pitney bowes

Outbound & Inbound Package Distribution

SendSuite® Link

Programmer's Guide

Admin Guide

US English Edition
SV63032 Rev. H
July 16, 2021

The use of this information by the recipient or others for purposes other than the training of customers on Pitney Bowes equipment may constitute an infringement of intellectual property rights of Pitney Bowes, and Pitney Bowes assumes no responsibility for any such use of the information.

We have made every reasonable effort to ensure the accuracy and usefulness of this manual. However, we cannot assume responsibility for errors or omissions or liability for the misuse or misapplication of our products.

Except as provided in writing, duly signed by an officer of Pitney Bowes, no license either express or implied, under any Pitney Bowes or third party's patent, copyright or other intellectual property rights is granted by providing this information.

Table of Contents

Overview

SendSuite Link offers Clients and Pitney Bowes Service Professionals the flexibility to ‘automate’ the communication with SendSuite Tracking to control the flow of data, ensure data integrity, and eliminate data entry in systems across your organization. It provides real-time data integration, versus traditional batch file imports and manually adding and/or updating various data elements within SendSuite Tracking.

SendSuite Link is available as an optional ‘toolkit’ installed as an add-on to SendSuite Tracking. It is based on Representational State Transfer (REST) style communications and uses secure HTTP request and response messages in the form of XML or JSON. It exposes REST-based web services for querying, adding, updating, or deleting data from SendSuite Tracking. These web services can be used from an external software application or system or also from within SendSuite Tracking using Powerlogic .NET. SendSuite Link’s service-oriented architecture (SOA) gives customers the ability to integrate their own applications with SendSuite Tracking through well-defined interfaces. The API’s are designed to be used as an integration tool to automate business processes within SendSuite Tracking. The API’s do not call out to other systems and they do not read or write data to external data sources. There are no end-user interactions or UI’s because it is intended to be used by developers or systems designed to access XML or JSON data sources.

All communication to SendSuite Link in the PBI hosting facility will use SSL. If SendSuite Link is installed on-premise, the customer can decide whether or not to use SSL. Authentication to SendSuite Link is granted using API keys which are generated in the SendSuite Administrator. You can generate multiple keys and assign them to different systems or servers to help manage identity. The keys can be disabled or deleted in the SendSuite Administrator to deny access to granted keys. There is no authorization configured for the keys. A valid key can access all API’s on SendSuite Link.

Examples of services that can be accessed via SendSuite Link include: • Employee information, as well as locations (sites, buildings, departments, mailstops, and maildrops). • Carriers & Services, Carrier Tracking (Expected mail), and Carrier barcode decoding. • Assets Management. • Mail and Package updates. • Alerts and Events. • SendSuite Tracking Reports. • Email notification

This document is a guide to getting started with writing code using SendSuite Link and SendSuite Tracking and is presented as a supplement to the existing documentation. It is assumed that the reader has an established background with structured and object-oriented programming and is familiar with software development tools such as Microsoft Visual Studio and the Microsoft .NET Framework.

Requirements

You must install the version of SendSuite Link that was issued with SendSuite Tracking. You do not need to write your integration code on the server itself. It can be done remotely on a workstation that is configured with the appropriate development tools and software and then deployed as needed.

If you are going to write code against SendSuite Link using the .NET platform the following is recommended as a toolset:

- Visual Studio 2010 or later
- Microsoft .NET Framework 4.0 or later

If you are going to use another platform to write code against SendSuite Link choose something that allows you to invoke REST-based web services. Ideally you should select a software framework that offers solid capabilities for handling XML, JSON, and HTTP requests and responses. SendSuite Link can be used with many programming languages such as the following:

- C#
- VB.NET
- Java
- Perl
- Ruby
- JavaScript

Installation

Before you begin you must have SendSuite Tracking and SendSuite Link installed on a server or on a workstation for development purposes.

1. Install the main SendSuite Tracking software.
2. Install SendSuite Link on the SendSuite Tracking server.
3. Install Microsoft Visual Studio or other integrated development environment (IDE) of choice along with the corresponding software development kits (SDKs).

Once SendSuite Link is installed API documentation and database schema reference can be found at the following URL:

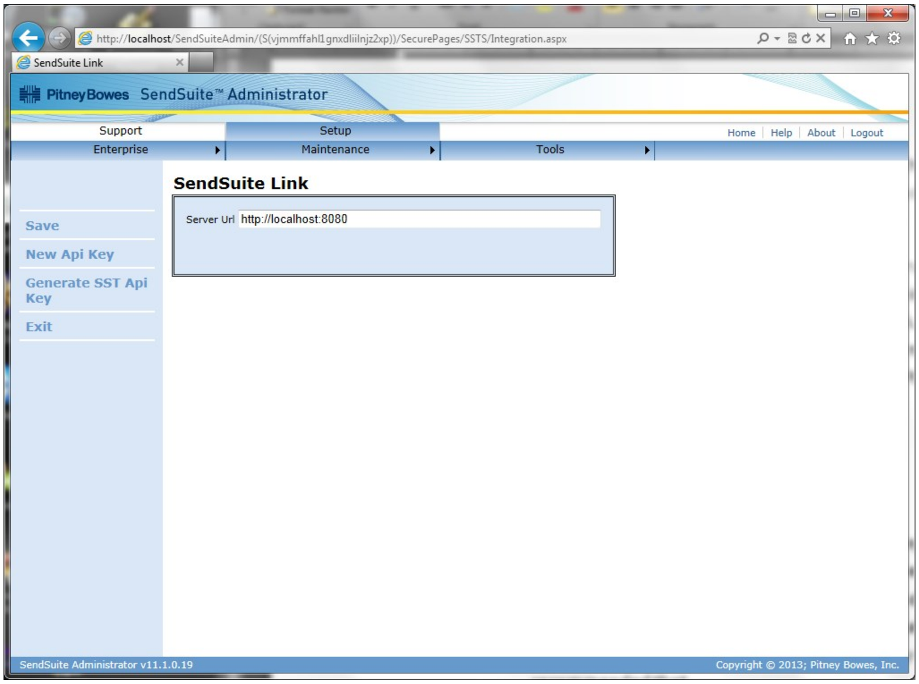
http://{MyServerURL}:8080/link

Configuration

Once installed SendSuite Link is configured and enabled using the browser-based SendSuite Administrator module within SendSuite Tracking. You may need to have your administrator do this for you if you do not have direct access to the server.

