



Solution Guide

SuiteSolutions - CSV Integrator

Solution Version: 2.0.1

© 2015 NetSuite Inc.

Any reproduction or distribution of any part of this document without the prior written permission of NetSuite Inc. is strictly prohibited.

The information included in this document is confidential and proprietary information of NetSuite Inc.

Document Version: 1. 9, May 16, 2016

NetSuite Inc.
2955 Campus Drive, Suite 100
San Mateo, CA. 99403-2511

Table of Contents

1	Document Overview	4
1.1	Symbols and Conventions.....	4
1.2	Terms and Definitions	4
2	Solution Overview.....	5
2.1	Supported Features	5
3	Setup and Configuration	6
3.1	Bundle Installation.....	6
3.2	Feature Dependencies.....	6
3.2.1	SuiteCloud Features.....	6
3.2.2	Token Based Authentication	6
3.3	JAVA Environment.....	7
3.4	Create an Application for Token Based Authentication.....	7
3.5	Create Access Token for a User	8
3.6	Deploy the Script to Preferred Roles	9
3.7	Get the Broker Files	9
3.8	Broker Configuration	9
3.8.1	Create Folders	9
3.8.2	Get the RESTlet's External URL	9
3.8.3	Define the Configuration File.....	10
3.8.4	Specify the Queue Number	11
3.9	Create Saved CSV Import.....	12
3.10	Create a Integration Mapping Record	12
3.11	CSV File Format	13
4	Working with CSV Integrator.....	14
4.1	Using the Integrator.....	14
4.1.1	Using the Integrator Broker User Interface - Local	14
4.1.2	Using the Integrator Broker User Interface - FTP	15
4.1.3	Using the Integrator Console Interface	17
4.1.4	Using the Integrator via Windows Scheduler	18
4.2	CSV File with More Than 24 999 Rows.....	19
4.3	Upload a JE CSV file with Debit and Credit rows NOT balanced.....	19




4.4	Upload a JE CSV file with Debit and Credit rows balanced	19
4.5	Error in Process	19
4.6	RUN UI	19
5	Appendix	20
5.1	Bundle Components.....	20
5.1.1	Custom Lists/Records	20
5.1.2	CSV Imports.....	20
5.1.3	Forms	20
5.1.4	Scripts.....	20
5.1.5	Roles and Permission	20
5.2	Limitations.....	20
5.3	Solution Design.....	21
5.3.1	RESTlet SuiteScript Automation to Import Files	21
5.3.2	NetSuite Properties Configuration File	21

1 Document Overview

SuiteSolutions are pre-built offerings used to accelerate the delivery of custom functionality into Customer's account. These solutions were designed and developed by NetSuite Professional Services. SuiteSolutions are delivered as Non-Managed bundles i.e. the custom code/configurations delivered with the bundle is not locked and can be further customized on a customer account to meet unique business requirements. For the Terms of Service for SuiteSolutions, refer to <http://www.netsuite.com/tos>. Notwithstanding anything to the contrary in this Solutions Guide or on SuiteAnswers, in the event of any conflict between this Solution Guide and the Terms of Service for SuiteSolutions, the Terms of Service for SuiteSolutions shall supersede and control.

This is the Solution Guide for SuiteSolution - CSV Integrator. This documentation will guide the user how to setup, configure and work with the solution with the given use cases/scenarios as examples.

1.1 Symbols and Conventions

Symbol	Description
	Indicates situation the user should be specifically aware of when completing a task.
 	Indicates helpful tips, shortcuts and suggestions. The 'bulb' icon explains general information around NetSuite while the 'record' icon gives important details related to records, fields, forms and validations.

Convention	Description
Setup > Custom > NSTS / Integration Mapping	The (>) symbol indicates a sequence of actions, such as selecting items from a menu or toolbar, or pressing buttons in a window. This example directs you to go to Setup tab and hover to Custom and select NSTS Integration Mapping

1.2 Terms and Definitions

Term	Definition
Integration Broker	A client external module hosted outside the NetSuite application and developed using core JAVA technology as an executable file (.exe). This can be run using the UI or the backend batch process.
RESTlet	A form of web services integration. Each RESTlet is a server-side script that operates in a request-response model, and is invoked by an HTTP request to a system-generated URL.
Mapping Record	Custom record that stores the CSV Import Mapping IDs

2 Solution Overview

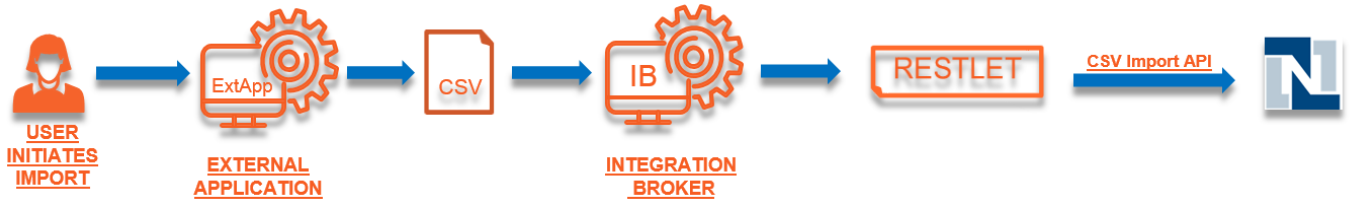


Figure 1 – CSV Integrator Process Flow

The integration is built using SuiteScript RESTlet and a broker in JAVA technologies to invoke the RESTlet. Below is an overview of the solution:

- The Integration Broker reads the CSV files on generated by the external applications on FTP Server or local folder and sends the raw data as JSON string to the RESTlet.
- The RESTlet will then convert the raw string into chunks of 25K (based on rows) and will invoke the CSV Import API to initiate the import process.
- *The CSV Import will be added to the import queue and functions as any other NetSuite CSV Import jobs.
- Once the import has been processed, the user will be notified of the import status using standard NetSuite functionality. The JOB ID will be sent back to the broker.
 - Once a success response is received, the CSV file will be moved into a processed folder.
 - CSV files from the FTP server will also be moved on the processed folder on the server and will be deleted on the original folder.
 - In case of errors, the file will be moved to the error folder and an error log will be added.

