Plugin Activity Stream

The activity stream is a Tiki12 feature allowing to create social network activity streams within Tiki. The feature depends on the Search and List from Unified Index and Elasticsearch is highly recommended to obtain decent performance.

In order to create activity streams, you will need to define what the important events are in your system. Events like "tracker item created" or "wiki page modified" will rarely make sense to your users looking at an activity stream. Instead, they may be interested when photos are posted by their friends. No matter where you decide to store pictures, whether they are in file galleries, or as tracker items with meta-data, the desired event for the end user remains the same.

The activity stream feature allows to intercept system events, filter them and trigger new events. These new events can be recorded and indexed, which will allow them to be displayed in an activity stream.

Video Introduction

Presentation of the activity stream feature during the TikiFestBootstrap in 2013 by Nelson Ko:

By Patrick Proulx

See also: Tiki Activity Stream Presentation

Parameters

Create a social network activity stream.
Introduced in Tiki 12.
Go to the source code
Preferences required: wikiplugin_activitystream, feature_search

Parameters	Accepted Values	Description	Default	Since
(body of plugin)		List configuration information		
`auto`	0 1	Automatically load next page of results when scrolling down.	0	12.0

Basic events

You can choose to enable recording of basic events through preferences in the community administration panel. This is primarily in place to quickly test the functionality.

Displaying Activity Streams

Activity streams are rendered as a wiki plugin. The base plugin will render all activities in the system (as long as permissions allow).

```
{ACTIVITYSTREAM(auto="1")} {ACTIVITYSTREAM}
```

You can create a simple stream for the user containing only his own activities.

```
{ACTIVITYSTREAM()} {filter personalize=self} {ACTIVITYSTREAM}
```

Auto-scrolling can be enabled on the plugin with the `auto` argument. It will cause the next set of results to be appended as the user scrolls past the end of the list. Note that your design will need to consider this behavior as the bottom of the page will effectively be unreachable.

Custom activities, Activity Rules

On the Admin Control Panel, Community, Social Interaction, Activity stream section (tiki-admin.php?page=community#contentadmin_community-2) you can enable "Custom activities" to create your own set of rules. After enabling it, you will be able to create rules under the "Activity Rules" link (at the top of the same page).

- **Sample Rule** stores the last received event data in a cache entry, allowing to view the variables available
- **Basic Rule (Record Event)** turns on recording of the activity and makes it available in streams.
- **Tracker Rule** triggers a new event after limiting the results to those from a single tracker. The data made available in the new event can be specified.
- **Advanced Rule** provides a way to use your own more advanced filters based on the Rating Language, detailed with the Calculations.

Once the basic rules are in place, they can be converted to advanced mode to further improve them.

Example

In this example we want to create a new event called *mysite.bugtracker.save* which only triggers if something in tracker 5 is created/updated. If I use the **Tracker Rule** UI it will generate this for me:

```
(if (equals args.trackerId 5) (event-trigger mysite.bugtracker.save (map (user args.user) (type args.type) (object args.object) (aggregate args.aggregate) )))
```

If I then want to extend this to filter only for situations where the value of the version field is 15, I would have to convert it to an Advanced rule and them modify it to do as follows. In the following example, I have also mapped the version field over to "version" in the new mysite.bugtracker.save for being used in templates etc.

```
(if and((equals args.trackerId 5) (equals 15 args.values_by_permname.version)) (event-trigger mysite.bugtracker.save (map (user args.user) (type args.type) (object args.object) (version args.values_by_permname.version) (aggregate args.aggregate) )))
```

After I set **Basic Rule (Record Event)** for *mysite.bugtracker.save*, that event will be recorded. The filter I can use is then: `{filter field="event" content="mysite.bugtracker.save"}`.

Filtering events

Multiple entries for the "personalize" filter can be separated by commas. Valid options are:

Option	Description
`self`	Filter entries in which the current user is the activity source
`groups`	Filter entries in which the current user shares groups with the activity source user
`follow`	Filter entries in which the user decided to friend/follow the activity source user through the Social Interaction feature

Filtering the information displayed on a wiki page

You can also filter the activity stream using the same syntax as filters in PluginList. For example, you could filter on *categories* to limit to a specific project. Note that you can only filter on the fields that are available (which is not the same as the fields of the object that triggered the activity).

