CIFOBJ
A Class Library of
mmCIF Access Tools
Reference Guide

CIFOBJ Version 3.01

Based on Dictionary Description Language v. 2.1
July 1996

Steven Schirripa
John D. Westbrook
Nucleic Acid Database Project
Department of Chemistry and Chemical Biology
Rutgers, The State University of New Jersey

Please direct comments on this document to sw-help@rcsb.rutgers.edu.

CIFOBJ - A Class Library of mmCIF Access Tools

Contents

1 Introduction

The CIFOBJ class library was developed in order to provide an object view of the mmCIF dictionary. The class library has two components as illustrated in Figure 1. The first component builds a persistent store of objects of type: item, sub-category, category, and dictionary. Each object is a container for all relevant attributes for that object type. CIBOBJ accesses and checks the dictionary contents using methods provided by CIFLIB. The CIFOBJ loader class assembles the dictionary objects and passes these to the object storage manager. The second component of the CIFOBJ class library provides methods for building dictionary objects from the persistent store. CIFOBJ implements methods to access all of the attributes for each object type and returns this information as a string or an array of strings.

- reads and writes mmCIF dictionary objects.
- provides read and store access to individual dictionary attributes.
- provides efficient persistent storage management of dictionary objects.
- provides a stable callable interface in the C++ programming language.

This document describes the public interface to CIFOBJ.

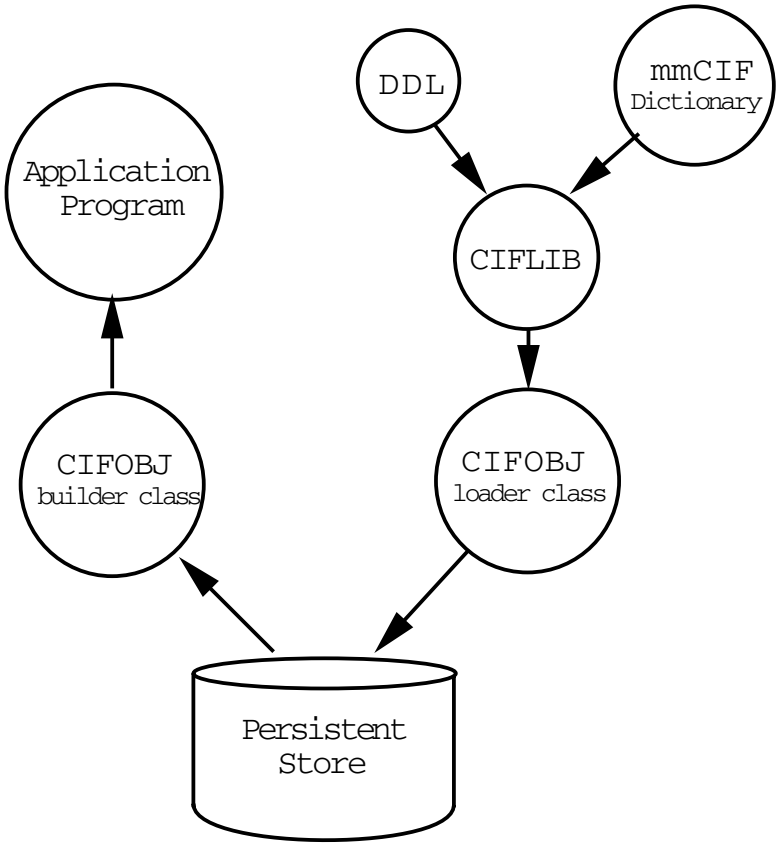

Figure 1: Functional diagram of the CIFOBJ class library.

2 Method Description

The following section describes the public interface methods provided by the CifBuilder class. It has been subdivided into six sections: general methods, item access methods, category access methods, subcategory access methods, dictionary access methods, and debugging methods.

2.1 General Methods

This section deals with basic methods that are necessary to call before using methods from any other section. These methods provide basic initialization of the CifBuilder class.

2.1.1 CifBuilder

NAME

CifBuilder

PROTOTYPE

```
#include "CifBuilder.h"

CifBuilder::CifBuilder(char * filename);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");
```

PURPOSE

CifBuilder is a constructor for the CifBuilder class

RECEIVES

`filename`	the name of the persistent storage file to be used in building the cifobjs

RETURN VALUE

None

REMARKS

See also: *OpenFile*

2.1.2 OpenFile

NAME

OpenFile

PROTOTYPE

```
#include "CifBuilder.h"

int CifBuilder::OpenFile(char * filename);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb();

cb.OpenFile("mmCif96.psf");
```

PURPOSE

OpenFile opens the persistent storage file named by filename and associates the CifBuilder object with it

RECEIVES

`filename`	the name of the persistent storage file to be used in building the cifobjs

RETURN VALUE

Returns an integer error code, with NO_ERROR upon success and a negative value for failure

REMARKS

See also: *CifBuilder*

2.1.3 BuildDictionaryList

NAME

BuildDictionaryList

PROTOTYPE

```
#include "CifBuilder.h"

int CifBuilder::BuildDictionaryList();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
```

PURPOSE

BuildDictionaryList is used to build the index used in reconstructing dictionary datablock objects from the persistent store. *This method must be called before any other methods that retrieve data from the persistent storage file.*

RECEIVES

`No Arguments`

RETURN VALUE

Returns an integer error code, with NO_ERROR upon success and a negative value for failure

