SSES courseweb 4: Profile management

JOHAN FALK
ONS, 2011-06-01 15:22

BLOGG: SSES COURSEWEB

This is a part of the experimental open techincal documentation for the SSES courseweb project. This part explains the profile and user management on the site, including the management of duplicate content.

RELATERAT

- Technical documentation for "SSES courseweb"
- SSES courseweb 2: Site architecture - node types and sections
- SSES courseweb 5: Automatic e-mail notifications
- SSES courseweb 3: Course content and access

The profile management is probably the most complex functionality the SSES courseweb. The functionality includes:

- Functions to import profile data from CSV files, as well as manual creation.
- Functions to recognise e-mail addresses across multiple values and automatically merge profiles.
- Functions to recognise matching first and last name and allow manual merging of profiles.
- Functions to automatically create user accounts based on profile data, *when necessary*.
- Functions to update profile e-mail when user account e-mail is updated.

These functions are described in more detail below.

THE DUPLICATE PROBLEM

The client has problems with duplicates in profiles – which was one of the reasons to move them to a new platform – and this has influenced the profile management quite a bit. The main problem is that there is no clear primary identifier. The client usually gets the profile data in CSV files from universities, meaning that the potential identifier of e-mail addresses is set to the school's own domain ("student1234@thisuniversity.se"). When the same student later on takes a course at another university – or registers his/her private e-mail address at an event – there is nothing except the name that says that two profiles actually represents the same person.

To have a chance to have a persistent identification key, and also send out e-mails correctly, students are encouraged to update their e-mail accounts to their private e-mails rather than their student e-mail accounts. But then, of course, the old e-mail address must be kept in order to identify existing profiles if the school e-mail address should appear in another CSV import.

FÖRFATTARE

JOHAN FALK
DRUPAL TRAINER / DEVELOPER

MER FRÅN JOHAN FALK

- Learn Drupal coding with NodeOne part 6: More arrays
- Learn Drupal coding with NodeOne part 5: Arrays
- Technical documentation for "SSES courseweb"

Redigera

DRUPALBOKEN

Redigera

LEARN DRUPAL

This debacle is solved in the several steps, the first being this:

- Profiles have a field "Primary e-mail address", which should always reflect the currently active e-mail address.
- Profiles also have a multi-value field "Additional e-mail addresses". The rule set "Actions when profiles are saved" copies any new e-mail address to this field every time profiles are updated (which actually means that the field label should be "All e-mail addresses").
- Since student's aren't allowed to edit their own profiles, there is also an action for copying any new e-mail address to their profile (in the Rule set "Actions when user accounts are updated").

IMPORTING PROFILES AND MERGING BY E-MAIL

Import of new profile data is a quite straight-forward process, thanks to the wonderful Feeds module. In the project we have defined a Google spreadsheet as template for import, to make sure that the correct table headers are kept on import and also assure that the CSV format works with the import scripts.

Feeds has very nice functions for updating existing content if a primary key matches, but unfortunately Feeds cannot use a multiple-value field as primary key. This, in combination with the client should be able to merge profiles manually, made us manage duplicates with – you guessed it – a rule set (in combination with Views Bulk Operations).

Rules Bonus Pack has a very useful condition "Check number of results from a view". The SSES site has a view (cw_potential_email_duplicates) that takes two arguments – an e-mail address and a NID – and returns all the profile nodes that contain the given address but *not* the given NID. This condition is used to check profiles for duplicates based on e-mail, and if found another rule set ("Merge two profiles") is called to merge the profiles. (To be able call this action, the rule set uses the action "Use Views to load a node" to get the other half of the duplicate pair.)

This procedure was initially included in the rule set "Actions when new profiles are created", but this made Feeds time out when importing. Instead, it is now included in a VBO (imported_profiles) displayed after an import is finished – along with instructions urging the registrar to have the imported profiles semi-automatically checked for e-mail duplicates.

The procedure is still kind of lengthy – it takes 1–2 seconds for each profile that is merged – but the solution was deemed good enough by the client, who finds this procedure much quicker than the manual work they had to endure previously. (Profiles that *don't* have duplicates are checked rather quickly, despite having a view load on each profile.)

Keeping the reference to import nodes

The client wanted to be able to filter imported profiles by their import occation (meaning import node). Feeds does have a nice Views integration that allows this, but it breaks if you merge two profile nodes – if a new profile is merged into an old one, it should still be possible to find it when filtering by your recent import.

This problem was solved by transforming the relation to the import node to a multiple-value node reference field. The rule set "Actions when new profiles are created" takes any Feeds import reference and adds it to the node reference field – and this data is respected when profiles are merged. Thanks again to Rules Bonus Pack, for providing the action "Get import nid for content".

COMPARING PROFILES AND MERGING BY NAME

The e-mail part of the duplicates were only parts of the problem – the client regularly has

WITH NODEONE

1. Drupal 7 introduction, 15 screencasts
2. Learn Views - Taming the Beast, 27 screencasts
3. Learn Flag, 8 screencasts
4. Learn Page manager!, 13 screencasts
5. Drupal learning curve for configurators
6. Learn Drupal coding (experimental screencast series, ongoing)
7. Rules for Drupal 6, 14 screencasts
8. Theming Drupal 6, 8 blog posts
9. 49 Drupal 6 modules you should know, 7 blog posts

We love sharing Drupal power.

