How Evergreen Notices Work

Rogan Hamby, MLIS

He/Him

Data and Project Manager

rogan.hamby@equinoxOLI.org

Bluesky: @rogan.hamby.bsky.social

Steve Callender

He/Him

Help Desk Manager

Equinox Open Library Initiative

stevecallender@equinoxoli.org

How Notices Work

This is all about notices. We are not going to get into everything that can be done with action triggers. However, once you learn notices a lot will be transferable to other kinds of action triggers.

This could easily be four hours so we are going to skim over a few things.

Fundamental Parts of an Action Trigger

Hook
Validator
Reactor
Delay
Max Delay
Opt In Setting
Template
Granularity
Group Field
Messages

The Very Very Abstract

Step 1. Have a hook that says "this is the condition that starts the notice."

Step 2. Have a reactor that responds and validates.

Step 3. Have the action trigger gather information and put it somewhere.

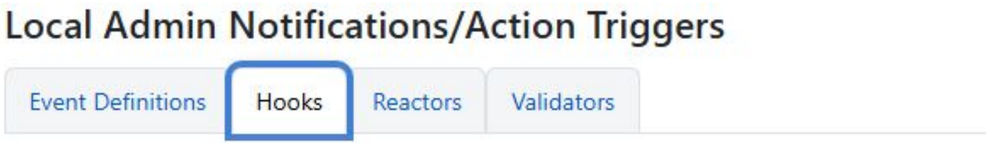
Step 4. Send notices.

Warning: We will talk about exceptions.

so let's start with hooks …

Hooks

Passive versus active triggers



Local Admin Notifications/Action Triggers

Non-Passive triggers will have their Passive setting set to No
Passive triggers will have their Passive setting set to Yes

An active trigger (non-passive) is one that automatically fire off based on an action done in Evergreen. They are built straight into the code.

Hooks

Some Hooks That Are Non-Passive

- checkin - An Item Was Checked In
- checkout - An Item Was Checked Out
- damaged - An Item Was Marked As Damaged
- hold.captured - An Item Was Captured For A Hold

Some Hooks That Are Passive

- au.expired - A Patron Account Is Past Their Expire Date
- checkout.due - An Item Is Overdue
- hold_request.shelf_expires_soon - A Hold Is About To Expire

Reactor

Notice reactors:

- ProcessTemplate - process through Template Toolkit (usually XML)
- SendEmail - ProcessTemplate + Send Email
- SendSMS - Process Template + Send Email Through SMS Gateway
- NOOP_True

Validator

Validators make sure that notices are still valid when they are sent. When they are sent is an important detail.

Delays & Max Delay

Little known fact: holds are picked up at 88 miles per hour.

So, sometimes we want to delay when the notice is sent.

Opt-In

A per user way to validate if a user should receive a notice with a user setting.

Common ones:

circ.send_email_checkout_receipts

opac.default_sms_notify

Messages

Obligatory cat picture.

Messages are notices that stay inside.

Templates

Templates are what create your notice content, whether it is plain text, HTML, XML or whatever else you want to create.

Templates

```
 Dear [% user.first_given_name %] [% user.family_name %],

Thank you for being a great library user!
```

vs

```
<first_name>[% helpers.escape_xml(user.first_given_name) %]</first_name>
```

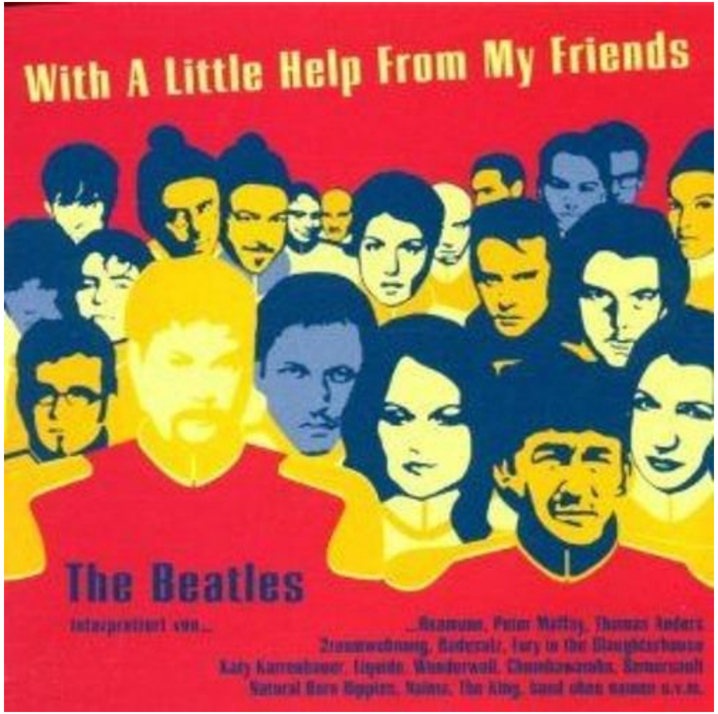
Templates

The two helper functions you must know:

```
helpers.get_org_setting(target.0.circ_lib.id,
        'lib.info_url')

helpers.escape_xml(wholenamenocap.trim)
```

And many more.


