AIMMS Function Reference - Model Edit Functions

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com

Model Edit Functions

AIMMS supports the following functions for model editing:

- me::AllowedAttribute
- me::ChangeType
- me::ChangeTypeAllowed
- me::ChildTypeAllowed
- me::Children
- me::Compile
- me::Create
- me::CreateLibrary
- me::Delete
- me::ExportNode
- me::GetAttribute
- me::ImportLibrary
- me::ImportNode
- me::IsRunnable
- me::Move
- me::Parent
- me::Rename
- me::SetAttribute

me::AllowedAttribute

The function `me::AllowedAttribute` returns 1 if the attribute is allowed for the runtime id.

```
me::AllowedAttribute(
        runtimeId,  ! (input) an element
        attr        ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

attr
: An element in the set `AllAttributeNames`

Return value:

Returns 1 if the attribute `attr` of runtime identifier `runtimeId` is allowed. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The procedures `me::SetAttribute` and `me::Create`.

me::ChangeType

The procedure `me::ChangeType` changes the type of a runtime identifier.

```
me::ChangeType(
        runtimeId,  ! (input) an element
        newType     ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

newType
: An element in the set `AllIdentifierTypes`.

Return value:

Returns 1 if the change type operation is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Create` and `me::Move`.

me::ChangeTypeAllowed

The function `me::ChangeTypeAllowed` returns 1 if the type of runtime identifier `runtimeId` can be changed into type `newType`.

```
me::ChangeTypeAllowed(
        runtimeId,  ! (input) an element
        newType     ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

newType
: An element in the set `AllIdentifierTypes`.

Return value:

Returns 1 if the identifier `runtimeId` can be changed into `newType`. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Create` and `me::Move`.

me::ChildTypeAllowed

The function `me::ChildTypeAllowed` returns 1 if a child of type `newType` can be added as a child to runtime identifier `runtimeId`..

```
me::ChildTypeAllowed(
        runtimeId,  ! (input) an element
        newType     ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

newType
: An element in the set `AllIdentifierTypes`.

Return value:

Returns 1 if the identifier of type `newType` can be added as a child to identifier `runtimeId`. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Create` and `me::Move`.

me::Children

The procedure `me::Children` returns the number of children of a runtime identifier and fills an output parameter with those children.

```
me::Children(
        runtimeId,          ! (input) an element
        runtimeChildren(i)  ! (output) indexed element parameter.
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

runtimeChildren
: The children in the runtime identifier tree. This parameter needs to be an output parameter indexed over a (subset of) the set of integers.

Return value:

This procedure returns the number of children of `runtimeId`. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Parent` and `me::GetAttribute`.

me::Compile

The procedure `me::Compile` compiles a runtime identifier and all runtime identifiers below that identifier. If that runtime identifier is a runtime library, all procedures can be run and set / parameter definitions can be evaluated provided there are no errors.

```
me::Compile(
        runtimeId  ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

Return value:

Returns 1 if the compilation operation is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::IsRunnable` and the `APPLY statement` 10.3.1.

me::Create

The function `me::Create` creates a runtime identifier.

```
me::Create(
        name,     ! (input) a string
        newType,  ! (input) an element
        parentId, ! (input) an element
        pos       ! (optional) an integer
)
```

Arguments:

name
: A string that is valid name for a runtime identifier.

newType
: An element in the set `AllIdentifierTypes`.

parentId
: An element in the set `AllSymbols` referencing a runtime identifier.

pos
: 1 is the first position, and 0 means ”place at end”, the default is 0.

Return value:

Returns an element in `AllSymbols` if successful or the empty element otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Delete` and `me::SetAttribute`.

me::CreateLibrary

The function `me::CreateLibrary` creates a new runtime library.

```
me::CreateLibrary(
        libraryName,   ! (input) a string
        prefixName     ! (optional) a string
)
```

Arguments:

libraryName
: The name of the new runtime library.

prefixName
: The name of the new prefix, when not specified one is generated from the `libraryName`.

Return value:

The function returns an element in the set `AllIdentifiers` referencing the library when successful and the empty element upon failure. In the latter case at least one error has been raised.

See also:

The functions `me::ImportLibrary` and `me::Create`.

me::Delete

The procedure `me::Delete` a runtime identifier and all runtime identifiers below that identifier.

```
me::Delete(
        runtimeId  ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

Return value:

Returns 1 if the delete operation is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Children` and `me::GetAttribute`.

me::ExportNode

The procedure `me::ExportNode` writes a section to file.

