EMu Documentation

The EMu Registry

Document Version 1

EMu version 4.3

EMu
Museum
Management
System

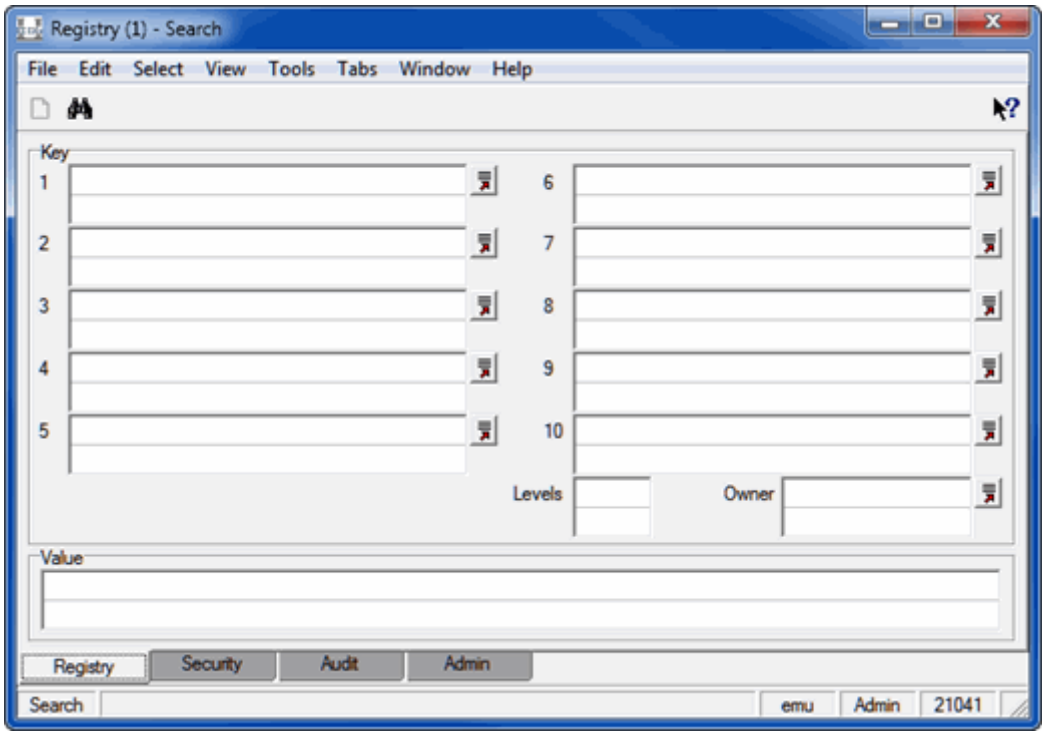
Contents

Overview of the Registry

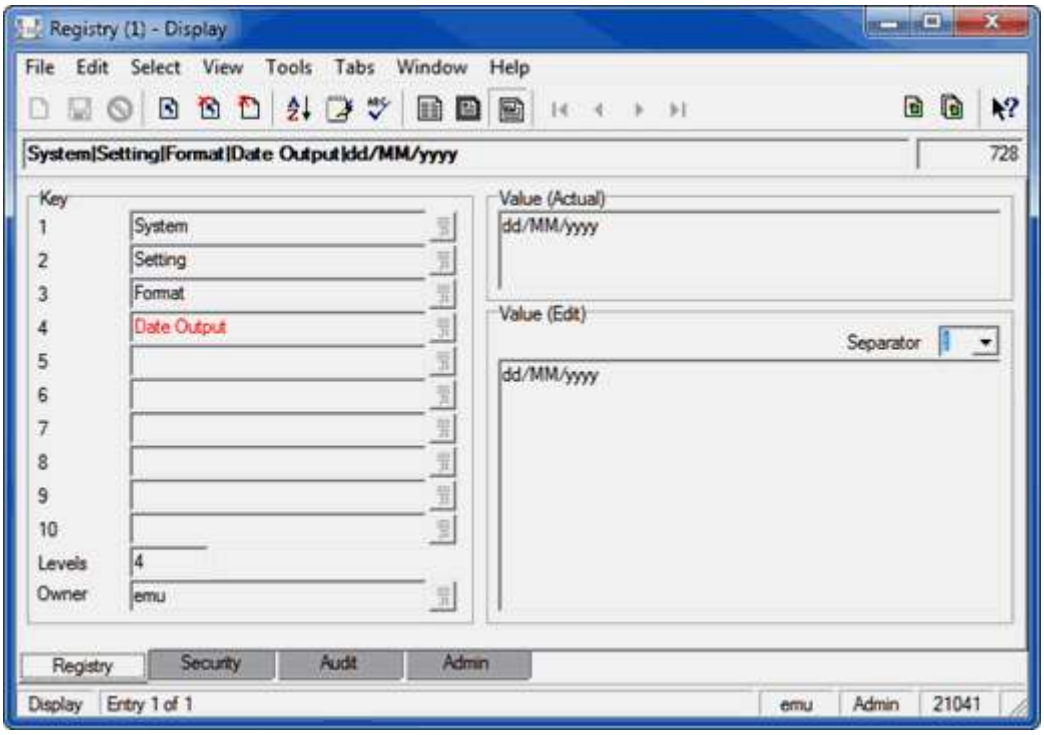
EMu is a flexible and extendable application that can be customised with relative ease to suit each institution's requirements. Almost anything in EMu can be customised, from the fields a user can view and search, to the printer that print job are sent to. The key to EMu's flexibility is its Registry. The EMu Registry stores System configuration information and provides a mechanism for configuring and customising EMu without the need to reprogram the application. It is used to define site-specific configuration parameters such as:

