



SITECATALYST

Implementation Manual

August 4, 2009

Version 6.3



©August 2009 by Omniture® or its subsidiaries.

All rights reserved.

Information contained in this document is proprietary to Omniture® and may be used or disclosed only with written permission from Omniture. This document, or any part thereof, may not be reproduced without the prior written permission of Omniture.

This document refers to numerous products by their trade names. In most, if not all, cases these designations are Trademarks or Registered Trademarks of their respective companies.

This document and the related software described in this manual are supplied under license or nondisclosure agreement and may be used or copied only in accordance with the terms of the agreement. The information in this document is subject to change without notice and does not represent a commitment on the part of Omniture.

The names of companies and individuals used in examples in the manuals, and in any sample databases provided, are fictitious and are intended to illustrate the use of the software. Any resemblance to actual organizations or individuals, whether past or present, is purely coincidental.

1	Getting Started.....	1
1.1	Welcome to SiteCatalyst.....	1
1.2	Implementing Omniture with the Fusion Methodology.....	1
1.2.1	Omniture Fusion Deployment Process.....	2
1.3	Implementing SiteCatalyst Code	5
1.3.1	Code to Paste.....	5
1.3.2	JavaScript Code	6
1.3.3	Caching	7
1.3.4	SiteCatalyst JavaScript Compression	7
1.4	How does SiteCatalyst Collect and Report Data?	7
1.5	Additional Implementation Resources.....	8
2	SiteCatalyst Variables	9
2.1	Variables and Limitations.....	9
2.1.1	Illegal JavaScript Characters	13
2.1.2	s_account	13
2.1.3	browserHeight.....	15
2.1.4	browserWidth.....	15
2.1.5	campaign	15
2.1.6	channel	17
2.1.7	charSet	17
2.1.8	colorDepth	18
2.1.9	connectionType	18
2.1.10	cookieDomainPeriods	19
2.1.11	cookieLifetime	19
2.1.12	cookiesEnabled.....	20
2.1.13	currencyCode.....	21
2.1.14	dc	22
2.1.15	doPlugins	22
2.1.16	dynamicAccountList	23
2.1.17	dynamicAccountMatch	24
2.1.18	dynamicAccountSelection	25
2.1.19	dynamicVariablePrefix.....	26
2.1.20	eVarN	26
2.1.21	events.....	28
2.1.22	fpCookieDomainPeriods.....	30
2.1.23	hierN	31
2.1.24	homepage	32
2.1.25	javaEnabled.....	32
2.1.26	javascriptVersion	33
2.1.27	linkDownloadFileTypes	33
2.1.28	linkExternalFilters	34
2.1.29	linkInternalFilters.....	35
2.1.30	linkLeaveQueryString.....	37
2.1.31	linkName	38
2.1.32	linkTrackEvents	38

2.1.33	linkTrackVars	39
2.1.34	linkType	41
2.1.35	mediaLength	41
2.1.36	mediaName	42
2.1.37	mediaPlayer	43
2.1.38	mediaSession	44
2.1.39	Media.trackEvents	45
2.1.40	Media.trackVars	45
2.1.41	mobile	46
2.1.42	s_objectID	46
2.1.43	pageName	47
2.1.44	pageType	48
2.1.45	pageURL	49
2.1.46	plugins	50
2.1.47	products	51
2.1.48	propN	53
2.1.49	purchaselD	54
2.1.50	referrer	54
2.1.51	resolution	55
2.1.52	server	55
2.1.53	state	56
2.1.54	trackDownloadLinks	57
2.1.55	trackExternalLinks	58
2.1.56	trackingServer	58
2.1.57	trackingServerSecure	59
2.1.58	trackInlineStats	59
2.1.59	transactionID	59
2.1.60	s_usePlugins	60
2.1.61	visitorID	61
2.1.62	visitorNamespace	61
2.1.63	zip	62
3	Traffic props and Conversion eVars	64
3.1	Using props vs. eVars	64
3.2	Using props as Counters	64
3.3	Counting Content Hierarchies	65
3.4	What is a Predefined Event?	66
3.4.1	Detailed Product View Page	66
3.5	What is a Custom Event?	67
3.6	Email Campaign Tracking	67
3.6.1	Implementing Email Campaign Tracking	68
3.6.2	Tracking External Email	69
3.7	Products	70
3.8	When Should the Product Variable be Set?	70
3.8.1	Setting a Product with an Event	70
3.8.2	Setting a Product without an Event	71
3.8.3	Field Definitions of the Product String	71
3.8.4	Sample Product Strings	71
4	Pathing	73
4.1	Enabling Pathing on a prop	73
4.2	Pathing by Campaign or Tracking Code	73
4.3	Reasons Pathing may not be Recorded	74
4.4	Moving from Section to Section	74
4.5	Moving from Page Template to Page Template	74
4.6	Segmenting Paths by User Type	75
5	Using Implementation Plug-ins	76

5.1	Calling Plug-ins with the doPlugins Function.....	76
5.1.1	Code Example	76
5.1.2	Renaming the doPlugins Function.....	76
5.1.3	Using doPlugins.....	76
5.1.4	Installed Plug-ins.....	77
5.2	Implementation Plug-ins.....	77
5.2.1	APL Plug-in Utility	78
5.2.2	getQueryParam.....	79
5.2.3	getValOnce.....	81
6	Testing and Validation	83
6.1	Using the JavaScript Debugger	83
6.1.1	Setting up the Debugger.....	83
6.1.2	Alternate Debugger Versions.....	84
6.1.3	Variables and Query String Parameters.....	85
6.1.4	Identifying the s_account Variable in the JavaScript Debugger	86
6.2	Packet Monitors	86
6.3	Deployment Validation.....	86
6.3.1	JavaScript JS File	87
6.3.2	Code Modifications	87
6.3.3	Variables and Values	88
6.3.4	Custom Links.....	88
6.3.5	Custom Variables	89
6.4	Implementation Acceptance	89
6.4.1	Data Accuracy Validation.....	90
7	Appendix A: Optimizing your SiteCatalyst Implementation	92
7.1	Variable Length.....	92
7.2	HTML Code Snippet	92
7.3	JavaScript Library File	92
7.4	Caching Directives	92
7.5	Tables	92
7.6	File Compression	93
7.7	Secure Pages	93
7.8	Content Delivery Services/Networks	93
7.9	JavaScript File Location and Concurrency.....	93
7.10	Peering.....	93
8	Appendix B: Implementing SiteCatalyst without JavaScript	95
8.1	PHP Measurement Library	95
8.2	Variable Names.....	96
8.3	Other Requirements	97
8.4	Sample Code	99
8.4.1	Example 1	99
8.4.2	Example 2	99
8.5	Supported SiteCatalyst Reports.....	100
9	Appendix C: Implementing SiteCatalyst for Media Tracking	101
9.1	Auto-track implementation	101
9.1.1	Sample Code	101
9.2	Custom implementation	101
9.2.1	Sample Code	102
10	Appendix D: Implementing SiteCatalyst on Mobile Sites.....	103
	Index.....	105

PREFACE

The SiteCatalyst Implementation Manual is a reference document that you can use to gather technical information prior to implementation, at any step during the implementation process, or after implementation is complete. Implementing SiteCatalyst can be a technical process, so this document is designed such that either a project manager or technical manager can use its contents to make business or technical decisions to fully utilize SiteCatalyst to collect the web analysis data you need to maximize the return on investment for your website.

Product Overview

SiteCatalyst™ delivers a powerful, intuitive, and industry-leading user interface, which enables users to quickly access actionable information. SiteCatalyst™ is the only solution that provides access to Traffic, Path, Campaign, Commerce, Segmentation, and Data Warehousing all in one interface. The easy-to-use approach allows Omniture customers of all skill levels to access actionable, real-time reports and dashboards. SiteCatalyst is an enterprise-grade solution designed for rapid user adoption across the organization.

SiteCatalyst™ comes with a comprehensive set of out-of-the-box reports that can be further tailored to suit special needs. Unique to SiteCatalyst™ is the ability to move between reports in an interactive fashion, allowing the user to keep the analysis momentum. SiteCatalyst is the only web analytics product that provides a comprehensive view with real-time reporting as well as a full data warehouse. SiteCatalyst provides superior scalability and reliability and is designed to handle massive traffic spikes. SiteCatalyst can report website analysis data for clients from different business models and/or industries, whether the client has a smaller online client base or a billion page views per month.

Document Conventions

To ensure that this document is as easy to read and understand as possible, the following conventions may be used throughout.



NOTE: A Note distinguishes helpful information of which you may not be aware.



TIP: A Tip adds insightful information that may help you use the system better.



CAUTION: A Caution makes you aware of the results of an action. The results may not necessarily be damaging, but they definitely are important.



WARNING! A Warning highlights an action that may result in damage to your system or an unforeseen loss of data.

Additional References/Resources

In addition to this document, Omniture offers white papers for several other topics. The white papers are available in PDF format. In order to view them, you must first have Adobe® Reader® installed on your system. Reader is a free download from Adobe at <http://www.adobe.com>. For more information on the available white papers, contact Omniture ClientCare.

About Omniture

Omniture, Inc., headquartered in Orem, Utah, is the pioneer of next-generation online analytics technology that delivers the essential intelligence needed by Web commerce leaders and innovators to drive the business decisions that increase ROI. Omniture is the largest, ASP based, online analytics provider by revenue, and Omniture's SiteCatalyst is the most mature and comprehensive technology on the market, offering industry leading scalability and flexibility combined with an intuitive user interface. Omniture is the only company in its market to offer a

comprehensive view of activity on a company's Website that includes historical (data warehouse) and real-time analysis and reporting. In addition, Omniture offers knowledgeable professional service teams, experienced in helping customers determine the questions they must ask to arrive at the answers they require. Proof of its world-class technology and outstanding team, Omniture has the highest level of retained and satisfied customers in the market, including eBay, AOL, Wal-Mart, Gannett, Microsoft, Oracle, Overstock.com, GM and Hewlett-Packard. www.omniture.com.

1 Getting Started

This document is to be used as a technical and functional reference to answer business and technical questions during and after implementation. Project managers can use the document to gain further understanding about the SiteCatalyst web analytics options and implementation considerations that are available to them. Technology managers can use the document to reference syntax and examples that will help them as they implement the web analytics options.

1.1 Welcome to SiteCatalyst

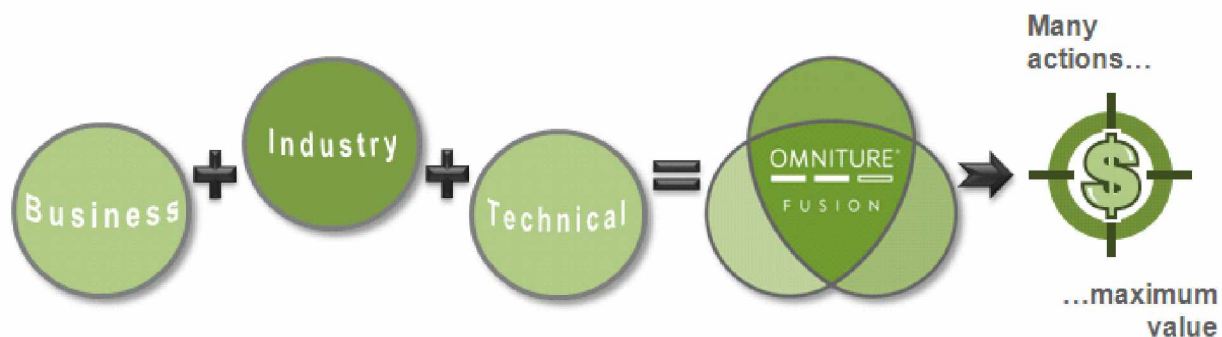
SiteCatalyst is the most easy-to-install and easy-to-use web analytics solution available. Its next-generation web analytics, accurate real-time reporting, unparalleled flexibility, unmatched scalability and enhanced accessibility make it the industry's most advanced Web analytics solution.

SiteCatalyst's advanced reporting capabilities generate over 150,000 report combinations. Traffic reporting lets you analyze all aspects of visitor activity, such as traffic patterns, popular channels, preferred technology and finding methods used. Customer data, including customer loyalty, sales cycle, products purchased, promotional activity, and affiliate programs are identified by conversion reporting. Pathing reports let you track and record entire browsing paths of both your visitors and customers, from start to finish. You can easily view your site traffic as it flows from one page or item to the next, discover new patterns and popular paths, or search out the specific paths that your visitors take. Additionally, custom traffic reporting and a fully interactive calendar let you quickly identify and track any factors you can identify, for any time period you choose.

These and other cutting-edge features present all the facts, trends, and insight you need to ensure your online success, empowering you to make truly intelligent business decisions about your website and audience. With the answers to those questions that drive you crazy firmly in your grasp, you can spend your time improving and executing your web initiatives rather than acquiring the information needed to justify changes.

1.2 Implementing Omniture with the Fusion Methodology

Omniture Fusion is the proprietary deployment methodology for all Omniture products. At the core of this methodology is the blending of your organization's business needs with Omniture's industry best practices and technical deployment expertise.



Implementing with Omniture Fusion ensures that not only are your business needs met, but also that return on your analytics investment is realized quickly and efficiently. Leveraging the experience from deploying SiteCatalyst over 3,000 times, Omniture has defined industry best practices across all major verticals. These industry fundamentals provide a foundation upon which your organization's custom business needs can be met. Since each industry and organization is a little different, each Omniture implementation is a custom fit to your organization. Contact Omniture Implementation Consulting for information on how to implement Omniture to best fit your organization's needs.

1.2.1 Omniture Fusion Deployment Process

The Omniture Fusion deployment methodology ensures that all industry fundamentals and any included custom business requirements are deployed in an accurate and effective manner. Omniture Fusion deployments are broken into three implementation phases and two post-implementation phases (the Adoption Phase and the Optimize Phase).



Define Phase

The object of the Define stage is to define and agree on project scope including industry fundamentals to implement, custom business requirements, sites to implement, and target timelines. The definition of scope is also determined by the contracted consulting hours and Statement of Work.

Table: Key Activities and Meetings

Activity	Duration	Description	Participants
Kick-off Call	1 hour	During this call roles and responsibilities are defined. The Omniture Fusion deployment process is reviewed, industry Fundamentals are highlighted, any custom reporting requirements are defined, and project timelines are agreed upon.	Implementation Consultant Account Manager (optional) Your Business Lead Your Technical Lead
Industry Fundamental Review Call (optional)	1 hour	This is an optional call to review industry Fundamentals in more detail. This call can also be used to discuss additional custom business requirements.	Implementation Consultant Your Business Lead
Begin First-Party Cookie Process	N/A	Your Implementation Consultant will provide information required to obtain a secure SSL certificate, and implement using 1 st party cookies	Implementation Consultant Your Technical Lead

The deliverables for this phase are listed below.

- Kick-off PowerPoint Slides containing:
 - Identified Roles & Responsibilities
 - Slides describing industry Fundamentals (optional)

- List of custom requirements (optional)
- Login Access to SiteCatalyst
- First-Party Cookie Request Form

Design Phase

The object of the Design Phase is to design a technical implementation solution that incorporates industry fundamentals and custom requirements.

Table: Key Activities and Meetings

Activity	Duration	Description	Participants
Prepare Solution Design	3-5 Days	Your Implementation Consultant will develop a Solution Design that meets all stated requirements agreed upon during the Define phase.	Implementation Consultant
Solution Design Review Call	1-2 hours	This is a call to review the Solution Design and ensure that those implementing the solution understand the Solution Design documentation.	Implementation Consultant Your Technical Lead Your Business Lead

The deliverables for this phase are listed below.

- Omniture Fusion Solution Design Reference
This is an Excel spreadsheet listing how each Omniture variable is used to meet the business requirements.
- Data Collection Code
- Implementation Manual (download via the Help section)

Business Questions

Metrics/KPIs

Sample Reports

Quick Wins

Deployment Instructions

Deploy Phase

The object of the Deploy Phase is to deploy the Omniture Fusion solution through iterative tagging and validation.



Note: This phase must be completed before the contracted Service Commencement Date; otherwise, Implementation Consulting hours will expire. The Service Commencement Date is also the date on which regular invoicing will begin—regardless of the status of your implementation.

Table: Key Activities and Meetings

Activity	Duration	Description	Participants
Tagging the site	4-6 weeks	Using the Omniture Fusion Playbook and Solution Design Reference, [Customer Name]'s development team will implement the solution on [Customer Domain.com]	Your Technical Lead
Regular Status Calls	30-60 minutes (weekly)	This call is to ensure the solution is being deployed within the agreed timeline. This call should also be used to address outstanding questions/issues, and may be used to cooperatively validate the deployment.	Implementation Consultant Your Technical Lead Your Business Lead
Complete Fusion Validation Checklist	1-2 days	Once the solution is deployed in a development environment, your Implementation Consultant will complete a validation checklist. Portions of the checklist will need to be jointly validated by [Customer Name]'s Business Lead to ensure data shown in the reports appears accurate.	Implementation Consultant Your Business Lead Your Technical Lead
Fusion Complete Call	30-60 minutes	This call is typically done post go-live, but may also be done once the solution is validated in the development environment. The timing of this call should be done prior to the contracted Service Commencement Date.	Implementation Consultant Account Manager Your Business Lead Your Technical Lead Your Executive Sponsor (optional)

The deliverables for this phase are listed below.

- Completed Omniture Fusion Validation Checklist
- Updated Solution Design Reference with validation notes

Adoption Phase

The object of the Adoption Phase is to ensure corporate adoption through end-user training and proper report dissemination.

Table: Key Activities and Meetings

Activity	Duration	Description	Participants
End-user Training	Few days/weeks	Attend Omniture University training courses, and/or watch on-demand training videos found in the Help section.	All Your Organization's Users
Create Dashboards and Distribution Reports	Few days/weeks	Identify end-user needs and schedule appropriate reports for delivery.	Your Business Lead Omniture Business Consultant (if contracted)

The deliverables for this phase are listed below.

- Regularly scheduled dashboards and reporting

Optimize Phase

The object of the Optimize Phase is to increase Return on Investment (ROI) through data-driven optimization.

Table: Key Activities and Meetings

Activity	Duration	Description	Participants
Data analysis and optimization	On-going	Analyze data trends and provide recommendations for optimization.	Your Business Lead Omniture Business Consultant (if contracted)

The deliverables for this phase are listed below.

- Results from data analysis
- Recommendations for enhancements
- Periodic presentation of recommendations to your organization's Executive Sponsor

1.3 Implementing SiteCatalyst Code

Though the Fusion deployment process may seem complex, the actual SiteCatalyst implementation is a simple process of adding the following two components to the pages of a website.

- SiteCatalyst HTML code
- JavaScript library file.

Deploying SiteCatalyst involves pasting Omniture HTML code onto each page (or page template) of a website. This HTML code contains variables and other identifiers that facilitate the data collection process. These variables may be dynamically populated with server or application variables. The code snippet also calls the JavaScript library file, which contains SiteCatalyst-specific JavaScript functions used during metrics collection. Client browsers cache this file after the first request, resulting in virtually no incremental server load for the instrumented site.

1.3.1 Page Code

The sample code below shows the SiteCatalyst page code within the HTML source.



NOTE: The variables would also be populated.

```

<html>
<head></head>
<body>
<!-- SiteCatalyst code version: H.20.3.
Copyright 1997-2009 Omniture, Inc. More info available at
http://www.omniture.com -->
<script language="JavaScript" type="text/javascript" src="http://INSERT-
    DOMAIN-AND-PATH-TO-CODE-HERE/s_code.js"></script>
<script language="JavaScript" type="text/javascript"><!--
/* You may give each page an identifying name, server, and channel on
the next lines. */
s.pageName=""
s.server=""
s.channel=""
s.pageType=""
s.prop1=""
s.prop2=""
s.prop3=""
s.prop4=""
s.prop5=""
/* Conversion Variables */
s.campaign=""
s.state=""
s.zip=""
s.events=""
s.products=""
s.purchaseID=""
s.eVar1=""
s.eVar2=""
s.eVar3=""
s.eVar4=""
s.eVar5=""
/***** DO NOT ALTER ANYTHING BELOW THIS LINE ! *****/
var s_code=s.t();if(s_code)document.write(s_code)//--></script>
<script language="JavaScript" type="text/javascript"><!--
if(navigator.appVersion.indexOf('MSIE')>=0)document.write(unescape('%3C')+'\!
- '+'-')
//--></script><noscript><a href="http://www.omniture.com" title="Web
    Analytics"></a></noscript><!--/DO NOT REMOVE/--
>
<!-- End SiteCatalyst code version: H.20.3. -->
</body>
</html>

```



NOTE: You will need to add your report suite ID to the code locations where you see **[YOUR-REPORT-SUITE-ID]**.

1.3.2 JavaScript Code

The sample code below shows the JavaScript code.

```

/* SiteCatalyst code version: H.17.
Copyright 1997-2008 Omniture, Inc. More info available at
http://www.omniture.com */

```

```
var s_account="[YOUR-REPORT-SUITE-ID]"
var s=s_gi(s_account)
/***** CONFIG SECTION *****/
/* You may add or alter any code config here. */
s.charset="ISO-8859-1"
/* Conversion Config */
s.currencyCode="USD"
/* Link Tracking Config */
s.trackDownloadLinks=true
s.trackExternalLinks=true
s.trackInlineStats=true
s.linkDownloadFileTypes="exe,zip,wav,mp3,mov,mpg,avi,wmv,doc,pdf,xls"
s.linkInternalFilters="javascript:,207,207,sitecatalyst,omniture,www.register
    at.com,thelink.omniture.com"
s.linkLeaveQueryString=false
s.linkTrackVars="None"
s.linkTrackEvents="None"

/* WARNING: Changing any of the below variables will cause drastic
changes to how your visitor data is collected.  Changes should only be
made when instructed to do so by your account manager.*/
s.dc="112"
...remainder of JavaScript code...
```



NOTE: You will need to add your report suite ID to the code locations where you see `[YOUR-REPORT-SUITE-ID]`.

1.3.3 Caching

The JavaScript file is cached in the visitor's browser after initially loaded. The file is only downloaded once per session and is not downloaded on each page even though it is used by every page on the site. Therefore, on a site where users average more than a couple of page views per session (most websites), transferring most of the JavaScript into this file results in less overall downloaded data.

1.3.4 SiteCatalyst JavaScript Compression

In cases where you may be concerned about page weight (i.e. size), Omniture recommends that you consider using GZIP, which is supported by all major browsers and offers better performance than JavaScript compression, to compress and decompress the core SiteCatalyst JavaScript file. The following three links help explain how you can use GZIP functionality on your site to handle compression of the SiteCatalyst JavaScript code.

- http://httpd.apache.org/docs/2.0/mod/mod_deflate.html
- <http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/25d2170b-09c0-45fd-8da4-898cf9a7d568.mspx?mfr=true>
- <http://www.cubicleman.com/2007/04/06/enabling-gzip-compression-with-tomcat-and-flex/>

1.4 How does SiteCatalyst Collect and Report Data?

The SiteCatalyst data collection process is fairly basic. The SiteCatalyst code (or Code to Paste) is placed in the HTML source code on the desired pages of your production website(s).

When a visitor enters your website, your landing page loads in their browser window. As the page loads, the SiteCatalyst Code to Paste sets certain variables, for example, the `pageName` variable, and the SiteCatalyst

JavaScript Include file. The variables (and other identifiers) are used to facilitate the data collection process, and can be dynamically populated with server or application variables.

The JavaScript Include file builds an image request, or web beacon. The image request uses a transparent 1x1 pixel image to pass data from the Web page to the Omniture data center. No personal data about the visitors is ever passed to Omniture. The process is repeated every time the visitor accesses a page on your site that contains SiteCatalyst code.



NOTE: The image request contains a query string that passes the data variables that are collected on the page. Some of these variables are set specifically in the Code to Paste, and some variables are set automatically within the JavaScript file. Additionally, the image request contains a number of variables in the HTTP header. All of these variables constitute the sum of the data collection elements.

When the HTML Code to Paste and JavaScript file are in place, and the Implementation Consultant has made the appropriate configurations, the Web analysis data for your website is stored in tables called report suites. The reports in the SiteCatalyst interface draw their data from the report suite tables.

