The Java `scriptmib` Package

Frank Strauß
Sven Mertens
Sven Brandenburg

Computer Science Department
Technical University Braunschweig
Bültenweg 74/75
38104 Braunschweig, Germany

{strauss,mertens,branden}@ibr.cs.tu-bs.de

December 23, 1999

Contents

1 Motivation

The Simple Network Management Protocol communicates only very low level operations between agents and network managers. Thus, MIBs have to define managed objects on an accordingly low level, agents have to implement them on such a low level and management applications have to work on the managed objects on this level. This means that even quite regular operations like table row creation and deletion have to be conveyed in (often multiple) simple Get- and Set-requests. Operations with multiple inter-table relations and error conditions are even more complicated. If we look at the manager side, as a consequence the development of applications that work on managed objects via SNMP might become error prone and time consuming.

The Java `scriptmib` package represents a prototype definition and implementation of a set of classes to the developers of Java management applications, that fulfills operations on Script MIB[1] agents. These classes represent Script MIB contexts and table rows. Their constructors and methods provide a means to operate on these tables on higher levels. Hence, they intend to make the development of Script MIB management applications more feasible.

2 Package Overview

The classes of the `scriptmib` package are roughly devided into two levels. On the higher level there is just one class:

- **ScriptMib** represents a specific Script MIB agent context which is parametered by an agent host, the agent UDP port, and an SNMPv1 community string[2]. Its methods allow high level operations, e.g. the retrieval of vectors of the available or installed script languages, language extensions, scripts, launch buttons, and running scripts. They also allow to install and execute scripts from URL source locations, including all required operations on the agent's Script MIB tables.

  Some methods of this class and the classes described below require polling on certain MIB table entries. The *ScriptMib* class provides some methods to tune the behavior of these polling operations with respect to their timeout and polling interval.

Below this class, there are five classes that represent the rows of the five most important tables of the Script MIB[3]. These classes are used by the *ScriptMib* class and also provided publicly to the package user:

- **Language** represents a script language that is provided by the agent. This class corresponds with the read-only Script MIB `smLangTable`.
- **Extension** similarly represents a language extension of a script language that is provided by the agent. This class corresponds with the optional read-only Script MIB `smExtsnTable`.

[1] RFC 2592
[2] SNMPv3 contexts are not yet supported.
[3] The `smCodeTable` is not mandatory in the Script MIB and is currently not support by the `scriptmib` package.

- **Script** represents a management script known to the agent. There are constructors to retrieve instances from the agent and to install new scripts within the agent. Other methods allow to get and set script parameters and to execute scripts, including all required operations on the `smLaunchTable` and `smRunTable`.
- **Launch** represents a script launch button. There are constructors to retrieve instances from the agent and to install new launch buttons within the agent. Other methods allow to get and set their parameters and to launch scripts, including the required operations on the `smRunTable`.
- **Run** represents a running (or terminated) script. There are constructors to retrieve instances from the agent. Other methods allow to get and set their parameters and to control their state (suspend/resume/abort).

Finally, there is one class, that represents Script MIB specific exceptions that can be thrown by methods of the above mentioned classes:

- **ScriptMibException** extends the class *Exception* and signals Script MIB specific exceptions that might occur due to access violations, SNMP communcation problems, concurrent MIB access, and other circumstances.

The following sections describe all classes and the signatures and semantics of all their public methods. Those sections have been automatically generated from the package's source code through the *javadoc* tool and the *TexDoclet*[4].

The `scriptmib` package is part of the Script MIB prototype project *Jasmin*[5].

[4] `http://www.c2-tech.com/java/TexDoclet/`
[5] `http://www.ibr.cs.tu-bs.de/projects/jasmin/`

3 Classes

3.1 *Class* ScriptMib

3.1.1 Declaration

public class ScriptMib

extends java.lang.Object

3.1.2 Constructors

- *ScriptMib*
  `public` **ScriptMib** `( )`
  - **Usage**
    * Create a ScriptMib object. Reasonable defaults are used for all SNMP parameters (host=localhost, community=public, port=161).

- *ScriptMib*
  `public` **ScriptMib** `( java.lang.String` **host** `)`
  - **Usage**
    * Create a ScriptMib object associated with a given agent host. Other SNMP communication parameters use reasonable defaults (community=public, port=161).
  - **Parameters**
    * `host` - the host running the Script MIB agent

- *ScriptMib*
  `public` **ScriptMib** `( java.lang.String` **host**`, int` **port**`, java.lang.String` **community** `)`
  - **Usage**
    * Create a ScriptMib object associated with a given set of SNMP parameters.
  - **Parameters**
    * `host` - the host running the Script MIB agent
    * `port` - the port on the SNMP agent host
    * `community` - the SNMP community string to be used

- *ScriptMib*
  `public` **ScriptMib** `( java.lang.String` **host**`, java.lang.String` **community** `)`

- **Usage**
  - Create a ScriptMib object associated with a given set of SNMP parameters. The agent port for SNMP communication is the well known port (161).
- **Parameters**
  - `host` - the host running the Script MIB agent
  - `community` - the SNMP community string to be used

3.1.3 Methods

- *call*
  `public String` **call**`( java.lang.String` **owner**`, script-mib.Language` **language**`, java.lang.String` **source**`, java.lang.String` **argument** `)`
  - **Usage**
    - Run a script from a given URL written in a given language, with a given argument, and with a given owner. Block until it has been terminated and return the result string. The required script and launch button are created and deleted implicitly.
  - **Parameters**
    - `owner` - the owner to use for the script and launch button
    - `language` - the script's language
    - `source` - URL to fetch the script
    - `argument` - argument string to pass to the script
  - **Returns** - the result string
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if any subsequent actions cannot be finished in time