- EMu's security system, including user and group permissions
- System settings, including locale information and file locations
- MIME types for multimedia resources and conversion information for text-based documents
- Reports
- Mandatory fields
- Lookup List permissions
- Sort options
- Tab order and display
- Languages in use and general language settings
- Unique values
- Order and appearance of tabs in different modes e.g. New, Search
- Spell checking options

Applying Registry settings to configure and customise EMu is performed in a module with a user-friendly interface much like that of any other EMu module:



In essence the EMu Registry is just another database, and much like the Windows Registry, EMu's Registry has a hierarchical key structure with a value:



On the left is a unique, hierarchical sequence of keys with a single *Value* entered on the right. There may be up to ten keys in the hierarchy, although most Registry entries use only the first four or five keys. The *Value* tells EMu which permission, condition or value to apply for the specific circumstance described by the preceding sequence of keys.

Although there is only one *Value* for each Key sequence, the *Value* can comprise multiple entries. In the example above, the *Value* is a single date format entry, but it could be a list of modules that a user can access, for example. In this case the entries are listed in one string and separated by a ; (semicolon):

```
eaccessionlots;eadmin;ebibliography;ecatalogue;econdi
tion;econservation;eevents;einsurance;einternal
```

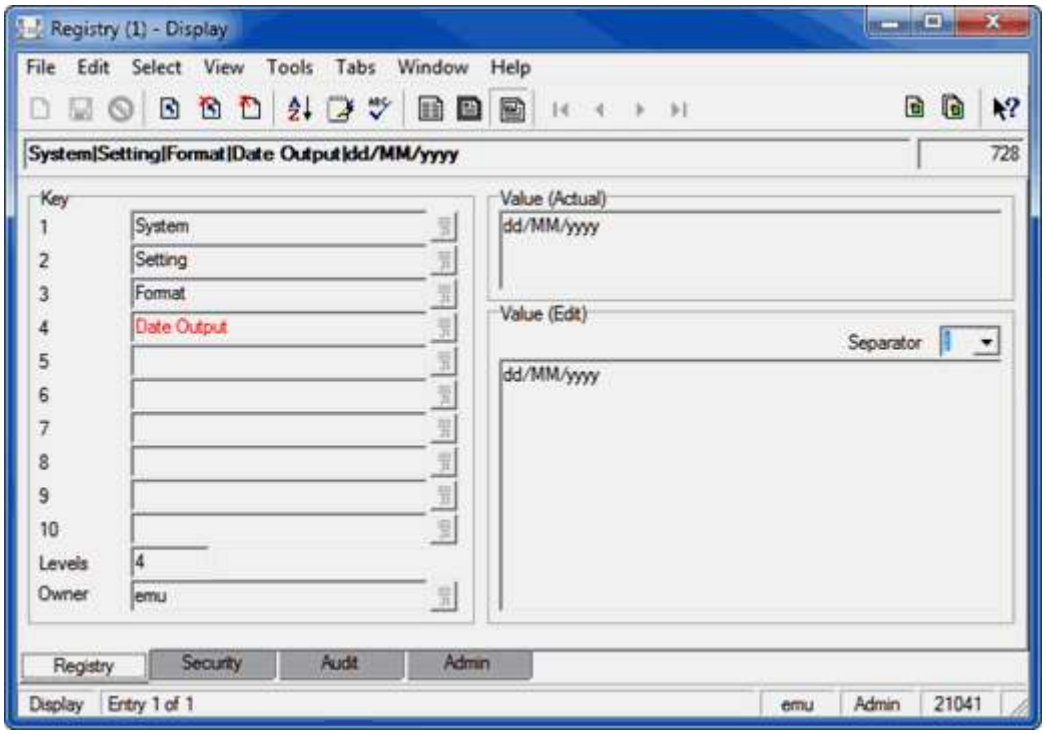
Almost every operation a user performs in EMu triggers a Registry entry lookup. This occurs behind the scenes and defines what action the System should take depending on the user's privileges, or the System settings.

By default the Registry module is only accessible to a user with an Administrator account, although Administrators can authorise other users to access the Registry module and to add and edit Registry entries.

While there are potentially hundreds of entries to be made in the Registry, for the most part the EMu Administrator only needs to be concerned with two types of Registry entries: System Settings (page 9) and Table and Column Settings (page 10).