REMARKS

None

2.1.4 SetContext

NAME

SetContext

PROTOTYPE

```
#include "CifBuilder.h"

void CifBuilder::SetContext(const char * dataBlockName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
```

PURPOSE

SetContext sets the current searchable dictionary datablock to *dataBlockName*, which must occur in the dictionary list built by BuildDictionaryList

RECEIVES

`dataBlockName`	the context into which subsequent searches will be made, i.e. the current dictionary datablock's name

RETURN VALUE

None

REMARKS

None

2.1.5 GetDictionaryNames

NAME

GetDictionaryNames

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryNames();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
char ** dictNames = cb.GetDictionaryNames();
```

PURPOSE

GetDictionaryNames returns an array of the dictionary datablock names in the persistent storage file

RECEIVES

```
No Arguments
```

RETURN VALUE

Returns an array of dictionary datablock names or a NULL value for failure. The number of names returned in the array can be retrieved by a call to CountDictionaries

REMARKS

See also: *CountDictionaries*

2.1.6 CountDictionaries

NAME

CountDictionaries

PROTOTYPE

```
#include "CifBuilder.h"

uWord CifBuilder::CountDictionaries();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
uWord numDict = cb.CountDictionaries();
```

PURPOSE

CountDictionaries returns the number of dictionary names in the persistent storage file. This will be the number of names returned by GetDictionaryNames.

RECEIVES

`No Arguments`

RETURN VALUE

Returns a uWord representing the number of dictionary datablocks in the persistent storage file.

REMARKS

See also: *GetDictionaryNames*

2.2 Item Access Methods

These methods provide context-dependent access into item objects via the dictionary object that defines the item.

2.2.1 GetItemDescription

NAME

GetItemDescription

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemDescription(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * description = cb.GetItemDescription("_atom_site.cartn_x");
```

PURPOSE

GetItemDescription returns the description for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the description as a char * or NULL.

REMARKS

None

2.2.2 GetItemType

NAME

GetItemType

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemType(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * type = cb.GetItemType("_atom_site.cartn_x");
```

PURPOSE

GetItemType returns the type for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the type as a char * or NULL.

REMARKS

None

2.2.3 GetItemPrimitiveCode

NAME

GetItemPrimitiveCode

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemPrimitiveCode(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * primitiveCode = cb.GetItemPrimitiveCode("_atom_site.cartn_x");
```

PURPOSE

GetItemPrimitiveCode returns the primitive code for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the primitive code as a char * or NULL.

REMARKS

None

2.2.4 GetItemRegex

NAME

GetItemRegex

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemRegex(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * regularExpression = cb.GetItemRegex("_atom_site.cartn_x");
```

PURPOSE

GetItemRegex returns the regular expression for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the regular expression as a char * or NULL.

REMARKS

None

2.2.5 GetItemMandatoryCode

NAME

GetItemMandatoryCode

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemMandatoryCode(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * mandatory = cb.GetItemMandatoryCode("_atom_site.cartn_x");
```

PURPOSE

GetItemMandatoryCode returns the mandatory code for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the mandatory code as a char * or NULL.

REMARKS

None

2.2.6 GetItemDefaultValue

NAME

GetItemDefaultValue

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemDefaultValue(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * default = cb.GetItemDefaultValue("_atom_site.cartn_x");
```

PURPOSE

GetItemDefaultValue returns the default value for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the default value as a char * or NULL.

REMARKS

None

2.2.7 GetItemUnits

NAME

GetItemUnits

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemUnits(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * units = cb.GetItemUnits("_atom_site.cartn_x");
```

PURPOSE

GetItemUnits returns the units for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the units as a char * or NULL.

REMARKS

None

2.2.8 GetItemItemStructure

NAME

GetItemItemStructure

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemItemStructure(const char * itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * iStructure = cb.GetItemItemStructure("_atom_site.cartn_x");
```

PURPOSE

GetItemItemStructure returns the item structure for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the item structure as a char * or NULL.

REMARKS

See also: *GetItemItemStructureOrganization*

2.2.9 GetItemItemStructureOrganization

NAME

GetItemItemStructureOrganization

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetItemItemStructureOrganization(const char *
                                                    itemName);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");
char * iStructureOrg =
       cb.GetItemItemStructureOrganization("_atom_site.cartn_x");
```

PURPOSE

GetItemItemStructureOrganization returns the item structure organization for the item *itemName* in the current context

RECEIVES

`itemName`	the name of the item

RETURN VALUE

Returns the item structure organization as a char * or NULL.

REMARKS

See also *GetItemItemStructure*

2.2.10 GetItemExamplesCase

NAME

GetItemExamplesCase

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemExamplesCase(const char * itemName,
                                        uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examples = cb.GetItemExamplesCase("_atom_site.cartn_x",
                                          numExamples);
```

PURPOSE

GetItemExamplesCase returns an array of examples for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the examples as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetItemExamplesDetail*

2.2.11 GetItemExamplesDetail

NAME

GetItemExamplesDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemExamplesDetail(const char * itemName,
                                          uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examplesDetail =
 cb.GetItemExamplesDetail("_atom_site.cartn_x", numExamples);
```

PURPOSE

GetItemExamplesDetail returns an array of example details for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the example details as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetItemExamplesCase*