---

- *call*
  `public String` **call**`( java.lang.String` **owner**`, java.lang.String` **pattern**`, java.lang.String` **source**`, java.lang.String` **argument** `)`
  - **Usage**
    - Run a script from a given URL written in a language described by a given description substring, with a given argument, and with a given owner. Block until it has been terminated and return the result string. The required script and launch button are created and deleted implicitly.
  - **Parameters**
    - `owner` - the owner to use for the script and launch button
    - `pattern` - description substring pattern to look up the language
    - `source` - URL to fetch the script
    - `argument` - argument string to pass to the script
  - **Returns** - the result string
  - **Exceptions**

* `scriptmib.ScriptMibException` - if any subsequent actions cannot be finished in time

---

- *getExtension*
  `public Extension` **getExtension** `( int [] ` **oidarray** ` )`
  - **Usage**
    * Get an Extension object by a given OID.
  - **Parameters**
    * `oidarray` - array of OID sub-identifiers
  - **Returns** - the Extension object
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the extension cannot be found

- *getExtension*
  `public Extension` **getExtension** `( java.lang.String ` **pattern** ` )`
  - **Usage**
    * Get an Extension object matching a description substring pattern. Note that only the first matching Extension object is found.
  - **Parameters**
    * `pattern` - the description substring pattern used for lookup
  - **Returns** - the Extension object
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the extension cannot be found

- *getExtensions*
  `public Vector` **getExtensions** `( )`
  - **Usage**
    * Get a vector of all Extension objects.
  - **Returns** - the vector of all Extension objects
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if a language/extension inconsistency occurs

- *getExtensions*
  `public Vector` **getExtensions** `( scriptmib.Language ` **language** ` )`
  - **Usage**
    * Get a vector of all Extension objects related to a given Language object.
  - **Parameters**
    * `the` - Language object
  - **Returns** - the vector of all matching Extension objects

---

- *getLanguage*
  `public Language` **getLanguage** `( int ` **index** ` )`

- **Usage**
  * Get a Language object by an smLangTable index.
- **Parameters**
  * `index` - index part of the Language table row
- **Returns** - the Language object
- **Exceptions**
  * `scriptmib.ScriptMibException` - if the Language cannot be found

- *getLanguage*
  `public Language` **getLanguage**`( int [] ` **oidarray** ` )`
  - **Usage**
    * Get a Language object by a given OID. Standard language OIDs are assigned by IANA and defined in the IANA-LANGUAGE-MIB module.
  - **Parameters**
    * `oidarray` - array of OID sub-identifiers
  - **Returns** - the Language object
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the Language cannot be found

- *getLanguage*
  `public Language` **getLanguage**`( java.lang.String ` **pattern** ` )`
  - **Usage**
    * Get a Language object matching a description substring pattern. Note that only the first matching Language object is found.
  - **Parameters**
    * `pattern` - the description substring pattern used for lookup
  - **Returns** - the Language object
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the Language cannot be found

- *getLanguages*
  `public Vector` **getLanguages**`( )`
  - **Usage**
    * Get a vector of all Language objects.
  - **Returns** - the vector of all Language objects

- *getLaunch*
  `public Launch` **getLaunch**`( java.lang.String ` **owner**`,`
  `java.lang.String ` **name** ` )`
  - **Usage**
    * Get a Launch object by a given owner and name.
  - **Parameters**

    * `owner` - the owner to look for
    * `name` - the name of the launch button to look for
  - **Returns** - the Launch object

- *getLaunches*
  `public Vector` **getLaunches** `( )`
  - **Usage**
    * Get a vector of all Launch objects.
  - **Returns** - the vector of Launch objects

- *getLaunches*
  `public Vector` **getLaunches** `( scriptmib.Script` **script** `)`
  - **Usage**
    * Get a vector of all Launch objects associated with a given Script object.
  - **Parameters**
    * `script` - the Script object
  - **Returns** - the vector of Launch objects

- *getLaunches*
  `public Vector` **getLaunches** `( java.lang.String` **owner** `)`
  - **Usage**
    * Get a vector of all Launch objects of a given owner.
  - **Parameters**
    * `owner` - the owner to look for
  - **Returns** - the vector of Launch objects

- *getPollFactor*
  `public int` **getPollFactor** `( )`
  - **Usage**
    * Return the percentage by which the poll interval increases between two polls (default: 150).
  - **Returns** - the interval factor

- *getPollInterval*
  `public int` **getPollInterval** `( )`
  - **Usage**
    * Return the start interval in milli seconds for methods of this ScriptMib object that perform SNMP polling operations (default: 100).
  - **Returns** - the interval in seconds

- *getPollTimeout*
  `public int` **getPollTimeout** `( )`

– **Usage**
  * Return the maximum polling time in milli seconds for methods of this ScriptMib object that perform SNMP polling operations (default: 10000).
– **Returns** - the maximum polling time in milli seconds

- *getRequest*
  `protected String` **getRequest**`( jmgmt.snmp.io.Varbind []` **varbinds** `)`

- *getRun*
  `public Run` **getRun**`( java.lang.String` **owner**`, java.lang.String` **name**`, int` **index** `)`

  – **Usage**
    * Get a Run object by a given launch button owner, launch button name, and index.
  – **Parameters**
    * `owner` - the owner to look for
    * `name` - the name of the launch button to look for
    * `index` - the run index to look for
  – **Returns** - the Run object

- *getRuns*
  `public Vector` **getRuns**`( )`

  – **Usage**
    * Get a vector of all Run objects.
  – **Returns** - the vector of Run objects

