StormPost

A DATRAN MEDIA COMPANY

DATA INTEGRATION OVERVIEW

STORMPOST OVERVIEW

StormPost is a web-based email marketing platform, providing clients with the ability to create, test and manage robust email programs.

Organizations utilize StormPost to send highly personalized, targeted email campaigns. With over two dozen reports, StormPost meets the various reporting needs of its client base. Each report presents clear and concise data pertaining to a campaign's performance and/or delivery statistics. StormPost is available as a software license or a hosted ASP. The two StormPost options share a common architecture, providing all StormPost customers with the same outstanding features and support. The common architecture between the two options makes moving from a hosted to license system (or vice versa) a seamless process. Clients adopting an alternative option may do so without the worry of excessive upgrade and switching costs, or data migration issues.

StormPost is enterprise-ready and supports databases housing millions of unique records with full demographic, geographical and behavioral information. In addition, StormPost boasts speeds of greater than one million messages per hour, meeting client needs for time sensitive campaigns.

DATA INTEGRATION OVERVIEW

StormPost provides clients with the ability to automatically import new member data directly into their mailing lists. Once added to one or more lists, clients may configure StormPost to send new members a real time triggered message. Sending an Event Triggered Message immediately after members perform an action helps establish brand awareness, as well as build strong relationships with new list members. Actions that generate an Event Triggered Message can range from subscribing to a new list, updating subscription settings or purchasing products or services.

Using StormPost to send real time messages provides visibility into the performance of transactional emails such as order confirmations, welcome messages, and other important notifications. This type of insight allows client to make any necessary changes to their email program to help increase ROI.

Clients may transfer data to StormPost using one of two methods:

- **SOAP API** – Administrators set up a connection to make SOAP calls directly to the StormPost application server.

- **FTP / Directory Watch** – StormPost automatically imports a file when the file is added to a specified directory. For customers in the hosted environment, the Datran Media Professional Services team can assist with writing custom sFTP scripts to pull files from external FTP servers.

! Datran Media does not allow clients to push files into the hosted environment for security reasons.

Figure 1. Example of how StormPost integrates with various systems.

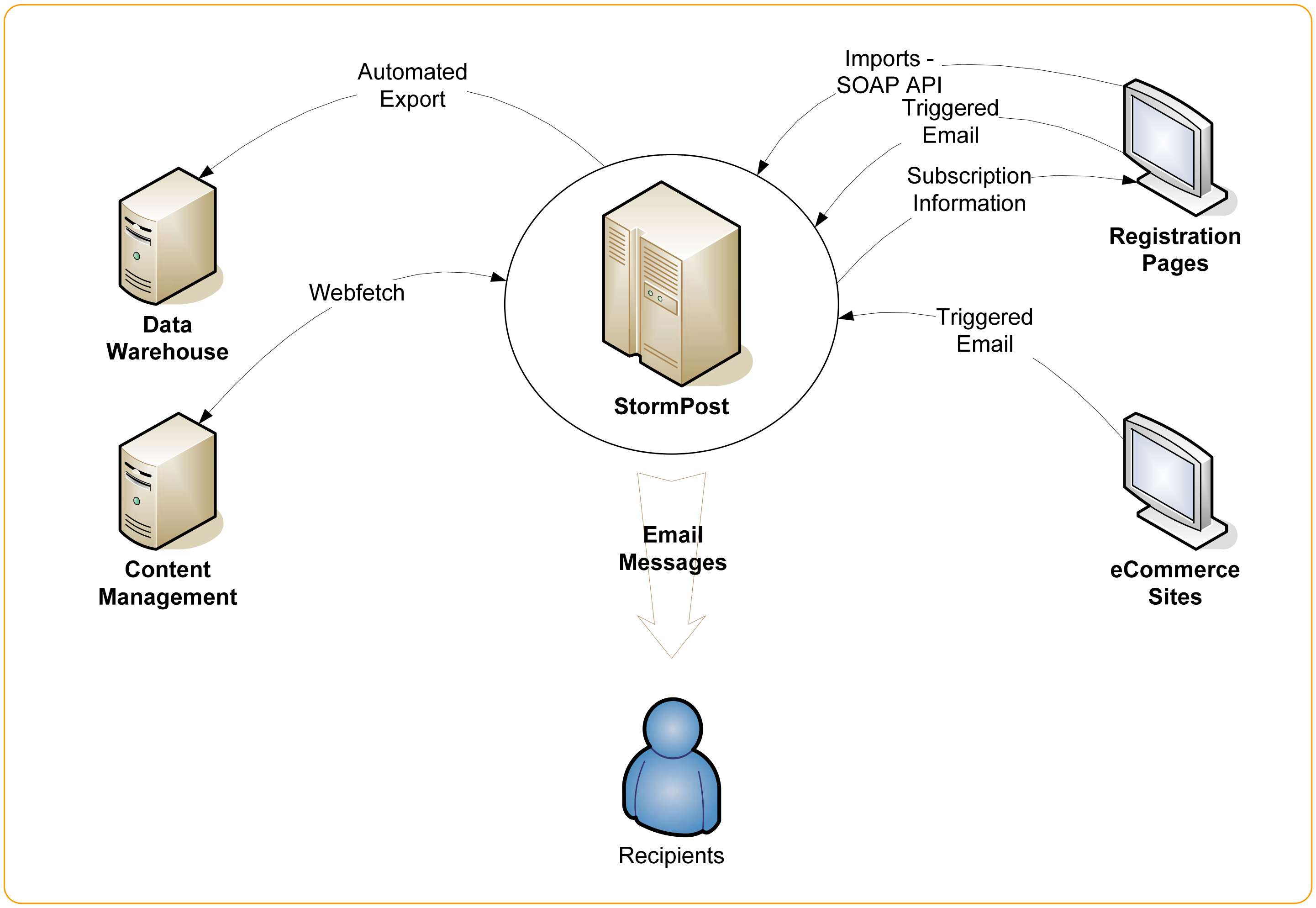


Figure 1

IMPORT & SEND TEMPLATES

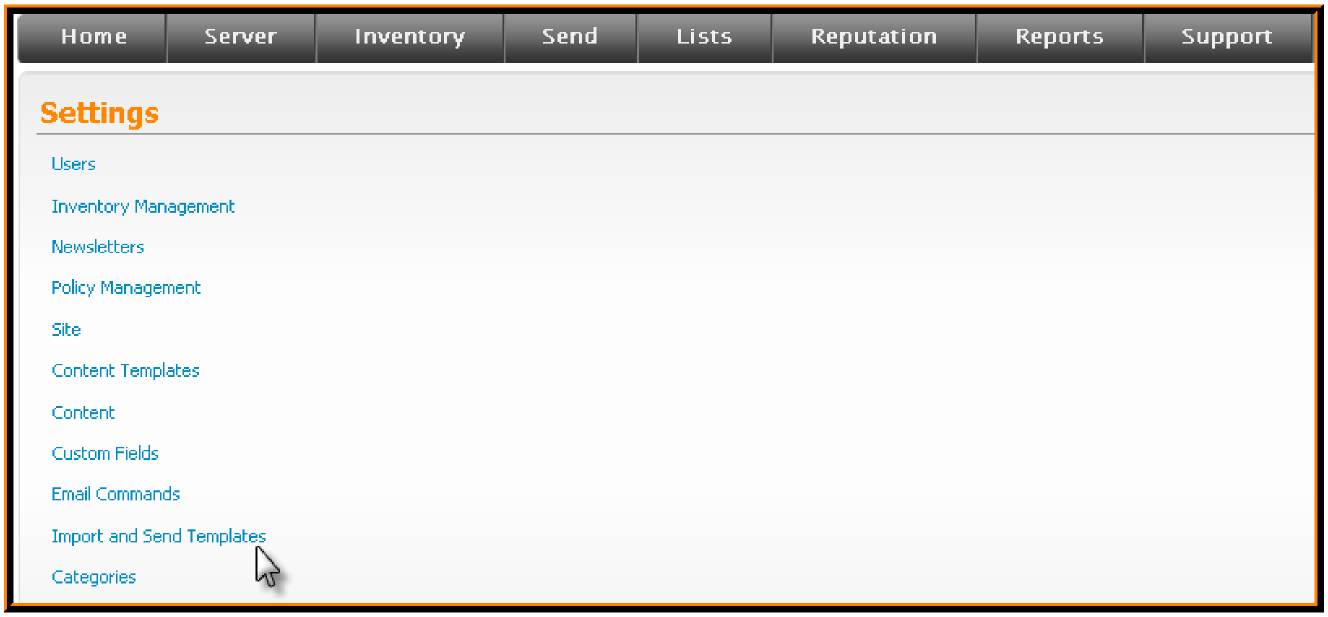
StormPost data integration usually relies on the use of an Import Template or a Send Template. These templates allow clients to:

- Configure all options for automatically importing data and the Event Triggered Messages
- Reference specific templates when using an API call or watch directory

