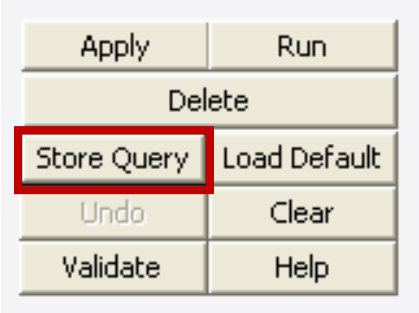
```
SELECT * FROM URLs WHERE (URL LIKE '%hotmail%') AND (URL LIKE '%yahoo%') AND (IsWebPage = True)
AND (HTMLBody LIKE '%mail%') AND (CacheFileExists = True)
AND (ActionDateUTC BETWEEN #7/17/2009# AND #6/17/2009#)
ORDER BY Host DESC, ActionDateUTC DESC
```

For the purpose of illustrating this more clearly, we have taken the liberty to format the SQL View using a couple of line breaks. Now that we have this query completed, we should consider saving it to our project file for future use.

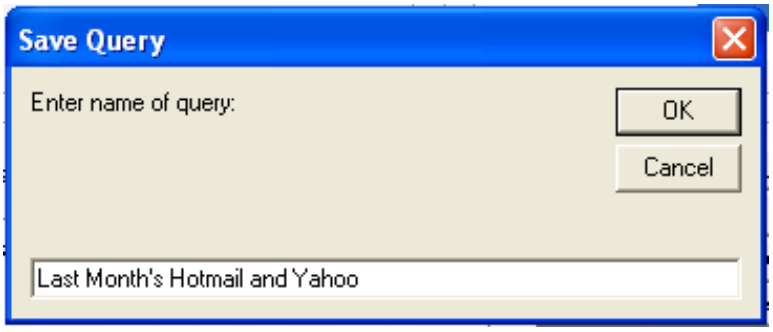
SAVING QUERIES

To save a query that has been created using the Query Tab, using the Query Builder (or by typing directly into the SQL View box):

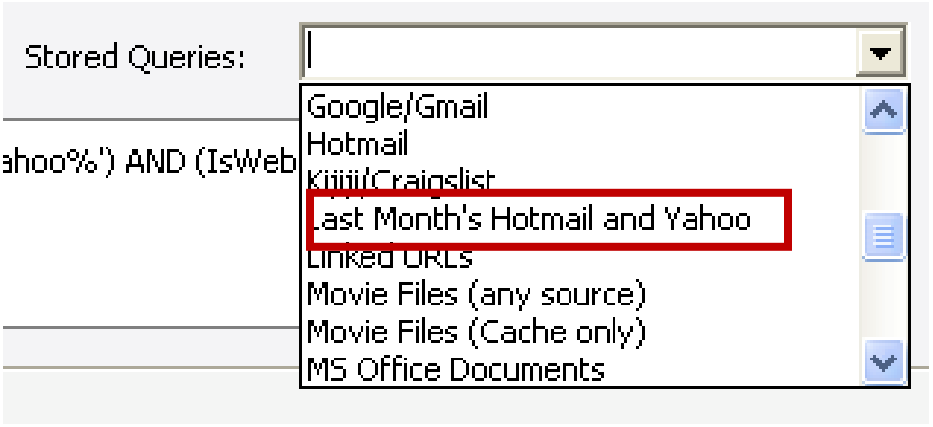
Step 1. Select the "Store Query" button on the Query Tab, which is located in the Data Pane.



Step 2. Enter a "name" for the query being sure not to use the name of an already existing query, particularly those that begin with an "_" (underscore).



Step 3. Verify that the query has been saved to the Stored Queries dropdown list box. This box is situated on the Query Tab, immediately beneath the Query Builder section.



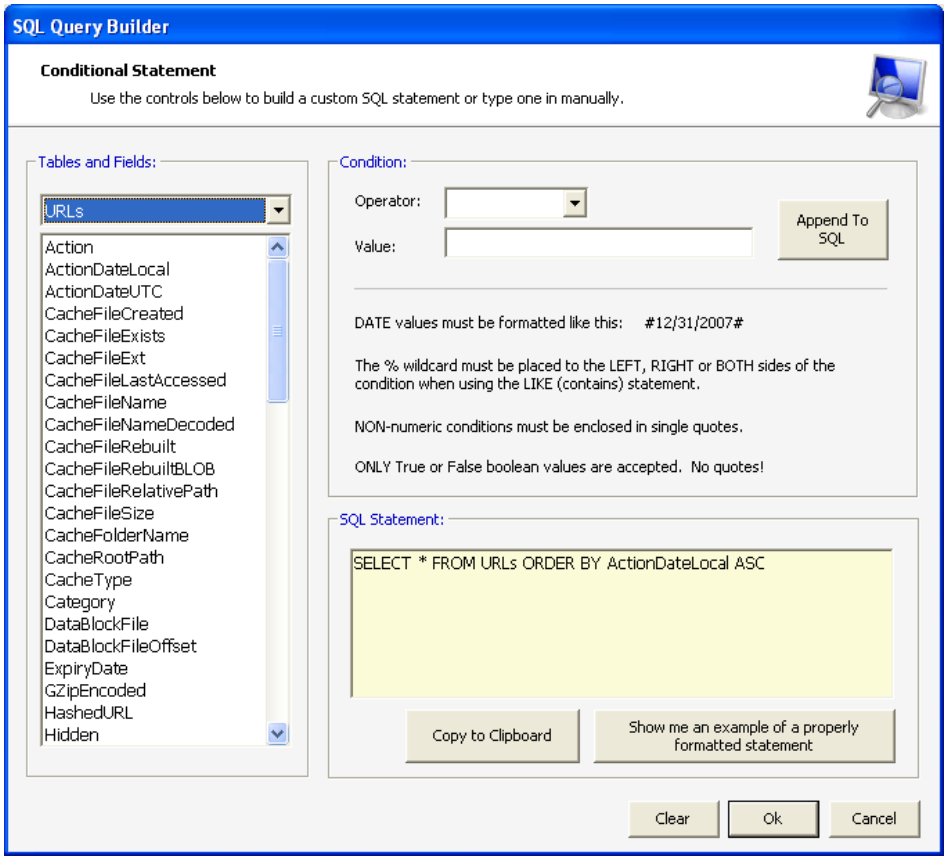
Using the SQL Query Builder Window

If you have read the preceding section on Using the Query Builder, then understanding how to use the SQL Query Builder Window will be very straight forward. It is so similar in fact that we have omitted any discussion on this feature because it would be redundant.

The only true differences between this feature and the Query Builder on the Query Tab are:

1. There is no Query Builder table where rows (conditions) are added and removed.
2. Conditions are appended to the SQL Statement using the Tables and Fields list box AND the Condition options.
3. You cannot save queries constructed using the SQL Query Builder Window.

Figure 1 – The SQL Query Builder Window.

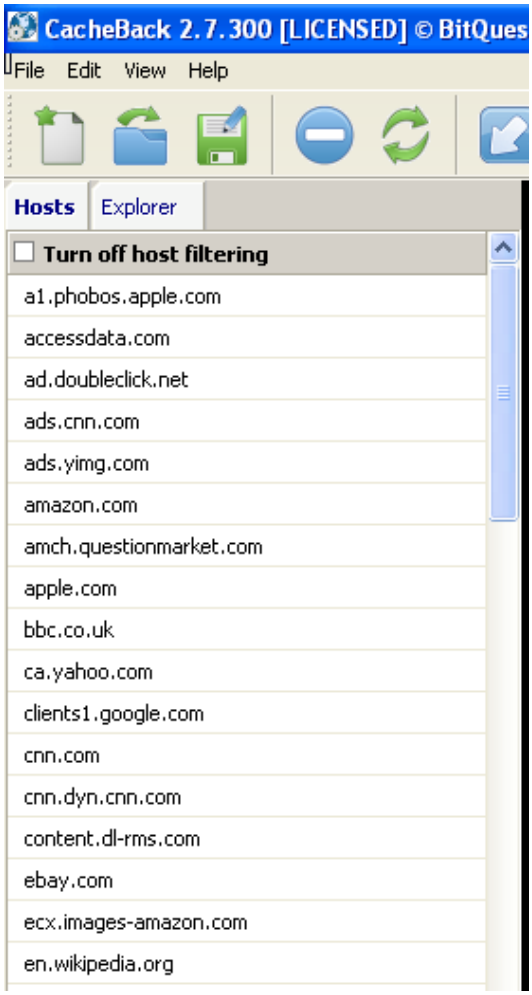


Managing Records Using Host Filtering

So far we've discussed the use of queries and filters. We have also shown you how to create custom queries which can then be saved to your CacheBack project file for future use. Now we will discuss the use of Host filtering.

Host filtering employs the use of the Host Tab located in the Features Pane as shown below.

Figure 1 – The Host Tab.



The first thing we need to do is distinguish our use of the word "filtering" in this context from our earlier discussion about using "Filters".

HERE'S HOW IT WORKS…

CacheBack's default query (when you create a new project) is defined as follows:

```
SELECT * FROM URLs ORDER BY ActionDateUTC ASC
```

This SQL definition becomes the program's **[_DEFAULT]** query which is stored inside the database as a properly formatted *Stored Procedure* (aka: query).

Whenever you select a Host from the list of hosts in the Host tab, CacheBack dynamically creates a new *stored procedure (query)* that requests only those results of the _DEFAULT query that match the selected Host. The following SQL statement illustrates this point:

```
SELECT [_DEFAULT].* FROM URLs WHERE [_DEFAULT].Host = 'google.com'
```

HERE'S WHY IT WORKS…

All queries in CacheBack utilize the "*" (asterisk) wildcard character which indicates that ALL COLUMNS (fields) are to be returned with any query results.

A simple query normally acts on the records in a single Table within a database. However, it is also possible for query to act on the records returned by another query. Think of this as layering. It would be the same as saying:

"Please show me all records, including all of their columns (*), that are found inside the _DEFAULT query's returned search results. However, make sure to show me only those records where the Host column has a value that is equal to 'google.com' "

This is why Host filtered results are so much quicker to return than by using filters alone.

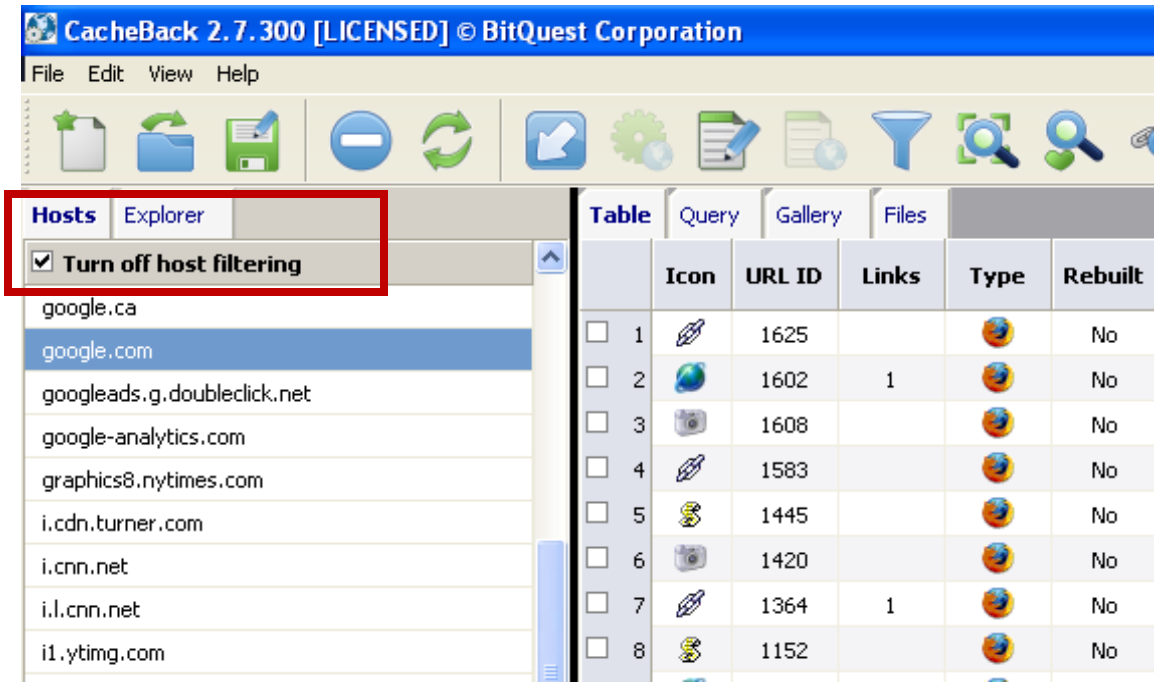
TURNING "OFF" HOST FILTERING

Once you have selected a Host from the Host tab, that selection will always run on top of the [_DEFAULT] query.

The only way to turn this off is to use the "**Turn off host filtering**" checkbox located at the top of the Host tab.

NOTE: You may have to click on the Refresh toolbar button in order to reload the Table or Gallery after turning Host Filtering off.

Figure 1 – Turn off host filtering.



Rebuilding Web Pages

Introduction

Before we can discuss the actual "rebuilding" of web pages, we first need to understand and appreciate that CacheBack requires "files" to work with. That being the case, it would be futile to ask CacheBack to rebuild a web page when the only URL records inside the current project file all originated from an Internet History. This is due to the simple fact that *history* URLs refer to a user's *activity* and make no reference to actual cache files.

Therefore, when we talk about rebuilding web pages, we are referring to "cache" content (files that exist) from any of the five (5) top browsers. This includes Internet Explorer, Firefox, Opera, Safari and Google Chrome. Each browser stores its offline content (aka: "cache") in different formats. Some use an external file indexing system (eg: IE, Firefox and Google Chrome) whereas others use an internal file indexing system (eg: Opera, Safari).

Each system, regardless of the browser, maintains the ingredients required to put a cached web page back together so that it looks the same was as it was originally viewed. To do this, these indexing systems catalog the ingredients based on their original URL and they also record certain dates and times related to the activity (eg: Last Visited).

When we rebuild web pages using CacheBack, we are instructing the program to look up all of the ingredients and, based on a complex series of conditional statements, sort through the available lists, and make the best decision about which ingredient (eg: picture) belongs to which web page, and where (within the file). The complex details of how this takes place is beyond this user manual. It suffices to say however that CacheBack does not "guess" about what goes where. Instead, it bases its results on carefully weighed conditions and balances of probability.

Since data sources provided to CacheBack can originate from various locations such as a read-only CD-ROM, all web pages are first COPIED to the user-specified Temp folder, where only then, is it manipulated as required. The original evidence is never altered.

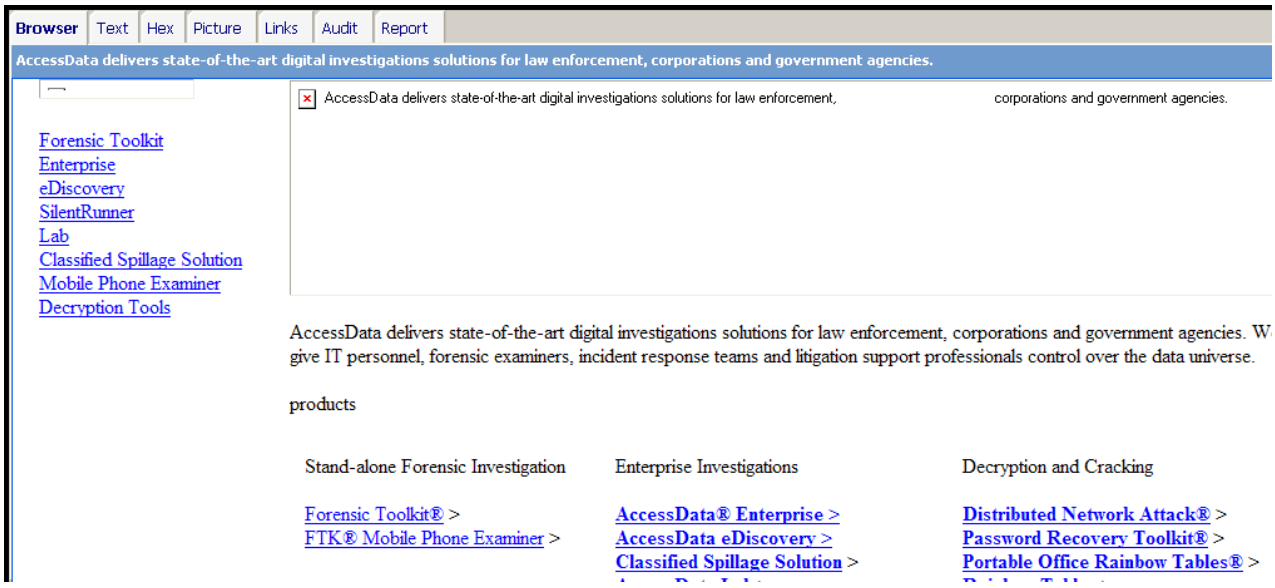
The Four Steps to Rebuilding a Web Page

In order that any given web page can be rebuilt using CacheBack, we must first have completed the following initial tasks:

1. Create a new project file.
2. Import Internet "cache" files using the Import Wizard.
3. Display one or more URL record entries in the Table or Gallery view where the column name "File Exists" has a value of True AND the Icon column in the Table (far left) has a picture of a "globe" to depict HTML content.
4. (optional) Confirm that the selected record(s) are web page(s) that are capable of being rebuilt. This may be as simple as previewing the contents of the file using the Browser viewer pane.

The following is a sample selected web page BEFORE being rebuilt. Notice how the font defaults to an unattractive Times New Roman and there appears to be some pictures missing.

Figure 1 – Web page BEFORE being rebuilt using CacheBack.

