SAND2019-5442C

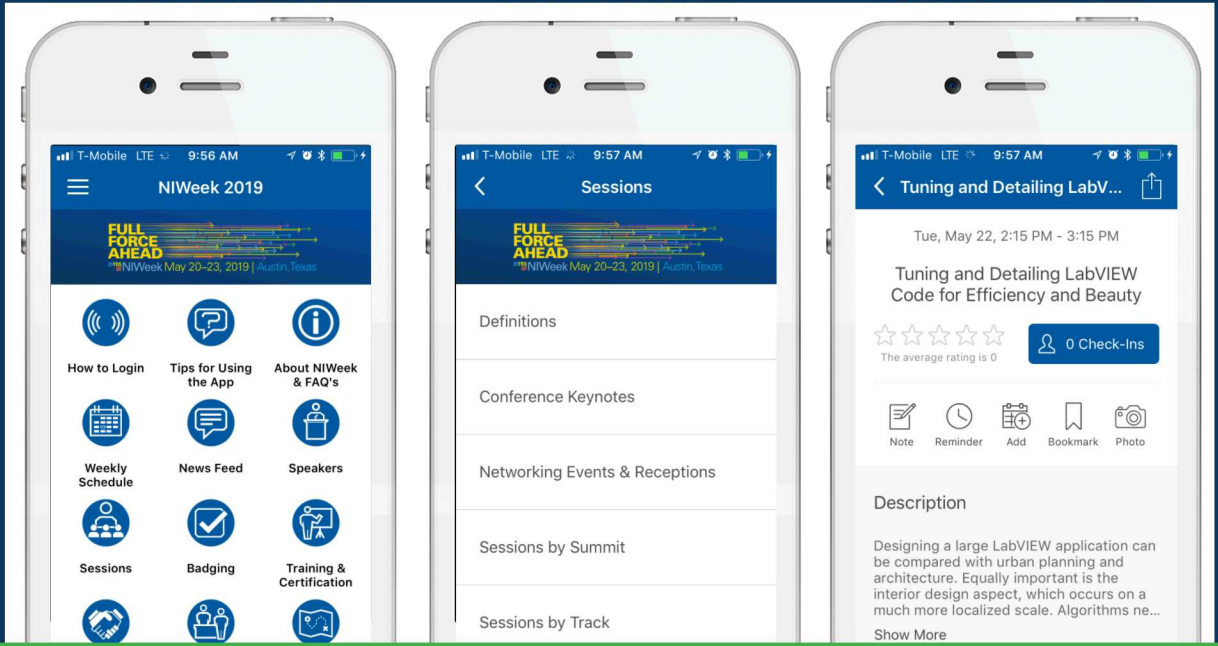
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Download and Login to the NIWeek Mobile App