How Registry entries are documented

In EMu documentation, a Registry entry can be represented in two ways. Consider the following example:



This can be represented:

1. By a pipe delimited string of keys, with the last being the value associated with the key:

   `System|Setting|Format|Date Output|dd/MM/yyyy`

   -OR-

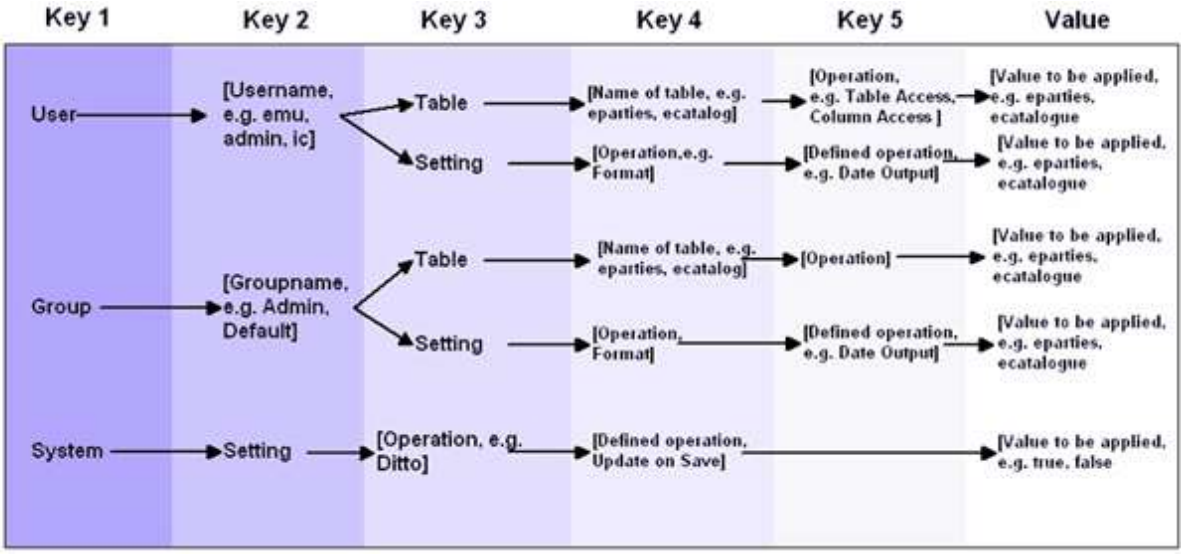
2. In tabular form:

Field	Value
Key 1	System
Key 2	Setting
Key 3	Format
Key 4	Date Output
Value	dd/MM/yyyy

The structure of a Registry entry

The EMu Registry module is a user-friendly interface to a database that stores access permissions, functions, display options and other settings applied to individual users, groups or the entire system.

The following diagram illustrates some of the possible values that can be used in the first five keys and how the selection of a value in one key can determine the possible options in the subsequent keys:



Elements of a Registry entry

As we can see in the illustration of the structure of a Registry entry (page 4), an entry is made up of a variety of elements: labels, variables, defaults and the value itself. In pipe delimited format (page 3), we can represent this as:

`User|`*user*`|Table|`*table*`|Entry|`*value*

`User|`*user*`|Table|Default|Entry|`*value*

`Group|`*group*`|Table|`*table*`|Entry|`*value*

`Group|`*group*`|Table|Default|Entry|`*value*

`Group|Default|Table|`*table*`|Entry|`*value*

`Group|Default|Table|Default|Entry|`*value*

where:

`User`	This entry is for a user whose name is given in *user*.
`Group`	This entry is for a group whose name is given in *group*.
user	The name of the user affected by this Registry entry.
group	The name of the group affected by this Registry entry.
`Table`	The EMu module that this entry affects. A module is a user-friendly interface to a database table.
table	The back-end name of an EMu module, e.g. eparties for the Parties module. eparties is the name of the database table that holds data entered and displayed in the Parties module. When a user logs in to EMu, the Registry is searched to determine which modules the user has access to (specified in a Table Access Registry entry) and the operations that the user can perform on those tables (specified in a range of Registry entries).