CREATING AN IMPORT TEMPLATE

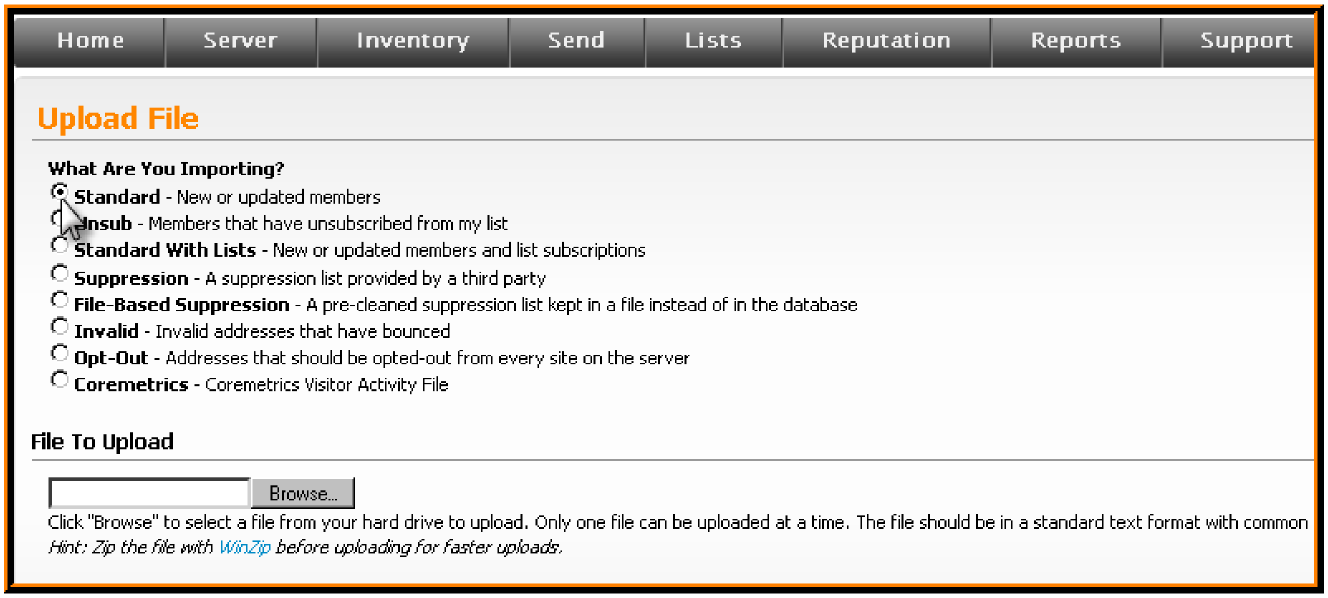
1. Log into StormPost, per normal procedure.

2. Click on the **Settings** tab.



3. Click on **Import and Send Templates**.

4. Click on **Create New Import Template**.



Upload File

What Are You Importing?

- [✓ **Standard** - New or updated members]
- **Unsub** - Members that have unsubscribed from my list
- **Standard With Lists** - New or updated members and list subscriptions
- **Suppression** - A suppression list provided by a third party
- **File-Based Suppression** - A pre-cleaned suppression list kept in a file instead of in the database
- **Invalid** - Invalid addresses that have bounced
- **Opt-Out** - Addresses that should be opted-out from every site on the server
- **Coremetrics** - Coremetrics Visitor Activity File

File To Upload

Browse...

Click "Browse" to select a file from your hard drive to upload. Only one file can be uploaded at a time. The file should be in a standard text format with common

Hint: Zip the file with WinZip before uploading for faster uploads.

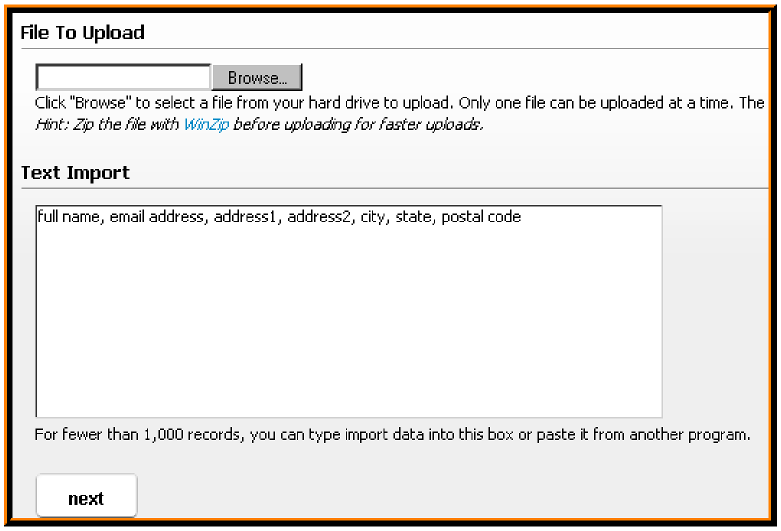
The process to create an import template is similar to the process used to manually import a file.

5. Click on the **Standard** radio button to import new members into the StormPost database.

Users may select a different import type based on the file type. For example: Suppression or Unsub.

6. Configure the data fields for the import template by:
   a. Selecting a **File to Upload** (or)
   b. Inserting the name of the data fields into the **Text Import** field.

7. Click on **Next**.



File To Upload

Browse...

Click "Browse" to select a file from your hard drive to upload. Only one file can be uploaded at a time. The

Hint: Zip the file with WinZip before uploading for faster uploads.

Text Import

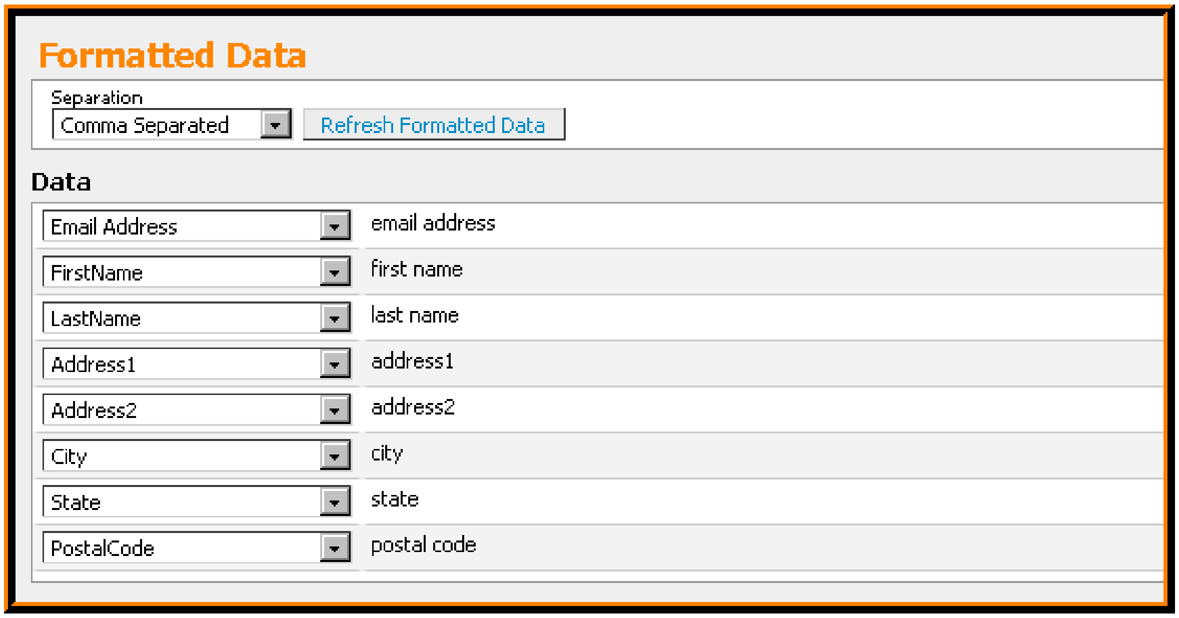
full name, email address, address1, address2, city, state, postal code

For fewer than 1,000 records, you can type import data into this box or paste it from another program.

next

❗ List only one email address per line and delimit fields by commas, tabs or pipes.

For example: *email address, full name, address1, address2, city, state, postal code*



Formatted Data

Separation
Comma Separated | Refresh Formatted Data

Data

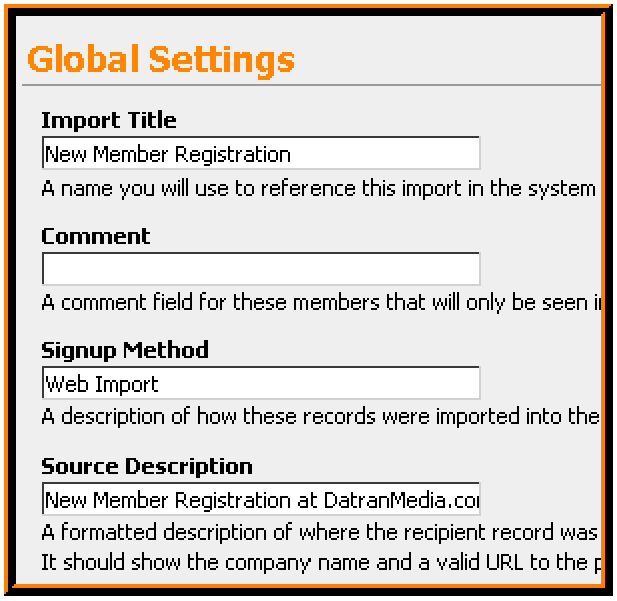
Email Address	email address
FirstName	first name
LastName	last name
Address1	address1
Address2	address2
City	city
State	state
PostalCode	postal code

8. Click on the **Data** field turnkeys to select the appropriate field names from the drop down menus. The selected fields determine how StormPost will map the recipient information to the database.