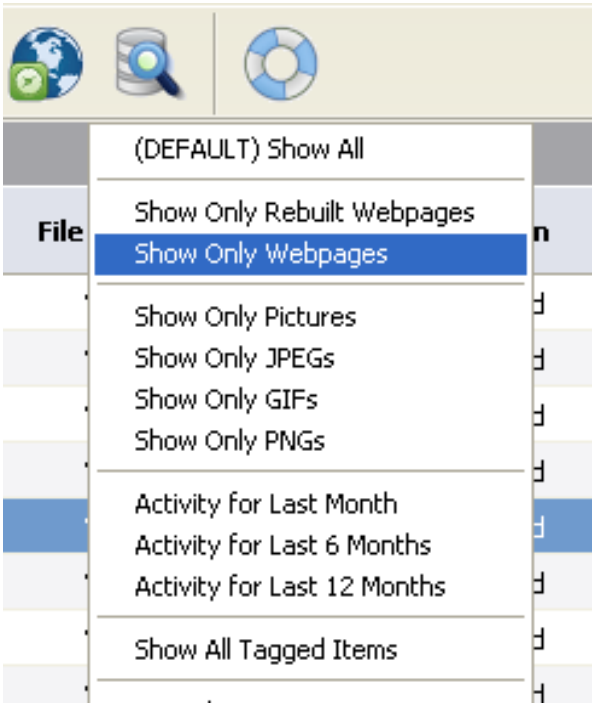


Once we have some valid records to work with (eg: in the Table), we then need to follow these next three simple steps.

Step 1: Obtain Working Data

After creating a new case, most users will want to get right down to business and start rebuilding any and all web pages. Due to the likelihood (in almost every case) of there being several hundred or thousand additional URLs (eg: pictures, history URLs) *that are not web pages*, it makes sense to try and work with a much smaller, and meaningful recordset.

To do this, we will want to use Filters, Queries and/or Host Filtering – or any combination of the above to narrow the number of records in the Table. However, we strongly recommend that users first try the Quick Query: "Show All Webpages". This feature is available via the Quick Queries toolbar button and displays a context (popup) menu listing the most common search queries.



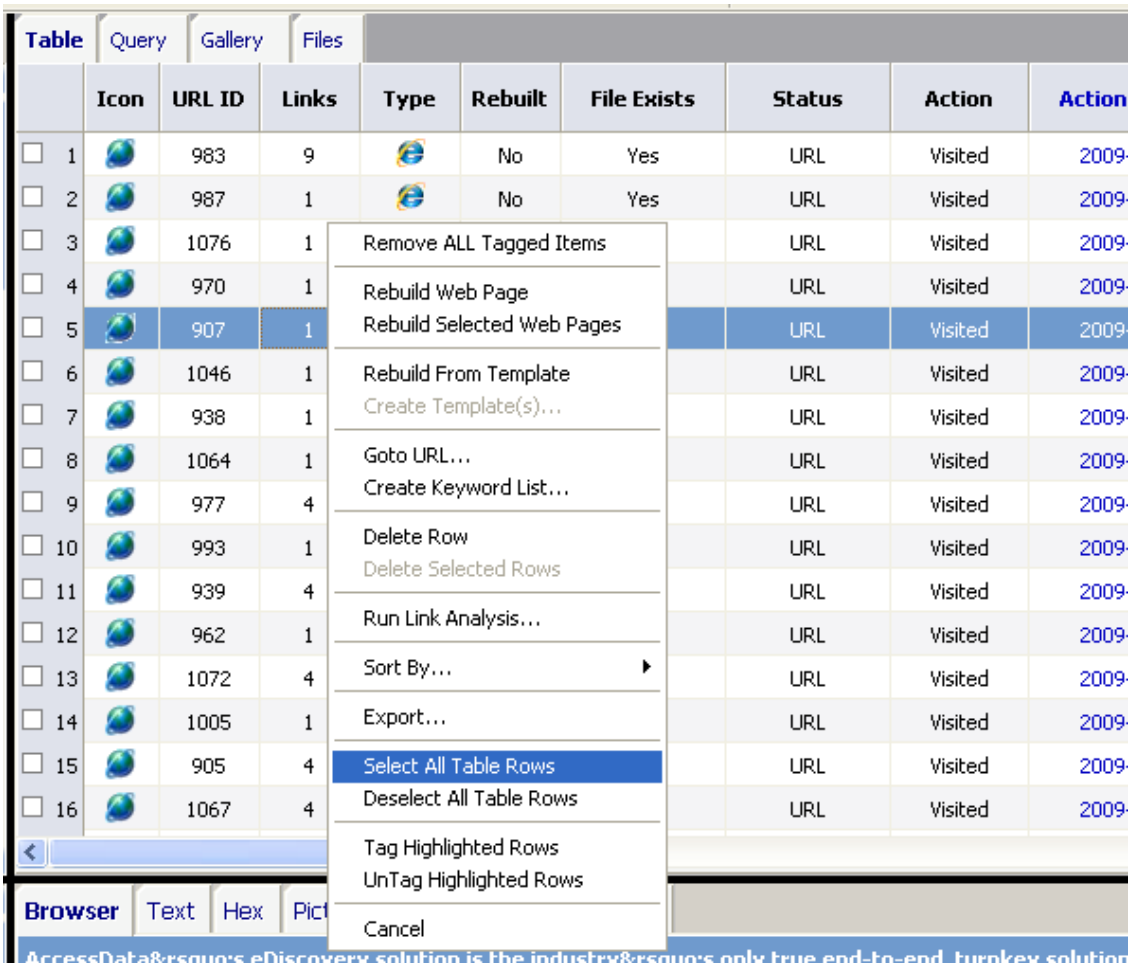
Step 2: Selecting Records

A checkmark must be present in the far left column for each URL record (web page file) that you intend on rebuilding. NOTE: It is not necessary at this point to know with absolute certainty if a selected record is (a) in fact a web page, and (b) if it can be rebuilt. CacheBack will figure this out for you.

Selecting records can be done in a few different ways. First, you can simply use the mouse pointer to click inside each record of interest.

Second, you can use the SHIFT + UP/DOWN arrow keys to sweep or highlight more than one record. Then simply right-mouse-click overtop of the Table and from the context (popup) menu that will then appear, select “Tag Highlighted Rows”.

And lastly, you can simply right-mouse-click overtop of the Table, and from the context (popup) menu that will appear, select “Select All Table Rows”.

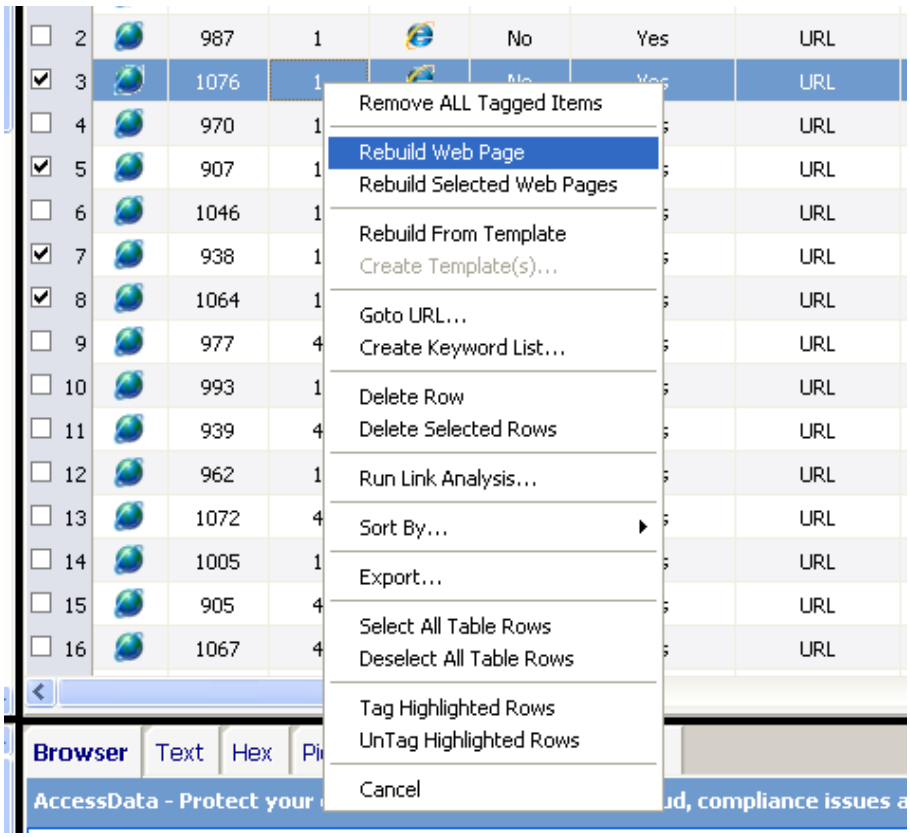


Step 3: Start the Rebuilding Process

Once the necessary URL records (web pages) have been identified and selected, the next step is to simply tell CacheBack to start the rebuilding process. There are two common ways of making this happen.

The first and easiest method is to click on the toolbar's Rebuild button ( ).

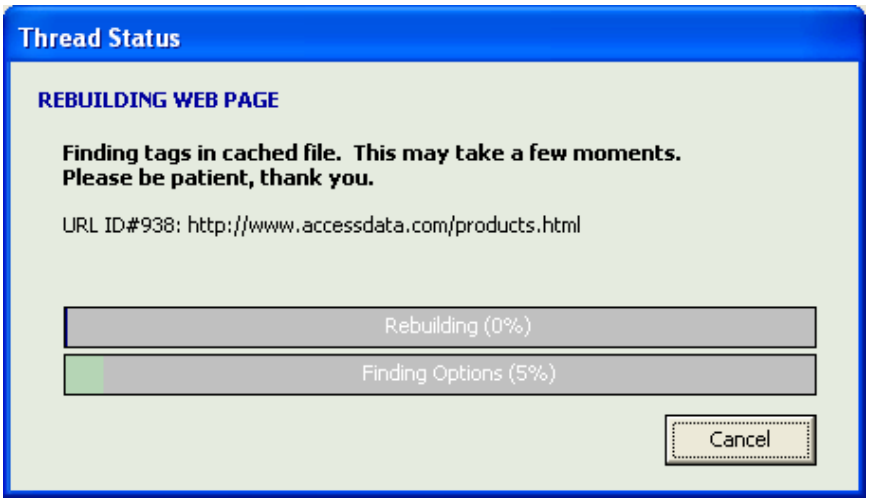
The second method is by using the context (popup) menu that appears when you right-mouse-click anywhere inside the Table like so:



NOTE: Using the single option of "Rebuild Web Page" will rebuild ONLY the URL record (web page) that is currently "highlighted". Using the plural option of "Rebuild Selected Web Pages" will rebuild all URLs that have been checkmarked.

Step 4: Rebuilding process takes place.

During the rebuilding process, the Thread (status) Window will appear and update the progress onscreen with one or more progress bars.



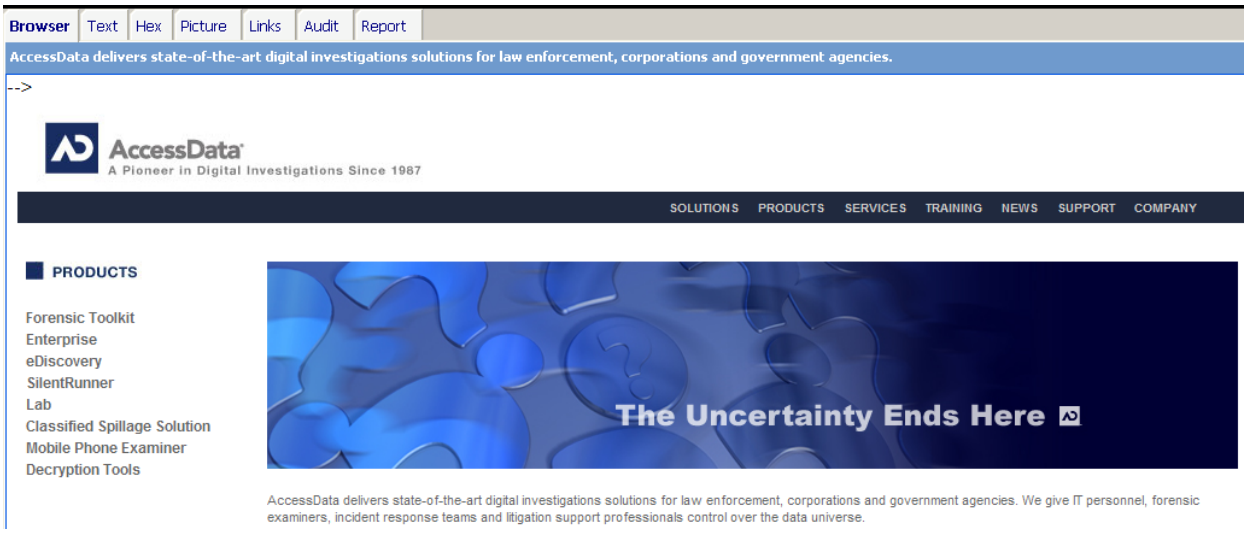
The following stages of activity will then take place, in this order:

1. All *original* TAGS within the selected file(s) are identified, parsed, and stored in the project file.
2. Each tag is cross referenced against ALL other URLs stored in the project file.
3. All potential matches are then stored in a separate relationships table.
4. Next, possible *replacement* tags are discovered within the project file and listed aside for validation.
5. Each identified *original* tag is compared against the list of "possible replacement tags" and a series of complex conditional statements are evaluated (eg: URL path, host / domain attribute, secure socket protocol, fully qualified path VS. relative path, etc.).
6. The best (or only) replacement option (tag) is then selected and stored in reference to the specific *original* tag that is being replaced.
7. An "Accuracy Score" is then registered and stored in relation to the replaced tag, for auditing purposes.

IMPORTANT: Large project files may take unusually long to process even a small number of web pages. This is usually due to additional, irrelevant "history" URLs being present in the project file WHICH have be included in the rebuilding process. This simply means that: for each URL in each web page that is being rebuilt, CacheBack needs to cycle through ALL URLs in the project file to find the list of compatible replacement tag options. **The only**

way to reduce this delay is to examine Cache artifacts separately, at least for the purpose of rebuilding web pages. Again, this only applies in the rare cases where there are tens or hundreds of thousands of URLs in the project file (most of which may originate from History files instead of Cache files).

Figure 2 – Web page AFTER being rebuilt using CacheBack.



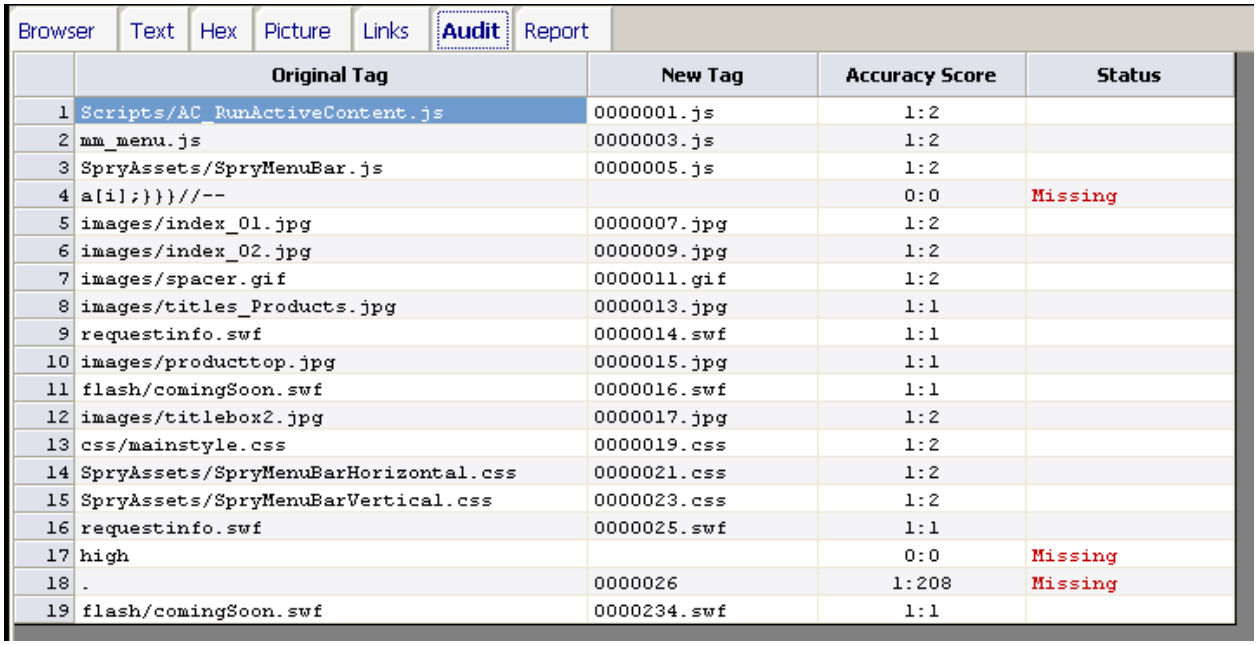
Notice how the fonts now appear much cleaner and likely. The graphics have also been provided for the company's logo, the square bullet for the Products title, and of course the main advertisement graphic depicted with the caption "The Uncertainty Ends Here".

Reviewing the Audit Tab

After a web page has been rebuilt, it is helpful to learn what items were successfully replaced within the page, and which were not. For these details, we simply switch over to the Audit Tab. Items with a Status of "Missing" simply indicate one of two things:

a) The original ingredient file could not be found in the cache. This sometimes happens if a webpage was "cancelled" half way through the download process. This also occurs if the threshold (capacity or retention date) for the cache has been exceeded and as a result, one or more items were discarded by the browser to make room for new content.

b) The value noted in the "Original Tag" column was an extra parsed value that does not conform to any known URL path.

