Moving applications from 1.10 to HDF5 1.12 release

March 20, 2020

The HDF Group

Elena Pourmal

epourmal@hdfgroup.org

Goals of today's Webinar

- Educate HDF5 users on
  - Compatibility issues in HDF5 1.12.0
  - How to move applications written for the earlier versions of HDF5 to the new 1.12.* releases *without* modifying applications.

- If you have questions please use Google document* or send email to help@hdfgroup.org. The link to the document is posted with this video and slides.

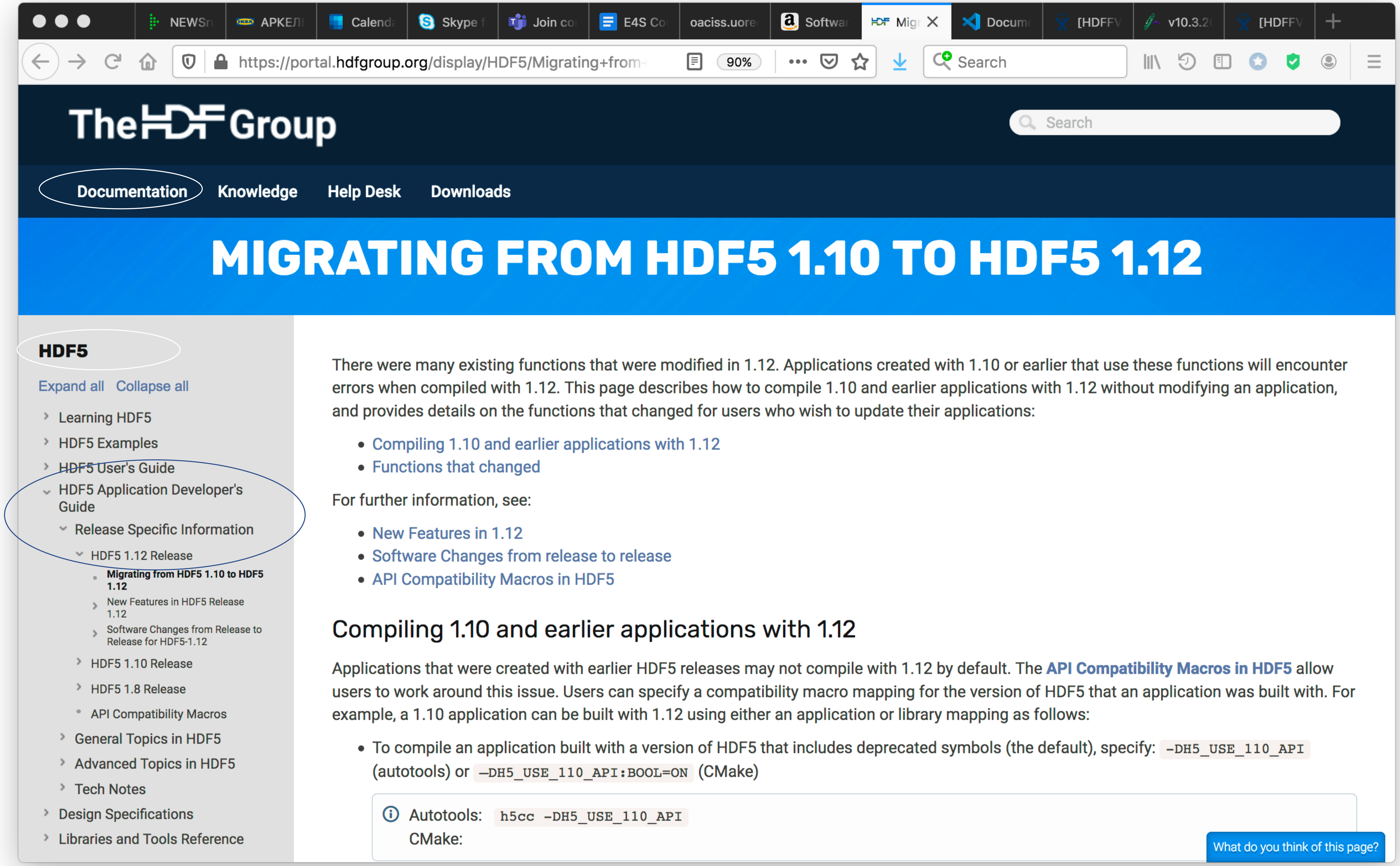
* https://bit.ly/Movingto112

HDF5 1.12.0 Release and Compatibility Issues

HDF5 1.12.0

- Released on February 29, 2020
- Major release contains
  - File format extensions and architectural and APIs changes to support
    - Virtual Object Layer (VOL) (*new library layer*)
    - External references and selections encoding (file format changes)
  - Order of magnitude performance enhancements for hyperslab selection
  - S3 and HDFS Virtual File Drivers (VFD)
- Available from The HDF Group Support Portal https://portal.hdfgroup.org/display/support/Downloads

Information on migration to 1.12.0



MIGRATING FROM HDF5 1.10 TO HDF5 1.12

HDF5

Expand all Collapse all

- Learning HDF5
- HDF5 Examples
- HDF5 User's Guide
- HDF5 Application Developer's Guide
  - Release Specific Information
    - HDF5 1.12 Release
      - **Migrating from HDF5 1.10 to HDF5 1.12**
      - New Features in HDF5 Release 1.12
      - Software Changes from Release to Release for HDF5-1.12
