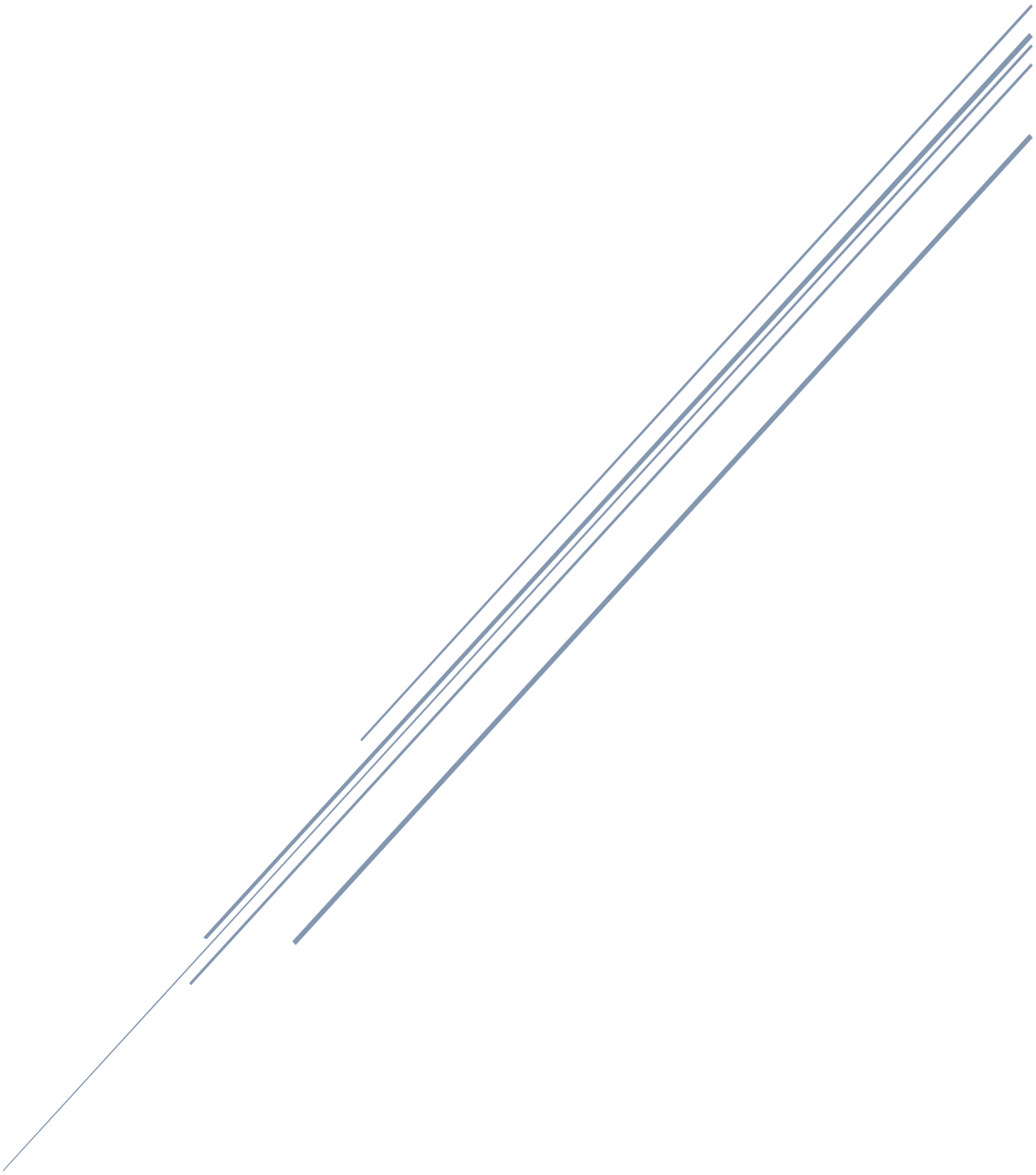
DEEPEN

atom-to-Device Explicit simulation Environment for Photonics and Electronics Nanostructures

