User data developer documentation

In order to comply with a new EU data protection law (GDPR) we need to provide more comprehensive and configurable ways of managing user data. In particular we need to be able to fully delete a user's personal data (or anonymise it when deletion isn't possible), as well as provide a machine-readable export of a user's data.

To facilitate this we are introducing a new component (/totara/userdata), which includes an 'item' class that can be implemented by components by implementing specific code in /mod_name/classes/userdata/itemname.php. The code implemented by a component provides the necessary functions to define any settings for that component as well as complete the tasks (purge, export) for data relating to that component. The code itself will be executed by the central totara/userdata component.

A 'Purge profile' can then be created by enabling one or more userdata items which implement the purge function. This purge profile can then be applied to a user. The enabled purge items will delete the relevant data. Note that this works independently of user deletion. Purge profiles can be applied to user in any state - active, suspended or deleted. Existing functionality when a user is deleted must be maintained - this ensures backwards compatibility.

Requirements for completing "user data items"

Investigation

First you need to investigate what user-related data exists. This will involve reading the code, looking at the database schema etc. Don't forget that not all user data is stored in the database (e.g. files). Some points to consider at this stage:

- What data exists that could be considered 'user data'?
- Who created the data?
- Who does the data relate to and is it different from the creator (e.g. a 360 feedback would be authored by one user but relates to the person being discussed)?
- Are there any data dependencies, either within the component (replies in a forum thread depend on the parent) or across components (activity completion could impact course completion)?
- What are the data relationships, e.g. if you tried to delete this data what else would be affected?
- Are any events triggered by this component when data is modified or deleted? What impact might this have on other areas of the system?
- Is there any data that can't be deleted (due to referential integrity, for example)? Can we anonymise it instead?
- What does the current code do in relation to user deletion (if anything)? Are there any existing API functions?
- Is there any data that shouldn't be exported as user data? E.g. web service tokens that may be security sensitive? Data the user should not be allowed to see?

Decide on how to implement count

As well as the purge() and export() methods there is a count() method which will return the number of items that will be affected. You should always implement the count method unless you have a very good reason not to (we've never seen a case such as this). Generally, count() should return the number of top-level items being affected - if you are deleting other child records as part of deleting, you don't need to include them in the count.

When implementing a class that includes both purge() and export() ensure that the two functions are acting on the same data - e.g. the count should apply to both number of items being deleted and number being exported. If the count would be different, consider implementing two user data item classes instead.

Development

Create user data items with purge and export functions to achieve what you decided above. Follow the 'How to implement...' section below. Ensure you write unit tests to confirm that, given each item, the correct data is affected, and that data belonging to other users is not affected. Your tests may also need to cover different behaviour depending on the user's status (active, suspended, deleted), and confirming that contexts are correctly handled.

Things to consider when designing user data items

Should there be more than one user data item?

Data items represent distinct settings in the interface that an admin can choose to purge. In some situations it makes sense to give the admin control to decide which specific pieces of data they want to purge. In other situations you may want to provide one setting that deletes data from several related items without giving the administrator fine-grained control. Some considerations that may impact your decision are:

- If you can imagine a scenario where an administrator may want to delete some of the data for this component but leave the rest of it, it might make sense to split into separate settings (for example you might choose to split a user's appraisal responses into those submitted as the appraisee and those submitted in another role like manager or appraiser).
- If there is data in multiple tables that is required as a whole to represent a single item, then you may choose to require them to purge it all or none of it. For example: when deleting a user's personal goal (from goal_personal table), it would make little sense to allow them to leave any goal custom field values entered by the user for that goal (from the goal_user_info_data table). Or when deleting certification completions you must delete corresponding program completion, course set completion and certification records at the same time.
- Could multiple pieces of data be deleted together in order to simplify the experience (e.g. "Delete all logs" rather than "Delete site logs" and "Delete reportbuilder logs")
- Do user data items overlap? Consider if the data deleted in one purge item is a subset of another. For example if one item deletes "All messages" and another other deletes "Only received messages" then there could be confusion when someone selects the first but not the second - they

might expect the received messages to be kept. You could instead write the settings to partition the full set of data, e.g. by only having "Received messages" and "Sent messages". Most situations can be resolved without resorting to overlapping user data items. For example, Notes has a purge user data item to remove all data, plus two export user data items to allow exporting of hidden and visible data separately.

Which user statuses can the purge be applied to?