    - HDF5 1.10 Release
    - HDF5 1.8 Release
    - API Compatibility Macros
  - General Topics in HDF5
  - Advanced Topics in HDF5
  - Tech Notes
- Design Specifications
- Libraries and Tools Reference

There were many existing functions that were modified in 1.12. Applications created with 1.10 or earlier that use these functions will encounter errors when compiled with 1.12. This page describes how to compile 1.10 and earlier applications with 1.12 without modifying an application, and provides details on the functions that changed for users who wish to update their applications:

- Compiling 1.10 and earlier applications with 1.12
- Functions that changed

For further information, see:

- New Features in 1.12
- Software Changes from release to release
- API Compatibility Macros in HDF5

Compiling 1.10 and earlier applications with 1.12

Applications that were created with earlier HDF5 releases may not compile with 1.12 by default. The **API Compatibility Macros in HDF5** allow users to work around this issue. Users can specify a compatibility macro mapping for the version of HDF5 that an application was built with. For example, a 1.10 application can be built with 1.12 using either an application or library mapping as follows:

- To compile an application built with a version of HDF5 that includes deprecated symbols (the default), specify: `-DH5_USE_110_API` (autotools) or `–DH5_USE_110_API:BOOL=ON` (CMake)

Autotools: `h5cc -DH5_USE_110_API`
CMake:

What do you think of this page?