1.5 Additional Implementation Resources

The primary purpose of this document is to give reference material for implementing SiteCatalyst. However, Omniture offers additional implementation resources, including the following white papers, which can be accessed via the Omniture Help site or the Omniture Knowledge Base.

- SiteCatalyst JavaScript Code – Upgrading from G Code to H Code
- SiteCatalyst JavaScript Code – Making an H Code JS File Compatible with G Page Code
- SiteCatalyst and Silverlight
- RightNow Technologies and SiteCatalyst
- Page Naming Strategies

In addition to these white papers, the Omniture Knowledge Base contains additional information on SiteCatalyst implementations and many other topics. To access the Knowledge Base, go to the Omniture Help site and click **Knowledge Base**.

2 SiteCatalyst Variables

SiteCatalyst uses several variables to store various values in memory. The values help perform various reporting functions in SiteCatalyst. For example, the value in the `pageName` variable is the name of the web page being reported in SiteCatalyst.

Omniure uses plug-ins, which are programs that are added to your browser by modifying the existing JavaScript code, to extend the capabilities of your browser to give you more functionality that is not available in the original application.

All of the variables and plug-ins that are created by Omniure for you to use are discussed below. Each variable and plug-in shows both syntax and other usage examples.

2.1 Variables and Limitations

The JavaScript file used to track web pages is extremely versatile; however, potential problems and pitfalls may arise with an increased level of flexibility. The following table lists the available SiteCatalyst variables.

Table: SiteCatalyst Variables

Variable	Description
<code>s_account</code>	The <code>s_account</code> variable determines the report suite where data will be stored and reported in SiteCatalyst.
<code>browserHeight</code>	The <code>browserHeight</code> variable displays the height of the browser window.
<code>browserWidth</code>	The <code>browserWidth</code> variable displays the width of the browser window.
<code>campaign</code>	The <code>campaign</code> variable identifies marketing campaigns used to bring visitors to your site. The value of <code>campaign</code> is usually taken from a query string parameter.
<code>channel</code>	The <code>channel</code> variable is most often used to identify a section of your site. For example, a merchant may have sections like Electronics, Toys, or Apparel. A media site may have sections like News, Sports or Business.
<code>charSet</code>	SiteCatalyst uses the <code>charSet</code> variable to translate the character set of the web page into UTF-8.
<code>colorDepth</code>	The <code>colorDepth</code> variable shows the number of bits used to display color on each pixel of the screen.
<code>connectionType</code>	The <code>connectionType</code> variable indicates (in Internet Explorer) whether the browser is configured on a LAN or modem connection.
<code>cookieDomainPeriods</code>	The <code>cookieDomainPeriods</code> variable is used to determine the domain with which cookies will be set. For <code>www.mysite.com</code> , <code>cookieDomainPeriods</code> should be "2." For <code>www.mysite.co.jp</code> , <code>cookieDomainPeriods</code> should be "3."
<code>cookieLifetime</code>	The <code>cookieLifetime</code> variable is used by both JavaScript and SiteCatalyst servers in determining the lifespan of a cookie.
<code>cookiesEnabled</code>	The <code>cookiesEnabled</code> variable indicates whether a first-party session cookie

	could be set by JavaScript.
currencyCode	The currencyCode variable is used to determine the conversion rate to be applied to revenue as it enters the SiteCatalyst databases. SiteCatalyst databases store all monetary amounts in a currency of your choice. If that currency is the same as that specified in currencyCode, or currencyCode is empty, no conversion is applied.
dc	Allows you to set the data center – Dallas or San Jose – to which your data will be sent.
doPlugins	doPlugins is a reference to the s_doPlugins function, and allows the s_doPlugins function to be called at the appropriate location within the JavaScript file.
dynamicAccountList	The SiteCatalyst JavaScript file can be used to dynamically select a report suite to which it will send data. The dynamicAccountList variable contains the rules that will be used to determine the destination report suite.
dynamicAccountMatch	The dynamicAccountMatch variable uses the DOM object to retrieve the section of the URL to which all rules in dynamicAccountList are applied. This variable is only valid when dynamicAccountSelection is set to 'True.'
dynamicAccountSelection	The dynamicAccountSelection variable enables you to dynamically select the report suite based on the URL of each page.
dynamicVariablePrefix	The dynamicVariablePrefix variable allows deployment to flag variables which should be populated dynamically. Cookies, request headers, and image query string parameters are available to be populated dynamically.
eVarN	eVar variables are used for building custom reports within SiteCatalyst's Conversion Module. When an eVar is set to a value for a visitor, SiteCatalyst remembers that value until it expires. Any success events that a visitor encounters while the eVar value is active are counted toward the eVar value.
events	The events variable is used to record common shopping cart success events as well as custom success events.
fpCookieDomainPeriods	The fpCookieDomainPeriods variable is used to determine the domain with which cookies will be set.
hierN	The hierarchy variable is used to determine the location of a page in your site's hierarchy. This variable is most useful for sites that have more than three levels in the site structure.
homepage	The homepage variable indicates (in Internet Explorer) whether the current page is set as the user's home page.
javaEnabled	The javaEnabled variable indicates whether Java is enabled in the browser.
javascriptVersion	The JavaScriptVersion variable displays the version of JavaScript supported by the browser.

linkDownloadFileTypes	The linkDownloadFileTypes variable is a comma-separated list of file extensions. If your site contains links to files with any of these extensions, the URLs of these links will appear in the File Downloads Report.
linkExternalFilters	If your site contains many links to external sites, and you don't want to track ALL exit links, linkExternalFilters can be used to report on a specific subset of exit links.
linkInternalFilters	The linkInternalFilters variable is used to determine which links on your site are exit links. It is a comma-separated list of filters that represent the links that are part of the site.
linkLeaveQueryString	The linkLeaveQueryString variable determines whether or not the query string should be included in the Exit Links and File Download reports.
linkName	linkName is an optional variable used in Link Tracking that determines the name of a custom, download, or exit link. linkName is not normally needed because the third parameter in the tl() function replaces it.
linkTrackEvents	linkTrackEvents contains the events that should be sent with custom, download and exit links. This variable is only considered if linkTrackVars contains "events."
linkTrackVars	linkTrackVars is a comma-separated list of variables that will be sent with custom, exit and download links. If linkTrackVars is set to "" all variables that have values will be sent with link data.
linkType	linkType is an optional variable used in link tracking that determines which report a Link Name or URL will appear (Custom, Download or Exit Links). linkType is not normally needed because the second parameter in the tl() function replaces it.
mediaLength	The mediaLength variable specifies the total length of the media being played.
mediaName	This variable specifies the name of the video or media item. Additionally, it is only available via the Data Insertion API and Full Processing Data Source.
mediaPlayer	This variable specifies the player used to consume a video or media item.
mediaSession	This variable specifies the segments of a video or media asset consumed.
Media.trackEvents	The Media.trackEvents variable identifies which events should be sent with a media hit. Additionally, it is only applicable with JavaScript and ActionSource.
Media.trackVars	The Media.trackVars variable identifies which variables should be sent with a media hit. Additionally, it is only applicable with JavaScript and ActionSource.
mobile	Controls the order in which cookies and subscriber ids are used to identify visitors.
s_objectID	The s_objectID variable is a global variable that should be set in the onClick event of a link. By creating a unique object ID for a link or link location on a

	page, you can either improve ClickMap tracking or use ClickMap to report on a link type or location, rather than the link URL.
pageName	The pageName variable contains the name of each page on your site. If pageName is left blank, the URL is used to represent the page name in SiteCatalyst.
pageType	The pageType variable is used only to designate a 404 Page Not Found Error Page. It only has one possible value, which is "errorPage." On a 404 Error Page, the pageName variable should not be populated.
pageURL	In rare cases, the URL of the page is not the URL that you would like reported in SiteCatalyst. To accommodate these situations, SiteCatalyst offers the pageURL variable, which overrides the actual URL of the page.
plugins	The plugins variable, in Netscape and Mozilla-based browsers, lists the plugins installed in the browser.
products	The products variable is used for tracking products and product categories as well as purchase quantity and purchase price. The products variable should always be set in conjunction with a success event. Optionally, the products variable can track custom incrementor events and Merchandising Evars.
propN	Property (prop) variables are used for building custom reports within SiteCatalyst's Traffic Module. props may be used as counters (to count the number of times a page view is sent), for pathing reports, or in correlation reports.
purchaseID	The purchaseID is used to keep an order from being counted multiple times by SiteCatalyst. Whenever the purchase event is used on your site, you should use the purchaseID variable.
referrer	The referrer variable may be used to restore lost referrer information.
resolution	The resolution variable displays the monitor resolution of the visitor viewing the web page.
server	The server variable is used to show either the domain of a web page (to show which domains people come to) or the server serving the page (for a load balancing quick reference).
state	The state variable captures the state in which a visitor to your site resides.
trackDownloadLinks	Set trackDownloadLinks to 'true' if you would like to track links to downloadable files on your site. If trackDownloadLinks is 'true,' linkDownloadFileTypes is used to determine which links are downloadable files.
trackExternalLinks	If trackExternalLinks is 'true,' linkInternalFilters and linkExternalFilters are used to determine whether any link clicked is an exit link.
trackingServer	Is used for first-party cookie implementation to specify the domain at which the image request and cookie will be written. Used for non-secure pages.

trackingServerSecure	Is used for first-party cookie implementation to specify the domain at which the image request and cookie will be written. Used for secure pages.
trackInlineStats	trackInlineStats determines whether ClickMap data is gathered.
transactionID	Integration Data Sources utilize a transaction ID to tie offline data to an online transaction (like a lead or purchase generated online). Each unique transactionID sent to Omniture will be recorded in preparation for a Data Sources upload of offline information about that transaction. Refer to the Data Sources User Guide for more information.
s_usePlugins	If the s_doPlugins function is available and contains useful code, s_usePlugins should be set to 'true.' When usePlugins is 'true,' the s_doPlugins function is called prior to each image request.
visitorID	Visitors may be identified by the visitorID tag, or by IP address/User Agent. The visitorID may be up to 100 alpha-numeric characters and must not contain a hyphen
visitorNamespace	If visitorNamespace is used in your JavaScript file, do not delete or alter it. This variable is used to identify the domain with which cookies are set. If visitorNamespace changes, all visitors reported in SiteCatalyst may become new visitors. In short, do not alter this variable without approval from an Omniture Implementation Consultant.
zip	The zip variable captures the zip code in which a visitor to your site resides.

The following sections offer guidelines, syntax, and examples of configuration settings for each variable in order to avoid potential problems.

2.1.1 Illegal JavaScript Characters

The following list outlines characters and strings that are never allowed in JavaScript variables.

- Tab (0x09)
- Carriage return (0x0D)
- Newline (0x0A)
- ASCII characters with codes above 127 (unless multi-byte characters are enabled and charSet is populated appropriately)
- HTML tags (e.g. or ™)

Many variables, most notably products, hierarchy, and events, have additional limitations or syntax requirements. For individual variable limitations and syntax requirements, refer to the section corresponding to the variable name below.

2.1.2 s_account

The s_account variable determines the report suite where data will be stored and reported in SiteCatalyst. If sending to multiple report suites (multi-suite tagging) s_account may be a comma-separated list of values. The report suite ID is determined by Omniture.

Table: s_account Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
40 Bytes	In URL Path	N/A	N/A



NOTE: Each report suite ID must match the value created in the Admin Console. Each report suite ID must be 40 bytes or less, but the aggregate of all report suites (the entire comma-separated list) has no limit.

The report suite is the most fundamental level of segmentation in SiteCatalyst reporting. You can set as many report suites as your contract allows. Each report suite refers to a dedicated set of tables that are populated in Omniture's collection servers. A report suite is identified by the `s_account` variable in your JavaScript code.

Within SiteCatalyst, the site drop-down box in the top left of the reports displays the current report suite. Each report suite has a unique identifier called a report suite ID. The `s_account` variable contains one or more report suite IDs to which data is sent. The report suite ID value, which is invisible to SiteCatalyst users, must be provided or approved by Omniture before you use it. Every report suite ID has an associated "friendly name" that may be changed in the Report Suites section of the Admin Console.

The `s_account` variable is normally declared inside the JavaScript file (`s_code.js`). However, you may declare the `s_account` variable on the HTML page, which is a common practice when the value of `s_account` may change from page to page. Since the `s_account` variable has global scope, it should be declared immediately before including Omniture's JavaScript file. If `s_account` does not have a value when the JavaScript file is loaded, no data will be sent to SiteCatalyst.

Omniture's JavaScript Debugger will show the value of `s_account` in the path of the URL that appears just below the word "Image," just after `/b/ss/`. In some cases, the value of `s_account` will also appear in the domain, before `112.2o7.net`. The value in the path is the only value that determines the destination report suite. The bold text below shows the report suites that data is sent to, as it will appear in the debugger. For more information on the JavaScript Debugger, refer to *JavaScript Debugger* in this document.

`http://mycompany.112.2o7.net/b/ss/mycompanycom,mycompanysection/1/H.1-pdv-2/s21553246810948?[AQB]`

Syntax and Possible Values

The report suite ID is an alphanumeric string of ASCII characters, no more than 40 bytes in length. The only non-alphanumeric character allowed is a hyphen; spaces, periods, commas and other punctuation are not allowed. The `s_account` variable may contain multiple report suites, all of which will receive data from that page.

```
var s_account="reportsuitecom[,reportsuite2[,reportsuite3]]"
```

All values of `s_account` must be provided or approved by Omniture.

Examples

```
var s_account="mycompanycom"
var s_account="mycompanycom,mycompanysection"
```

Configuring the Variable in SiteCatalyst

The friendly name associated with each report suite ID may be changed by Omniture ClientCare. The friendly name can be seen in SiteCatalyst in the site drop-down box in the top, left section of the screen.

Pitfalls, Questions and Tips

- If s_account is empty, not declared, or contains an unexpected value, no data will be collected.
- When the s_account variable is a comma-separated list (multi-suite tagging), do not put spaces between report suite IDs.
- If s.dynamicAccountSelection is set to 'True,' the URL will be used to determine the destination report suite. Use the JavaScript Debugger to determine the destination report suite(s).
- In some cases, VISTA may be used to alter the destination report suite. Using VISTA to either re-route or copy the data to another report suite is recommended when using first party cookies, or if your site has more than 20 active report suites. For more information, refer to *VISTA* in this document.
- Always declare s_account inside the JS file or just before it is included.

2.1.3 browserHeight

The browserHeight variable displays the height of the browser window. This variable is populated after the page code and before doPlugins is run.

 **WARNING!:** This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: browserHeight Variable Parameters

Query Param	Value	Example	Reports Affected
bh	A positive integer	865	Traffic > Technology > Browser Height

2.1.4 browserWidth

The browserWidth variable displays the width of the browser window. This variable is populated after the page code and before doPlugins is run.

 **WARNING!:** This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: browserWidth Variable Parameters

Query Param	Value	Example	Reports Affected
bw	A positive integer	1179	Traffic > Technology > Browser Width

2.1.5 campaign

The campaign variable identifies marketing campaigns used to bring visitors to your site. The value of campaign is usually taken from a query string parameter.

Table: campaign Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
255 Bytes	v0	Conversion > Campaigns > Tracking Code	""

Every element in a marketing campaign should have a unique tracking code associated with it. For example, a paid search engine keyword may have a tracking code of 112233. When someone clicks the keyword with the 112233 tracking code and is routed to the corresponding web site, the campaign variable is often used to record the tracking code.

There are two main ways to populate the campaign variable. First, the `getQueryParam` plug-in, used in the JavaScript file, retrieves a query string parameter from the URL. For more information on the `getQueryParam` plugin, refer to *Plug-ins* in this document. Second, you can assign a value to the campaign variable in the HTML on the web page.

With either method of populating the campaign variable, the Back button traffic may inflate the actual number of click-throughs from a campaign element. For example, a visitor enters your site by clicking a paid search keyword. When the visitor arrives on the landing page, the URL contains a query string parameter identifying the tracking code for the keyword. The visitor then clicks a link to another page, but then immediately clicks the Back button to return to the landing page. When the visitor arrives a second time on the landing page, the URL with the query string parameter identifies the tracking code again, and a second click-through is registered, thereby falsely inflating the number of click-throughs. To avoid this inflation of click-throughs, Omniture recommends using the `getValOnce` plugin to force each campaign click-through to be counted only once per session. For more information on the `getValOnce` plugin, refer to *Plug-ins* in this document.

Syntax and Possible Values

```
s.campaign="112233"
```

The campaign variable has the same limitations as all other variables. Omniture recommends limiting the value to standard ASCII characters.

Examples

```
s.campaign="112233"  
s.campaign=s.getQueryParam('cid');
```

SiteCatalyst Configuration Settings

Each campaign value remains active for a user, and receives credit for that user's activities and success events until it expires. You can change the expiration of the campaign variable in the Admin Console.

Pitfalls, Questions and Tips

- To keep click-throughs from being inflated, use the `getValOnce` plugin to allow the click-through for a campaign to be counted only once per session. For more information on the `getValOnce` plug-in, refer to *Plug-ins* in this document.
- For more information on tracking marketing campaigns and keyword buys in SiteCatalyst, refer to *Campaigns* in this document.
- Use the JavaScript Debugger to see the actual value of campaigns (v0 in the debugger) sent into SiteCatalyst. If v0 does not appear in the debugger, no campaign data will be recorded for that page.

2.1.6 channel

The channel variable is most often used to identify a section of your site. For example, a merchant may have sections like Electronics, Toys, or Apparel. A media site may have sections like News, Sports or Business.

Table: channel Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 Bytes	CH	Site Content > Site Sections	""

Omniture recommends populating the channel variable on every page. You may also want to turn on a correlation between the channel and page name variables.

When sections have one or more levels of subsections, you can either show those sections in the channel variable or use separate variables to identify such levels. For more information, refer to the *Channels and Hierarchies* white paper.

The Conversion > Site Path > Site Sections Report is only populated on pages that contain conversion. For example, if "Home" is the value channel on the Home Page, but Home Page has no conversion, you will probably never see "Home" in the Conversion Site Sections Report.

Syntax and Possible Values

```
s.channel="value"
```

The channel variable has no extra limitations on its values.

Examples

```
s.channel="Electronics"  
s.channel="Media"
```

SiteCatalyst Configuration Settings

Contact Omniture ClientCare if you need to change the name of the Most Popular Site Sections Report.

Pitfalls, Questions and Tips

If your site contains multiple levels, you may use the hierarchy or another variable to designate those levels. For more information, refer to the *Channels and Hierarchies* white paper. If you were to think of the channel variable as an eVar, it would expire on a "Page View" and would only be set if conversion is used on the page.

2.1.7 charSet

SiteCatalyst uses the charSet variable to translate the character set of the web page into UTF-8. If the charSet variable contains an incorrect value, the data in all other variables will be translated incorrectly. If JavaScript variables on your pages (e.g. pageName, prop1, channel) contain only ASCII characters, charSet does not need to be defined. However, if the variables on your pages contain non-ASCII characters, the charSet variable must be populated.

Table: charSet Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	CE	N/A	""

charSet is used to identify the character set of the page. For more information on character sets, refer to *Multi-byte Character Sets* in this document before using the charSet variable.

Syntax and Possible Values

The charSet variable may only contain one of a predefined set of values, as listed in *Multi-byte Character Sets* in this document.

```
s.charSet="character_set"
```

Examples

```
s.charSet="ISO-8859-1"
```

```
s.charSet="SJIS"
```

SiteCatalyst Configuration Settings

Please notify Omniture ClientCare when using the charSet variable to enable support so your reports will be displayed in the proper character set.

Pitfalls, Questions and Tips

- The value of charSet must match the possible values listed in *Multi-byte Character Sets* in this document.
- The value of charSet should reflect the character set of the page.

2.1.8 colorDepth

The colorDepth variable is used to show the number of bits used to display color on each pixel of the screen, for example, 32 represents 32 bits of color on the screen. This variable is populated after the page code and before doPlugins is run.



WARNING! This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: colorDepth Variable Parameters

Query Param	Value	Example	Reports Affected
c	8, 16, and 32	32	Traffic > Technology > Monitor Color Depth

2.1.9 connectionType

The connectionType variable, in Internet Explorer, indicates whether the browser is configured on a LAN or modem connection. This variable is populated after the page code and before doPlugins is run.



WARNING!: This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: connectionType Variable Parameters

Query Param	Value	Example	Reports Affected
ct	lan or modem	lan	Traffic > Technology > Connection Type

2.1.10 cookieDomainPeriods

The cookieDomainPeriods variable is used to determine the domain with which cookies will be set. For www.mysite.com, cookieDomainPeriods should be "2." For www.mysite.co.jp, cookieDomainPeriods should be "3."

Table: cookieDomainPeriods Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	CDP	Affects ClickMap and Traffic > Technology > Cookies	"2"

// sample code for setting cookie domain variables

```
s.cookieDomainPeriods="3"
s.fpCookieDomainPeriods="2"
var d=window.location.hostname
if(d.indexOf('.co.uk')>-1||d.indexOf('.com.au')>-1)
  s.fpCookieDomainPeriods="3"
```

For more information, refer to *fpCookieDomainPeriods* in this document.

2.1.11 cookieLifetime

The cookieLifetime variable is used by both JavaScript and SiteCatalyst servers in determining the lifespan of a cookie.

Table: cookieLifetime Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	cl	Traffic > Technology > Cookies All visitor-related reports	""

If cookieLifetime is set, it overrides any other cookie expirations for both JavaScript and SiteCatalyst servers, with one exception, described below. The cookieLifetime variable may have one of three values.

- A number representing the lifetime of a cookie in seconds (3600 for one hour)
- The word "SESSION," which means all cookies expire when the browser is closed

- The word "NONE," which means no cookies are set

The only time cookieLifetime will not be used to set cookie expiration is when cookieLifetime is a number and a session cookie is being written. The only time a session cookie is not written is when cookieLifetime is "NONE." If cookieLifetime is set to " " or not set at all, the default cookie lifetimes will be used. For more information on cookie expiration, refer to any of the following sections in this document.

- *SiteCatalyst Cookies*
- *Cookies*
- *JavaScript Settings and Plugins*

Syntax and Possible Values

```
s.cookieLifetime="value"
```

The possible values are listed as follows.

- ""
- "NONE"
- "SESSION"
- An integer representing the number of seconds until expiration

Examples

```
s.cookieLifetime="SESSION"  
s.cookieLifetime="86400" // one day in seconds
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips



WARNING! cookieLifetime will affect SiteCatalyst tracking. If, for example, cookieLifetime is two days, then monthly, quarterly, and yearly unique visitor reports will be incorrect. Use caution when setting cookieLifetime.

2.1.12 cookiesEnabled

The cookiesEnabled variable indicates whether a first-party session cookie could be set by JavaScript. This variable is populated after the page code and before doPlugins is run.



WARNING! This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: cookiesEnabled Variable Parameters

Query Param	Value	Example
k	Y or N	Y

2.1.13 currencyCode

The currencyCode variable is used to determine the conversion rate to be applied to revenue as it enters the SiteCatalyst databases. SiteCatalyst databases store all monetary amounts in a currency of your choice. If that currency is the same as that specified in currencyCode, or currencyCode is empty, no conversion is applied.

Table: currencyCode Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	cc	Conversion > Purchases > Revenue All Conversion reports showing Revenue or monetary values	"USD"

If your site allows visitors to purchase in multiple currencies, you will need to use the currencyCode variable to make sure revenue is stored in the appropriate currency. For example, if the base currency for your report suite is USD, and you sell an item for 40 Euros, you should populate the currencyCode with "EUR" on the HTML page. As soon as SiteCatalyst receives the data it will use the current conversion rate to convert the 40 Euros to its USD equivalent.

Populating the currencyCode variable on the HTML page instead of in the JavaScript file is recommend if you sell in multiple currencies. If you would like to use your own conversion rates rather than the conversion rates used by Omniture, just set the currencyCode to equal the base currency of your report suite, and convert all revenue before sending it into SiteCatalyst.

Currency conversion applies to both revenue and any incrementor events, which are events that are used to sum values similar to revenue, like tax and shipping. The revenue and incrementor events are specified in the products string (refer to *products* in this document for more information). For more details on how currencies are treated in SiteCatalyst, refer to *Multi-Currency Support* in this document.

Syntax and Possible Values

```
s.currencyCode="currency_code"
```

Only the currency codes listed in *Multi-Currency Support* in this document are allowed.

Examples

```
s.currencyCode="GBP"  
s.currencyCode="EUR"
```

SiteCatalyst Configuration Settings

Omniture ClientCare can change the default currency setting for your report suite. When you change the base currency for a report suite, the existing revenue in the system is not converted, but all new revenue values will be converted accordingly.

Pitfalls, Questions and Tips

- If you notice surprisingly large amounts of revenue in SiteCatalyst reports, check that the currencyCode variable and base currency of the report suite are set correctly.
- If you are using incrementor events for non-currency purposes, be sure that the value of currencyCode matches the base currency of your report suite so that the values are sent straight through.

- When the currencyCode variable is empty, no conversion is applied.

2.1.14 dc

The dc variable allows you to select the data center – Dallas or San Jose – to which your data will be sent. If dc is not defined (and trackingServer is not defined), data goes to 112.2o7.net.

Table: dc Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	112

The data center is identified in the dc variable in order to match ActionSource.

2.1.15 doPlugins

doPlugins is a reference to the s_doPlugins function, and allows the s_doPlugins function to be called at the appropriate location within the JavaScript file. The s_doPlugins function is called each time any of the following occurs.

- The t() function is called
- The tl() function is called
- An exit or download link is clicked
- Any page element being tracked by ClickMap is clicked.

The doPlugins function is used to run customized routines to gather or alter data. If you are using an object name other than “s,” make sure that the s_doPlugins is renamed appropriately. For example, if your object name is s_mc, the s_doPlugins function should be called s_mc_doPlugins. Refer to the example below for further illustration.

Syntax and Possible Values

The s_doPlugins function should not be in quotes, and doPlugins should always be assigned to the exact name of the s_doPlugins function (if that function is renamed).

