Syntax specification

Moodle High Level Script Engine

Syntax basics

The MoodleScript syntax aims to be simple, so we will avoid excess of formal syntactical forms and will prefer a keyword based language that results in naturally readable statements.

A moodle command is basically composed by keyword, arguments, variables and parameter lists. A command always starts with a verb (ADD, REMOVE, ENROL, BACKUP, etc.) and follows with an alternance of keywords and arguments, forming a readable sentence. Some commands may accept an argument list, f.e. to provide a set of attributes to an object to create, or giving conditional clauses for an object destruction.

The common form of a MoodleScript command is:

```
VERB KEYWORD1{1,n} [ [arg1]{0,n} KEYWORD1{0,n} ]{0,n}
```

A command that accepts an argument list as last parameters is of the form:

```
VERB KEYWORD1{1,n} [ [arg1]{0,n} KEYWORD1{0,n} ]{0,n} HAVING
key1: value1
[key2: value2]
...
[keyn: valuen]
```

Only one single argument list can be given for a single command.

Keywords

Keywords are reserved words that have a special syntactic meaning when present at some location in the command. Keywords are ALWAYS UPPERCASE.

Keywords are:

Verbs (always first command word)

Verbs are single tokens (one word) and must designate an action (e.g ADD or REMOVE)

Accepted verbs:

```
ADD, REMOVE, ENROL, BACKUP, MOVE
```

Object types

Object typenames designate a class of objects to administrate. They can be single words or multiple words (in a set of known expression).

```
COURSE, CATEGORY, ENROL METHOD, USER, COHORT, BLOCK, MODULE, etc.
```

Context articulations

context articulations are small words that tell what is the use of an argument, in order to provide naturzal readability of the script command.

```
IN, FOR, TO, IF EXISTS, IF NOT EXISTS
```

Arguments, identifiers and variables

Arguments are usually moodle object identifiers, terminal values or eventually global variables, for finding or setting values. the nature of the argument will vary across the syntax, and refers usually to the most common or trivial object type that is expected in the syntax.

F.E., for en enrolment syntax:

```
ENROL id:33 IN id:3 AS shortname:student USING manual
```

shows 4 attributes that are naturally refering to (successively) a user, a course, a role and an enrol method.

In case the expression have some possible ambiguity, additional keywords will be used to discriminate possible cases.

Identifiers

when the syntax requires to identify an existing object, and this object may be identified by several information, we will use an explicit field discriminator and value couple, f.e. for a user, there are usually 4 admitted possible identifiers as primary id, username, idnumber or email.

Thus the following identifiers are usable when searching for a user :

```
id:33
username:johndoe
idnumber:JD@35465
email:john.doe@gmail.com
```

Literal Argument

Literal arguments are simple words or strings. There is at the moment a restriction on syntax as strings are not specifically delimited (or only delimited by keywords). So strings should not contain keywords expressions. this is likely why we chose keywords in strict uppercase, to minimize syntactic collision with literal usual strings.

Variables

We may need to inject some environmental values in the script to replace some non terminal placeholders. An execution stack can be fed at launch time with a global context data stub that will be merged with each instruction local context (adding or overriding values). Global context variables can be placed whereever in script statements or attribute lists using the Moodle common SQL named variable form:

```
:varname
```

To be valid, the placeholder expression MUST have at least one space character before it.

Return to component index

From:
https://docsen.activeprolearn.com/ - **Documentation Moodle ActiveProLearn**

Permanent link:
https://docsen.activeprolearn.com/doku.php?id=local:moodlescript:syntaxspecification&rev=1519720539

Last update: **2026/01/13 08:58**