The OpenDDL-Parser

The OpenDDL-Parser is a small and easy to use library for OpenDDL-file-format-parsing. OpenDDL is the shortcut for Open Data Description Language, a data-declaration language introduced by Eric Lengyel. Please check http://openddl.org/ if you want to learn more about it.

Build status

Linux build status: build failing Current coverity check status: coverity passed Current test coverage: coverage 92% Get the source code =================== You can get the code from our git repository, which is located at GitHub. You can clone the repository with the following command:

> *git clone https://github.com/kimkulling/openddl-parser.git*

Building the source from the GitHub-Repo

To build the library you need to install cmake first (see http://www.cmake.org/ for more information). Make also sure that a compiler tool-chain is installed on your machine. After installing it you can open a console and enter:

> *cmake CMakeLists.txt*

This command will generate a build environment for your preferred build tool (for Visual-Studio-users the project files will be generated, for gcc-users the makefiles will be generated). When using an IDE open the IDE and run the build. When using GNU-make type in your console:

> *make*

and that's all.

When using Visual Studio CMake will generate you a solution for ythe library. Just build it there.

Use the library

To use the OpenDDL-parser you need to build the lib first. Now add the

> */include*

to your include-path and the

> */lib*

to your lib-folder. Link the openddl.lib to your application.

Here is a small example how to use the lib:

```
#include <iostream>
#include <cassert>
#include <openddlparser/OpenDDLParser.h>
```

```
USE_ODDLPARSER_NS;

int main( int argc, char *argv[] ) {
    if( argc < 3 ) {
        return 1;
    }

    char *filename( nullptr );
    if( 0 == strncmp( FileOption, argv[ 1 ], strlen( FileOption ) ) ) {
        filename = argv[ 2 ];
    }
    std::cout << "file to import: " << filename << std::endl;
    if( nullptr == filename ) {
        std::cerr << "Invalid filename." << std::endl;
        return Error;
    }

    FILE *fileStream = fopen( filename, "r+" );
    if( NULL == filename ) {
        std::cerr << "Cannot open file " << filename << std::endl;
        return 1;
    }

    // obtain file size:
    fseek( fileStream, 0, SEEK_END );
    const size_t size( ftell( fileStream ) );
    rewind( fileStream );
    if( size > 0 ) {
        char *buffer = new char[ size ];
        const size_t readSize( fread( buffer, sizeof( char ), size, fileStream ) );
        assert( readSize == size );
        OpenDDLParser theParser;
        theParser.setBuffer( buffer, size );
        const bool result( theParser.parse() );
        if( !result ) {
            std::cerr << "Error while parsing file " << filename << "." << std::endl;
        }
    }
    return 0;
}
```

How to access the imported data

The data is organized as a tree. You can get the root-node of the tree with the following code:

```
OpenDDLParser theParser;
theParser.setBuffer( buffer, size );
const bool result( theParser.parse() );
if ( result ) {
    DDLNode *root = theParser.getRoot();
    DDLNode::DllNodeList childs = root->getChildNodeList();
    for ( size_t i=0; i<childs.size(); i++ ) {
        DDLNode *child = childs[ i ];
        Property *prop   = child->getProperty(); // to get properties
        std::string type = child->getType();     // to get the node type
        Value *values    = child->getValue();    // to get the data;

        // to loop through all values
        while ( values != ddl_nullptr ) {
            int current = values->getInt32();
            values = value->getNext();
        }
    }
}
```

The node instance called root contains the data.

All data lists are organized as linked lists.

Reference documentation

Please check http://kimkulling.github.io/openddl-parser/doxygen_html/index.html.

Projects using OpenDDL-Parser

- Asset Importer Lib: https://github.com/assimp/assimp .