NAME

smiGetNode, smiGetNodeByOID, smiGetFirstNode, smiGetNextNode, smiGetParentNode, smiGetRelatedNode, smiGetFirstChildNode, smiGetNextChildNode, smiGetNodeModule, smiGetNodeType, smiGetNodeLine, smiGetFirstElement, smiGetNextElement, smiGetElementNode, smiGetFirstOption, smiGetNextOption, smiGetOptionNode, smiGetFirstRefinement, smiGetNextRefinement, smiGetRefinementModule, smiGetRefinementNode, smiGetRefinementType, smiGetRefinementWriteType − SMI type information routines

SYNOPSIS

#include <smi.h>

SmiNode *smiGetNode(SmiModule * *smiModulePtr***, char *** *node***);**

SmiNode *smiGetNodeByOID(unsigned int *oidlen***, SmiSubid** *oid[]***);**

SmiNode *smiGetFirstNode(SmiModule * *smiModulePtr***, SmiNodekind** *kinds***);**

SmiNode *smiGetNextNode(SmiNode * *smiNodePtr***, SmiNodekind** *kinds***);**

SmiNode *smiGetParentNode(SmiNode * *smiNodePtr***);**

SmiNode *smiGetRelatedNode(SmiNode * *smiNodePtr***);**

SmiNode *smiGetFirstChildNode(SmiNode * *smiNodePtr***);**

SmiNode *smiGetNextChildNode(SmiNode * *smiNodePtr***);**

SmiModule *smiGetNodeModule(SmiNode * *smiNodePtr***);**

SmiType *smiGetNodeType(SmiNode * *smiNodePtr***);**

int smiGetNodeLine(SmiNode * *smiNodePtr***);**

SmiElement *smiGetFirstElement(SmiNode * *smiNodePtr***);**

SmiElement *smiGetNextElement(SmiElement * *smiElementPtr***);**

SmiNode *smiGetElementNode(SmiElement * *smiElementPtr***);**

SmiOption *smiGetFirstOption(SmiNode * *smiComplianceNodePtr***);**

SmiOption *smiGetNextOption(SmiOption * *smiOptionPtr***);**

SmiNode *smiGetOptionNode(SmiOption * *smiOptionPtr***);**

SmiRefinement *smiGetFirstRefinement(SmiNode * *smiComplianceNodePtr***);**

SmiRefinement *smiGetNextRefinement(SmiRefinement * *smiRefinementPtr***);**

SmiNode *smiGetRefinementNode(SmiRefinement * *smiRefinementPtr***);**

SmiType *smiGetRefinementType(SmiRefinement * *smiRefinementPtr***);**

SmiType *smiGetRefinementWriteType(SmiRefinement * *smiRefinementPtr***);**

```
typedef struct SmiNode {
  SmiIdentifier     name;
  int               oidlen;
  SmiSubid          *oid;       /* array of length oidlen */
```

```
    SmiDecl         decl;
    SmiAccess       access;
    SmiStatus       status;
    char            *format;
    SmiValue        value;
    char            *units;
    char            *description;
    char            *reference;
    SmiIndexkind    indexkind;
    int             implied;
    int             create;
    SmiNodekind     nodekind;
} SmiNode;

typedef struct SmiElement {
    /* no visible attributes */
} SmiElement;

typedef struct SmiOption {
    char            *description;
} SmiOption;

typedef struct SmiRefinement {
    SmiAccess       access;
    char            *description;
} SmiRefinement;
```

DESCRIPTION

These functions retrieve information on any SMI node definition in the object identifier tree, these are ASN.1 object identifier assignments, MODULE-IDENTITYs, OBJECT-IDENTITYs, OBJECT-TYPEs, NOTIFICATION-TYPEs, TRAP-TYPEs, OBJECT-GROUPs, NOTIFICATION-GROUPs, MODULE-COMPLIANCEs, and AGENT-CAPABILITYs in SMIv1/v2 and node, scalar, table, row, column, notification, group, and compliance statements in SMIng.

The **smiGetNode()** function retrieves a **struct SmiNode** that represents a node of any kind. *Node* may be either a fully qualified descriptor, a simple node name, or a numerical OID. Nodes are also found, if *node* contains an instance identifier suffix. If *smiModulePtr* is not NULL it used to limit the search to the given module. If the node is not found, **smiGetNode()** returns NULL.

The **smiGetNodeByOID()** function retrieves a **struct SmiNode** that matches the longest prefix of the node that is specified by the object identifier *oid[]* with the length *oidlen*. If no such node is not found, **smiGetNodeByOID()** returns NULL.

The **smiGetFirstNode()** and **smiGetNextNode()** functions are used to iteratively retrieve **struct SmiNode**s in tree pre-order. **smiGetFirstNode()** returns the first node defined in the module specified by *smiModulePtr* that is of any kind specified in the *kinds* bitset. Subsequent calls to **smiGetNextNode()** return the next node of any kind specified in the *kinds* bitset. If there are no more node definitions in the module, NULL is returned.

The **smiGetFirstChildNode()** and **smiGetNextChildNode()** functions are used to iteratively retrieve **struct SmiNode**s that represent the immediate child nodes of the node specified by *smiNodePtr* passed to the **smiGetFirstChildNode()** call.

The **smiGetParentNode()** function is used to retrieve a **struct SmiNode**s that represents the parent node of the node specified by *smiNodePtr*.

The **smiGetRelatedNode()** function is used to retrieve a **struct SmiNode**s that is related to the node specified by *smiNodePtr*. Actually, this is used for SMIv2 table augmentation entries and similar SMIng constructs.

The **smiGetNodeModule()** function returns the module that defines the node given by *struct SmiNodePtr*.

The **smiGetNodeType()** function returns the type of the (scalar or columnar) node given by *struct SmiNodePtr*. If *struct SmiNodePtr* does not specify a scalar or columnar node, NULL is returned.

The **smiGetFirstElement()** and **smiGetNextElement()** functions are used to iteratively retrieve **struct SmiElement**s that represent elements of index clauses or notification object lists, groups of object types or notification types, and mandatory groups of module compliance statements. The node to which the list belongs has to be specified by *smiNodePtr*. To retrieve the node that is represented by a **struct SmiElement**, the **smiGetElementNode()** function has to be called.

The **smiGetFirstOption()** and **smiGetNextOption()** functions are used to iteratively retrieve **struct SmiOption**s that represent statements on optional (object or notification) groups within the compliance statement specified by *smiComplianceNodePtr*. The group node which is subject of such a statement can be retrieved by the **smiGetOptionNode()** function.

Similarly, the **smiGetFirstRefinement()** and **smiGetNextRefinement()** functions are used to iteratively retrieve **struct SmiRefinement**s that represent statements on optional object refinements within the compliance statement specified by *smiComplianceNodePtr*. The node which is subject of such a refinement can be retrieved by the **smiGetRefinementNode()** function. The optional refined type and write-type of a refinement can be retrieved by the **smiGetRefinementType()** and **smiGetRefinementWriteType()** functions. If they are not present, NULL is returned.

The **smiGetNodeLine()** function returns the line number within the module where the node specified by *smiNodePtr* is defined.

FILES

${prefix}/include/smi.h SMI library header file

SEE ALSO

libsmi(3), **smi_config**(3), **smi_type**(3), **smi_module**(3), **smi.h**

AUTHOR

(C) 1999-2004 Frank Strauss, TU Braunschweig, Germany <strauss AT ibr DOT cs DOT tu-bs DOT de>