**For SuiteCloud PLUS Licensed Accounts, a specified queue number (2-5) can be set on the CSV import.*

2.1 Supported Features

The following features are supported by the solution.

1. One Integration solution to import records supported by CSV Import functionality into NetSuite. Batch integration for periodic import of data from external applications.
 - On-demand via User Interface.
 - Only CSV File types are supported.
2. Allows to import files more than 25K in size, it can split up file into chunks of 25k. Also it makes sure that the transaction lines are not split across files.
3. Multiple file types can be imported via one source location using the default Broker.
4. Default Broker provided with the solution. The Broker retrieves the mapping id information of the Saved CSV Import based on the column headers.
5. Uses the standard CSV mapping feature in the CSV Import. Mapping should be created manually and added in the mapping definition record.

6. Supports both email/login based and Token Based Authentication for the authentication. Token based authentication is recommended but the user can continue to use the standard method of authentication if they choose to.
7. Multiple CSV files can be imported via one source location using the default Broker but only one mapping is allowed for each record type. Also for each record, a primary field can be specified in the mapping record to allow the broker to split up the files.
8. The solution can download files from FTP locations and import into NetSuite. The successful import processes are then moved to the processed folder on the FTP site if write access is enabled.
9. Supports multi-queue execution of the import processes. The CSV Import mapping has a queue number and this solution allows to provide a queue number if there is a desire to run the import on specific queues.
10. Includes support for Character encoding that are available in the CSV Import Mapping functionality.

3 Setup and Configuration

3.1 Bundle Installation

To install the SuiteSolutions – CSV Integrator, a user has to have an Administrator or SuiteBundler permission in an account.

1. Go to **Customization> SuiteBundler> Search & Install Bundles**.
2. Search for the **SuiteSolutions – CSV Integrator** bundle and click the name.
3. Click the **Install** button to begin the installation process.

Please refer to these NetSuite Help links for more information about bundle installation: [Installing a Bundle](#), [Choosing a Bundle to Install](#), [Understanding Bundle Searches](#).



It is recommended that process is validated in a sandbox prior to installation in a production account.

3.2 Feature Dependencies

The following features needs to be enabled to the account:

3.2.1 SuiteCloud Features

- Client SuiteScript
- Server SuiteScript
- Custom Records

3.2.2 Token Based Authentication

Enable the Token-Based Authentication by going to **Setup> Company> Enable Features** and under SuiteCloud tab, check the **Token-Based Authentication** field on the Manage Authentication section.

☒ **TOKEN-BASED AUTHENTICATION**

ENABLE TOKEN-BASED AUTHENTICATION AS AN ADDITIONAL AUTHENTICATION MECHANISM FOR YOUR USERS. BY ENABLING THIS FEATURE, YOU AGREE TO [SUITECLOUD TERMS OF SERVICE](#).

Figure 2 - Enable Token Based Authentication

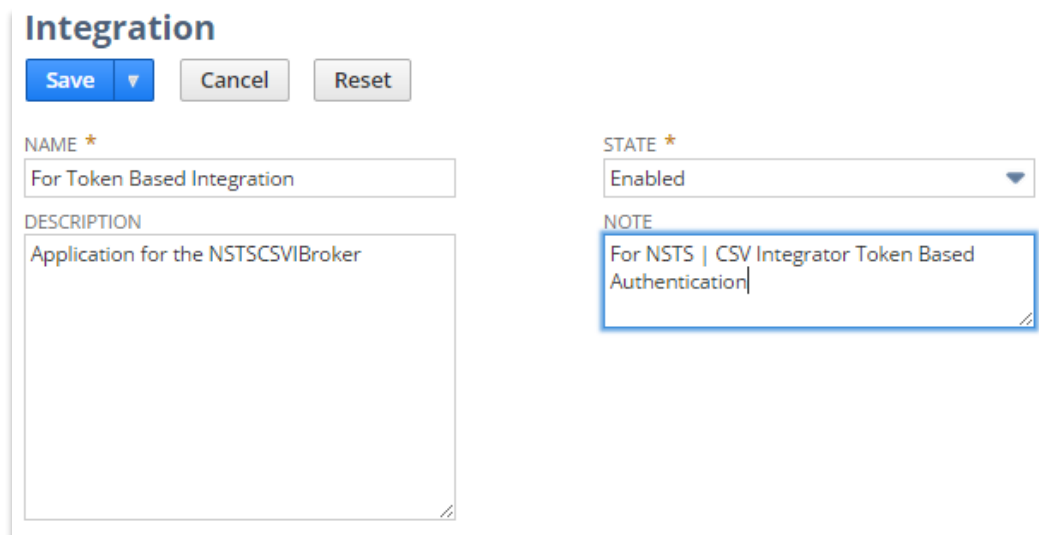
3.3 JAVA Environment

Requires installed JAVA environment. (E.g., JRE 1.7). The Java Software installed must be above version 7. The folder path where JAVA is located must be set on the Run Via Console.bat file or Run Via UI.bat file, see [Get Broker Files](#).

3.4 Create an Application for Token Based Authentication

After enabling the Token-Based Authentication, an application must be created for it. Below steps describes how to create an application for Token Based Authentication.

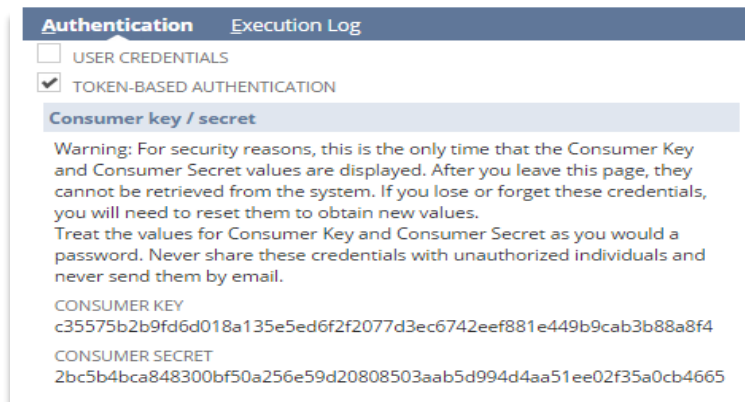
1. Go to **Setup> Integration> Manage Integrations> New**. The new Integration entry form will display.
2. Enter a name for the application.
3. The State is enabled by default.
4. On the authentication subtab:
 - a. Un-check the User Credentials checkbox.
 - b. Check the Token-Based Authentication checkbox.
5. Click **Save**.