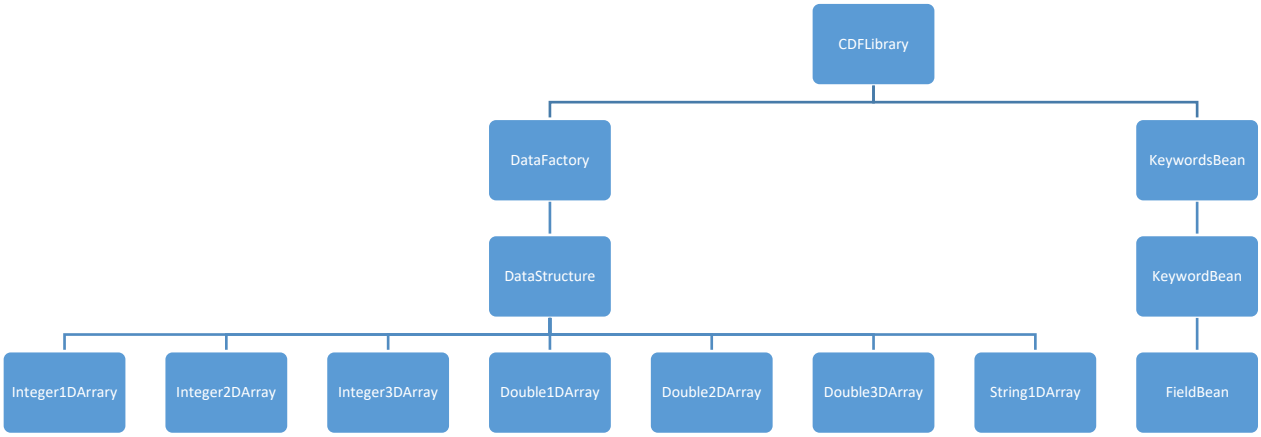


CDF API USE CASES

TABLE OF CONTENTS

CDF API USE CASES 2
HOW TO USE THE API 2
CREATE A NEW CDF FILE 3
OPEN AN EXISTING CDF FILE 3
CREATE A NEW GROUP 3
CREATE A NEW DATASET 3
GET DATA STRUCTURE 3
READ A DATASET 3
GET KEYWORDS LIST 3
SAVE KEYWORDS LIST 4
BIBLIOGRAFY 4

CDF API USE CASES



CDFLibrary is the main class and give you the all the read/write methods for CDF and Keywords system.

DataFactory is a singleton class that return the right type of *Data Structure* based on provided parameters. It used by *CDFLibrary*, but also by the developers when they need to fill a data structure to write in a CDF file.

DataStructure is an abstract class and the superclass of all data concrete class.

Integer1DArray is an implementation of *DataStructure* class. It handle integer data values arranged in a one dimension array.

Integer2DArray is an implementation of *DataStructure* class. It handle integer data values arranged in a 2 dimensions array.

The same description can be adapted to the other concrete implementations of the DataStructure class.

KeywordsBean class exposes the Map between the keyword names and their details data.

KeywordBean and **FieldBean** are the beans to store the detail fields of a keyword.

HOW TO USE THE API

The CDF API is useful in the development of translators when you need to read/write CDF file type, while the Keywords system is useful in the development of the GridBean other than for translators.

With the source code of the API, it's available the relative javadoc documentation which should be consulted for the right use of the API.

Below you can find some usage examples:

CREATE A NEW CDF FILE

```
int file_id = cdfLibrary.createFile("/home/user/test/myFile.h5");
```

OPEN AN EXISTING CDF FILE

```
int file_id = cdfLibrary.openFile("/home/user/test/myFile.h5");
```

CREATE A NEW GROUP

```
int group_id = cdfLibrary.createGroup(file_id, "/myTool/myNewGroup");
```

CREATE A NEW DATASET

```
cdfLibrary.createDataset(file_id, "myDatasetName", "myGroup", myData,
"Double", "Description of dataset content");
```

This method will create a new dataset called `myDatasetName` inside the group `/myTool/myNewGroup` in the CDF file `file_id`. The content of dataset will be `myData` that can be a multidimensional array of Integer, Double or String. The dataset metadata will be fill with the info for "Unit" (`Double`) and "Description" (`Description of dataset content`). "Unit" have to give the info on type of data while "Description" can be the description of the dataset and can give also info on how to use that data or how the data is organized.