Activity streams are basically a special kind of LIST of "indexed activities". All activities have global fields such as "categories", so if you want to filter by categories you can simply use the filter like in the LIST plugin as follows and it will work: `{filter field="categories" content="265 OR 187"}`

However, if you want to filter on the information in the activity (i.e. things about the object that it relates to etc...) you would need to determine the field names of the activity which would differ depending on which activity is it. To investigate what fields are in an activity, you would have to create a **Sample Rule** in the Activity Rules panel in the Community Control Panel: `tiki-admin.php?page=community`. For example, if you want to see what fields are available in *tiki.trackeritem.save* event, you would click on Activity Rules, and then setup a sample rule for *tiki.trackeritem.save* (this event covers both *tiki.trackeritem.create* and *tiki.trackeritem.update*). Then just edit any tracker item, and then click Edit for the rule again. It will show you the sample (i.e. the fields and values of the recently triggered event you just triggered). You will notice that there is a field "trackerId" so if you wanted to filter by tracker ID equals 5 for , you could just do: `{filter field="trackerId" content="5"}`

If you want then to record this event, you would need to use **Basic Rule (Record Event)** or if it is one of the basic events that have a preference to turn on recording, you can turn it on simply through preferences in the community administration panel. Or if you want to setup a custom event that is created based on a filtered version of this event you will use **Advanced Rule**. **Tracker Rule** is a convenience UI (compared to using an advanced rule) developed to create a simple filter of Tracker streams by tracker.

The event filtering logic is built using the Rating Language, detailed with the Calculations. Through the community administration panel, the rules can be managed. For advanced usage, the programming can be written directly. This is a new feature and over time, it is hoped that convenience UI such as the **Tracker Rule** will be created/improved to make it easier to setup custom rules without having to write Rating Language.

Rendering Activities

Within the activity stream, each activity will be rendered using a custom template. The template will be named according to the event name. If you create an event named *mysite.photo.upload*, the template file used will be `templates/activity/mysite.photo.upload.tpl`.

Grouping Similar Activities

It may occur frequently that multiple similar events get triggered by the system. For example, successive edits of a wiki page by a single user would trigger one event per edit. If you chose to trigger custom events for those modifications and to record them, the activity stream would become polluted.

The activity stream plugin contains a directive to allow grouping similar activities so that:

1. Jonny modified page X
2. Marc modified page X
3. Jonny modified page Y
4. Nelson modified page X
5. Marc modified page Y

Would become:

1. Jonny, Marc and Nelson modified page X
2. Jonny and Marc modified page Y

To do so, Tiki uses an aggregate hash field as part of the events, which is used to determine which objects should be grouped.

The convention is to use the field *aggregate* to contain the hash, but you could become creative about this.

- The following example explicitly states the default values. Ignoring the parameters on {group} would have no impact.
- Multiple fields can be listed in *collect*, separated by commas
- Using aggregation will break pagination as it reduces the amount of results on a per-page basis. However, this is not an issue for activity streams as pagination is not a typical concept in them.
- The base entry available in the search results is the first one encountered, so the event date is the most recent one.

```
{ACTIVITYSTREAM()} {filter personalize="self"} {group field="aggregate" collect="user"} {ACTIVITYSTREAM}
```

From the template files, the aggregate field will be replaced by an array containing the collected unique values. For example, in the previous example, for the first activity, **$activity.aggregate.user** is an array containing *{'Jonny', 'Marc', 'Nelson'}*, using the uselink modifier on the array converts it to the printed version of *Jonny, Marc and Nelson*.

To create a fixed-size list of *15 Recent Activities* with aggregation, the amount of results obtained can augment the amount of results returned and get the list to be truncated afterwards. The following snippet instructs to obtain 3 times more results than desired. After aggregation, will truncate the list back to 15.

The list may still contain fewer results than desired, but with a reasonable threshold, this should be reduced. *Using this option will disable pagination and auto-scrolling.*

```
{ACTIVITYSTREAM()} {pagination max="15"} {group boost="3"} {ACTIVITYSTREAM}
```

To assist in the creation of custom hash values, the hash function is available in the Rating Language

Related links

- Activity Notifications
- https://www.slideshare.net/patrickproulx/tiki-activity-stream-presentation
- https://tiki.org/forumthread54735-Activity-Stream
- Activity_Stream

Aliases

- PluginActivityStream