The screenshot shows the 'Integration' form in NetSuite. At the top, there are three buttons: 'Save' (blue), 'Cancel', and 'Reset'. Below these are two main sections. The left section has a 'NAME' field with a red asterisk, containing the text 'For Token Based Integration'. Below it is a 'DESCRIPTION' text area containing 'Application for the NSTSCSVIBroker'. The right section has a 'STATE' dropdown menu with a red asterisk, currently set to 'Enabled'. Below the state is a 'NOTE' text area containing 'For NSTS | CSV Integrator Token Based Authentication'.

Figure 3 - Integration for Token Based Authentication

6. After saving, the consumer key and consumer secret will display for the application under the Authentication subtab.



Authentication Execution Log

☐ USER CREDENTIALS

☒ TOKEN-BASED AUTHENTICATION

Consumer key / secret

Warning: For security reasons, this is the only time that the Consumer Key and Consumer Secret values are displayed. After you leave this page, they cannot be retrieved from the system. If you lose or forget these credentials, you will need to reset them to obtain new values. Treat the values for Consumer Key and Consumer Secret as you would a password. Never share these credentials with unauthorized individuals and never send them by email.

CONSUMER KEY
c35575b2b9fd6d018a135e5ed6f2f2077d3ec6742eef881e449b9cab3b88a8f4

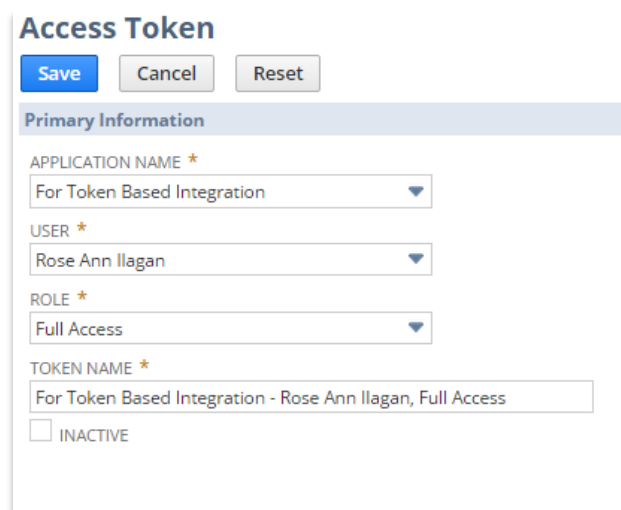
CONSUMER SECRET
2bc5b4bca848300bf50a256e59d20808503aab5d994d4aa51ee02f35a0cb4665

Figure 4 - Generated Consumer Key/Secret

7. The consumer key and consumer secret must be copied before leaving the page. These keys will only be displayed once and cannot be retrieved from the system.

3.5 Create Access Token for a User

1. Go to **Setup> User/Roles> Access Tokens> New**
2. On the Application Name, select the created application.
3. On the User field, select the name of the user to grant access.
4. On the Role field, select the type of access for the selected user.
5. **Save.**



Access Token

Primary Information

APPLICATION NAME *
For Token Based Integration

USER *
Rose Ann Ilagan

ROLE *
Full Access

TOKEN NAME *
For Token Based Integration - Rose Ann Ilagan, Full Access

☐ INACTIVE

Figure 5 - Access Token for User

3.6 Deploy the Script to Preferred Roles

The RESTlet must be deployed only to roles allowed to do the importing. Below steps describes how to deployment to preferred roles.

1. Open the **NSTS RESTlet Base Integrator** RESTlet via **Customization> Scripting> Scripts**
2. On the Deployments tab, open and edit the **NSTS RESTlet Base Integrator** deployment.
3. From the Audience section, select the preferred roles by unchecking the Select All checkbox.
4. Press CTRL to multi-select the preferred roles.
5. Save the deployment.

The selected role must also be specified also in the [configuration file](#). It is not recommended to store an Administrator user/password in the properties file.

3.7 Get the Broker Files

Get the broker and console files from the bundle by going to Documents> Files> File Cabinet> and find the bundle ID of the solution. Click the **NSTSCSVIBroker.zip** link to download the files.

Contents:

1. netsuite.properties
2. lib
3. Sample CSV Files
4. README.txt
5. NSTSCSVIBroker.jar
6. Run Via Console.bat
7. Run Via UI.bat

3.8 Broker Configuration

3.8.1 Create Folders

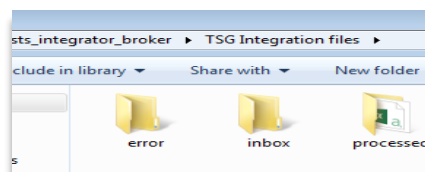
Create the Inbox, Processed and Error folders and get the folder directories.

Example:

```

.\\TSG Integration files\\inbox
.\\TSG Integration files\\processed
.\\TSG Integration files\\error

```



Note: When using Linux based OS, the folder needs to have read/write access upon creation.

3.8.2 Get the RESTlet's External URL

Access the RESTlet and find the **NSTS RESTlet Base Integrator** deployment to get Relative Path of URL.

Script Deployment

[Edit](#)
[Back](#)
[Actions](#)

SCRIPT
 NSTS RESTlet Base Integrator
TITLE
 NSTS RESTlet Base Integrator
ID
 customdeploy_tsg_integrator_reslet
☒ **DEPLOYED**

STATUS
 Released
LOG LEVEL
 Debug
URL
 /app/site/hosting/restlet.nl?script=85&deploy=1
EXTERNAL URL
 https://rest.na1.netsuite.com/app/site/hosting/restlet.nl?script=85&deploy=1

Figure 6 - Script Deployment

3.8.3 Define the Configuration File

Create a properties file with if no properties file exists yet and set the following details. Save it as **netsuite.properties**.

See [NetSuite Properties Configuration File](#) for a sample properties file.

