Overnight Automated Tasks

Contents

[GENERAL INFO 1](#_Toc427660123)

[Description 1](#_Toc427660124)

[Structure 1](#_Toc427660125)

[Windows Scheduled Task 1](#_Toc427660126)

[RunTasks.php 1](#_Toc427660127)

[Include Files 1](#_Toc427660128)

[TASKS 2](#_Toc427660129)

[AckLetterContactInfo 2](#_Toc427660130)

[BSDMerchantAccounts 3](#_Toc427660131)

[BSDOutreachPages 4](#_Toc427660132)

[BSDRecurringGifts 5](#_Toc427660133)

[BSDUpLink 6](#_Toc427660134)

[Classification 7](#_Toc427660135)

[FYSums 8](#_Toc427660136)

[GlobalChangeSchedule 9](#_Toc427660137)

[NewDonorGifts 10](#_Toc427660138)

[QuerySchedule 11](#_Toc427660139)

[REActionTrack 12](#_Toc427660140)

[WebImports 13](#_Toc427660141)

[ZDAutoTask 14](#_Toc427660142)

# GENERAL INFO

## Description

The overnight tasks run each night to do various automated cleanup or synchronization tasks. The whole process is kicked off from a windows scheduled task that calls a php script name RunTasks.php (found in Dropbox/Apps/Scheduled). This script then calls other php scripts which run the specific tasks. There are several different kinds of tasks but one of the most common structures is for the script to generate an import file (through a SQL call or web API call) and then import that file through Import-O-Matic (IOM) using the command line module. Each of these tasks have their own IOM profile which processes the import file. These tasks are critical to many functions of the PIH devo ops team and should be understood and maintained. The process is built so that old tasks can be easily updated and new tasks can be easily added.

## Structure

The tasks and their files are kept in Dropbox/Apps/Scheduled and then in their own sub-folders from there. For example the Classification task is in Dropbox/Apps/Scheduled/Classification. Inside the main sub-folder there are sometimes other sub-folders which hold the import files or downloaded data (usually in /files). The task is one file (ex. SCH\_ClassificationUpadte.php) which holds all of the php code for the task. There are variations on the typical task which should be clear from the folder structure and from the script itself. The log files in the subfolders are obsolete because now all of the tasks are run through RunTasks.php which creates its own log. Each task used to be run by its own windows scheduled task until I reorganized everything.

## Windows Scheduled Task

In Beyond Nines there is a scheduled task set up to run RunTasks.php. At 8PM each night the task runs and calls RunTasks.php through the CMD command and pushes the output to Dropbox/Apps/Scheduled/log.txt. This log file is where you look to see how the tasks run the night before if something goes wrong. The system will also send an email to the admin if there is a critical error in any of the IOM calls (like an import file is missing or the password is wrong). If there are just errors in the imports then they will only show up in the log file as many times errors are expected. The scheduled task is run under the SVC user account set up by Beyond Nines. The password is P@ssw0rd which you need to use to modify the task or create new ones.

## RunTasks.php

All of the tasks are run from this one script. It was used to consolidate all of the windows scheduled tasks which are hard to manage. It holds a list of all of the tasks and runs them in order. It can also be used to run a specific task as a one-off task with command line arguments. To run a specific task (or all tasks) from the command line you need to open up a CMD window in the Dropbox/Apps/Scheduled folder and type "php RunTasks.php -t <task name>". You can do RunTasks.php -l to get a list of tasks and -a will run all of them (which is what the windows scheduled task does). Note that you need admin rights to actually have the task run IOM otherwise that part will fail. Some of these tasks have a flag set to "cleanup files" which means that any output files more than 6 months old are automatically deleted to save space.

## Include Files

I have built some library files in PHP to help with running these tasks.

### REDB.php

Helper functions to log into the RE Database (the Reports database actually) and run SQL queries.

### IOMImport.php

Helper functions to run IOM from the command line under the user IMOImports with the password hard coded.

### wspFunctions.php

Some random helper functions

### ZDAPI.php

Allows the use of the Zendesk API

### bsd-api-php-master

For connecting to the BSD API (written by BSD but modified by me)

### BSDToolsPortal.php

Uses CURL to access the BSDTools internal pages

# TASKS

## AckLetterContactInfo

### Path

AckLetterContactInfo/AckLetterContactInfoImport.php

### Summary

Sets the RM Contact gift attribute on gifts from donors with a Relationship Manager.