VIEW WEEKLY SCHEDULE

FIND YOUR SESSION

BUILD YOUR SCHEDULE



More Information at ni.com/niweek


NATIONAL
INSTRUMENTS™

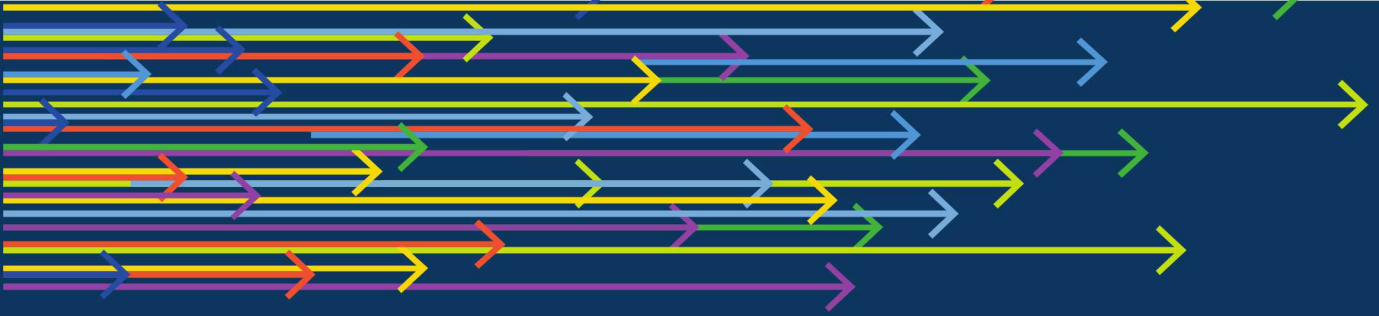
Scott Walkington

Sandia National Laboratory

Team Lead

CLD & CTD

NATIONAL INSTRUMENTS



Novel Approaches to Pass Data Between TestStand and LabVIEW

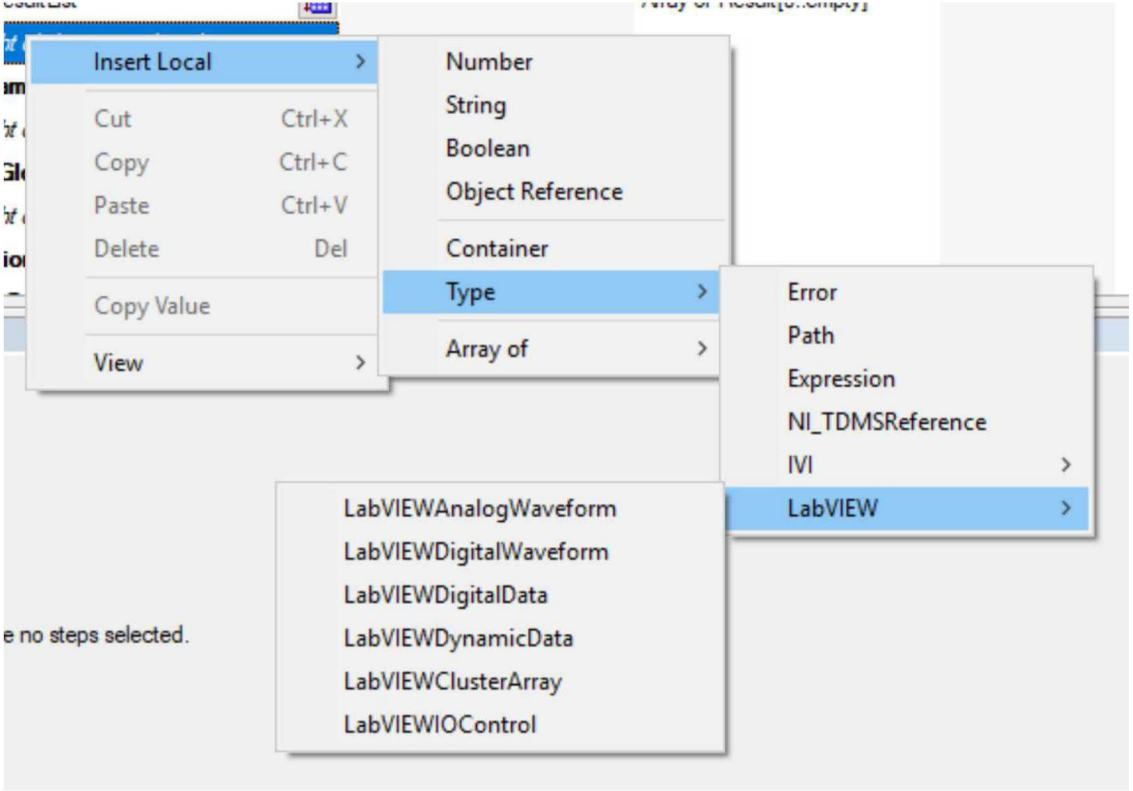
Who has struggled with deciding what is the best approach to share/pass data between TestStand and LabVIEW? This talk will examine the pro's and con's of using variants, variants with attributes, objects, images, DVR's, clusters, pass by reference, pass by value, queues, and events to share/pass data between LabVIEW and TestStand.

NATIONAL INSTRUMENTS

Outline

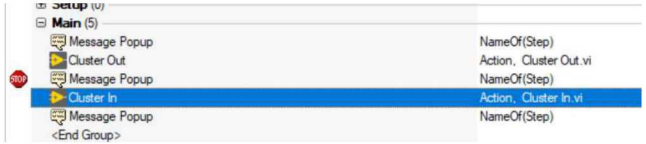
- Pass by Value
- Pass by Reference
- Clusters
- Objects
- Variants
- Variants with Attributes
- Images
- DVR’s
- Queues
- Events