NetSuite Connection	
netsuite.ws.url	Web Service URL. This varies on the account type (production, sandbox)
netsuite.account	NetSuite company account.
netsuite.email	Email address used to log in on NetSuite account.
netsuite.password	Password used to log in on NetSuite account.
netsuite.role	Role id to log in. Please see NetSuite Help for the list of IDs and Deploy the Script to Preferred Roles for more details.
netsuite.login	Login preference, OAuth for token based authentication and NLAUTH for standard login.
netsuite.csvStorage	CSV files storage, FTP if the files are to be read on ftp server, LOCAL if files are in local directory, this is set if application is executed on console.
netsuite.suiteCloudAcct	Set to true ONLY IF account is SuiteCloud Licensed Account to support multiQueueing of CSV imports, this is set if application is executed on console, else set this to false .
netsuite.csvImportQueue	See Specify the Queue Number .
Token Based Authentication	
netsuite.consumerKey	Consumer id generated for the application.
netsuite.consumerSecret	Consumer secret generated for the application.
netsuite.tokenId	Token id generated for the user/role.
netsuite.tokenSecret	Token secret generated for the user/role.
FTP	
netsuite.ftp.username	FTP username.
netsuite.ftp.password	FTP password.
netsuite.ftp.host	FTP host is required.
netsuite.ftp.path	FTP path: location of CSV files, must be a directory.
netsuite.ftp.port	FTP port (optional).

<code>netsuite.ftp.removeProcessedFile</code>	This is enabled when CSV files processed will be removed on the current Directory and move to 'processed'/error file.
<code>netsuite.ftp.errorDir</code>	Directory for the processed CSV files that were not successfully imported.
<code>netsuite.ftp.processedDir</code>	Directory for the processed CSV files that were successfully imported.
<code>netsuite.tsg.Integration.restlet.url</code>	Relative Path URL of the RESTLet from the deployment script. See Get the RESTLet's URL .
<code>baselogdir</code>	<code>.\logs</code>
<code>execution.log</code>	<code>executionLog.txt</code>
<code>success.log</code>	<code>successLog.txt</code>
<code>failure.log</code>	<code>failureLog.txt</code>
<code>files.path.inbox</code>	File input directory. See Create Folders .
<code>files.path.processed</code>	File processed directory. See Create Folders .
<code>files.path.error</code>	File error directory. See Create Folders .
<code>netsuite.tsg.JobName</code>	"SuiteSolution Demo: {comname} - date : {datetime} import service CSV rows: {csvrows} User : {user} File Name: {fileName}"

Note: When using Linux based OS, the following folders must be manually created by the user with the read/write permission:

- **baselogdir** – for defining the path separator, use forward-slash or "/" and add another "/" at the end of the folder name. (Example: `./logs/`)
- **files.path.inbox** – See [Create Folders](#).
- **files.path.processed** – See [Create Folders](#).
- **files.path.error** – See [Create Folders](#).

3.8.4 Specify the Queue Number

By default, all CSV import jobs use a single queue. This queue is defined as queue 1. In accounts that have purchased a SuiteCloud Plus license, five queues are available for import jobs, instead of a single queue. For CSV Integrator implementation with 'SuiteCloud Plus' Licensed Account, since there are 5 queues only, 1 is reserved and the user can specify the queue numbers 2-5 in the [configuration file](#).

```

netsuite.login = OAUTH
netsuite.csvStorage = LOCAL
netsuite.suiteCloudAcct = true
netsuite.csvImportQueue = 5

```

Figure 7 - Queue Number in the Configuration File

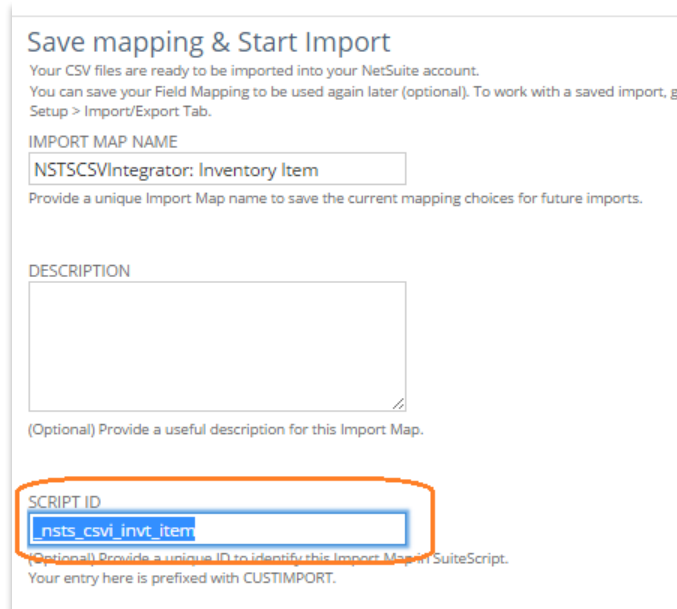
The broker will throw an error if the specified queue number is more than 5 or equal to 1.

Error messages:

- ERROR: CSV import queue number must be within 2-5
- ERROR: Invalid queue number (for non-integer value)
- ERROR: CSV import queue is required for multiQueueing. (if set to SuiteCloud Plus Licensed Account but no queue number is specified)

3.9 Create Saved CSV Import

Each record to be imported needs a CSV Import saved and should have a Script ID defined for each. Provide a Script ID for the Saved CSV Import to be used.



Save mapping & Start Import

Your CSV files are ready to be imported into your NetSuite account.
You can save your Field Mapping to be used again later (optional). To work with a saved import, go to **Setup > Import/Export Tab**.

IMPORT MAP NAME

 Provide a unique Import Map name to save the current mapping choices for future imports.

DESCRIPTION

 (Optional) Provide a useful description for this Import Map.

SCRIPT ID

 (Optional) Provide a unique ID to identify this Import Map in SuiteScript.
 Your entry here is prefixed with CUSTIMPORT.

Figure 8 - Sample Saved CSV Import

3.10 Create a Integration Mapping Record

This is the record to store the Saved CSV Import Template ID. An Integration Mapping record must be defined for each Saved CSV ID. To create a mapping record, go to **Setup > Custom > NSTS / Integration Mapping**.