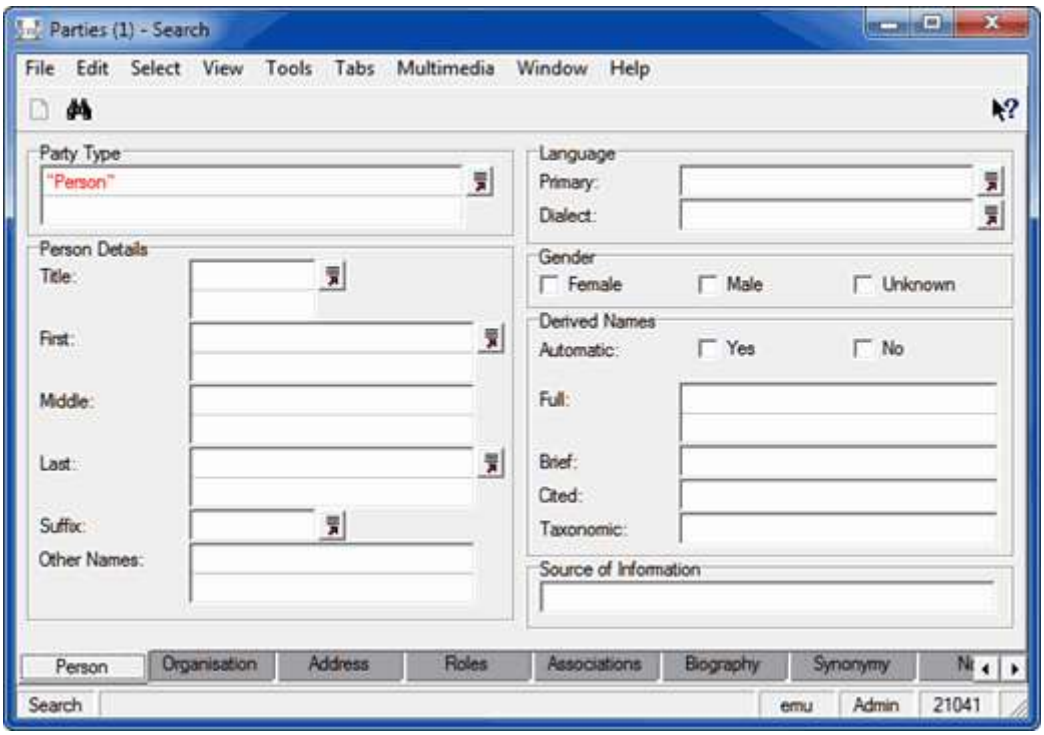
`Default`	If `Default` follows `Group` e.g.:
	`Group|Default|...`
	the Registry entry refers to ALL groups.
	If `Default` follows `Table` e.g.:
	`Table|Default|...`
	the Registry entry refers to ALL modules.

value	The *value* tells EMu which permission, condition or value to apply for the specific circumstance described by the preceding sequence of keys. *value* will often refer to one or more tabs or columns/fields in a module. Read on for details of how to reference Tabs (page 6) and Fields and Columns.

Tabs

A tab is a page in a module that displays a number of grouped fields. The Parties module, for example, can have tabs for:

- Person
- Organisation
- Address
- Roles
- Associations
- Biography
- History
- Notes
- Multimedia
- Security
- Admin
- Audit



Each tab has a unique name which consists of:

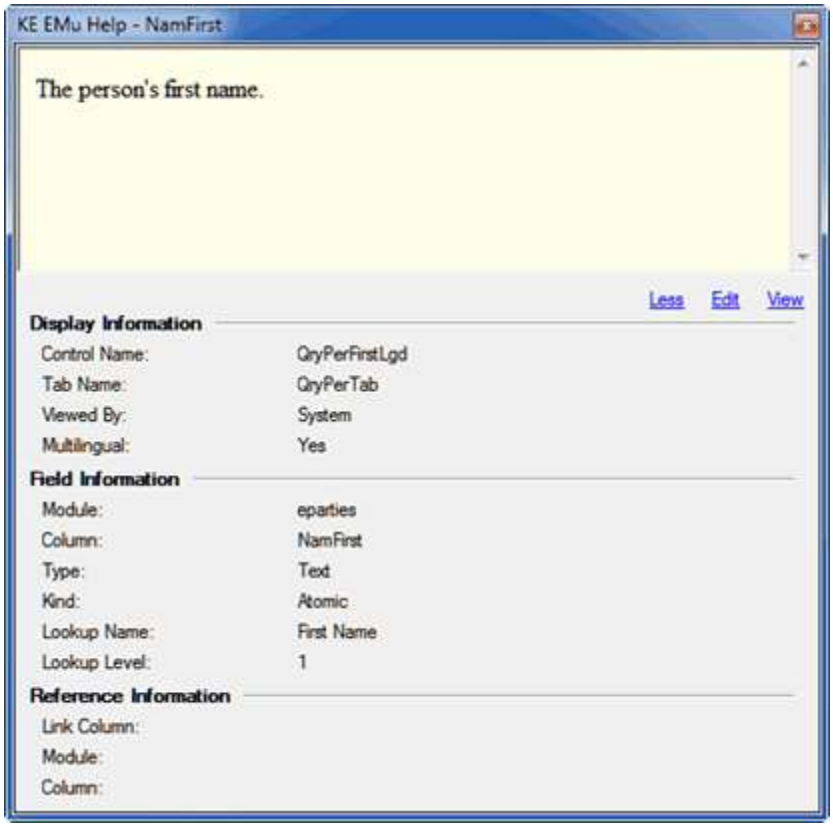
- A prefix of either `All` or `Qry` based on its use in Display (`All`) or Search (`Qry`) mode.
- A three-character identifier, usually comprising the first three characters of the label of the tab, e.g. `Per` for the Person tab in the Parties module.
- A suffix of `Tab`.

For example, when referencing the Person tab in an EMu Registry entry, its unique name is:

- `AllPerTab` in Display mode
- `QryPerTab` in Search mode

Rather than listing all tabs, a value of `All` can be used as a shortcut to display all of the available tabs in a module.

