NIWeek

ENGINEER NEXT

MAY 22-25, 2017 | AUSTIN, TEXAS

Personalizing NI TestStand with Custom Step Types

Grant Gothing

Bloomy

ATE Product Manager

grant.gothing@bloomy.com

Bloomy Quick Facts

- NI Platinum Alliance Partner
  - Awarded “Outstanding Technical Resources” 2013 and 2014
  - NI Certifications
    - 13 NI Certified LabVIEW Architects
    - 2 NI Certified LabVIEW Embedded Systems Developers
    - 2 NI Certified TestStand Architects
    - 3 NI Certified Training Centers
    - 13 NI Certified Professional Instructors
  - Published “The LabVIEW Style Book” © 2007, Prentice Hall
- ISO 9001:2008 Certified

About Grant Gothing

- 10 years at Bloomy
- ATE Product Manager
- CLA, CTA, CPI
- EFT Module for TestStand
  - Product manager and chief architect

Quick Survey

- Heard of TestStand Custom Step Types
- Developed a sequence using Custom Step Types
- Created or modified a Custom Step Type
- Deployed Custom Step Types to multiple developers/systems

Agenda

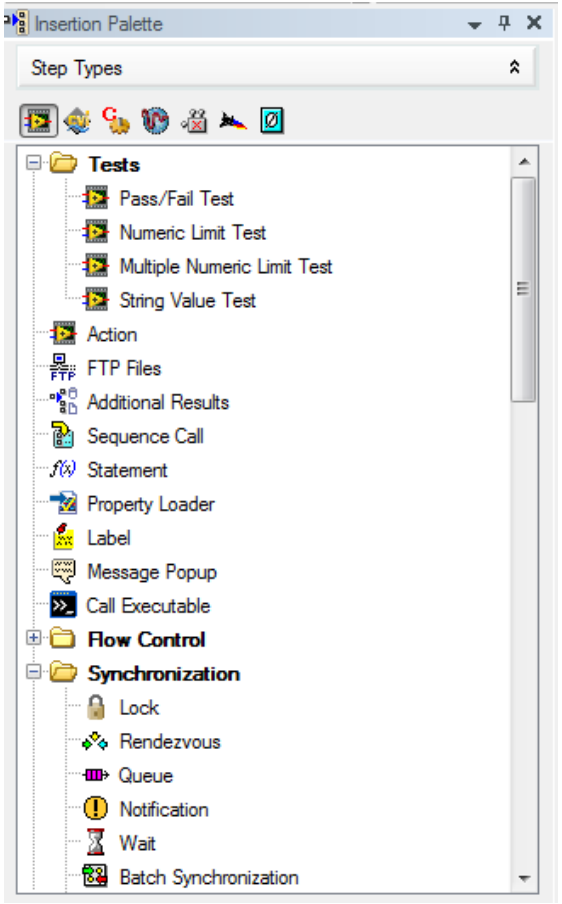
- Quick intro to TestStand Custom Step Types
- Why make a Custom Step Type?
- When should you create a Custom Step Type?
- Common applications of Custom Step Types
- How to make the most of your Step Type

Intro to Custom Step Types

Built-in Step Types

Generic interfaces

- Ship with TestStand
- Code modules & subsequences
- Statements, message popups, property loader
- Labels & comments
- Flow control & synchronization