1. Using your web browser open SendSuite Administrator at http://{MyServerURL}/SendSuiteAdmin using the “admin” user ID and password.
2. From the Setup menu, select Enterprise, SendSuite Link.

3. You will need to generate an API key. An API key is a unique identifier that must be provided as part of each SendSuite Link request as an authentication mechanism. The API keys are unique to each installation of SendSuite Tracking and SendSuite Link.


http://localhost/SendSuiteAdmin/(S(vjmmffahl1gnxdliilnjz2xp))/SecurePages/SSTS/Integration.aspx
SendSuite Link
PitneyBowes SendSuite™ Administrator
Support
Setup
Home
Help
About
Logout
Enterprise
Maintenance
Tools
Save
New Api Key
Generate SST Api Key
Exit
SendSuite Administrator v11.1.0.19
Copyright © 2013; Pitney Bowes, Inc.

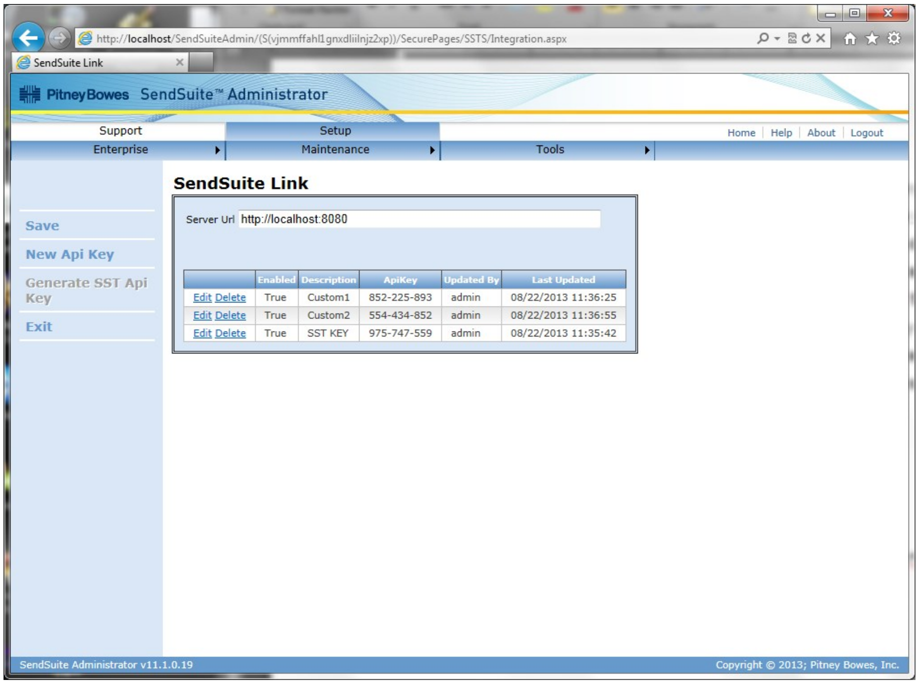
SendSuite Link

Server Url http://localhost:8080

   a. The Generate SST Api Key option creates a key that internally enables functions within Powerlogic .NET that use SendSuite Link. These functions are disabled until this key is generated. Generating this key allows you to add custom code to SendSuite Tracking screens using the SendSuite Designer application that access or manipulate other application data. Only one SST API key may be generated within the system.

   b. The New Api Key option creates a key for any other use of SendSuite Link. It is recommended that this type of key be generated and used for external applications requiring the use of SendSuite Link web services. More than one API key can be generated within the system.

The following screen shows SendSuite Link configured with several API keys.


http://localhost/SendSuiteAdmin/(S(vjmmffahl1gnxdliilnjz2xp))/SecurePages/SSTS/Integration.aspx
SendSuite Link
PitneyBowes SendSuite™ Administrator
Support
Setup
Enterprise
Maintenance
Tools
Home | Help | About | Logout
Save
New Api Key
Generate SST Api Key
Exit
SendSuite Administrator v11.1.0.19
Copyright © 2013; Pitney Bowes, Inc.

SendSuite Link

Server Url http://localhost:8080

	Enabled	Description	ApiKey	Updated By	Last Updated
Edit Delete	True	Custom1	852-225-893	admin	08/22/2013 11:36:25
Edit Delete	True	Custom2	554-434-852	admin	08/22/2013 11:36:55
Edit Delete	True	SST KEY	975-747-559	admin	08/22/2013 11:35:42

API Keys can be edited to change the description or to be enabled/disabled. The key itself cannot be altered once it has been generated. If an API key is no longer to be used it should be deleted.

Authentication

SendSuite Link can be configured for both **Basic Authentication** and **Token-Based Authentication**. This setting is changed in SendSuite Admin>Setup>General Settings. The checkbox is labeled “Enable Basic Authentication”.

Basic Authentication

When Basic Authentication is enabled, pass the credentials through the client. SendSuite Link will read the **AUTHENTICATION_HEADER** to validate. This is typically {user}+':'+{password} represented as a Base64 string. Consider the following example using the default SendSuite Tracking administrator user credentials:

User ID	Password	String	Base64
Admin	opu5d3i	admin:opu5d3i	YWRtaW46b3B1NWQzaQ==

The value in the **AUTHENTICATION_HEADER** would be "Basic YWRtaW46b3B1NWQzaQ==".