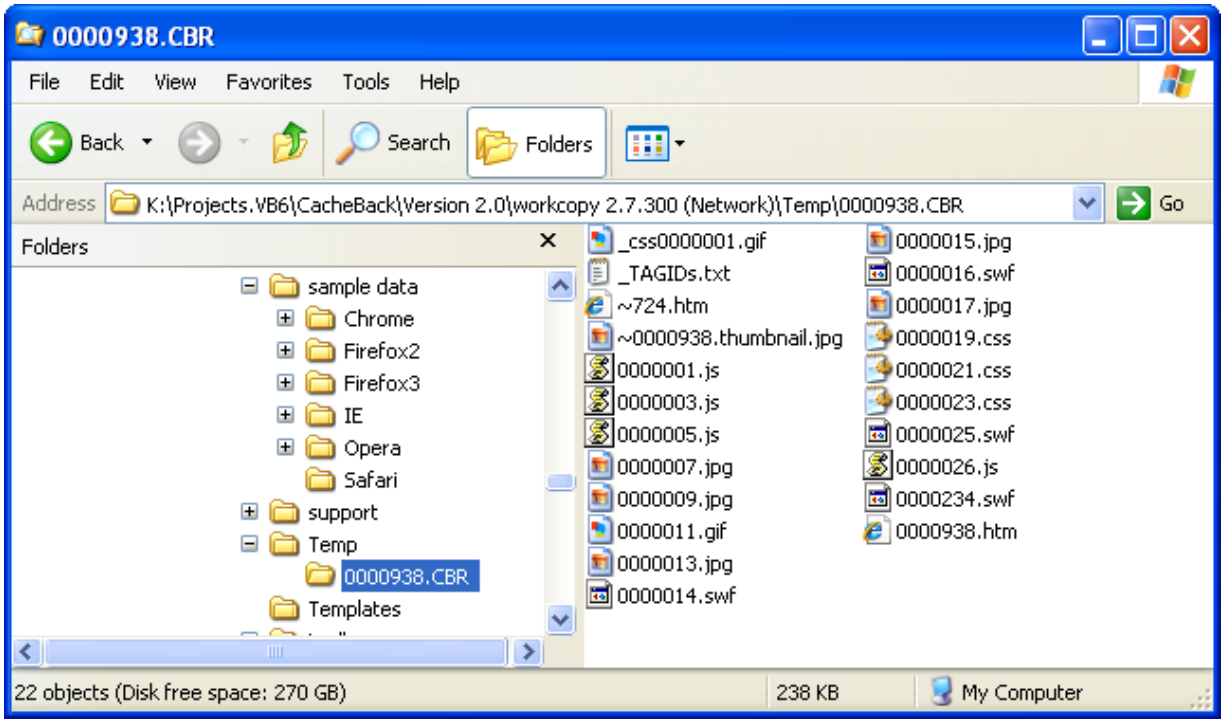


	Original Tag	New Tag	Accuracy Score	Status
1	Scripts/AC_RunActiveContent.js	0000001.js	1:2	
2	mm_menu.js	0000003.js	1:2	
3	SpryAssets/SpryMenuBar.js	0000005.js	1:2	
4	a[i];}}}//--		0:0	Missing
5	images/index_01.jpg	0000007.jpg	1:2	
6	images/index_02.jpg	0000009.jpg	1:2	
7	images/spacer.gif	0000011.gif	1:2	
8	images/titles_Products.jpg	0000013.jpg	1:1	
9	requestinfo.swf	0000014.swf	1:1	
10	images/producttop.jpg	0000015.jpg	1:1	
11	flash/comingSoon.swf	0000016.swf	1:1	
12	images/titlebox2.jpg	0000017.jpg	1:2	
13	css/mainstyle.css	0000019.css	1:2	
14	SpryAssets/SpryMenuBarHorizontal.css	0000021.css	1:2	
15	SpryAssets/SpryMenuBarVertical.css	0000023.css	1:2	
16	requestinfo.swf	0000025.swf	1:1	
17	high		0:0	Missing
18	.	0000026	1:208	Missing
19	flash/comingSoon.swf	0000234.swf	1:1	

Inspecting the .CBR (CacheBack Rebuilt) Folders

As a matter of interest, users can take a look inside the project file's Temp folder and look for a .CBR folder for each rebuilt web page. Each .CBR folder has a filename that is the URL_ID padded with zeros (for better sorting capabilities in Windows Explorer). The folder contains one or more files that makeup the rebuilt file page. In particular, the **_TAGIDs.txt** file contains an index or "lookup chart" of each ingredient's (original tag's) URL_ID, and the associated FILENAME that was assigned to the harvested (copied out) data.

The following is an example of the file contents for the .CBR folder from the preceding example, as viewed using Windows Explorer in Thumbnail View.



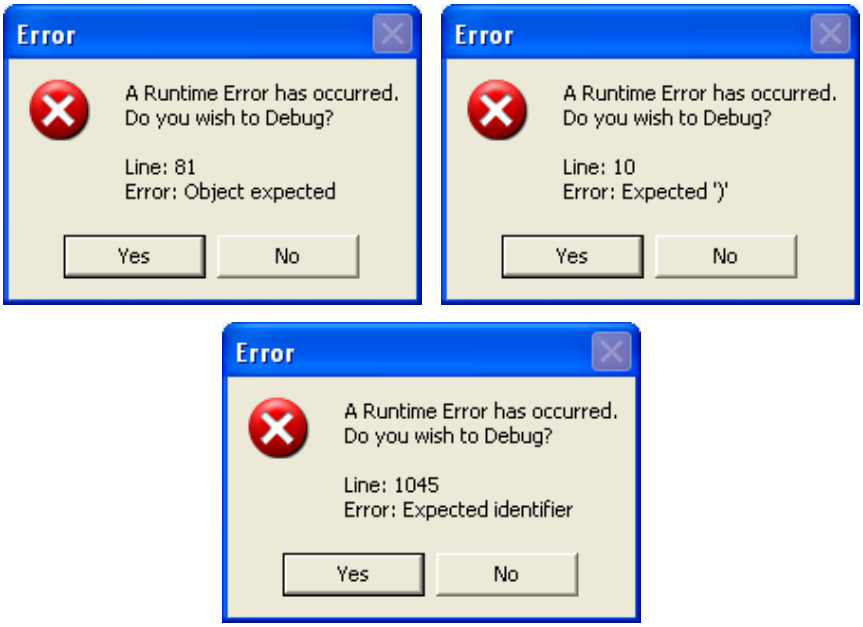
NOTE: Like each .CBR folder, the above folder contains the "**0000938.htm**" file which is the actual rebuilt web page. There is also a thumbnail version of that same web page which has the filename "**~0000938.thumbnail.jpg**". This naming convention is adopted for all rebuilt web pages.

Preventing Popups and Script Error Messages

Despite the fact that CacheBack uses an “offline” browser control (viewer) to display web pages, it does not prevent any embedded scripts from attempting to execute on the local machine (eg: javascript, vbscript). The behaviour of attempting to process dynamic web page content is inherent with the loading process for HTML formatted data. As a consequence, this type of activity is closely monitored by Windows. Depending on the security level of settings for Internet Explorer on a user’s system, it is possible that certain Message Boxes will appear (as popup windows, whenever a page is loaded) to warn the user about a potential problem or risk.

While this behaviour may be tolerable to a certain degree, it can easily become a nuisance when trying to navigate from URL to URL within CacheBack. This will also occur automatically whenever calling the Rebuild Web Page routines within CacheBack, especially as the program cycles through each selected web page.

Figure 1- Sample Script Errors.



Fortunately, there are a couple of ways to handle or prevent these annoyances and allow CacheBack to function normally.

Disabling Javascripts

At the start of each new project file, CacheBack, by default, sets a Global option that "Disables Javascripts". By toggling (enabling/disabling) this feature, users can control how CacheBack handles webpages that contain script elements.

Here is a sample snippet of source code from an offending web page. The opening (<script>) and closing (</script>) tags are enlarged for illustration purposes.

```
<script language="JavaScript" src="mm_menu.js"
type="text/javascript"></script>
<script src="SpryAssets/SpryMenuBar.js" type="text/javascript"></script>
<!--
function mmLoadMenus() {
if (window.mm_menu_0430033221_0) return;

window.mm_menu_0430033221_0 = new Menu("root",140,17,"Arial
Narrow",11,"#000033","#00CCCC","#D8F3FF","#000000","left","middle",3,0,1000,-
5,7,true,false,true,0,true,true);

mm_menu_0430033221_0.addMenuItem("Courses","location='training.html'"AD Ins
tructors"ACE Information");
   mm_menu_0430033221_0.hideOnMouseOut=true;
   mm_menu_0430033221_0.bgColor='#233455';
   mm_menu_0430033221_0.menuBorder=1;
   mm_menu_0430033221_0.menuLiteBgColor='#FFFFFF';
   mm_menu_0430033221_0.menuBorderBgColor='#000033';

  window.mm_menu_0430033425_0 = new Menu("root",140,17,"Arial
Narrow",11,"#000033","#00CCCC","#D8F3FF","#000000","left","middle",3,0,1000,-
5,7,true,false,true,0,true,true);

mm_menu_0430033425_0.addMenuItem("Support Center","window.open('http://supp
ort.accessdata.com', '_blank');");
   mm_menu_0430033425_0.hideOnMouseOut=true;
   mm_menu_0430033425_0.bgColor='#233455'
   mm_menu_0430033425_0.menuBorder=1;
   mm_menu_0430033425_0.menuLiteBgColor='#FFFFFF'
   mm_menu_0430033425_0.menuBorderBgColor='#000033'
window.mm_menu_0430033557_0 = new Menu("root",140,17,"Arial
Narrow",11,"#000033","#00CCCC","#D8F3FF","#000000","left","middle",3,0,1000,-
5,7,true,false,true,0,true,true);

mm_menu_0430033557_0.addMenuItem("Overview"Contact Us"Reseller"Career 
Opportunities"Management");
   mm_menu_0430033557_0.hideOnMouseOut=true;
   mm_menu_0430033557_0.bgColor='#233455'
   mm_menu_0430033557_0.menuBorder=1;
   mm_menu_0430033557_0.menuLiteBgColor='#FFFFFF'
   mm_menu_0430033557_0.menuBorderBgColor='#000033'

mm_menu_0430033557_0.writeMenus();
}// mmLoadMenus()
```

```
//-->
<!--
function MM_preloadImages() { //v3.0
  var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
    var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length;
i++)
    if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
}
//-->
<script language="JavaScript" type="text/javascript">
<!--
function MM_reloadPage(init) {  //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);

function MM_openBrWindow(theURL,winName,features) { //v2.0
  window.open(theURL,winName,features);
}

function MM_goToURL() { //v3.0
  var i, args=MM_goToURL.arguments; document.MM_returnValue = false;
  for (i=0; i<(args.length-1); i+=2) eval(args[i]+".location='"+args[i+1]+"'");
}
//-->
</script>
```

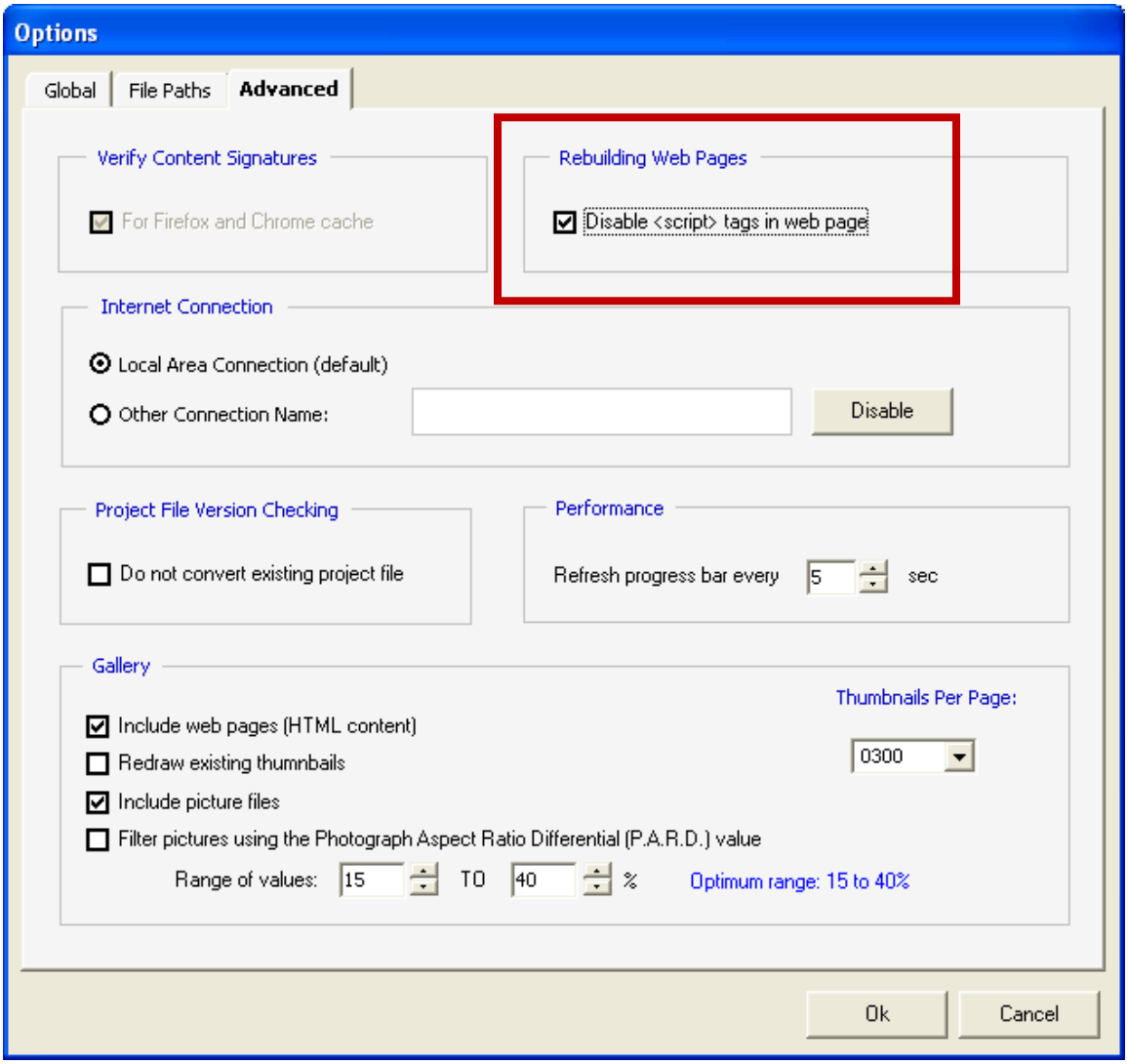
In order for the web page that contains this code to render without any error messages within CacheBack, there are a couple of things we can do. The first approach is to simply enable the “Disable <script> tags in web page” option in the global Options Window.

To access this feature, simply goto the View menu at the top of the window, then select Options…
Next, on the Advanced settings table, simply “check” or “uncheck” the above noted option as required. For this exercise, we would “enable” (check) this option.

The following page provides a screenshot of the Options window and the Advanced settings tab where the “**Disable <script> tags in web page**” option is found.

Figure 2 – The Options Window (Advanced Tab).

You will notice here that the "**Disable <script> tags in web page**" option is checked (activated).



When this option is selected, CacheBack will dynamically insert *comment out tags* (eg: <!-- and -->), next to the above noted <script> and </script> tags . For example, **<script>** would become **<!-- <script** and **</script>** would become **<script>-->**. *Comment out* tags will be ignored by the Browser viewer control (in CacheBack) and as a result, run-time script error messages (in most cases) should no longer appear.

Configuring IE on the User's Workstation

There are times, however, where embedded scripts are sophisticated enough to circumvent the aforementioned tactic, simply by the manner in which the blocks of code are generated. Normally, the code is pre-assembled and plainly visible in its syntactically correct format (eg: function and variable declarations, inline code statements). However, script writing today can become quite convoluted in a deliberate attempt to foil malware detection software, and the like.

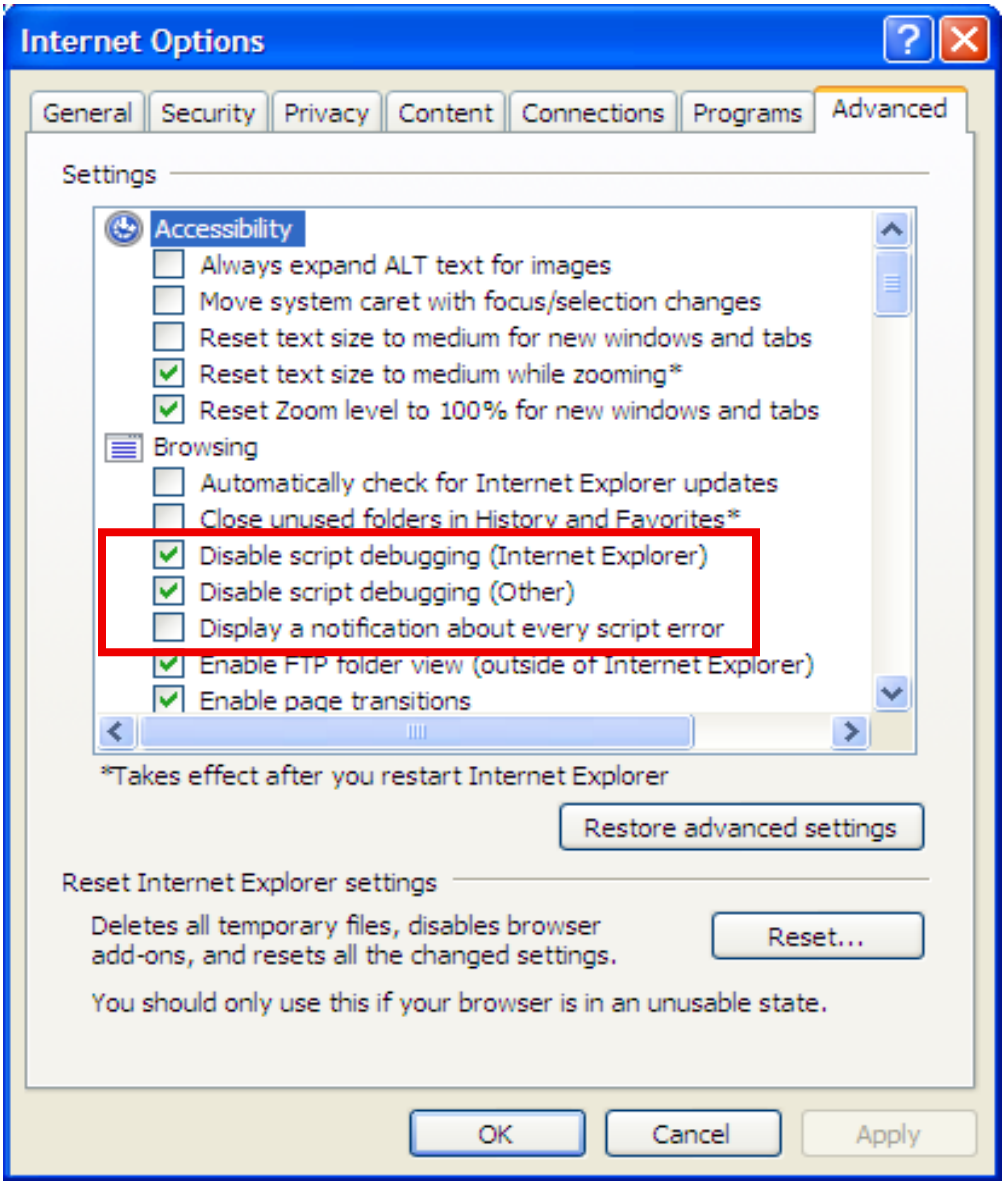
This behaviour is effected by "constructing" the syntactically correct "code blocks" by assembling or concatenating variables (placeholders for information) only at the time the code is executed. As a result, CacheBack may not always be able to prevent script errors from appearing.