NSTS | Integration Mapping ← → List Search Customize More

NSTSCSVIntegrator | Item (ITM Import.csv)

Save **Cancel** **Reset** | **Actions** ▼

Main

NAME *
NSTSCSVIntegrator | Item (ITM Import.csv)

MAPPING ID *
_nsts_csvi_invt_item

ID
2

☐ INACTIVE

CSV HEADER FORMAT *
Check #,Date,Table #,Open Time,Print Time,Server #,Covers,Membe

has Debit/Credit

PRIMARY ID *
Table #

Figure 9 - Sample Integration Mapping Record

Set the following fields:

- **Name** - Name of the mapping record.
- **Mapping ID** - From generated Script ID after [Saved CSV Import](#) creation.
- **CSV Header Format** – Actual header names of the uploaded CSV file.
- **Primary ID (has Debit/Credit)**

Save the mapping record.



The user/role to be used on running the application must have read/write or view access to the mapping record.

3.11 CSV File Format

Ensure that the file to be uploaded is saved in **.csv** format.

File name: Item Sample CSV Import.csv

Save as type: CSV (Comma delimited) (*.csv)

On the MS Excel application, go to **File> Save As** and select the **CSV (Comma delimited) (*.csv)** as the 'Save as type'.

4 Working with CSV Integrator

4.1 Using the Integrator

4.1.1 Using the Integrator Broker User Interface - Local

Pre-requisite:

- [CSV file for import.](#)
- [Broker Configuration.](#)
- [Saved CSV Import.](#)
- [Integration Mapping Record.](#)

1. Locate and open the **NSTSCSVIBroker.jar** file downloaded from the [File Cabinet](#).
2. The Broker User Interface will open.
3. On **CSV Storage**, dropdown, select **Local** on the CSV Storage dropdown.
4. On **Login Via**, the default value will be based on the login type specified on the configuration file:
 - **OAUTH** to login with authentication or
 - **NLAUTH** to login without authentication.
5. On **MAP ID**, select the [mapping record](#) created for the CSV.
6. On **CSV Import Queue**, select the # of queue to be used.
7. Click the browse button and locate the CSV file to import.
8. The file directory of the selected file will display on the Local File field.
9. Click the Process button.
10. A message box will display for status of process.
11. The JOB ID and import details will display on the log box of the UI.
12. A new job will be added to the CSV Import Job Status in NetSuite.



Figure 10 - Import Status Message Box

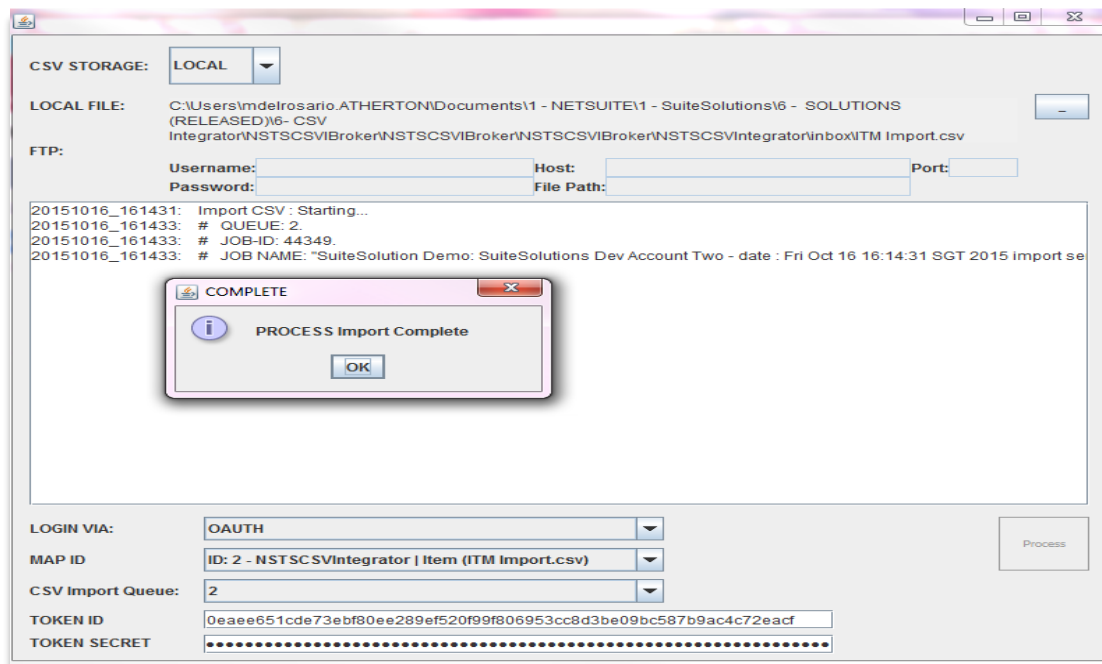


Figure 11 - Broker User Interface – Local

Job Status						
New Refresh		FILTERS				
DATE ▲	JOB NAME	STATUS	PERCENT COMPLETE	MESSAGE	CSV RESPONSE	QUEUE
10/16/2015 12:47 am	"SuiteSolution Demo: SuiteSolutions Dev Account Two - date : Fri Oct 16 15:47:37 SGT 2015 import service CSV rows: 169 User : rilagan@netsuite.com File Name: ITM Import.csv" Part:1	Complete	100.0%	168 of 168 records imported successfully	CSV Response	2

Figure 12 - CSV Import JOB Status

Note: The processed file will not be moved to the 'Processed' folder.

4.1.2 Using the Integrator Broker User Interface - FTP

Pre-requisite:

- CSV file for import.
- [Broken Configuration](#).
- [Saved CSV Import](#).
- [Integration Mapping Record](#).

1. Locate and open the **NSTSCSVIBroker.jar** file downloaded from the [File Cabinet](#).
2. The Broker User Interface will open.
3. On **CSV Storage**, dropdown, select **FTP** on the CSV Storage dropdown.
4. The browse button will be disabled.
5. Enter the following FTP details:
 - a. Username
 - b. Password
 - c. (Not specified)
 - d. File Path
 - e. Port

c. Host

6. On **Login Via**, the default value will be based on the login type specified on the configuration file:
 - **OAUTH** to login with authentication or
 - **NLAUTH** to login without authentication.
7. On **MAP ID**, select the [mapping record](#) created for the CSV.
8. On **CSV Import Queue**, select the # of queue to be used.
9. Click the Process button.
10. The broker will download the files from the specified file path and will be moved to the local inbox folder specified.
11. A message box will display for status of process.
12. The JOB ID and import details will display on the log box of the UI.
13. A new job will be added to the CSV Import Job Status in NetSuite.
14. The processed files will be moved to the local 'Processed' folder specified. If files had error upon import, it will be added moved on the local 'Error' folder specified.
15. If 'removeProcessedFile' parameter is set to true, the files will be moved to the specified 'error' and 'processed' folder on the FTP Server. The user must have a write access to the FTP server to successfully move the files.