### Purpose

Because we need to have the RM contact info in the Ack letter we needed a gift level tag with RM contact information so we made this system to add a particular alias from an RM record to the gift of a portfolio donor.

### Description

There is an RE Query that finds gifts by donors with an RM that do not already have the attribute and also the relevant contact information (stored in an Alias of the solicitor record) for the RM. Using RE Queue the query is output to a .csv file and the script then calls IOM to import the file and make the changes. This is a crappy system but it was the best that we could do. With the new VBA based Acknowledgement system we should ideally be able to get rid of this process and pull the contact info during the import but that has not been implemented yet.

### Configuration

No configuration options.

### Errors

There should be no errors when this is run.

### Troubleshooting

The most likely error is that the import file is not found because RE Queue did not run. RE Queue puts a file in /files called SCHEDULED\_NoAckLetterContactInfo.csv. Once it is processed by IOM it is renamed so that the next day it can get replaced. If that file is not there make sure that the REQueue process it running in Beyond Nines (just ask support@beyondnnines.com).

Note that this process will not update gifts that already have the contact attribute set.

### Dependencies

IOMImport.php  
REQueue

## BSDMerchantAccounts

### Path

BSDMerchantAccounts/get\_merchant\_accounts.php

### Summary

Gets daily reconciliation numbers from BSD.

### Purpose

Stores daily BSD revenue numbers in the database split by Merchant Account in order to populate the reconciliation report.

### Description

Calls out to BSD Tools using CURL to get daily reconciliation data used in the reconciliation report. It gets 7 days of data for each Merchant Account and adds or updates that data in a table in the Reports database (BSD\_RECON). This allows time for credits and adjustments to be correctly updated in BSD. The table data is then used in the Reconciliation report to check against RE.

### Configuration

If you need to add a new Merchant Account or change the number of days being downloaded you can do that in the PHP script.

### Errors

There should be no errors.

### Troubleshooting

Since this downloads all transactions in BSD for the last 7 days (including “today”) at 8pm it will not have the complete amount for “today” until the next day. So only reconcile starting with “yesterday” back.

### Dependencies

IOMImport.php   
REDB.php  
BSDToolsPortal.php

## BSDOutreachPages

### Path

BSDOutreachPages/get\_outreach\_pages.php

### Summary

Gets Engage outreach page information from BSD.

### Purpose

Adds/Updates Engage outreach page listings in the Reports database to be used by the outreach page Outreach Pages webapp. Also creates packages in RE as needed.

### Description

Calls out to BSD using CURL to get information about the Engage outreach pages. It then adds the pages to a table in the Reports database and creates packages in RE based on the ID of the page (using IOM) if necessary. The packages created in RE have the same ID as the BSD Page ID and the description for the package is the page url. That package description is used by the BSD gift import to correctly assign packages to gifts made through BSD to the engage pages.

The table in the database is used by the Outreach Page web app to look up engage package information based on information related to the page. This is used mostly when the gift processors get an offline gift that needs to be linked to an Engage page/package.

The code to import the packages is actually in ImportPackages.php.

### Configuration

No configuration options.

### Errors

There should be no errors when this is run

### Troubleshooting

### Dependencies

IOMImport.php   
REDB.php  
BSDToolsPortal.php  
wspFunctions.php

## BSDRecurringGifts

### Path

BSDRecurringGifts/get\_recurring.php

### Summary

Creates the recurring gift in RE that represents the recurring series in BSD.

### Purpose

Creates a recurring gift type gift in RE that holds meta information about a recurring gift series in BSD.

### Description

The task uses CURL to make an internal call to BSD Tools to download data about all of the recurring gifts. It then imports each recurring gift into RE using IOM to update information on the recurring gifts as needed. These recurring gifts show the current state of that gift (active/cancelled) and a historical list with attributes of previous updates to the gift. It also sets the constituent attribute Online Recurring Donor based on whether the record has any active recurring donations or not.

### Configuration

There is no configuration for this task.

### Errors

There are expected errors for this task. They happen when a constituent record is not yet in RE because the gift has not been imported. This task imports all recurring gifts up to the time of download so any recurring gifts created by new donors that day will cause an exception but will be picked up on the following day.