9. Change the **Import Title** from “User Input” to the name of the import template.

10. Enter the source signup URL or description into the **Source Description** field.

❗ The Source Description field is a required field.



Global Settings

Import Title
New Member Registration
A name you will use to reference this import in the system

Comment

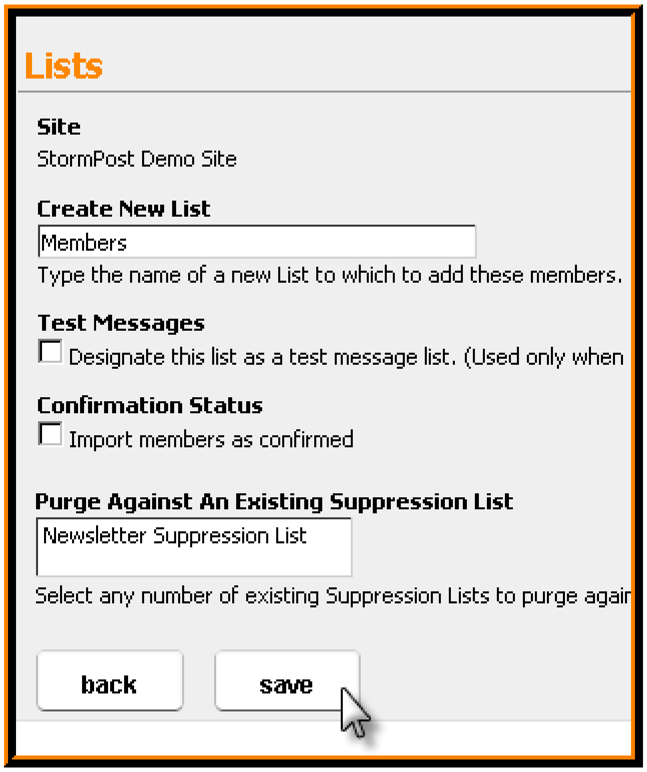
A comment field for these members that will only be seen i

Signup Method
Web Import
A description of how these records were imported into the

Source Description
New Member Registration at DatranMedia.coı
A formatted description of where the recipient record was
It should show the company name and a valid URL to the p

11. Enter the name of the mailing list into the **Create New List** field.

12. Click on **Save** to complete the creation process.



Lists

Site
StormPost Demo Site

Create New List
Members
Type the name of a new List to which to add these members.

Test Messages
☐ Designate this list as a test message list. (Used only when

Confirmation Status
☐ Import members as confirmed

Purge Against An Existing Suppression List
Newsletter Suppression List
Select any number of existing Suppression Lists to purge agair

CREATING A SEND TEMPLATE

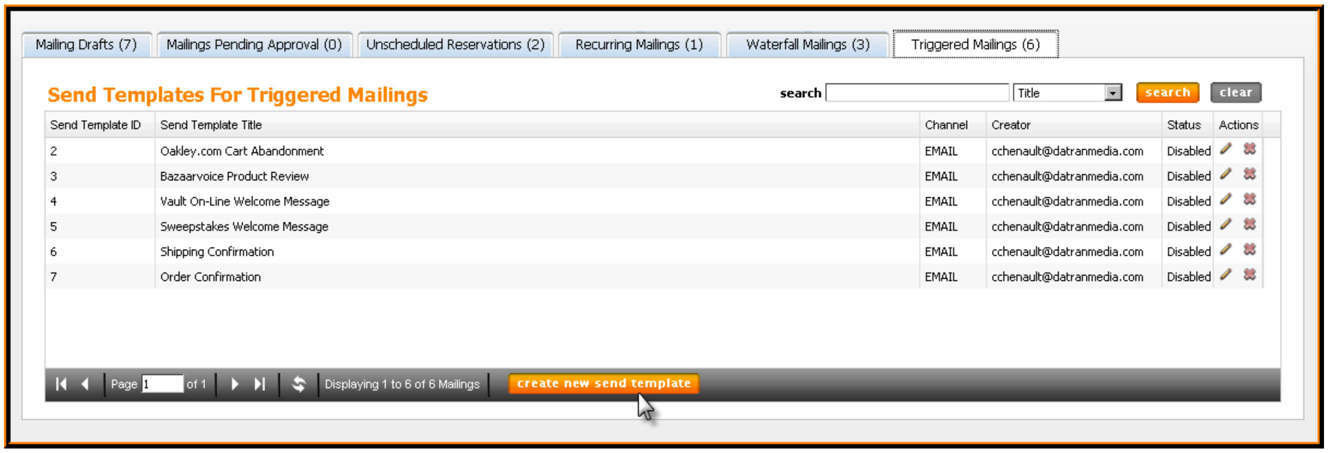
The process to build the send template is the same exact process used to schedule a one-off mailing, with one exception:

- The **Schedule** tab is not available for triggered mailings. StormPost will send the message immediately after receiving the API call.

Clients on StormPost 5.5 and above will access their send templates through the **Send** tab.
Clients on earlier versions of StormPost will access their send templates through the **Settings** tab.

1. Click on the **Send** tab.

2. Click on the **Triggered Mailings** tab, located in the Mailings section.



Send Template ID	Send Template Title	Channel	Creator	Status	Actions
2	Oakley.com Cart Abandonment	EMAIL	cchenault@datranmedia.com	Disabled	
3	Bazaarvoice Product Review	EMAIL	cchenault@datranmedia.com	Disabled	
4	Vault On-Line Welcome Message	EMAIL	cchenault@datranmedia.com	Disabled	
5	Sweepstakes Welcome Message	EMAIL	cchenault@datranmedia.com	Disabled	
6	Shipping Confirmation	EMAIL	cchenault@datranmedia.com	Disabled	
7	Order Confirmation	EMAIL	cchenault@datranmedia.com	Disabled	

3. Click on **Create New Send Template**.

4. Create the send template, following the same process used to build a one-off mailing.

WEBFETCH

StormPost's **webfetch** feature allows seamless integration with any content management system, capable of publishing content to a URL accessible to StormPost.

Webfetch retrieves the content at the time of message assembly and caches it locally. StormPost will then:

- Process the content
- Populate any available mailing tags with recipient data
- Assemble and send the message

Clients may make multiple webfetch calls within the body of a single message.

Common uses for webfetch include recurring newsletters, news alerts and other ongoing mailings.

To meet the content needs of various clients, StormPost supports the following:

- **Static Webfetch** – retrieves the same content for all recipients
- **Dynamic Webfetch** – retrieves different content depending on recipient attributes
- **Subject Line Webfetch** – retrieves content from the <title> of the URL and plugs it into the subject line

	Description	Tag Replacement	Velocity Variable Replacement	Velocity Dynamic Content	Open Tracking	Link Tracking	Click Stream Tracking
STATIC	Mailing Tag: *[-STATIC_WEB_FETCH(URL)-]* URL = Fully qualified URL that provides the content Example: [-STATIC_WEB_FETCH_(www.datran.com)-] Clients may embed the static webfetch call anywhere within the text or HTML body of the email message.	Yes	Yes	Yes	Yes	Yes	Yes
DYNAMIC	Example: *$Web.fetch("http://www.xyz.com/getcontent.asp?gender=$_gender")* Dynamic content indicates that the retrieved content will be different for all or some of the mailing recipients. For example, the following Web Fetch call listed above will display different content based on a member's gender. Clients may embed the dynamic webfetch call anywhere within the text or HTML body of the email message.	No	No	No	Yes	No	No
SUBJECT LINE	Mailing Tag: [-static_title_web_fetch(<URL>)-] URL = Fully qualified URL that provides the content The subject line webfetch returns the content of the <title> tag found at the specified web page. Clients can only embed the subject line webfetch call in the subject line of the email message.	Yes	NA	NA	NA	NA	NA

For more information about StormPost feature, please refer to the WebFetch document located in SalesForce Solutions.

SOAP API

Access to StormPost is also available through a SOAP API exposed through the main web interface. SOAP is a simple, portable way to make remote procedure calls over HTTP. Clients may use SOAP with Perl, Java, Python, C, C++, PHP and many other programming languages.

For more information on SOAP, visit: http://www.soapware.org

All SOAP APIs for StormPost are made available by connecting to:

https://api.stormpost.datranmedia.com/services/SoapRequestProcessor

The Web Service Description Language (WSDL) file for the StormPost SOAP API is made available by connecting to:

http://api.stormpost.datranmedia.com/services/SoapRequestProcessor?wsdl

The WSDL document describes the operations and the complex data types used as input or output parameters for retrieving information via SOAP calls. Many programming languages -offer tools that have the ability to convert the WSDL file into a set of stub classes, turning the SOAP calls into the API.