```
me::ExportNode(
        esection,  ! (input) section element.
        filename,  ! (input) a string
        password)  ! (optional) a password
```

Arguments:

esection
: An element in the set `AllIdentifiers` referencing a runtime library or a section in a runtime library.

filename
: The name of file to which the section is written. When the filename ends in `.amb` the `.amb` format will be used, otherwise the `.aim` format will be used.

password
: The `.amb` file will be protected with a developer password. When a non-empty password is given without using the `.amb` format an error will be raised, and the file will not be written.

Return value:

The procedure returns 1 if the file is written successfully. If the procedure fails to write the file it returns 0 after raising errors.

See also:

The functions `me::CreateLibrary`, `me::ImportLibrary` and `me::ImportNode`.

me::GetAttribute

The function `me::GetAttribute` returns the contents of an attribute as a string.

```
me::GetAttribute(
        runtimeId,  ! (input) an element
        attr        ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

attr
: An element in the set `AllAttributeNames`

Return value:

Returns the contents of the attribute `attr` of runtime identifier `runtimeId` as a string. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The procedures `me::SetAttribute` and `me::Create`.

me::ImportLibrary

The function `me::ImportLibrary` reads a runtime library from file.

```
me::ImportLibrary(
        filename,   ! (input) a string
        password)   ! (optional) a string
```

Arguments:

filename
: The name of file that contains a runtime library.

password
: When the import library is an `.amb` file protected with a developer password, that developer password will be checked against the password in the argument.

Return value:

The function returns an element in the set `AllIdentifiers` referencing the library when successful and the empty element upon failure. In the latter case at least one error has been raised.

See also:

The functions `me::CreateLibrary`, `me::ImportNode` and `me::ExportNode`.

me::ImportNode

The procedure `me::ImportNode` reads a section from file.

```
me::ImportNode(
        esection,  ! (input) section element.
        filename,  ! (input) a string
        password)  ! (optional) a string
```

Arguments:

esection
: An element in the set `AllIdentifiers` referencing a section in a runtime library.

filename
: The name of file that contains a runtime library.

password
: When the import library is an `.amb` file protected with a developer password, that developer password will be checked against the password in the argument.

Return value:

The procedure returns 1 if the file is read successfully. If the procedure fails to read the file it returns 0 after raising errors.

See also:

The functions `me::CreateLibrary` and `me::ExportNode`.

me::IsRunnable

The function `me::IsRunnable` determines whether or not the runtime identifier resides in a runtime library for which all procedures are runnable and all definitions can be evaluated.

```
me::IsRunnable(
        runtimeId   ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

Return value:

The function returns 1 iff `runtimeId` resides in a runtime library where all procedures are runnable and all definitions can be evaluated. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Compile` and `me::IsReadonly`.

me::Move

The procedure `me::Move` renames a runtime identifier. In addition, when the move changes the namespace of the runtime identifier all text within the runtime library referencing that runtime identifier will be adapted accordingly.

```
me::Move(
        runtimeId,  ! (input) an element
        parentid,   ! (input) an element
        pos         ! (input) integer
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

parentid
: An element in the set `AllIdentifiers` referencing a runtime identifeir in the same runtime library.

pos
: An integer position in the section. 1 is the first position, and 0 means ”place at end”.

Return value:

Returns 1 if the move operation is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn’t reference a runtime identifier an error will be raised.

Remarks:

The name change file is not supported for runtime libraries.

See also:

The functions `me::ChangeType` and `me::Rename`.

me::Parent

The function `me::Parent` returns the parent of a runtime identifier.

```
me::Parent(
        runtimeId   ! (input) an element
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

Return value:

The function returns an element in the set `AllIdentifiers` referencing the parent of the referenced identifier or the empty element if the referenced identifier is a runtime library. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The functions `me::Children` and `me::GetAttribute`.

me::Rename

The procedure `me::Rename` renames a runtime identifier. In addition, all text within the runtime library referencing that runtime identifier will be adapted accordingly.

```
me::Rename(
        runtimeId,  ! (input) an element
        newname     ! (input) a string
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

newname
: A string.

Return value:

Returns 1 if the rename operation is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

Remarks:

The name change file is not supported for runtime libraries.

See also:

The functions `me::ChangeType` and `me::Move`.

me::SetAttribute

The procedure `me::SetAttribute` changes the type of a runtime identifier.

```
me::SetAttribute(
        runtimeId,  ! (input) an element
        attr,       ! (input) an element
        txt         ! (input) a string expression
)
```

Arguments:

runtimeId
: An element in the set `AllIdentifiers` referencing a runtime identifier.

attr
: An element in the set `AllAttributeNames`

txt
: The text to be assigned. Using the empty string will effectively delete the attribute from the runtime identifier.

Return value:

Returns 1 if the text assignment to the attribute is successful, 0 otherwise. In the latter case error(s) have been raised. When `runtimeId` doesn't reference a runtime identifier an error will be raised.

See also:

The procedures `me::Create` and `me::ChangeType`.