2.2.12 GetItemEnumeration

NAME

GetItemEnumeration

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemEnumeration(const char * itemName,
                                       uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numEnumeration = 0;
char ** enumeration = cb.GetItemEnumeration("_atom_site.cartn_x",
                                            numEnumeration);
```

PURPOSE

GetItemEnumeration returns an enumeration for the item *itemName* as an array of char *.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the length of the enumeration

RETURN VALUE

Returns the enumeration as a char ** or NULL. The length of the enumeration will be passed back via the second argument.

REMARKS

See also: *GetItemEnumerationDetail*

2.2.13 GetItemEnumerationDetail

NAME

GetItemEnumerationDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemEnumerationDetail(const char * itemName,
                                             uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numEnumeration = 0;
char ** enumDetail =
 cb.GetItemEnumerationDetail("_atom_site.cartn_x", numEnumeration);
```

PURPOSE

GetItemEnumerationDetail returns an array of descriptions for the enumeration of the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of enumeration details

RETURN VALUE

Returns the enumeration details as a char ** or NULL. The number of entries in the enumeration will be passed back via the second argument.

REMARKS

See also: *GetItemEnumeration*

2.2.14 GetItemRangeMin

NAME

GetItemRangeMin

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemRangeMin(const char * itemName,
                                    uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numRange = 0;
char ** min = cb.GetItemRangeMin("_atom_site.cartn_x", numRange);
```

PURPOSE

GetItemRangeMin returns an array containing the range minimums for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of range minimums

RETURN VALUE

Returns the range minimums as a char ** or NULL. The number of minimums will be passed back via the second argument.

REMARKS

See also: *GetItemRangeMax*

2.2.15 GetItemRangeMax

NAME

GetItemRangeMax

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemRangeMax(const char * itemName,
                                    uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numRange = 0;
char ** max = cb.GetItemRangeMax("_atom_site.cartn_x", numRange);
```

PURPOSE

GetItemRangeMax returns an array containing the range maximums for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of range maximums

RETURN VALUE

Returns the range maximums as a char ** or NULL. The number of maximums will be passed back via the second argument.

REMARKS

See also: *GetItemRangeMin*

2.2.16 GetItemAliases

NAME

GetItemAliases

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemAliases(const char * itemName,
                                   uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numAliases = 0;
char ** aliases = cb.GetItemAliases("_atom_site.cartn_x",
                                    numAliases);
```

PURPOSE

GetItemAliases returns an array containing the aliases for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of aliases

RETURN VALUE

Returns the aliases as a char ** or NULL. The number of aliases will be passed back via the second argument.

REMARKS

See also: *GetItemAliasesDictionary*
GetItemAliasesDictionaryVersion

2.2.17 GetItemAliasesDictionary

NAME

GetItemAliasesDictionary

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemAliasesDictionary(const char * itemName,
                                             uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numAliases = 0;
char ** aliasDicts =
 cb.GetItemAliasesDictionary("_atom_site.cartn_x", numAliases);
```

PURPOSE

GetItemAliasesDictionary returns an array containing the names of the dictionaries from which the aliases of the item *itemName* came.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of aliases

RETURN VALUE

Returns the alias dictionary names as a char ** or NULL. The number of aliases will be passed back via the second argument.

REMARKS

See also: *GetItemAliases*
GetItemAliasesDictionaryVersion

2.2.18 GetItemAliasesDictionaryVersion

NAME

GetItemAliasesDictionaryVersion

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemAliasesDictionaryVersion(const char *
                                                    itemName,
                                                    uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numAliases = 0;
char ** aliasDictVersions =
  cb.GetItemAliasesDictionaryVersion("_atom_site.cartn_x",
                                     numAliases);
```

PURPOSE

GetItemAliasesDictionaryVersion returns an array containing the versions of the dictionaries from which the aliases of the item *itemName* came.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of aliases

RETURN VALUE

Returns the alias dictionary versions as a char ** or NULL. The number of aliases will be passed back via the second argument.

REMARKS

See also: *GetItemAliases*
GetItemAliasesDictionary

2.2.19 GetItemDependents

NAME

GetItemDependents

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemDependents(const char * itemName,
                                      uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numDependents = 0;
char ** dependents = cb.GetItemDependents("_atom_site.cartn_x",
                                          numDependents);
```

PURPOSE

GetItemDependents returns an array containing the names of the dependent items for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of dependent items

RETURN VALUE

Returns the dependent item names as a char ** or NULL. The number of dependent items will be passed back via the second argument.

REMARKS

None

2.2.20 GetItemRelated

NAME

GetItemRelated

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemRelated(const char * itemName,
                                   uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numRelated = 0;
char ** related = cb.GetItemRelated("_atom_site.cartn_x",
                                    numRelated);
```

PURPOSE

GetItemRelated returns an array containing the names of the related items for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of related items

RETURN VALUE

Returns the related item names as a char ** or NULL. The number of related items will be passed back via the second argument.

REMARKS

See also: *GetItemRelatedFunctionCode*

2.2.21 GetItemRelatedFunctionCode

NAME

GetItemRelatedFunctionCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemRelatedFunctionCode(const char *
                                               itemName,
                                               uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numRelated = 0;
char ** relatedFCode =
 cb.GetItemRelatedFunctionCode("_atom_site.cartn_x", numRelated);
```

PURPOSE

GetItemRelatedFunctionCode returns an array containing the function codes for the items related to the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of related items

RETURN VALUE

Returns the related item function codes as a char ** or NULL. The number of related items will be passed back via the second argument.