The client establishes authentication by passing the **username** and **password** parameters in the SOAP header.

❗ The username and password must map to a valid StormPost user login.

❗ Clients can only use a site level administrator's login for their API calls.

💡 The Integration Consulting team recommends that clients create a separate user account for this sole purpose. Creating a separate account prevents accidental deletion. Furthermore, clients may deselect the permission for **Admin Privileges** to ensure users do not use the account to log into the StormPost interface.

The following methods are available for clients to integrate StormPost with their internal applications and websites.

SOAP API VERSION

getVersion

Input	None
Output	Version of the SOAP API (String)
Notes	Returns the version number for the SOAP API as a String. This method takes no arguments and does not require authentication.

SUBSCRIPTION MANAGEMENT

RECIPIENTS

📝 Clients use the following SOAP methods for real-time updates to their StormPost database from either a subscription page or other web form.

getRecipientByAddress

Input	arg1	Email Address of Recip
Output	Recipient (object)	
Notes	Retrieves an existing recipient. A remote exception will be thrown if this call is attempted on an external ID system.	

getRecipientByExternalID

Input	arg1	External ID of Recip
Output	Recipient (object)	
Notes	Retrieves an existing recipient.	

createRecipient

Input	arg1	Recipient Object
Output	There is no output or return value.	
Notes	This method creates a new recipient identified by the address or external ID, depending on database type. The recipient will have NORMAL status. Values marked read-only (see the Recipient object below) An exception is thrown if: ▪ The address is not specified. ▪ The external ID is not specified for an external ID database. ▪ A recipient with that email address (external ID) already exists. (The exception message, in this case, should contain the status of the existing recipient). ▪ A demographic field name does not exist in the database. ▪ The value of a demographic field cannot be parsed as the correct type. ▪ The email address matches an import stop word.	

updateRecipient

Input	arg1	Recipient Object
Output	There is no output or return value (unless error is thrown)	
Notes	This method updates an existing recipient based upon the recipID. Read-only values (see the Recipient object below) will be ignored. An exception is thrown if: • The recipID is invalid • An email address change or external ID change causes a collision with another recipient. • A demographic field name does not exist in the database. • The value of a demographic field cannot be parsed as the correct type. • The email address matches an import stop word.	

Object: Recipient

A **Recipient** object consists of:	
RecipID	**(Read Only)** Unique number assigned to every record (int)
Address	Email Address of the recipient (string)
ExternalID	External ID of the recipient (string)
Protocol	**(Read Only)** (string)
Status	**(Read Only)** (string)
Comment	Note about the recipient (string)
ImportID	**(Read Only)** ID of the import that first introduced the recipient to the system (int)
Password	Password stored for the recipient (string)
Signup Method	Method the recipient used to signup (string)
Signup IP	IP where the record signed up (string)
SourceSignupDate	Date the recipient signed up (Date)
SourceDescription	Description of where the recipient signed up (string)
ThridPartySource	Third Party the recipient should be attributed to (string)
ThirdPartySignupDate	Date the recipient signed up with the third party (date)
DateLastClicked	**(Read Only)** Date the record last clicked on mailing (date)
DateLastOpened	**(Read Only)** Date the record last opened a mailing (date)
ClickCount	**(Read Only)** Number of times the recipient has ever clicked (int)
OpenCount	**(Read Only)** Number of times the recipient has ever opened (int)
NumBounces	**(Read Only)** Number of bounces the recipient currently has (int)
Block Code	**(Read Only)** (string)
DateJoined	**(Read Only)** Date the member was imported into StormPost (date)

DateBounce	**(Read Only)** Date that the record last Bounced (date)
DateHeld	**(Read Only)** Date that the record was Held (date)
DateUnsub	**(Read Only)** Date that the record was UNSUB (date)
DateOptout	**(Read Only)** Date that the record was Optout (date)
Demographics	Array of NameValue (name=value)

LISTS

getList

Input	ID of the List (int)
Output	List (object)
Notes	Returns data about a list. .

getLists

Input	Search Values of List
Output	Array of List
Notes	Returns data about a list. None of the array fields are required and leaving all fields empty will return all lists.

Object: List

A **List** object consists of:	
List ID	ID of the List (int)
ListTitle	Title of the List (string)
Description	Descrription field for the list (string)
CreateTime	Date the list was created (date)
Creator	Admin who created the list (string)

Populated	Indicates whether the List Recips table is populated (Boolean)
SeedListID	ID's of any seed lists for this list (int)
Global Unsub	Indicates if Global Unsub is turned on. (Boolean)
PublicSignup	Indicates if Public Signup is turned on. (Boolean)
BlockDomains	List of Block Domains for this list (array of string)
CountRecips	Indicates if Track Member Counts is turned on (Boolean)
CategoryID	(int)
ExternalID	External ID field tied to the list (string)
FriendlyFromName	Friendly From Name tied to the list (string)
FriendlyTitle	Friendly Title tied to the list (string)
Custom1	Custom field tied to the list (string)
FacebookList	Indicates whether this is a Facebook list (Boolean)
DisplayOrder	Order the list should appear on the subscription page (int)

LIST SUBSCRIPTIONS

getRecipientSubscriptions

Input	arg1	recipID (int)
Output	Array of ListSubscriptions (object).	
Notes	Retrieves an existing recipient.	

getRecipSubscriptionInfo (*Deprecated, to be removed as of StormPost 6)*

Input	arg1	The ID of the site (int)
	arg2	The email address of the recipient for which to retrieve information (string)
Output	SubscriptionInfo (Object)	
Notes	Retrieves list subscription information for a recipient.	

subscribeToList

Input	arg1	**(Required)** The ID of the Recip (int)
	arg2	The email address of the intended recipient (String)
	arg3	**(Required)** The ID of the List (int)
	arg4	The ID of the Mailing
	arg5	sourceID
Output	There is no output or return value (unless error is thrown)	
Notes	Subscribes a recipient to a list. An exception is thrown if the listID or recipID is invalid. The values of sourceID and mailingID are optional. If the call is made for a recipient that is already subscribed to the list, no exception will be thrown. The recipient will remain subscribed. If mailingID is specified and the recipient is successfully subscribed to the list, a subscribe Response will be added for the mailing.	

unsubscribeFromList

	arg1	**(Required)** The ID of the Recip (int)
	arg2	**(Required)** The ID of the List (int)
	arg3	The ID of the Mailing (int)
Output	There is no output or return value (unless error is thrown)	

Notes	This method unsubscribes a recipient from a list. An exception is thrown if the listID or recipID is invalid. The mailingID is optional. No exception will be thrown if the recipient is not subscribed to the list. In this case the recipient will remain unsubscribed. If mailingID is specified and the recipient is successfully unsubscribed from the list, an unsubscribe Response will be added for the mailing.

globalUnsubscribe

	arg1	**(Required)** The ID of the Recip (int)
	Arg2	The ID of the Mailing (int)
Output	There is no output or return value (unless error is thrown)	
Notes	This method set the recipient's status to UNSUB. An exception is thrown if the recipID is invalid. The mailingID is optional. If mailingID is specified and the recipient is successfully unsubscribed, an unsubscribe Response will be added for the mailing.	

globalUnsubscribeAll

	arg1	**(Required)** The email address of the intended recipient (String)
	Arg2	The ID of the Mailing (int)
Output	There is no output or return value (unless error is thrown)	
Notes	This method unsubscribes all recipients with the email address. The mailingID is optional. If mailingID is specified and the recipient is successfully unsubscribed, an unsubscribe Response will be added for the mailing.	

Object: List Subscription

A **ListSubscription** object consists of:	
ListID	ID of the List (int)
ListTitle	Title of the List (string)
Subscribed	Indicates if the recipient is subscribed to the list (Boolean)
Confirmed	Indicates if the recipient is confirmed to the list (Boolean)
DateJoined	Date the member joined the list (date)
DateUnsubbed	Date the member was unsubscribed from the list (date)
SourceID	(String)

(Deprecated, to be removed as of StormPost 6)

A **SubscriptionInfo** consists of:	
recipFields	An array of RecipField
listSubscriptions	An array of ListSubscription

(Deprecated, to be removed as of StormPost 6)

A **RecipField** consists of:	
name	The name of the attribute (string)
value	The value of the attribute (string)

IMPORTING DATA

getImportTemplates

Input	None
Output	Array of IdAndTitle objects, each containing the ID (int) and title (String) of an import template.
Notes	Returns IDs and Titles of the import templates available to the user. This method takes no arguments.

doImportFromTemplate

Input	arg1	ID of the import template that describes the format of the data being imported (int)
	arg2	The data to be imported – one record per line, all records in one string. (String)
Output	‘Request successfully processed’ (String)	
Notes	Performs an import of the provided data using the specified import template ID. The data set size is restricted to a maximum of 500 records. The Import Template must be created in advance in the StormPost interface under the Settings tab. This method will throw an error if the Import Template is invalid or unavailable. If individual lines fail but the import as a whole is successful, the call will return as a success. Returns either a “Request successfully processed” message or a SOAP fault error.	