Token-Based Authentication

Token-Based authentication is enabled in SendSuite Admin in Setup>SendSuite Mobile>General Settings. The checkbox is labeled “Include Apikey and User Password in Query String”. Be sure to select the desired timeout period in the “SSTMobile Auto Logoff idle time” dropdown box. When implementing Token-Based authentication, include the APIKey and Password in the query string. Consider the following examples:

User & Password

http://localhost:8080/link/rest/admin/session/getsessionid?u=<username>&h=<hashedPwd>

API Key

http://localhost:8080/link/rest/receiver?apikey=<apikey>

If the aforementioned checkbox is unchecked, the hashed password and APIkey must be passed as Header.

When checked, a token must be acquired from SendSuite Link before subsequent requests may be made.

A token may be refreshed. Consider the following example:

Refresh Token

http://localhost:8080/link/rest/admin/session/refreshsessionid/<username>?apikey=<apikey>

Guidelines and Notes

The following are some guidelines and notes to keep in mind when using SendSuite Link.

- SendSuite Link is an API and provides minimal to no business logic. It is simply a tool that can be used to get data in and out of a SendSuite Tracking system.
- SendSuite Link is a stateless API where each call to the server is independent and atomic.
- Data payloads provided to an API call should be in lower case and not mixed case. This is important in the submitted URL and also for any XML tags. However, this does not apply to actual data or field content.
    - **Correct** – http://localhost:8080/link/rest/receiver?apikey=975-747-559
    - **Incorrect** – http://localhost:8080/link/rest/Receiver?apiKey=975-747-559
- When retrieving the count of some data that meets a certain criteria, the count will be returned as numeric only and not as XML content.
- When providing an XML payload as part of the HTTP request, be sure to follow the exact case of the data element tags. SendSuite Link will ignore XML tags it does not recognize.
- SendSuite Link supports both XML and JSON data formats. The default is XML which is also used for the purposes of this document.
- Care should be taken to properly encode the submitted URL to accommodate any punctuation or special characters.

Sample Code

The accompanying ZIP file included with this guide (**SendSuiteLinkSamples.zip**) contains Visual C# and VB.NET sample projects written using Visual Studio 2010. The code snippets shown in this guide are Visual C# examples. The sample code is usable as-is with only minor modifications required for your specific environment, such as the server URL and the API key.

Data Entity Objects

The following table shows the various data entity objects and corresponding summary information.

Entity Object Name	Description	URL syntax	HTTP methods	Primary Database Table
Alert	System Alerts	\alert\	GET, POST, DELETE	REQUEST
Asset	Asset	\asset\	GET, POST, DELETE	ASSETS
AssetEvent	Asset Event	\assetevent\	GET, POST, DELETE	T1AST
Building	Building	\building\	GET, POST, DELETE	BLDNGS
Carrier	Carrier	\carrier\	GET, POST, DELETE	CARRIERS
Condition	System Conditions	\condition\	GET, POST, DELETE	CONDTION
Configurationsettings	Configuration Settings	\configsetting\	GET, POST, DELETE	CONFIGURATIONSETTINGS
Decode	Carrier Decoding	\decode\	GET	
Decodefield				DECODEFIELDS
Decodeformat		\mobile/decodeformat\	GET, POST, DELETE	DECODEFORMATS
DeliveryCommitment	Delivery Commitment	\deliverycommitment\	GET, POST, DELETE	DLVCMT
Department	Department	\department\	GET, POST, DELETE	DEPART
Email	Email	\email\	POST	

Entity Object Name	Description	URL syntax	HTTP methods	Primary Database Table
Event	Package Event	\event\	GET, POST, DELETE	T1VEW
ExpectedMail	Expected Mail	\expectedmail\	GET, POST, DELETE	EXPECT
GeneralDetail	General Detail	\generaldetail\	GET, POST, DELETE	GENDETL
GeneralMaster	General Master	\generalmaster\	GET, POST, DELETE	GENMAST
Image	Image	\image\	GET, POST, DELETE	RCV_SIG
Log		\mobile/log\	GET, POST, DELETE	LOG
Mail	Packages or Mail	\mail\	GET, POST, DELETE	MAILS
Maildrop	Mail Drop	\maildrop\	GET, POST, DELETE	MAILDROP
Mailroute	Mail Route	\mailroute\	GET, POST, DELETE	MAILROUT
Mailrun	Mail Run	\mailrun\	GET	MAILRUN
MailRun_MailRoute	Mail Run/Mail Route	\mailrun_mailroute\	GET	MAILRUN_MAILROUTE
Mailstop	Mail Stop	\mailstop\	GET, POST, DELETE	MAILSTOP
Mobilelayout		\mobile/layout\	POST	MOBILELAYOUT
Mobilepopuplist		\mobilepopuplist\	GET, POST, DELETE	MOBILEPOPUPLIST

Entity Object Name	Description	URL syntax	HTTP methods	Primary Database Table
Mobileschemaversion		\mobileschemaversion\	GET, POST, DELETE	MOBILE_SCHEMA_VERSION
PackageContent	Package Contents	\packagecontent\	GET, POST, DELETE	RCVSWG
Product	Product	\product\	GET, POST, DELETE	RCVPROD
PurchaseOrder	Purchase Order	\purchaseorder\	GET, POST, DELETE	PODETL
PurchaseOrderContent	Purchase Order Content	\purchaseordercontent\	GET, POST, DELETE	POSWG
Receiver	Receiver or Recipient	\receiver\	GET, POST, DELETE	RECVRS
Report	Report	\report\	GET	SMT_RPT
RouteLog	Route Log	\routelog\	GET, POST	ROUTELOG
Sender	Sender	\sender\	GET, POST, DELETE	SENDERS
Service	Carrier Service	\service\	GET, POST, DELETE	SERVICES
Site	Site	\site\	GET, POST, DELETE	RCVSITES
T1Vst			T1VST	
User	User	\user\	GET, POST, DELETE	USERS
Visitor	Visitor	\visitor\	GET, POST, DELETE	VISITORS
Visits	Visits	\visit\	GET	VISITS