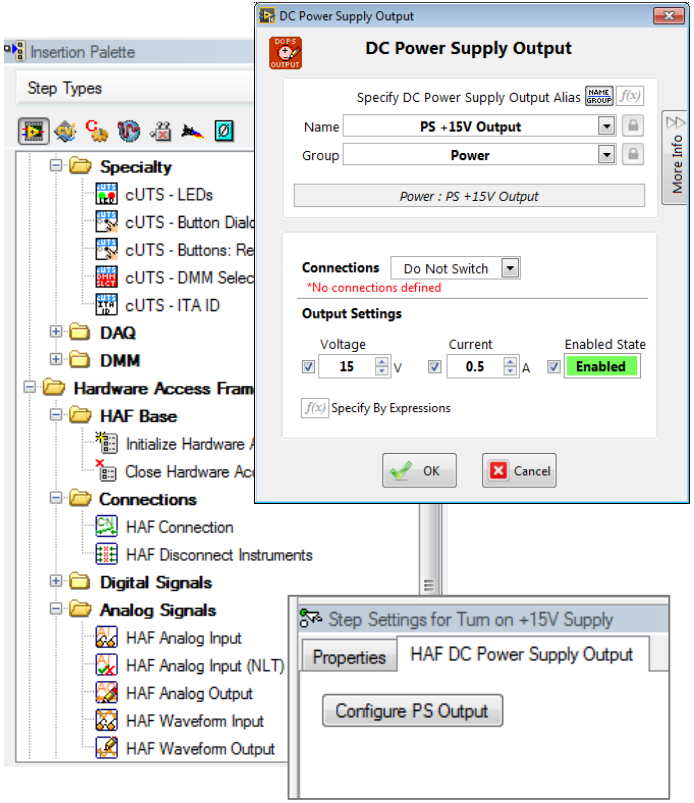


Intro to Custom Step Types

Custom Step Types

Application/architecture-specific interfaces

- User-Developed
- Wizard-like configuration dialogs
- May call code modules
- Hardware & file interfaces
- Custom data & reporting



Anatomy of a Custom Step Type

Step Data (variables)

- Configuration
- Results
- Custom data types

Step Properties

- Name, Description, Icon
- Run Options, Post-actions
- Default Expressions
- Default adapter and code module
- Disable Properties

Sub-Steps

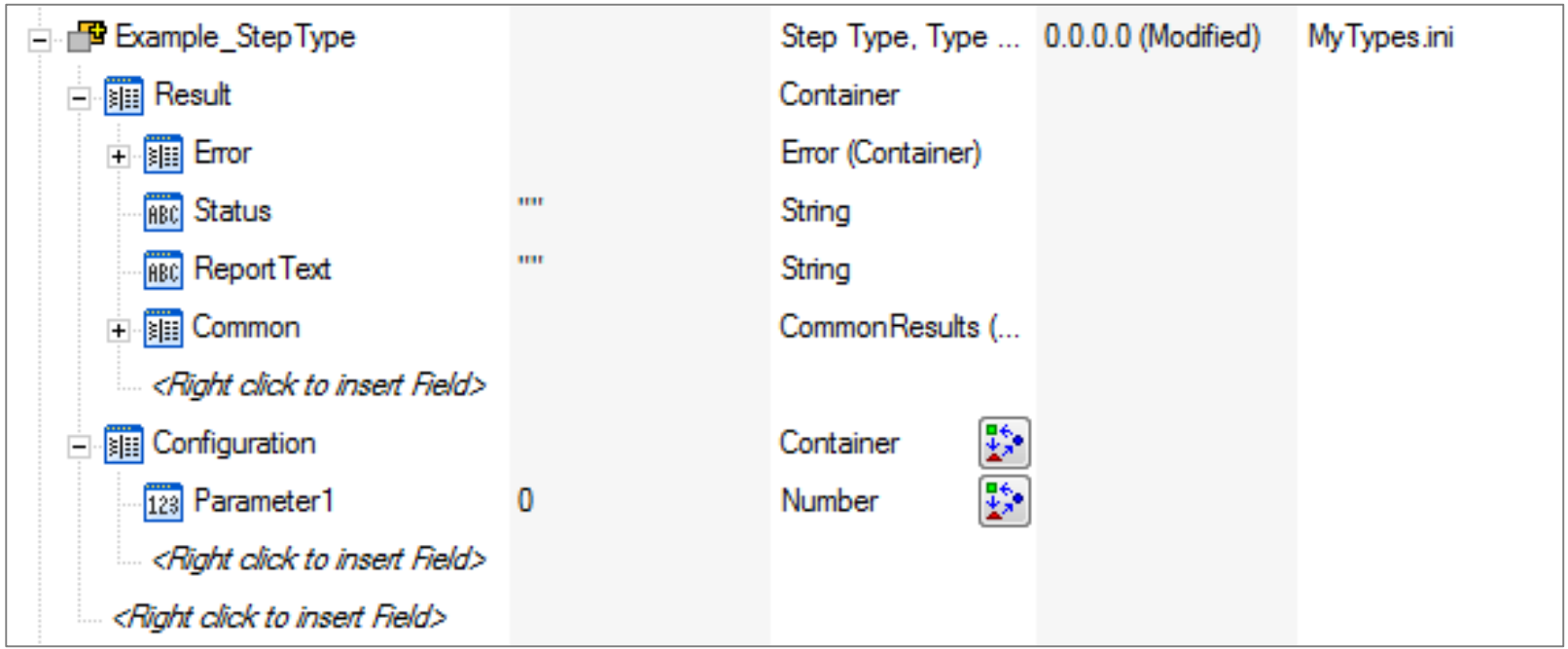
- Edit step code modules
- Pre-Step code modules
- Post-Step code modules