REMARKS

See also: *GetItemRelated*

2.2.22 GetItemSubcategories

NAME

GetItemSubcategories

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemSubcategories(const char * itemName,
                                         uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numSubcat = 0;
char ** subcategories =
 cb.GetItemSubcategories("_atom_site.cartn_x", numSubcat);
```

PURPOSE

GetItemSubcategories returns an array containing the subcategories that the item *itemName* belongs to.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of subcategories

RETURN VALUE

Returns the subcategory names as a char ** or NULL. The number of subcategories will be passed back via the second argument.

REMARKS

None

2.2.23 GetItemLinkedChildren

NAME

GetItemLinkedChildren

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemLinkedChildren(const char * itemName,
                                          uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numChild = 0;
char ** children =
 cb.GetItemLinkedChildren("_atom_site.cartn_x", numChild);
```

PURPOSE

GetItemLinkedChildren returns an array containing the child item names of the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of child items

RETURN VALUE

Returns the child item names as a char ** or NULL. The number of children will be passed back via the second argument.

REMARKS

See also: *GetItemLinkedParents*

2.2.24 GetItemLinkedParents

NAME

GetItemLinkedParents

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemLinkedParents(const char * itemName,
                                         uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numParent = 0;
char ** parents = cb.GetItemLinkedParents("_atom_site.cartn_x",
                                          numParent);
```

PURPOSE

GetItemLinkedParents returns an array containing the parent item names of the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of parent items

RETURN VALUE

Returns the parent item names as a char ** or NULL. The number of parents will be passed back via the second argument.

REMARKS

See also: *GetItemLinkedChildren*

2.2.25 GetItemMethods

NAME

GetItemMethods

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemMethods(const char * itemName,
                                   uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methods = cb.GetItemMethods("_atom_site.cartn_x",
                                    numMethods);
```

PURPOSE

GetItemMethods returns an array containing the method IDs for the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the methods IDs as a char ** or NULL. The number of methods will be passed back via the second argument.

REMARKS

None

2.2.26 GetItemTypeConditions

NAME

GetItemTypeConditions

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetItemTypeConditions(const char * itemName,
                                          uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numConditions = 0;
char ** conditions =
 cb.GetItemTypeConditions("_atom_site.cartn_x", numConditions);
```

PURPOSE

GetItemTypeConditions returns an array containing the item type conditions of the item *itemName*.

RECEIVES

`itemName`	the name of the item
`n`	a reference to a uWord to hold the number of type conditions

RETURN VALUE

Returns the type conditions as a char ** or NULL. The number of type conditions will be passed back via the second argument.

REMARKS

None

2.3 Category Access Methods

These methods provide context-dependent access into category objects via the dictionary object that defines the category.

2.3.1 GetCategoryDescription

NAME

GetCategoryDescription

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetCategoryDescription(const char * categoryId);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * description = cb.GetCategoryDescription("atom_site");
```

PURPOSE

GetCategoryDescription returns the description of a category in the current context.

RECEIVES

`categoryId`	the name of the category

RETURN VALUE

Returns the description of the category as a char *.

REMARKS

None

2.3.2 GetCategoryMandatoryCode

NAME

GetCategoryMandatoryCode

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetCategoryMandatoryCode(const char *
                                            categoryId);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * mandCode = cb.GetCategoryMandatoryCode("atom_site");
```

PURPOSE

GetCategoryDescription returns the mandatory code of a category in the current context.

RECEIVES

`categoryId`	the name of the category

RETURN VALUE

Returns the mandatory code of the category as a char *.

REMARKS

None

2.3.3 GetCategoryKeys

NAME

GetCategoryKeys

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetCategoryKeys(const char * categoryId,
                                    uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numKeys = 0;
char ** keys = cb.GetCategoryKeys("atom_site", numKeys);
```

PURPOSE

GetCategoryKeys returns an array containing the item names that are the keys to the category *categoryId*.

RECEIVES

`categoryId`	the name of the category
`n`	a reference to a uWord to hold the number of keys

RETURN VALUE

Returns the keys as a char ** or NULL. The number of keys will be passed back via the second argument.

REMARKS

None

2.3.4 GetCategoryExamplesCase

NAME

GetCategoryExamplesCase

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetCategoryExamplesCase(const char * categoryId,
                                            uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examples = cb.GetCategoryExamplesCase("atom_site",
                                              numExamples);
```

PURPOSE

GetCategoryExamplesCase returns an array of examples for the category *categoryId*.

RECEIVES

`categoryId`	the name of the category
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the examples as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetCategoryExamplesDetail*

2.3.5 GetCategoryExamplesDetail

NAME

GetCategoryExamplesDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetCategoryExamplesDetail(const char *
                                              categoryId,
                                              uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examplesDetail = cb.GetCategoryExamplesDetail("atom_site",
                                                      numExamples);
```

PURPOSE

GetCategoryExamplesDetail returns an array of example details for the category *categoryId*.