Querying Data

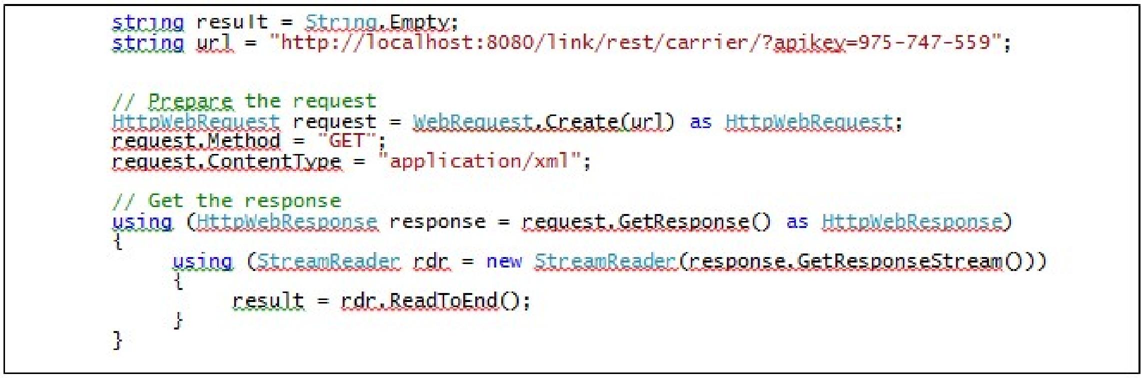
Queries within SendSuite Link are done via a HTTP GET method and make use of the following keywords in the URL string:

- **select** — A comma delimited list of fields to return (overrides the full parameter)
- **apikey** – Mandatory authentication data that must be provided in every request.
- **count** – Provides a count of the number of records that match the query.
- **full** – Provides all fields in the result set vs. the default summary set of fields.
- **skip** – The starting index to be used when paging results.
- **top** – The number of records to be returned when paging results.
- **orderby** – Specifies the field by which to order the results.
  - **asc** – Ascending order (default).
  - **desc** – Descending order.
- **filter** – Allows the filtering of result sets by specifying fields and search conditions using the following operators. This keyword behaves in the same fashion as the WHERE clause in SQL.
  - **eq** – Equal to.
  - **ne** – Not equal to.
  - **lt** – Less than.
  - **gt** – Greater than.
  - **ge** – Greater than or equal to.
  - **le** – Less than or equal to.
  - **and** – Combine two conditions and require that both conditions be true.
  - **or** – Combine two conditions.
  - **not** – Negation operator.
  - **like** – Determine whether a character string matches the provided pattern.
  - **between** – Specifies a range to test.
  - **in** – Determines whether a specified value matches any value in a subquery or a list.

The following query samples show the usage of the query keywords within SendSuite Link. For testing purposes the supplied URLs can be entered into the address bar of your web browser and the XML response will be displayed.

1. Count the number of records of a particular object type.
   http://localhost:8080/link/rest/mail/count/?apikey=975-747-559
2. Query for a list of all records using the object name.
   http://localhost:8080/link/rest/service/?apikey=975-747-559
3. Query for a list of records using the object name and a criteria.
   http://localhost:8080/link/rest/service?apikey=975-747-559&filter=carr eq 'FEX'
4. Query for a single record using the object name and primary key value.
   http://localhost:8080/link/rest/carrier/FEX?apikey=975-747-559
   This query can also be written as follows.
   http://localhost:8080/link/rest/carrier?apikey=975-747-559&filter=carr eq 'FEX'&full
5. Query for records matching values in a criteria list.
   http://localhost:8080/link/rest/service?apikey=975-747-559&filter=carr in ('FEX', 'UPS')
6. Query for records within a date range.
   http://localhost:8080/link/rest/mail?apikey=975-747-559&filter=rdate between '2013-01-01' and '2013-02-28'
7. Return specified columns only:
   http://localhost:8080/link/rest/carrier?apikey=975-747-559&select=carr,carrname,cmtcat

In order to execute a SendSuite Link query via some custom code you need to issue a request using the HTTP GET method and then obtain the resulting data from the response.



```
string result = String.Empty;
string url = "http://localhost:8080/link/rest/carrier/?apikey=975-747-559";

// Prepare the request
HttpWebRequest request = WebRequest.Create(url) as HttpWebRequest;
request.Method = "GET";
request.ContentType = "application/xml";

// Get the response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
     using (StreamReader rdr = new StreamReader(response.GetResponseStream()))
     {
          result = rdr.ReadToEnd();
     }
}
```

In this example, a URL was supplied to the web request in the following format.

{ServerName}/link/rest/*{ObjectName}*?apikey=000-000-000&*{Argument1}*&*{Argument2}*…

ServerName = The host name of the server hosting SendSuite Tracking and SendSuite Link.

ObjectName = The entity object name specifying what data is being used.

Argument(s) = A series of query keywords forming the criteria and options.

Encoding and Wildcards

The URI specification only allows certain characters within a URI string. It also reserves certain characters for its own specific use. Be advised of the following when constructing the URI string:

- The US-ASCII alphabetic characters a-z and A-Z are allowable.
- The decimal digit characters 0-9 are allowable.
- The following are allowable: _-!~'()*
- The following characters are allowed but are reserved for URI syntax: ,;:$&+=?/\[]@

For more information on the URI specification, visit https://www.w3.org/Addressing/URL/uri-spec.html

All other characters must be encoded using the '%' character followed by a two-digit hexadecimal number representing the desired character in the ASCII table. For example, the string john&doe must be encoded as john%26doe.

Utilizing '%' as a wild card requires '%25' (25 is the hex value for %)

Example:

http://localhost:8080/link/rest/mail/?apikey=###-###-###&filter=cartrk like ('%2510048')

Note:

Automated systems should encode automatically. Manual encoding/decoding should only be necessary when placing the URI string in a browser's address bar.