There is however another mechanism available and it is present within the user's system's Internet Explorer browser security settings.

The following diagram illustrates two (2) advanced settings that will directly affect the displaying of error messages by Internet Explorer, and any third party code libraries that leverage the IE APIs.

Figure 1 - Internet Explorer Advanced Options.

The following window is found under the “Tools”.....”Internet Options” menu within IE.



Notice that the first two options to “Disable script debugging” have been checked and that the next item to “Display a notification about every script error” is NOT checked. This combination of settings should prevent IE (and code referencing the IE APIs) from displaying any further script error messages.

Creating Reports

Introduction

The ability to create a report within CacheBack is tied directly to:

(a) The presence of one or more records in the Table or Gallery,

(b) The "selection" of one or more records in the Table or Gallery.

NOTE: Records in the Gallery can only be "checkmarked" whereas the Table, at minimum, can simply have a single row *highlighted*. If there are NO records highlighted or selected, then the "Report" Toolbar button will be *disabled*.

CacheBack 2.7.4 provides examiners with the option of creating a *Graphical Report* or a *Tabular Report*. A Graphical Report will provide a thumbnail representation of *web pages or picture files*, including a variety of metadata about each record (eg: Visit Count, ActionDateUTC, ActionDateLocal, URL, etc.). A Tabular Report on the other hand can report on ANY URL type and lists the records in a grid or spreadsheet styled format.

Both report types generate a single HTML document that is displayed inside the Report Tab which is found inside the Viewer Pane.

Picture 1 – A sample *Graphical Report* being displayed in the Report Tab.

Graphical Report

In order to create this type of report, at least one record in the Table or Gallery must be selected AND must be a *web page or picture file*.

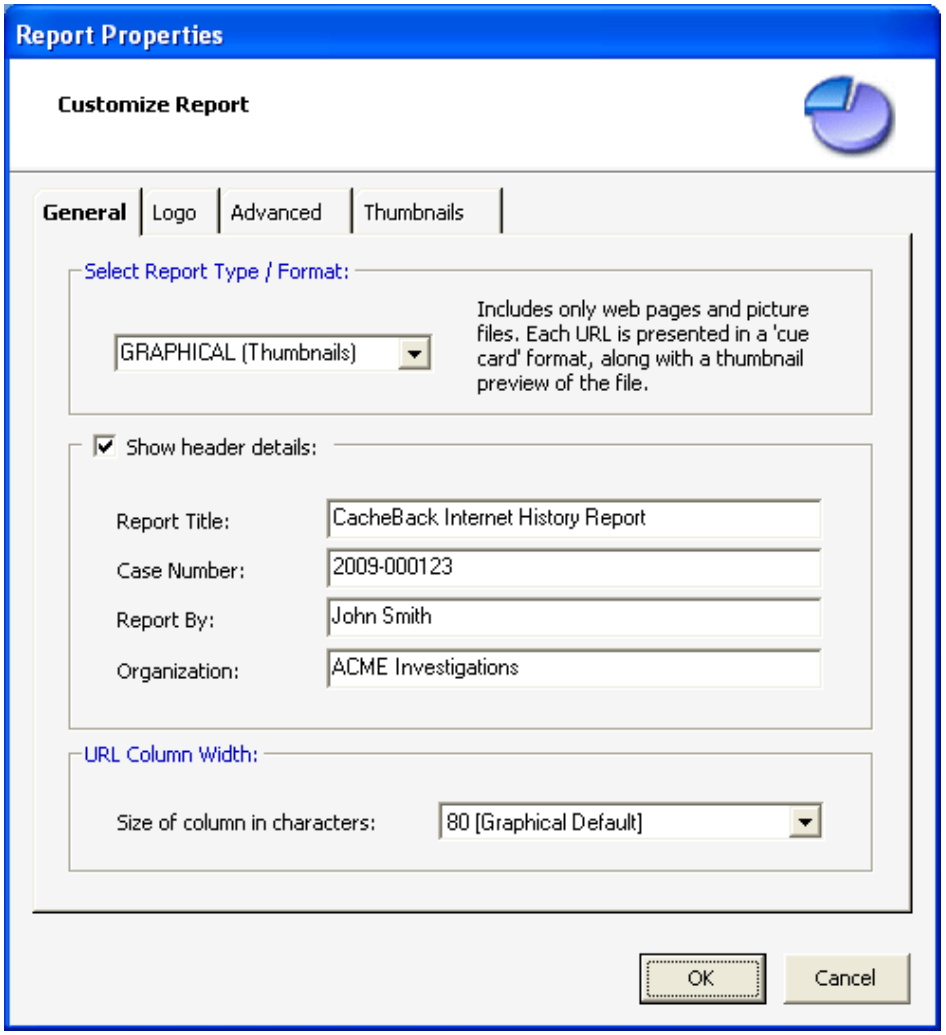
IMPORTANT: If you choose any other types of URLs and then choose to create a Graphical Report, do not be surprised if there are no thumbnails displayed in the report. This commonly occurs if examiners attempt to use this feature for URLs from a "History" data source only. It will also occur if the URLs originated from a "Cache"

data source but are file types other than web pages and pictures (eg: javascripts, cascading stylesheets, .zip files, etc.).

To create a Graphical Report, select the enabled "Report" () button from the Toolbar.

This will invoke the Report Window and default to the General Tab as shown below.

Picture 1 – The General Tab.



General Tab

This tab contains the Report Type dropdown list box which will default to a Graphical Report.

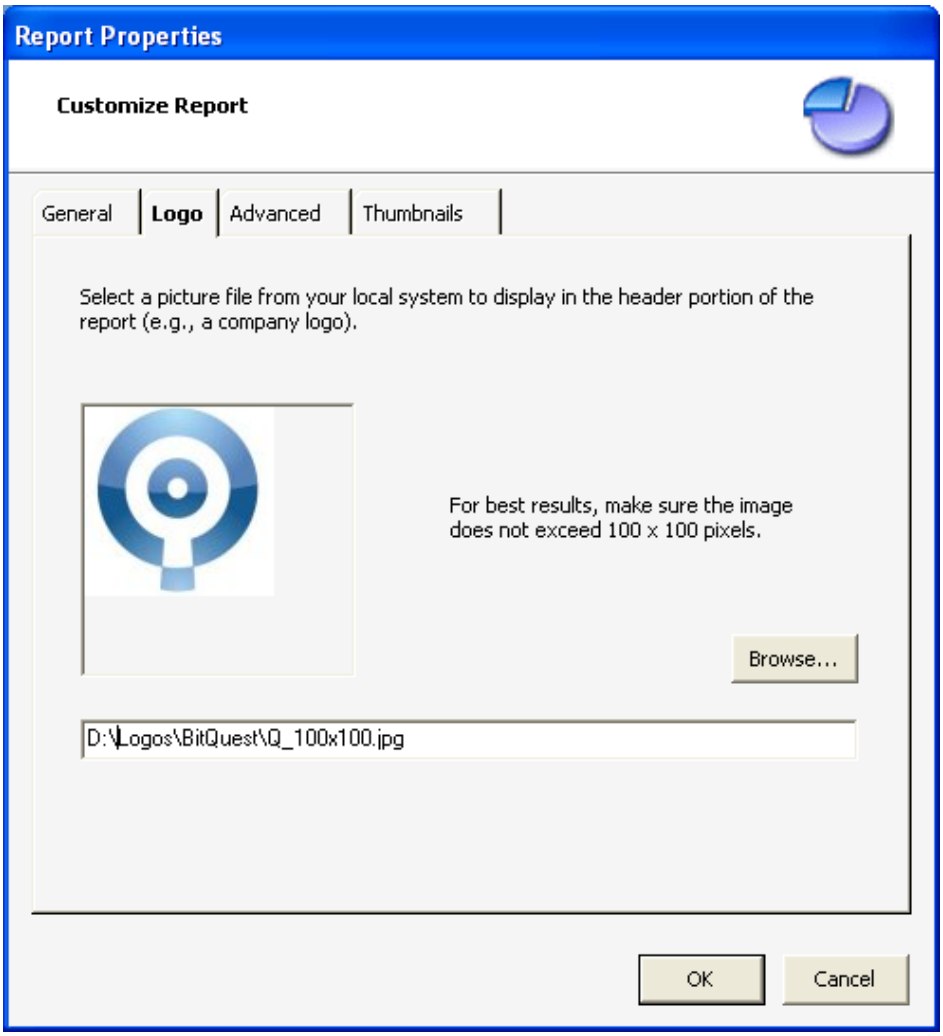
The “Show header details” section provides the examiner with the ability to customize the header portion of the report. The values entered here will be remembered for future reports.

The “URL Column Width” dropdown list box is used to define the width, in pixels, of the LAST COLUMN for each URL record. The purpose of this option is to ensure that unusually large URLs can be read without having to scroll way off the right of the screen. Secondly, it ensures that the printed version of the report will fit all URLs on the default Portrait report layout.

Logo Tab

This tab can be used to customize the logo that appears in the top right corner of the report. If no valid logo path is provided, then the CacheBack default Logo will be used.

Picture 2 – The Logo Tab.



Advanced Tab

This tab provides customized options for reporting additional details about each URL record.

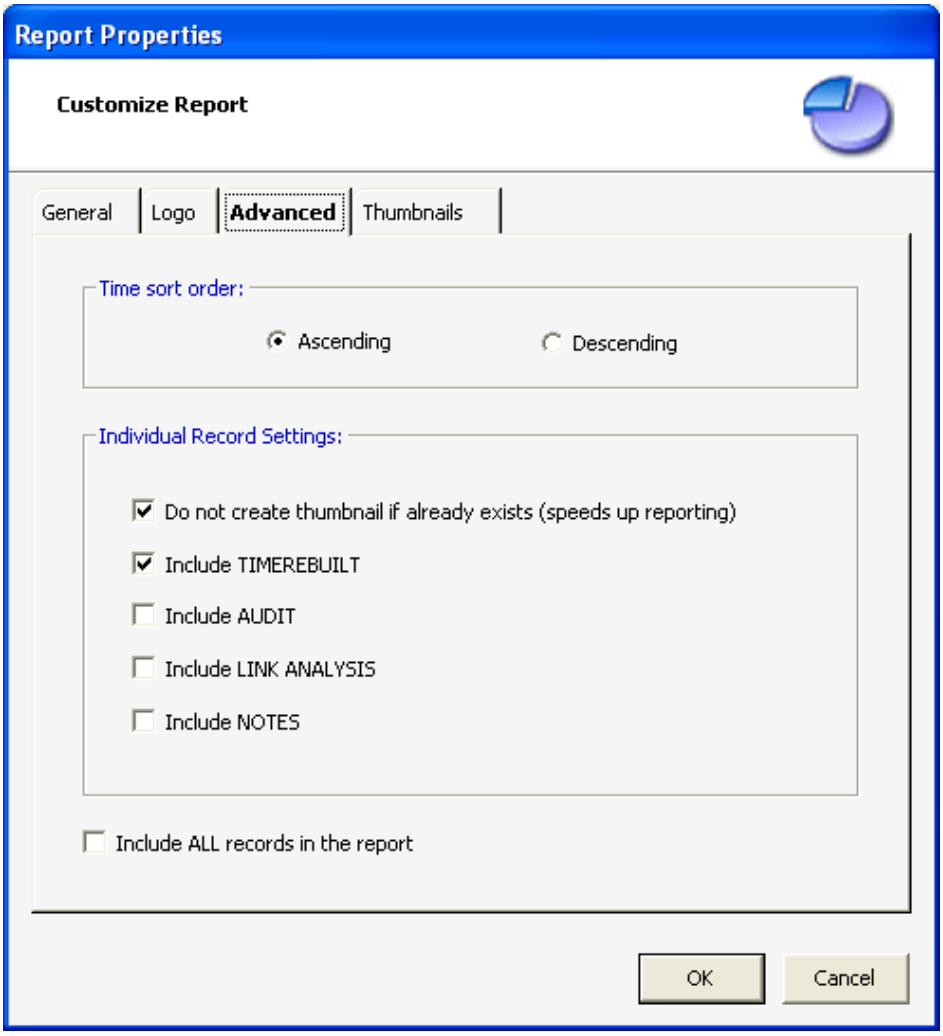
The “Include TIME REBUILT” option is provided by request of some CacheBack users who did not necessarily want the showcase the date and time that the examination took place.

The “Include AUDIT” option provides a list of the Tags that were replaced inside a rebuilt web page, almost identical to the information displayed on the Audit Tab in the Viewer Pane. (See discussion about the Audit Tab on page 31).

The “Include LINK ANALYIS” option provides a list of the related URLs (if any) that were discovered during the Link Analysis processing. (For more information about Link Analysis, please see page 105).

The “Include NOTES” option will print any *examiner notes* that were typed in for a given URL. (For more information about Notes, please see page 174).

Picture 3 – The Advanced Tab.



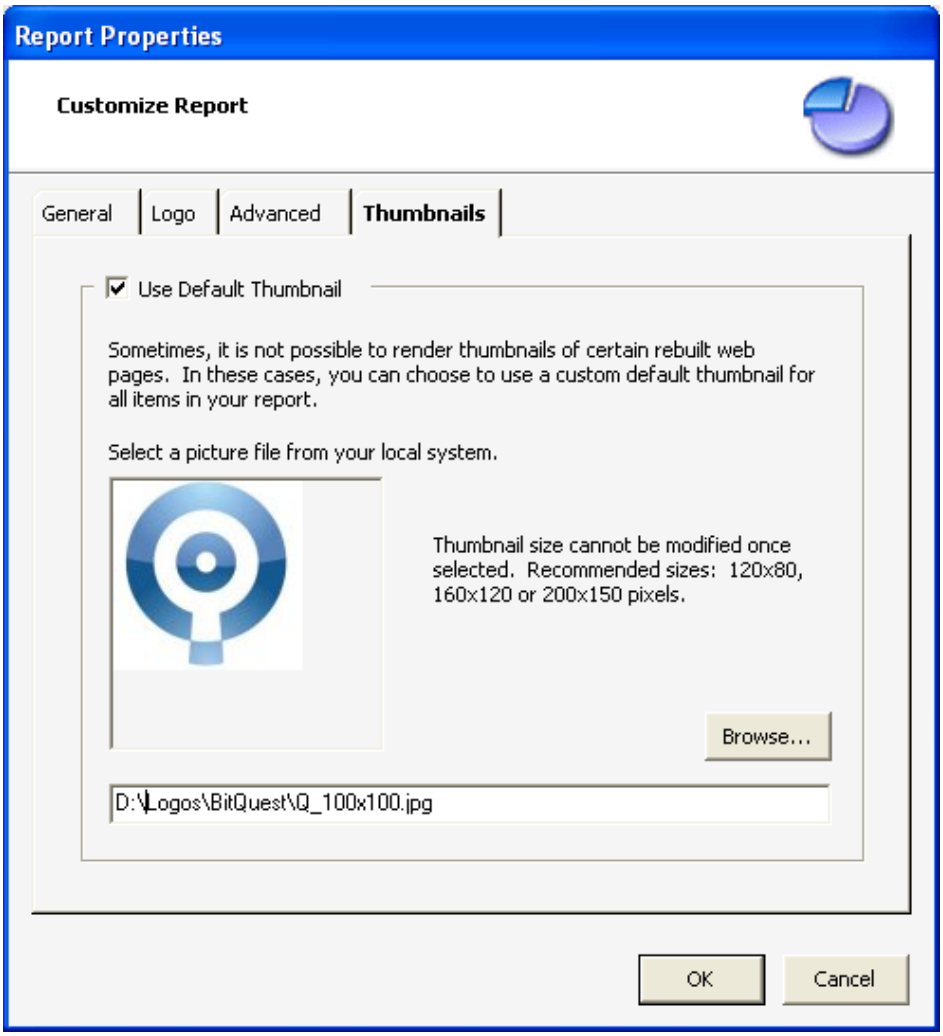
The Advanced Tab provides options to sort the records in the report in Ascending or Descending order. You can also show or hide extra metadata by the “Individual Record Settings” area (eg: Include Audit, Incude Notes).