A tab's name is listed under Display Information when accessing the Field Level Help for a field on the tab. For example, when we access Field Level Help for *First: (Person Details)* on the Person tab of the Parties module in Search mode, we see that the Tab name is listed as `QryPerTab`:



Tab names are used in Registry entries to determine what tab options are available to a user. For instance, a range of Tabs Registry entries can determine the number, type and order of tabs that display in a module for each user.

The order in which Registry entries are assigned

In general, Registry permissions can be set for:

- A user
- A group
- System-wide

When a user logs in to EMu a search is performed in the Registry to determine what permissions have been set for the user. The System looks for Registry entries in this order:

1. Entries for the user.
2. Entries for a group to which the user belongs.
3. Entries that apply system-wide.

When a match is found, the search ends. In other words, a permission set for a user has precedence over a permission set for a group which has precedence over a system-wide setting.

As we saw earlier (page 1), the EMu Administrator is typically concerned with two types of Registry entries:

- System settings (page 9)
  -AND-
- Table and Column settings (page 10)

In the following pages we look more closely at the sequence in which each type of Registry entry is assigned.

System settings

The first type of Registry entry an Administrator is concerned with is System Setting Registry entries. Customisations and configurations that should be effective across the system are set with a System|Setting Registry entry. The structure of this type of entry is:

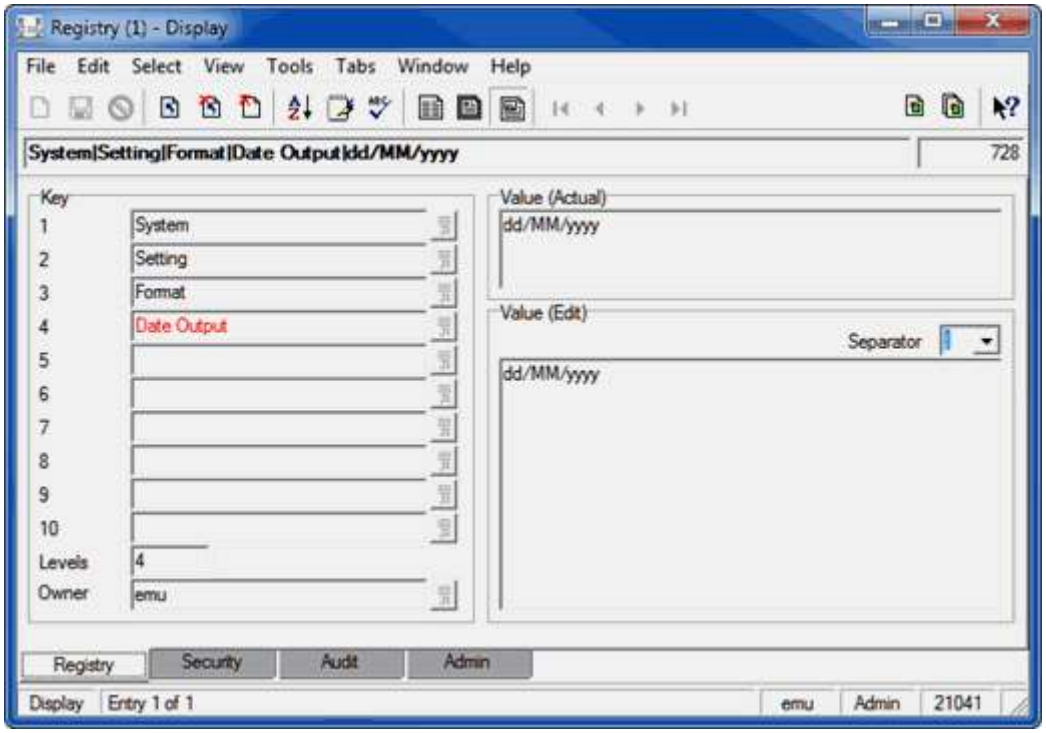
`System|Setting|subkey1|subkey2|...|`*value*

It is possible however to customise the system so that different users and groups have different values for the same setting.

Each time the System looks up a `System|Setting` Registry entry, it looks for entries in the following order:

1. `User|`*user*`|Setting|subkey1|subkey2|...|`*value*
2. `Group|`*group*`|Setting|subkey1|subkey2|...|`*value*
3. `System|Setting|subkey1|subkey2|...|`*value*

The following is an example of how EMu checks for a System setting for a user. Consider this entry which tells the System in what format dates should be displayed:



The user has the username `emu` and is in group `Admin`. The Registry is searched for an entry that defines the date format to be displayed when `emu` logs in to EMu:

Search	Description	Search Details	Results
1	The Registry is searched for an entry for `Date Output` that is defined for user `emu`.	`User|emu|Setting|Format|Date Output`	If this entry exists, the *Value* associated with it will be used as the Date Output format for user `emu`.
2	If the above entry does not exist, the System searches the Registry for an entry for group `Admin`.	`Group|Admin|Setting|Format|Date Output`	If this entry exists, the *Value* associated with it will be used as the Date Output format for all members of group `Admin`, which includes user `emu`.
3	If the above entry does not exist, the System searches for a group `Default` entry.	`Group|Default|Setting|Format|Date Output`	If this entry exists, the *Value* associated with it will be used as the Date Output format for every group, which includes user `emu`.
4	If this entry does not exist, the System searches for an entry that applies to all users, i.e. `System`	`System|Setting|Format|Date Output`	This entry will exist, and its value will be used as the Date Output format for everyone, including user `emu`.