```
s.doPlugins=s_doPlugins;
```

Examples

```
s.doPlugins=s_doPlugins;  
s_mc.doPlugins=s_mc_doPlugins;
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- The only reason to change the object name (e.g. from s to s_mc) is if you share content with or pull content from other SiteCatalyst customers. Renaming the s_doPlugins function to s_mc_doPlugins ensures that another client's JavaScript file does not overwrite your doPlugins function.
- If you unexpectedly start pulling in content from another Omniture customer, and your s_doPlugins function is being overwritten, it is possible to simply rename the s_doPlugins function without changing the object

name. While the best solution is to use a different object name than other JavaScript files on the same page, doing so is not required. Contact your Implementation Consultant for assistance.

2.1.16 dynamicAccountList

The SiteCatalyst JavaScript file can be used to dynamically select a report suite to which it will send data. The dynamicAccountList variable contains the rules that will be used to determine the destination report suite.

Table: dynamicAccountList Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	""

This variable is used in conjunction with the dynamicAccountSelection and dynamicAccountMatch variables. The rules in dynamicAccountList will only be applied if dynamicAccountSelection is set to 'true,' and they apply to the section of the URL specified in dynamicAccountMatch.

If none of the rules in dynamicAccountList match the URL of the page, the report suite identified in s_account is used. The rules listed in this variable are applied in a left to right order. If the page URL matches more than one rule, the left-most rule will be used to determine the report suite. As a result, your more generic rules should be moved to the right of the list.

In the following examples, the page URL is `http://www.mycompany.com/path1/?prod_id=12345` and dynamicAccountSelection is set to 'true' and s_account is set to "mysuitecom."

Table: dynamicAccountList Variable Example

DynamicAccountList Value	DynamicAccountMatch Value	Report Suite to Receive Data
<code>mysuite2=www2.mycompany.com;mysuite1=mycompany.com"</code>	<code>window.location.host</code>	mysuite1
<code>"mysuite1=path4,path1;mysuite2=path2"</code>	<code>window.location.pathname</code>	mysuite1, mysuite2
<code>"mysuite1=path5"</code>	<code>window.location.pathname</code>	Mysuitecom, mysuite1
<code>"myprodsuite=prod_id"</code>	<code>window.location.search?window.location.search:"?"</code>	myprodsuite

Syntax and Possible Values

The dynamicAccountList variable is a semicolon separated list of name=value pairs (rules). Each piece of the list should contain the following items.

- one or more report suite ID (separated by commas)
- an equal sign
- one or more URL filters (comma-separated)

```
s.dynamicAccountList=rs1[,rs2]=domain1.com[,domain2.com/path][;...]
```

Only standard ASCII characters should be used in the string (no spaces).

Examples

```
s.dynamicAccountList="mysuite2=www2.mycompany.com;mysuite1=mycompany.com"  
s.dynamicAccountList="ms1,ms2=site1.com;ms1,ms3=site3.com"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- If the page URL matches multiple rules, the furthest rule on the left will be used.
- If no rules match, the default report suite is used.
- If your page is saved to someone's hard drive or translated via a web-based translation engine (like Google's translated pages) the dynamic account selection probably won't work. For more precise tracking, populate the `s_account` variable server-side.
- The `dynamicAccountSelection` rules only apply to the section of the URL specified in `dynamicAccountMatch`.
- When using dynamic account selection, be sure to update `dynamicAccountList` every time you obtain a new domain.
- Use the SiteCatalyst JavaScript Debugger when trying to identify the destination report suite. The `dynamicAccountSelection` variable will always override the value of `s_account`.

2.1.17 dynamicAccountMatch

The `dynamicAccountMatch` variable uses the DOM object to retrieve the section of the URL to which all rules in `dynamicAccountList` are applied. This variable is only valid when `dynamicAccountSelection` is set to 'True.' Since the default value is `window.location.host`, this variable is not required for Dynamic Account Selection to work. For additional information, refer to *dynamicAccountList* in this document.

The rules found in `dynamicAccountList` are applied to the value of `dynamicAccountMatch`. If `dynamicAccountMatch` only contains `window.location.host` (default), the rules in `dynamicAccountList` apply only to the domain of the page.

Table: dynamicAccountMatch Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	<code>window.location.host</code>

Syntax and Possible Values

The `dynamicAccountMatch` variable is usually populated by the Omniture Implementation Consultant who provides the SiteCatalyst JavaScript file. However, the values listed below may be applied at any time.

```
s.dynamicAccountMatch=[DOM object]
```

Table: dynamicAccountMatch Possible Values

Description	Value
Domain (default)	<code>window.location.host</code>
Path	<code>window.location.pathname</code>

Query String	(window.location.search?window.location.search:"?")
Domain and Path	window.location.host+window.location.pathname
Path and Query String	window.location.pathname+(window.location.search?window.location.search:"?")
Full URL	window.location.href

Examples

```
s.dynamicAccountMatch=window.location.pathname  
s.dynamicAccountMatch=window.location.host+window.location.pathname
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- When pages are saved to a hard drive, window.location.host is empty, thus causing those page views to be sent to the default report suite (in s_account).
- When a page is translated via a web-based translation engine, i.e. Google, the Dynamic Account Selection will not work as designed. For more precise tracking populate the s_account variable server-side.

2.1.18 dynamicAccountSelection

The dynamicAccountSelection variable enables you to dynamically select the report suite based on the URL of each page.



NOTE: dynamicAccountSelection does not work with custom link tracking.

Table: dynamicAccountSelection Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	False



NOTE: Both dynamicAccountList and dynamicAccountMatch will be ignored if the dynamicAccountSelection variable is not declared or set to 'false.'

Syntax and Possible Values

```
s.dynamicAccountSelection=[true|false]
```

Only 'true' and 'false' are allowed as values of dynamicAccountSelection.

Examples

```
s.dynamicAccountSelection=true  
s.dynamicAccountSelection=false
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

Always use the SiteCatalyst JavaScript Debugger to determine which report suite is receiving data from each page.

2.1.19 dynamicVariablePrefix

Allows deployment to flag variables which should be populated dynamically. Cookies, request headers and image query string parameters are available to be populated dynamically.

Table: dynamicVariablePrefix Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	D=	Any	D=

Syntax and Possible Values

s.prop1="D=User-Agent"

OR USE CUSTOM FLAG FOR DYNAMIC VARIABLES

s.dynamicVariablePrefix=".."

Examples

s.prop1="D=User-Agent"

OR USE CUSTOM FLAG FOR DYNAMIC VARIABLES

s.dynamicVariablePrefix=".."

s.prop1="..User-Agent"

Pitfalls, Questions, and Tips

- See the *Mobile Implementation Guide* for more information on dynamic variables.
- Dynamic variables can be used to significantly reduce the total length of the URL by copying values into other variables.
- Dynamic variables can be used to collect data from headers and cookies not otherwise available for data collection.

2.1.20 eVarN

eVar variables are used for building custom reports within SiteCatalyst's Conversion Module. When an eVar is set to a value for a visitor, SiteCatalyst remembers that value until it expires. Any success events that a visitor encounters while the eVar value is active are counted toward the eVar value. For more information, refer to *What is an eVar?* In this document.

Table: eVarN Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
255 Bytes	V1-v50	Conversion > Custom eVars > Custom eVar 1-50	""

Contact Omniture ClientCare if you would like access to more eVar reports in SiteCatalyst. If you would like to use more than 20 eVars, make sure you are using JavaScript code version G.8 or higher.

Expiration

eVars expire at a time period you specify, and once the eVar expires, it no longer receives credit for success events. eVars can also be configured to expire on success events. For example, if you have an internal promotion that expires at the end of a visit, the internal promotion will only receive credit for purchases or registrations (for example) that occur during the visit in which they were activated.

There are two ways to expire an eVar. First, you can set the eVar to expire after a specified time period or event, or second, you can also use SiteCatalyst to force the expiration of an eVar, which is useful when repurposing a variable. If an eVar is used in May to reflect internal promotions and expires after 21 days, and in June it is used to capture internal search keywords, then on June 1st, you should force the expiration of, or reset, the variable. Doing so will help keep internal promotion values out of June's reports.

Case Sensitivity

eVars are case insensitive, but they are displayed in the capitalization of the first occurrence. For example, if the first instance of eVar1 is set to "Logged In," but all subsequent instances are passed as "logged in," SiteCatalyst reports will always show "Logged In" as the value of the eVar.

Counters

While eVars are most often used to hold string values, they may also be configured to act as counters. eVars are useful as counters when you are trying to count the number actions a user takes before an event. For example, you may use an eVar to capture the number of internal searches before purchase. Each time a visitor searches, the eVar should contain a value of '+1.' If a visitor searches four times before a purchase, you will see an instance for each total count: 1.00, 2.00, 3.00 and 4.00. However, only the 4.00 will receive credit for the purchase event (Orders and Revenue Metrics). Only positive numbers are allowed as values of an eVar counter.

Subrelations

Finally, a common requirement for a Custom eVar report is the ability to break down one Custom eVar report by another. For example, if one eVar contains gender, and another contains salary, you may ask the following question: of the female visitors to my site, how much revenue was generated by women who make more than \$50,000 per year. Any eVar that is fully sub-related allows this type of break down in SiteCatalyst reports. For example, if the gender eVar has full subrelations enabled, all other custom eVar reports can be broken down by gender, and gender can be broken down by all others. Thus, to see the relationship between two reports, only one of them needs full subrelations enabled. By default, Campaigns, Products, and Category reports are fully sub-related (any eVar can be broken down by campaign or products).

Visits and Visitors

In order to see the number of Visits or Daily Unique Visitors associated with a conversion variable, Omniture ClientCare will need to enable it in SiteCatalyst.

Syntax and Possible Values

While eVars may be renamed within SiteCatalyst, they should always be referred to in the JavaScript file by eVarX, where X is a number between 1 and 50.

```
s.eVarX="value"
```

When not used as a counter, eVars have the same limitations as all other variables. If the eVar is a “counter,” it is expected to receive numeric values like “1” or “2.5.” If more than two decimal places are given, the eVar counter will round to two decimal places. An eVar counter may NOT contain negative numbers.

Examples

```
s.eVar1="logged in"  
s.eVar23="internal spring promo 4"
```

SiteCatalyst Configuration Settings

eVars can be configured in the Admin Console. All eVars can be configured with a Name, Type, Allocation, Expire After Setting, or Reset. Each configuration setting will be addressed separately.

Table: events Variable Parameters

Setting	Description
Name	Name allows you to change the name of the eVar report within SiteCatalyst. NOTE: The eVar should still be referenced as s.eVarX in the JavaScript code, no matter what name is given to the report in SiteCatalyst.
Type	The Type Drop-Down Box allows you to show whether the eVar is a Text String or Counter.
Allocation	Allocation is used to configure which value of the eVar will receive credit for success events. If Allocation is set to “Most Recent (Last),” then B receives credit. If Allocation is set to “Original Value (First)” then A receives credit. If Allocation is set to “Linear”, then both A and B receive credit for half the purchase value.
Expire After	Expire After allows you to determine whether an eVar expires on a specific event, like purchase, or after a custom or predefined time period.
Reset	By selecting the Reset Checkbox for an eVar, and clicking Save at the bottom of the page, all values of that eVar will be immediately expired. Once this happens, only new values of the eVar will receive credit for success events.

Pitfalls, Questions and Tips

- Unlike prop variables, eVar variables are not allowed to be lists of delimited values. If you populate an eVar with a list of values, for example “one,two,three” then that exact string will appear in the SiteCatalyst reports.
- eVar counters may not contain negative numbers.

2.1.21 events

The events variable is used to record common shopping cart success events as well as custom success events.

Table: Configuration Settings

Max Size	Debugger Parameter	Reports Populated	Default Value
No Limit	events	Shopping Cart Events Custom Events	N/A

An event should be considered a "milestone" within a site. Success events are most commonly populated on the final confirmation page of a process such as a registration process or newsletter sign-up. Custom events are defined by populating the events variable with the literal values defined in the Possible Values section below.

By default, success events are configured as "Auto Incrementor" events. Auto Incrementor events simply count the number of times a success event is set (x+1). Events can also be configured as "Custom Incrementors" that allow you to specify the number to increment. For additional information on using Custom Incrementor events, refer to *Products* in this document.

Event Serialization

An event is counted every time the event is set. Event serialization may be used to de-duplicate events.

Syntax

```
s.events="event1[,event2]"
```

Examples

```
s.events="scAdd"  
s.events="scAdd,event1"
```

Possible Values

The following is a list of possible values for the events variable.

Table: Possible Values

Event	Description	Reports Populated
prodView	Product Views	Products
scOpen	Open / Initialize a new shopping cart	Carts
scAdd	Add item(s) to the shopping cart	Cart Additions
scRemove	Remove item(s) from the shopping cart	Cart Removals
scView	View shopping cart	Cart Views
scCheckout	Beginning of the checkout process	Checkouts
purchase	Completion of a purchase (order)	Orders
event1 – event20	Custom events	Custom Events

Configuring the Variable in SiteCatalyst

Events can be configured as Auto Incrementor events or Custom Incrementor events. This configuration is done by Omniture ClientCare.

2.1.22 fpCookieDomainPeriods

The fpCookieDomainPeriods variable is for cookies set by JavaScript (s_sq, s_cc, plug-ins). cookieDomainPeriods should never be dynamically set and must be correct for the collection domain (2o7.net or CNAME). If you use cookieDomainPeriods, it is good practice to specify a value for fpCookieDomainPeriods. fpCookieDomainPeriods inherits the cookieDomainPeriods value.

Table: fpCookieDomainPeriods Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	cookieDomainPeriods

// sample code for setting cookie domain variables

```
s.cookieDomainPeriods="3"
s.fpCookieDomainPeriods="2"
var d=window.location.hostname
if(d.indexOf('.co.uk')>-1||d.indexOf('.com.au')>-1)
  s.fpCookieDomainPeriods="3"
```

The name "cookieDomainPeriods" refers to the number of periods in the domain when the domain begins with "www". For example, www.mysite.com contains two periods (.), while www.mysite.co.jp contains three periods. Another way to describe the variable is the number of sections in the main domain of the site (two for mysite.com and three for mysite.co.jp).

The SiteCatalyst JavaScript file uses the cookieDomainPeriods variable to determine the domain with which to set cookies. There are at least two cookies affected by this variable, including s_sq and s_cc (used for ClickMap and cookie checking respectively). Cookies used by plugins like getValOnce will also be affected. For more information on SiteCatalyst cookies, refer to *SiteCatalyst Cookies* in this document. For more information on plug-ins that use cookies, refer to *JavaScript Plug-ins and Settings* in this document.

If cookieDomainPeriods is set to "2," but the domain of the site is mysite.co.jp, the JavaScript file will attempt to set a cookie with the domain "co.jp," which the browser will reject. In that case, the JavaScript will fail to set any cookies, causing ClickMap data to be lost, and the Traffic > Technology > Cookies Report to report that almost 100% of visitors rejected first party cookies.

On the other hand, if cookieDomainPeriods is set to "3," and the domain of the site is www2.mysite.com, the JavaScript file will successfully set a cookie with www2.mysite.com. However, when a visitor goes to www4.mysite.com, all cookies set with www2.mysite.com will be lost, thus losing ClickMap and some plug-in data.

For sites that have multiple domains, some of which contain more than two sections, the following pseudo-code may be used to set cookieDomainPeriods correctly.

```
s.cookieDomainPeriods="2"
if(window.location.indexOf(".co.jp")>-1 ||
  window.location.indexOf(".com.au")>-1)
  s.cookieDomainPeriods="3"
```

Using a value of “4” or higher may have a negative effect on the visitor ID cookie set by Omniture’s servers. If you would like to have cookieDomainPeriods set to “4” or higher, contact Omniture ClientCare about using the visitorNamespace variable, which will solve the problem.

If cookieDomainPeriods is higher than the number of sections in the domain, cookies will be set with the full domain. For example, if cookieDomainPeriods is “3” and a visitor is at http://mysite.com, cookies will be set at mysite.com. However, this can cause data loss when moving from www.mysite.com to search.mysite.com. Cookies set on www.mysite.com will not be shared with search.mysite.com and vice versa.

Syntax and Possible Values

The cookieDomainPeriods variable is expected to be a string, as shown below.

```
s.cookieDomainPeriods="3"
```

Omniture recommends using a value of “2” or “3” to match the number of top level domain sections.

Examples

```
s.cookieDomainPeriods="3"  
s.cookieDomainPeriods="2"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- If you notice that ClickMap data is absent, or that the Traffic > Technology > Cookies Report shows a large percentage of visitors who reject cookies, check that the value of cookieDomainPeriods is correct.
- If cookieDomainPeriods is higher than the number of sections in the domain, cookies will be set with the full domain, which can cause data loss as visitors switch between sub domains.

2.1.23 hierN

The hierarchy variable is used to determine the location of a page in your site’s hierarchy. This variable is most useful for sites that have more than three levels in the site structure. For example, a media site may have 4 levels to the Sports section: Sports, Local Sports, Baseball, Red Sox. If someone visits the Baseball page, then Sports, Local Sports and Baseball will all reflect that visit. For additional information, refer to the *Channels and Hierarchies* white paper.

Table: hierN Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
255 Bytes	H1-H5	Traffic > Hierarchies > Hierarchy N	""

There are five hierarchy variables available, which must be enabled by Omniture ClientCare. At the time the hierarchy is enabled, you will need to decide on a delimiter for the variable and the maximum number of levels for the hierarchy. For example, if the delimiter is a comma, the sports hierarchy may display as follows.

```
s.hier1="Sports,Local Sports,Baseball"
```

Make sure that none of your section names have the delimiter in them. For example, if one of your sections is called "Coach Griffin, Jim," then you should choose a delimiter other than comma. Each hierarchy section is limited to 255 bytes, while the total variable limit is 255 bytes. Once a delimiter is chosen (at the time the hierarchy is created) it is not easily changed. Please contact Omniture ClientCare about changing the delimiter for an existing hierarchy. Delimiters may also consist of multiple characters, like || or /\, which are less likely to appear in a hierarchy section.

Syntax and Possible Values

Do not put a space between each delimiter. In the following example syntax, N is a number between one and five.

```
s.hierN="Level 1[<delimiter>Level 2[<delimiter>Level 3[...]]]"
```

Do not use the delimiter except to delimit the levels of the hierarchy. The delimiter may be any character or characters of your choice.

Examples

```
s.hier1="Toys|Boys 6+|Legos|Super Block Tub"  
s.hier4="Sports/Local Sports/Baseball"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Before implementing hierarchies, refer to the *Channels and Hierarchies* white paper.
- The delimiter may not be changed once the hierarchy is setup. If the delimiter for your hierarchy must be changed, contact Omniture ClientCare.
- The number of levels may not be changed once the hierarchy is set up.



NOTE: Any changes to hierarchies require a PSR.

2.1.24 homepage

The homepage variable, in Internet Explorer, indicates whether the current page is set as the user's home page. This variable is populated after the page code and before doPlugins is run.



WARNING! This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: homepage Variable Parameters

Query Param	Value	Example	Reports Affected
hp	Y or N	Y	Traffic > Technology > Home Page

2.1.25 javaEnabled

The javaEnabled variable indicates whether Java is enabled on the browser. This variable is populated after the page code and before doPlugins is run.



WARNING!: This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: javaEnabled Variable Parameters

Query Param	Value	Example	Reports Affected
v	Y or N	Y	Traffic > Technology > Java

2.1.26 javascriptVersion

The javascriptVersion variable indicates the version of JavaScript supported by the browser. This variable is populated after the page code and before doPlugins is run.



WARNING!: This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: javascriptVersion Variable Parameters

Query Param	Value	Example	Reports Affected
j	1.0, 1.1, 1.2, ... 1.7	1.7	Traffic > Technology > JavaScript Version



NOTE: Version H.10 and higher of the JavaScript file accurately detect up to version 1.7 (the highest version at the time H.10 was released). Prior versions of the JavaScript file only detected up to version 1.3.

2.1.27 linkDownloadFileTypes

The linkDownloadFileTypes variable is a comma-separated list of file extensions. If your site contains links to files with any of these extensions, the URLs of these links will appear in the File Downloads Report. For additional information, refer to *Download Links* in this document.

Table: linkDownloadFileTypes Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Traffic > Site Traffic > File Downloads	"exe,zip,wav,mp3,mov,mpg,avi,wmv,doc,pdf,xls"



NOTE: The linkDownloadFileTypes variable is only relevant when trackDownloadLinks is set to 'True.'

Only left-mouse-clicks on a link are counted in the File Downloads Report. Thus, all file downloads that start automatically when a page loads, or that are only downloaded after a redirect, are NOT counted in the File Downloads Report. In addition, when you right-click a file and select the "Save Target As..." Option, it is not counted in the File Downloads Report.

The `linkDownloadFileTypes` Variable may be used to track clicks to RSS feeds. If you have links to RSS feeds with a .xml or other extension, appending “.xml” to the `linkDownloadFileTypes` list allows you to see how often each RSS link is clicked.

Syntax and Possible Values

Only include file extensions (no spaces).

```
s.linkDownloadFileTypes="type1[,type2[,type3[...]]]"
```

Any file extension may be included in the list. Be careful not to include a common file extension, like htm or aspx, in `linkDownloadFileTypes`. Doing so will cause an extra image request to be sent to SiteCatalyst for each click, which will be billed as a primary server call.

Examples

```
s.linkDownloadFileTypes="exe,zip,wav,mp3,mov,mpg,avi,wmv,doc,pdf,xls"
s.linkDownloadFileTypes="exe,zip,wav,mp3,mov,mpg,avi,wmv,doc,pdf,xls,xml"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Only left-clicks on download files will cause the URL to appear in the File Downloads Report.
- Including a common file extension in `linkDownloadFileTypes` may significantly increase the total server calls sent to Omniture’s servers.
- Links to server-side redirects or HTML pages that automatically begin downloading a file are not counted unless the file extension is in `linkDownloadFileTypes`.
- Links that use JavaScript (i.e., `javascript:openLink()`) will not be counted in file downloads.

2.1.28 linkExternalFilters

If your site contains many links to external sites, and you don’t want to track ALL exit links, `linkExternalFilters` can be used to report on a specific subset of exit links.

Table: `linkExternalFilters` Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Paths > Entries & Exits > Exit Links	""

`linkExternalFilters` is an optional variable used in conjunction with `linkInternalFilters` to determine whether a link is an exit link. An exit link is defined as any link that takes a visitor away from your site. Whether the target window of an exit link is a popup or the existing window will not affect whether the link appears in the exit links report. Exit links will only be tracked if `trackExternalLinks` is set to ‘true.’ The filters in `linkExternalFilters` and `linkInternalFilters` are case insensitive.