Examiners can also *override* any selected Table or Gallery records by using the “Include ALL records in the report” option.

Thumbnail Tab

This feature is used to define a “default thumbnail” to be used in reports wherever a given URL record does not contain thumbnail. As discussed earlier in the manual, some script-enabled web pages may prevent a thumbnail from being generated for the individual URL record. This option can also be used in situations where a Crown prosecutor does not want the original evidence to be displayed right away.

Picture 1 – A user-defined “default thumbnail”.



Running The Report

Once you have gone through all of the available options on the Report Window, it's time to create the report. Click to "OK" button to run the report.

The following actions will take place:

1. A "Report" sub-folder will be created in the project's defined TEMP folder path

   Example: `C:\Temp\Report`

2. For each selected URL in the Table (or Gallery), CacheBack will do the following:

   a. **If the URL record is a *web page***, CacheBack will create another sub-folder inside the Report folder to hold the *ingredient files* (eg: pictures, javascripts, etc.) that make up the file.

      Example: `C:\Temp\Report\0000346.CBR`

      i. All ingredient files are copied (carved) out to the .CBR folder and given a 7-digit (zero padded) filename that represents each file's unique "URL_ID" (as stored in the project file).

         Example: `C:\Temp\Report\0000346.CBR\logo.jpg`

      ii. A thumbnail of the web page is also copied to the same folder and given the name of: "`~0000346.thumbnail.jpg`" (where 0000346 is equivalent to the web page's URL_ID).

      iii. The actual web page itself is also copied out to this folder and given the name of `<URL_ID>` + "`.htm`".

         Example: `C:\Temp\Report\0000346.CBR\0000346.htm`

         This naming convention (zeros + URL_ID) is consistent for all files that are extracted from a cache to .CBR sub-folders.

b. **If the URL record is a *picture***, CacheBack will create the "pictures" sub-folder inside the Report folder (if it does not already exist) and copy out the picture file. When the picture file is copied to the "pictures" folder, it is given a name that consists of a 7-digit number. This number is a zero-padded copy of the URL's "URL_ID" value. A file extension that matches the picture type is then appended to the file name. This process is repeated for ALL *picture files.*

Example: `C:\Temp\Report\pictures\0000346.jpg`

c. **If the URL record is anything other than a web page or picture file**, then CacheBack will still include the record in the report. However, there will be no file to copy out to the Report folder, or any other sub-folder.

3. When .CBR folders are created, the thumbnail for the *web page* is also copied to the .CBR folder (if a thumbnail exists) using a filename similar to that shown in 2(a) above.

4. When *picture files* are copied out to the `\Report\pictures` sub-folder, there are no thumbnails created in the folder. Since the pictures themselves are already *pictures*, the *original* pictures are used in place of the thumbnails instead. This is done by simply using a specifc *height and width* attribute in the <IMG> tag used to show the thumbnail.

5. **If one of the copied ingredient files happens to be a cascading stylesheet that contains *relative or fully qualified paths* to pictures (or other files such as javascripts), CacheBack will replace these paths with the name of valid *ingredient files that have already been copied out*. These cascading stylesheet "ingredient" files are given a special name in the output folder, as follows:**

Example: `\Report\0000346.CBR\_css0000001.png`

This extended "find and replace process" and special naming convention will ensure that stylesheet properties are applied correctly to the .HTM file (eg: `\Report\0000346.CBR\0000346.htm`).

6. **For Graphical Reports**: Once all ingredient files and .CBR folders have been created, a temporary report file is created using the file name: "**~report.htm**". This file is later renamed to "**index-**

cache.htm" when and if the report is Published (see page 168 for more information on Publishing reports).

For Tabular Reports: The same "**~report.htm**" temporary file name is used for a Tabular report. When this type of report is published, it is COPIED and RENAMED to "**index-history.htm**"

HYPERTEXT MARKUP LANGUAGE (HTML)

Understanding Tags and Attributes

So far we've discussed the steps required to rebuild one or more web pages from an Internet cache. We also mentioned that the original cache files are NOT modified. This is because CacheBack makes a COPY of the original file before making any alterations to the underlying source code.

Having said that, it is obvious at this point that *some* alteration of the original source code (through the use of a copy) must be made in order to rebuild or recreate the file.

We have discussed very briefly WHAT and HOW certain elements of a web page are rebuilt.

This section takes a quick peek into the actual source code for a web page BEFORE and AFTER it is rebuilt. This sheds some light on what conditions and/or elements must be present in the source code to be considered "something that has to be replaced".

Relative Paths vs. Fully Qualified Paths

- A *fully qualified path* is a complete URL path to a resource object.
  Example: http://www.mywebsite.com/images/mylogo.jpg

- A *relative path* is that portion of the URL path that continues past the location of the web page that is calling on the resource. For example, if the default web page for "mywebsite.com" is "index.htm", the <IMG> tag in side that file might refer to the "images/mylogo.jpg" path. In this context, "images" is a child folder in relation to the file "index.htm", and "mylogo.jpg" is a child item to "images".

List 1 - Sample HTML tags and attributes that are subject to replacement.

The following is a sample list of valid HTML tags and their attributes that are commonly sought out by CacheBack and subject to replacement during the rebuilding process. In most cases, tags have a specific *attribute* (property of the tag) that calls for a *relative* or *fully qualified file path*.

Examples:

1. **<body background=”images/checkers.jpg”>**
   The BODY tag uses the BACKGROUND attribute to specify a valid picture file.

2. **<img src=”images/logo.jpg”>**
   The IMAGE tag uses the SRC (aka : source) to specify a path to a valid picture file.

3. **<link rel="stylesheet" type="text/css" href="theme.css">**
   The LINK tag uses the HREF attribute to specify a path to a valid cascading stylesheet file.

The following is the same list as how it would appear AFTER the web page has been rebuilt with CacheBack. You will notice that each *attribute* has been substituted with a filename comprising of a zero-padded version of the URL_ID that was chosen as the appropriate replacement object.

List 2 - Replaced attributes.

1. **<body background=”0000001.jpg”>**

2. **<img src=”0000002.jpg”>**

3. **<link rel="stylesheet" type="text/css" href="0000003.css">**

NOTE: The fact that each filename in List 2 contains only a single file name, and no additional path information, is an indicator that these are *relative paths*.

VBScript and Javascript

When we talk about HTML content for a web page, we are usually referring to the different tags and formatting used to render content to the user onscreen. This type of content, when "hard coded" into a page's source code file, is very easy for CacheBack to examine. Replacing this type of content when Rebuilding Webpages is relatively straight forward.

There are times however, when web pages employ the use of the <SCRIPT> and </SCRIPT> tags to enclose a block of *executable code.* This type of code commonly comes in two flavours: Javascript and VBScript. VBScript is short for Visual Basic Script, a scripting language provided by Microsoft.

Wikipedia.org defines the above two terms as follows:

VBScript *(short for* ***Visual Basic Scripting Edition****) is an Active Scripting language, developed by Microsoft, which uses the Component Object Model to access elements of the environment within which it is running (e.g. FileSystemObject or FSO used to create, read, update and delete files). The language's syntax reflects its origins as a limited variation of Microsoft's Visual Basic programming language.*

JavaScript *is a scripting language used to enable programmatic access to objects within other applications. It is primarily used in the form of client-side JavaScript for the development of dynamic websites. JavaScript is a dialect of the ECMAScript standard and is characterized as a dynamic, weakly typed, prototype-based language with first-class functions. JavaScript was influenced by many languages and was designed to look like Java, but to be easier for non-programmers to work with.*

Issues with JavaScripts

When it comes to examining and rebuilding web pages using CacheBack, javascript can sometimes cause the following *issues*:

1. Web pages do not display properly or not at all inside the Browser viewer tab.
2. Web pages are not able to be rebuilt.

3. Annoying “script error” message boxes may appear onscreen when attempting to view a cached web page in the Browser tab OR during a page Rebuild.
4. A selected web page file may appear to “hang” while the page loads into the Browser viewer tab.

The following are *explanations* of what is happening in behind the scene that would give rise to the above problems:

1. The javascript code is making reference to an external resources (eg: calling another javascript file or picture or other object that is accessible remotely on the Internet).
2. A “timeout” threshold needs to be reached (when no Internet connection is detected) before a page can complete its rendering within the Browser viewer.
3. *Code instructions* are assembled at “run time” (when the page is actually called and downloaded). This approach is becoming more and more common. *Fragmented* individual code line statements are assembled only when the function or script tags (that contain the code) is referenced.

By default, CacheBack project files are initiated with the “Disable script tags in web pages” option set to “True”. Previous sections of this user manual discuss how to prevent popup messages. This section here simply points out the fact that CacheBack’s ability to render scripted content is dependent upon the complexity of the code, and the resources that it may attempt to connect to.

Best tests show that simply disabling the <SCRIPT> tags will eliminate any safety concerns (about connecting to the Internet) and still allow the web page to render properly (or adequately). Some items that may be missing, by this course of action, include special layout formatting and dynamic, database-provided information.

Cascading Stylesheets

A Cascading Stylesheet file is an external plain text file that typically ends with the extension “css”. A stylesheet is referenced by a web page using the <LINK> tag option which is usually enclosed by the <HEAD></HEAD> tag. Stylesheet coding can also be rendered “inline” which means that the *styling options* are interpreted within the HTML file itself using the <STYLE></STYLE> tags.

Stylesheets are extremely common and are used to provide site-wide formatting options which facilitates changes to website design (eg: fonts, pictures, layout, colors).

The following is a very simple idea of a stylesheet file named “global.css”

Example 1 – Anchor tag specifying a special font, color and size for hyperlinks.

```
a {             font-family:tahoma;
                color:#lBl569;
                font-size:l2px;
                text-decoration:none;
}
```

Example 1 – A special block of code that formats a “search” option on a website.

```
.adblock_search {   backgroun:url(images/block_search.gif) no-repeat left top;
                    background-color:#F4F4F4;
                    padding-left:20px;
                    padding-right:20px;
                    height:59px;
                    width:232px;
}
```

IMPORTANT: Cascading stylesheets can contain paths to embeddable objects too! As a result, CacheBack needs to be aware of any references to stylesheets that may exist in web pages that it is rebuilding. During the normal rebuilding process, embedded objects are identified and harvested to the .CBR (CacheBack Rebuilt folder). CacheBack goes a 2nd step and rebuilds cascading stylesheets too to ensure that stylesheet designs are preserved!

The Decision Logic Behind Tag Replacement

CacheBack's primary feature is the ability to rebuild web pages. In order to complete this heavy task, a series of issues need to be carefully understood and measured during each rebuild. As a forensic examiner, it may be essential for you to have a solid understanding of "how" CacheBack goes about the page rebuilding process.

This section provides a detailed, ordinal review of each step undertaken in the rebuilding process. As you will see, CacheBack does not "guess" at which tags are to be replaced, and what items to replace tags with. Instead, a logical approach is taken in the selection process. Where there is a possibility that a tag is replaced with an unsuitable tag, CacheBack reports this as an "AccuracyScore". This is the same thing as saying: "Out of *n* number of possibilities, CacheBack selected 1 item" (eg: 1:8). Usually an AccuracyScore comes into play when there are more cache files with the same name from which to choose (eg: "logo.jpg" or "spacer.gif").

STEP 1: IDENTIFY TAGS

A web page that is being rebuilt is first copied as a temp file (eg: C:\Temp\~tmp). The page content is then loaded into memory and then CacheBack searches for, and records, the following *tags and/or attributes*:

	ATTRIBUTE or STYLESHEET PROPERTY	TAGS USING THE ATTRIBUTE (most common listed)
1.	src	IMG (image), BODY, TABLE
2.	background	BODY, TABLE
3.	href	LINK, A (anchor), IMG * also used by style *methods* - Example: commonCss.setAttribute("href=",
4.	background:url	DIV, STYLE (inline) or as *property in stylesheets*
5.	url(	*used a* STYLE *parameter (scripting)*
6.	value=	FORM
7.	background-image:url	using STYLE (inline) or as *property in stylesheets*

STEP 2: REMOVE PARENT PATHS

A parent path is a programming shortcut to insert a reference to the first parent folder *in a relative path*.

For example, a *nested* web page on a website (eg: http://www.mysite.com/ads/showads.html) might have an IMG (image) tag that displays an *advertisement* using a .PNG picture file. Many websites tend to place all of their common file types into one single folder.

In the case of our example above, the "ads" (picture files) might be stored in the folder: http://www.mysite.com/images .

The IMG tag on the "showads.html" web page might use the fully qualified path the "topseller.png" file like this: http://www.mysite.com/images/topseller.png . While this is perfectly fine, many website developers will prefer to use a shorter form that we refer to as a *relative path.* To rewrite this path using a relative path, we also need to include the Parent Paths abbreviation like this:

../images/topseller.png

This has the same effect as saying "go up one folder level, then from there, go down to the 'images' sub-folder and inside of there, grab the 'topseller.png' file. The use relative paths and Parent Paths is extremely common.

In order for CacheBack to make use the *relative path* in any kind of "Find and Replace" routines later on, we need to strip away the ".." portion. The result that is then stored inside the CacheBack project file would be "/images/topseller.png".

STEP 3: REMOVE DUPLICATES

It makes sense that some web pages will make use of the same picture file throughout the page layout. An example of this might be something as simple as “spacer.gif”. This is a programmer’s trick to “pad” blank spaces to force a particular layout. A transparent, 1 x 1 pixel, picture file called “spacer.gif” has simply become *the norm* for this type of trick.

In order to work with the smallest, distinct list of tags, CacheBack removes any duplicate tags from its list (of discovered tags).

STEP 4: FIND TAG OPTIONS

For each *distinct* tag found in the web page, CacheBack then searches the entire project (case) file for *possible tag replacement options*.

Once we understand that the tags from Step 3 are mostly incomplete tags (eg: relative paths), then it makes sense that our next step is to try and match each *relative path* with a *fully qualified URL path*.

Example:

Tag in web page:	/images/***myuniquelogo.jpg***
Matching URL from cache:	http://www.company.com/images/***myuniquelogo.jpg***

There are a series of conditional statements that have to be processed and in a very particular order (of precedence) so that CacheBack picks the best possible option.

Here is that order of precedence.

1. Tag Presence Required

This simply means that CacheBack MUST locate one or more URLs, from the database (project file), which contain, in part, the *relative path tag (sans Parent Paths)*.

The sample shown above is an example of a valid *possible match.* When a partial match (as show above) is made, CacheBack adds the URL to a list of *replacement candidates*.

2. Secure Socket Layers (SSL)

As a priority, web pages (URLs) that begin with HTTPS (SSL) will need to find replacement items that ALSO begin with HTTPS.

A cache can comprise of many pictures (for example) that have the same file name. A perfect example of this is "**logo.jpg**".

Here are some sample cached URL entries in a possible CacheBack project (case) file:

(a) **http**://www.google.com/**images/logo.jpg**

(b) **https**://www.google.com/shopping_cart/**images/logo.jpg**

Let's suppose for a moment that a web page that is being rebuild by CacheBack has a cached URL of:

https://www.google.com/shopping_cart/**mycart.htm**

Inside this web page is a picture (<IMG> tag) that has a relative path of ***images/logo.jpg***.

Example: <IMG SRC="**images/logo.jpg**">

When CacheBack goes looking for the cached "picture" URL that matches the <IMG> tag value of "**`/images/logo.jpg`**", the TWO URLs shown above (items (a) and (b)) will BOTH be added to the *replacement candidates list*.

However, CacheBack will choose item (b) because it has the same HTTPS (SSL) protocol as the web page that contains the tag being replaced!

3. URLs Grouped By Host (Domain)

When CacheBack attempts to locate replacement URL *candidates*, those that have the same Host (domain) as the web page URL being rebuilt will take precedence.
If not match is found, then other URLs (from other Hosts/domains) will be collected into the list as well.

4. When All Else Fails (Manual Selection) *

* We explained earlier that CacheBack does not "guess" at what cache items to use to replace tags in a web page. This still holds true, however, being that this step would be the *last option* to choose, it is the closest behavior to a guess.

For this very reason, CacheBack uses the AccuracyScore column (in the Audit Tab, in the Viewer Pane) to reveal what the number of choices were to pick from.

AccuracyScores that are rather high (eg: 1:25) usually indicate a common file that might be used among web pages for special formatting or layouts. They can also indicate the repeated use of a logo.

IMPORTANT: If the examiner has any doubt about the "match" made where an AccuracyScore is high, it is strongly recommended that a manual validation process be undertaken. CacheBack provides enough metadata and search options to help this process. In the rare event that an examiner chooses a picture file (as an example) over the choice made by CacheBack, then the new selection would have to be copied out manually.

The following explains how to do this.

a. *In the* **\Report\000000n.CBR** *folder, simply rename the old (unwanted picture file) from "***000000n.jpg***" (as an example filename) to "***_000000n.jpg***" to show that it has been replaced. You will want to keep the original file handy for continuity purposes.*

b. *Next, use the Picture Tab (in the Viewer Pane) in conjunction with the Table (in the Data Pane) to:*

   i. *Locate the specific picture that is desired,*
   ii. *Preview the picture in the Picture Tab,*
   iii. *Use the "***Send To:***" + "***File***" options to save the file to the* **\Report\000000n.CBR** *folder.*
   iv. *Give the file the SAME NAME as the one that is being substituted: "***000000n.jpg***"*
   v. *Write a note in your examination log that explains what actions you took and why.*

USER INTERFACE: ADVANCED

Table Tab

Column Headers

This section will review each of the columns available in the Table, which is found inside the Data Pane.