doImportAndSendFromTemplate

Input	arg1	ID of the import template that describes the format of the data being imported (int)
	arg2	ID of the send template that should be sent to the imported members (int)
	arg3	The data to be imported – one record per line, all records in one string. (String)
Output	Request successfully processed’ (String)	
Notes	Performs an import-and-send on the provided data using the specified import template ID and send template ID. The data set size is restricted to a maximum of 500 records. The Import Template and Send Templates must be created in advance in the StormPost interface under the Settings tab. This method will throw an error if the Import Template is	

	invalid, unavailable or is incompatible with a Send Template, or if the Send Template is invalid. If individual lines fail but the import as a whole is successful, the call will return as a success. Returns either a "Request successfully processed" message or a SOAP fault error.

getImportStatus

Input	arg1	The ID of the import (int)
	arg2	(optional) The name of the file if the import was triggered by a directory watch import. (string)
Output	Import status (string)	
Notes	Retrieves the import status for an existing import job. Possible status returns are: *Importing, Quarantine, Approving, Failed, Passed, Held, Bypass(indicates success).*	

SENDING MESSAGES

SEND TEMPLATES

Please refer to the following data points when sending triggered messages to individual recipients though StormPost, in real-time.

getSendTemplates

Input	None
Output	Array of IdAndTitle objects, each containing the ID (int) and title (String) of a send template.
Notes	Returns IDs and Titles of the send templates available to the user. This method takes no arguments.

createSendTemplate

Input	arg1	A sendTemplateBean describing the send template's parameters. The SendTemplateBean is defined above.
	arg2	The text content of the send template, or a blank string if there is none (String)
	arg3	The HTML content of the send template, or a blank string if there is none (String)
Output	The ID of the new Send Template (int)	
Notes	Creates a new Send Template record with the specified field values. The to, from and reply-to name and e-mail address fields may contain the same substitution tags which can be used when creating a Send Template through the web interface. Returns either the ID of the new send template or a SOAP fault error.	

getSendTemplateContent

Input	arg1	The ID of the send template (int)
	arg2	The part to retrieve. Must be “TEXT” or “HTML”. (String)
Output	The content part (String)	
Notes	Returns the contents of the given part for the send template.	

getSendTemplateDefinition

Input	arg1	The ID of the send template (int)
Output	A sendTemplateBean specifying the send template's fields. The SendTemplateBean is defined above.	
Notes	Retrieves the definition for the send template.	

updateSendTemplateDefinition

Input	arg1	The ID of the send template to update (int)
	arg2	A sendTemplateBean specifying the fields and values to be updated. The SendTemplateBean is defined above.
Output	Success message (string)	
Notes	Updates a send template definition with data from the given bean. Fields which are passed as NULL will be ignored – their values will remain the same.	

updateSendTemplateContents

Input	arg1	The ID of the send template to update (int)
	arg2	The text content of the send template, or blank if there is none (string)
	arg3	The HTML content of the send template, or blank if there is none (string)
Output	Success message (string)	
Notes	Updates the contents of the send template with the supplied values. Both the text and the HTML parts are replaced/updated when this method is called.	

The **SendTemplateBean** data structure is used in several SOAP methods above. It is composed of the following fields:	
Title	REQUIRED: The title of the send template (string)
externalID	An external identifier for the send template (string)
Subject	REQUIRED: Message subject (string)
fromEmail	REQUIRED: From e-mail address (string)
fromName	From name (string)
toEmail	REQUIRED: To e-mail address (string)
toName	To name (string)
replyToEmail	REQUIRED: Reply-to e-mail address (string)
replyToName	Reply-to name (string)

encoding	REQUIRED: The encoding to be used for the message. Use “quoted-printable” as a default. (string)
Charset	REQUIRED: The charset to be used for the message. Use “ISO-8859-1” as a default. (string)
blockDomains	Domains which will be blocked (array of strings)
trackType	REQUIRED: Type of link tracking to be used. Must be “NONE”, “ALL”, “HTML” or “USER”. (string)
openTrackType	REQUIRED: Type of open tracking to be used. Must be “NONE” or “HTML”. (string)
clickStreamType	REQUIRED: Type of click-stream tracking to be used. Must be “NONE” or “ALL”. (string)
purgeLists	IDs of lists which should be purged against (array of int)
purgeSuppressionLists	IDs of suppression lists which should be purged against (array of int)
campaignID	REQUIRED: The Campaign ID. Must be a valid Campaign for the Site. (int)
brandID	REQUIRED: The Brand ID. Must be a valid Brand for the Site. (int)
replyContentID	The ID for reply content. ID of a valid reply content record or zero. (int)
unsubContentID	The ID for unsubscribe content. ID of a valid unsubscribe content record or zero. (int)
footerContentID	The ID for footer content. ID of a valid footer content record or zero. (int)
headerContentID	The ID for header content. ID of a valid header content record or zero. (int)
forwardFriendContentID	The ID for forward-to-friend content. ID of a valid forward content record or zero. (int)
advertiserName	The advertiser supplying the suppression list (string)
unsubReportsAddress	Address to which to send unsubscribes (string)
unsubReportsSize	Maximum report size for unsubscribes (int)
enabled	Is the send template enabled? (Boolean)

TRIGGERED MESSAGES

sendMessageFromTemplate

Input	arg1	ID of the send template that should be sent to the recipient (int)
	arg2	The email address of the intended recipient (String)
	arg3	An array of name-value pairs to be used as substitution tag values (Array of Strings, each String in the format "name=value")
Output	'Ok' (string)	
Notes	This method is a highly performant mechanism for sending a message to a single recipient using the send template and the substitution tag values provided. This differs from *doImportAndSendFromTemplate()* in two important aspects: it can only be used to send to one member at a time, and no data is imported into the member record. If the email address does not exist, it will be created, but all demographics fields associated with the new member will be left blank. The Send Template must be created in advance in the StormPost interface under the Settings tab. This method will throw an error if the Send Template is invalid or unavailable. Returns either a "OK" message or a SOAP fault error.	

sendMessageFromTemplatewithReferenceNumber

Input	arg1	ID of the send template that should be sent to the recipient (int)
	arg2	The external ID of the intended recipient (String)
	arg3	An array of name-value pairs to be used as substitution tag values (Array of Strings, each String in the format "name=value")
Output	Ok' (string)	
Notes	This method is a highly performant mechanism for sending a message to a single recipient using the send template and the substitution tag values provided. This differs from *doImportAndSendFromTemplate()* in two important aspects: it can only be used to send to one member at a time, and no data is imported into the member record. **Please note that if the records does not exist, the call will throw an exception (as it will NOT create the recipient)**	

	The Send Template must be created in advance in the StormPost interface under the Settings tab. This method will throw an error if the Send Template is invalid or unavailable. Returns either a “OK” message or a SOAP fault error.

sendMessageFromTemplatewithReferenceNumberAndAddress

Input	arg1	ID of the send template that should be sent to the recipient (int)
	arg2	The external ID of the intended recipient (String)
	arg3	The email address of the intended recipient (String)
	arg4	An array of name-value pairs to be used as substitution tag values (Array of Strings, each String in the format “name=value”)
Output	‘OK’ (string)	
Notes	This method is a highly performant mechanism for sending a message to a single recipient using the send template and the substitution tag values provided. This differs from *doImportAndSendFromTemplate()* in two important aspects: it can only be used to send to one member at a time, and no data is imported into the member record. If the external ID does not exist, it will be created, but all demographics fields associated with the new member will be left blank. If the external ID exists and the email address does not match the current one listed, an exception will be thrown. The Send Template must be created in advance in the StormPost interface under the Settings tab. This method will throw an error if the Send Template is invalid or unavailable. Returns either a “OK” message or a SOAP fault error.	

STORMPOST AS A PASS THROUGH MESSAGES

Please use the following SOAP methods to define both the target audience and content that StormPost will need to deliver and process as in-bound results.