### Troubleshooting

It is not a great system but is the best I could do to get the recurring series data into RE. The data from BSD only has expected next charge date and a possible end date but no indication of whether it was cancelled by the user or if it failed because of a bad CC. I make some inferences based on the dates but it is not 100%. Also the data is not timestamped so the accuracy of the dates is highly dependent on having this import run every day to collect the new information. I have to assume that any changed from RE data happened on the day of import. This data is useful for many applications but always check BSD Tools for specifics if there is a question.

### Dependencies

IOMImport.php   
BSDToolsPortal.php

## BSDUpLink

### Path

BSDUpLink/BSDUpLink.php

### Summary

Uploads RE data to BSD

### Purpose

Syncs data from RE back up to BSD to be used for segmenting emails and tracking RE IDs.

### Description

This sends RE data up to BSD using CURL. It uses SQL (vBSD\_LINK\_CONS) to get recently updated records from the RE database. It then creates a csv file with that data and uses CURL to upload the data to BSD.

CURL allows us to recreate the file upload process that a user can do with BSD Tools. The reason we go through the BSDTools upload interface is that the BSD API doesn't use the same dupe matching rules and doesn't allow uploading of custom fields along with other data. The upload happens in two stages: first the file is sent; then it is mapped. Because of this the CURL calls need to capture the upload GUID sent back from the first call to send it to BSD for the second call.

The data currently sent is :

* email
* first name
  + Note this is set to be “Nickname” if that exists in RE, otherwise first name
* last name
* Classification
* ext\_id
* Solicit Code Exclusions
* RequestsNoEmail
* Deceased
* VIP
  + This is set if they (or their spouse) are LG or part of the board or have one of the special attention attributes.
* REInterests
* LastREUpdate

### Configuration

The SQL is stored in the view vBSD\_LINK\_CONS in the Reports database. It uses the Date Last Changed on the constituent record to filter the results. If you want to add data to the upload you have to change this view to return the correct column as well as add the parsing and column header in the PHP file. You also need to add the new column to the BSD upload mapping.

The mapping of the upload columns is stored in the php script but is not very obvious because it uses the BSD field (numeric) ids which we cannot see normally. The way to get the mapping correct is to do an upload with BSD Tools and use the browser developer console to sniff the POST data from the upload.

### Errors

There should be no errors when this is run

### Troubleshooting

Because this filters based on the constituent date last changed it might not pick up changes in things like addresses or gifts so check that. Also the date last changed is calculated based on the date in the name of the previous upload file so make sure that the upload folder only has correctly formatted file names in it (no Copy of blah.csv).

### Dependencies

IOMImport.php   
REDB.php  
BSDToolsPortal.php

## Classification

### Path

Classification/SCH\_ClassificationUpdate.php

### Summary

Sets the RE Prospect Classification for all records in RE.

### Purpose

Used to recalculate and set the Classification of all records in RE based on past giving and solicitor assignments.

### Description

Uses a SQL stored procedure (wsp\_ClassificationOnlyUpdate) to calculate the Classification of each record and returns only the records where the calculated classification is different than the current classification. It then creates an import file of the changes and calls IOM to make the changes. The IOM profile also creates an attribute on any records that are changed that shows the previous and new classification as well as the date it was changed. The stored procedure is fairly complicated as each classification has its own set of criteria that cascades down. The process specifically ignores any records with the Foundation Relations or Public Sector classification (leaving them as they are no matter what the calculated classification is). It then sets all records with an RM to be Leadership Gifts (and removes any that do not have an RM). It then uses giving history to set Mid Level and Annual Giving Classifications. Lapsed and Non-Donor are then calculated if they did not fall into any of the previous categories. This is one of the most critical tasks as many other processes depend on classification for grouping.

### Configuration

The specifics of the calculations are found in the stored procedure (wsp\_ClassificationOnlyUpdate) in the Reports database. If the calculations need to change they need to be changed using SQL in the stored procedure.

### Errors

There should be no errors when this is run

### Troubleshooting

If you get an error it will most likely be that the record being updated is open or otherwise not updateable for some reason. Since it runs each night and does the same calculation it should try and update that record again the next night.

The import file contains the previous classification, calculated classification, and the totals that were used to make the calculations. This is sometimes helpful in tracking down why someone was changed. Remember that the totals include soft credits which can be confusing sometimes.

### Dependencies

IOMImport.php  
REDB.php

## FYSums

### Path

FYSumAtts/FYSumAtts.php

### Summary

Sets the FYXX Giving Total attributes on constituent records.

