Feature # 2585 (Resolved): Infrastructure supporting external API

**API status object**

08/14/2018 04:48 PM - Eric Irrgang

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>gmxapi</td>
</tr>
<tr>
<td>Category:</td>
<td>gmxapi</td>
</tr>
<tr>
<td>Target version:</td>
<td>uncategorized</td>
</tr>
<tr>
<td>Difficulty:</td>
<td>uncategorized</td>
</tr>
</tbody>
</table>

**Description**

gmxapi milestone 17 as described in [#2585](#2585)

Clients of a GROMACS installation need a robust and well-defined way to manage the results or failure of API operations.

A Status type defines the interface for discovering operation success or failure, plus details.

A consistent Status object interface is portable across Python, C++, and C

Status object can be used to ferry information across API boundaries from exceptions thrown. Exceptions could be chained / status nested.

API client Status objects may be proxies for the actual operation results, allowing deferred and/or non-local execution of operations.

**Questions:**

- Should the API (or APIContext) keep a Status singleton? A Status stack? Or should operations create ephemeral Status objects, or objects implementing a Status interface, or should API objects have a Status member?
- What are concerns and solutions for memory allocation for status objects? Should objects own one or generate one on function return?
- How lightweight does a status object need to be? Should the status object contain strings, reference strings mapped by enum, or defer textual messages to messaging and logging facilities?

My inclination is to have a Status object that can hold the success or failure of simple operations, but not to require operations to return a Status object when it is reasonable to return a richer API object. At the very least, there should be a consistent style for accessing details of failures in Status object or other API returned types.

Initial implementation could be a very thin wrapper around `bool`, indicating success or failure.

Resolution of this issue enables gmxapi milestone 16 to proceed.

**Associated revisions**

**Revision 01ec383e - 10/02/2018 10:41 AM - Eric Irrgang**

Add gmxapi::Status.

The class is very minimal, but provides a placeholder so that other API operations have a stable return value. It is used in cases where some other return value has no obvious choice or is not implemented.

Relates to gmxapi milestone 17. Refs #2610.

Change-Id: i23675b4f3bd022ea6986b2e95ca838185327a8b7

**History**

#1 - 09/06/2018 05:35 PM - Gerrit Code Review Bot

Gerrit received a related patchset `7` for Issue #2610.

Uploader: M. Eric Irrgang (ericirrgang@gmail.com)

Change-Id: gromacs~master~I23675b4f3bd022ea6986b2e95ca838185327a8b7

02/22/2020
gmxbpi::Status has been introduced in a minimal form. In general, more useful return values should be produced that do not allow invalid results to go undetected while providing something useful. In cases where there is no clear returned data, but the API operation may have its execution deferred, gmxbpi::Status should evolve to encapsulate a "future" style result, but this not addressed in this issue or in GROMACS 2019.

#3 - 01/07/2019 01:39 AM - Mark Abraham
- Category set to gmxbpi
- Status changed from Resolved to Closed