With A Little Help From My Friends
The Beatles
interpretiert von...
...Peter Maffay, Thomas Anders
Fury in the Slaughterhouse
Katy Karrenbauer, Liquido, Wonderwall, Chumbawamba
Natural Born Hippies, The King u.v.m.

Templates

You can do logic too:

```
 [% IF circ.usr.first_given_name == 'Rogan' %]
     Its cool man, turn it back in when it makes you happy.
[% ELSE %]
     Your item is overdue, the library police are en route now.
[% END %]
```

https://template-toolkit.org/

Templates

Oh yeah, a little trick :

```
[% user.family_name %]
```

vs

```
[%- user.family_name -%]
```

Parameters

Values you pass in.

editor '1' (not a notice)

xml_notice_last_processed_id 115761

Templates - Data and Environments

```
[%- SET user = target.usr -%]

[% user.first_given_name %]  = target.usr.first_given_name

[% user.card.barcode %]
```

Data and Environments

```
[%- SET user = target.usr -%]


[% user.first_given_name %]  = target.usr.first_given_name

[% user.card.barcode %]
```

Data and Environments

circ_lib.billing_address

circ_lib.mailing_address

target_copy.call_number

target_copy.location

usr.billing_address

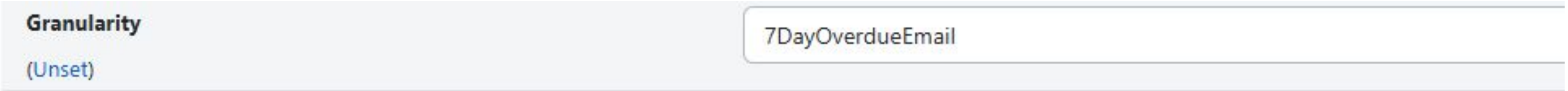
usr.card

usr.mailing_address

Group Fields

Cron Stuff

Granularity is a method for scheduling triggers to run at the times you want them to run.



Granularity

(Unset)

7DayOverdueEmail

Maybe you want to…

- Send Emails/SMS messages only during business hours
- Organize notices into categories to keep organized

Cron Stuff

Cron examples,

Standard Granularities

```
*/2 * * * * . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --run-pending
0 * * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity hourly --granularity-only
5 3 * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity daily --granularity-only
10 3 * * 1-5 . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity weekdays --granularity-only
15 3 * * 0   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity weekly --granularity-only
20 3 1 * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity monthly --granularity-only
25 3 1 1 *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity yearly --granularity-only
```

Cron Stuff

Cron examples,

Different Granularities

```
0 2 * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity OverDues --granularity-only
0 3 * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity PreDues --granularity-only
0 4 * * * . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity LostNotices --granularity-only
15 10-17 * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending --granularity SMSNotices --granularity-only
```

Cron Stuff

Action_trigger_runner.pl parameters,

- --process-hooks - Follow the rules set for the hook to create the notice up to the pending state.
- –run-pending - Process all the events in a “Pending” state to completion.
- –granularity <granularity> - Process only those hooks using this granularity.
- –granularity-only - Only run the pending triggers using provided granularity
- –lock-file <name> - Provide a specific lock file to prevent duplicate running.
- –custom-filters <name> - Provide a custom filter to use for hook
- –verbose and –debug-stdout - Used for debugging problems.

Filter Example

When to use filters,

- Setting certain items to lost earlier than others
- Creating a notice for a specific circulation modifier
- Creating a notice for a specific profile group
- Modifying existing hooks to do more (or less)

Filter Example

Sample filter file can be found in ./Open-ILS/examples/action_trigger_filters.json.example

Example stock filter for the hook checkout.due (JSON)

```
"checkout.due" :
  { "context_org" : "circ_lib",
   "filter"     :
      { "checkin_time"  : null,
       "-or"          :
          [ { "stop_fines"  : ["MAXFINES"] },
           { "stop_fines"  : null }
          ]
      }
  },
```

Filter Example

Add LONGOVERDUES and CLAIMSRETURNED to the checkout.due hook

Change,

```
	[ { "stop_fines"  : ["MAXFINES"] },
```

To

```
	{ "stop_fines" : ["MAXFINES", "LONGOVERDUE", "CLAIMSRETURNED "] },
```

```
5 2 * * *   . ~/.bashrc && /openils/bin/action_trigger_runner.pl --osrf-config $SRF_CORE --process-hooks --run-pending
--granularity 7DayOverdue --granularity-only /openils/a_t_filters/custom_overdue_filter.json
```