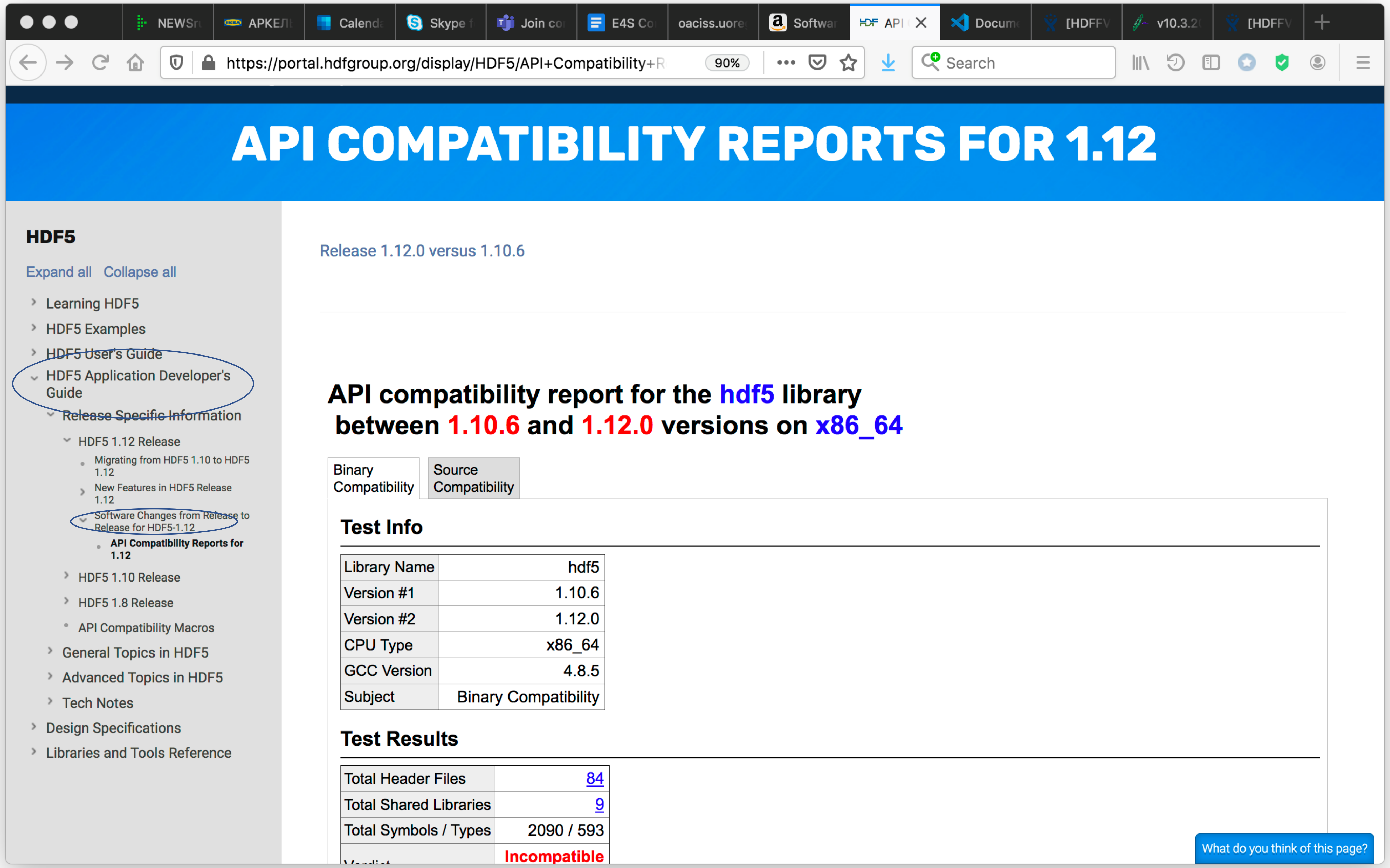
HDF5 1.12.0 file format compatibility

- *Backward compatibility:* Applications built with HDF5 1.12.0 can always access files created by the previous version of the library.
- *Forward compatibility:* If no new features of HDF5 1.12.0 are used and no latest file format is specified using the H5Pset_libver_bounds function, HDF5 files created by an HDF5 application linked with the 1.12.0 release should be accessible by applications built with 1.10.*

Report file format compatibility problems to help@hdfgroup.org

HDF5 1.12.0 API compatibility

- HDF5 1.12.0 is **not compatible** with HDF5 1.10.*
- See “API Compatibility Reports for 1.12”



https://portal.hdfgroup.org/display/HDF5/API+Compatibility+R

API COMPATIBILITY REPORTS FOR 1.12

HDF5

Expand all   Collapse all

- Learning HDF5
- HDF5 Examples
- HDF5 User's Guide
- HDF5 Application Developer's Guide
  - Release Specific Information
    - HDF5 1.12 Release
      - Migrating from HDF5 1.10 to HDF5 1.12
      - New Features in HDF5 Release 1.12
      - Software Changes from Release to Release for HDF5-1.12
        - **API Compatibility Reports for 1.12**
    - HDF5 1.10 Release
    - HDF5 1.8 Release
    - API Compatibility Macros
  - General Topics in HDF5
  - Advanced Topics in HDF5
  - Tech Notes
- Design Specifications
- Libraries and Tools Reference