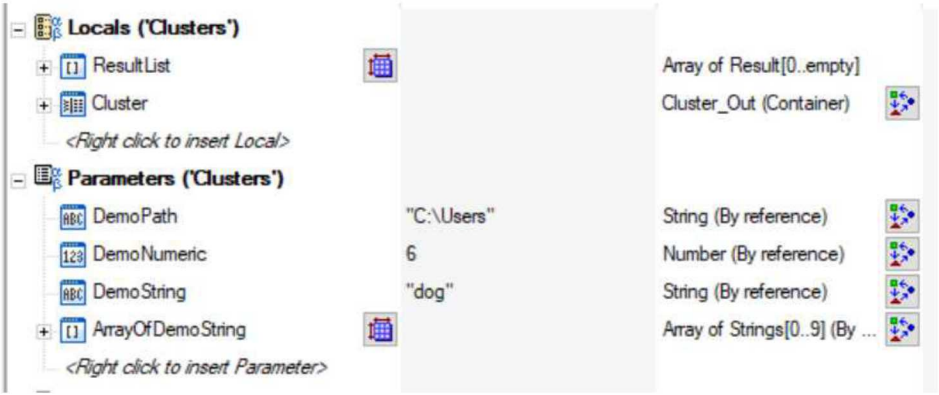
Standard Variable Options

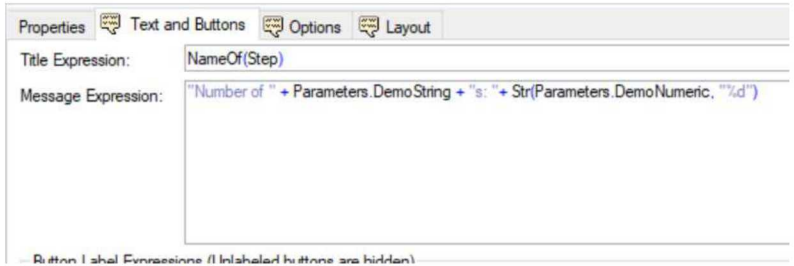


NATIONAL INSTRUMENTS

Basic Structure of LabVIEW VI's in Demo

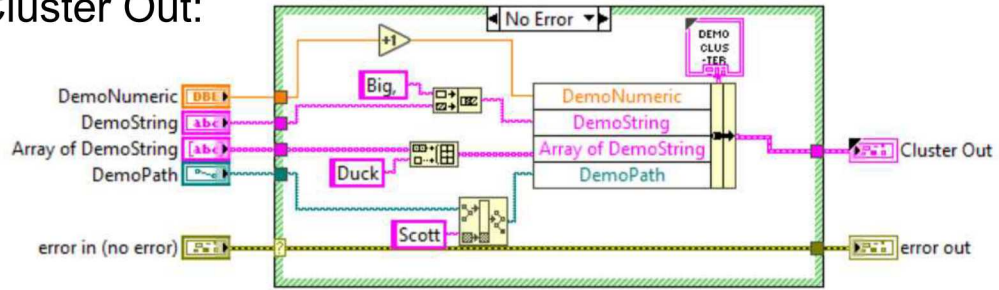




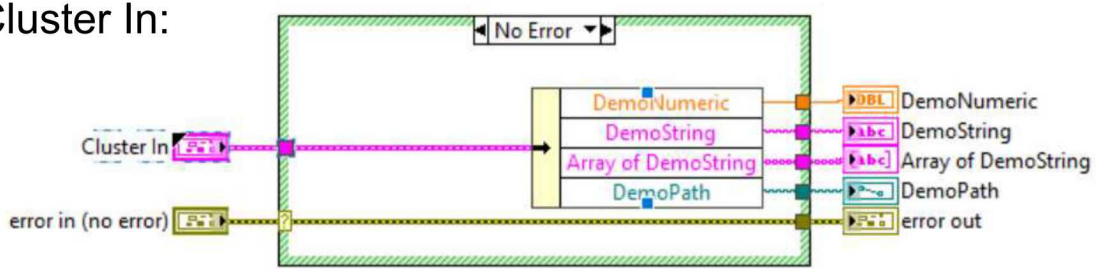


Basic Structure of TestStand Sequence in Demo

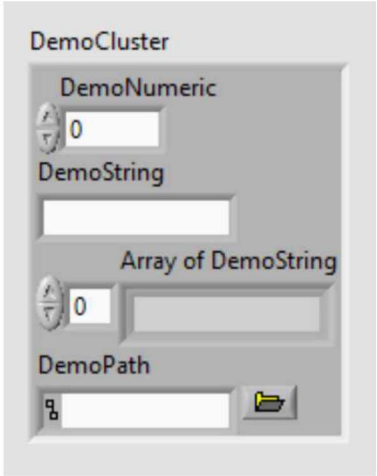
Cluster Out:



Cluster In:



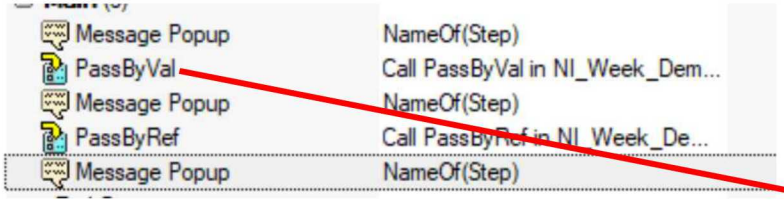
Cluster:



NATIONAL INSTRUMENTS

Pass by Value

- Parameter value passed into subsequence does not get passed out



- Pro or Con: Variable of Calling Sequence stays the same

Parameter Name	Type	Log	Default	Value	How Passed	C
DemoNumeric	Number	☐	☐	Parameters.DemoNumeric	by value	
DemoString	String	☐	☐	Parameters.DemoString	by value	

Subsequence:

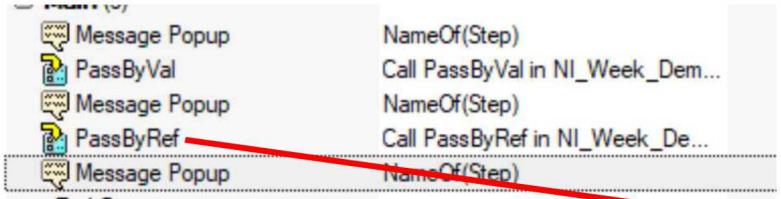
Main (3)

f(x) Change Dogs to Cats	Parameters.DemoString = "Cats"
f(x) Increment Number	Parameters.DemoNumeric = Parameters.DemoNumeric +1
Message Popup	NameOf(Step)
<End Group>	

NATIONAL INSTRUMENTS

Pass by Reference

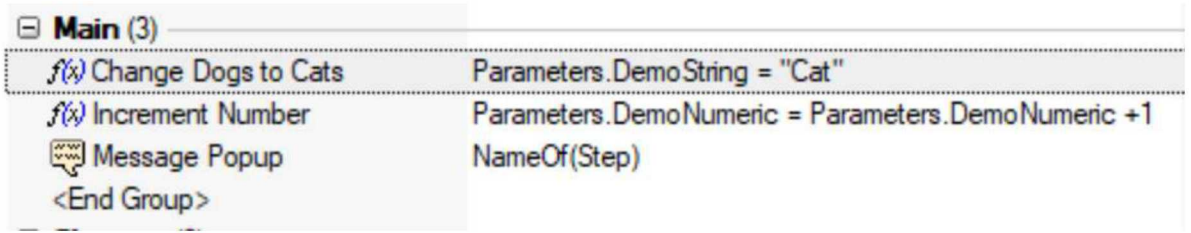
- Default setting in TestStand for all variables



- Pro or Con: Variable of Calling Sequence is updated

Parameter Name	Type	Log	Default	Value	How Passed
DemoNumeric	Number	☐	☐	Parameters.DemoNumeric	by reference
DemoString	String	☐	☐	Parameters.DemoString	by reference

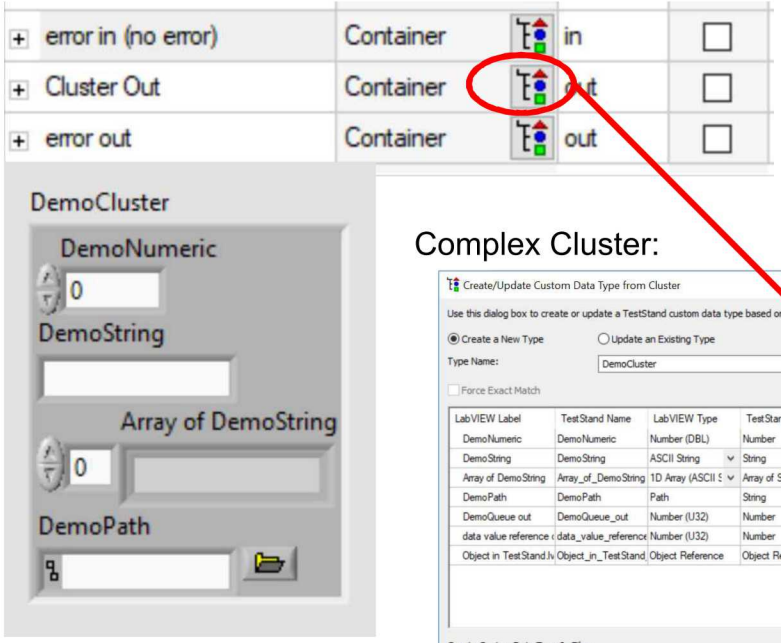
Subsequence:



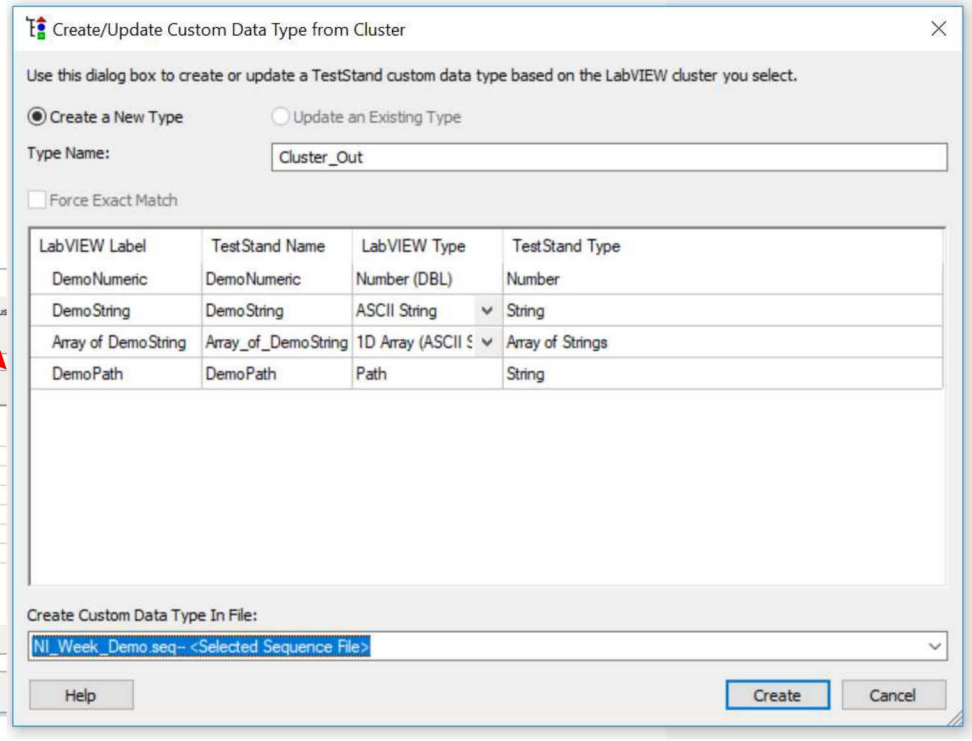
NATIONAL INSTRUMENTS

Clusters

- Need to “Create/Update Custom Data Type from Cluster”
- Good for passing data in Sequential steps



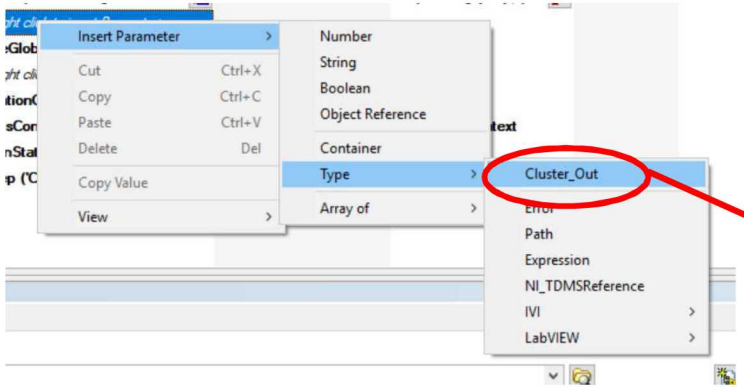
Complex Cluster:



NATIONAL INSTRUMENTS

Clusters Continued

- Custom Data types appear in Insert -> Type-> Custom Data Type

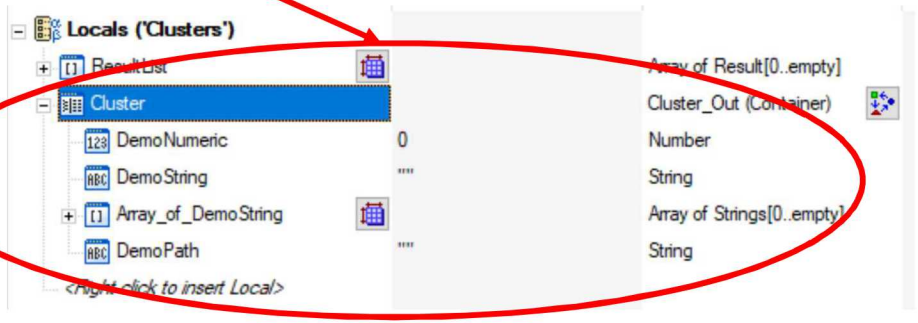


Pro's:

- Data easily visible in TestStand
- Custom data type can be used easily in multiple places
- Better to use Custom Data Type then creating a container with variables
  - Changes to a Custom Data Type will populate throughout sequence

Con's:

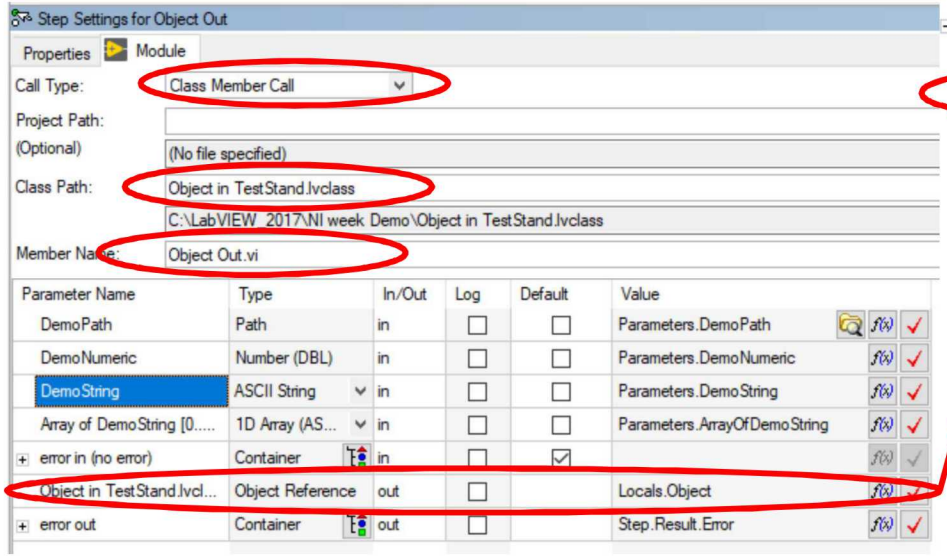
- If using just a container and you change the cluster then you must change the container everywhere
- Not very flexible if still making lots of changes in Cluster/Type def.