- *getRuns*
  `public Vector` **getRuns**`( scriptmib.Launch` **launch** `)`

  – **Usage**
    * Get a vector of all Run objects started from a given launch button.
  – **Parameters**
    * `launch` - the launch button
  – **Returns** - the vector of Run objects

- *getRuns*
  `public Vector` **getRuns**`( java.lang.String` **owner** `)`

  – **Usage**
    * Get a vector of all Run objects started from launch buttons of a given owner.
  – **Parameters**
    * `owner` - the owner to look for
  – **Returns** - the vector of Run objects

- *getScript*
`public Script` **getScript**`( java.lang.String` **owner**`,`
`java.lang.String` **name** `)`
  - **Usage**
    - Get a Script object by a given owner and name.
  - **Parameters**
    - `owner` - the owner to look for
    - `name` - the name of the script to look for
  - **Returns** - the Script object

- *getScripts*
`public Vector` **getScripts**`( )`
  - **Usage**
    - Get a vector of all Script objects.
  - **Returns** - the vector of Script objects

- *getScripts*
`public Vector` **getScripts**`( scriptmib.Language` **language** `)`
  - **Usage**
    - Get a vector of all Script objects written in a given language.
  - **Parameters**
    - `language` - the language to look for
  - **Returns** - the vector of Script objects

- *getScripts*
`public Vector` **getScripts**`( java.lang.String` **owner** `)`
  - **Usage**
    - Get a vector of all Script objects of a given owner.
  - **Parameters**
    - `owner` - the owner to look for
  - **Returns** - the vector of Script objects

- *installLaunch*
`public Launch` **installLaunch**`( java.lang.String` **owner**`,`
`java.lang.String` **name**`, scriptmib.Script` **script** `)`
  - **Usage**
    - Create and install a Launch object by a given owner, name, and script.
  - **Parameters**
    - `owner` - the owner
    - `name` - the name of the script
    - `script` - the script to be started by this launch button
  - **Returns** - the Launch object

- **Exceptions**
  - `scriptmib.ScriptMibException` - if a matching launch button already exists

- *installScript*
  `public Script` **installScript** `( java.lang.String` **owner**`, java.lang.String` **name**`, java.lang.String` **description**`, scriptmib.Language` **language**`, java.lang.String` **source** `)`
  - **Usage**
    - Create and install a Script object by a given owner and name. Also set the description, language, and source, and enable it.
  - **Parameters**
    - `owner` - the owner
    - `name` - the name of the script
    - `description` - the description string
    - `language` - the language this script is written in
    - `source` - the source URL (might be an empty string)
  - **Returns** - the Script object
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if a matching script already exists

- *setPollFactor*
  `public void` **setPollFactor** `( int` **factor** `)`
  - **Usage**
    - Set the percentage by which the poll interval increases between two polls. A value of 100 keeps the poll interval constant (default: 150).
  - **Parameters**
    - `factor` - the interval factor

- *setPollInterval*
  `public void` **setPollInterval** `( int` **interval** `)`
  - **Usage**
    - Set the start interval in milli seconds for methods of this ScriptMib object that perform SNMP polling operations (default: 100).
  - **Parameters**
    - `interval` - the interval in seconds

- *setPollTimeout*
  `public void` **setPollTimeout** `( int` **timeout** `)`
  - **Usage**
    - Set the maximum polling time in milli seconds for methods of this ScriptMib object that perform SNMP polling operations (default: 10000).
  - **Parameters**
    - `limit` - the maximum polling time in milli seconds

- *setRequest*
  `protected String` **setRequest**`( java.lang.String` **oid**`, long` **value** `)`

- *setRequest*
  `protected String` **setRequest**`( java.lang.String` **oid**`,`
  `java.lang.String` **value** `)`

- *setRequest*
  `protected String` **setRequest**`( java.util.Vector` **varbinds** `)`

3.2 *Class* ScriptMibException

3.2.1 Declaration

public class ScriptMibException
extends java.lang.Exception

3.2.2 Constructors

- *ScriptMibException*
  `public` **ScriptMibException**`( )`
- *ScriptMibException*
  `public` **ScriptMibException**`( java.lang.String` **msg** `)`

3.2.3 Methods inherited from class `java.lang.Exception`

3.2.4 Methods inherited from class `java.lang.Throwable`

- *fillInStackTrace*
  `public native Throwable` **fillInStackTrace**`( )`
- *getLocalizedMessage*
  `public String` **getLocalizedMessage**`( )`
- *getMessage*
  `public String` **getMessage**`( )`
- *printStackTrace*
  `public void` **printStackTrace**`( )`
- *printStackTrace*
  `public void` **printStackTrace**`( java.io.PrintStream )`
- *printStackTrace*
  `public void` **printStackTrace**`( java.io.PrintWriter )`
- *toString*
  `public String` **toString**`( )`

3.3 *Class* **Language**

3.3.1 Declaration

public class Language

extends java.lang.Object

3.3.2 Constructors

- *Language*
  `public` **Language**`( scriptmib.ScriptMib` **scriptMib**`, int` **index** `)`

  - **Usage**
    * Create a Language object for a given ScriptMib object and language index.

3.3.3 Methods

- *getDescription*
  `public String` **getDescription**`( )`

  - **Usage**
    * Return the description string (smLangDescr).

- *getIndex*
  `public int` **getIndex**`( )`

  - **Usage**
    * Return the unique index part of the Language used in the smLangTable (smLangIndex).

- *getOid*
  `public String` **getOid**`( )`

  - **Usage**
    * Return the object identifier of the Language object (smLangLanguage) as a dotted decimal string.

