TikiSignal

Previously known as TikiEvents

Status/RoadMap

- Where are we?

We have a watch feature that emails users/admin when wiki pages change
We have email notifications that email observers, covering about 5 type of event
We have a tikilog that records most types of event
We have a new events notification framework that covers just 'user_registers' and only from tiki-register.php - as a first proof of concept.

- Where do we want to be?

An events notification framework that covers most of the major events:
1. user registers
2. user preferences have changed
3. user information has changed
4. content published/modified
5 content unpublished/expired
6. user banned
7. user logged in
8. user becomes dormant <-- this one might be difficult to code. needed by things like the active users module.
9. feature enabled
10 feature disabled

- Who is working on what? (Priorities/goals/majors issues/roles)

First step is a proper proof of concept, being worked upon by mdavey

TikiTeam

Who is working here generally? Link UserPage.

- mdavey - coding a proof of concept
- amette - providing support, encouragement and backup

Trackers

- Bugs
- RFEs
- tech support
- patches

Competition and standards

List of other products with similar/interesting/related features.

How to implement?

- SOAP
- XMLRPC
- PHP callback function
- Message service

For the initial work, I have chosen to implement as PHP callback function only.

CVS Doc section

This is where new features being developed and only in CVS are documented. When the CVS becomes RC/official release, the info in the CVS docs is transferred to update the official docs (FeatureXDoc).

possible functions:

notifications.php
register_event_callback(event, callback_type, method, self);
raise_event(event, data[], self);
unregister_callback(event, callback_type, method, self);

...where callback_type is one of:

1. - early_callback - callback before standard callbacks - allows setup of temporary data structures and pre-processing of event data
2. - standard_callback
3. - late_callback - callback after standard callbacks - allows destruction/cleanup of temporary data structures and post-processing of event data

example callback function:

registrationlib.php
callback_function_eventA(raisedBy, data[])

database schema (MySQL):

```
CREATE TABLE `tiki_events` (
  `callback_type` int(1) NOT NULL default '3',
  `order` int(2) NOT NULL default '50',
  `event` varchar(20) NOT NULL default '',
  `object` varchar(200) NOT NULL default '',
  `method` varchar(200) NOT NULL default '',
  PRIMARY KEY  (`callback_type`,`order`)
) TYPE=MyISAM;

INSERT IGNORE INTO tiki_events(`callback_type`,`order`,`event`,`object`,`method`)
    VALUES ('1', '20', 'user_registers', 'registrationlib', 'setup_custom_fields');
INSERT IGNORE INTO tiki_events(`event`,`object`,`method`)
    VALUES ('user_registers', 'registrationlib', 'saveRegistration');
```

```
INSERT IGNORE INTO tiki_events(`callback_type`,`order`,`event`,`object`,`method`)
    VALUES ('5', '20', 'user_registers', 'registrationlib',
'_tikisignal_logslib_user_registers');
INSERT IGNORE INTO tiki_events(`callback_type`,`order`,`event`,`object`,`method`)
    VALUES ('5', '25', 'user_registers', 'registrationlib', 'sendEmail');
INSERT IGNORE INTO tiki_events(`callback_type`,`order`,`event`,`object`,`method`)
    VALUES ('5', '30', 'user_registers', 'registrationlib', '_tikisignal__user_registers');
```

Explanation

Observers register a callback on an event. Two specialised forms of callback allow an observer to be notified immediately before the main list so they can prepare, or after the main list so they can tidy up.

An observable raises an event notification by calling tikiinternaleventlib->raise_event(...). The library then calls all the functions in the pre_event, event and post_event lists in turn:

pseudocode:

```
$result = $this->query("select * from `tiki_events`
        where event like $event order by `callback_type`, `order`");

$continue = true;
while ($continue && $res = $result->fetchRow()) {
        $class = $res['object'];
        $method = $res['method'];
        if ( is_callable(array(get_class($class), $method)) ) {
                $continue = $class->$method($raisedBy, &$data);
        }
}
```

When an observer no longer needs to be notified of an event, it calls notificationslib->unregister_callback(...) with the same arguments it used to request a callback.

Discussion/participation

Where ideas can be exchanged, debated, etc. Interested people can subscribe to the wiki page and/or to these forums as they would a mailing list.

IRC log of July 16th 2005 that started the whole thing

[+]

More discussion

- the "user becomes dormant". In 1.10, there is the timeout and the "I am back" for a user in the tiki logs. The delay is still in the code 5mn. sylvie