NATIONAL INSTRUMENTS

Objects

- Object Oriented Programming fully supported in TestStand
- Ok for passing data in Sequential steps



Pro's:

- Supports Dynamic dispatch
- Encapsulates Data
- Cannot accidentally change data in TestStand
- Good for passing data in Sequential steps

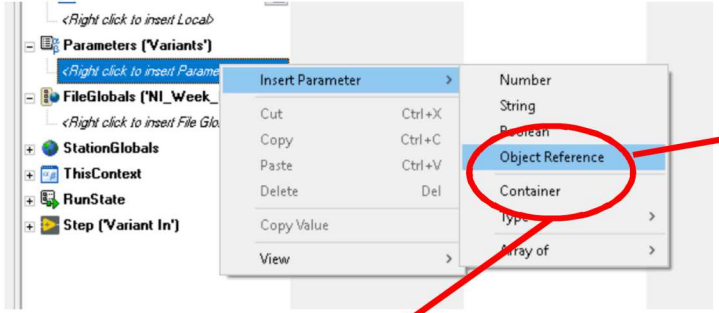
Con's:

- Can not see what is contained in object during execution
- Can not change data in TestStand

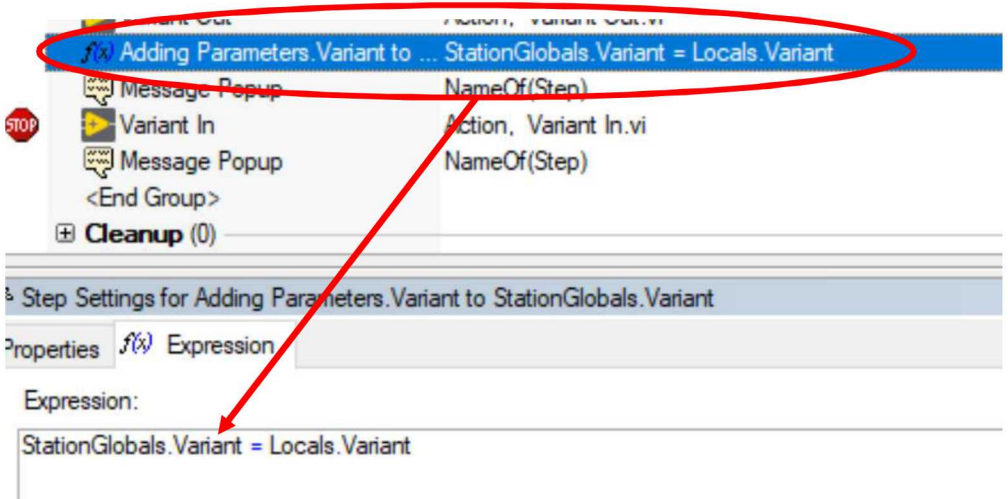
NATIONAL INSTRUMENTS

Variants

- Initially, Create an Object Reference or Container



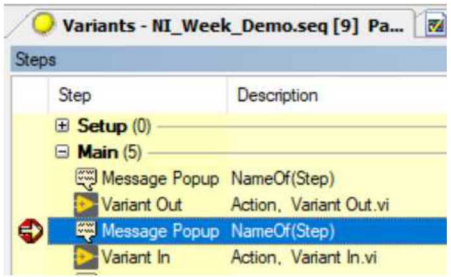
- For troubleshooting create variant in StationGlobals

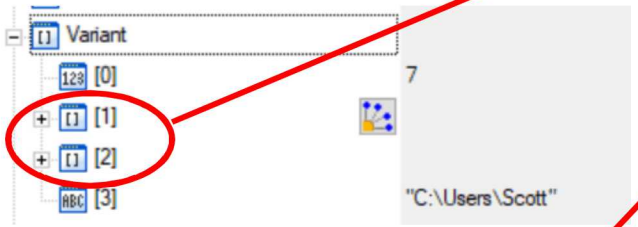


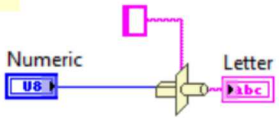
NATIONAL INSTRUMENTS

Variants Continued

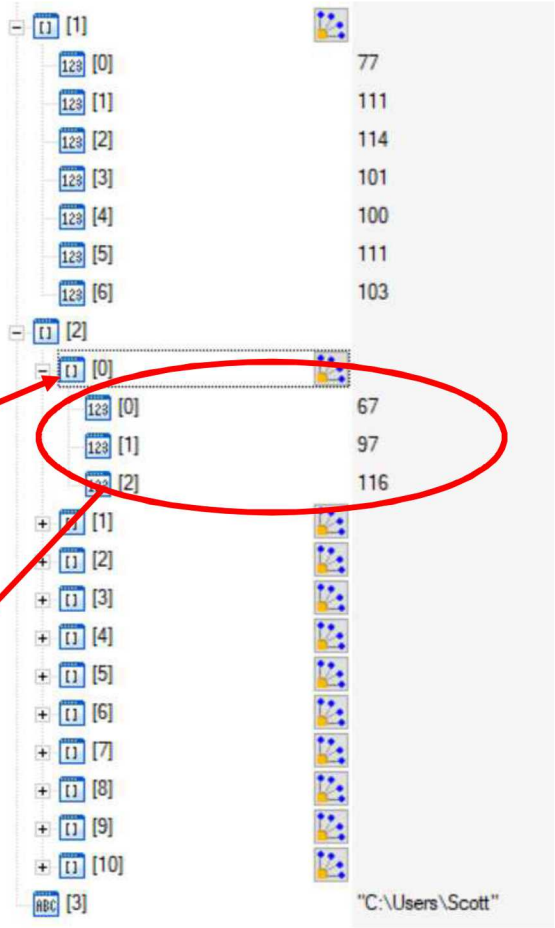
- During runtime Container/Object Reference changes to “Array of Results”
- Variant Values can only be seen in variables tab during execution
  - Variant created in StationGlobals will retain values after run for trouble shooting