Release 1.12.0 versus 1.10.6

API compatibility report for the hdf5 library between 1.10.6 and 1.12.0 versions on x86_64

Binary Compatibility | Source Compatibility

Test Info

Library Name	hdf5
Version #1	1.10.6
Version #2	1.12.0
CPU Type	x86_64
GCC Version	4.8.5
Subject	Binary Compatibility

Test Results

Total Header Files	84
Total Shared Libraries	9
Total Symbols / Types	2090 / 593
Verdict	Incompatible

HDF5 1.12.0 API compatibility (cont’d)

- HDF5 applications that use H5L and H5O interfaces to iterate over the objects in an HDF5 file and to find information about the objects will be affected.
- HDF5 applications that use haddr_t type variables (and related HDF5 functions) will be affected.
- Compilation will fail with the error message similar to

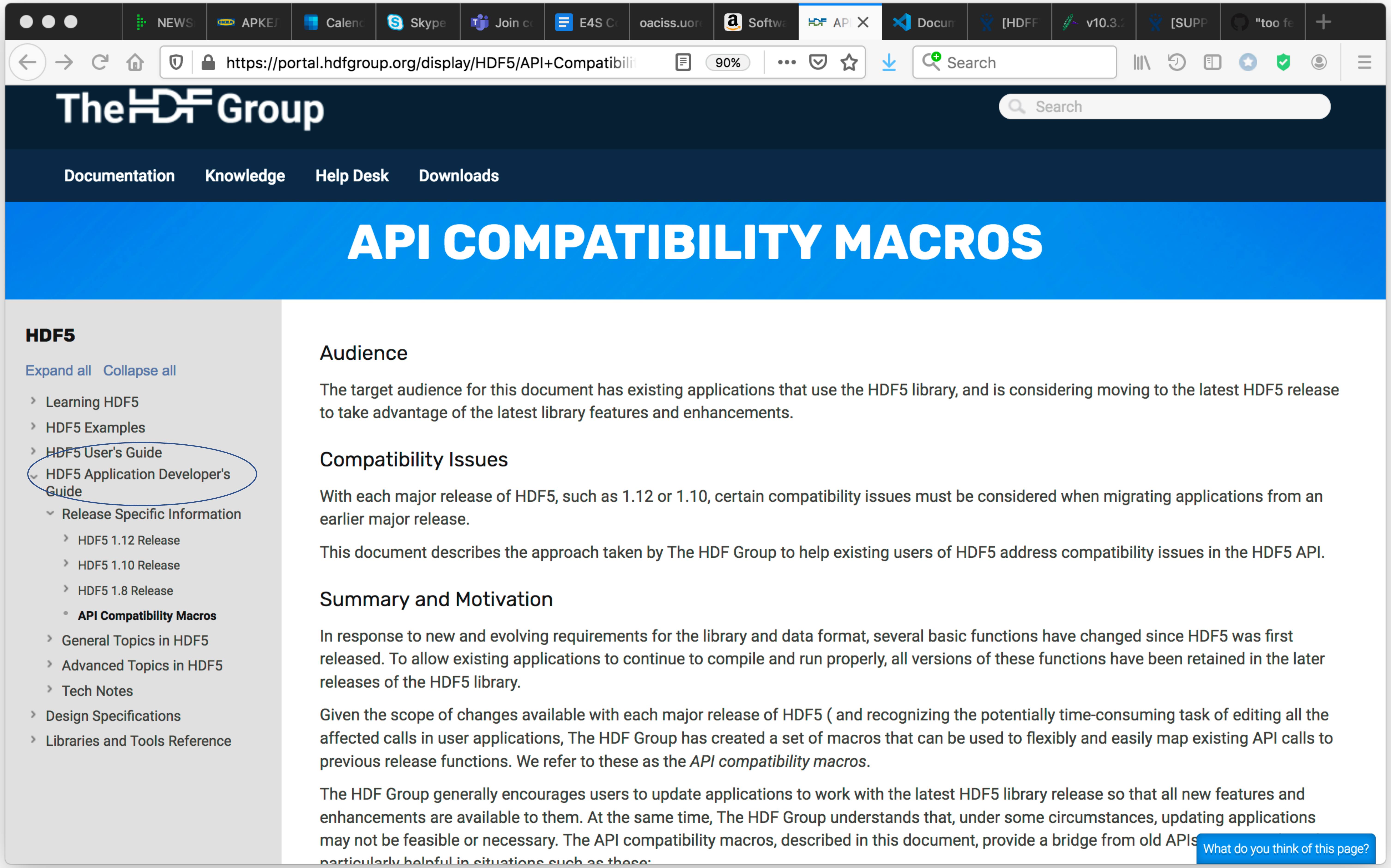
```
h5py/defs.c:16062:56: error: too few arguments to function call,
expected 3, have 2
```