- *getRevision*
  `public String` **getRevision**`( )`

- **Usage**
  - Return the revision of the Language object (smLangRevision).

- *getVendor*
  `public String` **getVendor** `( )`
  - **Usage**
    - Return the vendor of the Language object (smLangVendor) as a dotted decimal string.

- *getVersion*
  `public String` **getVersion** `( )`
  - **Usage**
    - Return the version of the Language object (smLangVersion).

3.4 *Class* Extension

3.4.1 Declaration

public class Extension

extends java.lang.Object

3.4.2 Constructors

- *Extension*
  `public` **Extension** `( scriptmib.ScriptMib` **scriptMib**`, scriptmib.Language` **language**`, int` **index** `)`
  - **Usage**
    - Create an Extension object for a given ScriptMib object, Language object, and extension index.
  - **Parameters**
    - `scriptMib` - the ScriptMib object associated with this extension
    - `language` - the Language object associated with this extension
    - `index` - the table index of this extension

3.4.3 Methods

- *getDescription*
  `public String` **getDescription** `( )`
  - **Usage**
    - Return the description string (smExtsnDescr).
  - **Returns** - the description string of this extension

- *getIndex*
  `public int` **getIndex** `( )`
  - **Usage**
    - Return the unique index part of the Extension used in the smExtsnTable.
  - **Returns** - the extension's index

- *getOid*
  `public String` **getOid** `( )`

- **Usage**
  - Return the object identifier of the Extension object (smExtsnExtension) as a dotted decimal string.
- **Returns** - the OID that represents this extension

- *getRevision*
  `public String` **getRevision**`( )`
  - **Usage**
    - Return the revision of the Extension object (smExtsnRevision).
  - **Returns** - the revision of this extension

- *getVendor*
  `public String` **getVendor**`( )`
  - **Usage**
    - Return the vendor of the Extension object (smExtsnVendor) as a dotted decimal string.
  - **Returns** - the vendor of this extension

- *getVersion*
  `public String` **getVersion**`( )`
  - **Usage**
    - Return the version of the Extension object (smExtsnVersion).
  - **Returns** - the version of this extension

3.5 *Class* Script

3.5.1 Declaration

public class Script

extends java.lang.Object

3.5.2 Constructors

- *Script*
  `public` **Script**`( scriptmib.ScriptMib` **scriptMib**`, java.lang.String` **owner**`, java.lang.String` **name** `)`
  - **Usage**
    * Create a Script object for a given ScriptMib object indexed by a given owner and name by retrieval from the agent.
  - **Parameters**
    * `scriptMib` - the ScriptMib object associated with this script
    * `owner` - the script owner
    * `name` - the script name

- *Script*
  `public` **Script**`( scriptmib.ScriptMib` **scriptMib**`, java.lang.String` **owner**`, java.lang.String` **name**`, java.lang.String` **description**`, scriptmib.Language` **language**`, java.lang.String` **source** `)`
  - **Usage**
    * Create a Script object for a given ScriptMib object indexed by a given owner and name by creating it at the agent. Description, language, and source get also set, but the script does not get enabled.
  - **Parameters**
    * `scriptMib` - the ScriptMib object associated with this script
    * `owner` - the script owner
    * `name` - the script name
    * `description` - the script description string
    * `language` - the script language
    * `source` - the URL to fetch the script code

3.5.3 Methods

- *call*
  `public String` **call**`( )`
  - **Usage**
    * Start this script without arguments, block until it has been terminated and return the result string. The required launch button is created implicitly.
  - **Returns** - the result string
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if any of the subsequent actions cannot be finished in time

---

- *call*
  `public String` **call**`( java.lang.String` **argument** `)`
  - **Usage**
    * Start this script with a given argument, block until it has been terminated and return the result string. The required launch button is created and deleted implicitly.
  - **Parameters**
    * `agument` - the argument string to be passed to the script
  - **Returns** - the result string
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if any of the subsequent actions cannot be finished in time

---

- *disable*
  `public void` **disable**`( )`
  - **Usage**
    * Disable the script. Wait synchronously until the agent confirmed that the script is disabled.
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the script cannot be disabled in time

- *enable*
  `public void` **enable**`( )`
  - **Usage**
    * Enable this script. Wait synchronously until the agent confirmed that the script is enabled. Otherwise, a ScriptMibException will be thrown.
  - **Exceptions**
    * `scriptmib.ScriptMibException` - if the script cannot be enabled in time

- *getAdminStatus*
  `public int` **getAdminStatus**`( )`
  - **Usage**
    * Return an integer value representing the smScriptAdminStatus. This method should not be used by high level Script MIB applications.

– **Returns** - the script's smScriptAdminStatus value

- *getDescription*
  `public String` **getDescription** `( )`
  – **Usage**
    * Return the description string (smScriptDescr).
  – **Returns** - the script description string

- *getLanguage*
  `public Language` **getLanguage** `( )`
  – **Usage**
    * Return the Language object that represents the language this script is written in (smScriptLanguage).
  – **Returns** - the Language object representing the script's language
  – **Exceptions**
    * `scriptmib.ScriptMibException` - if the language cannot be found

- *getName*
  `public String` **getName** `( )`
  – **Usage**
    * Return the name of this Script (smScriptName).
  – **Returns** - the script name

- *getOperStatus*
  `public int` **getOperStatus** `( )`
  – **Usage**
    * Return an integer value representing the smScriptOperStatus. This method should not be used by high level Script MIB applications.
  – **Returns** - the script's smScriptOperStatus value

- *getOwner*
  `public String` **getOwner** `( )`
  – **Usage**
    * Return the owner of this Script (smScriptOwner).
  – **Returns** - the script owner

- *getRowStatus*
  `public int` **getRowStatus** `( )`
  – **Usage**
    * Return an integer value representing the smScriptRowStatus. This method should not be used by high level Script MIB applications.
  – **Returns** - the script's smScriptRowStatus value

- *getSource*
  `public String` **getSource** `( )`

  – **Usage**
    * Return the script source string (smScriptSource). Usually, it should be either an URL or an empty string.

