

Error_Services

Error tracking and reporting utility.

Syntax

```
Response = Error_Services(@Service, @Params)
```

Returns

The meaning of the response value depends on the service.

Parameters

Parameter	Description
@Service	The name of the service being requested. Required.
@Params	Generic parameters. Refer to a specific service to determine the actual parameters used.

Remarks

This SRP FrameWorks utility service is a general purpose error tracking and reporting utility. Unlike OpenInsight error routines, [Error_Services](#) does not interfere with normal operations even if a previous error condition was set. The developer is given the right (and responsibility) to make these decisions.

Services

Service	Description
Set	Usage: Error_Services('Set', ErrorMessage) Comments: Sets an error to the stack. This will automatically clear any existing error conditions first so this error will be the only one on the stack. Returns: N/A
Add	Usage: Error_Services('Add', ErrorMessage) Comments: Adds an error to the stack. This will not clear existing error conditions first. It is intended to allow higher level routines to add more information to an existing error condition or simply to maintain an ongoing error log for some troubleshooting or debugging purposes. Returns: N/A
Clear	Usage: Error_Services('Clear') Comments: Clears all error conditions and related information. Returns: N/A
GetMessage	Usage: Error_Services('GetMessage') Comments: Returns the most current error message. Returns: The most current error message.

GetMessages	Usage: Error_Services('GetMessages') Comments: Returns the stack of error messages. This will be @FM delimited. Returns: The stack of error messages.
HasError	Usage: Error_Services('HasError') Comments: Returns <i>True</i> if there is an error condition, <i>False</i> if there is no error condition. Caller will still need to use the GetMessage or GetMessages service to determine what the error is. The HasError service allows the caller to embed the Error_Services service call inside of a conditional statement like this: <pre>If Error_Services('HasError') then * An error has occurred. Proceed accordingly. ErrorMessage = Error_Services('GetMessage') end else * No error has occurred. end</pre> Returns: <i>True</i> if there is an error condition, <i>False</i> if there is no error condition.
NoError	Usage: Error_Services('NoError') Comments: Returns <i>True</i> if there are no error conditions, <i>False</i> if there is an error condition. This is the opposite of the HasError service and exists for improved readability. Returns: <i>True</i> if there are no error conditions, <i>False</i> if there is an error condition.
DisplayError	Usage: Error_Services('DisplayError') Comments: Displays the current error message to the end user. Error_Services is designed to avoid any user interface so it can be safe to use in application contexts where no presentation server context is available (e.g., web applications). For convenience, the DisplayError service was added to make it easy to display the most recent error added to the stock. Returns: N/A

Params

The proper use of the generic arguments are defined in the definition of each service above.