RECEIVES

`categoryId`	the name of the category
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the example details as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetCategoryExamplesCase*

2.3.6 GetCategoryCategoryGroups

NAME

GetCategoryCategoryGroups

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetCategoryCategoryGroups(const char *
                                              categoryId,
                                              uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numGroups = 0;
char ** categoryGroups = cb.GetCategoryCategoryGroups("atom_site",
                                                      numGroups);
```

PURPOSE

GetCategoryCategoryGroups returns an array of category group names to which the category *categoryId* belongs.

RECEIVES

`categoryId`	the name of the category
`n`	a reference to a uWord to hold the number of category groups

RETURN VALUE

Returns the category group names as a char ** or NULL. The number of category groups will be passed back via the second argument.

REMARKS

None

2.3.7 GetCategoryMethods

NAME

GetCategoryMethods

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetCategoryMethods(const char * categoryId,
                                       uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methods = cb.GetCategoryMethods("atom_site", numMethods);
```

PURPOSE

GetCategoryMethods returns an array of category method IDs that are associated with the category *categoryId*.

RECEIVES

`categoryId`	the name of the category
`n`	a reference to a uWord to hold the number of category methods

RETURN VALUE

Returns the category method IDs as a char ** or NULL. The number of methods will be passed back via the second argument.

REMARKS

None

2.4 Subcategory Access Methods

These methods provide context-dependent access into subcategory objects via the dictionary object that contains it.

2.4.1 GetSubcategoryDescription

NAME

GetSubcategoryDescription

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetSubcategoryDescription(const char *
                                                subcategoryId);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * description = cb.GetSubcategoryDescription("matrix");
```

PURPOSE

GetSubcategoryDescription returns the description of a subcategory in the current context.

RECEIVES

`subcategoryId`	the name of the subcategory

RETURN VALUE

Returns the description of the subcategory as a char * or NULL.

REMARKS

None

2.4.2 GetSubcategoryExamplesCase

NAME

GetSubcategoryExamplesCase

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetSubcategoryExamplesCase(const char *
                                               subcategoryId,
                                               uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examples = cb.GetSubcategoryExamplesCase("matrix",
                                                 numExamples);
```

PURPOSE

GetSubcategoryExamplesCase returns an array of examples for the subcategory *subcategoryId*.

RECEIVES

`subcategoryId`	the name of the subcategory
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the examples as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetSubcategoryExamplesDetail*

2.4.3 GetSubcategoryExamplesDetail

NAME

GetSubcategoryExamplesDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetSubcategoryExamplesDetail(const char *
                                                 subcategoryId,
                                                 uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numExamples = 0;
char ** examplesDetail =
 cb.GetSubcategoryExamplesDetail("matrix", numExamples);
```

PURPOSE

GetSubcategoryExamplesDetail returns an array of example descriptions for the subcategory *subcategoryId*.

RECEIVES

`subcategoryId`	the name of the subcategory
`n`	a reference to a uWord to hold the number of examples

RETURN VALUE

Returns the example details as a char ** or NULL. The number of examples will be passed back via the second argument.

REMARKS

See also: *GetSubcategoryExamplesCase*

2.4.4 GetSubcategoryMethods

NAME

GetSubcategoryMethods

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetSubcategoryMethods(const char *
                                          subcategoryId,
                                          uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methods = cb.GetSubcategoryMethods("matrix", numMethods);
```

PURPOSE

GetSubcategoryMethods returns an array of subcategory method IDs that are associated with the subcategory *subcategoryId*.

RECEIVES

`subcategoryId`	the name of the subcategory
`n`	a reference to a uWord to hold the number of subcategory methods

RETURN VALUE

Returns the subcategory method IDs as a char ** or NULL. The number of methods will be passed back via the second argument.

REMARKS

None

2.5 Dictionary Access Methods

These methods provide access into the current dictionary object (context), including access into various static tables.

2.5.1 GetDictionaryTitle

NAME

GetDictionaryTitle

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetDictionaryTitle();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * title = cb.GetDictionaryTitle();
```

PURPOSE

GetDictionaryTitle returns the title of the current dictionary datablock (context).

RECEIVES

`No Arguments`

RETURN VALUE

Returns the title of the dictionary datablock as a char * or NULL.

REMARKS

None

2.5.2 GetDictionaryVersion

NAME

GetDictionaryVersion

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetDictionaryVersion();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * version = cb.GetDictionaryVersion();
```

PURPOSE

GetDictionaryVersion returns the version of the current dictionary datablock (context).

RECEIVES

`No Arguments`

RETURN VALUE

Returns the version of the dictionary datablock as a char * or NULL.

REMARKS

None

2.5.3 GetDictionaryDescription

NAME

GetDictionaryDescription

PROTOTYPE

```
#include "CifBuilder.h"

char * CifBuilder::GetDictionaryDescription();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

char * description = cb.GetDictionaryDescription();
```

PURPOSE

GetDictionaryDescription returns the description of the current dictionary datablock (context).

RECEIVES

`No Arguments`

RETURN VALUE

Returns the description of the dictionary datablock as a char * or NULL.

REMARKS

None

2.5.4 GetDictionaryHistoryVersion

NAME

GetDictionaryHistoryVersion

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryHistoryVersion(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numV = 0;
char ** versions = cb.GetDictionaryHistoryVersion(numV);
```

PURPOSE

GetDictionaryHistoryVersion returns an array of versions that are associated with the history of the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of versions

RETURN VALUE

Returns the versions as a char ** or NULL. The number of versions will be passed back via the argument.

REMARKS

See also: *GetDictionaryHistoryUpdate*
GetDictionaryHistoryRevision

2.5.5 GetDictionaryHistoryUpdate

NAME

GetDictionaryHistoryUpdate

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryHistoryUpdate(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numU = 0;
char ** updates = cb.GetDictionaryHistoryUpdate(numU);
```

PURPOSE

GetDictionaryHistoryUpdate returns an array of dates on which the history revisions of the current dictionary datablock (context) were made.