Perhaps one of the first features that needs to be discussed in this section is the topic of "sorting by columns". CacheBack provides the ability to quickly sort the contents of the Table by simply clicking in the header portion (caption) at the top of each column. Doing so will sort the Table using the select column as the *sort column*, defaulting to an *ascending order.* Clicking on the same header a 2nd time will conduct a *reverse-sort (eg: descending order).*

NOTE: Some columns, such as the Icon column, cannot be sorted using the column header.

The following is an alphabetically sorted list of each column in the Table with a brief description about the significance behind each.

Action	A *verb* that indicates what type of activity was associated to the current URL. The most common value for Action is "Visited". NOTE: A value of "Loaded" is a metadata value that is unique to Opera. In future releases of CacheBack, expect further options for this column (eg: "Bookmarked", "Downloaded", "Shared", "Aborted").
Action Date (Local)	This column is assigned a *date* value that is result of the *ActionDateUTC* column being converted to the active Time Zone setting in CacheBack. This value, when it is being *displayed or reported* within CacheBack, *will change* according to the active Time Zone setting. However, the value of *ActionDateLocal* "stored inside the project file" will NOT change as it is

	assigned a value only ONCE, during the data Importing process.
Action Date UTC	A *date* value that is stored in Universal Coordinated Time (or sometimes referred to as GMT Time). NOTE: Since UTC time is considered the most accurate time format, some forensic examiners (organizations) will choose to report strictly in UTC.
BasePath	This is the root drive letter of the URL record's host file (eg: the index.dat file for Internet Explorer). The purpose of this value is to provide examiners the ability to inform CacheBack where the original cache files reside. This option became important when the drive letters on an examiner's workstation would change AFTER cache files were imported. *BasePath* can be changed via the Options Window which is accessed from the View menu (above the Toolbar). Once changed, ALL URLs in the project (case) file will be updated with the new value. NOTE: A value of *Orphaned* in the *Status* column is a clear indicator that the original source files cannot be located by CacheBack (likely due to a thumbdrive or removable being added or removed from the system). Changing the value of *BasePath* to the correct root drive letter will fix this problem.
(Cache) File Size	A numerical value representing the size of a cache file, in bytes.
(Cache) Folder	The name of the *parent folder* that contains the cache file associated to the current URL record. This value is associated to cache artifacts only.
Cache Root Path	This is the fully qualified path to the *parent folder* that contains the cache files. For example, Internet Explorer stores its cache in a folder called Content.IE5 which is found inside the Temporary Internet Files. The actual "cache" is comprised of the "index.dat" file and the immediate children folders used to store the cache content. In this case, the *CacheRootPath* might look like this: **C:\Documents and Settings\John Doe\Local Settings\Temporary Internet**

	Files\Content.IE5
(Cache) Type	An *icon* (logo, picture) used to depict the type of data source for the current URL.
Category	A text value that groups URLs by similar file type. This applies to both *history and cache* URLs, despite the fact that History URLs do not point to (or relate to) any existing cache files. This association can be made by inference that at some point in time, the History URL did in fact point to an actual file *that fit the category*. Examples: Pictures, HTML.
Data Block (File)	Contains the unique "block file" name which contains the metadata associated to the current URL. Typically, this column applies to cache URLs from Firefox2, Firefox3, and Google Chrome.
(Data) Block (File) Offset	A numerical value that represents the *logical file offset to the start of a block of data* (metadata) associated to the current URL. A value in this column is typically associated to cache URLs from Firefox2, Firefox3 and Google Chrome.
Data Source	A *text* value indicating the browser and *major* build number for the current URL. Example: *Firefox 3.0*
Date Other	Most, if not all, URL records contain datestamps that usually indicate when a particular URL was visited. Some browsers maintain more than one datestamp and provide metadata that explains the purpose of the additional datestamp(s). As a result, CacheBack captures at least one of these extra datestamps here.
Date Other Meaning	A simple text value that describes the meaning of the *DateOther* value, if present.

Expiry Date	Some browsers record an expiration date for cache URLs. This date value typically informs the system when to refresh the cache content for a given URL. It is also used for housekeeping purposes when the system has to decide about which cache items can be purged to make room for new content.
File Created	A *date* value representing when a cache file was created on the source volume. NOTE: This value is only helpful if the cache (from which the current URL was parsed) was imported / parsed in its original location (eg: a subject's hard drive). If a data collection tool was used to copy evidence to a target folder (eg: CacheGrab) where the evidence was then scanned and imported into CacheBack, then this date will reflect the date and time that the COPY was created. NOTE: A future release of CacheGrab will address this issue by logging the original metadata and making it available to CacheBack.
File Exists	A *Boolean* (true or false) value that indicates if a cache file exists. This value is determined during the import of cache files.
File Ext	A text value that is parsed from the current URL that indicates the associated file extension. If the URL does not contain a known file extension or no file extension exists, and provided the URL refers to a cache item, then CacheBack will inspect the cache file for known *file signatures*.
File Last Accessed	A *date* value representing when a cached file was last accessed on the source volume. NOTE: This value is accurate ONLY when the cached file (URL) was imported / scanned into CacheBack from its original source. If the cached file was copied to a new location prior to being imported, then this datestamp will reflect the date and time that the file was COPIED.
File Relative Path	A *text* value that contains the complete file path to a cached file *but without* the *drive letter* (or BasePath). This column is used in conjunction

	with the BasePath setting to help CacheBack find the original cache files. This becomes necessary when drive letters on an examiner's workstation may change AFTER the original cache has been imported into CacheBack. For example: "C:\Temp\MyLogo.jpg" would be stored in this column as "Temp\MyLogo.jpg".
GZip (Encoded)	A *Boolean* value (True or False) that indicates if the cache data (associated to a particular URL) is encoded in the Gzip format. This is usually only found in Firefox2, Firefox3 and Google Chrome cache *data* files.
(History) Version	A *numeric* value that indicates the version of the history file or the cache index file that was used to store the current URL. This value should not be associated *strictly* with the Build Version of a given browser program. While the *HistoryVersion* will in most cases refer to the *major and minor* version of the browser (eg: 2.0, 2.5), it refers to something different for Internet Explorer. Internet Explorer Version 8 is the latest release for Microsoft's Internet browser. However, the *code implementation, that manages the cache and history storage for IE through the use of index.dat files,* has its own versioning information. NOTE: Internet Explorer's index.dat files used to map / store History, Cache and Cookies has not changed since IE Version 5.2. In fact, IE (the browser) Versions 5, 6, 7 and 8 all use *cache build version 5.2*. For all other browsers, *HistoryVersion* means simply the *major number portion of the* browser build number. Example: Firefox 3.5 would be stored as "3.0". CacheBack determines this value by looking at the History or Cache file (name, location, datatype, etc.) that is being examined.
Host	A *text* value that represents the domain name portion of a URL.

	Example: The host name for “http://www.google.com” would be “google.com”.
HTTP Metadata	A *text* value that contains the HTTP Response metadata that may be associated to a cache URL. This may include a error or success codes as well as what type of data was being referenced (eg: image/jpeg, text/ascii).
Icon	A simple picture used by CacheBack to best describe, visually, the contents or type of file being referenced by a given URL. The *FileExt* value is one means that CacheBack uses to assign a picture (icon) to this column. NOTE: A globe typically represents a web page or a file containing HTML content. Any URL that cannot be associated to a known file type or content type will be assigned a default “chainlink” icon.
Links	A *numeric* value that represents the number of possible *relationships or associations* to other URLs within the same project (case) file. The Links column will contain a value ONLY AFTER the Link Analysis feature is launched, provided a relationship with one or more other URLs has been established. NOTE: Link Analysis can only be effective once all files have been imported and parsed.
Map File Offset	A *numeric* value representing the logical file offset to the map entry point in a cache *index* file. This value would correspond to the actual record offsets in an index.dat file for IE. For Firefox2 and Firefox3, this would be the map record offset for the _CACHE_MAP_ file. For Google Chrome, this would be the map record offset for the *index* file.
MIME Type	A *text* value that describes the type of data that the URL refers to (eg: image/jpeg, text/ascii). NOTE: The data stored by this column is seldom captured by most browsers.
PARD	An acronym that stands for “Photograph Aspect Ratio Differential”. This is a *numeric* value that represents the difference, in percentage points,

	between the height and width of a picture. The PARD value is used to automatically categorize photographs (picture files) based on their dimensions. NOTE: The Photographic Image Aspect Ratio Theory is the concept upon which the PARD value was derived. Their combined usage within CacheBack help investigators *dramatically* expedite the categorization and examination of photographs and pictures.
(Picture) Height	A *numeric* value representing the height of a picture file, in pixels.
(Picture) Width	A *numeric* value representing the width of a picture file, in pixels.
Rebuilt	A *Boolean* value (Yes or No) that indicates if a cached file was successfully rebuilt by CacheBack. By default, all URLs are initially set to "No". As files are rebuilt, the column value will change to a RED colored "Yes" (if successful).
Rebuilt Method	A *text* value that indicates if a cache file was rebuilt automatically by CacheBack, or using a Template library. The topic of Templates is excluded from this manual as this feature is being re-introduced in a future dot release of the software.
Status	A *text* value indicating the "state" or "form" taken by the URL. For Internet Explorer URLs, this will often simply be a value of "URL". IE supports other Status values such as LEAK, FILE and REDR. For Opera, a URL that was successfully loaded into a browser would be recorded as "Loaded". For most other browsers, the *Status* column will be left blank. NOTE: This may change in future versions to offer or present a default behavior of the browsers (eg: "URL").
Source History File	A *text* value representing the complete file path to the History or Cache index file that was imported / parsed by CacheBack.

	Example: C:\Temp\Index.dat
Time Added	A *date* value indicating the exact date and time that a given URL was added to the CacheBack Project (case) file.
Time Rebuilt	A *date* value indicating the exact date and time that a cache file was *last* rebuilt using CacheBack.
URL File Name	A *text* value that indicates the *name portion only* for a filename that was parsed from a URL. A file does not have to exist in order for this column to contain a value. Since URLs from both a History or a Cache point to a filename of some sort, this value can be parsed directly from the URL.
URL MD5 Hash	A *32 character (text / string)* value representing the unique MD5 hash digest for a given URL. This value is used (a) for comparison purposes, and (b) to circumvent syntactical issues that arise when using URLs in query statements, which contain invalid (SQL reserved) characters.
URL Raw Value	The original, unparsed / unformatted version of a URL that has been parsed from an Internet History (or Cache). This column addresses special metadata found in Internet Explorer URLs derived from History index.dat files. For all other browsers, the URLRawValue will not contain any unusual formatting as seen in the below example. Example: **Visited: Examiner1@http://www.cacheback.ca** In the above example, “Visited: Examiner@” is pre-pended to URLs that are stored by Internet Explorer’s Master index.dat file.
URL Variables	A *text* value containing the form variables in a URL expression that are provided to the browser as part of a “form submission” (such as a search engine query).

	For example, in the following URL: **http://www.google.ca/search?hl=en&q=hotmail&aq=f&oq=** …the values that follow the question mark would constitute the *variables*: • hl=en • q=hotmail • aq=f • oq= Some of the obvious variables may be interpreted in plain English as follows: • home language = English • query term = hotmail TIP: CacheBack provides the ability to create a *keyword list* from these variables and export them to disk as a plain text file. To do this, simply right-mouse-click inside the Table which is found in the Data Pane. From the context (popup) menu that will appear, select "**Create Keyword List…**".
Web Page Title	A *text* value containing the value of the <TITLE></TITLE> tags found inside a cached web page. This is the text that appears in the browser window's title bar when visiting a website (web page) that indeed contains a title. NOTE: Most websites and web pages contain titles.

Resizing, Hiding and Moving Columns

When working within the Table of the Data Pane, it is possible to resize, move and hide different columns.

RESIZING COLUMNS

To resize a column, simply hover the mouse overtop of the outer edge of any given column until the mouse pointer changes to a splitter character (looks like an uppercase letter "I"). This will signal that you are now able to resize the column. To continue with the resizing, depress and hold down the left mouse button while sliding (moving) the mouse in the direction desired (to either shrink or enlarge the column).

MOVING COLUMNS

To move a column, simply depress the left mouse button inside of any particular column header, hold the button down, and then drag the column to the left, or to the right. As you move the mouse, you will see a dark, vertical line appear inside the Table. This line indicates, at which position, the column being moved will be placed (if or once you let go of the mouse button).

NOTE: Column positions will *persist* (be remembered by CacheBack) after you close the program. This makes it easier to restore your preferred layout the next time you open a particular CacheBack Project (case) file.

A diagram on the following page illustrates this point.

Diagram 1 – ActionDate*Local* column in the middle of being moved to the left.

In the following screenshot, the ActionDate*Local* column *(depicted below with the black column header background),* is in the process of being moved in between the Status and the Action columns. Unfortunately, our picture below does not reveal the mousepointer which happens to be immediately above the dark, vertical line (to the left of the Action column).

BEFORE

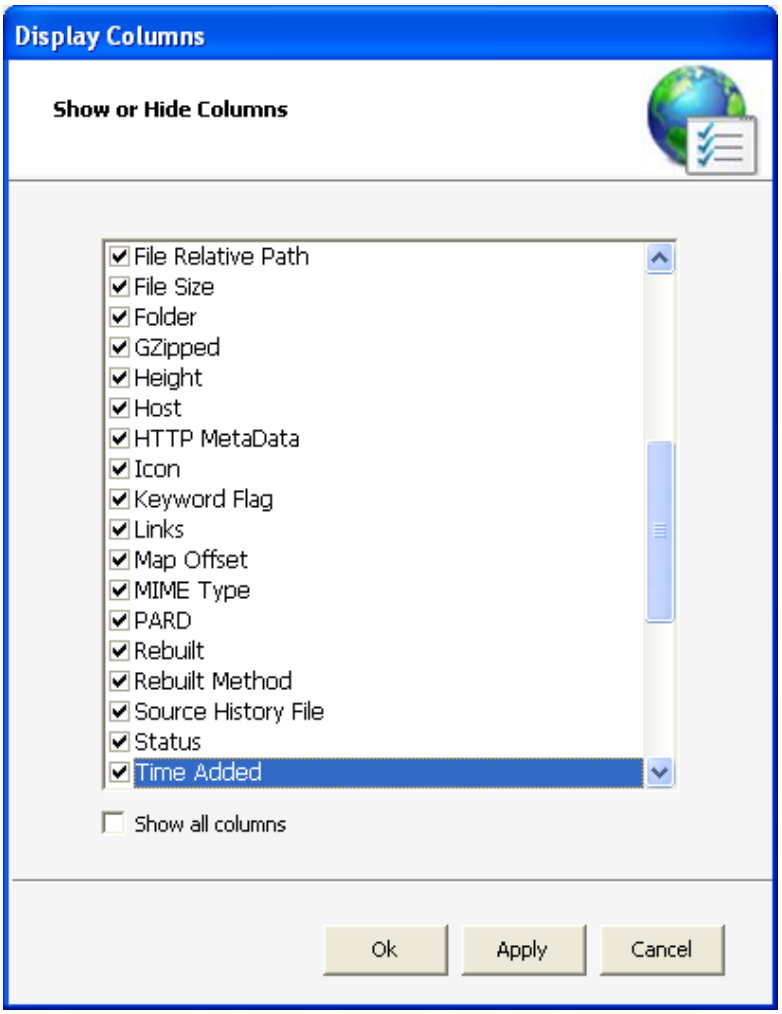
xists	Status	Action	Action Date [UTC]	Action Date [UTC +0200]
o	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
o	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD
es	URL	Visited	2009-06-02 05:13:06	2009-06-02 07:13:06 STD

AFTER

Status	Action Date [UTC +0200]	Action	Action Date [UTC]
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06
URL	2009-06-02 07:13:06 STD	Visited	2009-06-02 05:13:06

HIDING COLUMNS

To hide or show columns, go to the View menu and select “Columns…” to display the Columns Window. Hiding and showing columns is then simply a matter of placing a checkmark next to the names in the list, of those columns that you want to “show”, as shown below.



Icons

CacheBack utilizes a number of different icons to provide visual feedback about the type of data being referenced. Icons exist in a couple of different places within the program, the most noticeable being in the Table, which is located in the Data Pane.

Icons are also found in Reports that are created in the Tabular (grid) format, typically for reporting Internet *History* activity.

The following chart provides a legend for the some of the different icons that are used within the program. While this is not a complete list, it presents some of the most common icons that users will come across. Please note that some icons below vary slightly in appearance (resolution) for the purpose of clarifying their meaning.