  – **Returns** - the script source string

- *getStorageType*
  `public int` **getStorageType** `( )`

  – **Usage**
    * Return an integer value representing the smScriptStorageType. This method should not be used by high level Script MIB applications.

  – **Returns** - the script's smScriptStorageType value

- *isEnabled*
  `public boolean` **isEnabled** `( )`

  – **Usage**
    * Return true, if this script is enabled (smScriptOperStatus == enabled(1)).

  – **Returns** - true, if script is enabled

- *isNonVolatile*
  `public boolean` **isNonVolatile** `( )`

  – **Usage**
    * Return true, if this script is stored non-volatile at the agent. (smScriptStorageType == nonVolatile(3)).

  – **Returns** - true, if script is volatile

- *isPermanent*
  `public boolean` **isPermanent** `( )`

  – **Usage**
    * Return true, if this script is stored permanently at the agent. (smScriptStorageType == permanent(4)).

  – **Returns** - true, if script is permanent

- *isReadOnly*
  `public boolean` **isReadOnly** `( )`

  – **Usage**
    * Return true, if this script is stored read-only at the agent. (smScriptStorageType == readOnly(5)).

  – **Returns** - true, if script is read-only

- *isVolatile*
  `public boolean` **isVolatile**`( )`

  – **Usage**
    * Return true, if this script is stored volatile at the agent. (smScriptStorageType == volatile(2)).

  – **Returns** - true, if script is volatile

- *remove*
  `public void` **remove**`( )`

  – **Usage**
    * Remove this Script object. All Launch objects associated with this script get also removed implicitly. Similarly, all Run objects associated with those launch buttons get also aborted and removed implicitly.

  – **Exceptions**
    * `scriptmib.ScriptMibException` - if any of the subsequent actions cannot be finished in time

---

- *setAdminStatus*
  `public void` **setAdminStatus**`( int` **status** `)`

  – **Usage**
    * Set the smScriptAdminStatus represented by an integer value. This method should not be used by high level Script MIB applications.

  – **Parameters**
    * `the` - script's smScriptAdminStatus value

- *setDescription*
  `public void` **setDescription**`( java.lang.String` **description** `)`

  – **Usage**
    * Set the description string (smScriptDescr).

  – **Parameters**
    * `the` - script description string

- *setLanguage*
  `public void` **setLanguage**`( scriptmib.Language` **language** `)`

  – **Usage**
    * Set the Language object that represents the language this script is written in (smScriptLanguage).

  – **Parameters**
    * `the` - Language object representing the script's language

---

- *setNonVolatile*
  `public void` **setNonVolatile**`( )`

– **Usage**
  * Make this script stored non-volatile at the agent. (smScriptStorageType == nonVolatile(3)).

- *setPermanent*
  `public void` **setPermanent** `( )`
  – **Usage**
    * Make this script stored permanently at the agent. (smScriptStorageType == permanent(4)).

- *setReadOnly*
  `public void` **setReadOnly** `( )`
  – **Usage**
    * Make this script stored read-only at the agent. (smScriptStorageType == readOnly(5)).

- *setRowStatus*
  `public void` **setRowStatus** `( int` **status** `)`
  – **Usage**
    * Return an integer value representing the smScriptRowStatus. This method should not be used by high level Script MIB applications.
  – **Parameters**
    * `the` - script's smScriptRowStatus value

- *setSource*
  `public void` **setSource** `( java.lang.String` **source** `)`
  – **Usage**
    * Set the script source string (smScriptSource). Usually, it should be either an URL or an empty string.
  – **Parameters**
    * `the` - script source string

- *setStorageType*
  `public void` **setStorageType** `( int` **storageType** `)`
  – **Usage**
    * Set the smScriptStorageType represented by an integer value. This method should not be used by high level Script MIB applications.
  – **Parameters**
    * `the` - script's smScriptStorageType value

- *setVolatile*
  `public void` **setVolatile** `( )`
  – **Usage**
    * Make this script stored volatile at the agent. (smScriptStorageType == volatile(2)).

3.6 *Class* **Launch**

3.6.1 Declaration

public class Launch

extends java.lang.Object

3.6.2 Constructors

- *Launch*
  `public` **Launch**`( scriptmib.ScriptMib` **scriptMib**`, java.lang.String` **owner**`, java.lang.String` **name** `)`
  - **Usage**
    * Create a Launch object for a given ScriptMib object indexed by a given owner and name by retrieval from the agent.
  - **Parameters**
    * `scriptMib` - the ScriptMib object associated with this launch button
    * `owner` - the launch button owner
    * `name` - the launch button name

- *Launch*
  `public` **Launch**`( scriptmib.ScriptMib` **scriptMib**`, java.lang.String` **owner**`, java.lang.String` **name**`, scriptmib.Script` **script** `)`
  - **Usage**
    * Create a Launch object for a given ScriptMib object and a given Script object indexed by a given owner and name by creating it at the agent.
  - **Parameters**
    * `scriptMib` - the ScriptMib object associated with this launch button
    * `owner` - the launch button owner
    * `name` - the launch button name
    * `script` - the script to be launched by this launch button