Some entries may not require a default System Setting entry as the System may have a default operation built into it. Consult the documentation of the particular Registry entry for this information.

Table and Column settings

While configurations can be applied to individual users and groups, others can be applied to individual tables and even columns for each user or group. In this way the System can be customised such that EMu's behaviour can be quite different visually and internally from user to user.

Just as for System|Setting (page 9) Registry entries, with Table and Column entries the System first queries the Registry for a user entry; if a user entry is not located, a more generic entry is sought that can be applied to the user.

Generally the queries are in the following form and precedence:

`User|`*user*`|Table|`*table*`|Setting|`*column*`|`*value*

`User|`*user*`|Table|`*table*`|Setting|Default|`*value*

`User|`*user*`|Table|Default|Setting|`*column*`|`*value*

`User|`*user*`|Table|Default|Setting|Default|`*value*

`Group|`*group*`|Table|`*table*`|Setting|`*column*`|`*value*

`Group|`*group*`|Table|`*table*`|Setting|Default|`*value*

`Group|`*group*`|Table|Default|Setting|`*column*`|`*value*

`Group|`*group*`|Table|Default|Setting|Default|`*value*

`Group|Default|Table|`*table*`|Setting|`*column*`|`*value*

`Group|Default|Table|`*table*`|Setting|Default|`*value*

`Group|Default|Table|Default|Setting|`*column*`|`*value*

`Group|Default|Table|Default|Setting|Default|`*value*

These entries demonstrate the basic structure of the querying hierarchy.

Not all entries have exactly the same structure as set out above. Indeed, if the configuration is to be set on a Table basis rather than on a Column within a Table, there are only six variations of the query.

The queries begin by looking for an entry that is specific to a user, table and column within that table. In the absence of such an entry, the search continues for an entry that is applied System-wide regardless of user, table, or column.

As soon as an entry is found, the *Value* for that entry is applied to the user and no more querying takes place.

A setting of `Default` may be used where a *group*, *user*, *table* or *column* is expected. This specifies that the entry can apply to *all groups, all users, all tables*, or *all columns*.

In the entries above, `Setting` is the name of the entry, and may span several keys to achieve the desired precision. Consider the following entry:

```
Group|Registrations|Table|emultimedia|Repository|List|
Registrations Repository
```

In this entry the `Setting` contains two keys, `Repository` and `List`. Consult the Registry documentation to find out the precise syntax for a particular entry.

Example

The following is an example of how the System will check for a Table setting for a user. The user has the username `emu` and is in group `Admin`. The System is looking for an entry that defines the type of access user `emu` has to the eparties table:

Search	Description	Search Details	Results
1	The System looks for an `Operations` entry that is defined for user `emu` for the eparties table.	`User|emu|Table|eparties|Operations`	If this entry exists, the *Value* associated with it defines the table operations for user `emu` when accessing the Parties module.
2	If the above entry does not exist, the System searches for an entry for user `emu` for all tables.	`User|emu|Table|Default|Operations`	If this entry exists, the *Value* associated with it defines the table operations for user `emu` when accessing any module.
3	If the above entry does not exist, the System searches for an entry for group `Admin` for the eparties table.	`Group|Admin|Table|eparties|Operations`	If this entry exists, the *Value* associated with it defines the table operations for the Parties module when accessed by group `Admin`, which includes user `emu`.
4	If the above entry does not exist, the System searches for an entry for group `Admin` for all tables.	`Group|Admin|Table|Default|Operations`	If this entry exists, the *Value* associated with it defines the table operations for all tables accessed by group `Admin`, which includes user `emu`.
5	If the above entry does not exist, the System searches for an entry for all groups and all tables.	`Group|Default|Table|Default|Operations`	If this entry exists, the *Value* associated with it defines the table operations for all tables accessed by all groups, which includes user `emu`.

How to search for a Registry entry

To search for a Registry entry:

1. In the Registry module in Search mode, enter search terms in one or more fields.
   If you enter search terms in the *Summary Data* field, the search will look for matching terms in both the *Key* and *Value* fields. For example, entering `Query Defaults` in the *Summary Data* field will search for the value in the *Key* or *Value* fields.
2. Run the search (`Ctrl+F`).
   Matching records are returned.

Examples

To search for Registry entries that apply to user `emu`, enter `emu` in *Key 2*.