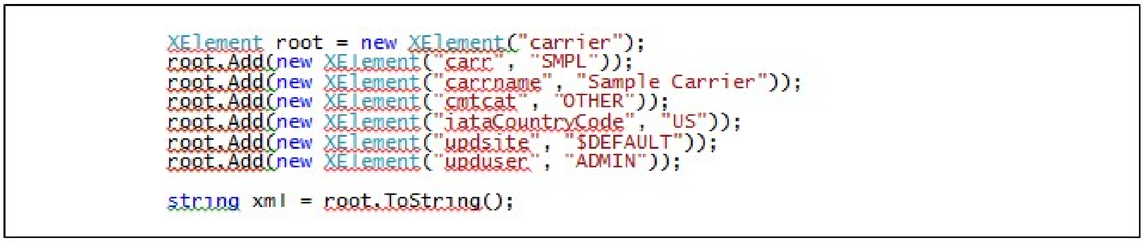
Saving Data

SendSuite link will perform a uniform save operation in that it will insert a new record if one does not exist with the provided primary key value. If a record exists based on the provided primary key value SendSuite Link will perform an update of that record. It is important to note that in the case of an update all fields in the record will be updated. For this reason the recommended technique for updates should be as follows:

1. Retrieve a specific record using a query operation.
2. Update the resulting data with the required changes.
3. Save the updated data.

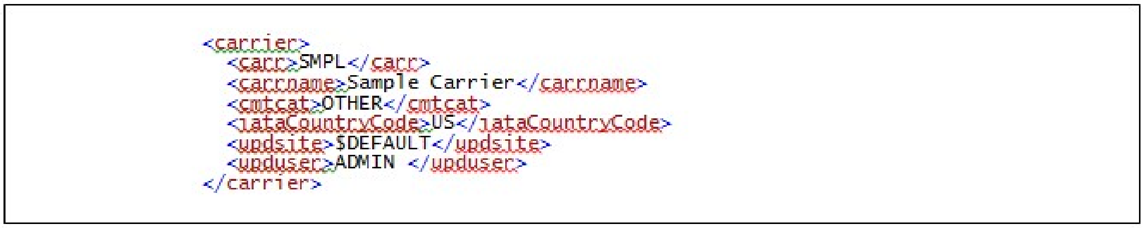
In order to save data using SendSuite Link via some custom code you need to issue a request using the HTTP POST method and provide an accompanying data payload of the data to be saved. Data to be saved can be provided as XML or JSON content within your code.

Using the .NET Framework, an XML document can be assembled based on the XSD data structures that are viewable once SendSuite Link is installed. The following code snippet illustrates the steps required to prepare an XML data structure for a new Carrier record using Microsoft's LINQ to XML classes.



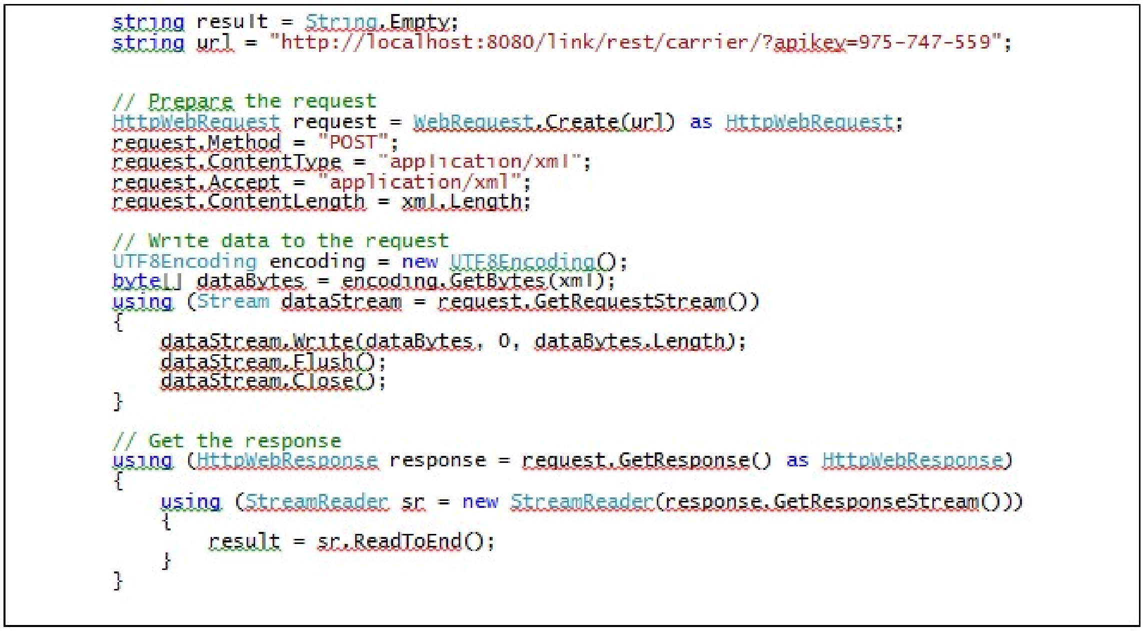
```
XElement root = new XElement("carrier");
root.Add(new XElement("carr", "SMPL"));
root.Add(new XElement("carrname", "Sample Carrier"));
root.Add(new XElement("cmtcat", "OTHER"));
root.Add(new XElement("iataCountryCode", "US"));
root.Add(new XElement("updsite", "$DEFAULT"));
root.Add(new XElement("upduser", "ADMIN"));

string xml = root.ToString();
```

The above code will produce an XML document that looks like this.



```
<carrier>
  <carr>SMPL</carr>
  <carrname>Sample Carrier</carrname>
  <cmtcat>OTHER</cmtcat>
  <iataCountryCode>US</iataCountryCode>
  <updsite>$DEFAULT</updsite>
  <upduser>ADMIN </upduser>
</carrier>
```