3.6.3 Methods

- *call*
  `public String` **call**`( )`

- **Usage**
    - Start a script from this launch button, block until it has been terminated and return the result string.
- **Returns** - the result string
- **Exceptions**
    - `scriptmib.ScriptMibException` - if some actions cannot be finished in time

- *disable*
  `public void` **disable** `( )`
  - **Usage**
    - Disable the launch button. Wait synchronously until the agent confirmed that the launch button is disabled.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if disabling cannot be finished in time

- *enable*
  `public void` **enable** `( )`
  - **Usage**
    - Enable this launch button. Wait synchronously until the agent confirmed that the launch button is enabled. Otherwise, a ScriptMibException will be thrown.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if enabling cannot be finished in time

- *getAdminStatus*
  `public int` **getAdminStatus** `( )`
  - **Usage**
    - Return an integer value representing the smLaunchAdminStatus. This method should not be used by high level Script MIB applications.
  - **Returns** - the smLaunchAdminStatus value

- *getArgument*
  `public String` **getArgument** `( )`
  - **Usage**
    - Return the argument string to be passed to the script launched by this launch button (smLaunchArgument).
  - **Returns** - the argument string

- *getControl*
  `public int` **getControl** `( )`
  - **Usage**
    - Return an integer value representing the smLaunchControl value. This method should not be used by high level Script MIB applications.

– **Returns** - the smLaunchControl value

- *getExpireTime*
`public long` **getExpireTime**`( )`
  – **Usage**
    * Return the maximum amount of time in milli seconds that terminated scripts started from this launch button may be retained as valid Run objects (smLaunch-ExpireTime).
  – **Returns** - the expire time

- *getLifeTime*
`public long` **getLifeTime**`( )`
  – **Usage**
    * Return the maximum amount of time in milli seconds that scripts started from this launch button may run (smLaunchLifeTime).
  – **Returns** - the maximum life time

- *getMaxCompleted*
`public long` **getMaxCompleted**`( )`
  – **Usage**
    * Return the maximum number of terminated scripts started from this launch button and retained as valid Run objects (smLaunchMaxCompleted).
  – **Returns** - the maximum number of completed scripts

- *getMaxRunning*
`public long` **getMaxRunning**`( )`
  – **Usage**
    * Return the maximum number of running scripts started from this launch button (smLaunchMaxRunning).
  – **Returns** - the maximum number of running scripts

- *getName*
`public String` **getName**`( )`
  – **Usage**
    * Return the name of this Launch object (smLaunchName).
  – **Returns** - the launch button name

- *getOperStatus*
`public int` **getOperStatus**`( )`
  – **Usage**
    * Return an integer value representing the smLaunchOperStatus. This method should not be used by high level Script MIB applications.

– **Returns** - the smScriptOperStatus value

- *getOwner*
`public String` **getOwner** `( )`

  – **Usage**
    * Return the owner of this Launch object (smScriptOwner).

  – **Returns** - the launch button owner

- *getRowStatus*
`public int` **getRowStatus** `( )`

  – **Usage**
    * Return an integer value representing the smLaunchRowStatus. This method should not be used by high level Script MIB applications.

  – **Returns** - the smLaunchRowStatus value

- *getRunIndexNext*
`public int` **getRunIndexNext** `( )`

  – **Usage**
    * Return an integer value representing the smLaunchRunIndexNext. This method should not be used by high level Script MIB applications.

  – **Returns** - the smLaunchRunIndexNext value

- *getScript*
`public Script` **getScript** `( )`

  – **Usage**
    * Return the Script object that may be started by this Launch object.

  – **Returns** - the script launched by this launch button

- *getScriptName*
`public String` **getScriptName** `( )`

  – **Usage**
    * Return the script name (smLaunchScriptName).

  – **Returns** - the name of the script launched by this launch button

---

- *getScriptOwner*
`public String` **getScriptOwner** `( )`

  – **Usage**
    * Return the script owner (smLaunchScriptOwner).

  – **Returns** - the owner of the script launched by this launch button

---

- *getStart*
`public int` **getStart** `( )`

- **Usage**
  - Return an integer value representing the smLaunchStart value. This method should not be used by high level Script MIB applications.
- **Returns** - the smLaunchStart value

- *getStorageType*
  `public int` **getStorageType** `( )`
  - **Usage**
    - Return an integer value representing the smLaunchStorageType. This method should not be used by high level Script MIB applications.
  - **Returns** - the smLaunchStorageType value

- *isEnabled*
  `public boolean` **isEnabled** `( )`
  - **Usage**
    - Return true, if this script is enabled (smScriptOperStatus == enabled(1)).
  - **Parameters**
    - `the` - smScriptOperStatus value

- *isNonVolatile*
  `public boolean` **isNonVolatile** `( )`
  - **Usage**
    - Return true, if this launch button is stored non-volatile at the agent. (smLaunch-StorageType == nonVolatile(3)).
  - **Returns** - true, if launch button is volatile

- *isPermanent*
  `public boolean` **isPermanent** `( )`
  - **Usage**
    - Return true, if this launch button is stored permanently at the agent. (smLaunch-StorageType == permanent(4)).
  - **Returns** - true, if launch button is permanent

- *isReadOnly*
  `public boolean` **isReadOnly** `( )`
  - **Usage**
    - Return true, if this launch button is stored read-only at the agent. (smLaunchStorageType == readOnly(5)).
  - **Returns** - true, if launch button is read-only

- *isVolatile*
  `public boolean` **isVolatile** `( )`

- **Usage**
  - Return true, if this launch button is stored volatile at the agent. (smLaunchStorageType == volatile(2)).
- **Returns** - true, if launch button is volatile