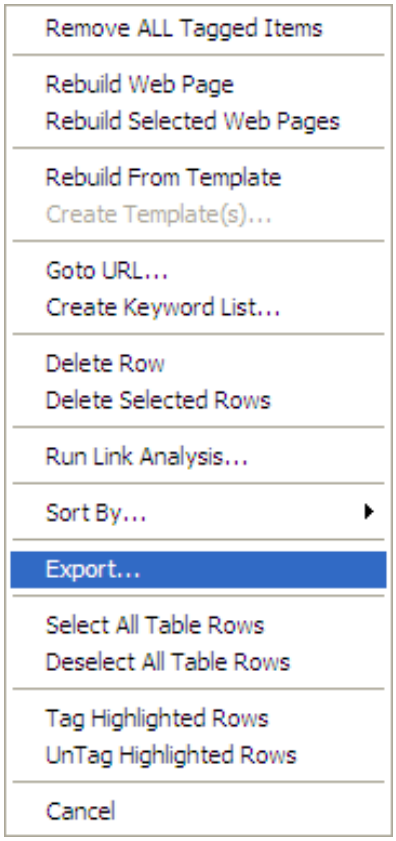
	Indicates that the cache file is a web page and/or contains HTML		Google Chrome browser artifact
	Picture (.jpg, .bmp, .gif, .png)		Opera browser artifact
	Multimedia file (.mov, .wmv, .avi, .mpg...)		Safari browser artifact
	Default icon used to represent any URL not classified by another icon		Mozilla Firefox 3 browser artifact
	Javascript		Internet Explorer browser artifact

Exporting Records

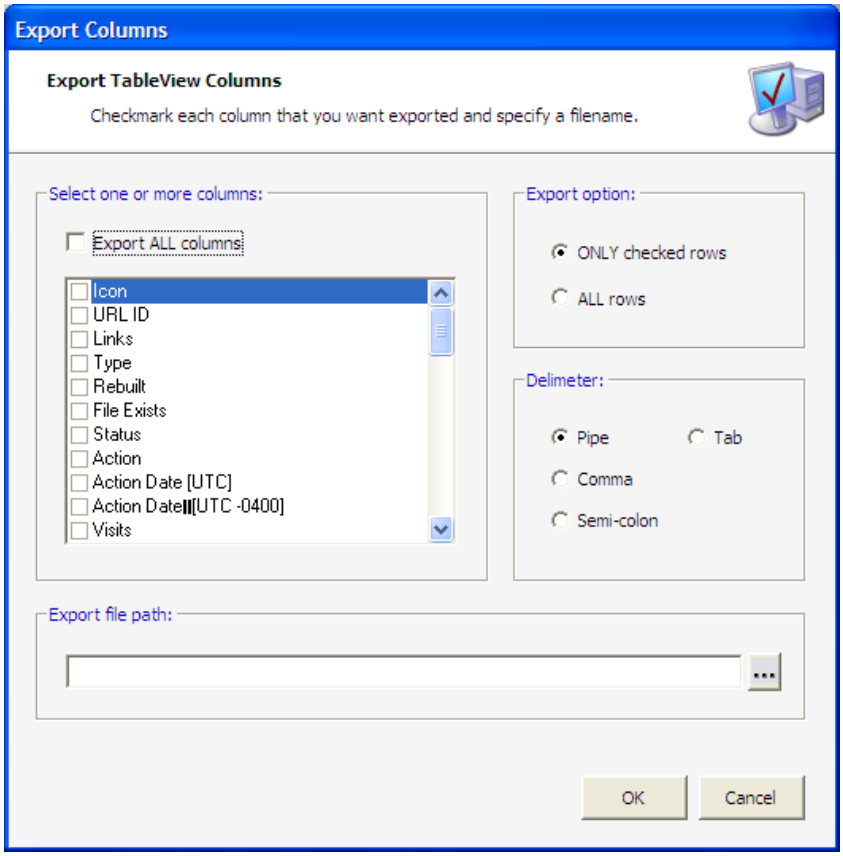
Examiners who are examining records within CacheBack, at some point, may wish to extract URL records from CacheBack to a disk file so that the data can be imported into another program. This is often the case when data is needed to be incorporated into another report format, such as an Excel spreadsheet or Word document.

CacheBack accommodates this task by providing an Export function that harvests selected URLs (Table records) to a plain text (eg: comma separated value) file. To access the export feature, simply right-mouse-click anywhere inside the Table to invoke the context (popup) menu. From the menu, select the "Export ..." option to invoke the Export Window as shown below.

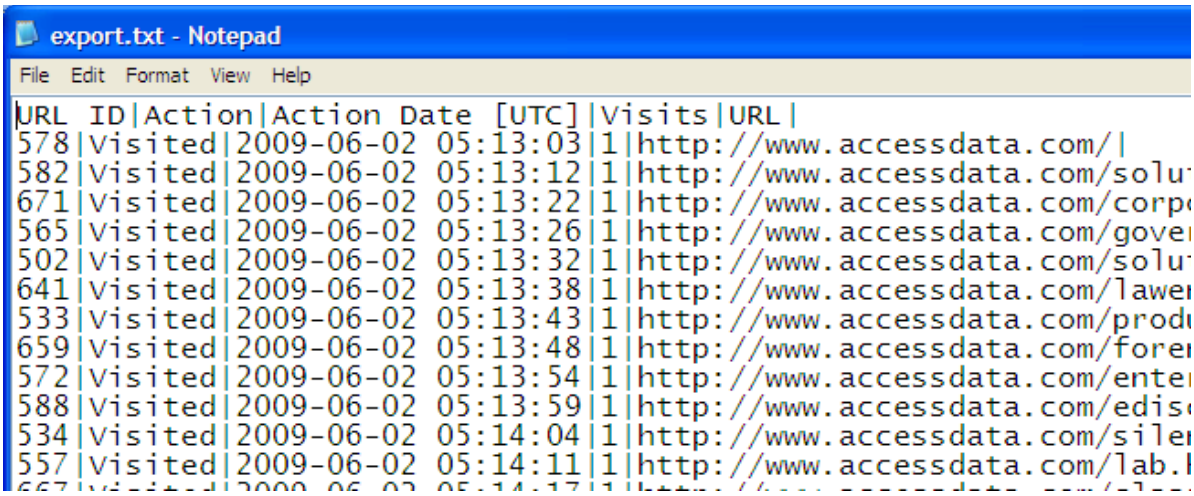
Step 1 - Right-mouse-click in the Table to invoke the context (popup) menu.



Step 2 - Choose specific columns and options to being the export.



The following is a sample of records that have been exported using a PIPE character as a column delimiter.



Tagging Records

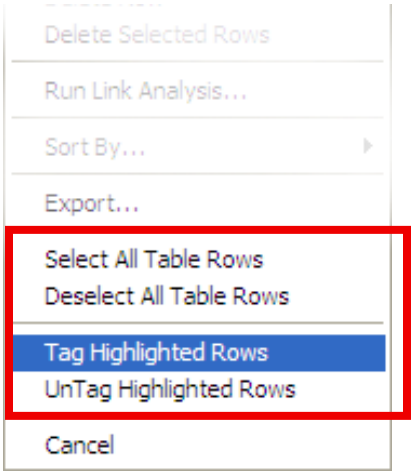
In order to use many of the features within CacheBack, examiners must first select one or more URL records from the dataset current displayed. These records are shown either in the Table or the Gallery, which are both found in the Data Pane.

The *selection* process requires that a checkmark be placed in the checkbox situated next to each URL record. For URL records in the Table, the checkbox appears in the column farthest to the left. For the Gallery, a checkbox appears immediately below each record.

There are three ways to select records. The first method is by clicking inside the individual checkboxes, next to select URL records (rows), using the mouse. This approach is ideal if the number of records to be selected is limited.

The second method is to use the context (popup) menu's "**Select All Table Rows**" or "**Deselect All Table Rows**" options. To invoke the menu, simply right-mouse-click anywhere inside the Table or the Gallery.

The third and last method requires two separate steps and relates only to the use of the Table. Part one requires the use of the keyboard, in combination with the **SHIFT** + **DOWN/UP** arrows to "highlight" contiguous rows. Once a range of rows have been highlighted, click the right-mouse-button inside the Table to invoke the context (popup) menu. Once the menu appears, select the "**Tag Highlighted Rows**" or the "**UnTag Highlighted Rows**", as required.



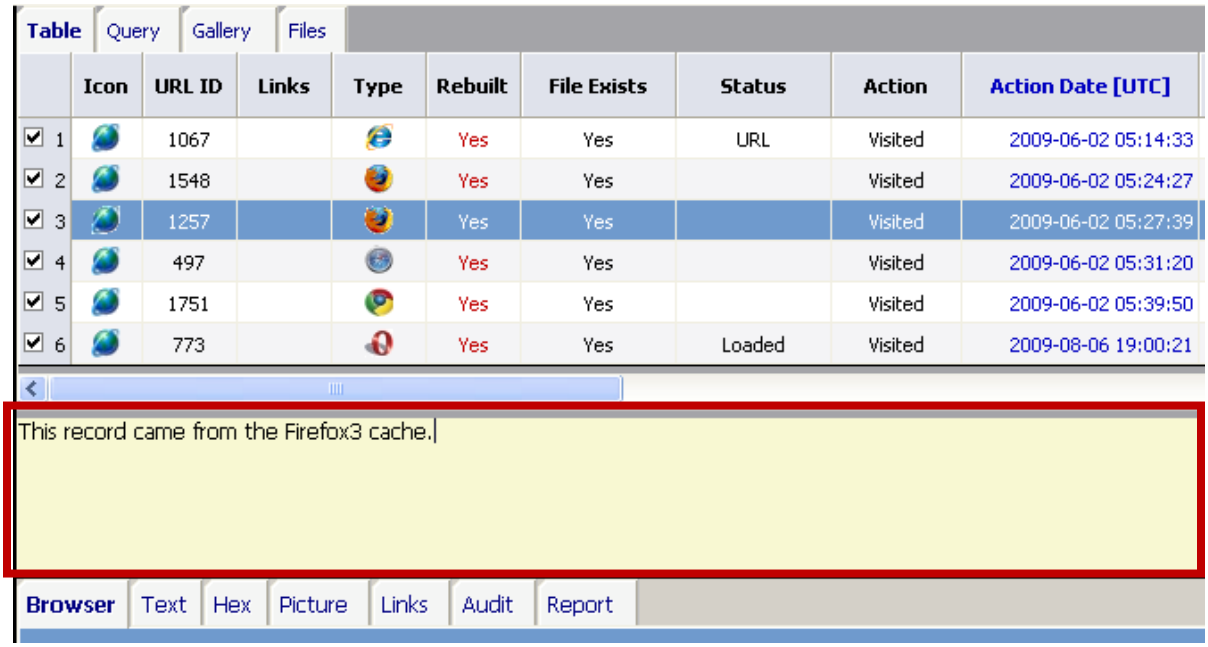
NOTE: Tagged records will persist in the project file until they are "untagged". This has no real negative impact on the use of the program. However, in a situation where there are hundreds or thousands of records being displayed inside the Table, examiners must be mindful of tagged records that exist outside of the immediate viewing area.

TIP: A sure way to quickly remove "untag" records is to the use the "Deselect All Table Rows" option from the context menu (as discussed earlier).

Creating Notes

Examiners may, at times, wish to make certain notes or comments about a given URL. CacheBack accommodates this need using the Notes text box, which is only available on the Table Tab, in the Data Pane. The Notes text box is made visible (or not visible) via the View menu (about the Toolbar) and then selecting "Notes". Examiners can type freely into the box. Once the program's focus leaves the Notes box (eg: selecting another URL from the Table), the notes are saved automatically.

Picture 1 – The Notes text box.



Gallery Tab

While we introduced the Gallery earlier in the user manual, we really haven't spent a whole lot of time discussing the features in any great detail. The following sections will do just that and provide you with a much clearer understanding about what type of data is displayed in the Gallery, and some of the benefits of viewing data using the Gallery, as opposed to the Table.

Supported File Types

The first thing we need to understand is that the Gallery is made up of "thumbnail" versions of the original data. Data in this context will usually include a *web page* or a *picture file*. Sometimes, javascript and XML files will be displayed here too and this is simply an exception. Usually when that happens, it is due to an HTML tag (eg: <BODY> or <HTML>) being detected within the file during the initial Importing of cache files into CacheBack.

Since the early release of Version 2.7, CacheBack now supports BOTH *html file content* and *actual pictures*. Originally, the Gallery was used to display thumbnail versions of web pages found inside the imported cache. However, examiners expressed an interest to use the Gallery to view the "pictures" found in the cache – preferably as a first investigative tactic.

Generally, the Gallery supports thumbnail creation for the following types of files, expressed here using their known file extensions for simplicity: .JPG, .BMP, .PNG, .GIF, .HTML and .ASP.

Thumbnails are created twice during an examination. The first time is when cache files (and pictures) are first imported from a browser's cached data. The second time (and each and everytime after that) occurs immediately after a web page is rebuilt. This is to ensure that the Gallery library of thumbnails is kept up to date. Thumbnails are stored as BLOB (binary data types) inside the project (case) file.

Also, you notice that the name associated to each individual thumbnail in the Gallery is a numerical value followed by the .JPG file extension. This "name" is simply a zero-padded copy of the URL_ID (as displayed in the Table). This uniquely identifies each thumbnail and corresponds directly to specific URL records in the project (case) file. The .JPG file extension is generated by CacheBack and has no other meaning except that it is the required format to render thumbnails in the Gallery.

Image 1 – The Gallery (before pages are rebuilt)

Notice that the thumbnails' filenames are actually the URL_IDs for the individual records. The numbers in this case are not *in ascending order* because the query that controls the data is sorted (in this example) using the *ActionDateLocal* column.

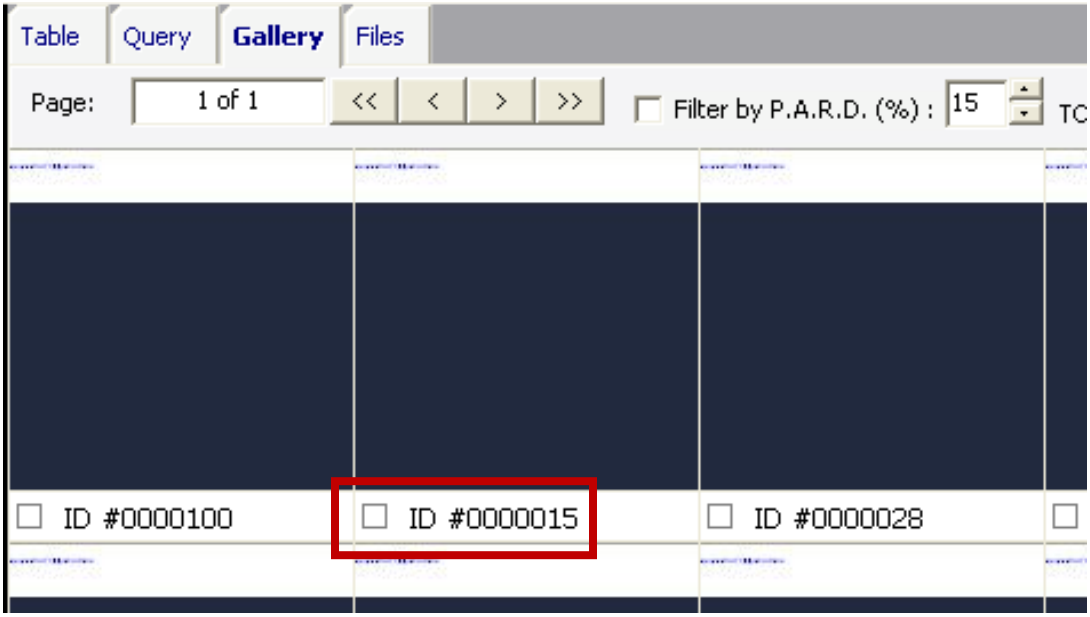
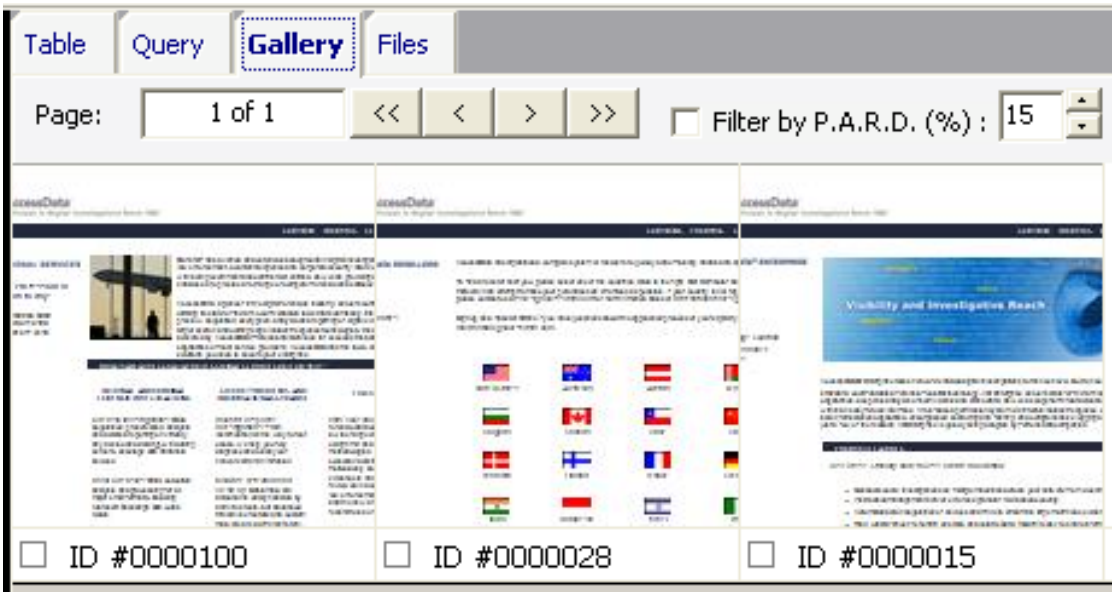


Image 2 - The Gallery (after pages are rebuilt)



Where the Data Comes From

The information that is displayed inside the Gallery is derived directly from the results of the *active or current quer,* which is defined inside the SQL View of the Query Tab. Therefore, the records in the Gallery should correspond to the same records displayed in the Table, from the Table Tab. This includes any results that have been filtered using the Filters or Host Filtering options.

NOTE: An extra layer of filtering is introduced on the Gallery tab with the P.A.R.D. value and picture size checkboxes. This will be discussed in the next section of the manual.

NOTE: When CacheBack attempts to generate a thumbnail for a particular cache file, sometimes this is not possible to create the thumbnail due to the file's contents (eg: scripts). Each cache file (web page or picture) has to be loaded into memory. By this measure, any scripts embedded in the file will be "blocked" and therefore may prevent the page from rendering (loading). When this happens, CacheBack will simply omit the thumbnail for that particular file.

Advanced Filtering Using P.A.R.D.

Before we being this next section, the very first question we need to ask is "What is P.A.R.D.?".

Here is a rather technical explanation for those examiners wanting a more technical answer. A more simplified discussion / explanation follows below.