### Purpose

Updates the various giving total attributes on constituent records nightly in order to keep them current.

### Description

Uses the stored procedure wsp\_SumsUpdate (which has the same calculations as the Classification procedure) to output the FY giving totals for the last 3 FY plus the totals for the two last 12 month periods and the current FYTD. This makes building queries and exports easier since you do not have to build summaries for these totals. The totals are generally meant for IG because they include soft credits. The stored procedure calculates the totals for everyone and compares against the existing attribute to return the records that need to be updated. A .csv file is generated with the updates and IOM is called to do the import.

### Configuration

If changes are made to wsp\_ClassificationOnlyUpdate for calculating giving totals then they should also be reflected in wsp\_SumsUpdate and vice-versa.

### Errors

There should be no errors when this is run

### Troubleshooting

If you get an error it will most likely be that the record being updated is open or otherwise not updateable for some reason. Since it runs each night and does the same calculation it should try and update that record again the next night.

### Dependencies

IOMImport.php   
REDB.php

## GlobalChangeSchedule

### Path

GlobalChangeSchedule/GlobalChangeScedule.php

### Summary

Runs a list of Global Changes in RE.

### Purpose

Used to do basic coding cleanup like changing gifts under $10 to automatically be marked as Do Not Acknowledge.

### Description

The script calls IOM to "import" GlobalChangeSchedule.csv which holds a list of all of the global changes to run (identified by system ID). The profile in IOM will then read through each row in the file and run the listed RE Global Change. Note that the global change will run on the saved query so need to be able to define a query that captures which records to change without input from the user.

### Configuration

To add or remove a Global Change from the list simply modify the GlobalChangeSchedule.csv file by adding or removing a row with the relevant information.

### Errors

There should be no errors when this is run

### Troubleshooting

If you get an error it will most likely be that the ID of the Global Change is incorrect (either it doesn’t exist anymore or was changed). Make sure that the ID in GlobalChangeSchedule.csv corresponds to the correct Global Change.

It might also be possible to get an error if the Query referenced by the Global Change does not exist anymore.

### Dependencies

GlobalChangeSchedule/GlobalChangeSchedule.csv

IOMImport.php

## NewDonorGifts

### Path

NewDonorGifts/NewDonorGiftImport.php

### Summary

Sets the NewDonorGift attribute in RE.

### Purpose

The NewDonorGift attribute is used in the Acknowledgement process to generate the New Donor letter.

### Description

There is an RE Query that finds first gifts on a records that do not already have the New Donor attribute on any gift. The query is run through RE Queue to create an output csv file. The script runs that csv file through IOM to set the attribute on those records. It counts soft credits when calculating a first gift. The attribute is then used in the Acknowledgement process to generate a New Donor letter.

The reason that we can’t use a global change for this is that you need to use a constituent query to find out if any other gift has the New Donor attribute yet but you want to update on the first gift.

### Configuration

No configuration options.

### Errors

There should be no errors when this is run

### Troubleshooting

The most likely error is that the import file is not found because RE Queue did not run. RE Queue puts a file in /files called “Scheduled New Donor Gifts.csv”. Once it is processed by IOM it is renamed so that the next day it can get replaced. If that file is not there make sure that the REQueue process is running in Beyond Nines (just ask support@beyondnnines.com).

### Dependencies

IOMImport.php  
REQueue

## QuerySchedule

### Path

QuerySchedule/QuerySchedule.php

### Summary

Runs a list of queries from RE and sends the results out as an email to specified addresses.

### Purpose

Used to send basic “reports” or alerts through email. For example it is used to send out an alert to Relationship Managers when one of their donors gives a gift.

### Description

The script calls IOM to "import" QuerySchedule.csv which holds a list of the queries to run (by query name), who to send the results to, and how often to run it. The IOM Profile is set up to go through the file row by row to run the specified query, compose the email, and then send it out to the specified addresses.

### Configuration

The queries have a couple of options in the .csv file. You can choose to send an email regardless of the number of results or only send an email if there are results from the query (basically Summary vs. Alert). You can also choose the frequency with which the queries are run (daily, weekly, or monthly) which is checked by the IOM Profile before running that row. To add, edit, or remove a query from the run simply modify QuerySchedule.csv.