Do you like these learning series? Check out our pre-conference training at DrupalCon London!

Redigera

profile data that represent the same person but with *different* e-mail. These duplicates are recognised by their names, and the client wanted to be able to merge name duplicates after manual inspection (since, obviously, people can have the same name without being the same person).

The solution for this merge is similar to the one used for e-mail duplicates:

- A view (cw_all_potentioal_duplicates) takes first name, last name and NID as argument and returns the first matching profile (that hasn't the given NID).
- The output of the view is rewritten to a link on the form "node/NID/duplicates", with the text "Potential name duplicate".
- This view is embedded in the list showing the results of profile imports – making each profile with a potential name duplicate appear with a red link "Potential name duplicate".
- The custom page at node/NID/duplicates is displayed as a tab on profile nodes. It lists the NID profile in the left column, and *all* potential name duplicates in the right column (using another display in the aforementioned view). Each potential duplicate is listed along with a button-styled link "Merge into this profile", leading to the path "merge/from-NID/to-NID".
- The merge path has a triggered rule listening to it, checking permissions for the acting user and then firing the rule set "Merge two profiles", and then redirecting to the merged-into profile for checking any further duplicates.

Some notes on profile merging by name

The normal workflow for managing name duplicates is to take care of them directly at import, but since there is a "Name duplicates" tab on profile pages this can also be carried out on a profile-by-profile basis by any registrar.

THE ACTUAL MERGING

All this management of profile duplicates wouldn't be very effective if there were no actual merge taking place. The rule set "Merge two profiles" deserves a quick look. This is what is does:

1. The two account case: If both profiles have user accounts, we end up in a pretty complex case where content, comments and user references must be transferred. This case is *not* supported, and a warning message is displayed telling the acting user that the merge must be made manually. (See below for comments on why this was left as a manual case.)
2. Only source account case: If the source profile but not the target profile has a user account, that account is set as the target profile owner and the source profile marked for deletion. Also, a confirmation message is displayed.
3. No account to transfer case: In case the source profile had no user account – the most frequent one – a confirmation message is displayed to let the actin user know that all went well.
4. Merge profile data (always carried out): The actual merge of data is carried out in three steps – replacing a lot of CCK data in the target profile, adding the import reference to the target profile, and adding the source's primary e-mail address to the target's additional e-mail addresses. The former is carried out with a singe action – Copy multiple field content between nodes – allowing to skip any fields that were empty in the source profile, while the latter two utilize the action "Insert a value in a multiple-value field". Thanks again to Rules Bonus Pack.

Some notes about the two-account case

We started to develop a solution for managing the case of two user profiles each having a user account, but rather quickly concluded that it would become a mess to solve it.

Since the most likely case of duplicate *accounts* (not only profiles) is a newly created account without any history on the site, it would be fairly safe to assume that it could just be deleted. However, it *might* be the case that the account has history on the site, and then a deletion could be really bad. Discussions with the client led us to the conclusion that the rare cases where duplicate accounts would emerge could be managed by hand, and the time needed to automate that process would be better spent on other features. Hats off for cost/benefit analysis.

ADDING PROFILES MANUALLY AND CREATING USER ACCOUNTS

Registrars and course administrators have the option to add profiles manually. When doing so, there are checkboxes "teacher", "registrar" and "administrator" used to giving the associated account privileges on the site. This functionality is managed by the rule set "Actions when profiles are saved" in combination with "Assure that profile has user account". The first rule set has rules setting account role, as well as removing it if the

checkboxes are changed later on.

There is a potential for the profile and user account becoming out of sync, since there are no rules updating profile checkboxes on account updates. This problem is strongly limited by the fact that registrars cannot manage user accounts directly, and the administration accounts are only used by one or two site administrators with good knowledge about the site.

Some notes about creating user accounts automatically

One problem of automatically creating user names is the risk of duplicates names, which Drupal will refuse to create – with the consequence of the user/profile handling becoming out of sync. This is at the time of writing handled by appending the NID for the profile node to the user name, but will eventually be replaced by the Rules Bonus Pack action "Generate a valid user name" which only appends a number if necessary to make the name unique. (Having user names without additional numbers makes it possible to display user name istead of profile name, which saves a lot of work.)

During the development we discovered that the automatic user creation sometimes broke, and eventually found that the reason was that some imported names ended with a space. This yielded a double space in the user name, which Drupal (rightly) refuses to accept. At the time of writing we still have left to implement the "Generate a valid user name" action to bypass this problem. This functionality will go into the two rule sets "Assure that profile has user account" and "Assure that profile has student user account".

A final note about logins

The site uses LoginToboggan to allow users to login with their e-mail addresses. On login, a rule redirects users to the front page.

Du bevakar detta inlägg, klicka för att sluta bevaka

Publicera på Drupal Planet

Kommentarer

Skriv ny kommentar

Ditt namn:
Johan Falk

Kommentar: *

Webbadresser och e-postadresser görs automatiskt till länkar.
Lägger till typografiska justeringar.
Tillåtna HTML-taggar: <a>   <cite> <code>   <dl> <dt> <dd>
Rader och stycken bryts automatiskt.
Du kan skriva kod med <code>...</code> (generiska) eller <?php ... ?> (markerade PHP) taggar

Mer information om formateringsmöjligheter

Spara | Förhandsvisa