Module

- Adapter
- Code Modules

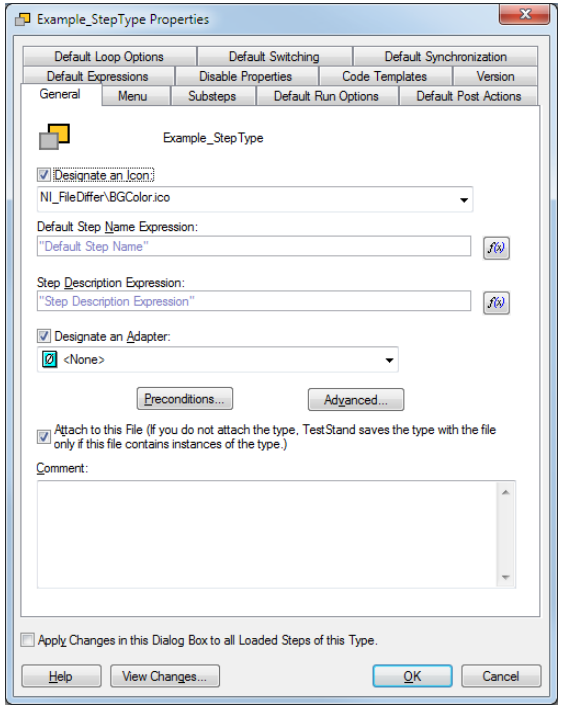
Anatomy of a Custom Step Type

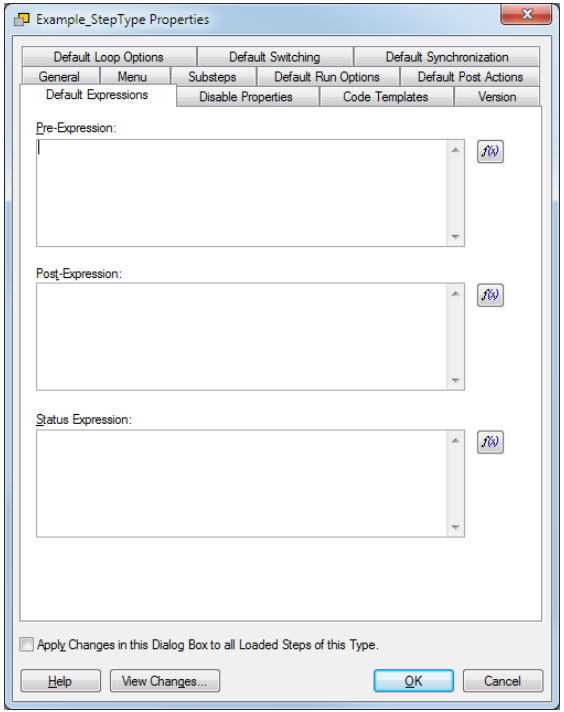
Step Data (variables)