- *remove*
  `public void` **remove**`( )`
  - **Usage**
    - Remove this Launch object. All Run objects associated with this launch button get also aborted and removed implicitly.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if the launch button cannot be disabled in time

- *setAdminStatus*
  `public void` **setAdminStatus**`( int` **status** `)`
  - **Usage**
    - Set the smLaunchAdminStatus represented by an integer value. This method should not be used by high level Script MIB applications.
  - **Parameters**
    - `the` - smLaunchAdminStatus value

- *setArgument*
  `public void` **setArgument**`( java.lang.String` **argument** `)`
  - **Usage**
    - Set the argument string to be passed to the script launched by this launch button (smLaunchArgument).
  - **Parameters**
    - `the` - argument string

- *setControl*
  `public void` **setControl**`( int` **control** `)`
  - **Usage**
    - Set the smLaunchControl value to a given integer value. This method should not be used by high level Script MIB applications.
  - **Parameters**
    - `the` - smLaunchControl value

- *setExpireTime*
  `public void` **setExpireTime**`( long` **time** `)`
  - **Usage**

    * Set the maximum amount of time in milli seconds that terminated scripts started from this launch button may be retained as valid Run objects (smLaunchExpireTime).
  – **Parameters**
    * `the` - expire time

- *setLifeTime*
  `public void` **setLifeTime** `( long` **time** `)`
  – **Usage**
    * Set the maximum amount of time in milli seconds that scripts started from this launch button may run (smLaunchLifeTime).
  – **Parameters**
    * `the` - maximum life time

- *setMaxCompleted*
  `public void` **setMaxCompleted** `( long` **max** `)`
  – **Usage**
    * Set the maximum number of terminated scripts started from this launch button and retained as valid Run objects (smLaunchMaxCompleted).
  – **Parameters**
    * `the` - maximum number of completed scripts

- *setMaxRunning*
  `public void` **setMaxRunning** `( long` **max** `)`
  – **Usage**
    * Set the maximum number of running scripts started from this launch button (smLaunchMaxRunning).
  – **Parameters**
    * `the` - maximum number of running scripts

- *setNonVolatile*
  `public void` **setNonVolatile** `( )`
  – **Usage**
    * Make this launch button stored non-volatile at the agent. (smLaunchStorageType == nonVolatile(3)).

- *setPermanent*
  `public void` **setPermanent** `( )`
  – **Usage**
    * Make this launch button stored permanently at the agent. (smLaunchStorageType == permanent(4)).

- *setReadOnly*
  `public void` **setReadOnly** `( )`
  - **Usage**
    - Make this launch button stored read-only at the agent. (smLaunchStorageType == readOnly(5)).

- *setRowStatus*
  `public void` **setRowStatus** `( int` **status** `)`
  - **Usage**
    - Return an integer value representing the smLaunchRowStatus. This method should not be used by high level Script MIB applications.
  - **Parameters**
    - `the` - smLaunchRowStatus value

- *setScript*
  `public void` **setScript** `( scriptmib.Script` **script** `)`
  - **Usage**
    - Set the Script object that may be started by this Launch object.
  - **Parameters**
    - `the` - script to be launched by this launch button

- *setScriptName*
  `public void` **setScriptName** `( java.lang.String` **scriptname** `)`
  - **Usage**
    - Set the script name (smLaunchScriptName).
  - **Parameters**
    - `the` - name of the script to be launched by this launch button

---

- *setScriptOwner*
  `public void` **setScriptOwner** `( java.lang.String` **scriptowner** `)`
  - **Usage**
    - Set the script owner (smLaunchScriptOwner).
  - **Parameters**
    - `the` - owner of the script to be launched by this launch button

---

- *setStart*
  `public void` **setStart** `( int` **runindex** `)`
  - **Usage**
    - Set the smLaunchStart value to a given integer value. This method should not be used by high level Script MIB applications.
  - **Parameters**
    - `the` - smLaunchStart value

- *setStorageType*
  `public void` **setStorageType**`( int` **storageType** `)`

  – **Usage**
    * Set the smLaunchStorageType represented by an integer value. This method should not be used by high level Script MIB applications.

  – **Parameters**
    * `the` - smLaunchStorageType value

- *setVolatile*
  `public void` **setVolatile**`( )`

  – **Usage**
    * Make this launch button stored volatile at the agent. (smLaunchStorageType == volatile(2)).

- *start*
  `public Run` **start**`( )`

  – **Usage**
    * Start a script from this launch button and return the created Run object.

  – **Returns** - the created Run object

  – **Exceptions**
    * `scriptmib.ScriptMibException` - if script initialization cannot be finished in time

---

- *start*
  `public Run` **start**`( int` **runindex** `)`

  – **Usage**
    * Start a script with a given run index from this launch button and return the created Run object.

  – **Parameters**
    * `runindex` - the run index to be used

  – **Returns** - the created Run object

  – **Exceptions**
    * `scriptmib.ScriptMibException` - if script initialization cannot be finished in time

3.7 *Class* **Run**

3.7.1 Declaration

public class Run

extends java.lang.Object

3.7.2 Constructors

- *Run*
  `public` **Run**`( scriptmib.ScriptMib` **scriptMib**`, scriptmib.Launch` **launch**`, int` **index** `)`
  - **Usage**
    - Create a Run object for a given ScriptMib object indexed by the owner and name of a given Launch object, and a given index by retrieval from the agent.
  - **Parameters**
    - `scriptMib` - the ScriptMib object associated with this Run object
    - `launch` - the Launch object
    - `index` - the run index related to the launch button

- *Run*
  `public` **Run**`( scriptmib.ScriptMib` **scriptMib**`, java.lang.String` **owner**`, java.lang.String` **name**`, int` **index** `)`
  - **Usage**
    - Create a Run object for a given ScriptMib object indexed by a given launch button owner, launch button name, and index by retrieval from the agent.
  - **Parameters**
    - `scriptMib` - the ScriptMib object associated with this Run object
    - `owner` - the launch button owner
    - `name` - the launch button name
    - `index` - the run index related to the launch button

3.7.3 Methods

- *abort*
  `public void` **abort**`( )`

- **Usage**
  - Abort the script.
- **Exceptions**
  - `scriptmib.ScriptMibException` - if the script cannot be aborted in time

- *block*
  `public void` **block** `( )`
  - **Usage**
    - Block until this script has been terminated.