Once you have the prepared the data that needs to be saved, you need to prepare the request specifying the object type and the HTTP POST method. You must also write the XML data to the request before submitting it to the server.



```
string result = String.Empty;
string url = "http://localhost:8080/link/rest/carrier/?apikey=975-747-559";


// Prepare the request
HttpWebRequest request = WebRequest.Create(url) as HttpWebRequest;
request.Method = "POST";
request.ContentType = "application/xml";
request.Accept = "application/xml";
request.ContentLength = xml.Length;

// Write data to the request
UTF8Encoding encoding = new UTF8Encoding();
byte[] dataBytes = encoding.GetBytes(xml);
using (Stream dataStream = request.GetRequestStream())
{
    dataStream.Write(dataBytes, 0, dataBytes.Length);
    dataStream.Flush();
    dataStream.Close();
}

// Get the response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
    using (StreamReader sr = new StreamReader(response.GetResponseStream()))
    {
        result = sr.ReadToEnd();
    }
}
```

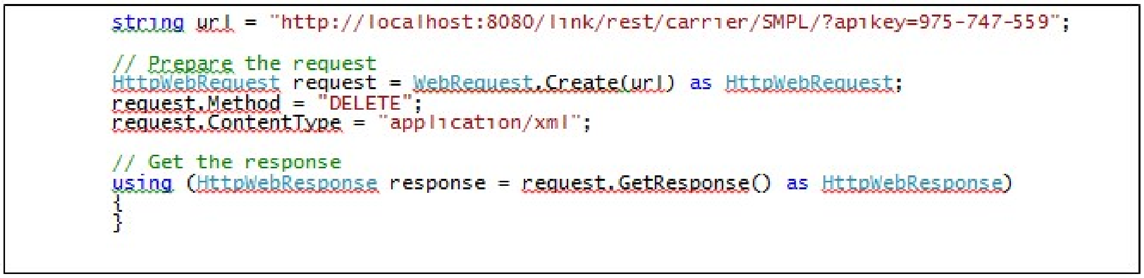
Once the request has been submitted to the server the resulting data is returned back as part of the response. In the case of an inserted record, a system generated primary key value will be included in the response data.

In the case of updating an existing record, the same approach is used except for the fact that the XML data will not be assembled ahead of time and will instead result from a SendSuite Link query.

Deleting Data

SendSuite Link deletes single records from the system using a primary key value. This is done by issuing a request via the HTTP DELETE method.

In the example below, the Carrier record with a unique ID of ‘SMPL’ is deleted and the primary key value is specified in the submitted URL.



```
string url = "http://localhost:8080/link/rest/carrier/SMPL/?apikey=975-747-559";

// Prepare the request
HttpWebRequest request = WebRequest.Create(url) as HttpWebRequest;
request.Method = "DELETE";
request.ContentType = "application/xml";

// Get the response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
}
```

Sending Email

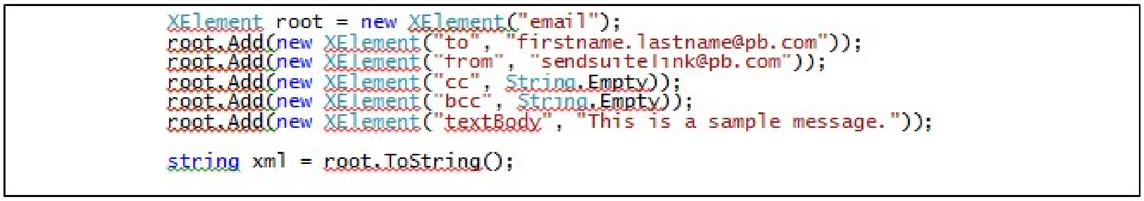
SMTP settings must be configured on the application server within SendSuite Administrator in order for the SendSuite Link email API to work. This can be done as follows.

1. Using your web browser open SendSuite Administrator at http://{MyServerURL}/SendSuiteAdmin using the “admin” user ID and password.
2. From the Setup menu, select Enterprise, SMTP Settings.

SendSuite Link will make use of the SMTP configuration settings for the email subject and the from address unless they are overridden by values in the XML data payload you provide.

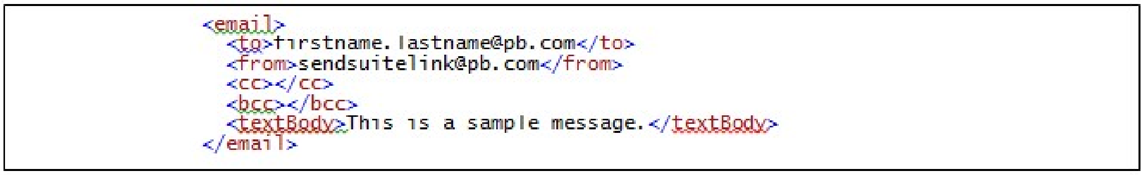
In order to send an email using SendSuite Link via some custom code you need to issue a request using the HTTP POST method and provide an accompanying data payload of the email message. The email message data can be provided as XML or JSON content within your code. The email API within SendSuite Link is used in the same way as the manner in which data is saved. The difference is in the data being submitted with the request.

The following code snippet illustrates the steps required to prepare an XML data structure for a new email message using Microsoft’s LINQ to XML classes.



```
XElement root = new XElement("email");
root.Add(new XElement("to", "firstname.lastname@pb.com"));
root.Add(new XElement("from", "sendsuitelink@pb.com"));
root.Add(new XElement("cc", String.Empty));
root.Add(new XElement("bcc", String.Empty));
root.Add(new XElement("textBody", "This is a sample message."));

string xml = root.ToString();
```

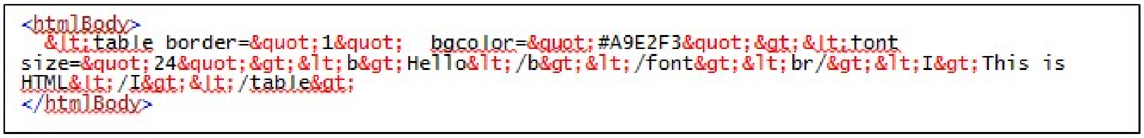
The above code will produce an XML document that looks like this.



```
<email>
  <to>firstname.lastname@pb.com</to>
  <from>sendsuitelink@pb.com</from>
  <cc></cc>
  <bcc></bcc>
  <textBody>This is a sample message.</textBody>
</email>
```