### Errors

There should be no errors when this is run

### Troubleshooting

If you get an error it will most likely be that some data in QuerySchedule.csv is wrong. The Query might not exist (could have been deleted or renamed). The email addresses might be wrong (should be a comma delimited list). The frequency column might have a wrong value (yes/no for daily, Weekday spelled out for Weekly, and numeric day of the month for Month so avoid 29+ because of February).

### Dependencies

QuerySchedule/QuerySchedule.csv

IOMImport.php

## REActionTrack

### Path

REActionTrack/REActionTrackService.php

### Summary

Gets ticket information from Zendesk and creates actions in RE.

### Purpose

Automatically generates actions in RE based on ticket settings in Zendesk so that users do not have to manually update RE.

### Description

This script calls Zendesk using their API to pull tickets marked with the "Track in RE" checkbox (through the Zendesk plugin REActionTrackApp). It then builds an import file from those tickets and calls IOM to import them. The tickets are imported as actions on the constituent record (which is indicated by the ID passed from Zendesk). The Action Type, date, notepad type, notepad description, and notes are all automatically generated from either Zendesk data (topic) or hard coded in the PHP script/IOM Profile.

The action is overwritten every night so that it captures the latest data. The action is identified by the RE action notepad title which is the id of the ticket from Zendesk. This means that any changes to the action from inside RE are overwritten each night. The tickets are pulled as long as they are not marked "Closed" which is the status given after being "Solved" for 7 days.

### Configuration

There are no configuration options.

### Errors

There is sometimes an error when compiling the ticket information that says a Zendesk user cannot be found. I think this happens when a new user is added to Zendesk but I am not 100% sure. Either way it doesn’t really change anything (other than having an ID instead of a name for the author of the comment) so you can ignore it.

### Troubleshooting

The main issue with this would be if the Zendesk API changed which will need to be updated in the library file. It is also possible that the format of the automated emails changes which would be updated in the PHP script.

### Dependencies

IOMImport.php   
ZDAPI.php

## WebImports

### Path

WebImports/ProcessImports.php

### Summary

Processes the approved imports from the WebImports app.

### Purpose

This looks at the import schedule from the WebImports app and runs the import file/IOM Profile combo for the rows with the status of “Approved”.

### Description

This is the task that processes Approved web imports (created through the WebImport app). It loops through the import schedule and if an import has the status of Approved then it kicks off the relevant IOM Import. This is also where the success and failure emails are sent from.

### Configuration

The only configuration here might be the email addresses of the “admins” that receive all of the internal notifications about the Imports. They are listed in the PHP file.

### Errors

If there are errors from the imports they are emailed out. If there is a critical error (like a missing file) then only the admins are emailed.

### Troubleshooting

Just guess ☺

### Dependencies

IOMImport.php   
webapps/REInfo/IOMWebClasses.php  
PHPMailer-master\class.phpmailer.php

## ZDAutoTask

### Path

ZDAutoTask/ZDAutoTask.php

### Summary

Processes automated requests for DNM from Zendesk

### Purpose

This pulls down and imports automated DNM requests from Zendesk in order to avoid having to do them by hand.

### Description

Using the Organization records in Zendesk we automatically put emails from automated DNM services (currently Paper Karma and Catalog Choice) into the group AutoTask. There is a trigger (Reply to Auto DNM Requests [AutoTask]) in Zendesk that automatically sets the Topic and Status for tickets in the AutoTask group. It also automatically replies to the requester saying everything is taken care of.

This script calls Zendesk each night through the API to pull all of the tickets tagged by the trigger and generates an import file by parsing out the body of the email. IOM is then called with the import file to set the DNM solicit code and the No Share attribute.

### Configuration

If you want to add a new auto-DNM vendor then you will need to create the organization in Zendesk to assign the group and you will need to update the PHP script to parse out the address/name from the body of the email.

### Errors

Errors are expected in this run because people are often not found in RE (usually acquisition names). If the record is not found then we keep the exception logs and send them to Hillary manually to be added to the "kill list".

I built a small script (CombineCSV.php) that will compile all of the exception files into one file to send to Hillary. It will automatically compile only the files created since the last time the script was run. This script is run by opening a command window and typing “php CombineCSV.php”. The compiled files are stored in the folder called notinre.

### Troubleshooting

The main issue with this would be if the Zendesk API changed which will need to be updated in the library file. It is also possible that the format of the automated emails changes which would be updated in the PHP script.

### Dependencies

IOMImport.php   
ZDAPI.php