Network Endpoint Actors Review – Response

**Examples**

I have added instructions on how to run the example to both launcher VIs. I have added block diagram comments to the block diagrams of the launcher VIs and the relevant VIs of the Chat Window class. I have also added content to the documentation properties of the Chat Window class and several member VIs.

The example now installs with the main toolkit package, so we can cancel my request to update the separate example package. (The existing example package should be retained for users of older versions of the toolkit.)

**API Design**

*API VIs contain error terminals on the bottom left and bottom right of the connector pane.*

There are several VIs in this tool that do not meet this requirement, and they are, in fact, the ones that will be called directly by users. This is a design choice, not a defect.

The VIs in question fall into two broad categories: data member accessors and object placement VIs.

Data member accessors manipulate the data of their respective objects. These VIs include anything in the API whose name is of the form Read <item>.vi, Write <item>.vi, or Create <stream type>.vi. These VIs only read or write class member data. They cannot fail, and failing to read or write because of an upstream can lead to confusing results. Furthermore, passing a wire through the accessor forces serialization and undercuts the ability of the compiler to optimize code. For these reasons, the standard has emerged in the object-oriented portion of the LabVIEW community that these VIs should *not* include error wires, unless required for use as a class property node.

The remaining VIs (Caller Endpoint.vi, Nested Endpoint.vi, Legacy Nested Endpoint.vi, and Notifying Nested Endpoint.vi) exist simply to drop a class constant on a block diagram. Each VI only contains an instance of its respective class, and the palette items are configured to merge the VI contents onto the target block diagram. Error connectors are contraindicated for these VIs.

The aforementioned VIs should be exempted from this API design requirement. Please list any other VIs in the API that you feel require error clusters, and I will address them.

*All API VIs contain VI, VI input and output descriptions visible through Context Help.*

I believe I have addressed this for all relevant VIs, including many that are not strictly part of the API. I have *not* provided context help for private or protected scope messages, as those cannot be called by the user.

Please let me know if I’ve missed any.

*[Recommended] API VI descriptions contain a link to the API's detailed help.*

I don’t’ have experience preparing .chm or .hlp files. As this is recommended, and I do provide a PDF file, I’m going to skip this step. I will provide access through the Help menu, as specified.

**Detailed Help**

The PDF that was included in this package is now available through the LabVIEW Help menu. The document describes the package API. It has been updated to note that the package is not formally supported, but that users can seek guidance on the forum post dedicated to this package.