When the broker is executed on a scheduled interval and 'removeProcessedFile' is set to false, it will always import the CSV files on the FTP inbox folder even those that were processed before, unless manually deleted on the FTP inbox folder.

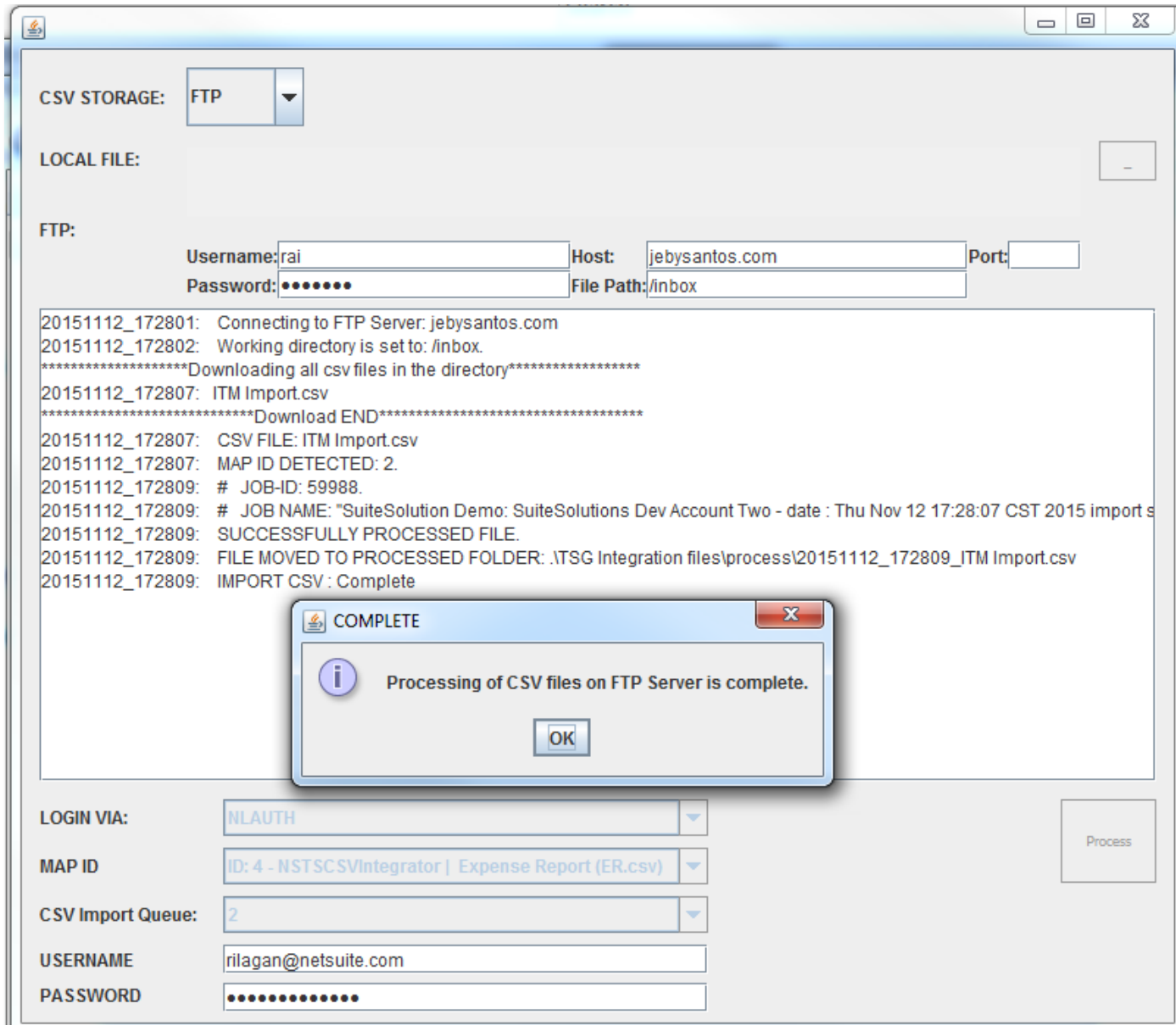


Figure 13 - Integrator Broker User Interface - FTP

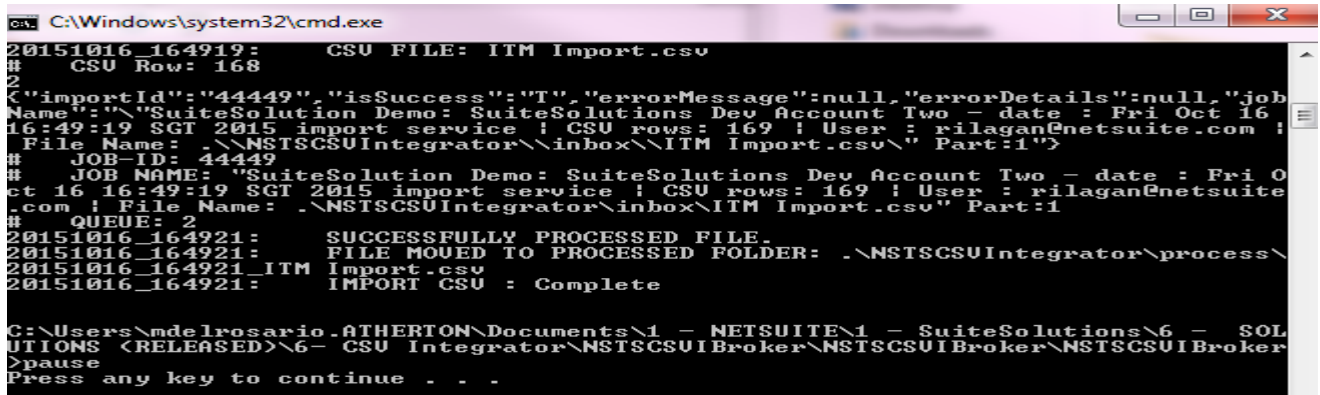
4.1.3 Using the Integrator Console Interface

Pre-requisite:

- CSV file for import.
- [Broken Configuration](#).
- [Saved CSV Import](#).
- [Integration Mapping Record](#).