- *getArgument*
  `public String` **getArgument** `( )`
  - **Usage**
    - Return the argument string (smRunArgument).
  - **Returns** - the argument string

- *getEnd*
  `public Date` **getEnd** `( )`
  - **Usage**
    - Return the end time as a Date object (smRunEndTime).
  - **Returns** - Date object representing the script's end time, or null if the script is not yet terminated

---

- *getEndTime*
  `public int` **getEndTime** `( )`
  - **Usage**
    - Return the end time as an array of integer values representing the MIB value's octets (smRunEndTime). This method should not be used by high level Script MIB applications.
  - **Returns** - the array of octets of the smRunEndTime

- *getError*
  `public String` **getError** `( )`
  - **Usage**
    - Return the error string (smRunError).
  - **Returns** - the error string

- *getExitCode*
  `public int` **getExitCode** `( )`
  - **Usage**
    - Return the exit code (smRunExitCode).
  - **Returns** - the exit code

- *getExpireTime*
  `public long` **getExpireTime** `( )`
  - **Usage**
    - Return the script's expire time after termination in milli seconds (smRunExpireTime).
  - **Returns** - the expire time in milli seconds

- *getIndex*
  `public int` **getIndex** `( )`
  - **Usage**
    - Return the index of this Run object according to the launch button from which this Run object has been created (smRunIndex).
  - **Returns** - the run index

- *getLaunch*
  `public Launch` **getLaunch** `( )`
  - **Usage**
    - Return the Launch object from which this Run object has been created.
  - **Returns** - the Launch object

- *getLifeTime*
  `public long` **getLifeTime** `( )`
  - **Usage**
    - Return the script's life time in milli seconds (smRunLifeTime).
  - **Returns** - the life time in milli seconds

- *getName*
  `public String` **getName** `( )`
  - **Usage**
    - Return the name of the launch button from which this Run object has been created (smLaunchName).
  - **Returns** - the launch button name

- *getOwner*
  `public String` **getOwner** `( )`
  - **Usage**
    - Return the owner of the launch button from which this Run object has been created (smLaunchOwner).
  - **Returns** - the launch button owner

- *getResult*
  `public String` **getResult** `( )`

- **Usage**
  - Return the result string (smRunResult).
- **Returns** - the result string

- *getScript*
  `public Script` **getScript** `( )`

  - **Usage**
    - Return the Script object that is executed by this Run object.
  - **Returns** - the Script object

- *getStart*
  `public Date` **getStart** `( )`

  - **Usage**
    - Return the start time as a Date object (smRunStartTime).
  - **Returns** - Date object representing the script's start time

---

- *getStartTime*
  `public int` **getStartTime** `( )`

  - **Usage**
    - Return the start time as an array of integer values representing the MIB value's octets (smRunStartTime). This method should not be used by high level Script MIB applications.
  - **Returns** - the array of octets of the smRunStartTime

- *getState*
  `public int` **getState** `( )`

  - **Usage**
    - Return an integer value representing the smRunState. This method should not be used by high level Script MIB applications.
  - **Returns** - the smRunState value

- *isExecuting*
  `public boolean` **isExecuting** `( )`

  - **Usage**
    - Return true, if this script is executing (smRunState == executing(2).
  - **Returns** - true, if script is executing

- *isInitializing*
  `public boolean` **isInitializing** `( )`

  - **Usage**
    - Return true, if this script is initializing (smRunState == initializing(1).
  - **Returns** - true, if script is initializing

- *isSuspended*
  `public boolean` **isSuspended** `( )`
  - **Usage**
    - Return true, if this script is suspended (smRunState == suspended(4).
  - **Returns** - true, if script is suspended

- *isTerminated*
  `public boolean` **isTerminated** `( )`
  - **Usage**
    - Return true, if this script is terminated (smRunState == terminated(7).
  - **Returns** - true, if script is terminated

- *remove*
  `public void` **remove** `( )`
  - **Usage**
    - Remove this Run object.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if the script cannot be aborted in time

- *resume*
  `public void` **resume** `( )`
  - **Usage**
    - Resume the script.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if the script cannot be resumed in time

- *setControl*
  `public void` **setControl** `( int` **control** `)`
  - **Usage**
    - Set the smRunControl value to a given integer value. This method should not be used by high level Script MIB applications.

- *setExpireTime*
  `public void` **setExpireTime** `( long` **time** `)`
  - **Usage**
    - Set the script's expire time after termination in milli seconds (smRunExpireTime).
  - **Returns** - the expire time in milli seconds

- *setLifeTime*
  `public void` **setLifeTime** `( long` **time** `)`

- **Usage**
  - Set the script's life time in milli seconds (smRunLifeTime).
- **Parameters**
  - `the` - life time in milli seconds

- *suspend*
  `public void` **suspend** `( )`
  - **Usage**
    - Suspend the script.
  - **Exceptions**
    - `scriptmib.ScriptMibException` - if the script cannot be suspended in time