RECEIVES

`n`	a reference to a uWord to hold the number of dates

RETURN VALUE

Returns the dates as a char ** or NULL. The number of dates will be passed back via the argument.

REMARKS

See also: *GetDictionaryHistoryVersion*
GetDictionaryHistoryRevision

2.5.6 GetDictionaryHistoryRevision

NAME

GetDictionaryHistoryRevision

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryHistoryRevision(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numRev = 0;
char ** revisions = cb.GetDictionaryHistoryRevision(numRev);
```

PURPOSE

GetDictionaryHistoryRevision returns an array of history revisions of the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of revisions

RETURN VALUE

Returns the revisions as a char ** or NULL. The number of revisions will be passed back via the argument.

REMARKS

See also: *GetDictionaryHistoryUpdate*
GetDictionaryHistoryVersion

2.5.7 GetDictionarySubcategories

NAME

GetDictionarySubcategories

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionarySubcategories(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numSubcat = 0;
char ** subcategories = cb.GetDictionarySubcategories(numSubcat);
```

PURPOSE

GetDictionarySubcategories returns an array of the subcategory names in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of subcategories

RETURN VALUE

Returns the subcategories as a char ** or NULL. The number of subcategories will be passed back via the argument.

REMARKS

None

2.5.8 GetDictionaryCategories

NAME

GetDictionaryCategories

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryCategories(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numCat = 0;
char ** categories = cb.GetDictionaryCategories(numCat);
```

PURPOSE

GetDictionaryCategories returns an array of the category names in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of categories

RETURN VALUE

Returns the categories as a char ** or NULL. The number of categories will be passed back via the argument.

REMARKS

None

2.5.9 GetDictionaryItems

NAME

GetDictionaryItems

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItems(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numItems = 0;
char ** items = cb.GetDictionaryItems(numItems);
```

PURPOSE

GetDictionaryItems returns an array of the item names in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of items

RETURN VALUE

Returns the items as a char ** or NULL. The number of items will be passed back via the argument.

REMARKS

None

2.5.10 GetDictionaryMethods

NAME

GetDictionaryMethods

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethods(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methods = cb.GetDictionaryMethods(numMethods);
```

PURPOSE

GetDictionaryMethodsId returns an array of the "methods" for the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the methods as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

None

2.5.11 GetDictionaryMethodsId

NAME

GetDictionaryMethodsId

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsId(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methods = cb.GetDictionaryMethodsId(numMethods);
```

PURPOSE

GetDictionaryMethodsId returns an array of the method IDs for the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the method IDs as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

None

2.5.12 GetDictionaryMethodsList

NAME

GetDictionaryMethodsList

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsList(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methodsList = cb.GetDictionaryMethodsList(numMethods);
```

PURPOSE

GetDictionaryMethodsList returns an array of the method IDs in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the method IDs as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

See also: *GetDictionaryMethodsListDetail*
GetDictionaryMethodsListInline
GetDictionaryMethodsListCode
GetDictionaryMethodsListLanguage

2.5.13 GetDictionaryMethodsListDetail

NAME

GetDictionaryMethodsListDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsListDetail(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryListDetail();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methodsListDetail =
 cb.GetDictionaryMethodsListDetail(numMethods);
```

PURPOSE

GetDictionaryMethodsListDetail returns an array of char * that describes the methods in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the method details as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

See also: *GetDictionaryMethodsList*
GetDictionaryMethodsListInline
GetDictionaryMethodsListCode
GetDictionaryMethodsListLanguage

2.5.14 GetDictionaryMethodsListInline

NAME

GetDictionaryMethodsListInline

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsListInline(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryListInline();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methodsListInline =
 cb.GetDictionaryMethodsListInline(numMethods);
```

PURPOSE

GetDictionaryMethodsListInline returns an array of char * that holds the inline code of the methods in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the inline code as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

See also: *GetDictionaryMethodsList*
GetDictionaryMethodsListDetail
GetDictionaryMethodsListCode
GetDictionaryMethodsListLanguage

2.5.15 GetDictionaryMethodsListCode

NAME

GetDictionaryMethodsListCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsListCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryListCode();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methodsListCode =
 cb.GetDictionaryMethodsListCode(numMethods);
```

PURPOSE

GetDictionaryMethodsListCode returns an array of char * that holds the descriptions of the functions of the methods in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the codes as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

See also: *GetDictionaryMethodsList*
GetDictionaryMethodsListDetail
GetDictionaryMethodsListInline
GetDictionaryMethodsListLanguage

2.5.16 GetDictionaryMethodsListLanguage

NAME

GetDictionaryMethodsListLanguage

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryMethodsListLanguage(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numMethods = 0;
char ** methodsListLanguage =
 cb.GetDictionaryMethodsListLanguage(numMethods);
```

PURPOSE

GetDictionaryMethodsListLanguage returns an array of char * that holds the languages with which the methods in the current dictionary datablock (context) were written.

RECEIVES

`n`	a reference to a uWord to hold the number of methods

RETURN VALUE

Returns the languages as a char ** or NULL. The number of methods will be passed back via the argument.

REMARKS

See also: *GetDictionaryMethodsList*
GetDictionaryMethodsListDetail
GetDictionaryMethodsListInline
GetDictionaryMethodsListCode

2.5.17 GetDictionaryCategoryGroupList

NAME

GetDictionaryCategoryGroupList

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryCategoryGroupList(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numGroups = 0;
char ** CategoryGroupList =
 cb.GetDictionaryCategoryGroupList(numGroups);
```

PURPOSE

GetDictionaryCategoryGroupList returns an array of char *, which is a list of the category groups in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of category groups

RETURN VALUE

Returns the list of category groups as a char ** or NULL. The number of groups will be passed back via the argument.