To search for any Column Access Registry entries set, enter `Column Access` in *Key 5*. Why? When specifying a Column Access Registry entry, the format (for a single user for example) is:

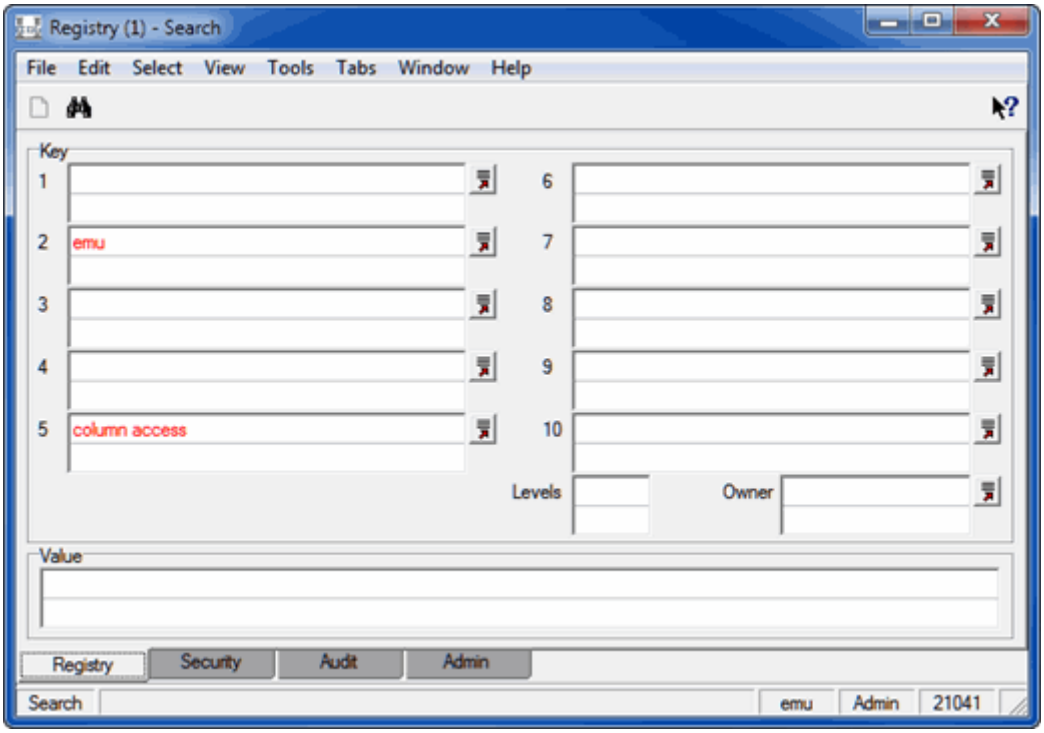
`User|`*user*`|Table|`*table*`|Column Access|`*column*`|`*priv;priv;...*

or alternatively:

Field	Value
Key 1	`User`
Key 2	*user*
Key 3	`Table`
Key 4	*table*
Key 5	`Column Access`
Key 6	*column*
Value	*priv;priv;...*

As you can see, `Column Access` is entered in Key 5.

To search for any Column Access Registry entries specified for user `emu`, enter both values:


Registry (1) - Search
File Edit Select View Tools Tabs Window Help
Key
1
2 emu
3
4
5 column access
6
7
8
9
10
Levels
Owner
Value
Registry Security Audit Admin
Search
emu Admin 21041

How to add a Registry entry

To add a Registry entry:

1. Add a New record (`Ctrl+N`) in the Registry module.
2. Enter values into as many of the *Key* fields as required (use the Lookup List for the field to view existing values).

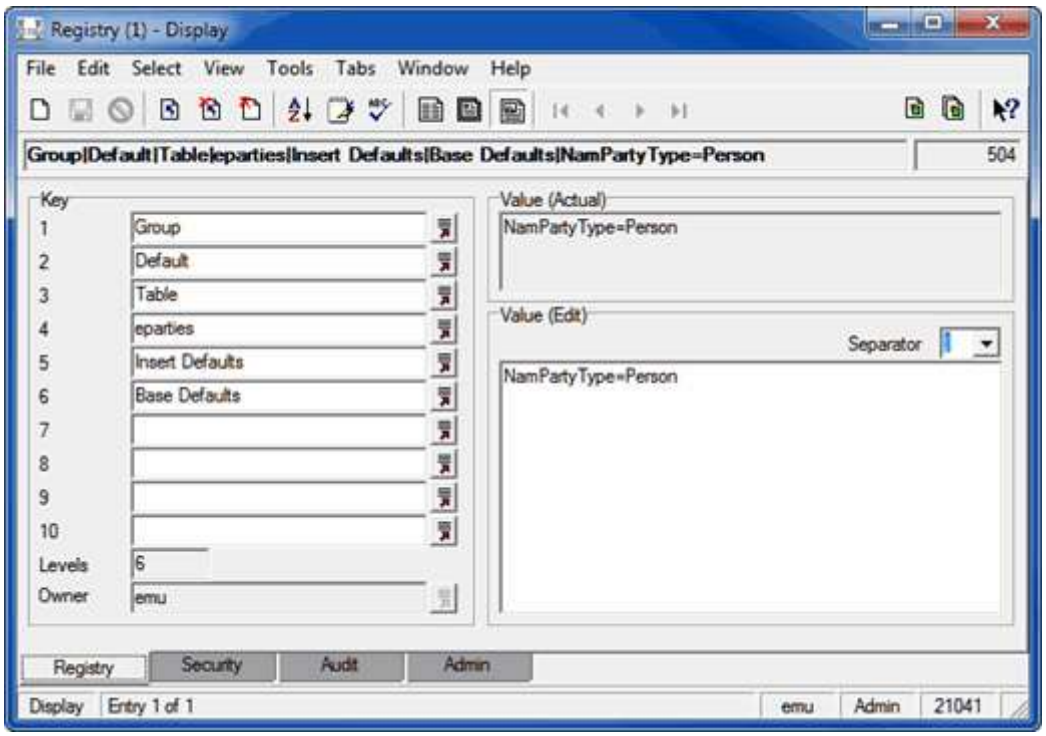
   Take care when adding Registry entries: an incorrect Registry entry is generally ignored by EMu but does have the potential to result in unexpected behaviour.