NOTE: If you don’t want to use `linkExternalFilters`, just delete it or set it to “”.

The filters list in `linkExternalFilters` and `linkInternalFilters` apply to the domain and path of any link by default. If `linkLeaveQueryString` is set to 'true,' then the filters apply to the entire URL (domain, path and query string). These filters are always applied to the absolute path of the URL, even if a relative path is used as the href value.

Most companies find that `linkInternalFilters` gives them enough control over exit links that they don't need `linkExternalFilters`. Using `linkExternalFilters` simply decreases the likelihood that an exit link is considered external. If `linkExternalFilters` has a value, then a link is only considered external if it does NOT match `linkInternalFilters` and DOES match `linkExternalFilters`. The following example illustrates how this variable is used. In this example, the URL of the page is `http://www.mysite.com/index.html`.

```
s.trackExternalLinks=true
s.linkInternalFilters="javascript:,mysite.com"
s.linkExternalFilters="site1.com,site2.com,site3.com/partners"
s.linkLeaveQueryString=false
...
<a href="http://www.mysite.com">Not an Exit Link</a>
<a href="/careers/job_list.html">Not an Exit Link</a>
<a href="http://www2.site3.com">Not an Exit Link</a>
<a href="http://www.site1.com">Exit Link</a>
<a href="http://www2.site3.com/partners/offer.asp">Exit Link</a>
```

Syntax and Possible Values

`linkExternalFilters` is a comma-separated list of ASCII characters. No spaces are allowed.

```
s.linkExternalFilters="site1.com[,site2.com[,site3.net[...]]]"
```

Any portion of a URL may be included in `linkExternalFilters`, separated by commas.

Examples

```
s.linkExternalFilters="partnersite.com,partnertwo.net/path/"
s.linkExternalFilters=""
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Using `linkExternalFilters` will make fewer links on your site Exit links. It is NOT used to “trump” `linkInternalFilters`, or force some links that are internal to become exit links.
- If `linkExternalFilters` should be applied to the query string of a link, make sure `linkLeaveQueryString` is set to 'true.' Please refer to `linkLeaveQueryString` in this document before setting to 'true.'
- To disable exit link tracking, set `trackExternalLinks` to 'false.'

2.1.29 linkInternalFilters

The `linkInternalFilters` variable is used to determine which links on your site are exit links. It is a comma-separated list of filters that represent the links that are part of the site.

Table: linkInternalFilters Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Paths > Entries & Exits > Exit Links	"javascript:"

linkInternalFilters is used to determine whether a link is an exit link which is defined as any link that takes a visitor away from your site. Whether the target window of an exit link is a popup, or the existing window does not affect whether the link appears in the exit links report, exit links will only be tracked if trackExternalLinks is set to 'true.' The filters in linkInternalFilters are not case-sensitive.

The list of filters in linkInternalFilters applies to the domain and path of any link by default. If linkLeaveQueryString is set to 'true,' then the filters apply to the entire URL (domain, path and query string). The filters are always applied to the absolute path of the URL, even if a relative path is used as the href value.

Be careful that ALL the domains of your site (and any partners who will be using your JavaScript file) are included in linkInternalFilters. If you don't have all domains included in the list, then all links on and to those domains will be considered exit links, thus increasing the server calls sent to SiteCatalyst. If you would like multiple domains or companies to use a single SiteCatalyst JavaScript file, you may consider populating linkInternalFilters on the page, thus overriding the value specified in the JavaScript file. On the other hand, if you have vanity domains that immediately redirect to your main domain, those vanity domains do not need to be included in the list.

The following example illustrates how this variable is used. In this example, the URL of the page is <http://www.mysite.com/index.html>.

```
s.trackExternalLinks=true
s.linkInternalFilters="javascript:,mysite.com"
s.linkExternalFilters=""
s.linkLeaveQueryString=false
...
<a href="http://www.mysite.com">Not an Exit Link</a>
<a href="/careers/job_list.html">Not an Exit Link</a>
<a href="http://www2.site3.com">Exit Link</a>
<a href="http://www2.site1.com/partners/">Exit Link</a>
```

Syntax and Possible Values

linkInternalFilters is a comma-separated list of ASCII characters. No spaces are allowed.

```
s.linkInternalFilters="javascript:,site1.com[,site2.com[,site3.net[...]]]"
```

The default value of linkInternalFilters contains "javascript:" because any links to a JavaScript function usually are not exit links. If "javascript:" were not in the list then all links to javascript functions would be considered external.

Examples

```
s.linkInternalFilters="javascript:,mysite.com"
s.linkInternalFilters="javascript:,mysite.com,mysite.net,vanity1.com"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Include all domains that the SiteCatalyst JavaScript file may be served under in the filter list.
- Periodically check the Paths > Entries & Exits > Exit Links report to make sure that none of the entries in that report are incorrect.

2.1.30 linkLeaveQueryString

By default, query strings are excluded from all SiteCatalyst reports. For some Exit Links and Download Links the important portion of the URL may be in the query string, as shown in the following sample URL.

`http://www.mycompany.com/download.asp?filename=myfile.exe`

The download file name may be defined in the query string and, consequently, the query string is necessary to make the File Downloads Report more accurate.

The linkLeaveQueryString variable determines whether or not the query string should be included in the Exit Links and File Download reports.

Table: linkLeaveQueryString Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Exit Links File Downloads	false



CAUTION: Setting linkLeaveQueryString=true will include all query string parameters for all Exit Links and Download Links

Syntax

```
s.linkLeaveQueryString=[false/true]
```

Examples

```
s.linkLeaveQueryString=false
```

Possible Values

```
s.linkLeaveQueryString=false
```

```
s.linkLeaveQueryString=true
```

Configuring the Variable in SiteCatalyst

No SiteCatalyst configuration is necessary for this variable.

Pitfalls, Questions and Tips

- Setting s.linkLeaveQueryString=true will include all query string parameters for all Exit Links and Download Links
- linkLeaveQueryString does not affect recorded page URLs, ClickMap or Path reports

2.1.31 linkName

linkName is an optional variable used in Link Tracking that determines the name of a custom, download, or exit link. linkName is not normally needed because the third parameter in the tl() function replaces it.

Table: linkName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	pev2	File Downloads Custom Links Exit Links	""

Custom Links refer to links that send data to SiteCatalyst. The linkName variable (or the third parameter in the tl() function) is used to identify the value that will appear in the Custom, Download or Exit Links Report. If linkName is not populated, the URL of the link will appear in the report.

Syntax and Possible Values

```
s.linkName="Link Name"
```

There are no limitations on linkName outside of the standard variable limitations.

Examples

```
s.linkName="Nav Bar Home Link"  
s.linkName="Partner Link to A.com"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- The linkName variable is replaced by the third parameter in the tl() function.
- If the linkName variable and the third parameter in the tl() function are blank, the full URL of the link (with the exception of the query string) will appear in the report (even if the link is relative).

2.1.32 linkTrackEvents

linkTrackEvents is a comma-separated list of events that will be sent with a custom, exit, or download link. If an event is not in linkTrackEvents, then it will not be sent to SiteCatalyst, even if it is populated in the onClick event of a link, as shown in the following example.

```
s.linkTrackVars="events"  
s.events="event1,event2"  
s.t() // both event1 and event2 are recorded  
<a href="help.php" onClick="s=s_gi('rs1');s.tl(this,'o')">event1 is  
    recorded</a>  
<a href="test.php"  
    onClick="s=s_gi('rs1');s.events='event2';s.tl(this,'o')">No events  
    are recorded</a>
```

In the first link to help.php, notice that the events variable retains the value that was set before the link was clicked, thus allowing event1 to be sent with the custom link. In the second example, the link to test.php, event2 is not recorded because it is not listed in linkTrackEvents.

To avoid confusion and potential problems, Omniture recommends populating linkTrackVars and linkTrackEvents in the onClick event of a link that is used for Link Tracking.

linkTrackEvents contains the events that should be sent with custom, download and exit links. This variable is only considered if linkTrackVars contains "events."

Table: linkTrackEvents Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Conversion	"None"

Syntax and Possible Values

linkTrackEvents is a comma-separated list of events (no spaces).

```
s.linkTrackEvents="event1[,event2[,event3[...]]]"
```

Only event names are allowed in linkTrackEvents. These events are listed in *Events* in this document. If a space appears before or after the event name, the event will not be sent with any link image requests.

Examples

```
s.linkTrackEvents="purchase,event1"  
s.linkTrackEvents="scAdd,scCheckout,purchase,event14"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- The JavaScript file will only use linkTrackEvents if linkTrackVars contains the "events" variable.
- Beware that if an event is fired on a page, and is listed in linkTrackEvents, that event will be recorded again with any exit, download or custom links unless the events variable is reset prior to that event (in the onClick of a link or after the call to the t() function).
- If linkTrackEvents contains spaces between event names, the events will not be recorded.

2.1.33 linkTrackVars

linkTrackVars is a comma-separated list of variables that will be sent with custom, exit and download links. If linkTrackVars is set to "" all variables that have values will be sent with link data. To avoid inflation of instances or page views associated with other variables, Omniture recommends populating linkTrackVars and linkTrackEvents in the onClick event of a link that is used for link tracking.

All variables that should be sent with link data (custom, exit and download links) should be listed in linkTrackVars. If linkTrackEvents is used, linkTrackVars should contain "events."

Table: linkTrackVars Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	Any	"None"

When populating linkTrackVars, do not use the 's.' prefix for variables. For example, instead of populating linkTrackVars with "s.prop1," you should populate it with "prop1." The following example illustrates how linkTrackVars should be used.

```
s.linkTrackVars="eVar1,events"
s.linkTrackEvents="event1"
s.events="event1"
s.eVar1="value A"
s.eVar2="value B"
s.t() // eVar1, event1 and event2 are recorded
<a href="http://google.com">event1 and eVar1 are recorded</a>
<a href="test.php" onClick="s=s_gi('rs1');s.eVar1='value
C';s.events='';s.tl(this,'o')">eVar1 is recorded</a>
```

Since the link to google.com is an exit link (unless you are Google), event1 and eVar1 will be sent with the exit link data, thus increasing the instances associated with eVar1 and the number of times event1 is fired. In the link to test.php, eVar1 is sent with a value of 'value C' because that is the current value of eVar1 at the time that tl() is called.

Syntax and Possible Values

linkTrackVars is a case-sensitive, comma-separated list of variable names, without the object name prefix. In other words, use 'eVar1' instead of 's.eVar1.'

```
s.linkTrackVars="variable_name[,variable_name[...]]"
```

linkTrackVars may only contain variables that are sent to SiteCatalyst, namely: events, campaign, purchaseID, products, eVar1-50, prop1-50, hier1-5, channel, server, state, zip, and pageType.

Examples

```
s.linkTrackVars="events,prop1,eVar49"
s.linkTrackVars="products"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- If linkTrackVars is blank, ALL variables that have values will be sent to SiteCatalyst with all server calls.
- Any variable listed in linkTrackVars that has a value at the time of any download, exit or custom link, will be sent to SiteCatalyst.
- If linkTrackEvents is used, linkTrackVars must contain "events."
- Do not use the "s." or "s_objectname." prefix for variables.

2.1.34 linkType

linkType is an optional variable used in link tracking that determines which report a Link Name or URL will appear (Custom, Download or Exit Links). linkType is not normally needed because the second parameter in the tl() function replaces it.

Table: linkType Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
One character	pe=[lnk_o lnk_d lnk_e]	File Downloads Custom Links Exit Links	""

Custom links send data to SiteCatalyst. The linkType variable (or the second parameter in the tl() function) is used to identify the report in which the link name or URL will appear (Custom, Download or Exit Links Report).

For Exit and Download Links, the linkType variable is automatically populated depending on whether the link clicked is an Exit or Download Link. A custom link may be configured to send data to any of the three reports with this variable or with the second parameter in the tl() function. By setting linkType to 'o', 'e' or 'd', the linkName or link URL will be sent to the Custom Links, Exit Links or File Downloads Report respectively.

Syntax and Possible Values

linkType may only contain a single character, namely 'o', 'e', or 'd.'

```
s.tl(this,'o','Link Name');
```

Examples

```
<a href="index.html" onClick="
  var s=s_gi('rsid'); **see note below on the rsid**
  s.tl(this,'o','Link Name');
">My Page</a>
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- If linkType is not specified, custom links ('o') is assumed.

2.1.35 mediaLength

This variable specifies the total length of the media being played.

Table: mediaLength Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
No max size for entire pev3 request – size is limited to browser's URL	pev3	Time Spent on Video; Video Segments Viewed	None

length limit.			
---------------	--	--	--

Syntax and Possible Values

autoTrack Method:

If using `s.Media.autoTrack`, the `mediaLength` does not need to be implemented explicitly. It will be determined automatically by the SiteCatalyst JavaScript code.

Manual Tracking Method:

Syntax: `s.Media.open(mediaName,mediaLength,mediaPlayerName)`

Possible Values: `s.Media.open("de_bofr_1045Making_400k", "414", "Windows Media Player 11.0.5721.5230")`

Examples

`s.Media.open("de_bofr_1045Making_400k", "414", "Windows Media Player 11.0.5721.5230")`

Resulting `pev3` parameter syntax: `pev3= [Asset Name]--**--[Total length of asset]--**--[Player name]--**--[Total seconds consumed]--**--[Timestamp]--**--[Chronological record of all starts and stops along with accompanying markers]`

Possible `pev3` values: `pev3=de_bofr_1045Making_400k--**--414--**--Windows Media Player 11.0.5721.5230--**--288--**--1207893838--**--S0E0S0E256S0E32`

Pitfalls, Questions, and Tips

- Only need to call the media tracking methods if player cannot be tracked using `s.Media.autoTrack = true` (see Media Tracking User Guide).
- If not tracking using `autoTrack`, be sure to set the length in seconds.

2.1.36 mediaName

This variable specifies the name of the video or media item. Additionally, it is only available via the Data Insertion API and Full Processing Data Source.

Table: mediaName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 Bytes	pev3	Videos; Next Video Flow; Previous Video Flow; Video Segments Viewed; Time Spent on Video	None

Syntax and Possible Values

autoTrack Method:

If using `s.Media.autoTrack`, the `mediaName` does not need to be implemented explicitly. It will be determined automatically by the SiteCatalyst JavaScript code. Manual Tracking Method:

Syntax:

```
s.Media.open(mediaName,mediaLength,mediaPlayerName)
```

```
s.Media.play(mediaName,mediaOffset)
```

```
s.Media.stop(mediaName,mediaOffset)
```

```
s.Media.close(mediaName)
```

Possible Values:

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

```
s.Media.play("de_bofr_1045Making_400k", "0")
```

```
s.Media.play("de_bofr_1045Making_400k", "414")
```

```
s.Media.close("de_bofr_1045Making_400k")
```

Examples

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

```
s.Media.play("de_bofr_1045Making_400k", "0")
```

```
s.Media.play("de_bofr_1045Making_400k", "414")
```

```
s.Media.close("de_bofr_1045Making_400k")
```

Resulting pev3 parameter syntax: pev3=[Asset Name]--**--[Total length of asset]--**--[Player name]--**--[Total seconds consumed]--**--[Timestamp]--**--[Chronological record of all starts and stops along with accompanying markers]

Possible pev3 Values: pev3=de_bofr_1045Making_400k--**--414--**--Windows Media Player 11.0.5721.5230--**--288--**--1207893838--**--S0E0S0E256S0E32

Pitfalls, Questions, and Tips

- Only need to call the media tracking methods if player cannot be tracked using s.Media.autoTrack = true (refer to the *Media Tracking User Guide*).

2.1.37 mediaPlayer

This variable specifies the player used to consume a video or media item.

Table: mediaName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 Bytes	pev3	Video Players	None

Syntax and Possible Values

autoTrack Method:

```
s.Media.playerName = "My Custom Player Name" //configure player name in global JavaScript or ActionSource
```

Manual Tracking Method:

```
s.Media.open(mediaName,mediaLength,mediaPlayerName)
```

Possible Values:

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

Examples

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

Resulting pev3 parameter syntax: pev3=[Asset Name]--**--[Total length of asset]--**--[Player name]--**--[Total seconds consumed]--**--[Timestamp]--**--[Chronological record of all starts and stops along with accompanying markers]

Possible pev3 Values: pev3=de_bofr_1045Making_400k--**--414--**--Windows Media Player 11.0.5721.5230--**--288--**--1207893838--**--S0E0S0E256S0E32Pitfalls, Questions, and Tips

Pitfalls, Questions, and Tips

Only need to call the media tracking methods if player cannot be tracked using s.Media.autoTrack = true (refer to the *Media Tracking User Guide*).

2.1.38 mediaSession

This variable specifies the segments of a video or media asset consumed.

Table: mediaName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
255 Bytes	pev3	Time Spent on Video; Video Segments Viewed	None

Syntax and Possible Values

autoTrack Method:

If using s.Media.autoTrack, the mediaName does not need to be implemented explicitly. It will be determined automatically by the SiteCatalyst JavaScript code.

Manual Tracking Method:

Syntax:

```
s.Media.open(mediaName,mediaLength,mediaPlayerName)
```

```
s.Media.play(mediaName,mediaOffset)
```

```
s.Media.stop(mediaName,mediaOffset)
```

Possible Values:

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

```
s.Media.play("de_bofr_1045Making_400k", "0")
```

```
s.Media.play("de_bofr_1045Making_400k", "414")
```

Examples

```
s.Media.open("de_bofr_1045Making_400k", "414","Windows Media Player 11.0.5721.5230")
```

```
s.Media.play("de_bofr_1045Making_400k", "0")
```

```
s.Media.play("de_bofr_1045Making_400k", "414")
```

Resulting pev3 parameter syntax: pev3=[Asset Name]--**--[Total length of asset]--**--[Player name]--**--[Total seconds consumed]--**--[Timestamp]--**--[Chronological record of all starts and stops along with accompanying markers]

Possible pev3 Values: pev3=de_bofr_1045Making_400k--**--414--**--Windows Media Player 11.0.5721.5230--**--288--**--1207893838--**--S0E0S0E256S0E32

Pitfalls, Questions, and Tips

Only need to call the media tracking methods if player cannot be tracked using s.Media.autoTrack = true (see media tracking whitepaper).

2.1.39 Media.trackEvents

The Media.trackEvents variable identifies which events should be sent with a media hit. Additionally, it is only applicable with JavaScript and ActionSource.

Table: mediaName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	s.Media.trackEvents="None"

Syntax and Possible Values

Event names such as event1, purchase

Examples

```
s.Media.trackEvents="event1,purchase"
```

Pitfalls, Questions, and Tips

Make sure to populate trackVars with "events" whenever this variable is populated.

2.1.40 Media.trackVars

The Media.trackVars variable identifies which variables should be sent with a media hit. Additionally, it is only applicable with JavaScript and ActionSource.

Table: mediaName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	s.Media.trackVars="None"

Syntax and Possible Values

Variable names such as propN, eVarN, events, channel, etc.

Examples

```
s.Media.trackVars="prop2,events,eVar3"
```

Pitfalls, Questions, and Tips

- Even if eVar3 is specified in trackVars, it will be sent with the media hit.

2.1.41 mobile

Controls the order in which cookies and subscriber ids are used to identify visitors.

Table: mobile Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	/5/ or /1/ in path of image url	N/A	None

Syntax and Possible Values

```
s.mobile="any_string" //subscriber id used first, produces /5/ in path of image url
```

```
s.mobile="" //cookies used first, produces /5/ in path of image url
```

Examples

Pitfalls, Questions, and Tips

Changing the order cookies and subscriber ids are used for identifying visitors by adjusting s.mobile variable will cliff visitors which have both.

2.1.42 s_objectID

The s_objectID variable is a global variable that should be set in the onClick event of a link. By creating a unique object ID for a link or link location on a page, you can either improve ClickMap tracking or use ClickMap to report on a link type or location, rather than the link URL.

Table: s_objectID Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 Bytes	OID	ClickMap	The Absolute URL of a Clicked Link

There are three common reasons to use s_objectID.

- To aggregate clicks on links that change often during a day
- To separate clicks on links that ClickMap combines
- To improve accuracy of ClickMap data reporting

Each reason for using s_objectID is addressed briefly here. For additional information on ClickMap, refer to *ClickMap* in this document.

Aggregate Clicks on Highly Dynamic Links

If your site is highly dynamic, and links on some pages change throughout the day, s_objectID may be used to identify the location of a link on the page. If s_objectID is set to "top left 1" or "top left 2," which represents the first link in the top left of the page, for example, then all links that appear in that location (or that have s_objectID set to the same value) will be reported together with ClickMap. If you don't use s_objectID, you will see the number of times that a specific link was clicked, but you will lose insight into how all the other links in that location were used by visitors to your site.

Separate Clicks Combined

If the pageName variable on your site is used to show the section or template a visitor is viewing, rather than the specific page the visitor is viewing, you may want to use s_objectID to separate links that appear on multiple versions of that page template. For example, if you have a template page for all products on your site, it is likely that there is a link to your home page and to a search box from that template on all pages. If you want to see how those links are used on an individual product basis (rather than a template basis), then you can populate s_objectID with a product specific value such as "prod 123789 home page" or "prod 123789 search." Once completed, ClickMap will report on those links at an individual product basis.

Improving ClickMap Accuracy

In some cases, browsers other than Internet Explorer, Firefox, Netscape, Opera, and Safari are not reported. Although this is a small percentage, it accounts for some clicks and other metrics. Using the "s_objectID" within links to uniquely identify them addresses the browser reporting issue. The following is an example of how to update your links to use s_objectID.

```
<a href="/art.jsp?id=559" onClick="s_objectID='top left 1'">Article 559</a>
<a href="/home.jsp" onClick="s_objectID='prod 123789 home page'">Home</a>
```

Syntax and Possible Values

s_objectID may contain any text identifier.

```
s_objectID="unique_id"
```

There are no limitations on s_objectID outside of the standard variable limitations.

Examples

```
s_objectID="top left 2"
s_objectID="prod 123789 search"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

Before implementing s_objectID, refer to *ClickMap* in this document.

2.1.43 pageName

The pageName variable contains the name of each page on your site.

Table: pageName Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	pageName	Traffic > Segmentation > Most Popular Pages Paths	page URL

The pageName variable should be populated with a value that business users will recognize. In most cases the pageName value is NOT the URL or the path to the file. Common pageName values include names like "Home Page," "Checkout," "Purchase Thank you," or "Registration." For more information on page naming best practices, refer to *Page Naming* in this document.

Be careful not to allow new-line, -em or -en dashes, or any HTML characters to appear in the page name (and other variables, for that matter). Some browsers send new line characters while others don't, which causes the data in SiteCatalyst to be split between two seemingly identical page names. Many word processors and email clients will automatically convert a hyphen into an -en or -em dash when typing. Since -en and -em dashes are illegal characters in SiteCatalyst variables (ASCII characters with codes above 127), SiteCatalyst will not record the page name containing the illegal character and show the URL instead.

If pageName is left blank, the URL is used to represent the page name in SiteCatalyst. Leaving pageName blank is often problematic because the URL may not always be the same for a page (www.mysite.com and mysite.com are the same page with different URLs).

Syntax and Possible Values

The pageName variable should contain a useful identifier for business users of SiteCatalyst.

```
s.pageName="page_name"
```

There are no limitations on pageName outside of the standard variable limitations.

Examples

```
s.pageName="Search Results"  
s.pageName="Standard Offer List"
```

SiteCatalyst Configuration Settings

Administrators have the ability to change the visible page name in SiteCatalyst with the Name Pages Tool, which is potentially dangerous and may negatively affect your reports. Please contact Omniture ClientCare before using the Name Pages Tool.

Pitfalls, Questions and Tips

Make sure the pageName doesn't contain illegal characters.

2.1.44 pageType

The pageType variable is used only to designate a 404 Page Not Found Error Page. It only has one possible value, which is "errorPage." On a 404 Error Page, the pageName variable should not be populated.

Table: pageType Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
20 bytes	pageType	Paths > Pages > Pages Not Found	""

The pageType variable captures the errant URL when a 404 Error Page is displayed, which allows you to quickly find broken links and paths that are no longer valid on the custom site. Set up the pageType variable on the error page exactly as shown below. Do not use the page name variable on 404 error pages. The pageType variable is only used for the 404 Error Page.



NOTE: In most cases, the 404 Error Page is a static page that is hard-coded. In these cases, it is important that the reference to the .JS file is set to an appropriate global or relative path/directory.

Syntax and Possible Values

The only allowable value of pageType is "errorPage" as shown below.