Why APIs were changed?

- In releases prior to 1.12.0, HDF5 file address of an object header was used by the library to create object identifier.
- There are several HDF5 functions that use file address as a parameter to access and find information about the object.
- In general, this mechanism doesn't work when HDF5 “file” stored in non-POSIX store (e.g., Object Store).
- A token type H5O_token_t was introduced to address the issue and H5O_token_t replaced haddr_t to support VOL and VOL connectors.
- Functions were versioned and macros were added to map to the default 1.10 or 1.12 signatures.

Information about HDF5 macros

https://portal.hdfgroup.org/display/HDF5/API+Compatibility+Macros



API COMPATIBILITY MACROS

HDF5

Expand all Collapse all

- Learning HDF5
- HDF5 Examples
- HDF5 User's Guide
- HDF5 Application Developer's Guide
  - Release Specific Information
    - HDF5 1.12 Release
    - HDF5 1.10 Release
    - HDF5 1.8 Release
    - **API Compatibility Macros**
  - General Topics in HDF5
  - Advanced Topics in HDF5
  - Tech Notes
- Design Specifications
- Libraries and Tools Reference

Audience

The target audience for this document has existing applications that use the HDF5 library, and is considering moving to the latest HDF5 release to take advantage of the latest library features and enhancements.

Compatibility Issues

With each major release of HDF5, such as 1.12 or 1.10, certain compatibility issues must be considered when migrating applications from an earlier major release.

This document describes the approach taken by The HDF Group to help existing users of HDF5 address compatibility issues in the HDF5 API.

