How to add support for my tool in Site Stats

Table of Contents

Requirements

Site Stats trunk or >= 2.0 (unreleased) and EntityBroker >= 1.3.5 (present in Sakai 2.6 or trunk) are required for the features demonstrated on this page.

1. How it works

SiteStats consults the EntityProviderManager service to find which tools implement the Statisticable capability from EntityBroker. This capability - implemented by tools - provides the following information to Site Stats:

- Associated sakai tool id
- List of events to be supported in SiteStats
- Localized events description

2. Example

Suppose you have already a basic Entity Provider capability, as shown on the next code listing:

TestToolEntityProvider.java with basic AutoRegisterEntityProvider capability

```
package org.sakaiproject.testtool.logic.entity;

import org.sakaiproject.entitybroker.entityprovider.capabilities.AutoRegisterEntityProvider;

public class TestToolEntityProvider implements AutoRegisterEntityProvider {

    // 'AutoRegisterEntityProvider' capability:
    public final static String               PREFIX                       = "testtool";

    public String getEntityPrefix() {
        return PREFIX;
    }
}
```

Adding the Statisticable capability to your class, will require you to implement 3 new methods. The code belows assumes that a 'Events.properties' message bundle exists and contains the event descriptions.

TestToolEntityProvider.java with Statisticable capability

```
package org.sakaiproject.testtool.logic.entity;

import java.util.HashMap;
import java.util.Locale;
import java.util.Map;
import org.sakaiproject.entitybroker.entityprovider.capabilities.AutoRegisterEntityProvider;
import org.sakaiproject.entitybroker.entityprovider.capabilities.Statisticable;
import org.sakaiproject.util.ResourceLoader;


public class TestToolEntityProvider implements AutoRegisterEntityProvider, Statisticable {

    // 'AutoRegisterEntityProvider' capability: ----------------------------------
    public final static String               PREFIX                        = "testtool";

    public String getEntityPrefix() {
        return PREFIX;
    }

    // 'Statisticable' capability: -----------------------------------------------
    public final static String               TOOL_ID                         = "sakai.testtool";
    public final static String               EVENT_NEW                 = "testtool.new";
    public final static String               EVENT_EDIT                 = "testtool.edit";
    public final static String               EVENT_DELETE                 = "testtool.delete";
    public final static String               EVENT_READ                 = "testtool.read";
    public final static String[]        EVENT_KEYS                =
                                            new String[] {
                                                    EVENT_NEW, EVENT_EDIT, EVENT_DELETE, EVENT_READ
                                            };

   /**
     * Return the associated common tool.id for this tool
     *
     * @return the tool id (example: "sakai.messages")
     */
    public String getAssociatedToolId() {
        return TOOL_ID;
    }

    /**
     * Return an array of all the event keys which should be tracked for statistics
     *
     * @return an array if event keys (example: "message.new" , "message.delete")
     */
    public String[] getEventKeys() {
        return EVENT_KEYS;
    }

    /**
     * OPTIONAL: return null if you do not want to implement this<br/>
     * Return the event key => event name map for a given Locale,
     * allows the author to create human readable i18n names for their event keys
     *
     * @param locale the locale to return the names for
     * @return the map of event key => event name (example: for a 'en' locale: {"message.new","A new message"})
OR null to use the event keys
     */
    public Map<String, String> getEventNames(Locale locale) {
        Map<String, String> localeEventNames = new HashMap<String, String>();
        ResourceLoader msgs = new ResourceLoader("Events");
        msgs.setContextLocale(locale);
        for(int i=0; i<EVENT_KEYS.length; i++) {
                localeEventNames.put(EVENT_KEYS[i], msgs.getString(EVENT_KEYS[i]));
        }
        return localeEventNames;
    }
}
```

After this, you can log events as usual using the EventTrackingService and these events will be automatically supported in SiteStats (using cover for simplicity):

Example of event logging

```
EventTrackingService.post(
  EventTrackingService.newEvent(
      TestToolEntityProvider.EVENT_DELETE,
      item.getReference().toString(),
      true)
);
```

3. Related configuration bits

By default, SiteStats will use the (localized) event descriptions provided by the Statisticable interfaces and, if not found, fallback to local event descriptions provided within the Site Stats bundles. You can change this behavior with the following setting in sakai.properties:

- **checkLocalEventNamesFirst**@org.sakaiproject.sitestats.api.event.EntityBrokerEventRegistry = true