```
s.pageType="errorPage"
```

Examples

```
s.pageType="errorPage"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

To capture other server-side errors (like 500 errors), use a prop to capture the error message and put "500 Error: <URL>" where <URL> is the URL requested, in the pageName variable. By following this course of action, you can use Pathing Reports to see what paths caused users to generate 500 errors, and the prop will explain which error message is given by the server.

2.1.45 pageURL

In rare cases, the URL of the page is not the URL that you would like reported in SiteCatalyst. To accommodate these situations, SiteCatalyst offers the pageURL variable, which overrides the actual URL of the page.

Table: pageURL Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
256 bytes	G	Traffic > Segmentation > Most Popular Pages Paths	Page URL



NOTE: Contact your Omniture Implementation Consultant before using the pageURL variable.

Syntax and Possible Values

pageURL must be a valid URL, with a valid protocol. The domain will be forced to display in lower-case before being populated in SiteCatalyst reports, and the query string may be stripped, depending on SiteCatalyst settings.

```
s.pageURL="proto://domain/path?query_string"
```

Only URL-compatible characters are allowed as the page URL.

Examples

```
s.pageURL="http://mysite.com/home.jsp?id=1224"
s.pageURL="http://www.mysite.com/"
```

SiteCatalyst Configuration Settings

None

2.1.46 plugins

The plugins variable, in Netscape and Mozilla-based browsers, lists the plugins installed on the browser. This variable is populated after the page code and before doPlugins is run.

 **WARNING!:** This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: plugins Variable Parameters

Query Param	Value	Example	Reports Affected
p	Recognized plugins	IE Tab Plug-in;QuickTime Plug-in 7.1.6;Mozilla Default Plug-in;iTunes Application Detector;Adobe Acrobat;ActiveTouch General Plugin Container;Shockwave Flash;Microsoft Office 2003;Java(TM) Platform SE 6 UI;Windows Media Player Plug-in Dynamic Link Library;Microsoft® DRM;	Traffic > Technology > Plugins

2.1.47 products

The products variable is used for tracking products and product categories as well as purchase quantity and purchase price. The products variable should always be set in conjunction with a success event. Optionally, the products variable can track custom incrementor events and Merchandising Evars.

Table: products Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
No Limit*	products	Products Categories (optional) Revenue (optional) Units (optional) Custom Events (optional) eVars (optional)	" "



NOTE: Although Omniture does not impose a size limit on products, most browsers impose a size limit on the URL of SiteCatalyst image request. The "Product" and "Category" sections of products each have a limit of 100 bytes.

The products variable tracks how users interact with products on your site. For instance, the products variable can track how many times a product is viewed, added to the shopping cart, checked out and purchased. It can also track the relative effectiveness of merchandising categories on your site. The scenarios below are common for using the products variable.

Setting products with Non Purchase Events

The products variable must be set in conjunction with a success event.

Syntax

```
s.events="prodView"  
s.products="Category;Product[,Category;Product]"
```

In the examples below, product attributes (category) are separated by semicolons. Multiple products are separated by commas.

Example 1: Single Product

```
s.events="prodView"  
s.products="Running;Shoe"
```

Example 2: Multiple Products

```
s.events="prodView"  
s.products="Running;Shoe,Running;Socks"
```

Example 3: Omitting Category

```
s.events="prodView"  
s.products=";Shoe,;Socks"
```



NOTE: The category/product delimiter (;) is required as a place holder when omitting categories.

Setting products with a Purchase Event

The purchase event should be set on the final confirmation ("Thank You!") page of the order process. The product name, category, quantity and price are all captured in the products variable. Although the purchaseID variable is not required, it is strongly recommended in order to prevent duplicate orders.

Syntax

```
s.events="purchase"  
s.products="Category;Product;Quantity;Price[,Category;Product;Quantity;Price]"  
s.purchaseID="1234567890"
```

Example 1: Single Product

```
s.events="purchase"  
s.products="Running;Shoe;1;69.95"  
s.purchaseID="1234567890"
```

Example 2: Multiple Products

```
s.events="purchase"  
s.products="Running;Shoe;1;69.95,Running;Socks;10;29.99"  
s.purchaseID="1234567890"
```



NOTE: Price refers to the total price (unit price x units). For instance, 3 widgets purchased at 19.99 each would equal 59.97 (i.e., "Category;Widget;3;59.97").

Example 3: Omitting Category

```
s.events="purchase"  
s.products=";Shoe;1;69.95,;Socks;10;29.99"  
s.purchaseID="1234567890"
```



NOTE: The category/product delimiter (;) is required as a place holder when omitting categories.

Setting products with Custom Incrementor Events

By default, success events are configured as Auto Incrementor events. Auto Incrementor events simply count the number of times a success event is set. Some success event uses require that an event be incremented by some custom amount.

Syntax

```
s.events="event1"  
s.products="Category;Product;Quantity;Price;eventN=X[|eventN=X][,Category;Product;Quantity;Price;eventN=X]"
```

Example 1: Single Custom Incrementor Event

```
s.events="purchase,event1"  
s.products="Running;Shoe;1;69.95;event1=7.59"
```



NOTE: Custom Incrementor events require configuration by Omniture ClientCare.

Example 2: Multiple Custom Incrementor Events

```
s.events="purchase,event1,event2"  
s.products="Running;Shoe;1;69.95;event1=7.59|event2=19.45"
```

Setting products with Merchandising Evars

For information on Merchandising Evars, refer to the *Merchandising* white paper.

Configuring the Variable in SiteCatalyst

Configuration for Custom Incrementor Events and Merchandising Evars must be done by Omniture ClientCare.

Pitfalls, Questions and Tips

- The products variable should always be set in conjunction with a success event (events). If no success event is specified, the default event is prodView.
- Strip all commas and semicolons from product and category names before populating products.
- Strip all HTML characters (registered symbols, trademarks, etc.).
- Strip currency symbols (\$) from price.
- The category represents the "Home" category for the product. The product-category relationship is created when a product is first recorded and persists indefinitely. All subsequent success events recorded for the product will automatically be credited to the product's "Home" category.

2.1.48 propN

Property (prop) variables are used for building custom reports within SiteCatalyst's Traffic Module. props may be used as counters (to count the number of times a page view is sent), for pathing reports, or in correlation reports.

Table: propN Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	c1-c50	Custom Traffic X	""

Custom traffic variables, or props, are counters that count the number of times each value is sent into SiteCatalyst. For example, property variables can be used to show content type, sub section, or template name. The resulting Custom Traffic reports will show which content type, sub section or template is viewed most often.

Syntax and Possible Values

s.propN="value"

There are no limitations on property variables outside of the standard variable limitations.

Examples

```
s.prop2="editorial"  
s.prop15="toy category"
```

SiteCatalyst Configuration Settings

Contact Omniture ClientCare about showing Visit, Visitor, and Path metrics for prop variables.

2.1.49 purchaseID

The purchaseID is used to keep an order from being counted multiple times by SiteCatalyst. Whenever the purchase event is used on your site, you should use the purchaseID variable.

Table: purchaseID Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
20 bytes	purchaseID	Conversion > Purchases > Revenue Conversion	""

When a visitor purchases an item from your site, purchaseID is populated on the "Thank You" page at the same place the purchase event is fired. If the purchaseID is populated, the products on the "Thank You" page will only be counted once per purchaseID, which is critical because many visitors to your site will save the "Thank You" or "Confirmation Page" for their own purposes. The purchaseID keeps purchases from being counted each time the page is viewed.

In addition to keeping the purchase data from being counted twice, the purchaseID, when used, keeps ALL conversion data from being double counted in reports.

Syntax and Possible Values

```
s.purchaseID="unique_id"
```

The purchase ID must be 20 characters or less, and be standard ASCII.

Examples

```
s.purchaseID="11223344"  
s.purchaseID="a8g784hjqlmnp3"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

The purchaseID will allow ALL conversion variables on the page to be counted only once in SiteCatalyst reports.

2.1.50 referrer

Server-side and JavaScript redirects are often used to route visitors to proper locations. However, when a browser is redirected, the original referring URL is lost. The referrer variable may be used to restore lost referrer information.

Table: referrer Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
255 bytes	R	Traffic > Finding Methods Conversion > Finding Methods	document.referrer

Many companies use redirects in many places throughout their web sites. For example, a visitor may be sent through a redirect from a search engine paid search result. When a browser is redirected, the referrer is often lost. The referrer variable may be used to restore the original referrer value on the first page after a redirect. The referrer may be populated server-side, or via JavaScript from the query string.

For SiteCatalyst to record a referrer, it must be “well formed,” meaning that it must follow the standard URL format, with a protocol and appropriate location.

Syntax and Possible Values

```
s.referrer="URL"
```

Only URL-compatible values should be in the referrer. Make sure the string is URL encoded (no spaces).

Examples

```
s.referrer="http://www.google.com/search?q=search+string"
s.referrer=<%=referrerVar%> // populated server-side
if(s.getQueryParam('ref'))
s.referrer=s.getQueryParam('ref')
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

The referrer must look like a standard URL and include a protocol.

2.1.51 resolution

The resolution variable indicates the monitor resolution of the visitor viewing the web page. This variable is populated after the page code and before doPlugins is run.



WARNING! This variable should only be read and never set.

You may read these values and copy them into props/eVars, but you should never alter them. This variable is introduced with version H.11 of the JavaScript file.

Table: resolution Variable Parameters

Query Param	Value	Example	Reports Affected
s	WxH	1680x1050	Traffic > Technology > Monitor Resolution

2.1.52 server

server is used to show either the domain of a web page (to show which domains people come to) or the server serving the page (for a load balancing quick reference).

Table: server Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	server	Traffic > Segmentation > Most Popular Servers	""

If your site has more than one domain serving the same content, the server variable may be used to track which of those domains visitors are using. The following JavaScript will populate the domain of the page into the server variable.

```
s.server=window.location.hostname
```

If you are using the server variable to give a quick guide to load balancing, you could put a server name or number into the server variable, as shown in the following example.

```
s.server="server 14"
```

While the Most Popular Servers Report may be used as a load balancing quick reference, it is not a precise measure of server load. For example, back-button traffic does not increase server load, but is shown in SC reports. Also, the SiteCatalyst report will not show which servers are serving images or large downloads.

Syntax and Possible Values

```
s.server="server_name"
```

There are no limitations on the server variable outside of the standard variable limitations.

Examples

```
s.server="server 18"  
s.server=window.location.hostname
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

The server variable can be used to show either which domains are most popular or which servers are serving the most pages.

2.1.53 state

The state and zip variables are conversion variables. They are like eVars in that they capture events, but unlike eVars, they don't persist. In other words, the zip and state variables are like eVars that expire immediately.

Table: state Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
50 bytes	state	Conversion > Visitor Profile > States	""

Since the state and zip variables expire immediately, the only events associated with them are events that are fired on the same page on which they are populated. For example, if you are using state to compare conversion rates by state, you should populate the state variable on every page of the checkout process. For conversion sites, Omniture recommends using the billing address as the source for the zip code, but you may choose to use the shipping address instead (assuming there is only one shipping address for the order). A media site may choose to use zip and state for registration or ad click-through tracking.

Syntax and Possible Values

```
s.state="state"
```

The state variable does not impose any special value or format restrictions. There are no limitations on state outside of the standard variable limitations.

Examples

```
s.state="california"
s.state="prince edward island"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Populate state on every page that a relevant event is fired (e.g. each page of the checkout process).
- The zip and state variables act like eVars that expire on the Page View.

2.1.54 trackDownloadLinks

Set trackDownloadLinks to 'true' if you would like to track links to downloadable files on your site. If trackDownloadLinks is 'true,' linkDownloadFileTypes is used to determine which links are downloadable files.

Table: trackDownloadLinks Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	True

trackDownloadLinks should only be set to 'false' if there are no links to downloadable files on your site, or you don't care to track the number of clicks on downloadable files. If trackDownloadLinks is 'true,' then when a file download link is clicked, data is immediately sent to SiteCatalyst. The data that is sent with an download link includes the link download URL, and ClickMap data for that link. If trackDownloadLinks is 'false,' then ClickMap data for links to downloadable files on your site is likely to be underreported.

Syntax and Possible Values

trackDownloadLinks is expected to be either 'true' or 'false.'

```
s.trackDownloadLinks=true|false
```

Examples

```
s.trackDownloadLinks=true
s.trackDownloadLinks=false
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- When trackDownloadLinks is 'false,' links that people use to download files on your site are likely to be underreported in ClickMap.
- When trackDownloadLinks is 'true,' data is sent to SiteCatalyst each time a visitor clicks a file download link.

2.1.55 trackExternalLinks

If trackExternalLinks is 'true,' linkInternalFilters and linkExternalFilters are used to determine whether any link clicked is an exit link.

Table: trackExternalLinks Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	True

trackExternalLinks should only be set to 'false' if there are no exit links on your site, or if you don't care to track the number of clicks on those exit links. An exit link is any link that takes a visitor off of your site. If trackExternalLinks is 'true,' then when you click an exit link, data is immediately sent to SiteCatalyst. The data that is sent with an exit link includes the link URL, link name and ClickMap data for that link. If trackExternalLinks is 'false,' then ClickMap data for exit links on your site is likely to be underreported.

Syntax and Possible Values

trackExternalLinks is expected to be either 'true' or 'false.'

```
s.trackExternalLinks=true|false
```

Examples

```
s.trackExternalLinks=true  
s.trackExternalLinks=false
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- When trackExternalLinks is 'false,' links that take people away from your site are likely to be under reported in ClickMap.
- When trackExternalLinks is 'true,' data is sent to SiteCatalyst each time a visitor clicks on an exit link (before link target loads).

2.1.56 trackingServer

The trackingServer variable is used for first-party cookie implementation to specify the domain at which the image request and cookie will be written. Used for non-secure pages. If trackingServer is defined, nothing goes to 2o7.net. If trackingServer is not defined (and dc is not defined), data goes to 112.2o7.net.

Table: trackingServer Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	66177

2.1.57 trackingServerSecure

The trackingServerSecure variable is used for first-party cookie implementation to specify the domain at which the image request and cookie will be written. Used for secure pages. If trackingServerSecure is not defined, SSL data goes to trackingServer.

Table: trackingServerSecure Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	66177

2.1.58 trackInlineStats

trackInlineStats determines whether ClickMap data is gathered. If trackInlineStats is 'true,' data about the page and link clicked are stored in a cookie called s_sq. If 'false,' s_sq will have a value of "[[B]]," which is considered null.

Table: trackInlineStats Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	ClickMap	True

Syntax and Possible Values

```
s.trackInlineStats=true|false
```

trackInlineStats is expected to be either 'true' or 'false.'

If you would like to use ClickMap, trackInlineStats should be set to 'True.'

Examples

```
s.trackInlineStats=true  
s.trackInlineStats=false
```

SiteCatalyst Configuration Settings

None

2.1.59 transactionID

Integration Data Sources utilize a transaction ID to tie offline data to an online transaction (like a lead or purchase generated online). Each unique transactionID sent to Omniture will be recorded in preparation for a Data Sources upload of offline information about that transaction. Refer to the Data Sources User Guide for more information.

Table: transactionID Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	xact	n/a	""

Syntax and Possible Values

```
s.transactionID="unique_id"
```

The transactionID should not contain a pipe character.

Examples

```
s.transactionID="11123456"
```

```
s.transactionID="lead_12345xyz"
```

```
s.transactionID=s.purchaseID
```

SiteCatalyst Configuration Settings

Before transactionID values will be recorded, transactionID recording must be enabled by an Omniture representative. To see whether transactionID recording is enabled for a report suite, go to SiteCatalyst > Data Sources > Manage.

Pitfalls, Questions and Tips

If multiple transactionIDs should be recorded in a single hit, you can use a pipe character to delimit multiple values. If transactionID recording is not enabled, transactionID values will be discarded and unavailable for use with Integration Data Sources. Make sure to set a conversion variable or event (an eVar or the events variable) on the page where transactionID is set. Otherwise, no data will be recorded for the transactionID.

If you are recording transactionIDs for multiple systems, like purchases and leads, make sure the value in transactionID is always unique. This may be accomplished by adding a prefix to the ID, like lead_1234 and purchase_1234. Integration Data Sources will not function as expected (Data Source data will tie to the wrong data) if a unique transactionID is seen twice.

By default, transactionID values are remembered for 90 days. If your offline interaction process is longer than 90 days, contact an Omniture representative to have the limit extended.



NOTE: The transactionID variable can contain any character other than a comma, which should be in the same location where the character limit (100 bytes) is specified. If multi-byte characters are used, multi-byte character support must be enabled for the report suite in order to avoid any problems with unexpected characters in the transactionID.

2.1.60 s_usePlugins

If the s_doPlugins function is available and contains useful code, s_usePlugins should be set to 'true.' When usePlugins is 'true,' the s_doPlugins function is called prior to each image request.

Table: s_usePlugins Function Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	True

Syntax and Possible Values

```
s.usePlugins=true|false
```

usePlugins is expected to be either 'true' or 'false.'

Examples

```
s.usePlugins=true  
s.usePlugins=false
```

usePlugins should only be false (or not declared) if the s_doPlugins function is not declared in your JavaScript file.

SiteCatalyst Configuration Settings

None

2.1.61 visitorID

Visitors may be identified by the visitorID variable, or by IP address/User Agent. The visitorID may be up to 100 alpha-numeric characters and must not contain a hyphen.

Table: visitorID Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
100 bytes	vid	n/a	""

Syntax and Possible Values

```
s.visitorID="visitor_id"
```



NOTE: The visitorID variable should not contain a hyphen.

Examples

```
s.visitorID="abc123"
```

SiteCatalyst Configuration Settings

n/a

2.1.62 visitorNamespace

If visitorNamespace is used in your JavaScript file, do not delete or alter it. This variable is used to identify the domain with which cookies are set. If visitorNamespace changes, all visitors reported in SiteCatalyst may become new visitors. In short, do not alter this variable without approval from an Omniture Implementation Consultant.

Table: visitorNamespace Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
N/A	N/A	N/A	""

SiteCatalyst uses a cookie to uniquely identify visitors to your site. If visitorNamespace is not used, the cookie is associated 2o7.net. If visitorNamespace is used, the cookie is associated with a sub-domain of 2o7.net. All visitors to your site should have their cookies associated with the same domain or sub-domain. If you change the value of visitorNamespace without first consulting an Omniture Implementation Consultant, you are likely to lose data.

The reason for using the visitorNamespace variable is to avoid the possibility of overloading a browser's cookie limit. Internet Explorer imposes a limit of 20 cookies per domain. By using the visitorNamespace variable, other companies' SiteCatalyst cookies will not conflict with your visitors' cookies.

Syntax and Possible Values

The value of visitorNamespace must be provided by Omniture and is a string of ASCII characters that will not contain commas, periods, spaces, or special characters.

```
s.visitorNamespace="company_specific_value"
```

Examples

```
s.visitorNamespace="company_name"  
s.visitorNamespace="omniture"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Please contact your Omniture Implementation Consultant if you would like to use the visitorNamespace variable.
- If you use a CNAME change to make SiteCatalyst cookies into first party cookies, the visitorNamespace variable should be left blank.

2.1.63 zip

The state and zip variables are conversion variables. They are like eVars in that they capture events, but unlike eVars, they don't persist. In other words, the zip and state variables are like eVars that expire immediately.

Table: zip Variable Parameters

Max Size	Debugger Parameter	Reports Populated	Default Value
50 bytes	zip	Conversion > Visitor Profile > ZIP/Postal Codes	""

Since the state and zip variables expire immediately, the only events associated with them are events that are fired on the same page that they are populated. For example, if you are using zip to compare conversion rates by zip code, you should populate zip on every page of the checkout process. Omniture recommends using the billing address as the source for the zip code, but you may choose to use the shipping address instead (assuming there is

only one shipping address for the order). A media site may choose to use zip and state for registration or ad click-through tracking.

Syntax and Possible Values

```
s.zip="zip_code"
```

The zip variable does not impose any value or format restrictions. There are no limitations on zip outside of the standard variable limitations.

Examples

```
s.zip="92806"  
s.zip="92806-4115"
```

SiteCatalyst Configuration Settings

None

Pitfalls, Questions and Tips

- Populate zip on every page in which a relevant event is fired (e.g. each page of the checkout process).
- The zip and state variables act like eVars that expire on the Page View.

3 Traffic props and Conversion eVars

Custom traffic variables, also called props (s.prop) or property variables, are counters that count the number of times each value is sent into SiteCatalyst. They also enable you to correlate custom data with specific traffic-related events

These variables are embedded in the SiteCatalyst code on each page of your website. Through s.prop variables, SiteCatalyst allows you to create custom reports, unique to your organization, industry, and business objectives.

For example, if you are an automobile manufacturer, you may be interested in seeing "Most Popular Car Model" to complement your "Pages" report. You can accomplish this by allocating one of your traffic properties to represent car model and then implement your code to pass in car model on the appropriate pages.



NOTE: SiteCatalyst supports up to 50 s.prop variables.

Props can be used in pathing reports or in correlation reports. For example, property variables can be used to show content type, sub-section, or template name. The resulting Custom Traffic reports will show which content type, sub-section, or template is viewed most often.

There are endless business questions that can be answered through the custom traffic variables, depending on what you are capturing from your website. The following list contains a few common goals and objectives.

- Understanding user navigation through the website
- Understanding internal user search behavior
- Segmenting traffic by navigation or category
- Segmenting visitor behavior by demographics

eVars (or Custom Conversion Insight Variables) are used to identify how well specific attributes or actions contribute to success events on your site. For example, for a media site, eVars may be used to identify how well internal promotions bring visitors to register. When a visitor clicks on the internal promotion, an eVar can be used to store a unique identifier for that promotion. When the same visitor completes registration and a custom success event is fired, the original unique identifier will receive credit for the registration event.

In a conversion site, eVars may be used to track how logged in visitors compare to non-logged in visitors in completing a purchase. When a visitor logs in, an eVar is set to "logged in." When that visitor reaches the checkout page, the checkout event will be attributed to the "logged in" value. Again, when a visitor reaches the Thank You page after purchasing, the products and purchase amounts will be attributed to the "logged in" value. The resulting Custom eVar Report will show the total number of checkouts and orders for "logged in" and "non-logged in" visitors.

3.1 Using props vs. eVars

Props are non-persistent variables that are generally used to tie "traffic" data to your custom variable. Traffic data includes visits, visitors, and page views. eVars, on the other hand, are persistent variables that are generally used to tie "success" events to your custom variable. Custom events include revenue, orders, leads, bookings, registrations, logins, etc. Basically, if you want to tie your data to traffic metrics, use a prop. If you want to tie your data to your site's custom success events, use an eVar.

3.2 Using props as Counters

A counter stores (and sometimes displays) the number of times a particular event or process has occurred. You can use a prop to count the number of times an event occurs. For example, you want to track the use of the Real Player vs. the Windows Media Player on your site. Each page contains Code to Paste, in which you will see s.prop variables. Use s.prop 1 to track the players. For page A, enter a value in s.prop1 to represent Real Player.