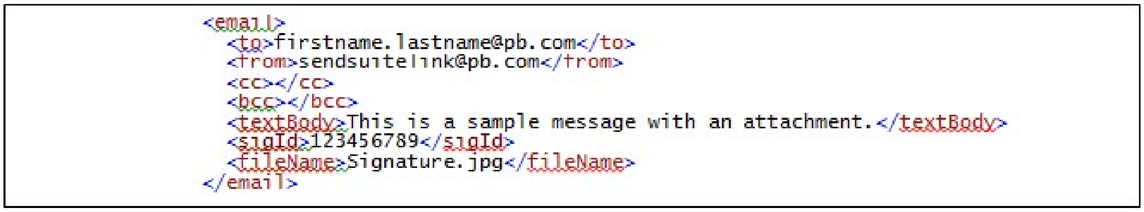
Once you have created the email message data structure it can be sent in the same manner as with saving or updating data. Refer to the C# code example in the “Saving Data”section that makes an HTTP POST request.

SendSuite Link allows you to send email messages in either plain text format or HTML format. This is done by including either the <textBody> or <htmlBody> XML tags, respectively. When providing HTML content within the <htmlBody> tags of an email message the data must be properly encoded using ampersands as shown in the following example.



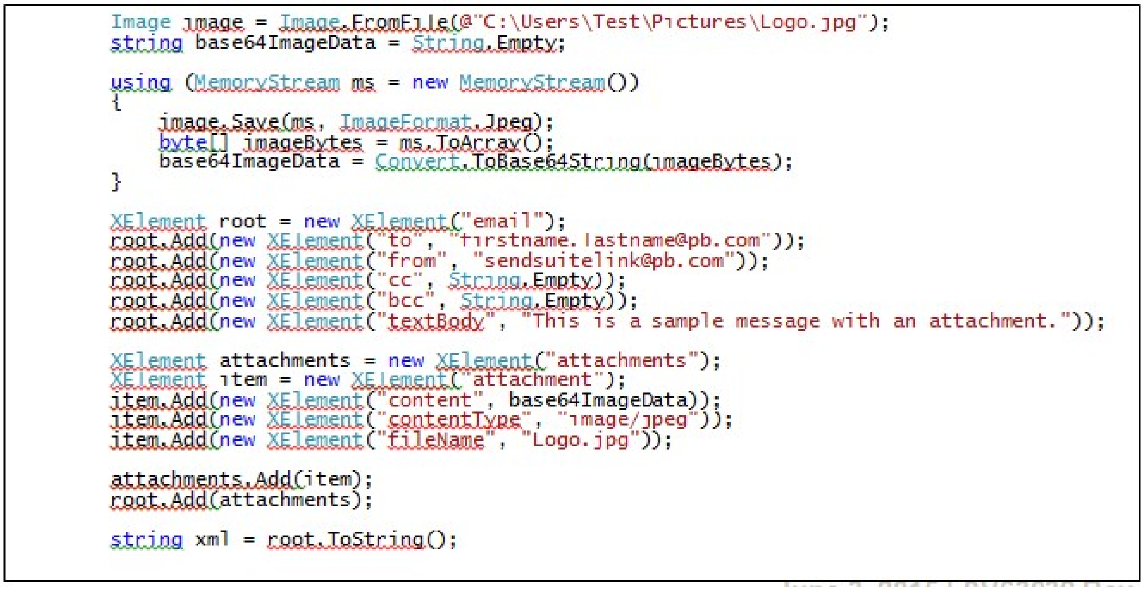
```
<htmlBody>
  &lt;table border=&quot;1&quot;   bgcolor=&quot;#A9E2F3&quot;&gt;&lt;font
size=&quot;24&quot;&gt;&lt;b&gt;Hello&lt;/b&gt;&lt;/font&gt;&lt;br/&gt;&lt;I&gt;This is
HTML&lt;/I&gt;&lt;/table&gt;
</htmlBody>
```

Email messages within SendSuite Link can also include attachments of images that are already stored in the SendSuite Tracking application database. This can be done by including the <sigId> tag in the XML data structure that references an image ID in the RCV_SIG table of the SendSuite Tracking database. In this case SendSuite Link will automatically include an email attachment of the image with the associated ID.



```
<email>
  <to>firstname.lastname@pb.com</to>
  <from>sendsuitelink@pb.com</from>
  <cc></cc>
  <bcc></bcc>
  <textBody>This is a sample message with an attachment.</textBody>
  <sigId>123456789</sigId>
  <fileName>Signature.jpg</fileName>
</email>
```

External images loaded from the file system can also be included as attachments. This can be done by supplying a 64-bit encoded string of the image data as part of the XML data structure. The following code example shows how to convert an image to a Base64 encoded string and include it within the XML data structure.



```
Image image = Image.FromFile(@"C:\Users\Test\Pictures\Logo.jpg");
string base64ImageData = String.Empty;

using (MemoryStream ms = new MemoryStream())
{
    image.Save(ms, ImageFormat.Jpeg);
    byte[] imageBytes = ms.ToArray();
    base64ImageData = Convert.ToBase64String(imageBytes);
}

XElement root = new XElement("email");
root.Add(new XElement("to", "firstname.lastname@pb.com"));
root.Add(new XElement("from", "sendsuitelink@pb.com"));
root.Add(new XElement("cc", String.Empty));
root.Add(new XElement("bcc", String.Empty));
root.Add(new XElement("textBody", "This is a sample message with an attachment."));

XElement attachments = new XElement("attachments");
XElement item = new XElement("attachment");
item.Add(new XElement("content", base64ImageData));
item.Add(new XElement("contentType", "image/jpeg"));
item.Add(new XElement("fileName", "Logo.jpg"));

attachments.Add(item);
root.Add(attachments);

string xml = root.ToString();
```

Reports

In order to access reports via SendSuite Link via some custom code you need to issue a request using the HTTP GET method and then obtain the resulting data from the response. SendSuite Link provides the following queries as a means of obtaining information about reports within a particular SendSuite Tracking system.