67 97 116
“D” “o” “g”



NATIONAL INSTRUMENTS

Variants

Pro's:

- Works well for sequential steps
- Provides some idea of data contained
- Very flexible if you are still changing/deciding the data to pass between steps

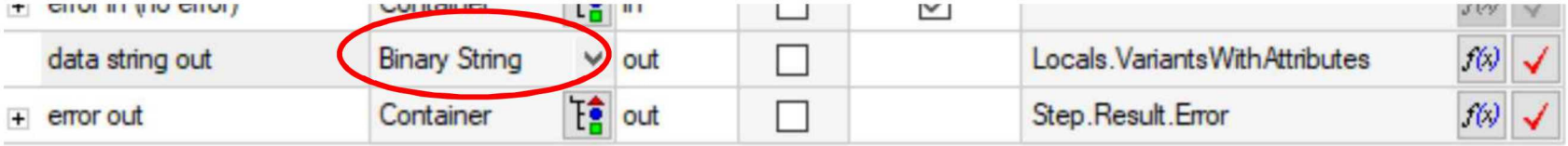
Con's:

- Can be easily overwritten
- Can be confusing when working with large clusters in the variant
- Can be confusing when passing around strings

Variants with Attributes

- Create String Variable

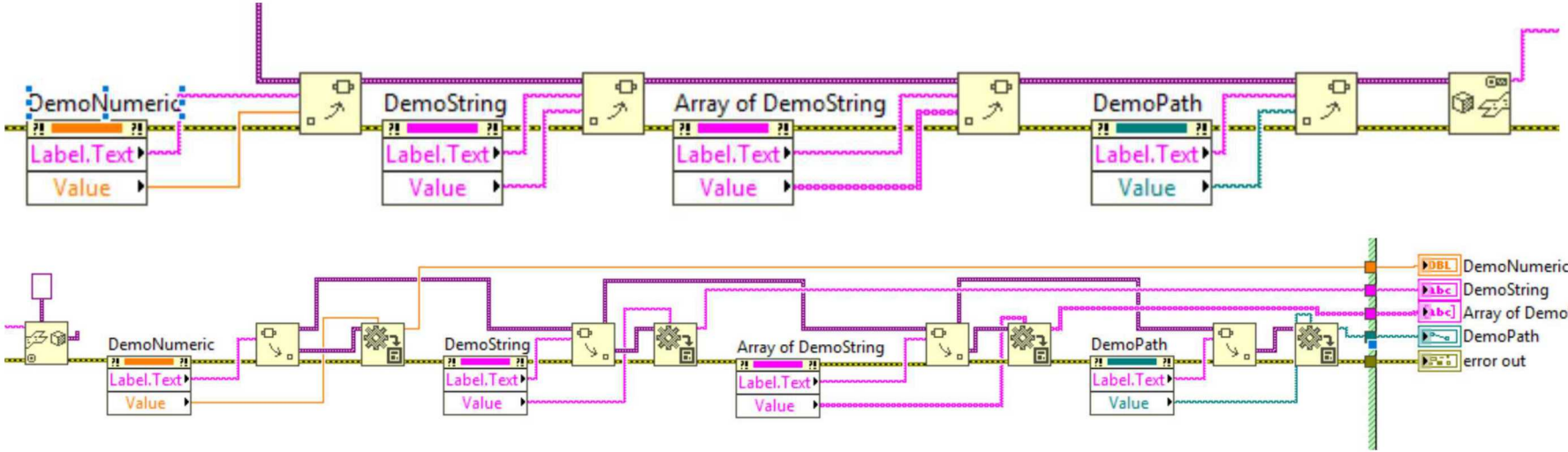
- MUST select Binary String



NATIONAL INSTRUMENTS

Variants with Attributes

- Must Use “Flatten to String”
  - Don’t use “Variant to Flatten String”
    - The help says this function strips out all variant attributes.
    - Guessing this is what TestStand is Leveraging under the hood



NATIONAL INSTRUMENTS

Variants with Attributes

Pro's:

- Works ok for sequential steps
- Very flexible if you are still changing/deciding the data to pass between steps
- Handy for Variant Attribute Look up tables

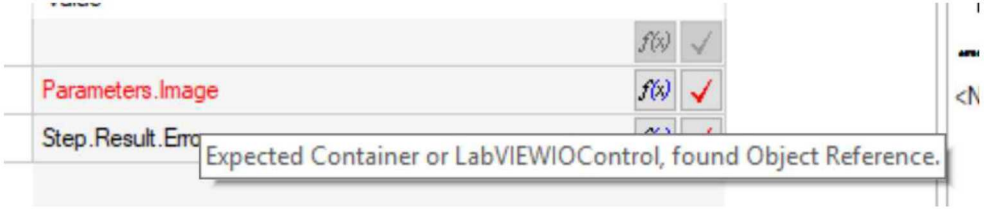
Con's:

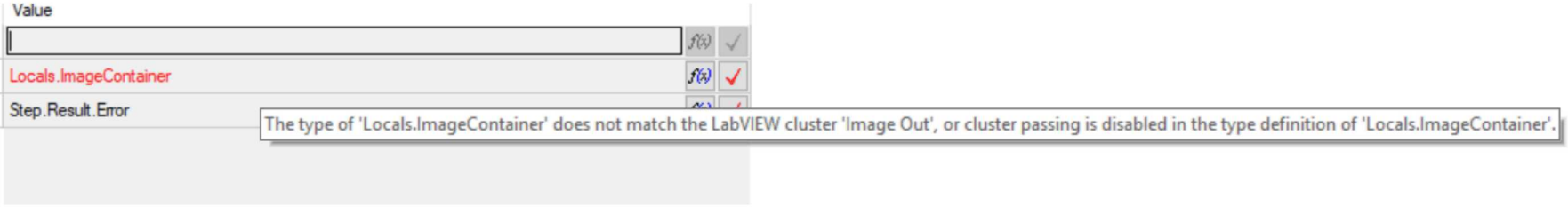
- Forget to change from ASCII string to Binary String
- Previous Variant stuff does not work.
  - Strips the attribute
- Long String of junk