REMARKS

See also: *GetDictionaryCategoryGroupListParents*
GetDictionaryCategoryGroupListDescription

2.5.18 GetDictionaryCategoryGroupListParents

NAME

GetDictionaryCategoryGroupListParents

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryCategoryGroupListParents(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numGroups = 0;
char ** CategoryGroupListParents =
 cb.GetDictionaryCategoryGroupListParents(numGroups);
```

PURPOSE

GetDictionaryCategoryGroupListParents returns an array of char *, which is a list of the (optional) parents of the category groups in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of category groups

RETURN VALUE

Returns the list of category group parent names as a char ** or NULL. The number of groups will be passed back via the argument.

REMARKS

See also: *GetDictionaryCategoryGroupList*
GetDictionaryCategoryGroupListDescription

2.5.19 GetDictionaryCategoryGroupListDescription

NAME

GetDictionaryCategoryGroupListDescription

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryCategoryGroupListDescription(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numGroups = 0;
char ** CategoryGroupListDescription =
 cb.GetDictionaryCategoryGroupListDescription(numGroups);
```

PURPOSE

GetDictionaryCategoryGroupListDescription returns an array of char *, which is a list of the descriptions of the category groups in the current dictionary datablock (context).

RECEIVES

n	a reference to a uWord to hold the number of category groups

RETURN VALUE

Returns the list of category group descriptions as a char ** or NULL. The number of groups will be passed back via the argument.

REMARKS

See also: *GetDictionaryCategoryGroupList*
GetDictionaryCategoryGroupListParents

2.5.20 GetDictionaryItemStructureListCode

NAME

GetDictionaryItemStructureListCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemStructureListCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numS = 0;
char ** itemStructListCode =
 cb.GetDictionaryItemStructureListCode(numS);
```

PURPOSE

GetDictionaryItemStructureListCode returns an array of names of item structures in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item structures

RETURN VALUE

Returns the list of item structure names as a char ** or NULL. The number of item structures will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemStructureListIndex*
GetDictionaryItemStructureListDimension

2.5.21 GetDictionaryItemStructureListIndex

NAME

GetDictionaryItemStructureListIndex

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemStructureListIndex(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numI = 0;
char ** itemStructListIndex =
 cb.GetDictionaryItemStructureListIndex(numI);
```

PURPOSE

GetDictionaryItemStructureListIndex returns an array of indices of the item structures in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item structures

RETURN VALUE

Returns the list of indices as a char ** or NULL. The number of item structures will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemStructureList*
GetDictionaryItemStructureListDimension

2.5.22 GetDictionaryItemStructureListDimension

NAME

GetDictionaryItemStructureListDimension

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemStructureListDimension(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numD = 0;
char ** itemStructListDim =
 cb.GetDictionaryItemStructureListDimension(numD);
```

PURPOSE

GetDictionaryItemStructureListDimension returns the lengths of the item structure rows/columns in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item structures

RETURN VALUE

Returns the list of lengths as a char ** or NULL. The number of item structures will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemStructureList*
GetDictionaryItemStructureListIndex

2.5.23 GetDictionaryItemTypeListCode

NAME

GetDictionaryItemTypeListCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemTypeListCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numT = 0;
char ** itemTypeListCode =
 cb.GetDictionaryItemTypeListCode(numT);
```

PURPOSE

GetDictionaryItemTypeListCode returns a list of the data types in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item data types

RETURN VALUE

Returns the list of data types as a char ** or NULL. The number of item data types will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemTypeListPrimitiveCode*
GetDictionaryItemTypeListConstruct
GetDictionaryItemTypeListDetail

2.5.24 GetDictionaryItemTypeListPrimitiveCode

NAME

GetDictionaryItemTypeListPrimitiveCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemTypeListPrimitiveCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numPC = 0;
char ** itemTypeListPrimitiveCode =
 cb.GetDictionaryItemTypeListPrimitiveCode(numPC);
```

PURPOSE

GetDictionaryItemTypeListPrimitiveCode returns a list of the primitive codes of the item data types in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item data types

RETURN VALUE

Returns the list of primitive codes as a char ** or NULL. The number of item data types will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemTypeListCode*
GetDictionaryItemTypeListConstruct
GetDictionaryItemTypeListDetail

2.5.25 GetDictionaryItemTypeListConstruct

NAME

GetDictionaryItemTypeListConstruct

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemTypeListConstruct(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numC = 0;
char ** itemTypeListConstruct =
 cb.GetDictionaryItemTypeListConstruct(numC);
```

PURPOSE

GetDictionaryItemTypeListConstruct returns a list of the constructs of the item data types in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item data types