- Get a list of all reports.
  http://localhost:8080/link/rest/report/?apikey=975-747-559
- Get a list of all reports with descriptions and associated parameters.
  http://localhost:8080/link/rest/report/?apikey=975-747-559&full
- Run a specified report using the report name.
  http://localhost:8080/link/rest/report/run?apikey=975-747-559&__r=Buildings.rpt

Run Report Parameters

The following parameters are available when requesting to run a report:

Parameter Name	Permitted Values	Description
&ReportFormat=<value>	PDF; HTML	Specifies the report output type.
&BtnText=<value>	String value	Specifies the button text to be displayed; default is “Run Report”
&MenuShow=<value>	True; False	Shows or hides the upper blue menu on the report execution page.

Examples 1 and 2 above can be executed via custom code in the same manner as other queries within SendSuite Link. Refer to the C# code example in the “Querying Data”section that makes an HTTP GET request.

Since Example 3 above executes a report, it is likely that a typical scenario would be to generate a hyperlink or open a web browser with the URL to execute the report.

Return Codes

Since SendSuite Link relies on HTTP requests any errors will be returned as part of the response. The general rule is that a status code of 300 or greater denotes a failure. Additional information can be found in the SendSuite Link documentation.

Common status codes that will be returned from the server are as follows:

HTTP Status Code	HTTP Status Description	SendSuite Link Usage
200	OK	Indicates success.
201	Created	Indicates a successful record insert.
204	No Content	Indicates a successful deletion.
400	Bad Request	Indicates that supplied data was not in the correct format.
401	Unauthorized	Invalid or incorrect API key.
404	Not Found	Data was not found.
405	Method Not Allowed	Unsupported method.
500	Server Error	A server error occurred.

Powerlogic .NET

SendSuite Designer offers a programmability feature called PowerLogic .NET which allows for custom code to be written and executed within SendSuite Tracking screens. Once SendSuite Link is installed and configured with the SST API key, some additional functions can be used from within PowerLogic .NET. This extends the use of SendSuite Link to offer customization capability within SendSuite Tracking as well as extending it to other systems.

The following Powerlogic .NET functions encapsulate and make use of SendSuite Link capabilities. Essentially, these functions provide an internal wrapper around the SendSuite Link REST API.

- **LinkCount** - Count the number of records in the specified table.
  - String (objectName): The name of the table to count.
  - String (filter): Criteria by which to filter the result set.
- **LinkGetList** - Retrieve all records from the specified table filtering by the specified criteria.
  - String (objectName): The name of the table to retrieve from.
  - Integer (skip): The number of records to skip.
  - Integer (top): The number of records to retrieve.
  - String (filter): The criteria by which to filter the result set.
  - String (orderby): The column to order the result set by.
  - Boolean (full): Flag to return full column list or name/value pair.
- **LinkGetItem** - Retrieve the record with the specified ID from the specified table.
  - String (objectName): The name of the table to retrieve from.
  - String (id): The ID of the record to retrieve.
  - String (nameOfId): Some tables have multiple IDs. This field tells the system which ID to retrieve by. (For example: MAILS can use PIECEID, CARTRK, TRACKNO, or POUCHID).
- **LinkSaveItem** - Save the specified item in the specified table.
  - String (objectName): The name of the table to insert into.
  - Collection (data): The collection of values to insert into the table.
- **LinkDeleteItem** - Delete a record from the specified table.
  - String (objectName): The name of the table to delete from.
  - String (id): The ID of the record to delete.

If SendSuite Link is not installed or configured properly an attempt to use one of these functions will throw a FeatureNotAvailable exception.

The SendSuite Link functions within Powerlogic .NET each return a LinkResponse object. A LinkResponse object contains:

- **Code** (Integer) – The code of the HTTP response returned from SendSuite Link.
- **Message** (String) – The message from the HTTP response.
- **Count** (Integer) – The result from a LinkCount call or the number of records returned from a call to LinkGetList.
- **Rows** – Collection of records returned from LinkGetItem or LinkGetList. LinkGetItem returns a single row, while LinkGetList returns multiple rows.
- **FirstRow** – The first row in the Rows collection.

For more information and examples refer to the Powerlogic .NET Programmer's Reference and Powerlogic .NET Function reference within the SendSuite Designer online help.

Version Reference

The table below shows the releases of SendSuite Tracking that have a corresponding version of SendSuite Link available.

Note:

In SendSuite Link v1.90 the URL query syntax was revised and which makes it incompatible with the previous version.

SendSuite Tracking Version	SendSuite Link Version
10.80	1.80.17
10.90	1.90.08
11.0	2.0.2
11.0.1	2.0.3
11.1	2.0.8
11.1.1	2.0.9
11.4.4	2.6.66
11.5	2.6.66
11.6	2.7.13
11.7	2.8.50
11.8.1	2.91.3
11.9	2.94.2
12.0	2.93.15
12.1	2.94
12.2	2.95.1

SendSuite Tracking Version	SendSuite Link Version
12.3	2.96.4
12.4	2.97.5
12.5	2.97.5
12.6	2.99.14
12.7	2.100.5
12.8	2.101.17
12.9	2.102.10
14.0	2.103.14
14.1	2.104.20
14.2	2.105.28
14.3	2.106.14
14.4	2.107.9
14.5	2.108.5
14.6	2.109.2
14.7	2.110.8

3001 Summer Street
Stamford, Connecticut 06926
www.pitneybowes.com

SV63032 Rev. H