The following two methods are new to StormPost 5.5 and use a new object called the **RecipientData Object**.

sendMessagesFromTemplate

Input	arg1	ID of the send template that should be sent to the recipient (int)
	arg2	RecipientData Object
Output	Mailing ID	
Notes	The sendMessagesFromTemplate call does essentially the same thing as the sendMessageFromTemplate call, but it sends message to multiple recipients. This call should be used when sending content to more than 1000 recipients. The general use case would be to create a Send Template using the createSendTemplate method and then call the sendMessagesFromTemplate call multiple times using the Send Template that was created. The length of the recipientData array is limited to 1000. A new Mailing is created every day for the Send Template just as it is for the sendMessageFromTemplate call. The method returns the mailing ID of the mailing used to send the messages.	

sendEmailMessages

Input	arg1	ID of the send template that should be sent to the recipient (int)
	arg2	Mailing Title (optional) (string)
	arg3	EmailContent Object
	arg4	RecipientData Object
Output	Mailing ID	
Notes	The sendEmailMessages method is designed to be used when a user wants to send some email content to fewer than 1000 recipients. Each time the method is called, a new mailing is created based upon the given Send Template. Most mailing parameters are derived from the Send Template, but the content for the mailing can be overridden from the Email Content object supplied with the call. If any element of the Email Content object is blank,	

	the content element from the Send Template will be used. Recipient records are created for each address in the Recipient Data that does not already exist in the Recips table. A Mailing Recip record is create for each row of recipient data. The mailing title parameter is optional. If it is specified, the mailing will have this title; otherwise, the title will be "<send template title> for <today's date>"

Object: Recipient Data

The new *StormPost as a Pass Through* API, introduced in StormPost 5.5, uses the **Recipient** data structure.	
address	The email address of the intended recipient (String)
External ID	The external ID of the intended recipient (String)
name-value pairs	An array of name-value pairs to be used as substitution tag values (Array of Strings, each String in the format “name=value”)
Notes	Depending on DB email or External ID is required

MAILINGS (SENDING MESSAGES TO EXISTING LISTS)

Please refer to the following SOAP methods in order to mimic the StormPost UI send process and target existing lists with new content.

createEmailMailing

Input	arg1	Mailing Object
	arg2	EmailContent Object
Output	Mailing ID	
Notes	Creates a new mailing. Once created the mailing appears in the “unsent mailing” list until enabled using enableMailing.	

sendTestMessage

Input	arg1	ID of the mailing located in Unsent Mailings (int)
	arg2	The email address of the intended recipient. Multiple emails can be separated by commas. (String)
	arg3	A string of either 'TEXT' or 'HTML.' Leaving this empty will send the HTML version of the message if both options are available. (String)
Output	There is no output or return value (unless error is thrown)	
Notes	This method will send a test message with an [SP] in front of the subject line for quick identification. A test message can be sent from any unsent mailing on the StormPost site. . Returns either a "success" message or a SOAP fault error.	

updateEmailMailing

Input	arg1	ID of the mailing to be updated (integer)
	arg2	Mailing data structure
	arg3	EmailContent data structure
Output	There is no output or return value (unless error is thrown)	
Notes	Updates an existing mailing.	

enableMailing

Input	arg1	ID of the mailing to be enabled (integer)
Output	There is no output or return value (unless error is thrown)	
Notes	Enables and existing mailing. Once the send time comes, the mailing will be queued and sent.	

getMailingStatus

Input	arg1	The ID of the mailing (int)
Output	Mailing status (string)	
Notes	Returns the text status of the specified mailing. Possible return values ("RECURRING", "PENDING", "RETRY", "RETRY_PENDING", "DONE")	

Object: Mailing

The **Mailing** object consists of:	
title	**(REQUIRED)** The title of the mailing (string)
externalID	An external identifier for the mailing (string)
comment	A free text field for notes about the mailing (string)
protocol	Defines the type of mailing. Acceptable values are "Email" or "Facebook" (string)
campaignID	**(REQUIRED)** The ID of the campaign that this mailing should be added to (integer)
brandID	**(REQUIRED)** The ID of the brand that this mailing should use (integer)
listID	The ID of the list that the mailing should be sent to (string)
additionalLists	Array of ListIDs used for sending to multiple lists (array of integers)
conditionOperator	What type of conditions are being passed in for targeting. Acceptable values are "AND" or "OR" (string)
condition1Column	Column name for the first condition (string)
condition1Value	Value for the first condition (string)
Condition1Operator	Operator for the first condition ("=", "<", ">", "!=", "contains", "not_contain", "starts", "ends", "is_blank", "not_blank", "is_null", "not_null") (string)
condition2Column	Column name for the second condition (string)
condition2Operator	Value for the second condition (string)

condition2Value	Operator for the second condition (“=”, ”<”, ”>”, ”!=”, ”contains”, “not_contain”, “starts”, “ends”, “is_blank”, “not_blank”, “is_null”, “not_null”) (string)
condition3Column	Column name for the third condition (string)
condition3Operator	Value for the third condition (string)
condition3Value	Operator for the third condition (“=”, ”<”, ”>”, ”!=”, ”contains”, “not_contain”, “starts”, “ends”, “is_blank”, “not_blank”, “is_null”, “not_null”) (string)
condition4Column	Column name for the fourth condition (string)
condition4Operator	Value for the fourth condition (string)
condition4Value	Operator for the fourth condition (“=”, ”<”, ”>”, ”!=”, ”contains”, “not_contain”, “starts”, “ends”, “is_blank”, “not_blank”, “is_null”, “not_null”) (string)
condition5Column	Column name for the fifth condition (string)
condition5Operator	Value for the fifth condition (string)
condition5Value	Operator for the fifth condition (“=”, ”<”, ”>”, ”!=”, ”contains”, “not_contain”, “starts”, “ends”, “is_blank”, “not_blank”, “is_null”, “not_null”) (string)
maxRecipients	Maximum number of recipients who should receive this mailing (integer)
seeds	Array of email addresses to be seeded on this mailing (array of strings)
blockDomains	Array of domains that should be excluded from this mailing (array of strings)
purgeLists	Array of ListIDs to purge against (array of integers)
purgeSuppressionLists	Array of ListIDs to suppress against (array of integers)
queueTime	What time should this mailing be sent (datetime)
trackType	Link tracking type. (“NONE”, “ALL”, “HTML”, “USER”) (string)
openTrackType	Open tracking type. (“NONE”, “HTML”) (string)
clickStreamType	Conversion tracking type. (“NONE”, “ALL”) (string)

advertiserName	Name of the advertiser responsible for the message (string)
unsubReportsAddress	If using UnsubCentral, what address should the reports be mailed to (string)
unsubReportsSize	Maximum file size for UnsubCentral reports in kb (“0” No limit, “5120”, “2048”, “1024”, “500”, “100”, “20”, “10”) (integer)
Priority	HIGH, NORMAL, LOW

Object: EmailContent

The **EmailContent** object consists of:	
subject	**(REQUIRED)** The subject line of the email (string)
fromEmail	**(REQUIRED)** The from address for the email (string)
fromName	The friendly from name for the email (string)
toEmail	**(REQUIRED)** The destination email address for the email (string)
toName	The friendly to name for the email (string)
replyToEmail	The reply-to address for the email (string)
replyToName	The friendly reply-to address for the email (string)
charset	The character set for this mailing. English/Latin “ISO-8859-1” is the default, other acceptable values are (“ISO-8859-8” ,”WINDOWS-1255”, “UTF-8”, “SHIFT_JIS”, “ISO-2022-JP”, “GB2312”, “Big5”, “ISO-8859-7”, “ISO-8859-9”, “EUC-KR”, “ISO-8859-2”, “TIS-620”, “ISO-8859-15”, “WINDOWS-1251”) (string)
encoding	The encoding type for the message. The default value is “quoted-printable”. Other acceptable values are (“8bit”, “7bit”, “Mixed”) (string)
htmlContent	The HTML content for the mailing (string)
textContent	The text content for the mailing (string)
unsubContentID	The ID of the unsubscribe footer (integer)
replyContentID	The ID of the reply-to content (integer)

headerContentID	The ID of the header content (integer)
footerContentID	The ID of the footer content (integer)
forwardToFriendContentID	The ID of the forward to a friend content (integer)

REPORTS

getDetailedMailingReport

Input	arg1	The ID of the mailing (int)
Output		Returns a single DetailedMailingReport
Notes	This is a SOAP method to retrieve the information available in a detailed mailing report in StormPost.	

getMailingReportSummaries

Input	arg1	The ID of the Site for which to get mailing summaries (int)
	arg2	From-Date for filter (Date)
	arg3	To-Date for filter (Date)
	arg4	Report type for filter (must be “Mailings”, “Triggered Mailings”, or “Campaigns”) (String)
	arg5	ID of a List to filter by (use 0 for all lists) (int)
	arg6	Maximum number of results to return (int)
	arg7	String to search for in the title – to filter results (String)
Output	An array of **SummaryMailingReport** or an array of SummaryCampaignReport.	
Notes	Retrieves a list of mailing report summaries or campaign report summaries that match the given query parameters. This is similar to the mailing summaries which are available on the main Reports tab in the web interface.	