1. Locate and open the **Run Console.bat** file downloaded from the [File Cabinet](#).

2. The console will automatically process all of the files inside the inbox folder.
3. The import status of the CSV will display on the console.
4. A new job will be added to the CSV Import Job Status in NetSuite.
5. The processed files will be moved to the 'Processed' folder.



```

C:\Windows\system32\cmd.exe
20151016 164919: CSU FILE: ITM Import.csv
# CSU Row: 168
2
{"importId":"44449","isSuccess":"T","errorMessage":null,"errorDetails":null,"job
Name":"SuiteSolution Demo: SuiteSolutions Dev Account Two - date : Fri Oct 16
16:49:19 SGT 2015 import service ! CSU rows: 169 ! User : rilagan@netsuite.com !
File Name: .\NSTSCSUIntegrator\inbox\ITM Import.csv\" Part:1"}
# JOB-ID: 44449
# JOB NAME: "SuiteSolution Demo: SuiteSolutions Dev Account Two - date : Fri O
ct 16 16:49:19 SGT 2015 import service ! CSU rows: 169 ! User : rilagan@netsuite
.com ! File Name: .\NSTSCSUIntegrator\inbox\ITM Import.csv\" Part:1
# QUEUE: 2
20151016 164921: SUCCESSFULLY PROCESSED FILE.
20151016 164921: FILE MOVED TO PROCESSED FOLDER: .\NSTSCSUIntegrator\process\
20151016 164921: ITM Import.csv
20151016 164921: IMPORT CSU : Complete

C:\Users\mdelrosario.ATHERTON\Documents\1 - NETSUITE\1 - SuiteSolutions\6 - SOL
UTIONS <RELEASED>\6- CSU Integrator\NSTSCSUIBroker\NSTSCSUIBroker\NSTSCSUIBroker
>pause
Press any key to continue . . .
  
```

Figure 14 - Integrator Console Interface

4.1.4 Using the Integrator via Windows Scheduler

To run the integrator via Windows Scheduler, a task must be created using the Windows Task Scheduler.

1. Select Create Task from the Actions panel.
2. On the General tab, enter the name of the task.
3. On the Trigger tab, click the 'New' button to specify the schedule settings of the task.
4. On the Actions tab, click the 'New' button to set the program or script to run. Browse for the Run Console batch file associated with the [Broker Files](#).
5. Set the other preferences on the Conditions and Settings tab.
6. Click OK to save.
7. A new task will be added to the Task Scheduler Library.
8. The task will run on the schedule defined.

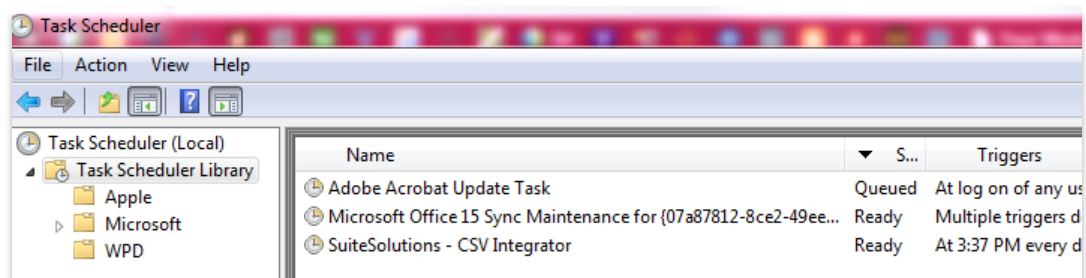


Figure 15 - Created Windows Task for CSV Integrator

4.2 CSV File with More Than 24 999 Rows

If a file has more than 24,999 rows, upon parsing of the data by the RESTlet; it will divide the rows into chunks of 25K rows and will invoke the CSV process. The mapping to use will be defined as a parameter on the deployment.

4.3 Upload a JE CSV file with Debit and Credit rows NOT balanced

If a JE's Debit and Credit line do not balance on CSV process, the import job will fail and the user will be notified of the error.

4.4 Upload a JE CSV file with Debit and Credit rows balanced

If a JE's Debit and Credit line do balance on CSV process, the import job will succeed and the user will be notified of the status.

4.5 Error in Process

In case of errors, the file will be moved to the error folder and an error log will be added.

4.6 RUN UI

Clicking the RUN UI file will display both the UI interface and the console that runs the JAR file.

5 Appendix

5.1 Bundle Components

Listed on this section are the components of the solution such as the custom record, custom fields, saved searches, scripts, etc., that were not mentioned in [3 Setup and Configuration](#).

Name: SuiteSolutions – CSV Integrator

Version: 2.0.1

5.1.1 Custom Lists/Records

- **Records**

Integration Mapping Record

Name	Name of the mapping record.
Mapping ID	From generated Script ID after Saved CSV Import creation.
CSV Header Format	Actual header names of the uploaded CSV file.
Primary ID (has Debit/Credit)	

5.1.2 CSV Imports

- **Saved CSV Imports**
 - NSTSCSVIntegrator: Journal Entry
 - NSTSCSVIntegrator: Expense Report (ER.csv)
 - NSTSCSVIntegrator: Inventory Item

5.1.3 Forms

- **Transaction Forms**
 - NSTS | GAW - Expense Report
 - Custom Journal Entry

5.1.4 Scripts

- **RESTlet**
 - NSTS RESTlet Base Integrator

5.1.5 Roles and Permission

The access for configuring and processing SuiteSolutions – CSV Integrator should be aligned with the standard NS security. Role used must have access to the mapping records (NSTS | Integration Mapping)

5.2 Limitations

The following items are not supported or a limitation of the solution.