NATIONAL INSTRUMENTS

Images

- Need NI Vision
- Could only get the LabVIEWIOControl to work
  - Not sure why the Container did not work

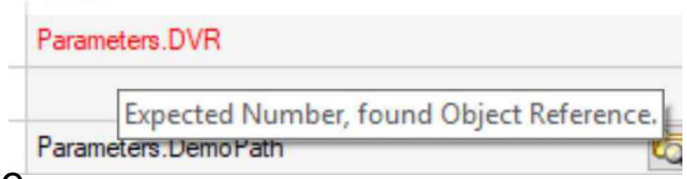




NATIONAL INSTRUMENTS

DVR’s

- Pass DVR as a Number
- Pass Reference to DVR as a Number
- Pass DVR as a Data String like variant attribute

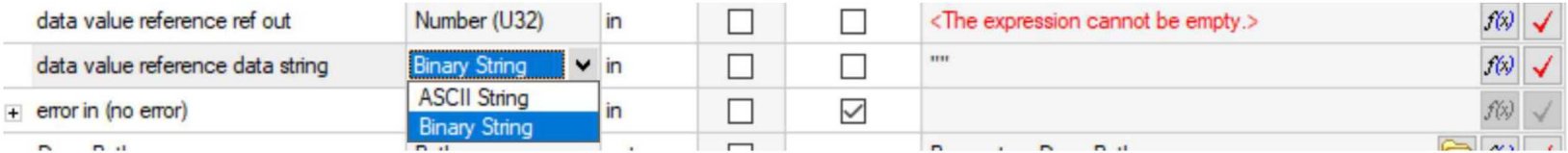


Pro’s:

- Can be nice for large data sets

Con’s:

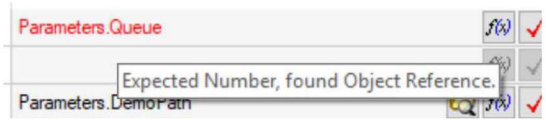
- Forget to change from ASCII string to Binary String
- Long String of junk

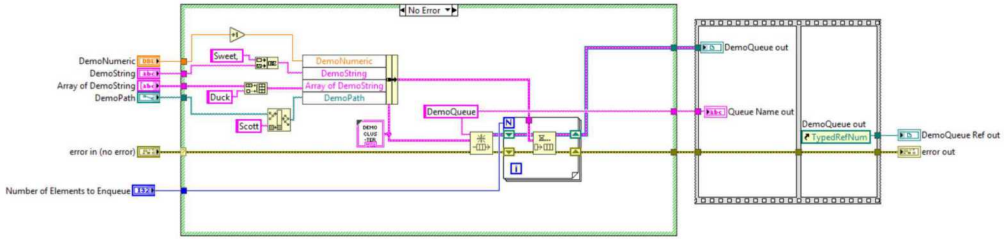


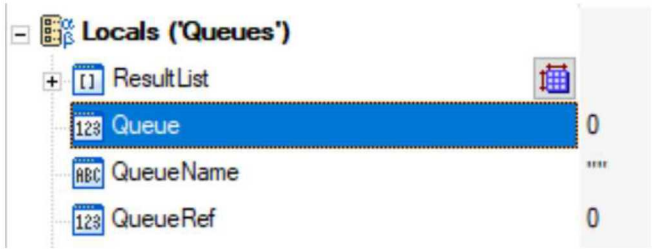
NATIONAL INSTRUMENTS

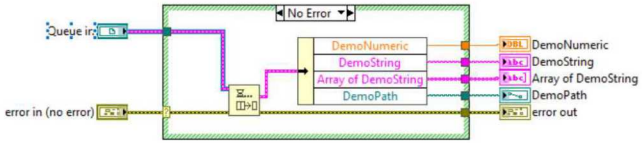
Queues

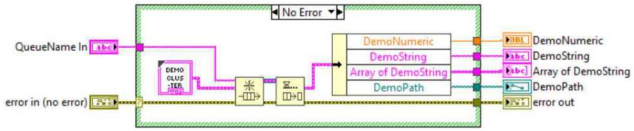
- Pass Queue as a Number
- Pass Reference to Queue as a Number
- Pass Queue as a Data String like variant attribute
- Pass Queue Name

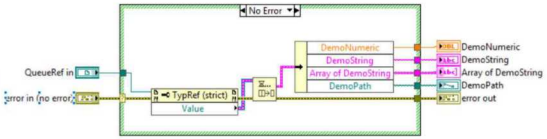












NATIONAL INSTRUMENTS

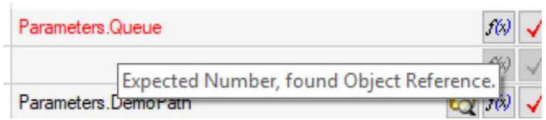
Queues Continued

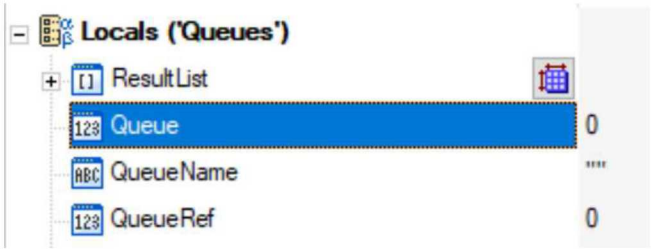
Pro's:

- Can store multiple pieces of data
  - Can be from multiple steps
  - Can then be processed in a single step