Purging can be applied to users in three states: active, suspended or deleted. However there are some types of data that cannot be deleted in all of those states (for example we can't delete a user's username if they are still active). Therefore we need to consider which user states allow the data to be deleted. This is used by the system to prevent data from being deleted by a user in the wrong state.

In addition, we may need to take a different approach to deleting the data depending on the user's status. For example, for deleted users we are probably safe to perform a low-level delete of database records since we don't expect any additional activity in relation to the user. But for an active user, we may need to trigger events to allow other parts of the system to update based on the change we are performing (for example if their activity could impact the grade in a group activity).

Does the item need to consider the context data is being deleted in?

When deleting, data a context can be provided to limit the data deletion to that context. While the interface does not currently allow purge or export to be executed in a particular context (other than the top level "system"), this may be possible in the future, or customisations may be made which utilise this feature. Contexts are not relevant for all types of data, for example user profile data relates to the user at the site level. On the other hand certain data such as data relating to course activities may be deleted within the context a specific category, course or activity. Consider whether your component acts at the system level or a specific context. If it is context specific then ensure your code considers the context that is passed and that your unit tests have positive and negative tests to ensure the correct data is deleted.

Does the order of data deletion matter?

In some cases, the order of data deletion can be important. For example, if you delete a course completion record before you remove the related activity completion records then recalculation of completion will regenerate the overall completion. If you notice any dependencies (your user data item must be deleted before or after another user data item) then this might indicate that the two user data items should be merged together.

Who does the data belong to?

The purpose of deleting user data is to allow a user to remove data, which relates to them, from the system. However Totara allows administrators and course creators to create a lot of content that is not necessarily personal to them. We need to make the distinction between a user's personal data and data that they happened to create.

It is hard to provide clear guidance on this as it is somewhat subjective, but generally if the data is made available to other users and serves a clear purpose for their use of the site then it may be appropriate to say it's not personal to that user.

Examples of user created data that is **probably not** personal to them (and therefore should **not** be deleted):

- Courses and course content
- Organisational hierarchies/frameworks
- Company goal frameworks
- Totara menu items

Examples of user created data that **may be** personal to them (and therefore should be possible to delete):

- Forum posts
- Quiz answers and grades
- User profile information

Below are a couple of specific examples where careful consideration is required:

Dashboard/blocks (purge example)

To give another example where there is some complexity, consider blocks and dashboards. Dashboards can be site wide pages visible to multiple users, but some dashboards can also be customised to create user-specific dashboards. When the dashboard is personal to a specific user, if the user data includes dashboards then you would expect all block instances on their personal dashboards to be fully deleted as a consequence of deleting the dashboard.

However if the dashboard is shared, and it contains blocks that contain user specific data, then you may need to provide the ability to delete the data for a specific user from the block. In this situation it would make sense to implement a user data item for the block, designed to just delete user specific data. Independently there would be a userdata item for dashboards, which would be limited to personal dashboards but which would delete the full block instances.

Custom user profile fields (user export example)

Imagine a situation where a site has multiple custom user profile fields. Some are visible to the user, and contain personal information. Others are hidden custom profile fields which the user cannot see, perhaps containing sensitive business-specific information about that user.

If we provide only a single user data item for 'user profile fields' then the administrator would need to choose whether to include it in exports (leading to users being able to export and view the hidden fields) or not (leading to personal data being excluded from the export). An alternative approach would be to recognise the difference between the two types of fields, and provide separate user data items for 'visible custom profile fields' and 'hidden custom profile fields'.

What are the consequences to other users if the data is removed?

If the data belonging to one user is removed, what effect will this have on other users of the system? Is that acceptable or do we need to mitigate that somehow (for example by anonymising instead of deleting).

Some examples of this might be:

- Group assignments. If one group member's data is deleted could that impact the results for other members of the group?
- Forum threads. What will happen to replies if the parent is deleted? Will the discussions make sense?
- 360 feedback. Will a user's ratings change if the data belonging to one or more respondees is removed?
- Removing course completion of a user could affect the number of staff that their manager has supervised through completion.
- An appraisal might relate to a user, but could be filled in only by a manager and appraiser. If the appraisee is deleted, should the manager and appraiser's comments be deleted? If the manager is deleted, should their comments be deleted from the appraisee's appraisal?

If the data cannot be completely removed, is there some way of anonymising the data, other than deleting it? Consider what it should be replaced with.