```
s.prop1="RealPlayer"
```


For page B, enter a similar value for in s.prop1 for Windows Media Player, as shown below.

s.prop1="WindowsMP"



NOTE: Omniture offers up to 50 s.prop variables for you to use.

As visitors come to your site and visit the pages containing the Real Player or Windows Media Player, SiteCatalyst will be able to segment the users based on which pages they visited. The SiteCatalyst Custom Traffic report will then be able to show the number of visits to each page.



NOTE: The name of the Custom Traffic report can be customized. For example, the Custom Traffic report can be renamed to "Player Types Report."

3.3 Counting Content Hierarchies

What is a content hierarchy? A content hierarchy can be used for various things. A common usage of content hierarchies is to show the different paths visitors have taken from a certain page, level, etc. How Should I track my Content Hierarchy? First, you must first understand the reporting requirements for tracking content hierarchies. If the requirements for tracking the hierarchy are very detailed, often times the hierarchy (hier) variable is recommended. Hierarchies usually require a strict, pre-defined taxonomy where the same child node rarely lives under multiple parent nodes. Consider the following example.

Global Hierarchy

All Sites > Regions > Countries > Language > Category

In this example, the hierarchy could begin to break down at the language level. If a requirement is to report on overall "English" traffic then you will run into the problem where English appears under USA, England, Australia, etc. Hierarchies allow you to only drill down. In order to slice horizontally across multiple hierarchies, the best practice is to use a custom traffic variable (prop).

If you want to provide users with the ability to drill down through the site (similar to how users would browse the site) and report on Unique Visitors at each level of the hierarchy, the hierarchy variable would be recommended.

There are occasions when using both props and the hier variable makes sense. Refer to the list below on what is supported for each variable type when making your decision.

Table: Supported prop and hier Variables

	Props	Hierarchy
Correlations	✓	✓
Pathing	✓	
Page View	✓	✓
Unique Visitors	✓	✓
Classifications	✓	

3.4 What is a Predefined Event?

Omniure offers several predefined events, as described in the following table.

Table: Predefined Events

prodView	Success event occurs any time a visitor views a product.
scView	Success event occurs any time a shopping cart is viewed.
scOpen	Success event occurs any time a visitor opens a shopping cart for the first time.
scAdd	Success event occurs any time a product is added to a shopping cart.
scRemove	Success event occurs any time an item is taken out of a shopping cart.
scCheckout	Success event occurs on the first page of a checkout.
purchase	Success event occurs on the final page of a checkout (includes Revenue, Orders, and Units).

When any of the predefined events above occurs, an instance of the event is incremented, and you can view the metrics related to the event in several different SiteCatalyst reports. The following is an example of the code used to configure predefined events.

```
s.events="scAdd"  
s.events="scOpen,scAdd"
```

In the first example above, scAdd is the value of the event. Any time an item is added to the shopping cart, the event is incremented. In the second example, two values are captured at the same time. When multiple success events occur on the same page, each event is incremented.

3.4.1 Detailed Product View Page

In the example below, the products variable is used for tracking products and product categories (as well as purchase quantity and purchase price). A success event should always be set in conjunction with the products variable.

```
s.events="prodView"
```



NOTE: While prodView is treated in the implementation like an event, it does not have the same flexibility in the interface. The “prodView” event is really an instance of the product and is only available in the products report. As such Omniure recommends that you use a custom event in addition to the “prodView” event. This way, you can see the product view metrics alongside other metrics in other conversion reports.

```
s.products=";diamond earrings (54321)"
```



NOTE: The products string syntax must begin with a semi-colon. This is a legacy element for SiteCatalyst. We use to use that to delimit the category and product, but that creates a limitation within the interface should you ever want to change how you are classifying products. In order to have the maximum flexibility in your reporting it is best to leave this blank and use Classifications in SiteCatalyst to setup categories.

Shopping Cart Page (scOpen, scAdd, scRemove)

```
s.events="scOpen,scAdd"  
s.products=";SKU"
```

First Checkout Page

```
s.events="scCheckout"  
s.products=";SKU"
```

Confirmation Page (Purchase Complete)

```
s.events="purchase"  
s.products=";SKU"
```



NOTE: While using the SKU in the product string may make the products report less readable it will provide you the maximum flexibility later when you want to classify your products as you can easily create categories from the SKU that indicate finish, manufacturer, category and sub-category.

When the products variable is set in conjunction with the purchase event, the purchase quantity and total purchase price are included in the products value as shown above.

3.5 What is a Custom Event?

Similar to predefined events, custom events enable you to define the success type that you want to track. However, unlike the predefined events, the custom events allow you to define your own success metric. For example, if you have a newsletter, your success event could be "Registration." Clearly, "Registration" is not part of the predefined events, but by using a custom event, you can track the number of visitors who register for your newsletter. Custom events follow the standard syntax shown below.

```
s.events="event3"
```

The code above shows how you would assign an event to the events variable. However, if you do not modify the event name in the interface, then "event3" would display in the interface.

By default, success events are configured as Auto Incrementor events or standard counters. Auto Incrementor events simply count the number of times a success event is set. Some success event applications require that an event be incremented by some custom amount. This configuration is done by Omniture ClientCare.

3.6 Email Campaign Tracking

Companies often develop costly e-mail campaigns to market their products. The companies send bulk emails to customers with the expectation that some customers will access the site and a conversion will result. The companies use SiteCatalyst to determine the success of the e-mail campaign. SiteCatalyst can report campaign analysis data in several key metrics, including the following.

Table: Metrics

Metric	Description
Click-throughs	Displays the number of click-throughs tracked from the email to the landing page.
Purchases	Displays the number of purchases resulting from the email.
Segmentation by Subcategory	Displays comparison data between different types of users.

Modifications to the HTML email body and the JavaScript library are required in order to capture the key metrics shown above.

3.6.1 Implementing Email Campaign Tracking

There are several steps to follow in order to successfully display email campaign analysis data in SiteCatalyst. The steps are listed and described as follows.

1. Create unique tracking codes.

Though each tracking code must be unique, it does not need to be any more than a number. The list below shows examples of unique tracking codes.

- Email Opens: 112233A
- Promo Link 1: 112233B
- Promo Link 2: 112233C

Contact Omniture ClientCare for details on setting up and using tracking codes.

2. Add query string parameters to HTML email links.

In order to track a user click-through and subsequent success events, a query string parameter needs to be added to each link within the HTML email. You may choose to either track each link separately or to track all links together. In other words, each link can have a unique tracking code or all links can have the same tracking code. Consider the following hypothetical link within the email to a website.

```
<a href="http://www.mycompany.com/index.asp">Visit our home page</a>
```

The following query string parameters should be added to the link above.

```
<a href=
"http://www.mycompany.com/index.asp?sc_cid=112233B&sc_v1=SUBCATEGORY">Visit our
home page</a>
```

3. Update the JavaScript library.

Altering SiteCatalyst JavaScript code in the JavaScript file, s_code.js, allows SiteCatalyst to capture how many users (and which users) clicked-through from the email and participated in subsequent success events. There are two steps to updating the JavaScript library.

- a. Customize s_code.js by calling getQueryParam.

The s_code.js file should be placed in a location on the web server where each web page can access it. The doPlugins function within this file should be altered so that it will capture the query string parameters on the email links, for example:

```
/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {
  /* Add calls to plugins here */
  // External Campaigns
  s.campaign=s.getQueryParam('sc_cid')
}
s.doPlugins=s_doPlugins
```

Each query string parameter that needs to be copied into a variable should have one getQueryParam call. In the example above, the query string parameter sc_cid is copied into campaign and sc_v1 is copied into eVar1.



NOTE: Only the first call to `getQueryParam` is required to capture click-throughs. Contact your Omniture Implementation Engineer to implement this function and to ensure that your version of the JavaScript file contains the `getQueryParam` plug-in.

- b. Make sure that the "Code to Paste" JavaScript tags are on all landing pages. This code to paste must reference the version of `s_code.js` altered in Part A.

The following points are important to remember when updating the JavaScript library. These points are listed below.

- The query string parameters "sc_cid" and "sc_v1" must be visible in the URL on the final landing page; otherwise, no click-through conversion will be recorded.
- "sc_cid" and "sc_v1" are examples of query string parameters. Any query string parameter can be used and captured by the `getQueryParam` plugin. Make sure, however, that the query string parameters are only used for campaign tracking. Any time the parameters appear in a query string, their values will be copied into campaign and eVar1.

4. Use SAINT to classify campaign tracking codes (optional).

The SAINT Campaign Management Tool can be used to convert tracking codes into user-friendly names. It can also be used to summarize the success of each email campaign. The following steps outline the process required to set up an email campaign in SiteCatalyst. For more information on SAINT, refer to the *Campaign Management* white paper.

5. See pathing by email campaign (optional).

Pathing analysis by email campaign can be accomplished similarly to pathing by another campaign. You can use a variable to show pathing by campaign, as explained in the following steps.

- a. Consult Omniture ClientCare about turning on pathing for a custom traffic variable (prop).
- b. On all pages, copy the page name into the designated `s.prop`.
- c. On the email landing page, append the name of the email campaign to the prop. The result will display as shown below.

```
s.prop1="Home Page : Spring Promo Email"
```

When pathing is enabled for the custom traffic variable, you can use Path reports (Next Page Flow, Fallout, et cetera) to see visitor navigation from the landing page.

3.6.2 Tracking External Email

In addition to the external campaigns described in the previous section, SiteCatalyst can report campaign analysis data in several additional key metrics with configurations made by Omniture ClientCare.

Table: Email Campaigns

Campaign Type	Description
Emails Sent	Displays the total number of emails sent by the company; this metric is provided by the email vendor.
Emails Delivered	Displays the total number of emails delivered to email servers; this metric is provided by the email vendor.
Emails Opened	Displays the number of times the visitors opened the email.

For more information on tracking email campaign analysis data, contact Omniture ClientCare. Tracking email campaign analysis data is considered on a case-by-case basis.

3.7 Products

Products refer to one of many values contained in the products variable, which is used for tracking multiple products and product categories as well as purchase quantity and purchase price, and event serialization and merchandising.

3.8 When Should the Product Variable be Set?

The products variable should be set on pages when you need to track information about a product or multiple products. These types of pages generally show product information to the user. Some specific examples might include (but are not limited to) product detail pages, shopping carts, promotion pages, and landing pages. The most common use of products variable is in the stages of the purchase process.

3.8.1 Setting a Product with an Event

Omniture best practice is to set the products variable on the same page as an event. This ties the product(s) on the page to the event for reporting in SiteCatalyst. While any event can be used to tie data to products, the most common events are those associated with the shopping cart and purchase. These events include:

- Product views (prodView event)
- Cart additions (scAdd event)
- Checkouts (scCheckout event)
- Purchases (purchase event)

An example of each is included below:

Product Views

```
s.events="prodView"  
s.products=";PRODUCT_NAME"
```

Cart Additions:

```
s.events="scAdd"  
s.products=";PRODUCT_NAME "
```

Checkout Page:

```
s.events="scCheckout"  
s.products=";PRODUCT_NAME "
```

Purchase Page:

```
s.events="purchase"  
s.products=";product_name;units;total_cost;events;evars, product_name;units;total_cost;events;evars "  
s.purchaseID="1234567890"
```



NOTE: Multiple events and eVars are delimited with a pipe (|). In addition, multiple product strings are delimited with a comma (as shown in the Purchase Page example above).

3.8.2 Setting a Product without an Event

Omniiture recommends setting an event every time the products variable is used. If omitted, the system tracks a product view (prodView) event by default.

3.8.3 Field Definitions of the Product String

The following table contains the field definitions contained in the product string.

Field	Definition
product_name	The identifier used to track a product. This identifier is used to populate the Products report in SiteCatalyst. Be sure to use the same identifier through the checkout process.
units	The number of units purchased. This field must be set with a purchase event in order to be recorded.
total_cost	Refers not to the unit cost, but to the combined cost of the total units purchased. This field must be set with a purchase event in order to be recorded.
events	Custom events associated to a particular product; i.e., discounts.
eVars	Values associated with a specific product, such as merchandising category.

3.8.4 Sample Product Strings

Each of the following examples shows the product string with a different configuration.

- `s.products=";ABC123"`
- `s.products=";ABC123;;ABC456"`
- `s.products=";ABC123;1;10"`
- `s.products=";ABC123;1;10;;ABC456;2;19.98"`
- `s.events="event1"`
`s.products=";ABC123;;;event1=1.99"`
- `s.events="event1"`
`s.products=";ABC123;1;10;event1=1.99"`
- `s.events="event1"`
`s.products=";ABC123;1;10;event1=1.99;;ABC123;2;19.98;event1=1.99"`
- `s.events="event1,event2"`
`s.products=";ABC123;1;10;event1=1.99|event2=25"`
- `s.events="event1,event2"`
`s.products=";ABC123;1;10;event1=1.99|event2=25;evar1=Super Fast Shipping"`
- `s.events="event1,event2"`
`s.products=";ABC123;1;10;event1=1.99|event2=25;evar1=Super Fast Shipping|evar2=3 Stars"`
- `s.events="event1,event2"`
`s.products=";ABC123;1;10;event1=1.99|event2=25;evar1=Super Fast Shipping,;ABC456;2;19.98;event1=1.99|event2=100;evar1=really slow shipping"`

- **s.events="event1,event2,event3"**
s.products=";ABC123;1;10;event1=1.99|event2=25;evar1=Super Fast
Shipping,;ABC456;2;19.98;event1=1.99|event2=100;evar1=really slow
shipping,;;;event3=2.9;evar3=20% off"
- **s.events="event1,event2,event3"**
s.products=";ABC123;;;ABC456;2;19.98;event1=1.99|event2=100;evar1=really slow
shipping,;;;event3=2.9;evar3=20% off"

4 Pathing

Pathing is defined as the path that users take through your site. For example, a visitor went to page A, then page B, then page C. Pathing is one of the very powerful features of SiteCatalyst. The tremendous insight it brings is critical to businesses looking to understand visitor traffic patterns.

Out-of-the-box SiteCatalyst provides pathing at the page level. The basic idea behind pathing is you are shown the order of pages that users saw during their visits. This data is presented in several different reports which format the data in different ways depending on what you are trying to see.

For a detailed understanding of using these reports, we have a training video on “Path Analysis” in the Help section of SiteCatalyst, as well as other knowledge base articles on the subject. This section will focus on the implementation issues of pathing and extending pathing functionality to be able to segment your path reports.

4.1 *Enabling Pathing on a prop*

Though SiteCatalyst pathing reports are available out-of-the-box for pages, pathing can also be enabled for prop variables which are custom traffic variables. This setting is not enabled for any props by default because it is only appropriate in certain cases. Enabling pathing on a prop can only be done by Omniture Support (ClientCare, Account Managers and Implementation Consultants).

Depending on your contract, there may be an additional fee to enable pathing per prop. Your Account Manager or ClientCare representative will let you know of any applicable additional fees.

We first need to clearly understand the business questions that are being asked. Enabling pathing for a custom prop may be necessary to answer that question. We need to ensure that the data being populated into the prop will make sure that once we have enabled pathing on the prop it will make sense in our pathing reports.

A few examples of a business question that can be solved by turning on pathing for a prop are listed below.

- After a user comes to my site from a campaign, where then do they go on my site?
- What are the common site sections my users go to in their visits?
- How do I see page pathing by user type?

Essentially, what we’re trying to do is segment our pathing reports by various dimensions (campaigns, user type, etc...). In the sections below, we provide examples of how to use pathing on props to answer these business questions.

It is important to note that though this data can be achieved by enabling pathing on props and structuring the data in a specific way as outlined below, that the optimal way to segment pathing reports is using the Discover tool. Discover has been specifically designed to provide segmentation on any report on the fly without needing to enable this setting, or structure data in the prop in certain ways. It is the easiest and most optimal way to segment your path reports as well.

When pathing is enabled for a prop, we get a new set of pathing reports that are found below the standard pathing reports. Remember that pathing reports will tell you the order in which it saw the pages come in for that visitor. When we enable pathing on a prop, it does the same thing, which is, it tells us the order of the prop values it saw come in for the user. This order of values is displayed in different ways depending on the path report you are viewing.

4.2 *Pathing by Campaign or Tracking Code*

In this section we will cover the best way to answer the question, “After a user clicks through to my site from a campaign, where then do they go on my site?”. To answer this question, we will need to set aside a sprop to be used for this report. We then need to structure the data we populate into the prop in such a way that it will make sense when pathing is enabled on the prop.

For this example we will say we are going to use prop1 as our campaign pathing prop. Now we want to populate this sprop with the same value we are putting in the page name variable. Something like this:

```
s.prop1=s.pageName;
```

We would do this on all pages unless the person has clicked through from a campaign. If they have clicked through from a campaign and are on the landing page of the campaign, then we want to populate the prop with a concatenation of the campaign and the pagename. It would look something like this:

```
s.prop1=s.campaign + ' : ' + s.pageName;
```

If the campaign they clicked through on was named “banner1234”, and the page that it landed on was named “Home Page”, then the value in that prop would end up being “banner1234 : Home Page”. Then on every subsequent page we are just putting the pagename in the prop as shown above.

Now when a user clicks through on this campaign, and views four total pages in that visit, we would get the following values in the sprop in this order:

```
“banner1234 : Home Page” > “Page 2” > “Page 3” > “Page 4”
```

Obviously the page names will be different on your site, but you get the point of how the data will look. Now with our data captured in prop1 in this way, with pathing enabled on this prop we can now look at one of many of the various pathing reports to understand how they path through the site after they click-through from a campaign.

We can then login to SiteCatalyst and go to our campaign pathing reports and view a report such as the full paths report as shown in the screen shot below.

We can specify which start page we want to have our path report start from. In this case, we selected “banner1234 : Home Page”, because we wanted to know where they went after the clicked through from campaign “banner 1234”. This report is only an example of one of many path reports that could be run to see this data.

4.3 Reasons Pathing may not be Recorded

The only reasons pathing information might not be recorded and displayed in SiteCatalyst are listed below.

- Pathing has not been enabled for that prop in that report suite. This enablement is report suite specific.
- You are not passing data into the correct prop.
- Pathing has been enabled and the data is in the prop but it just hasn't shown up in the reports. Though the sprop data may show up within 10 minutes, the pathing data will not show up until the visitor's session has ended. It will end after 30 minutes of inactivity. It then needs another 10 minutes to finish the processing of the path data for final presentation in the SiteCatalyst reports.

If you have checked on all of these and the data is still not showing up, check with ClientCare for further debugging.

4.4 Moving from Section to Section

If you want to know how your visitors move from section to section on your site, you first need to ensure you are tagging your sections with the channel variable. With this data in place, you then just need to have pathing enabled on the channel variable. If a user starts on the home page and moves to the Sports section, then to the News section, this activity will show up in the Site Section path reports that will show up after pathing has been enabled on this variable.

4.5 Moving from Page Template to Page Template

If your site has types of pages or page templates, you could use pathing to understand how they move from type to type. You would do this in the same way we did in the previous section on moving from section to section.

You first need to capture the page type or template in a prop set aside for only that purpose. Then have Omniture ClientCare enable pathing on that prop. You can now view your page template pathing reports to understand how users move from template to template on your site.

4.6 Segmenting Paths by User Type

Segmenting paths by user type is a common request for trying to understand how specific user types path on your site. For this we will employ a similar technique as we did in the section above on campaign pathing. We will concatenate the user type and page name into a sprop and enable pathing on the sprop.

For example, let's say we have two user types: "Registered" users and "Non-Registered" users. You first need to be able to distinguish between these two user types on each page and be able to put these values into your designated sprop. When you populate the prop, it should display as shown below.

```
s.prop1="Registered : " + s.pageName;
```

If the user was a registered user and visited the home page, the value in the prop would show up as follows.

```
"Registered : Home Page"
```

If they then clicked to another page named "Page 2", the value on that page would display as follows.

```
"Registered : Page 2"
```

With Pathing turned on, we can now see those 2 values in succession. If we want to know how registered users move from the home page, we can find the value "Registered : Home Page" in one of the path reports and see the next pages they visited. In this case that they next went to "Page 2".

5 Using Implementation Plug-ins

SiteCatalyst JavaScript plug-ins are programs or functions that perform several advanced functions, which are listed below.

- Retrieve query string parameter values
- Detect Flash
- Write cookies to store session data
- Meet other business requirements

These plug-ins extend the capabilities of your JavaScript file to give you more functionality that is not available with a basic implementation. Even though Omniture offers several plug-ins, only a few of them are implementation-specific. Therefore, these plug-ins are documented here. For information on the other Omniture plug-ins, refer to the Omniture Knowledge Base.

5.1 Calling Plug-ins with the doPlugins Function

JavaScript plug-ins are usually called by the doPlugins function, which is executed when the t() function is called in the Code to Paste. Consequently, if you set a variable in the doPlugins function, you may overwrite a variable you set on the HTML page. The only time the doPlugins function is not called is when the usePlugins variable is set to 'false.'

5.1.1 Code Example

The code example below is what the doPlugins function looks like in your SiteCatalyst JavaScript file (s_code.js).

```
/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {
    /* Add calls to plugins here */
}
s.doPlugins=s_doPlugins
```

5.1.2 Renaming the doPlugins Function

The doPlugins function is typically called s_doPlugins. However, in certain circumstances, (usually when more than one version of SiteCatalyst code may appear on a single page) the doPlugins function name may be changed. If the standard doPlugins function needs to be renamed to avoid conflicts, make sure doPlugins is assigned the correct function name, as shown in the example below.

```
/* Plugin Config */
s_mc.usePlugins=true
function s_mc_doPlugins(s_mc) {
    /* Add calls to plugins here */
}
s_mc.doPlugins=s_mc_doPlugins
```

5.1.3 Using doPlugins

The doPlugins function provides an easy way to give default values to variables, or to take values from query string parameters on any page of the site. Using doPlugins is often easier than populating the values in the HTML page because only one file needs to be updated. Keep in mind, however, that changes to the JavaScript file are not always immediate. Return visitors to your site are often using cached versions of the JavaScript file, meaning that updates to the file may not be applied to all visitors for up to one month after the change is made.

The following examples show how the doPlugins function can be used to set a default value for a variable and to get a value from the query string.

```
/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {
    /* Add calls to plugins here */
    // if prop1 doesn't have a value, set it to "Default Value"
    if(!s.prop1)
s.prop1="Default Value"

    // if campaign doesn't have a value, get cid from the query string
    if(!s.campaign)
s.campaign=getQueryParam('cid');
}
s.doPlugins=s_doPlugins
```

5.1.4 Installed Plug-ins

To find out whether a plug-in is included in your JavaScript file and ready for use, look in the Plugins Section of the JavaScript file. The following example shows what the getQueryParam function looks like in the Plugins Section.

```
/****** PLUGINS SECTION *****/
/* You may insert any plugins you wish to use here. */
/*
 * Plugin: getQueryParam 1.3 - Return query string parameter values
 */
s.getQueryParam=new Function("qp","d",""
+"var s=this,v='',i,t;d=d?d:'';while(qp){i=qp.indexOf(',');i=i<0?qp.1"
//
// ... more code below ...
//
```

5.2 Implementation Plug-ins

The following is a description of the plug-ins to use in a SiteCatalyst implementation. Contact your Omniture Implementation Consultant to get the most recent version of each plug-in.

Table: Implementation Plug-ins

Plug-in	Description
apl Plug-in Utility	API Plug-in Utility appends a value to any delimited lists.
getQueryParam	getQueryParam returns the value of the query string parameter found in the current URL. If no query string parameter is found with that value, an empty string is returned.
getValOnce	getValOnce is used to force a variable to be populated only once within a single session or time period. The most common reason for doing this is to keep campaign click-throughs from being inflated.

5.2.1 APL Plug-in Utility

The apl (or AppendList) Plug-in Utility provides a simple mechanism to append a value to any delimited lists, with the option of a case-sensitive or case-insensitive check to ensure the value doesn't already exist in the list. The apl plug-in is referenced by several standard plug-ins but can be used directly in a variety of situations.

Common Uses

Examples of when the apl plug-in can be used include, but are not limited to the following:

- Adding an event to the current events variable
- Adding a value to a list variable without duplicating a value in the list
- Adding a product to the current products variable based on some page logic
- Adding values to the parameters linkTrackVars and linkTrackEvents

Use Cases/Examples

Example 1

scenario	Add "event1" to the current events variable while ensuring the event isn't duplicated. s.events="scCheckout"
code	s.events=s.apl(s.events,"event1","",1)
results	s.events="scCheckout,event1"

Example 2

scenario	Add the value "history" to the list variable prop1, with "history" and "History" considered the same value. s.prop1="Science,History"
code	s.prop1=s.apl(s.prop1,"history","",2)
results	s.prop1="Science,History" "history" is not added because "History" is already in the list

Implementation Steps

To use the apl Plug-in Utility, follow the steps below.

1. Download the plug-in from the Admin Console in SiteCatalyst or request it from Omniture.
2. Add call(s) to the API function as needed within the s_doPlugins function

The following example shows how the code may look in your JavaScript file.