1. Integrating/importing of records not supported through NetSuite's CSV import functionality.

2. Broker cannot validate sanctity of the data in the CSV File.
3. There is a limit of 10 MB per string used as RESTlet input or output (NS Limitation)
4. Does not support multi-file import.
5. UTF8 with BOM encoding is not supported.
6. Partial Fulfillments via CSV Import are **not** allowed since we can only import sales order numbers and not the line items. (Please see [SuiteAnswers ID: 38467, Rules on Importing Item Fulfillments via CSV](#))
7. CSV Integrator solution doesn't support SFTP/FTPS connections. It cannot download files using SFTP or FTPS in the current version.

5.3 Solution Design

5.3.1 RESTlet SuiteScript Automation to Import Files

- This SuiteScript automation will be invoke via a POST to a published RESTlet endpoint
- Accepting a JSON request originating from the Integration Broker, the RESTlet will determine the CSV import map id to be used for the import
- Next, the script will pass a raw string (CSV – retrieved from the JSON Request) as the Primary File
- NetSuite will queue the file for import and pass back a response containing the CSV Import Job ID

5.3.2 NetSuite Properties Configuration File

```
# Web Service (WSDL) URL
# [netsuite.wsdl.url] Webservices port location
# for sandbox account use
https://webservices.sandbox.netsuite.com/services/NetSuitePort_2015_2
netsuite.ws.url =
https://webservices.netsuite.com/services/NetSuitePort_2015_1
# LOGIN PREFERENCE
# {login} OAUTH for Token Based Authentication and NLAUTH for Standard Login
netsuite.login = NLAUTH

# ACCOUNT DETAILS (ALL REQUIRED)
# {account} netsuite company account
# {suiteCloudAcct} Set to true ONLY IF account is SuiteCloud Licensed Account
to support multiQueueing of csv imports
# {csvImportQueue} Set this if account is suitecloud licensed account
({suiteCloudAcct} is set to true), this will be the queue no to be used on csv
import. Select from queue no 2-5 only.
# Integration Restlet URL can be seen on Customization -> Script Deployments -
> NSTS RESTlet Base Integrator -> URL (relative path)
netsuite.account = 4125738
netsuite.suiteCloudAcct = false
netsuite.csvImportQueue = 3
netsuite.tsg.Integration.restlet.url =
/app/site/hosting/restlet.nl?script=127&deploy=1

# (FOR NLAUTH LOGIN PREFERENCE ONLY) LOGIN CREDENTIALS
```

```
# {email} email address used to log in on netsuite account
# {password} password used to log in on netsuite account
# {role} role id to log in
netsuite.email = suitesolutiondemo@netsuite.com
netsuite.password =
netsuite.role = 3

# (FOR OAUTH LOGIN PREFERENCE ONLY) LOGIN CREDENTIALS
# Note that failed login attempts must be less than 6
# {consumerKey} consumer id generated for the application
# {consumerSecret} consumer secret generated for the application
# {tokenId} token id generated for the user/role
# {tokenSecret} token secret generated for the user/role
# See suiteanswers for getting started on token based authentication
https://netsuite.custhelp.com/app/answers/detail/a\_id/41898
netsuite.consumerKey =
netsuite.consumerSecret =
netsuite.tokenId =
netsuite.tokenSecret =

# CSV FILES STORAGE
# Add the CSV Storage details if broker is run via console.
# {csvStorage} FTP if the files are to be read on ftp server, LOCAL if files
are in local directory
netsuite.csvStorage = LOCAL

# (FOR LOCAL AND FTP) CSV FILES STORAGE
# {inbox} File inbox directory, If FTP is enabled, CSV files will be
downloaded here. If LOCAL is enabled, CSV files are stored here.
# {processed} CSV files successfully processed on the inbox directory will be
deleted on the inbox and moved on this processed directory
# {error} CSV files not successfully processed on the inbox directory will be
deleted on the inbox and moved on this error directory
files.path.inbox = .\\TSG Integration files\\inbox
files.path.processed = .\\TSG Integration files\\process
files.path.error = .\\TSG Integration files\\error

# (FOR FTP PREFERENCE ONLY) CSV FILES STORAGE
# {username} ftp username
# {password} ftp password
# {host} ftp host is required
# {path} ftp path: location of csv files,must be a directory
# {port} ftp port
netsuite.ftp.username =
netsuite.ftp.password =
netsuite.ftp.host =
netsuite.ftp.path =
netsuite.ftp.port =

# (FOR FTP PREFERENCE ONLY) CSV FILES STORAGE
```

```
# {removeProcessedFile} Set to true if files processed will be moved on
processed or error folder on the server and deleted on the current folder
# {errorDir} Directory on the server for files that have an error upon process
# {processedDir} Directory on the server for files that are successfully sent
to CSV Import
netsuite.ftp.removeProcessedFile      = false
netsuite.ftp.errorDir                 = /error
netsuite.ftp.processedDir             = /processed

# JobName definition
# {datetime} concat the Netsuite date time
# {csvrows} concat the csv row count
# {compname} concat the company name
# {fileName} concat the file name uploaded
# {user} concat the user email
netsuite.tsg.JobName = "SuiteSolution Demo: {compname} - date : {datetime}
import service | CSV rows: {csvrows} | User : {user} | File Name: {fileName}"

# User Log files
baselogdir = .\\logs
execution.log = executionLog.txt
success.log = successLog.txt
failure.log = failureLog.txt

# Character Encoding
# Character Encoding must be the same for CSV Import configuration
# US-ASCII : Seven-bit ASCII, a.k.a. ISO646-US, a.k.a. the Basic
Latin block of the Unicode character set
# ISO-8859-1 : ISO Latin Alphabet No. 1, a.k.a. ISO-LATIN-1- Western
(Windows 1252) encoding
# UTF-8 : Eight-bit UCS Transformation Format (Unicode)- UTF-8
encoding
# UTF-16BE : Sixteen-bit UCS Transformation Format, big-endian
byte order
# UTF-16LE : Sixteen-bit UCS Transformation Format, little-
endian byte order
# UTF-16 : Sixteen-bit UCS Transformation Format, byte order
identified by an optional byte-order mark
# windows-1252 : Windows Latin-1
# GB18030 : Simplified Chinese, PRC standard
# Shift_JIS : Shift-JIS, Japanese
# MacRoman : MacRoman
files.Charset = ISO-8859
```

Note: To connect to the sandbox account, change the **netsuite.ws.url** to the appropriate NetSuite sandbox WebService url.