P.A.R.D. stands for "Photograph Aspect Ratio Differential", a unique percentage value that is used to filter out pictures. A PARD value represents the difference between 100% AND the percentage (out of a 100) of the shorter of the two sides of a picture (height vs. width), compared directly to the longer of the two sides (height vs. width), where the longer side represents 100%. For example, a picture that measures 100 pixels wide, and 80 pixels high would have a PARD value of 20. A picture that measures 100 x 75 would have a PARD value of 25.

The PARD value provides a gauge about how "square" or "rectangular" a picture file is, regardless of whether it is a landscaped, or portrait. After careful study of the most commonly found sizes of printed photographs (eg: 4x6, 5x7, 8x11), an association can be made between the PARD value, and the likelihood that a computer image (picture file) conforms to the dimensions of a photograph. Using this approach, it is then possible to identify potential photographs on a computer with little human intervention to make that decision. This flexibility provides exponential processing power to pre-categorize images thereby reducing the overal number of images needed to be reviewed by a forensic examiner.

The PARD value was built upon the *Photographic Image Aspect Ratio Theory (PIART or PART for short),* which were both developed by the lead developer at BitQuest. While the PARD concept is relatively new, the theory itself dates back to the Fall of 2003.

Q. Why is PIART and PARD relevant to computer forensics?

A. Simple. It resolves a major problem faced by examiners who have to categorize or filter pictures based on content (eg: is it a photograph? is it a website ad/banner?). This is the case for most investigators of child exploitation related offences. Since these types of investigations can often yield hundreds of thousands of pictures, there needs to be a faster way to expose only those images that are of forensic interest, and put aside those that are not.

Q. Why is this relevant to CacheBack and Internet browser forensics?

A. CacheBack has the capability of dealing with large volumes of cache data which contains pictures. In extra-ordinarily large projects (cases), time may be of the essence and so the same concerns apply: Photographs need to be weeded out as accurately and quickly as possible.

Q. How does a PARD value factor into an analysis using CacheBack?

A. The Gallery tab provides an immediate filter option that will display only those thumbnails that fall within a define range of PARD values. Case studies have revealed that a PARD value range of 15-40 is optimal. Increasing or descreasing this range of values only moves the *match criteria* further and further away from know photograph sizes (dimensions).

PARD values can also be used to define custom queries using the *Query Builder* found on the Query Tab, in the Data Pane.

Q. What kind of results can I expect using PARD in my analysis?

A. Results vary from case to case, but in most situations, the number of images that fall within the PARD range will represent only a fraction of the total number of images available for analysis.

In a real case involving approx. 250,000 images on a single hard drive using Windows, the PARD filtering technique was able to reduce the examinable quantity to approx. 18,000. Of those results, the images were then grouped into categories of X-SM, SM, MED, LRG, X-LRG.

In cases involving child exploitation related offences, the groups for LRG and X-LRG tended to yield the most valuable evidence. This further reduced the examinable quantity to less than 5,000.

Q. What is the speed or performance that results from using PARD?

A. Most forensic examiners who investigate CE/CP related offences MUST sift through all images and label them into one of several examiner-defined categories. This can literally take days and sometimes weeks to finish. CacheBack's use of PARD can reduce this time down to a matter of hours in most cases. This is because the pre-categorization ability has a very high degree of accuracy. Images that fall outside of the default PARD value, in most cases, tend not to be of forensic value when it comes to CE/CP type investigations. This translates into a dramatic increase in case processing capabilties, decreased reporting times, and decreased manpower hours per case.

The following pictures illustrate the application of PARD values to filter out images that do not conform to standard photograph sizes.

Picture 1 – A typical banner image found inside a cache folder.

Banner advertisements are very common in web pages and obviously would not conform to the dimensions of a standard photograh as demonstrated below. In this case, the image measures 763 x 88 pixels.

To calculate the PARD value for this file, we use the following equation:

PARD = 100 – ((short side * 100) / (longer side))

PARD = 100 – ((88 * 100) / (763))

PARD = 100 – (8800 / 763)

PARD = 100 – (12) NOTE: We round down to the nearest integer.

PARD = 88

The PARD value noted above clearly falls outside of the recommended 15-40 range. Therefore, it would be excluded from the results.

Picture 2 – Here is a photograph quality image found also inside a cache folder.

This image measures 500 x 332 pixels. The PARD value is calculated as follows:

PARD = 100 – ((short side * 100) / (longer side))

PARD = 100 – ((322 * 100) / (500))

PARD = 100 – (32200 / 500)

PARD = 100 – (64) NOTE: We round down to the nearest integer.

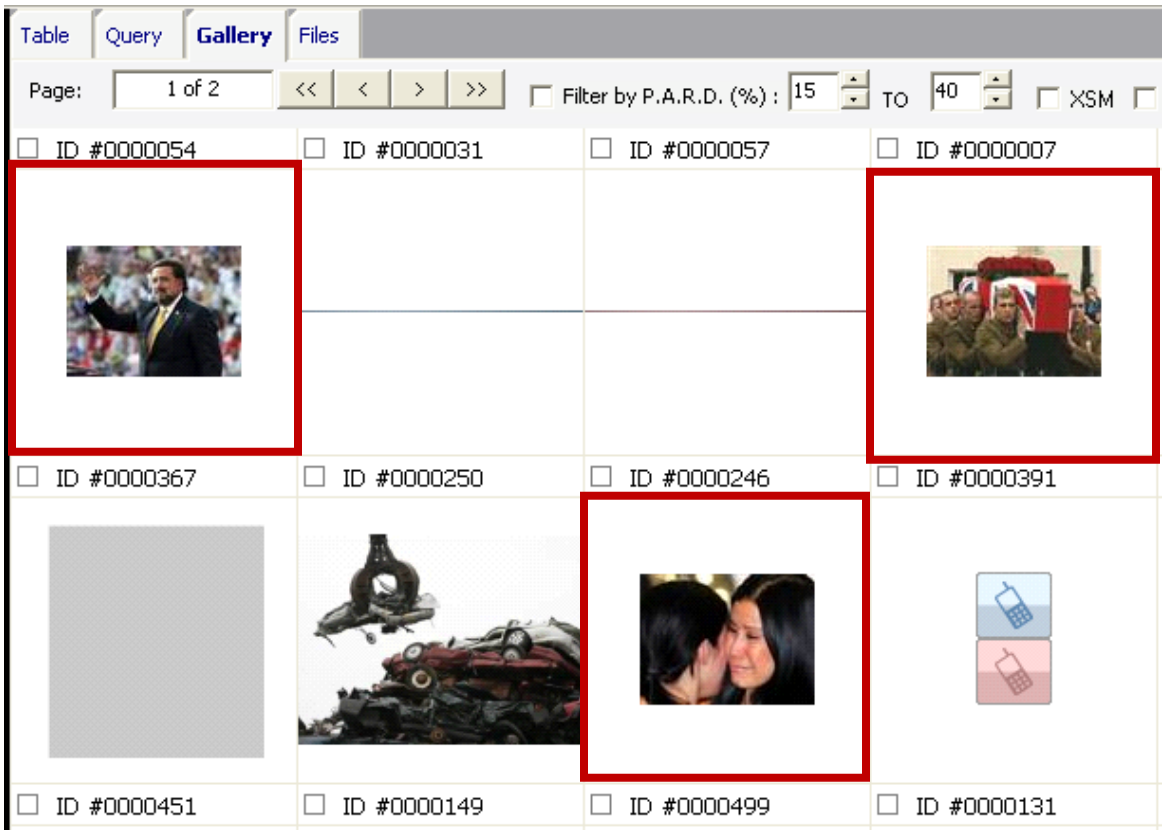
PARD = 36

As you can see here, the PARD value is 36 and therefore fits easily within our default range of 15-40. This means that CacheBack would *include* this image in the results.

IMPORTANT: One the benefits of using the PARD analysis technique is that it can be applied to any image size.

Scenario 1 – Using PARD to <u>even further reduce</u> examinable recordsets.

In the following screenshot, CacheBack's Gallery has been used to display ALL pictures from our sample project (case) file. After using the Quick Query button from the Toolbar to "Show Only Pictures", our Gallery still displays a total of 329 pictures. As shown below, there are clearly some images that are NOT photographs and appear to have been simply apart of the webpage. What we also see are a couple of images that appear to be photographs.



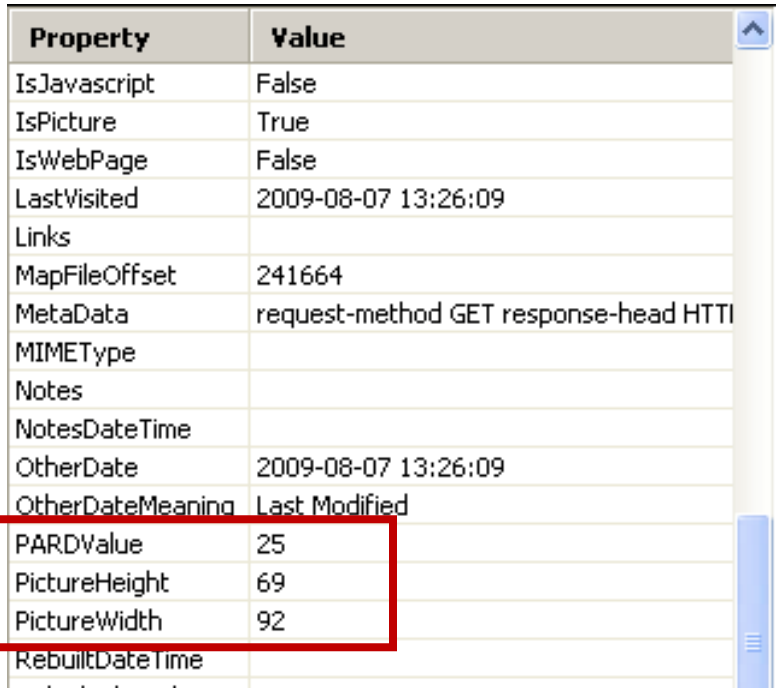
Let's now focus on the three (of the 4) images which are identifed as #367, 391 and 499. Suppose that the dimensions of these particular images *are of forensic interest* and let's further suppose that *these particular size images* will be the source of our evidence for this sample project.

How can we now use PARD to further eliminate other images that are not of forensic interest?

Let's see….

Picture 3 – Properties Pane content for thumbnail #499.

Using the metadata provided by the Properties Pane for any of the three images of interest (in this case #499), we can determine the image's height and width. We can use this information to calculate the PARD value as we have done in previous examples. However, this is not necessary because the Properties Pane contains the PARD value, already calculated for us.

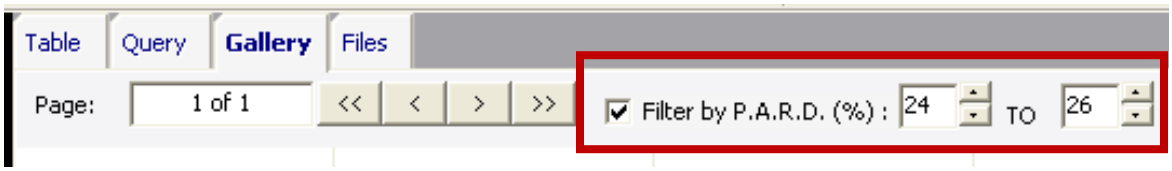


Property	Value
IsJavascript	False
IsPicture	True
IsWebPage	False
LastVisited	2009-08-07 13:26:09
Links	
MapFileOffset	241664
MetaData	request-method GET response-head HTTI
MIMEType	
Notes	
NotesDateTime	
OtherDate	2009-08-07 13:26:09
OtherDateMeaning	Last Modified
PARDValue	25
PictureHeight	69
PictureWidth	92
RebuiltDateTime	

As we can see, the PARD value is 25. What we can now do is "fine-tune" our PARD range of values using the scroll buttons on the Gallery tab. If our interest now is to find *all other similarly shaped images*, then we can set the PARD values at 24 to 26.

Picture 4 – PARD values re-configured to a range of 24 to 26.

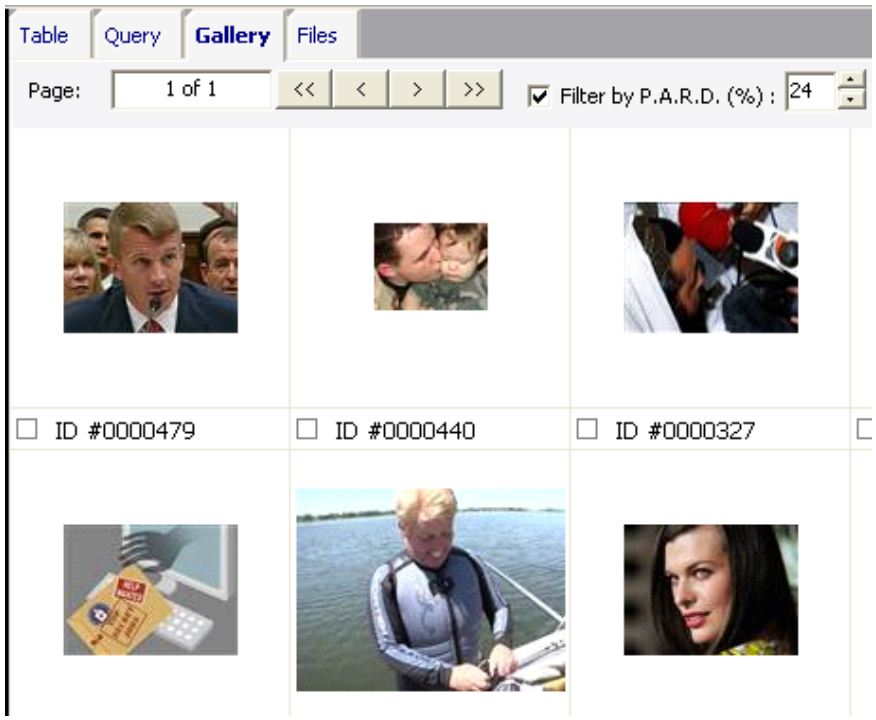
NOTE: The "Filter by P.A.R.D." must be selected as well!



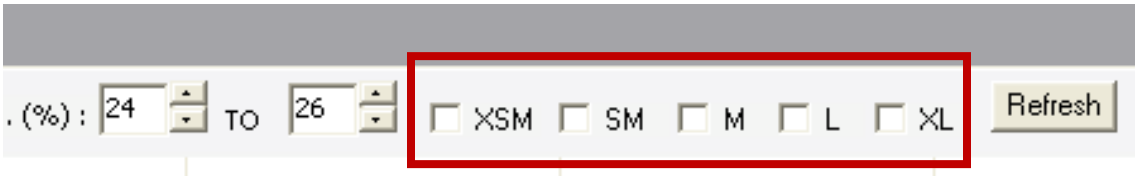
We then need to click on the Refresh button off to the right in order to reload the Gallery.

Picture 5 – A new, smaller Gallery.

Our results in the Gallery are quite obvious. Now instead of 329 images to review, we have a more relevant gallery of just 99 images. The following is a screenshot that shows our new gallery. Notice how we still have a larger photo among our results.



TIP: To tidy up our Gallery even further, we can make use of the optional PARD Filters (checkboxes) found on the Gallery Navigation Bar (which is located on the Gallery Tab). The use of these filters can only be used if the "Filter by P.A.R.D." option is selected. Each checkbox corresponds to a "range" of "preset dimensions" for comparison purposes and cannot be modified.



Remember, any changes made to the PARD Navigation Bar require you to click on the Refresh button to reload the Gallery.

GLOSSARY OF TERMS

The following is a list of some of the more important terms that are used within this user manual.

CacheBack Project(.CBP) File

Description: Project files in CacheBack are database files that conform to the Microsoft Access Database architecture. A project file is essentially a "case file" which is perhaps the more common term in use by most forensic examiners. For the purpose of this manual, the terms "project file" and "case file" are used interchangeably and are jointly meant to refer to the former term: "project file".

Daylight Period

Description: The defined number of months, weeks and days in a given calendar year that represent a period of time in which Daylight Savings is observed. Daylight Period if often used to differentiate between *Summer Time* and *Winter Time, where Summer Time* is typically the "Daylight Period". For countries in the Northern Hemisphere (above the Equator), Daylight Savings starts in the Spring and ends in the Fall. For countries in the Southern Hemisphere (below the Equator), Daylight Savings stars in the Fall and ends in the Spring.

Unallocated Clusters (or Space)

Description: *Clusters* is a term that represents a group of individual "sectors" on a hard disk drive, which collectively, are used to *store* file data. These storage units are therefore said to be *allocated*. Operating systems will use one or more sectors to store file data. When a file is deleted, the file system will simply mark the specific sectors as "available" for use, however, the original data will still occupy the original sectors until overwritten. As a result, these *freed* sectors are better described as *unallocated.* While it would arguably more correct to refer the smallest unit of measure in this context (eg: unallocated sectors), the accepted practice is to make reference to the actual *clusters (or "space")*.

Abbreviation(s): UC

Universal Resource Locator (URL)

Description:

1. Uniform Resource Locator: a protocol for specifying addresses on the Internet.
2. An address that identifies a particular file on the Internet, usually consisting of the protocol, as http, followed by the domain name.

Example: http://www.google.com

Wikipedia.org

Description:

Wikipedia (pronounced /ˌ wi: ki' pi: diə/) is a multilingual, Web-based, free-content encyclopedia project. The name "Wikipedia" is a portmanteau of the words *wiki* (a type of collaborative Web site) and *encyclopedia*. Wikipedia's articles provide links to guide the user to related pages with additional information.

Wikipedia is written collaboratively by volunteers from all around the world. Anyone with internet access can make changes to Wikipedia articles.

Disclaimer:

While most articles at the Wikipedia.org website appear to be fairly accurate, wiki information unfortunately cannot be 100% guaranteed. Therefore, forensic examiners should always seek to qualify particular facts through alternate sources.

NOTE: Certain references to the site have been made within this manual as the content description being referred, appears to accurately reflect the topic being discussed.