```
/* Plugin Config */
s.usePlugins=true
function s_doPlugins(s) {
    /* Add calls to plugins here */
    s.events=s.apl(s.events,"event1","",1)
}
```

```
s.doPlugins=s_doPlugins
```

Supported Browsers

This plug-in requires that the browser supports JavaScript version 1.0.

Notes

The source list L can be an empty list, i.e. L="". The returned value will either be an empty list, or a list of one value.

Plug-in Information

Plug-in Information	Description
Parameters	<code>apl((L,v,d,u)</code> L= source list, empty list is accepted v= value to append L delimited list to append to v value to append d list delimiter u (optional, defaults to 0) Unique value check. 0=no unique check, value is always appended. 1=case-insensitive check, append only if value isn't in list. 2=case-sensitive check, append only if value isn't in list.
Return Value	original list, with appended value if added
Usage Examples	<code>s.events=s.apl(s.events,"event1","",1);</code>

5.2.2 getQueryParam

The getQueryParam plug-in captures a value from the query string of a URL. Once captured, the value can be populated into any other JavaScript variable.

Plug-in Information

Plug-in Information	Description
Parameters	<code>getQueryParam(p,d,u);</code> p: (required) comma delimited list of case insensitive query string parameter names d: (optional) delimiter used to separate query string parameter values if multiple values are found. If omitted and multiple parameters from p are found, the strings are appended to each other without a delimiter. u: (optional) URL to take query string from. If omitted, pageURL or window.location is used. If 'f', the top-most frameset URL is used (in case you want to use the URL from

	the address bar and the code is inside a frame).
Configuration Variables	None
Return Value	The query string parameter specified 'True' if the query string parameter exists without a value A delimited list of values if multiple parameters are found
Cookies Set	None
Global Variables	None
Usage Examples	//single parameter s.campaign=s.getQueryParam('cid'); //multiple parameters s.campaign=s.getQueryParam('cid,sid',''); //non-page URL example s.campaign=s.getQueryParam('cid','',document.referrer); //parent frame example s.campaign=s.getQueryParam('cid','','f');

Implementation Steps

Obtain the JavaScript plug-in through Omniture ClientCare or your Implementation Consultant. Include the plug-in function in the *****Plug-ins***** section of the Omniture JavaScript file (see example below).

```
s.usePlugins=true
function s_doPlugins(s) {
    /* Add calls to plugins here */
}
s.doPlugins=s_doPlugins
/***** PLUGINS SECTION *****/
/* You may insert any plugins you wish to use here. */

[Insert Plug-in Code Here]

...Rest of the JS File...
```

Call the plug-in from within the doPlugins section of the JS file as shown below

```
s.usePlugins=true
function s_doPlugins(s) {
    /* Add calls to plugins here */
    [Insert Calls to Plug-ins Here]
}
s.doPlugins=s_doPlugins
/***** PLUGINS SECTION *****/
/* You may insert any plugins you wish to use here. */

[Insert Plug-in Code Here]
```



```
...Rest of the JS File...
```

Notes

This plug-in supports all browsers that are JavaScript 1.0 compatible. The first parameter passed into the `getQueryParam()` function must exactly match (except for case) the desired search parameter

5.2.3 getValOnce

`getValOnce` is used to force a variable to be populated only once within a single session or time period. The most common reason for doing this is to keep campaign click-throughs from being inflated.

Inflated click-throughs are caused by back-button traffic in conjunction with using the query string to identify campaign click-throughs. For example, when visitors click-through to your site from a campaign ad, the landing URL usually contains a query string parameter that is used to populate the campaign variable. When visitors leave the landing page and use the Back button to return to it, the campaign variable will again be populated from the query string on the landing page. Each time the campaign variable contains a value, a click-through is counted in SiteCatalyst. Using `getValOnce` will keep campaign click-throughs from being overcounted. In the following example, `getValOnce` is called.

```
s.eVar1=s.getValOnce(s.eVar1,'s_var_1',0)
```

The following table illustrates the order in which pages are viewed and demonstrate how `getValOnce` functions. The pre-`getValOnce` value of `eVar1` is the value set on the page or by a plug-in. The post-`getValOnce` value of `eVar1` is the value sent to SiteCatalyst.

Table: Page View Order

Page Number	eVar1 Value Pre-getValOnce	eVar1 Value Post-getValOnce	Value of Cookie 's_var_1'
1	123	123	123
2	123	""	123
3	ABC	ABC	ABC
4	ABC	""	ABC
5	123	123	123

`getValOnce` writes a cookie with a value `v` and expiration in `e` days or the end of the current browser session. `getValOnce` will only return '`v`' if it doesn't match the value of the cookie '`c`.' If '`v`' evaluates to '`false`' (i.e. the value is null, empty string false or 0) then empty string is returned and the cookie is not updated. The cookie value is updated each time its value does not match '`v`.' Since `getValOnce` writes a cookie with a value, it may be used to do things like allow an event to be set once per unique value, as shown in the following example. For additional information on keeping events from being fired more than once per unique value, refer to Event Serialization in this document.

```
var tempVar=s.getValOnce(s.getQueryParam('order_id'),'s_orderid')
if(tempVar)
    s.events=s.events?s.events+","event1":"event1"
```

Table: getValOnce

Plug-in Information	Description
Parameters	<code>getValOnce(v, c, e)</code> v Value. The value that should only be returned once per time period specified by e c Cooke name. The name of the cookie that will contain the value v e (Optional) days to Expiration. The number of days until the cookie expires. If omitted or set to 0, the cookie is a session cookie.
Return Value	Returns v or "" depending on whether the v matches the value of the cookie c.
Usage Examples	<code>s.eVar6=s.getValOnce(s.eVar6,'s_var_6')</code> <code>s.campaign=s.getValOnce(s.getQueryParam('cid'),'cmp_cookie',30)</code>



NOTE: If cookies are blocked, v is always returned.

6 Testing and Validation

Validation and testing are used to ensure data reporting accuracy. Validation and testing should always be done on the development report suite.

6.1 Using the JavaScript Debugger

The JavaScript-based debugger allows you to view the parameters sent in an image request. The debugger supports Microsoft Internet Explorer, Mozilla Firefox, and other web browsers.



NOTE: Add-ons will sometimes cause issues with the JavaScript Debugger, specifically Adblocker Plus, which will cause the Debugger window to display with a blank, white screen. To resolve this problem, disable the add-on when you want to use the Debugger, and then enable it again when you are done using the Debugger.

6.1.1 Setting up the Debugger

Follow the steps below to set up the JavaScript Debugger.

1. Open a browser window.
2. Go to any URL, for example, www.omniture.com.
3. In the browser window, click **Favorites**.
4. Click **Add to Favorites**.
5. Type the name of the favorite as JavaScript Debugger or some other unique name that will help you identify it.
6. Click **Favorites** again.
7. Right-click the JavaScript Debugger favorite you just created.
8. Click **Properties**.
9. Delete all text from the URL field of the Properties window.
10. Paste the following text into the URL field of the Properties window.

```
javascript:void(window.open('%22%22,%22stats_debugger%22,%22width=600,height=600,location=0,menubar=0,status=1,toolbar=0,resizable=1,scrollbars=1%22).document.write('%22<script language=\%22JavaScript\%22src=\%22https://sitecatalyst.omniture.com/sc_tools/stats_debugger.html\%22></%22+%22script>%22 + %22<script language=\%22JavaScript\%22>window.focus();</script>%22));
```



NOTE: This version of the Debugger can be used for debugging flash applications, but cannot be used for versions of Omniture code prior to H.

11. Click **OK**.
12. Click **Favorites** again.
13. Click the name of the favorite you have selected for the debugger, which will display in a new window as shown below.

Figure: JavaScript Debugger



NOTE: The JavaScript Debugger will not open if you have popups blocked. However, if you press the <CTRL> key when you click the JavaScript Debugger favorite, the popup blocker will be ignored. In addition, the debugger does not support HTML frames.

6.1.2 Alternate Debugger Versions

The version of the debugger received is based on the code that is typed into the URL Field of the Properties Window, as shown in *Setting up the Debugger* in this document. However, you can use other versions of the debugger as described in the following section.

Simple Debugger

The following code, which can be used in lieu of the code displayed on page 2 when setting up the Debugger, will display a standard "Message Box." The simple Debugger does not display as a pop up and does not offer cookie-related information as is displayed the Debugger used in the steps in *Setting up the Debugger*.

```
javascript:var j=document.styleSheets,i=document.images,r='';for(var
x=0;x<j.length;x++)if(j[x].imports)for(var
y=0;y<j[x].imports.length;y++)if(j[x].imports[y].href.toLowerCase().indexOf('
/b/ss/')>=0)r+=j[x].imports[y].href+"\n\n";for(var
x=0;x<i.length;x++)if(i[x].src.toLowerCase().indexOf('/b/ss/')>=0)r+=i[x].src
+"\n\n";for(w_m in
window)if(w_m.substring(0,4)=='s_i_')&&window[w_m].src)if(window[w_m].src.inde
xOf('/b/ss/')>=0)r+=window[w_m].src;void(alert(unescape(r).replace(/&/g,'\\n'))
))
```

6.1.3 Variables and Query String Parameters

The following tables displays the mapping between the JavaScript variable and the query string parameter in the debugger.

Table: JavaScript Variables

JavaScript Variable	Query String Parameter
pageName	pageName
server	server
pageType	pageType
channel	ch
prop1 - prop50	c1 - c50
hier1 - hier5	h1 - h5
campaign	v0
state	state
zip	zip
events	events
products	products
purchaseID	purchaseID
eVar1 - eVar50	v1 - v50
charSet	ce
currencyCode	cc
visitorNamespace	Ns
Referring URL	R
Current URL	g

Additional Parameters

Table: Other Parameters

t	Browser time information "DAY/MONTH/YEAR HOUR:MIN:SEC WEEKDAY TIMEZONEOFFSET"
s	Screen resolution (Width x height)
c	Screen color depth (8, 16, 32, etc.)
j	JavaScript version (1.0, 1.2, 1.3, etc.)
v	Java enabled ('Y' or 'N')
k	Cookies supported ('Y' or 'N')
bw	Browser client window width in pixels
bh	Browser client window height in pixels
ct	Connection type ('modem' or 'lan')
hp	Is current page browser's home page ('Y' or 'N')
p	',' Separated list of Netscape plug-in names
pid	Page identifier for ClickMap
pidt	Page identifier type for ClickMap
ndh	Identifies when an image request is sent from the JavaScript file
oid	Object identifier for ClickMap
ot	Object tag name for ClickMap

Pe (lnk_d, lnk_e, lnk_o)	s.linkType
pev1	Link URL
pev2	s.linkname

6.1.4 Identifying the s_account Variable in the JavaScript Debugger

When you run the JavaScript Debugger, you may want to look for the s_account variable. The following figure shows the location of the s_account variable.

Figure: JavaScript Debugger Code

```
Image
http://omniture.112.2o7.net/b/ss/omnicom/1/G.9p2/s9569
8397722543?[AQB]
ndh=1
t=12/9/2006 11:2:17 4 360
pageName=Homepage
g=http://www.omniture.com/
cc=USD
c1=Homepage
c2=Homepage
v9=Homepage
v15=general
[AQE]
```

6.2 Packet Monitors

Packet monitors provide excellent insight into the success of a SiteCatalyst implementation. Like the Debugger, a packet monitor shows what data parameters are being passed in an image request; however, packet monitors add the ability to view non-page view (custom link, download link, and exit link) image requests, as well as image requests using implementation methods other than JavaScript, including ActionSource and hard-coded image requests. Additionally, in very rare cases, the Debugger will report an image request although no request is actually made. Using a packet monitor is a great way to ensure that an image request is actually being sent to Omniture servers successfully.

While Omniture does not provide an official packet monitor, you can find a wide range of them on the Internet. Some packet monitors, such as *Tamper Data* for Firefox or *HTTPWatch* for Internet Explorer, operate as browser plug-ins, while others, such as *Wireshark* or *Charles*, are standalone applications. Please consult the user manuals for the packet monitor of your choice to ensure that it is configured correctly to capture data on the Omniture image request.

6.3 Deployment Validation

The SiteCatalyst code can be placed anywhere inside BODY tags (<BODY></BODY>) of a well-formed HTML page. Browsers do not interpret image requests placed inside HEAD tags (<HEAD></HEAD>) correctly, which can cause the code to function incorrectly. We recommend placing the code in a global include file at the top of the page (inside the HTML body tag), however the code can be placed anywhere on the page, except as noted below.

- Do not locate the code within graphical elements or tables, and ensure that the code does not move any elements (images, tables, etc.) unintentionally.
- Place the HTML and reference to the .JS file as high on the page as possible in order to measure more pageviews before a visitor clicks off to another page.
- Place the code only within the <td></td> tags, if placed within a table.

- The code that sets the variables must occur prior to the reference to the .JS file.
- Make certain that the Report Suite IDs (shown below in the sample as "sampleco") are set correctly. The code should be supplied by the Omniture Implementation Consultant with the Report Suite ID in place.

6.3.1 JavaScript JS File

Verify that the .JS file is correctly referenced from the page (the path may be specified either relative to the current document, or an absolute path name may be used, as shown below).

```
<script language="JavaScript"
src="http://www.sampleco.com/javascript/includes/s_code.js"></script>
```

If some pages of the site are loaded in a secure protocol (https:), and reference the SiteCatalyst .JS file, ensure that the reference to the file is either secure (via https:), or code the reference as shown below (this example adopts the protocol of the current page, and prevents the warning that "some elements are non-secure").

```
<script language="JavaScript"
src="//www.sampleco.com/javascript/includes/s_code.js"></script>
```

Ensure that the .JS file on the web servers have permissions appropriately set so that the file may be downloaded and executed by web site visitors. If a different .JS file is used on development servers, set the "read only" attribute for the .JS file on production servers to avoid an overwrite. If altered, ensure that the following settings are set appropriately at the top of the .JS file.

```
/* ***** CONFIG SECTION ***** */
/* You may add or alter any code config here. */
/* Link Tracking Config */
s.trackDownloadLinks=false /* true for download tracking */
s.trackExternalLinks=false /* true for exit link tracking */
s.trackInlineStats=false /* true for ClickMap support */
s.linkDownloadFileTypes="exe,zip,wav,mp3,mov,mpg,avi,doc,pdf,xls"
s.linkInternalFilters="javascript:"
s.linkLeaveQueryString=false
s.linkTrackVars="None" /* see your Omniture Implementation Consultant */
s.linkTrackEvents="None" /* see your Omniture Implementation Consultant */
```

If "s_account" is assigned a value at the top of the .JS file, ensure that the Report Suite ID (populated in the s_account variable) is correct. Also ensure that the code in the page is not also setting the Report Suite ID (s_account variable).

Examine the image request and variables to ensure that the "fallback method" (the third part of the "split" code in the example above) is not creating the image request instead of the .JS file, which can be determined since the "fallback" method only creates an image request with minimal information.

6.3.2 Code Modifications

Testing any modifications to the .JS file or HTML code is the responsibility of the customer, and should be completed prior to publishing the modifications to production Web sites.

Ensure that the linefeeds/return characters are not stripped or altered from the code that is placed within the HTML, or from within the .JS file. Additionally, ensure that the JavaScript executes without an error on all pages and page templates (in IE's Internet Options, select the Advanced Tab, and click **Display a Notification about Every Script Error**).

When testing for errors, paste the SiteCatalyst code into a default HTML page to determine if the error is occurring because of other page elements/objects.

```
<html><head></head><body>
...paste SiteCatalyst code here to debug...
</body></html>
```



TIP: When testing for errors, an HTML page may be saved locally and executed/evaluated with different values, etc.

6.3.3 Variables and Values

Ensure that the variables that are populated from server scripting or code cannot output any quotation marks that will interfere with the values. For instance:

```
s.pageName="Article: "Article Name""
s.pageName='Company's Information'
```

Ensure that the values of the variables do not exceed 100 characters. Additional characters are cropped at the data collection servers, but may interfere with the total length, increase bandwidth unnecessarily, and may cause other issues).

Do not use \$, ™, ®, ©, or commas (,) in the products variable. Generally, these are not useful in any SiteCatalyst variables, and may interfere with the ability to interpret or export fields. The best practice is to limit characters to the first 127 ASCII characters.

Ensure that the events variable is populated with an appropriate value ("prodView", "purchase", "scAdd", "scRemove", "scOpen", or "event1"- "event5") whenever products is populated. Additionally, ensure that the case of all SiteCatalyst variables and functions are maintained, as shown below.

```
s.pageName
s.server
s.channel
s.pageType
s.prop1 - s.prop20
s.campaign
s.state
s.zip
s.events
s.products
s.purchaseID
s.eVar1 - s.eVar20
var s_code=s.t();if(s_code)document.write(s_code)//--></script>
```

Page names are case sensitive, and differences will create additional page records. "Home" and "home" are two different pages within SiteCatalyst.



NOTE: Multiple page records cannot be combined within SiteCatalyst reports.

6.3.4 Custom Links

Custom links are link tags that have been implemented with code that is inserted in each link. This code identifies the link, and then calls a function in order to track the link. Additionally, if there are any properties, eVars, or events set in the link, the variables/events must be included in the .JS file in the "track" variables, as shown below.