3. Enter a value into the *Value (Edit)* field.
4. Save the record (`Ctrl+S`).

   The full entry displays in the *Summary Data* field.

An example Registry entry

This Registry entry defines a default value that will be displayed when a new record is created in the Parties module.



The Registry entry is:

Field	Value	Description
Key 1	`Group`	This Key specifies whether the entry applies to a group or user (in this case a group).
Key 2	`Default`	This Key specifies the name of a specific group or user or all groups (in this case it applies to all groups).
Key 3	`Table`	This Key specifies whether the entry concerns a Table, Report, Settings, etc. (in this case it concerns a Table).
Key 4	`eparties`	In this case, this Key specifies which Table (eparties).
Key 5	`Insert Defaults`	The value in this Key indicates that the entry concerns default fields in New (Insert) mode.
Key 6	`Base Defaults`	This is the name of the setting.
Value	`NamPartyType=Person`	This is the value for this entry: when any user enters a new record in the Parties module, the *NamPartyType* field defaults to `Person`.

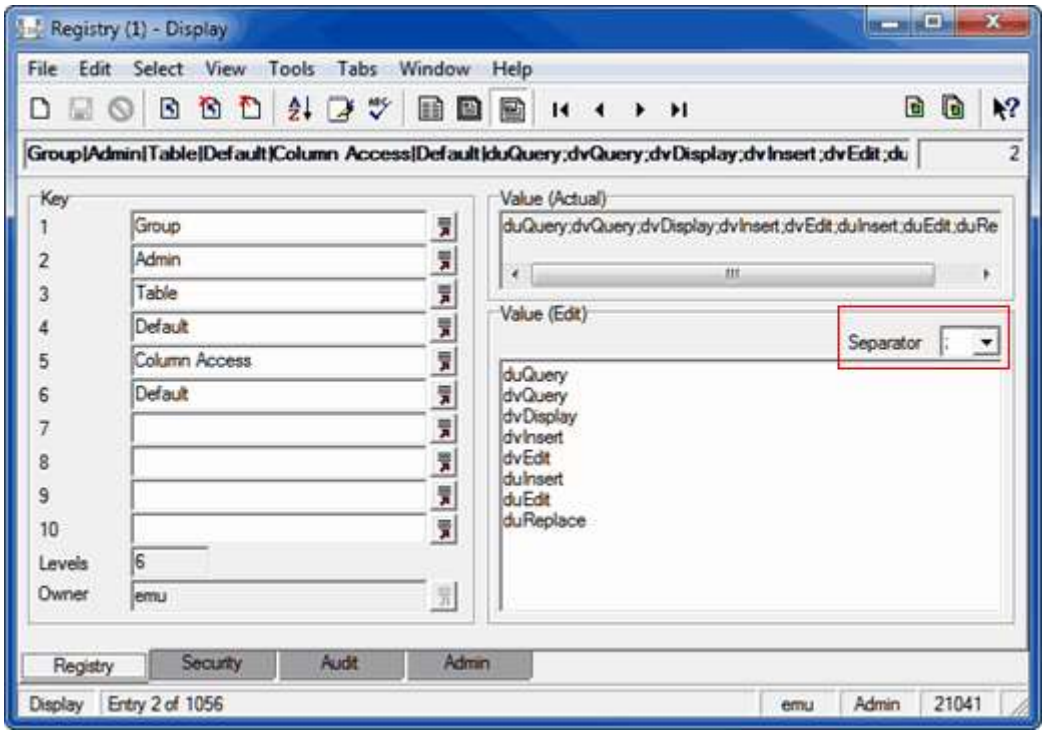
An alternative representation of this Registry entry is:

`Group|Default|Table|`*eparties*`|Insert Defaults|Base Defaults|`*NamPartyType=Person*

The delimiter used to separate values in a Registry entry

Values can be displayed in the *Value* field of a Registry entry either as individual items on separate lines or as a string separated by a delimiter.

The default delimiter in EMu is a semicolon: when individual values are entered line by line in a Registry entry, they display in the EMu back-end as a list of values separated by semicolons. For example, the value in the following entry:



will display in the back-end as:

```
duQuery;dvQuery;dvDisplay;dvInsert;dvEdit;duInsert;duEdit;
duReplace
```

It is possible to select a different delimiter (e.g. #) from the Separator drop list (highlighted in the screenshot).

Index