If data is not purged, is it possible that it could be used to identify the person the data belongs to? We need to provide a way to delete enough data that the remaining data cannot be connected with a real world person. E.g. in logs, we can't remove the records, but we could perhaps null out the IP address. In forums, we can't break threads, but we can remove the content of posts and replace it with "Author was deleted".

Which data should be exported?

The user data export is being done to comply with two parts of GDPR:

1. Article 20: Right of data portability
   This gives the user the ability leave the provider and take their data with them. The key clause is:
   "The data subject shall have the right to receive the **personal data concerning him or her**, **which he or she has provided** to a controller"
   Therefore there are three criteria which define whether data should be included to meet this clause **all** of which must be met for data to need to be exported:
   a. The data must be concerning them.
   b. The data must be personal data. Personal data is separately defined here as:
      'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person
   c. The user must have provided the data themselves.
2. Article 15: Right of access
   This gives the user the ability to receive a copy of the data about them, so that they can confirm what data the organisation is holding about them. The key clause is:
   "The data subject shall have the right to obtain from the controller confirmation as to **whether or not personal data concerning him or her are being processed**, and, where that is the case, **access to the personal data**"
   Therefore there are two criteria which define whether data should be included to meet this clause all of which must be met for data to need to be exported:
   a. The data must be concerning them.
   b. The data must be personal data.

We need to include data that meets the criteria for either Article 15 or Article 20. Based on these criteria we can essentially ignore 1c, because if an item meets 1a and 1b it is required to be exported under Article 15 anyway.

When deciding on if a piece of data is personal or not, some subjectivity is required. Of course it's always possible that a user could identify themselves anywhere that they could enter text (I could save my name, address and social security number in the site name if I wanted!). We need to be pragmatic and decide if it's *reasonable to expect* the data would contain personal data *in normal circumstances*.

Similarly the term "personal data concerning him or her" requires some subjectivity to decide when it applies. Just because a piece of data is *connected* to a user in some way (for example via a user id because they created it) that doesn't automatically mean it concerns them. To determine this consider whether this piece of data is related *specifically to that user*, or whether it may be *equally* related to other users of the site.

The table below gives criteria for some specific examples:

Data	**Is concerning them?** (1a + 2a)	**Is personal data?** (1b + 2b)	**User provided the data?** (1c)	**Need to export under Article 20?**	**Need to export under Article 15?**	**Need to export?**
User profile data entered by the user. (e.g. City, interests, contact number)	✅	✅	✅	✅	✅	✅
User profile data created by other people. (e.g. hidden profile fields, automatically synced profile data)	✅	✅	❌	❌	✅	✅

Forum posts	✓	✓	✓	✓	✓	✓
Grades / completion data	✓	✓	✗	✗	✓	✓
Assignment submissions / Quiz answers	✓	✓	✓	✓	✓	✓
Feedback from the trainer on your assignments (exporter a learner)	✓	✓	✗	✗	✓	✓
Feedback from the trainer on assignments (exporter is the trainer)	?	✗	✓	✗	✗	✗
Organisation frameworks you created	?	✗	✓	✗	✗	✗
Courses created as a trainer	?	✗	✓	✗	✗	✗
Site configuration settings	✗	✗	✓	✗	✗	✗
Custom field definitions set up as an administrator	✗	✗	✓	✗	✗	✗
Positions of blocks on your personal dashboard	?	✗	✓	✗	✗	✗
Site logs of your activity	✓	✓	✗	✗	✓	✓
Featured link tiles you created on shared dashboards as an admin	✗	✗	✓	✗	✗	✗
Featured links tiles you created on a personal dashboard (visible only to yourself)	✓	✓	✓	✓	✓	✓
Appraisal answers given by the appraisee (exporter is the appraisee)	✓	✓	✓	✓	✓	✓
Appraisal answers given by the manager (exporter is the appraisee)	✓	✓	✗	✓	✓	✓
Appraisal answers given by the appraisee (exporter is the manager)	?	✓	✗	✗?	✗?	✗?
Appraisal answers given by the manager (exporter is the manager)	?	✓	✓	✓?	✓?	✓?

What are the consequences of allowing the user to export their user data?

Consider what happens when you export the data. If one userdata item contains both purge and export, do they relate to the same data?