Con's

- Must make sure you enqueue enough elements otherwise code will hang waiting to dequeue element
- Can be confusing where elements are enqueue if you have many asynchronously running VI's

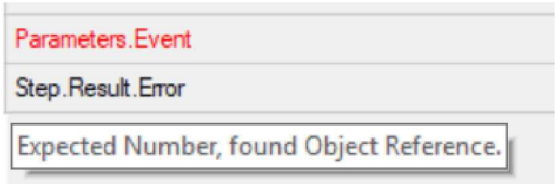


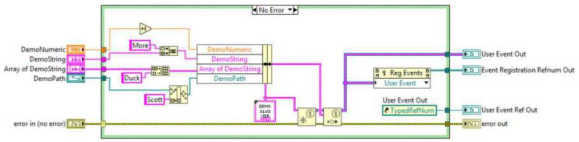


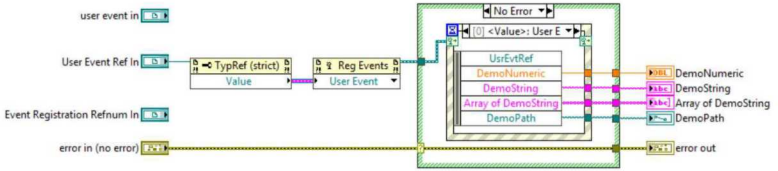
NATIONAL INSTRUMENTS

Events

- Cannot pass data between steps in traditional sense
  - Events are not like queues. Once the event fires if nothing is a registered listener then the data is gone.
- Must have a Asynchronously running VI registered to listen to event.
  - Can have as many clones running as you would like
  - All will simultaneously receive data.





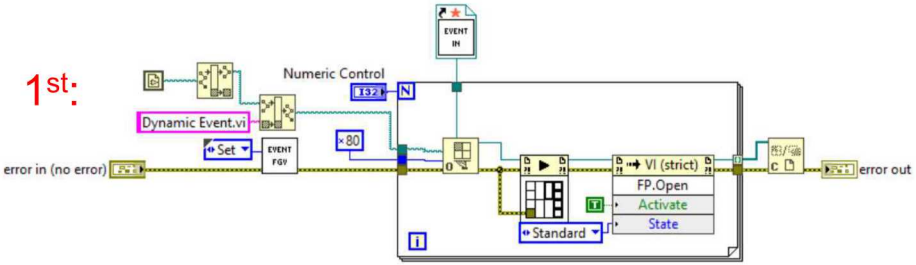


NATIONAL INSTRUMENTS

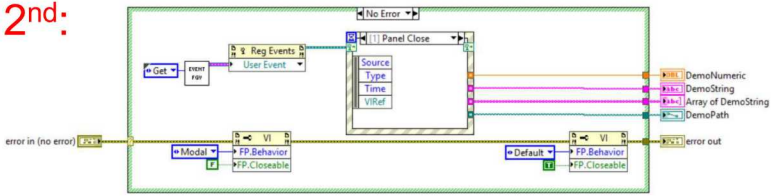
Asynchronous Dynamic Events

- Can easily do One to Many

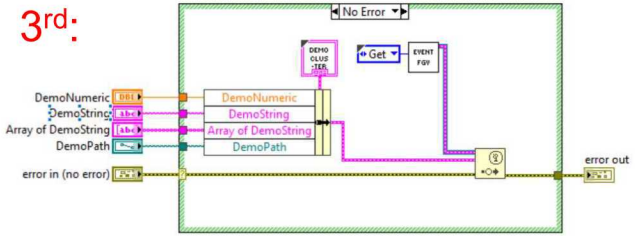
1st:



2nd:



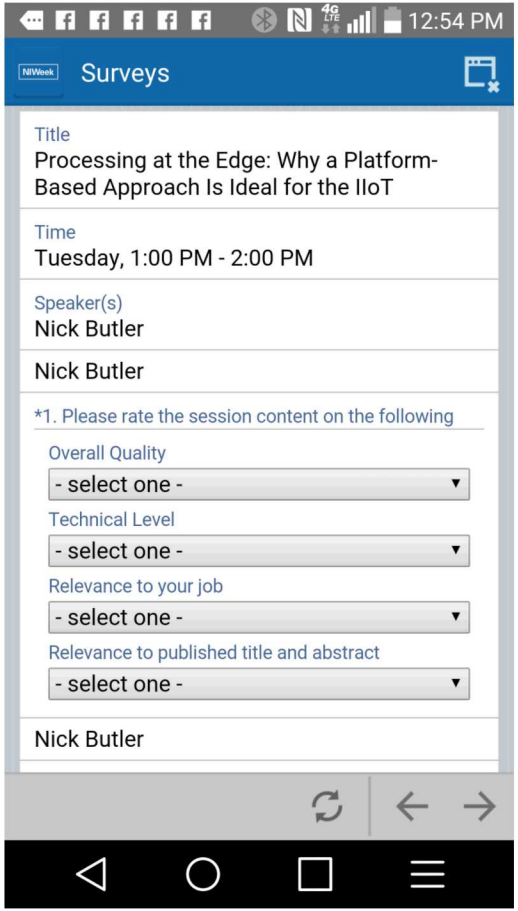
3rd:



NATIONAL INSTRUMENTS

Conclusion

- Can put almost anything into an Object, Cluster, Variant or Data String and pass it between steps
- LabVIEW fully supports Object Orientated Programing
- Queues can be used, just make sure to enqueue enough elements
- Dynamic Events can easily send messages/data to many VI's simultaneously
- Variants work well for sequential steps or Dynamic dispatch
- Varying abilities to view data in TestStand
  1. High – Cluster
  2. Medium – Variant
  3. Low – Object, Data String, Queues, DVRs,



Before you go, take the survey.

NATIONAL INSTRUMENTS

Slide Divider Topic Place Here

Stay Connected During and After NIWeek

ni.com/niweekcommunity

facebook.com/NationalInstruments

twitter.com/niglobal

youtube.com/nationalinstruments