Summary and Motivation

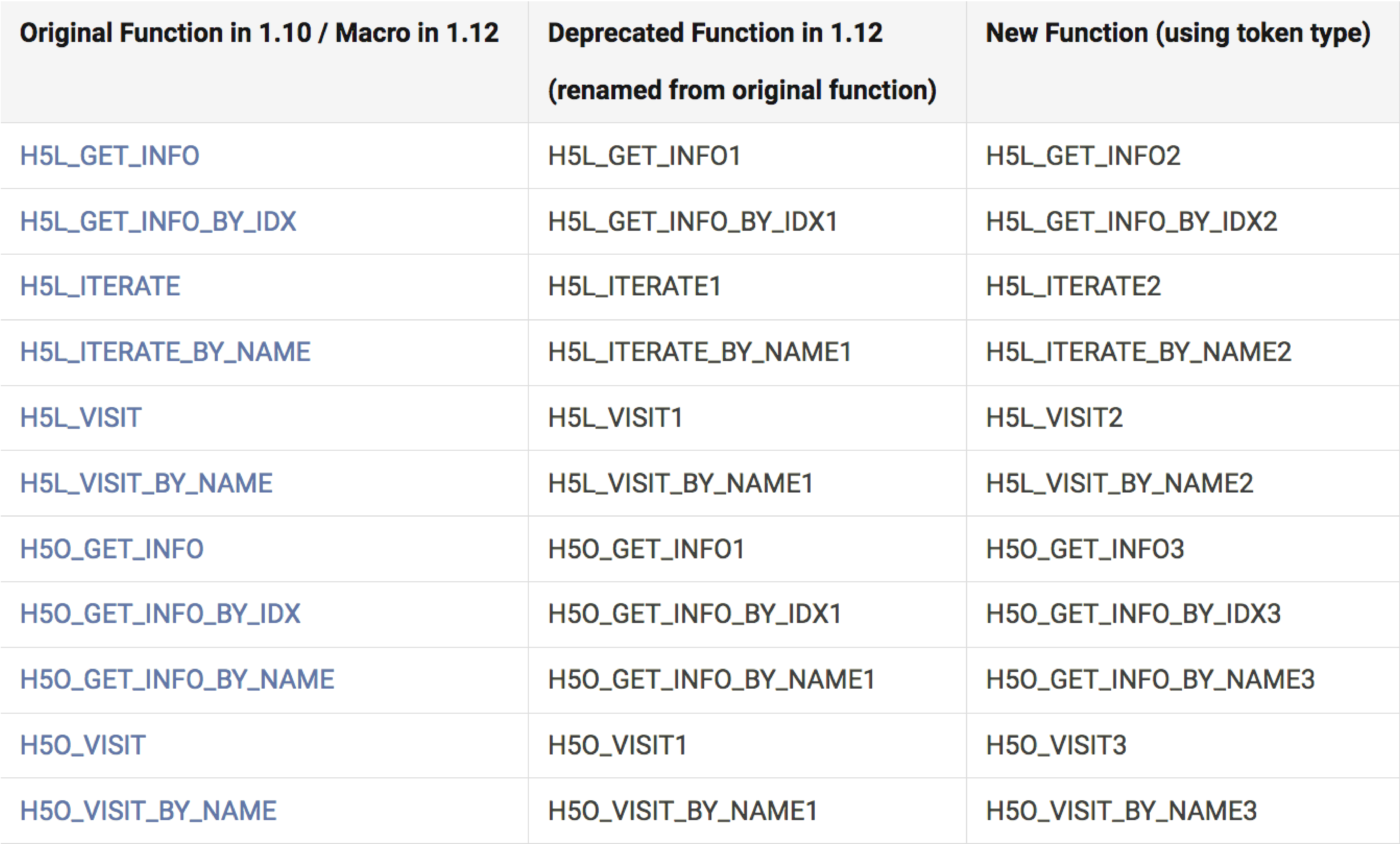
In response to new and evolving requirements for the library and data format, several basic functions have changed since HDF5 was first released. To allow existing applications to continue to compile and run properly, all versions of these functions have been retained in the later releases of the HDF5 library.

Given the scope of changes available with each major release of HDF5 (and recognizing the potentially time-consuming task of editing all the affected calls in user applications, The HDF Group has created a set of macros that can be used to flexibly and easily map existing API calls to previous release functions. We refer to these as the *API compatibility macros*.