Does the export provide a way for a user to access data that they normally wouldn't have access to, even if it 'belongs' to them? E.g. if all user profile custom fields were exported (due to them containing data belonging to the user, e.g. "Hobbies") could there be hidden fields which were never meant for the users to see, e.g. "Maximum potential salary".

Does the item need to be broken down to allow personal data export and prevent secret data being revealed (e.g. An item for visible data and another for hidden data)?

Is this a complex item that may require time and care to be accurate about what a user should be able to see? If there is not a straightforward divide between what is visible and not visible for export, it is acceptable to export everything for that item, providing this is made clear in the UI. The strings for each item are shared between export and purge types, so you may need to create a separate item for the export, allowing you to define a specific string for it where you can advise that this may ignore whether or not this is visible to the user.

What end-user documentation is required?

Consider the best name for the setting, as well as whether any supplementary help is required, via the help button or in the help documentation. In some cases a setting may warrant a special note or warning. For example if deleting the data might impact compliance reporting or not deleting data might impact GDPR compliance for that organisation.

Guiding principles

To help inform the approach we've tried to express some general guiding principles to apply:

- All else being equal, it's probably better to have less separate user data items rather than more. It is easier to split one later than to merge them.
- User data items should be created when the data **might** need to be deleted for GDPR compliance, even if we consider the scenario where the data would not be purged is the more common scenario.
- When creating user data items, try to ensure that items don't overlap (e.g. the items deleted by one setting will not delete data that is described by another, separate setting). For example instead of "Delete all messages" and "Delete unread messages" you should create "Delete unread messages" and "Delete read messages".
- Existing 'delete user' behaviour will remain exactly the same for backwards compatibility. Purging provides an additional, completely optional mechanism for removing data, but does not replace the existing data deletion process. Despite this, we still need to implement user data purge for data that is deleted when a user is deleted, even if it doesn't apply to active or suspended users.

How to implement a userdata item class

- Create a new class in the <plugin>/classes/userdata folder. Give it a name which matches the thing that it relates to - the data that it is purging and/or exporting.
- Namespace <plugin>\userdata.
- Extend \totara_userdata\userdata\item
- By default the string that appears in the user interface is identified by '*userdataitem***itemname**' (i.e. userdataitemforum_post) and the component of the item. Override ***get_fullname_string()*** if you need to use a custom key.
- A help icon can be added by creating a string that uses the same key and component as above, but with '_help' appended to the key. However, to keep the UI clean, this should only be done if necessary.
- Implement ***get_compatible_context_levels()*** - indicates how narrow a context the item purge and execute could (potentially) be applied in. E.g. you could delete quiz data for a user for the whole site, just one category, just one course, or for a specific quiz (activity) This isn't currently fully utilised, but would be good to get it right to start with. Either put the correct contexts here and implement the related code, or, if it applies to several contexts but you're only implementing "system" then make sure to document it in code (E.g. "TODO: Only implemented SYSTEM but this could be extended to apply at COURSE context").
- If the item relates to data to be purged then ***is_purgeable()*** needs to return true. Then implement purge, which takes a target_user and a context.
- If the item relates to data to be exported then ***is_exportable()*** needs to return true. Then implement export, which takes a target_user and a context.
- If the item can count data relating to a user (e.g. "How many appraisals does this user have?") then ***is_countable()*** needs to return true. Then implement count, which takes a target_user and a context. Generally, is_countable will always be required to return true.

Export

Export is fairly straightforward. Instantiate a new \totara_userdata\userdata\export object. Populate the data property with an array of the related data. Your data items can be arrays or objects - they will be json encoded before being presented to the user.

There's a convention on how files should be added to the export. This makes sure the export of files will happen consistently throughout the items.

See the Export files convention page for more information.

Count

Count is similarly simple. It should be a count of data represented by the item name. Example 1 - if the item is something like "Phone number" then it could report 1 if there is something in the field or 0 if it is empty. Example 2 - if a user has 2 dashboards, containing 8 blocks each, and the item is called dashboard, it should return a count of 2, even if export and purge affect the blocks as well.

Purge

Purge is the difficult function to write. It needs to take into account all the considerations mentioned in the rest of this page.

If your purge function will make more than one DB update or delete call, and if the data getting into an inconsistent state could cause something in Totara to fail, then you should surround those DB calls with a transaction. See how existing user data items handle transaction - much of the work is done in the calling function, so be sure to follow the pattern of other user data items.