```
s.linkTrackVars="prop1,eVar1"
s.linkTrackEvents="purchase,event1"
```


Use an Ethernet packet sniffer utility in order to test links and ensure that the image request is properly formed and correctly called. Ethereal (<http://www.ethereal.com>) is a Java-based IP packet sniffer used by the Engineering and Implementation Teams at Omniture.

Validate that links are reported in the Custom Links report. Ensure that the correct parameters are passed to the tl function. For more information on custom links, refer to the *Link Tracking* white paper.

6.3.5 Custom Variables

The table below outlines the custom variables used in SiteCatalyst.

Table: Custom Variables

Variable	Description
Traffic Properties	<p>Check the value of prop1 - 50. Here is a checklist of items to check.</p> <ul style="list-style-type: none"> Is the correct case used? "ValueA" is a different record than "valueA." You may wish to use all lower case since a very small subset of browsers will convert all variables to lower case. Are the values less than 100 characters in length? If not, some clipping of the values may occur. Are all the values in a single property variable related, or do some values look "out of place?"
eConversion Variables	<p>Econversion variables include eVar 1 – 50. Here is a list of issues to check for the following.</p> <ul style="list-style-type: none"> Is the correct case used? "ValueA" is a different record than "valueA." You may wish to use all lower case since a very small subset of browsers will convert all variables to lower case. Are the values less than 100 characters in length? If not, some clipping of the values may occur. Are all the values in a single eVar related, or do some values look "out of place?"
Custom Events	<p>Events include both standard values (prodView, scOpen, scAdd, scCheckout, purchase), as well as custom events from "event1" to "event20." All events are sent in the events variable, and multiple events on the same page should be comma-delimited (no white space).</p> <ul style="list-style-type: none"> For all the standard conversion events, products should also be populated with the applicable products. For all events except purchase, the "qty" and "price" elements are optional. For the purchase event, it must be set only once in a session, after the purchase has been completed and confirmed.

6.4 Implementation Acceptance

The following steps outline the implementation process.

1. The Omniture Implementation Consultant gathers report requirements and creates a data collection plan based on those requirements. The data collection plan includes variable definitions, required VISTA rules and custom

JavaScript, data correlation, and all SiteCatalyst settings for each report suite. The client completes the Implementation Questionnaire.

2. Technical resources on the client side implement the code, site-specific JavaScript, and server-side variables.
3. The Omniture Implementation Consultant addresses technical issues during the implementation, and assists in devising solutions as required.
4. Technical resources on the client side unit test the implementation by logging into SiteCatalyst and verifying all variables (page name, channel, server, events, campaign, econversion variables, custom traffic variables, products, and all other SiteCatalyst variables).
5. The client notifies Omniture that the implementation is complete. The client provides a validation sample (data sample) to the Omniture Implementation Consultant in order to validate data accuracy. (VISTA-generated report suites will be validated by comparing appropriate metrics. A client-Omniture agreement of the metrics to be validated for such report suites shall be made in advance, at the time of the VISTA rule creation.)
6. The client uses Maximine or other vendor to ensure that all pages contain the SiteCatalyst code. Through a business partnership with Omniture, Maximine offers an introductory report at about \$200.00 per site to spider all pages for specific content (such as the SiteCatalyst .JS file or <noscript> image). The Omniture Implementation Consultant can assist in suggesting the most appropriate reporting mechanism, and make the initial introduction to Maximine. (This is not necessary for "Implementation Verification", although Omniture does not guarantee that all pages are implemented - just that implemented pages are being accurately tracked).
7. The client faxes (or signs online) an Implementation Acceptance and Agreement for the appropriate site(s).
8. Once the acceptance has been received, the Omniture Implementation Consultant enables the "Omniture Best Practices - Implementation Verification" certification within the interface. This should be a prominent "Award" icon, applicable per report suite ID.
9. Optionally, the client may elect to contract with Omniture for monitoring services for key pages of the implemented site (generally, these are the primary "templates", home page, and critical entry pages). This monitoring software is described in a separate document, but tracks pages by loading and executing the page, then comparing the SiteCatalyst image request to a baseline stored in a database. If any differences are detected, the software notifies specified Omniture (AM/IE) and client personnel via email.

The following items help to ensure a successful implementation.

- Best Practices document, similar to this one, which is client-facing and explains the processes in detail.
- The Validation Document that the customer uses to unit test the implementation.
- An Implementation Acceptance and Agreement form for the client to sign.
- A monitoring application that continuously validates the tags.
- Verify the relationship with Maximine and formalize the touch points for handoff of the implementation testing.
- Utilities and/or tools for comparison of pageviews and/or orders. Those comparisons can get fairly difficult.
- A method and/or process to quickly obtain the debug log for a given day, by report suite ID.

6.4.1 Data Accuracy Validation

Data accuracy validation is a process of comparing SiteCatalyst report data with known and verifiable data points. The validation process should be completed by Omniture personnel, preferably by the Omniture Implementation Consultant (the person most familiar with the technical implementation details).

The preferred data points for this validation, in order of preference, are listed as follows.

- (Econversion sites) Comparison of econversion orders for a single day.
- Comparison of known "success events", especially logged data where IP address and other browser information generally stored in web server logs can be compared to the data collected by SiteCatalyst.
- Comparison of page views, although for large sites only an hour or less may be compared.



NOTE: Main, or default pages - such as "index.html" - often receive more automated or monitoring traffic and represent a greater difference to browser-based data collection than pages that are routinely more "internal" top the site.

All three types of validation require a "debug log" or data feed for the time period in question. This is generally one day or less.

It is expected that orders or success events can be measured to within 2-3% of actual values (sometimes reaching higher accuracy levels), using "standard" JavaScript-based implementations. This assumes an SSL page, since SSL pages are cached much less frequently, and by definition they should not be cached. An implementation with fully server-side image requests on an SSL page should come within about 0-1% of actual values. Non-secure pages may experience higher differences, but still within 5% of actual values.

When comparing page views for a single time period, it is expected that the page views can be accounted for within five percent of actual values, not including monitoring (such as Keynote or WhatsUpGold) or automated traffic, i.e. spiders, bots and scripts.

Data accuracy comparisons need to take into account the following items.

- QA or other types of "internal" testing that may be filtered by IP addresses or VISTA rules.
- "Smart" tags that only generate tags for certain types of orders or traffic.
- Queries for comparison must take into account what is being measured by the web site (not including returns, orders placed by customer service personnel, or other "special conditions").
- Ensure that the time zone differences between the query and the SiteCatalyst report suite match.
- Custom Keynote or similar traffic (Keynote Transaction, etc.) that measure the ordering process and may be reflected in tags but removed from ordering systems.
- Account for the client's de-duping processes
- Reloads of the order page (SiteCatalyst de-dupes the orders based on purchaseID).

7 Appendix A: Optimizing your SiteCatalyst Implementation

SiteCatalyst deployment is organized into three major steps. The first step involves pasting a snippet of HTML code onto each page (or page template) of a web site. The HTML code snippet is very small (400 to 1,000 bytes) and contains JavaScript variables and other identifiers that facilitate the data collection process.

In the second step, the code snippet calls a JavaScript library file, which contains SiteCatalyst-specific JavaScript functions used during metrics collection. If the SiteCatalyst code is implemented correctly, the time required for the browser to execute the JavaScript library file is usually negligible.

Finally, the library file makes an image request to an Omniture data collection server that collects the data being submitted and returns a 1x1 transparent image to the visitor's browser. The third step adds an insignificant increment to the total page download time. For more information, refer to the *Image Request Download Times* white paper.



NOTE: Customers can take additional steps to minimize SiteCatalyst overhead.

7.1 Variable Length

The length of SiteCatalyst variables can impact the size of the HTML code snippet, JavaScript library file, and image request. If a customer has many variables that are "long" (60 characters or more) the values can be replaced with shorter identifiers. Data classifications or VISTA rules can be used to translate the identifiers to "friendly" names.



NOTE: Most SiteCatalyst variables have a maximum of 100 characters (eVars have a maximum of 255). Internet Explorer allows an overall maximum of 2,048 characters in a GET "image request" URL. The image request limit applies not only to the variables, but also to information about the browser, operating system, and browser plug-ins (Netscape/Mozilla only).

7.2 HTML Code Snippet

Many SiteCatalyst customers have SiteCatalyst variables declared, but no value is assigned to the variable. Removing unused variables helps to reduce the page size.

7.3 JavaScript Library File

Page-specific variables should be placed in the HTML snippet. All other variables should be put in the JavaScript library file. The JavaScript library file is intended to be cached in the user's browser after the initial load, which limits the amount of data that needs to be downloaded.

7.4 Caching Directives

The JavaScript library file is intended to be cached in the user's browser after the first time the user loads a page. Omniture customers should ensure that their web servers are set up to take advantage of this functionality. For example, make sure that the `NO-CACHE` setting is set to "false." Additionally, ensure that the expiration date is sufficiently long. Make sure any proxy caches are set up with the correct configuration. The customer's web server documentation will provide more information.

7.5 Tables

Many web browsers will not start displaying the contents of a table until the browser has compiled the entire table. If the entire contents of the page are inside one big table, the browser will have to compile the entire contents of the page before anything will be displayed.

Placing the call to the JavaScript library file outside table tags ensures that the call to the SiteCatalyst servers does not impact the displaying of the page content.



NOTE: The file should be placed in a “legal” position for images, and must appear between the opening <body> tag and the closing </body> tag.

7.6 File Compression

Customers may compress the JavaScript library file by using standards-based encoding (i.e., gzip). Typical compression algorithms can reduce the size of the JavaScript file by 40-60% or more.



NOTE: Not all browsers support all file compression standards or interpret the compressed files in the same manner. Omniture does **NOT** guarantee reliable file compression in all environments. Customers should test compression in their supported browsers and configurations before deploying.

7.7 Secure Pages

Secure pages (pages loaded under https://) encrypt the image request and add to the total download time. Secure pages can add as much as 50-75% overhead to the image request. Customers should ensure that https is used only where necessary.

7.8 Content Delivery Services/Networks

Content Delivery Services or Content Distribution Networks (CDNs) such as Akamai and Speedera push Web content closer to the edge of the network - keeping frequently-requested documents close to the location where they are accessed. Typically, this reduces access latency, bandwidth usage, and infrastructure cost.

The SiteCatalyst JavaScript library file may be delivered via a CDN to enhance performance and delivery of the file to the site visitor. Omniture customers need to ensure that they have configured their CDN services correctly. CDNs are a common reason for fluctuations in download times and should be considered the most probable cause for any changes in download times.

7.9 JavaScript File Location and Concurrency

Most Web browsers download images concurrently. Typically three to four images can be downloaded simultaneously. Placing the call to the JavaScript library file at the top of the page ensures that the SiteCatalyst image is among the first elements to be downloaded.

Since most Web browsers download elements concurrently, the Status Bar of many common browsers (including Internet Explorer) does not accurately reflect which element the browser is trying to load. For example, your Status Bar may report that your browser is waiting for image 1 to download when network packet tests show your browser has already received image 1 and is currently waiting for image 2.



NOTE: Since third-party Internet Performance Audit providers (i.e. Keynote Systems) download page image elements sequentially, not concurrently, they do not mimic the typical user experience.

7.10 Peering

Private Network Peering enables data to pass from an ISP's network to the SiteCatalyst network more efficiently. Peering is simply the agreement to interconnect and exchange routing information without the need of the public network. Please contact Omniture Engineering if an ISP is interested in establishing a peering relationship.

Although peering may provide some benefits to Omniture ISP customers, optimizing the HTML snippet and the Web servers caching directives will likely have a much greater impact. High-volume ISP customers will realize the most benefit from peering.

8 Appendix B: Implementing SiteCatalyst without JavaScript

SiteCatalyst data collection is usually implemented using an HTML image tag that is created using JavaScript. The browser then requests the image. Data "piggybacks" this image request by placing variables into the query string of the image request. The JavaScript combines browser-level variables with page-level variables for a comprehensive data collection solution. In some cases, a fully server-created image tag may be appropriate. The standard elements of a JavaScript-based implementation are listed as follows.

Table: Standard Elements of a JavaScript-based Implementation

HTML Code	This portion consists of JavaScript code that is placed in HTML pages (or templates) which set the value of JavaScript variables, and then references a JavaScript library file which is then loaded and executed.
JavaScript Library	This file contains common code that (a) queries the browser about various properties such as JavaScript version, OS version, the size and resolution of monitor is being used, and other variables; (b) encodes and concatenates all the variables into an image request () that transports these variables to the SiteCatalyst data collection servers. This file is cached in the user's browser and only downloaded once per session.
<noscript> tag	A simplified version of the image request is placed within a <noscript> tag that executes if the user has disabled JavaScript, or does not have JavaScript capabilities. This part of the implementation is optional and generally only applies to approximately 2% of the Internet population.

JavaScript can detect browser settings that are not available to a server, such as browser window height/width, monitor resolution, and Netscape plug-ins. By using a server-side method to create an image tag, these variables cannot be captured. The JavaScript sets a random number in the image request in order to overcome browser and proxy server caching, and thus allows all pageviews to be accurately tracked. In certain situations, server-side code has advantages over the JavaScript-based code, including the following.

- JavaScript is very accurate (98-100%), but there are times when the utmost accuracy may be desired, even in situations where a user may quickly click to another page before the JavaScript has executed. Creating the image tag server-side increases the accuracy level by several percentage points.
- For https: (secure) connections, or for tracking conversion events, such as purchases, where accuracy is very important.
- This strategy may also be used to fully populate the image request within the <noscript> tag for tracking users without JavaScript, or with JavaScript disabled.



WARNING! The use of server-generated image tags requires additional time to implement, and may be more difficult to debug, deploy, and maintain. Omniture strongly encourages clients to use JavaScript-based data collection on every page where possible. Various reports and features, including ClickMap, Download Links, Exit Links and browser-based variables (browser width/height, etc.) cannot be collected or supported using this implementation method.

8.1 PHP Measurement Library

Omniture offers a PHP measurement library containing a PHP class that you can implement on your Web server to dynamically build non-JavaScript image requests. This class allows you to populate any SiteCatalyst variables and manage visitor IDs, server-side; to some extent. This solution automates the building of the image tag described in this section, making it very easy to get started on a non-JavaScript implementation using PHP. You can download the necessary PHP code from the Code Manager in the SiteCatalyst Admin Console. In addition, you can reference the Measurement Library (PHP) User Manual from the Omniture Help > Manuals page in the Omniture Suite.

8.2 Variable Names

The names of SiteCatalyst variables differ between the image request and the JavaScript variable name. The Implementation Manual uses the JavaScript variable name exclusively. The following table maps the JavaScript variable to the query string parameter name in the SiteCatalyst image request.

Table: JavaScript Variables

JavaScript Variable	Query String Parameter
s.pageName	pageName
s.server	server
s.pageType	pageType
s.channel	ch
s.prop1 – s.prop50	c1 through c50
s.campaign	v0
s.state	state
s.zip	zip
s.events	events
s.products	products
s.purchaseID	purchaseID
s.eVar1 – s.eVar50	v1 through v50
*Link Type	pe (lnk_d, lnk_e, lnk_o)
*Link URL	pev2
Referring URL	r
Current URL	g




NOTE: When using the image request to track links, the type of link (download=lnk_d, exit=lnk_e, or custom link=lnk_o) must be defined, as does the Link URL/Name (pev2). Note that links would require implementation by hand by inserting code within the <a href> tag.





Additionally, with server-generated image tags, the image request is executed by a web browser. If you need to send data directly from your servers to Omniture's servers, contact Omniture ClientCare to determine the best transfer method.

8.3 Other Requirements

The following table outlines additional requirements/configurations for implementing SiteCatalyst without JavaScript. Additionally, you can view sample code to further understand the implementation.

Table 8-A: Additional Requirements

Requirement	Description						
Case-Sensitive	The parameter names (pageName, purchaseID, etc.) are case-sensitive and will not properly record data unless they appear as designated in the table displayed in <i>Variable Names</i> in this document.						
Encode Query Parameters	<p> NOTE: The values for each of the query string parameters must be URL encoded. URL encoding converts characters that are normally “illegal” when appearing in a query string, such as a space character, into an encoded character beginning with “%.” For example, a space character is converted into “%20.”</p> <p>The JavaScript version of this function is called “escape” (and to decode, “unescape”). Microsoft IIS Version 5.0 also includes an “Escape” and “Unescape” function for encoding query strings. Other web server scripting languages also provide encoding/decoding utilities.</p>						
Maximum Variable Length	The maximum length of any single parameter is 100 characters. The exception to this list is Current URL and Referring URL, which may be up to 255 characters. Additionally, the “products” variable may be longer, but each individual “sub-field” (such as “Category” or “Product”) must not be longer than 100 characters.						
Invalid Characters	Characters with character codes above decimal 128 are invalid, as are not-printing character codes under 128. HTML formatting (“<h1>”) is also invalid, as are trademark, registered trademark, and copyright symbols.						
Secure (https:> vs. Non-Secure (http:) Image Requests	<p>ON pages that are accessed via https (secure protocol), the URL portion of the image request changes to accommodate a different set of data collection servers. The following table illustrates the different URLs used for secure and non-secure image requests.</p> <table border="1"> <thead> <tr> <th>Protocol</th><th>URL</th></tr> </thead> <tbody> <tr> <td>https:</td><td>https://namespace.<data center-specific...>.2o7.net/b/ss/reportsuite/1/G.5--NS/...</td></tr> <tr> <td>http:</td><td>http://namespace.<data center-specific URL*>.2O7.net/b/ss/reportsuite/1/G.5--NS/...</td></tr> </tbody> </table> <p> NOTE: The * in the URL above denotes a data-center specific URL that will be provided to you by your Omniture Implementation Consultant. Omniture uses several data centers, and it is necessary to implement the correct URL to which your organization has been assigned.</p> <p>For clients who use multiple report suites, the multiple report suites should be listed</p>	Protocol	URL	https:	https://namespace.<data center-specific...>.2o7.net/b/ss/reportsuite/1/G.5--NS/...	http:	http://namespace.<data center-specific URL*>.2O7.net/b/ss/reportsuite/1/G.5--NS/...
Protocol	URL						
https:	https://namespace.<data center-specific...>.2o7.net/b/ss/reportsuite/1/G.5--NS/...						
http:	http://namespace.<data center-specific URL*>.2O7.net/b/ss/reportsuite/1/G.5--NS/...						

	<p>only in the directory section, and not the domain section of the URL, as shown below.</p> <table border="1"> <tr> <th>Protocol</th><th>URL</th></tr> <tr> <td>https:</td><td>https://102.112.207.net/b/ss/<u>suite1,suite2</u>/1/G.5--NS/...</td></tr> <tr> <td>http:</td><td>http://<u>suite1</u>.112.207.net/b/ss/<u>suite1,suite2</u>/1/G.5--NS/...</td></tr> </table> <p> NOTE: The * in the URL above denotes a data-center specific URL that will be provided to you by your Omniture Implementation Engineer. Omniture uses several data centers, and it is necessary to implement the correct URL to which your organization has been assigned.</p>	Protocol	URL	https:	https://102.112.207.net/b/ss/ <u>suite1,suite2</u> /1/G.5--NS/...	http:	http:// <u>suite1</u> .112.207.net/b/ss/ <u>suite1,suite2</u> /1/G.5--NS/...
Protocol	URL						
https:	https://102.112.207.net/b/ss/ <u>suite1,suite2</u> /1/G.5--NS/...						
http:	http:// <u>suite1</u> .112.207.net/b/ss/ <u>suite1,suite2</u> /1/G.5--NS/...						
URL and Referring URL	<p>The URL and Referring URL may be populated from the server in the “g=” and “r=” variables. Use the “Request ServerVariables (“HTTP_REFERER”) or “Request ServerVariables (“URL”)” (IIS/ASP), or the appropriate variable for your server/scripting technology. The referring URL (r=) is extremely important for tracking referring URLs, domains, search engines, and search terms.</p> <p> NOTE: If “pageName” is not being used, it is imperative that the Current URL field is uniquely populated. If neither pageName nor Current URL (g=) are populated, the record is invalid and is not processed. At a minimum, the URL is a required field in order to process the record.</p>						
Effects of Caching	<p>HTML and other web pages can be cached by browsers or servers that are between the visitor and the web site that is serving the content. Caching prevents an accurate count of page views and other events unless a “cache-busting” technique is employed.</p> <p> NOTE: Omniture’s standard JavaScript includes a dynamic method of changing the image request to avoid page and image caching, allowing an accurate count of page views. However, in creating a server-side image request, this randomization does not occur. Page reloads and cached pages (either in the browser’s cache or in a proxy server) will not be counted in certain cases when using server-side image requests. SSL (https:) pages are not, by definition, ever cached so this warning applies only to non-secure (http:) pages. Additionally, pages with parameters (http://www.samplesite.com/page.asp?parameter=1) or certain file extensions (.asp, .jsp, etc.) are also not cached.</p> <p>The examples below also illustrate a “minimal JavaScript” solution that primarily assembles the image request server-side, and then tacks on a random number in the browser. This method overcomes the caching that would otherwise be encountered on static HTML pages accessed via the http: protocol.</p>						
nameSpace Variable	<p>The nameSpace query string parameter is required for non-JavaScript implementations.</p> <p>Example: ns=nameSpace</p> <p> NOTE: Contact your Omniture Implementation Consultant to obtain your</p>						

	organization's nameSpace value.
--	---------------------------------

8.4 Sample Code

The following examples illustrate the use of a server-generated image tag within a HTML sample page. The table below displays the values used in the sample.

Table 8-B: Values Used in the Sample Code

Variable	Value
pageName	Order Confirmation
Current URL	https://www.somesite.com/cart/confirmation.asp
events	purchase,event1
c1	Registered
purchaseID	0123456
products	Books;Book Name;1;19.95
state	CA
zip	90210
a random #	123456

8.4.1 Example 1

The example below displays a server-side image tag. The random number (in yellow) prevents caching of the image.

```
<html>
<head>
</head>
<body>
Order Confirmation<br>
Thanks for your order #0123456.

</body>
</html>
```

8.4.2 Example 2

The example below shows a minimal JavaScript image tag.

```
<html>
<head>
</head>
<body>
Order Confirmation<br>
Thanks for your order #0123456.
<script language="javascript"><!--
s.s_date = new Date();
s.s_rdm = s.s_date.getTime();
s.s_desturl="<img width=\"1\" height=\"1\"
src=\"https://102.112.207.net/b/ss/suite1,suite2/1/G.4--NS/" + s.s_rdm +
"?pageName=Order%20Confirmation&events=purchase%20Event1&c1=Registered
&purchaseID=0123456&products=Books%3BBook%20Name%3B1%3B19.95&state=CA&zip=90210&g=http
s%3A//www.somesite.com/cart/confirmation.asp\">";
document.write(s.s_desturl);
//--></script>
</body>
</html>
```

8.5 Supported SiteCatalyst Reports

Most of the out-of-the-box SiteCatalyst reports are available with a non-JavaScript implementation. However, some of the reports do require JavaScript in order to show data; i.e., the Browser Height report takes JavaScript information from the browser. Therefore, if you decide you do not want to implement with JavaScript, then you will not have every SiteCatalyst report available for use, but you can work with Omniture ClientCare to use other variables to populate those reports. For more information, contact Omniture ClientCare.

9 Appendix C: Implementing SiteCatalyst for Media Tracking

You can track media on your site by gathering basic information from the media player and building a session of events that are sent to Omniture's collection servers for processing. You can collect the following information using media tracking with Omniture.

- Media Name
- Media Length (in seconds)
- Media Player Name

Omniture's JavaScript collection file (version H.15 and higher) supports the media tracking module. If "s" is the name of your JavaScript object, then reference the media module as s.Media. With a custom implementation, you can manually track a media session by calling four functions. Alternatively, you can automatically track media by setting the s.Media.autoTrack variable to "true."



NOTE: Omniture collects data related to the portions of the video that the user views. The data is typically sent at when the user has finished viewing the video. Because a media player or browser window may be closed before the media has finished playing, the collection code buffers the media session in a cookie that is passed in with the next page view. The cookie and shared object are named "s_br." You can disable the cookie by setting the s.disableBufferedRequests variable to "true." This cookie is retained for 30 minutes because the data must be associated with the visit in which it occurred.

To implement media tracking for a JavaScript implementation, obtain a set of the Omniture JavaScript code (version H.15 or higher with the media tracking module installed), which you can get from Omniture ClientCare or your Omniture Implementation Consultant.



NOTE: Media tracking can be implemented either as an auto-track implementation or a custom implementation, as shown in the following sections.

9.1 Auto-track implementation

Follow the steps below to implement media tracking using the auto-track feature.

1. In the s_code.js file, set s.Media.autoTrack=true.
2. Set s.Media.playerName to your chosen media player name as you want it to display in SiteCatalyst.

The media module will then "listen" for media activity such as video starts, video stops, and other such activities.

9.1.1 Sample Code

```
}  
s.doPlugins=s_doPlugins  
  
s.loadModule("Media")  
s.Media.autoTrack=true  
s.Media.trackVars="None"  
s.Media.trackEvents="None"
```

9.2 Custom implementation

Follow the steps below to implement explicit function calls to track media events at selected time periods.

1. Make sure JavaScript can be applied to the media player you want to track.



NOTE: Supported media includes Flash, Windows Media (in Internet Explorer only), Real Player, and Quicktime (Firefox and Safari only).

2. Identify the points at which you want to call the four functions (open, play, stop, close).
 - a. For example, when you select a video to view, you may want to call the s.Media.open function to initialize tracking of the video. Similarly, if you pause a video, you would call s.Media.close and then s.Media.play if video playback is started again.
3. In your code, call the functions at the appropriate times. See the sample code below.

9.2.1 Sample Code

```
function loadMovie(movieName, movieLength){
/* Code to retrieve and start the movie */

/*
 * Omniture Video Tracking
 *   Open the movie
 */
var s=s_gi('report_suite_id');
s.Media.open(movieName, movieLength, "Example Media Player 1.2")
}

function playMovie(movieName, offset){
/* Code to begin playing the movie */

/*
 * Omniture Video Tracking
 *   Open the movie
 */
var s=s_gi('report_suite_id');
s.Media.play(movieName, offset)
}

function pauseMovie(movieName, offset){
/* Code to pause the movie */

/*
 * Omniture Video Tracking
 *   Pause the movie
 */
var s=s_gi('report_suite_id');
s.Media.stop(movieName, offset)
}

function endMovie(movieName, offset){
/* Code to end playing the movie */

/*
 * Omniture Video Tracking
 *   Stop and close the movie
 */
var s=s_gi('report_suite_id');
s.Media.stop(movieName, offset)
s.Media.close(movieName)
}
```

10 Appendix D: Implementing SiteCatalyst on Mobile Sites

Increasingly, people are using mobile devices such as cell phones, personal data assistants, and portable media players to access information via the internet. Because of the increasing use of Mobile devices, Omniture gives you the option of tracking mobile users in SiteCatalyst, Discover, Data Warehouse, and other Omniture applications to effectively monitor those accessing your Web sites using mobile devices.

Because most mobile devices do not currently support JavaScript, the standard JavaScript tracking beacon cannot be used. To track mobile device users, you need to implement an alternative tracking method on your Web pages. You can use JavaScript and the mobile tracking method concurrently.

Since many mobile devices do not execute JavaScript, the Omniture best practice is to deploy a hard-coded tracking beacon for mobile sites. For information on configuring SiteCatalyst to report mobile devices, refer to the *Mobile Tracking User Guide* and refer to *Implementing without JavaScript* in this document.

In addition, refer to the following table, which shows the resulting URL parameters you can populate.

Table: URL Parameters

Query String Parameter	JavaScript Variable Equivalent
gn	s.pageName
sv	s.server
gt	s.pageType
ch	s.channel
c1 - c50	s.prop1 – s.prop50
h1 - h5	s.hier1 - s.hier5
v0	s.campaign
state	s.state
zip	s.zip
ev	s.events
pl	s.products
pi	s.purchaseID
v1 - v50	s.eVar1 – s.eVar50
ce	s.charSet

cc	s.currencyCode
pe (lnk_d, lnk_e, lnk_o)	Link Type
pev1	Link URL
pev2	Link Name
pev3	Video Reports
r	Referring URL
g	Current URL
D	s.dynamicVariablePrefix
cdp	s.cookieDomainPeriods
cl	s.cookieLifetime (s_vi cookie lifetime in seconds)
/5/ or /1/	s.mobile

Index

Additional Implementation Resources	8	Implementing Omniture	1
Adoption Phase	4	Implementing SiteCatalyst	5
BODY tags	86	Installed Plug-ins	77
browserHeight	15	javaEnabled	32
browserWidth	15	JavaScript Code	6
campaign	15	JavaScript Debugger	83
channel	17	javascriptVersion	33
charSet	17	linkDownloadFileTypes	33
Code Modifications	87	linkExternalFilters	34
Code to Paste	5	linkInternalFilters	35
Collecting Data	7	linkLeaveQueryString	37
colorDepth	18	linkName	38
Confirmation Page	67	linkTrackEvents	38
connectionType	18	linkTrackVars	39
Content Hierarchies	65	linkType	41
cookieDomainPeriods	19	Media Tracking	101
cookieLifetime	19	Media.trackEvents	45
cookiesEnabled	20	Media.trackVars	45
currencyCode	21	mediaLength	41
Custom Events	67	mediaName	42
Custom Insight Variable	69	mediaPlayer	43
Custom Links	88	mediaSession	44
Custom Variables	89	Moving from Page Template to Page Template	74
Data Accuracy Validation	90	Moving from Section to Section	74
dc	22	Optimize Phase	5
Define Phase	2	Packet Monitors	86
Deploy Phase	4	pageName	47
Deployment Process	2	pageType	48
Deployment Validation	86	pageURL	49
Design Phase	3	Pathing	73
doPlugins	22, 76	Pathing by Campaign or Tracking Code	73
dynamicAccountList	23	Pathing on a prop	73
dynamicAccountMatch	24	PHP Measurement Library	95
dynamicAccountSelection	25	plugins	50
dynamicVariablePrefix	26	Plug-ins	77
Email Campaign Tracking	67	Predefined Events	66
Errors	87	Preface	i
eVar	64	Product View Page	66
eVarN	26	products	51
events	28	Products	70
Field Definitions of the Product String	71	propN	53
First Checkout Page	67	props as Counters	64
fpCookieDomainPeriods	30	props vs. eVars	64
Fusion	1	purchaseID	54
getQueryParam	79	Reasons Pathing may not be Recorded	74
Getting Started	1	referrer	54
getValOnce	81	Renaming doPlugins	76
HEAD tags	86	resolution	55
hierN	31	s.prop	64
homepage	32	s_account	13
Illegal JavaScript Characters	13	s_objectID	46
Implementation Acceptance	89		

s_usePlugins	60	SiteCatalyst JavaScript Compression.....	7
Sample Product Strings.....	71	SiteCatalyst Variables	9
Secure protocol.....	87	state	56
Segmenting Paths by User Type.....	75	trackDownloadLinks.....	57
server.....	55	trackExternalLinks.....	58
Setting a Product with an Event	70	Tracking External Email	69
Setting a Product without an Event	71	trackInlineStats	59
Setting products with a Purchase Event.....	52	transactionID	59
Setting products with Custom Incrementor		Using doPlugins	76
Events.....	52	Variable Names	96
Setting products with Non Purchase Events ..	51	visitorNamespace.....	61
Setting the Product Variable	70	Welcome to SiteCatalyst	1
Shopping Cart Page.....	66	zip.....	62
Simple Debugger.....	84		



CALL 1.877.722.7088
1.801.722.0139

www.omniture.com
info@omniture.com

550 East Timpanogos Circle
Orem, Utah 84097

OMNITURE™
— — —