GET DATA STRUCTURE

```
long[] dims = {20, 30};
DataStructure<?> h_data = DataFactory.getInstance().
                          getDataStructure(DataFactory.DOUBLE, 2, dims);
```

This method will return a DataStructure object with modeled data in a 2-dimensional array of double. The first column will have dimension 20, the secondo one 30.

The method is useful when you need to store data from an external file in a cdf file.

READ A DATASET

```
DataStructure<?> dataStructure = cdfLibrary.
                 readStringDataset(file_id, "/myTool/myGroup/myDataset");
```

This method will return a DataStructure object filled with the content of `myDataset` dataset. The type of values will be String. The API will set the right dimensions for data arrays.

GET KEYWORDS LIST

```
KeywordsBean keywords = cdfLibrary.getKeywords();
```

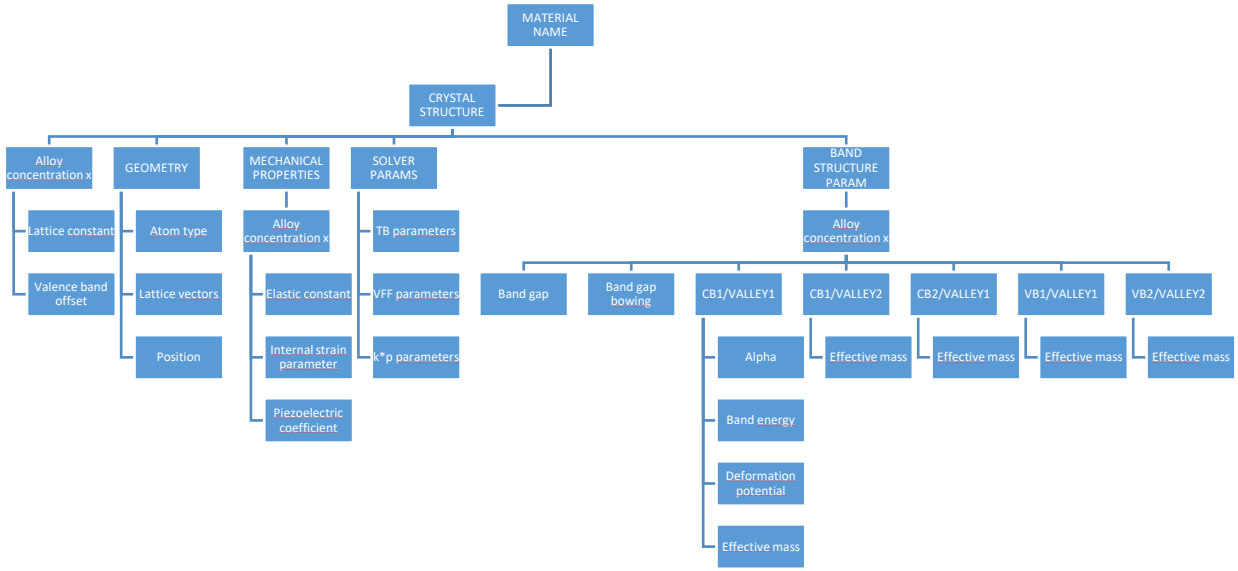
This method will return the list of defined keywords after parsing the `keywords.xml` file that has to be present in the working directory.

SAVE KEYWORDS LIST

```
Boolean result = cdfLibrary.saveKeywords(KeywordsBean keywords);
```

This method will save in xml format the map keyword->keyword_properties. The file will be named `keywords.xml`

DATABASE STRUCTURE



In the figure above we have the DEEPEN database structure and that's the reference for groups and dataset names.

BIBLIOGRAFY

Javadoc CDF API – www.tibercad.org

DEEPEN Tool Translators source code – www.tibercad.org

DEEPEN GRIDBEANS source code – www.tibercad.org