Implement tests

Tests are extremely important for purge items. You should be testing each combination of possible user status and context. Your tests need to prove that the function deletes the data that it is supposed to delete, and doesn't affect any data that it is not supposed to delete.

For example, if your purge item can be applied in the course context then you should show that:

- Deleting in the system context deletes all of that item belonging to the target user on the site, and none belonging to any other user.
- Deleting in the course category context deletes all of that item belonging to the target user in the target category, and not any belonging to the user in any other category, nor any belonging to any other user in any category (including the target category).
- Deleting in the course context deletes all of that item belonging to the target user in the course, and not any belonging to the user in any other course, nor any belonging to any other user in any course (including the target course).

In this example, your minimum data set would be two users, one item for each user in the target course in the target category, another item for the target user in another course which is in the target category, another item for the target user in another course which is in another category.

You should include tests for the following:

- is_purgeable, is_exportable, is_countable - always test these
- purge - only if is_purgeable is true, you need to call execute_purge in your test
- export - only if is_exportable is true, your need to call execute_export in your test
- count - only if is_countable is true, your need to call execute_count in your test
- get_compatible_context_levels - your test needs to check if *get_compatible_context_levels* returns the expected levels

Note that you don't need to test purge, export or count with contexts which are not allow by get_compatible_context_levels, because the execute_xxx functions strictly control this for you.

Example components

There should already be quite a few user data item example, so you can look for code in classes/userdata for examples of how they are written. Some specific examples and important files relating to this work are listed below:

- Base item class, defines the methods to be implemented: /totara/userdata/classes/userdata/item.php
- Target item class, passed into purge/export method: /totara/userdata/classes/userdata/target_user.php
- Example purge method with different user status behaviour: user/classes/userdata/
- Example purge method with contexts: /mod/glossary/classes/userdata/entries.php
- Example export method: /mod/glossary/classes/userdata/entries.php
- Example of unit tests: mod/forum/tests/userdata_posts_test.php

Site data directory

The following information described how Totara Learn stores user data.

Data is stored in the site data directory, which is broken up by subfolders - each serving a specific purpose.

Directory	Description
cache	Used by the internal caching system. Out of the box only used by the file cache store, this will contain all kinds of information both user and system oriented. In a horizontally scaled environment this must be shared.
filedir	Used to store all permanent assets provided by end users, whether they are personal (such as evidence and private files) or for use within resources (files within courses and activities).
lang	Used to store installed language packs.
localcache	Used for cache data again, but this is safe to store locally in a horizontally scaled environment.
muc	Used to store the cache system configuration.
sessions	Used to store user sessions.
temp	Used to store temporary information during active processes.
trashdir	Once a file in the filedir directory no longer has any links it is moved to the trashdir and cleaned up after a period of time (configured via scheduled tasks).

There is no way presently to separate personal files from resource files.

It's also worth noting that files are stored in a structure based upon their content hash with all actual file information being stored in Totara. If several users all upload an identical file it is only stored on disk once, and links to this file are maintained within the product.
It's optimal for web use, and modelled around WebDav, however it would make it incredibly hard to perform any kind of separation in user files.

Other considerations

- Be aware that purge and export code will be running in the context of the person deleting, **NOT** in the context of the user being deleted. When writing the purge and export code, if you call any existing API functions you need to review the code to ensure that it is not dependent on the user who is running the deletion.
- When implementing a userdata item, provide good documentation in the docblock at the top of the class explaining what you are doing, why and any gotchas related to this component.
- If using an existing API to perform an action, make yourself aware of what the code does. Things like visibility of items and permissions of the admin doing the deletion must not impact the data being deleted. If an existing function only deletes data where visible = 1 for example, you will need to take another approach.
- Events correspond to data you are purging should be triggered in most cases. However, some events should not be triggered for deleted users. These include events that take a user context,. If the data affects anyone other than the deleted user, the event should normally be fired. But please review how the event is used and include manual testing.
- For course modules, you do not need to purge the centralised completion data, e.g. from the course_modules_completion table. But you do need to purge data owned by the module (typically the table name will be prefixed by the module name)
- Also for course modules, you should check if you need to re-evaluate grades after purging. This is typically done with a function called {modulename}_update_grades(). Aside from that you don't need to purge grades directly as this will be dealt with by a separate grades setting - this doesn't apply where there are 'grades' tables owned by the module, e.g. the table called workshop_grades still needs to be dealt with by the workshop module.