Object: DetailedMailingReport

A **DetailedMailingReport** consists of:	
additionalLists	The IDs of any additional lists used by the mailing (array of string)
aggregateDeliveryCounts	Does this report use aggregate delivery counts? (boolean)
Archived	Has the mailing been archived? (boolean)
brandTitle	The name of the brand for the mailing (string)
blockDomains	Domains which were blocked (array of string)
blockedCount	Number of blocked messages (int)
blockedPercent	Percentage of messages which were blocked (double)
campaignTitle	The campaign title (string)
campaignReferenceNumber	The identifier of the campaign in the customers' database (string)
clickRate	Percent of messages which had clicks (double)
clickStreamType	The type of clickstream tracking to use (string)
countNumRecips	Whether the mailing is an aggregate mailing. (boolean)
complaintResponsesCount	Number of complaint responses (int)
complaintResponsesPercent	Percentage of complaint responses (double)
confirmCount	The number of recipients who confirmed. (int)
confirmRate	The percentage of recipients who confirmed (double)
Contents	Which content parts the mailing had (may contain TEXT and/or HTML) (string)
countNumRecips	Was the count estimated recipients feature used prior to sending (boolean)
createdDate	Date this mailing was created (date)
customerReferenceNumber	Identifier for the mailing in the customer's database (string)
deliveredCount	Number of messages delivered (int)
deliveredPercent	Percentage of messages delivered (double)

failedCount	Number of message which failed (int)
failedPercent	Percentage of messages which failed (double)
hardCount	Number of messages which had a hard bounce (int)
hardPercent	Percentage of messages which had a hard bounce (double)
linkReport	An Array of LinkReport (detailed below)
listTitle	The name of the list to which the mailing was sent (string)
mailingID	The ID of the mailing (int)
mailingParts	Array of MailingPartReport
mailingStatus	The status of the mailing (string)
mailingSubject	The mailing subject (string)
mailingTitle	The mailing title (string)
maxRecips	The maximum number of recipients allowed for the mailing (int)
openLinkCount	The number of open links in the mailing (int)
openRate	Percent of messages which were opened (double)
purgeLists	List of identifiers of lists to purge against (array of string)
purgeMailings	List of identifiers of mailings to purge against (array of string)
queueComment	A comment added by StormPost during queuing (string)
queueTime	Time at which the mailing was scheduled to be queued (string)
retriesRemaining	The number of retries remaining (int)
replyResponsesCount	The number of replies (int)
replyResponsesPercent	The percentage of recipients who replied (double)
scheduledCount	Number of messages scheduled (int)
scompResponsesCount	Number of scomp responses (int)
scompResponsesPercent	Percentage of scomp responses (double)
sendStartTime	The time when the mailing started sending (date)

sendFinishTime	The time when the mailing finished sending (date)
sentTime	Date this mailing was sent (string)
sendTemplateID	The identifier of the send template used to create this mailing (int)
softCount	Number of message which had a soft bounce (int)
softPercent	Percentage of messages which had a soft bounce (double)
suppressionLists	A list of suppression lists that the mailing is purge against (array of strings)
totalBouncePercent	Total percentage of messages which bounced (double)
totalBounceCount	Total number of messages which bounced (int)
totalClicksCount	The total number of clicks for the mailing (int)
totalClicksPercent	The number of clicks divided by the number of opens times 100 (double)
totalQueueTime	The number of seconds in took to queue the mailing (int)
totalResponsesCount	Total number of responses (int)
totalResponsesPercent	Total percentage of responses (double)
trackedLinkCount	The number of tracked links in the mailing content (int)
undeliveredCount	Number of messages which were undelivered (int)
undeliveredPercent	Percentage of messages undelivered (double)
uniqueClicksCount	The number of unique clicks for the mailing (int)
uniqueOpensCount	The number of unique opens (int)
uniqueUsersCount	The number of unique users (int)
uniqueUsersPercent	The percentage of unique users (double)
unsubscribeResponsesCount	Number of unsubscribe responses (int)
unsubscribeResponsesPercent	Percentage of unsubscribe responses (double)

A **LinkReport** consists of:	
linkID	The ID of the tracked link (int)
url	The URL of the tracked link (string)
part	Which part (TEXT,HTML) the tracked link is in (string)
totalClicks	Total clicks for this link (int)
uniqueClicks	Unique clicks for this link (int)
successRate	Percentage of recipients who clicked on the link (double)
totalCSClicks	Total clickstream clicks for this link (int)
totalCSAmount	Total dollar amount for clickstream clicks (int)

A **MailingPartReport** consists of:	
part	Which parts had links tracked
numTrackedUrls	Number of links tracked in this mailing.
totalClicks	Total clicks recorded in response to this message.
uniqueClicks	Number of unique clicks recorded on the links.
uniqueUsers	Number of users who clicked on at least one link.
totalOpens	Number of message opens recorded.

A **SummaryMailingReport** consists of:	
mailingID	Mailing ID (int)
mailingTitle	Title of the mailing (string)
campaignTitle	Title of the campaign (string)
brandTitle	Title of the brand (string)
listTitle	Name of the list to which the mailing was sent (string)
parts	The mailing parts (TEXT,HTML) (array of string)

status	Mailing status (string)
scheduledTime	The time at which the mailing was scheduled to be queue (string)
sentTime	Date and time the mailing was sent (date)
queueComment	A comment added by StormPost during queuing (string)
scheduledCount	Number of messages scheduled (int)
deliveredCount	Number of messages delivered (int)
deliveredRate	Percentage of messages delivered (double)
opensCount	Number of messages opened (int)
openRate	Percentage of messages opened (double)
clicksCount	Number of clicks (int)
clickRate	Percentage of opened messages that were clicked (double)
unsubsCount	Number of unsubscribes (int)
unsubRate	Percentage of recipients who unsubscribed (double)
optoutsCount	Number of optouts (int)
optoutRate	Percentage of recipients who opted out (double)
customerReferenceNumber	Identifier of the mailing in the customer’s database (string)
parentID	ID of the parent mailing (int)
ABSplit	NONE, PARENT or CHILD (string)
creator	The user who created the mailing (string)
domain	The domain associated with the mailing (string)
subject	The subject of the mailing (string)
fromName	Name in the mailing from (string)
campaignComment	A comment associated with the mailing’s campaign (string)
bounceCount	The number of bounces for the mailing (int)
blockBounceCount	The number of block bounces for the mailing (int)

A **SummaryCampaignReport** consists of:	
campaignID	The ID of the campaign (int)
campaignTitle	The title of the campaign (string)
createTime	The time when the campaign was created (date)
externalID	The ID of the campaign in the customer's database (string)
scheduledCount	The number of messages scheduled for delivery (int)
deliveredCount	The number of messages delivered (int)
deliveredRate	The percentage of message delivered (double)
opensCount	The number of opens (int)
openRate	The percentage of messages that were opened (double)
clicksCount	The number of clicks (int)
clickRate	The percentage of opened messages that were opened (double)
unsubsCount	The number of recipients who unsubscribed (int)
unsubRate	The percentage of recipients who unsubscribed (double)
optoutsCount	The number of recipients who opted out (int)
optoutRate	The percentage of recipients who opted out (double)
scompsCount	The number of scomps received (int)
softBouncedCount	The number of soft bounces received (int)
hardBouncedCount	The number of hard bounces received (int)
blockBouncedCount	The number of block bounces received (int)
uniqueClicksCount	The number of unique clicks received (int)
uniqueUsersCount	The number of unique recipients who clicked on tracked links (int)

HISTORY OF SOAP API CHANGES

STORMPOST VERSION	CHANGES INTRODUCED
5.5	Added new methods: • sendMessagesFromTemplate • sendEmailMessaegs Updated Existing methods* • createEmailMailing • createFacebookMailing • updateEmailMailing • updateFacebookMailnig *The update was actually made to the 'Mailing' object which now includes and 'additional lists' parameter to target multiple lists in StormPost. These four calls use this object as a parameter.
5.0	Added new methods: • createRecipient • updateRecipient • getRecipientByAddress • getRecipientByExternalID • getRecipientSubscriptions • subscribeToList • unsubscribeFromList • globalUnsubscribe • globalUnsubscribeAll • getList • getLists • sendMessageFromTemplatewithReferenceNumber

STORMPOST VERSION	CHANGES INTRODUCED
	• sendMessageFromTemplatewithReferenceNumberAndAddress Added new objects to support methods: • Recipient • List • Modified ListSubscription Removed Methods • queueMailing • queueClickstreamMailing Deprecated methods *(to be removed by StormPost 6)* • getRecipSubscriptionInfo o Deprecated associated object: SubscriptionInfo o Deprecated associated object: RecipField o Modified associated object: ListSubscription
4.0	Added new methods • getImportStatus • getMailingStatus • createEmailMailing • updateEmailMailing • enableEmailMailing Added data structures for EmailMailing APIs • Mailing data structure • EmailContent data structure

STORMPOST VERSION	CHANGES INTRODUCED
2.5.3	Added new methods: • getRecipSubscriptionInfo • queueClickStreamMailing queueMailing ▪ Add new input parameters getDetailedMailingReport ▪ Added new output parameters getMailingReportSummaries ▪ Add new output parameters
2.5	Added new methods: • getMailingReportSummaries() • getDetailedMailingReport() • createSendTemplate() • getSendTemplateContents() • getSendTemplateDefinition() • updateSendTemplateContents() • updateSendTemplateDefinition() Changed return type of: 1. getImportTemplates() 2. getSendTemplates() to return an IdAndTitle bean instead of just the ID.
2.4.9	Added Brand support for queueMailing() -- additional BrandID input parameter
2.4.5	Added new methods: • queueMailing() • sendMessageFromTemplate() • doImportFromTemplate()

STORMPOST VERSION	CHANGES INTRODUCED	
	New name	Old name
	getAvailableImportTemplateIds()	getImportTemplates()
	getAvailableSendTemplateIds()	getSendTemplates()
	doRealtimeImport()	doImportAndSendFromTemplate()
	doRealtimeImportOnly()	doImportFromTemplate()
	sendMailing()	queueMailing()
	sendDynamicMailing()	sendMessageFromTemplate()
2.4.2	StormPost API first created. Available methods: • getVersion() • getAvailableImportTemplateIds() • getAvailableSendTemplateIds() • doRealtimeImport()	

USE CASES

📝 The following section provides guidance on which SOAP methods to use based on some high-level use cases. Please consult with your StormPost Integration consultant and discuss your specific needs before implementing these calls.