Filter Example

Filter By Circ Modifier

```
{
  "checkout.due" : {
    "context_org" : "circ_lib",
    "filter" : {
      "checkin_time" : null,
      "-or" : [
        { "stop_fines" : ["MAXFINES", "LONGOVERDUE"] },
        { "stop_fines" : null }
      ],
      "-and" : [
        { "-exists" : {
         "select" : {"acp" : ["id"]},
         "from"   : "acp",
         "where"  : {
           "circ_modifier" : ["Best Seller", "New Fiction", "New Nonfiction", "Periodical", "DVD", "Video" ],
           "id" : { "=" : {"+circ" : "target_copy"} }
          }
         }
        }]
    }
  }
}
```

Filter Example

Filter by User Profile

```
{
    "checkout.due": {
        "context_org": "circ_lib",
        "filter": {
            "checkin_time": null,
            "-or": [
                {"stop_fines": ["MAXFINES", "LONGOVERDUE", ]},
                {"stop_fines": null}
            ],
            "-and": [
                {
                    "-exists": {
                        "select": {"au": ["id"]},
                        "from": "au",
                        "where": {
                            "profile": [34, 35, 36],
                            "id": {"=": {"+circ": "usr"}}
                        }
                    }
                }
            ]
        }
    }
}
```

Filter Example

Filter by Shelving Location

```
{
    "checkout.due" : {
        "context_org" : "circ_lib",
        "filter" : {
            "checkin_time" : null,
            "-or" : [
                { "stop_fines" : ["MAXFINES", "LONGOVERDUE"] },
                { "stop_fines" : null }
            ],
            "-and" : [
                { "-exists" : {
                  "select" : {"acp" : ["id"]},
                  "from"   : "acp",
                  "where"  : {
                      "location" : ["6537"],
                      "id" : { "=" : {"+circ" : "target_copy"} }
                    }
                  }
                }]
        }
    }
}
```

Filter Example

A filter for circ modifier, profile, and circulation library

```
{
    "checkout.due" : {
        "context_org" : "circ_lib",
        "filter" : {
            "checkin_time" : null,
            "-or" : [
                { "stop_fines" : ["MAXFINES", "LONGOVERDUE", "CLAIMSRETURNED"] },
                { "stop_fines" : null }
            ],
            "-and" : [
                { "-exists" : {
                  "select" : {"acp" : ["id"]},
                  "from"   : "acp",
                  "where"  : {
                      "circ_modifier" : ["Best Seller", "New Fiction", "New Nonfiction", "Periodical", "DVD", "DVD 2", "Video", "Video 2"],
                      "id" : { "=" : {"+circ" : "target_copy"} },
                                                                          "+circ" : { "circ_lib" : [154, 155, 156] }
                    }
                  }
                }]
```

Filter Example

Continued

```
                     "-and" : [
        { "-exists" : {
         "select" : {"au" : ["id"]},
         "from"   : "au",
         "where"  : {
           "profile" : [34],
           "id" : { "=" : {"+circ" : "usr"} }
          }
         }
        }]
    }
  }
}
```

Filter Example

Custom filters are good for

- Fine tuning a hook for specific scenario / patron group / item type

Not good for

- Multiple versions of the same notice

Filter Example

IF statements are great for having multiple text for different libraries / profiles

```
[%- IF (user.profile == 6)  or  (user.profile == 7) %]
Take your time returning your item!
[%- ELSE %]
Please return your item immediately!
[%- END %]
```

Or

```
[%- IF (lib.shortname == 'LIBRARYA') or (lib.shortname == 'LIBRARYB') or (lib.shortname == 'LIBRARYC') %]
Thank you for using our library!
[%- ELSE %]
Please return your item immediately!
[%- END %]
```

Troubleshooting

```
WITH source AS (
    SELECT ate.id AS ate_id, ated.active, ated.name, ated.id AS ated_id, ated.granularity,
        ated.reactor, ate.state, ate.template_output, LENGTH(COALESCE(ateo.data,'')) AS datalength
    FROM action_trigger.event_definition ated
    LEFT JOIN action_trigger.event ate ON ate.event_def = ated.id
    LEFT JOIN action_trigger.event_output ateo ON ateo.id = ate.template_output
    JOIN actor.org_unit aou ON aou.id = ated.owner
    WHERE aou.id IN (SELECT id FROM actor.org_unit_descendants((SELECT id FROM actor.org_unit WHERE shortname = 'FOO')))
)
SELECT COUNT(ate_id) AS events, ated_id, active, LEFT(name,25),
    granularity, reactor, state, COUNT(template_output) AS template_outputs,
    CASE WHEN datalength = 0 THEN 'zed' WHEN datalength BETWEEN 1 AND 500 THEN '<500' WHEN datalength > 500 THEN
'>500' END AS "length"
FROM source
GROUP BY 2, 3, 4, 5, 6, 7, 9
ORDER BY 6, 4;
```

Troubleshooting

```
select data from action_trigger.event_output where id in
	(select template_output from action_trigger.event where event_def = 136)
order by id desc;


select data from action_trigger.event_output where id in
	(select error_output from action_trigger.event where event_def = 136)
order by id desc;
```

Bonus!

But what about print notices!

Emails get emails. SMS gets emailed, unless you have an external service but what happenes to print?

SQL Query -> XML to XSLT with XSL -> PDF

Questions