The HDF Group generally encourages users to update applications to work with the latest HDF5 library release so that all new features and enhancements are available to them. At the same time, The HDF Group understands that, under some circumstances, updating applications may not be feasible or necessary. The API compatibility macros, described in this document, provide a bridge from old APIs

particularly helpful in situations such as these:

Functions with changed signatures



Original Function in 1.10 / Macro in 1.12	Deprecated Function in 1.12 (renamed from original function)	New Function (using token type)
H5L_GET_INFO	H5L_GET_INFO1	H5L_GET_INFO2
H5L_GET_INFO_BY_IDX	H5L_GET_INFO_BY_IDX1	H5L_GET_INFO_BY_IDX2
H5L_ITERATE	H5L_ITERATE1	H5L_ITERATE2
H5L_ITERATE_BY_NAME	H5L_ITERATE_BY_NAME1	H5L_ITERATE_BY_NAME2
H5L_VISIT	H5L_VISIT1	H5L_VISIT2
H5L_VISIT_BY_NAME	H5L_VISIT_BY_NAME1	H5L_VISIT_BY_NAME2
H5O_GET_INFO	H5O_GET_INFO1	H5O_GET_INFO3
H5O_GET_INFO_BY_IDX	H5O_GET_INFO_BY_IDX1	H5O_GET_INFO_BY_IDX3
H5O_GET_INFO_BY_NAME	H5O_GET_INFO_BY_NAME1	H5O_GET_INFO_BY_NAME3
H5O_VISIT	H5O_VISIT1	H5O_VISIT3
H5O_VISIT_BY_NAME	H5O_VISIT_BY_NAME1	H5O_VISIT_BY_NAME3

H5O_info_t and H5L_info_t structures are also versioned.

Macros and deprecated symbols

- By default
  - HDF5 library is configured with deprecated APIs
  - Macros point to 1.12 API version
    - Example: both H5Literate1 and H5Literate2 are present, H5Literate macro corresponds to H5Literate2
    - Language wrappers always use the latest version of API
- Earlier versions of APIs are deprecated
  - Marked in RM as deprecated
  - --disable-deprecated-symbols configure flag allows to build HDF5 without deprecated symbols
- Check libhdf5.settings file for default version of APIs and presence of deprecated symbols

Moving to HDF5 1.12.0 without changing applications

User’s options to use 1.12.0 *without* changing application

Build an HDF5 library with the 1.10 APIs

- When using Autotools use configure flag

```
./configure --with-default-api-version=v110 …
```

- When using CMake packaged code, edit HDF5options.cmake to add this line:

```
set(ADD_BUILD_OPTIONS "${ADD_BUILD_OPTIONS}
-DDEFAULT_API_VERSION:STRING=v110")
```

- This method is less disruptive for applications. Remember that you can always use versioned functions to use new features!
- Check libhdf5.settings file how for default version of APIs.

User's options to use 1.12.0 *without* changing application (cont'd)

If you are using installed library with 1.12 default APIs, compile application using -DH5_USE_110_API flag

```
h5cc  -DH5_USE_110_API …
cmake -DCMAKE_C_FLAGS="-DH5_USE_110_API" …
```

- When using the compiler flag macros point to the specified *version 1.10 (deprecated APIs are present by default)*
  - Example: both H5Literate1 and H5Literate2 are present, H5Literate macro points to function H5Literate1 with 1.10 signature.

User’s options to use 1.12.0 *without* changing application (cont’d)

Compile application and specify version of API and corresponding data structure (not recommended)

- Include each function and mapped structure plus EVERY function that uses the mapped structure, whether or not that function is used in the application. *In 1.12, mappings of structures are used by the H5L and H5O function mappings.*

```
h5cc -DH5Lvisit_vers=1 -DH5Ovisit_vers=1 -DH5Oget_info_by_name_vers=1
-DH5Lvisit_by_name_vers=1 -DH5Literate_vers=1 -
DH5Literate_by_name_vers=1 -DH5O_info_t_vers=1 -DH5L_info_t_vers=1 -
DH5L_iterate_t_vers=1 -DH5Lget_info_by_idx_vers=1 -DH5Lget_info_vers=1
h5ex_g_visit.c
```

Thank you!

If you have questions or suggestions please use Google document or send email to help@hdfgroup.org