WHAT METHODS SHOULD I USE TO IMPORT DATA?

Clients should work with their StormPost Integration consultant to select the right set of SOAP methods in order to properly import new member information, as well as profile updates, into their StormPost database. The selected SOAP methods may affect system performance, scalability and the speed of imports.

Please consider the following questions when choosing which methods to use when importing data into your StormPost database.

1. Is StormPost the system of record for your subscriptions?

If the answer to this question is 'Yes' then you will want to use the Subscription Management calls (see Subscription Management below).

If your answer is 'No' you should consider questions 2 and 3.

2. Does the profile or demographic data need to be in the StormPost database immediately?

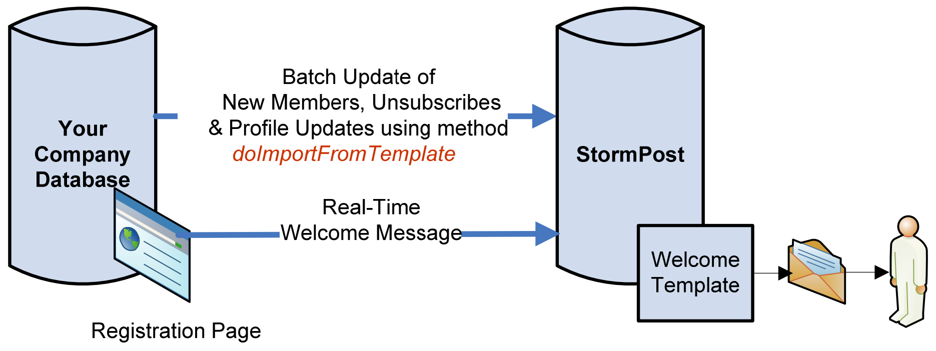
3. Do you need import history and reporting for new member and profile updates?

Clients should note that StormPost has a separate SOAP method that can be leveraged to send real-time messaging, regardless of the selected import method. Clients may also pass in personalization elements in the real-time messages- without actually importing data into StormPost.

So do you need the profile information (i.e. First Name, Address, Gender, etc.) in the StormPost database immediately? This likely depends on how quickly you plan to send personalized and targeted messaging (following the welcome message) to users in StormPost.

BATCH IMPORTS

Clients that answer 'No' to question 2, and 'Yes' to question 3 will likely use is the SOAP method **doImportFromTemplate**.' This SOAP method requires an import template and batches in nightly or hourly intervals.. The import template provides clients with the ability to view reporting on any import associated with the template.


Your Company Database
Batch Update of New Members, Unsubscribes & Profile Updates using method *doImportFromTemplate*
StormPost
Real-Time Welcome Message
Welcome Template
Registration Page

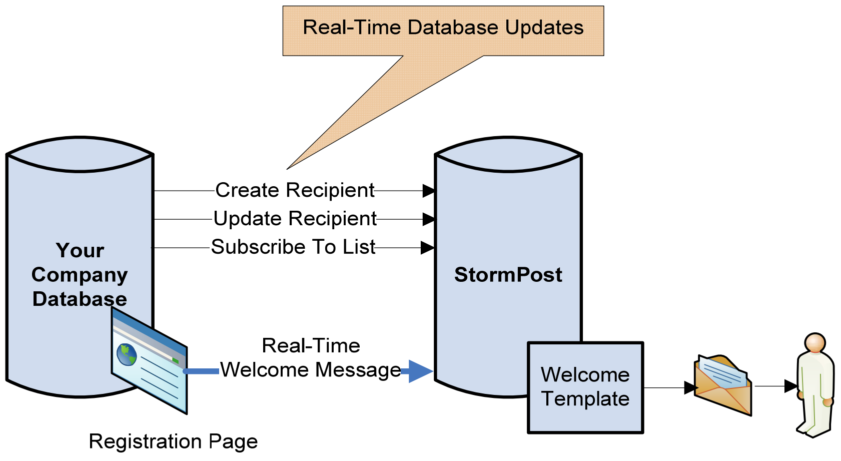
SUBSCRIPTION MANAGEMENT (REAL-TIME UPDATES)

Clients that answer 'Yes' to Question 1, or answered Yes/No to question 2 and 3, will use the SOAP methods under Subscription Management.

When collecting real-time subscriptions from a registration or preference management center, use the following Subscription Management calls

- Create Recipient
- Update Recipient
- Subscribe to List

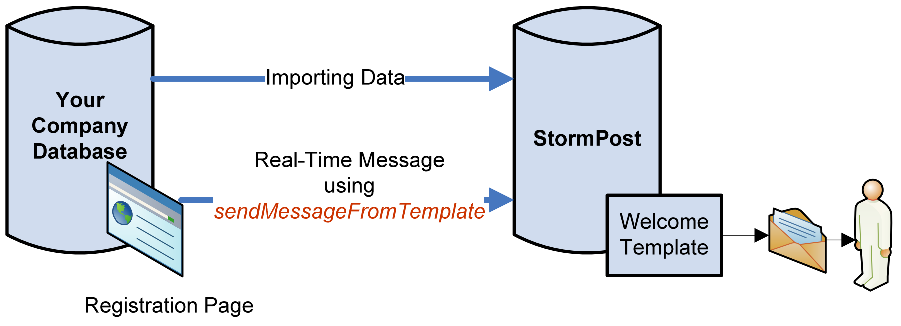
These methods allow clients to make real-time calls to the StormPost database to create the record, update the profile information, and update the list subscriptions. Although three separate calls are required to accomplish each update, they do not kick-off import jobs and therefore, do not require system resources.



WHAT METHODS SHOULD I USE TO SEND MESSAGES?

TRIGGERED MESSAGES

Clients sending **real-time messages**, such as welcome messages, order confirmations, password reminders should use the **sendMessageFromTemplate** method. This method is considered a 'Fast SOAP' call because it does not import any information other than the member's email address. Clients may use personalization in the message by passing in name-value pairs with the SOAP call.

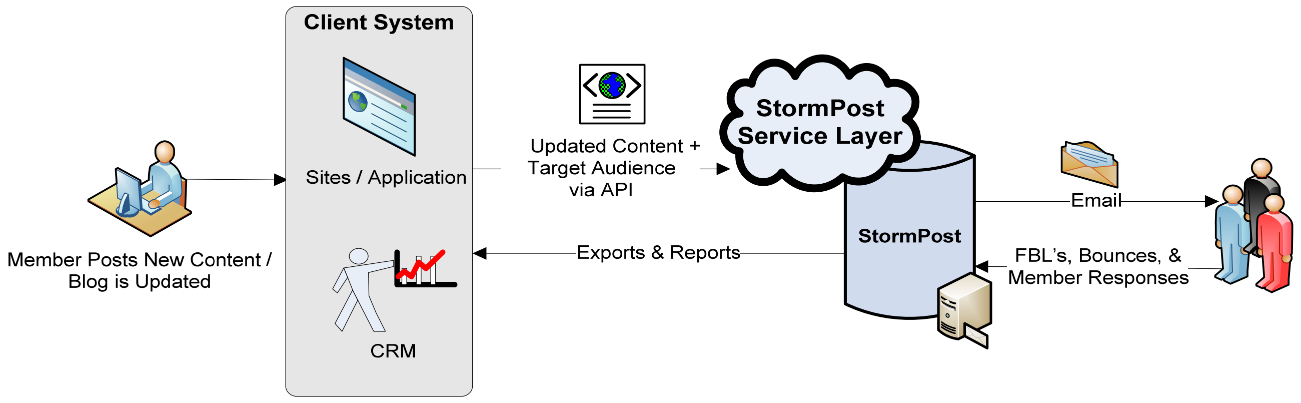


PASS-THROUGH MESSAGES

Clients sending **messages to new target audiences** each time should take advantage of **StormPost as A Pass Through**, using SOAP method, **sendMessagesFromTemplate**.

A few examples include

- A company hosting a blog needs a way to message followers anytime the blog is updated. The number of followers could range from a few people to thousands of people.
- A gaming / social media needs a way to message users of a particular application or game each time users reached a new status or level. The target audience for this type of messaging could range from a few users to tens of thousands of users.


Client System
Sites / Application
CRM
Member Posts New Content / Blog is Updated
Updated Content + Target Audience via API
StormPost Service Layer
StormPost
Exports & Reports
Email
FBL's, Bounces, & Member Responses

MAILINGS

When sending a **message to a target audience that already exists in StormPost**, such as a one-time promotional message or a one-time newsletter, clients should use the methods:

- createEmailMailing
- enableMailing

This SOAP call mimics many of the workflow options that clients see when scheduling a mailing through the StormPost User Interface. Clients can entirely bypass using the StormPost interface by targeting and segmenting existing StormPost lists via the API.