RETURN VALUE

Returns the list of constructs as a char ** or NULL. The number of item data types will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemTypeListCode*
GetDictionaryItemTypeListPrimitiveCode
GetDictionaryItemTypeListDetail

2.5.26 GetDictionaryItemTypeListDetail

NAME

GetDictionaryItemTypeListDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemTypeListDetail(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numD = 0;
char ** itemTypeListDetail =
 cb.GetDictionaryItemTypeListDetail(numD);
```

PURPOSE

GetDictionaryItemTypeListDetail returns a list of the descriptions of the item data types in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item data types

RETURN VALUE

Returns the list of descriptions as a char ** or NULL. The number of item data types will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemTypeListCode*
GetDictionaryItemTypeListPrimitiveCode
GetDictionaryItemTypeListConstruct

2.5.27 GetDictionaryItemUnitsListCode

NAME

GetDictionaryItemUnitsListCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsListCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnits = 0;
char ** itemUnitsListCode =
 cb.GetDictionaryItemUnitsListCode(numUnits);
```

PURPOSE

GetDictionaryItemUnitsListCode returns a list of units in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item units

RETURN VALUE

Returns the list of units as a char ** or NULL. The number of item units will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsListDetail*

2.5.28 GetDictionaryItemUnitsListDetail

NAME

GetDictionaryItemUnitsListDetail

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsListDetail(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnits = 0;
char ** itemUnitsListDetail =
 cb.GetDictionaryItemUnitsListDetail(numUnits);
```

PURPOSE

GetDictionaryItemUnitsListDetail returns a list of descriptions of the units in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item units

RETURN VALUE

Returns the list of descriptions as a char ** or NULL. The number of item units will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsListCode*

2.5.29 GetDictionaryItemUnitsConversionOperator

NAME

GetDictionaryItemUnitsConversionOperator

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsConversionOperator(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnitsConv = 0;
char ** itemUnitsConversionOperators =
 cb.GetDictionaryItemUnitsConversionOperator(numUnitsConv);
```

PURPOSE

GetDictionaryItemUnitsConversionOperator returns a list of arithmetic operators for converting between the various units in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item unit conversions

RETURN VALUE

Returns the list of operators as a char ** or NULL. The number of item unit conversions will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsConversionFactor*
GetDictionaryItemUnitsConversionToCode
GetDictionaryItemUnitsConversionFromCode

2.5.30 GetDictionaryItemUnitsConversionFactor

NAME

GetDictionaryItemUnitsConversionFactor

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsConversionFactor(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnitsConv = 0;
char ** itemUnitsConversionFactors =
 cb.GetDictionaryItemUnitsConversionFactor(numUnitsConv);
```

PURPOSE

GetDictionaryItemUnitsConversionFactor returns a list of the arithmetic factors for converting between the various units in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item unit conversions

RETURN VALUE

Returns the list of factors as a char ** or NULL. The number of item unit conversions will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsConversionOperator*
GetDictionaryItemUnitsConversionToCode
GetDictionaryItemUnitsConversionFromCode

2.5.31 GetDictionaryItemUnitsConversionToCode

NAME

GetDictionaryItemUnitsConversionToCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsConversionToCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnitsConv = 0;
char ** itemUnitsConversionToCode =
 cb.GetDictionaryItemUnitsConversionToCode(numUnitsConv);
```

PURPOSE

GetDictionaryItemUnitsConversionToCode returns a list of the produced units after converting between the various units in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item unit conversions

RETURN VALUE

Returns the list of results as a char ** or NULL. The number of item unit conversions will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsConversionOperator*
GetDictionaryItemUnitsConversionFactor
GetDictionaryItemUnitsConversionFromCode

2.5.32 GetDictionaryItemUnitsConversionFromCode

NAME

GetDictionaryItemUnitsConversionFromCode

PROTOTYPE

```
#include "CifBuilder.h"

char ** CifBuilder::GetDictionaryItemUnitsConversionFromCode(uWord & n);
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();
cb.SetContext("cifdic.m96");

uWord numUnitsConv = 0;
char ** itemUnitsConversionFromCode =
 cb.GetDictionaryItemUnitsConversionFromCode(numUnitsConv);
```

PURPOSE

GetDictionaryItemUnitsConversionFromCode returns a list of the unit systems to be converted in the current dictionary datablock (context).

RECEIVES

`n`	a reference to a uWord to hold the number of item unit conversions

RETURN VALUE

Returns the list of target units as a char ** or NULL. The number of item unit conversions will be passed back via the argument.

REMARKS

See also: *GetDictionaryItemUnitsConversionOperator*
GetDictionaryItemUnitsConversionFactor
GetDictionaryItemUnitsConversionToCode

2.6 Debugging Methods

These methods provide debugging functionality for the CifBuilder class.

2.6.1 GetAllDictionaries

NAME

GetAllDictionaries

PROTOTYPE

```
#include "CifBuilder.h"

void CifBuilder::GetAllDictionaries();
```

EXAMPLE

```
#include "CifBuilder.h"

CifBuilder cb("mmCif96.psf");

cb.BuildDictionaryList();

cb.GetAllDictionaries();
```

PURPOSE

GetAllDictionaries gets all of the dictionaries in the persistent storage file, all of their items, categories, and subcategories, and prints all of this out.

RECEIVES

```
No Arguments
```

RETURN VALUE

None

REMARKS

For debugging