**More Types for Advanced Metadata**

|  |  | Concepts and Training |
| --- | --- | --- |
| Author | Richard Klees |
| Version | V1.0 |
| Date | 2014-03-07 |

# **Versions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Changes** | **Version** | **Editor** | **Date** |
| first draft | v0.1 | Richard Klees | 2014-02-26 |
| unsigned int => int + options, options for floats | V1.0 | Richard Klees | 2014-03-07 |

**Settings**

|  |  |
| --- | --- |
| Package Shortname | AMDTypes |
| ILIAS-Version | 4.4 |
| Branch | svn.ilias.de/svn/ilias/branches/CaT/patches/4\_4\_AMDTypes |
| CaT Contact | Richard Klees <richard.klees@concepts-and-training.de> |
| Development by | Leifos GmbH |
| Developers Contact | Jörg Lützenkirchen < luetzenkirchen@leifos.com> |
| Required Documentation | classes, public and protected methods including parameters, inline documentation of intricate parts of code |

# Intent

The Advanced Metadata (AMD) Feature of ILIAS allows to attach additional information to courses, categories and glossaries. The attached information is typed, currently there are four different types available. The types are used to check and restrict user input to a certain range of allowed values.

On the one hand this helps users to make a correct input, on the other hand this is crucial to beeing able to use AMD in plugins or patches, i.e. employing the data for more than just plain displaying for users. It is eligible to use AMD for plugins, since AMD offers a standardized way to attach new data to objects which automatically makes that data accessible through forms and in views.

To make AMD more usefull in scenarios where it is used by plugins or patches, there is the need to have more types available. This would also cater scenarios without plugins or patches, where the data is just editable and gets displayed.

# New Datatypes

## Integer

It should be possible to define an AMD-field having the type integer. For those fields it is also possible to define a range for the allowed inputs or define the field to be unsigned. When using an AMD-input of that type, the input should get rejected if it is not numeric or numeric but not in the defined range. When querying for that data, the querying code should be able to assume that the retreived value is an integer which is equal or larger then zero.

## Numbers with Decimal Place

It should be possible to define an AMD-field having the type number with decimal place. For those fields it is also possible to define a range for the allowed inputs or define the field to be unsigned. When using an AMD-Input of that type, the input should be accepted if it is a number with decimal places, where the separator is either a point or a comma. Code querying that value should be able to assume that the retreived value is a float.

## Locations

It should be possible to define an AMD-field having the type location. A location consists of a longitude, a latitude and a degree of zoom, like the coordinates that are already used for displaying a map from GoogleMaps in ILIAS. For setting a location field there should be the possibility to directly set latitude, longitude and zoom, to search for an adress or to select the displayed latitude, longitude and zoom by using a map component. This would be exactly the form found at the input for the default location of an ILIAS installation.

When displayed, the location should be shown as point on a map according to latitude and longitude, where the map is zoomed to the set value.

# Refactoring

Currently the types supported by AMD are encoded via constants, where the different behaviour for fields with different types is realized via switch statesment in place. To accomodate creation of new types for AMD in the future, this implementation should be changed in a way that the type of a field is factored out of the AMD logic.

The different types should be represented as classes. A class for a type should be able to

* check, weather a given input satisfies the type
* transform a given input to an appropriate PHP-representation of its value
* give a standard FormGUI for setting a value
* give a standard FormGUI to filter values
* turn an appropriately represented PHP-value to a representation for a view

This goal should not be realized via static methods, since that would circumvent to derive new types from existing ones. Furthermore it could be nice to have the possibility to put further constraints on an input of a given type in the future, e.g. defining a regexp for an string field.

The classes for types should implement a common interface or derive from a common base class, to force the implementation of methods that are needed for a type and use type hinting at appropriate locations. The already existing classes of ADM should refer only to the generall type-class or -interface.

The available types should be accessible via one class, that is able to list all available types and create instances of a given type. This would make it possible to implement a class for a new type for AMD and add it in one place only.

# Outlook

The notion of types is not only used for ADM in ILIAS. There are other components, that use this notion as well, e.g.

* DataCollection
* UserDefinedFields
* Forms

It is desireable to generalize the notion of types to a new service that could be used in every place that concept is needed. Factoring the type out of AMD could be a first step towards that goal.