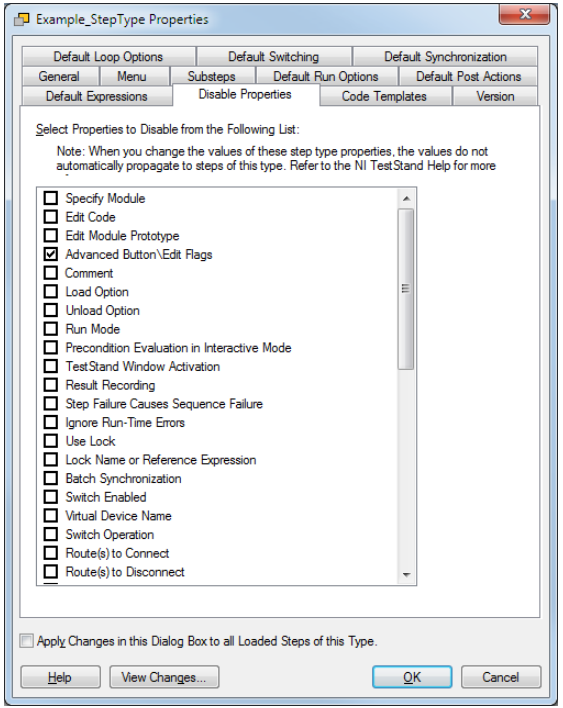


Anatomy of a Custom Step Type

Step Properties



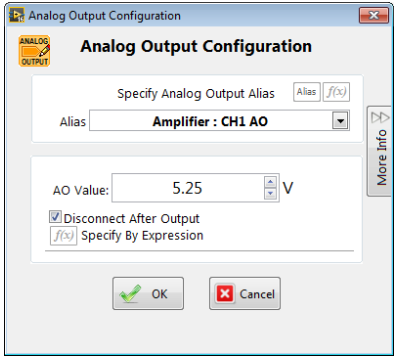


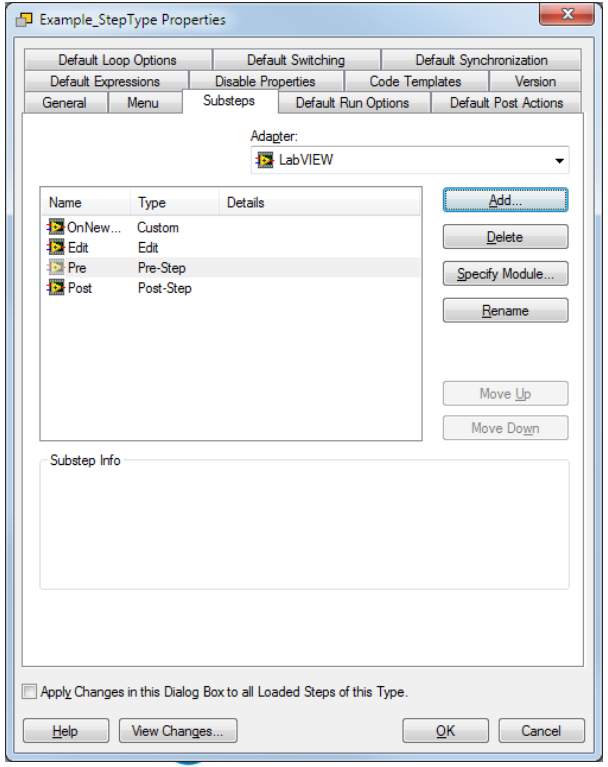


Anatomy of a Custom Step Type

Sub Steps & Code Module

Edit Substep(s)





Preconditions

Pre-Expression

Pre-Step substep(s)

Main module

Post-Step substep(s)

Post-Expression

Status expression

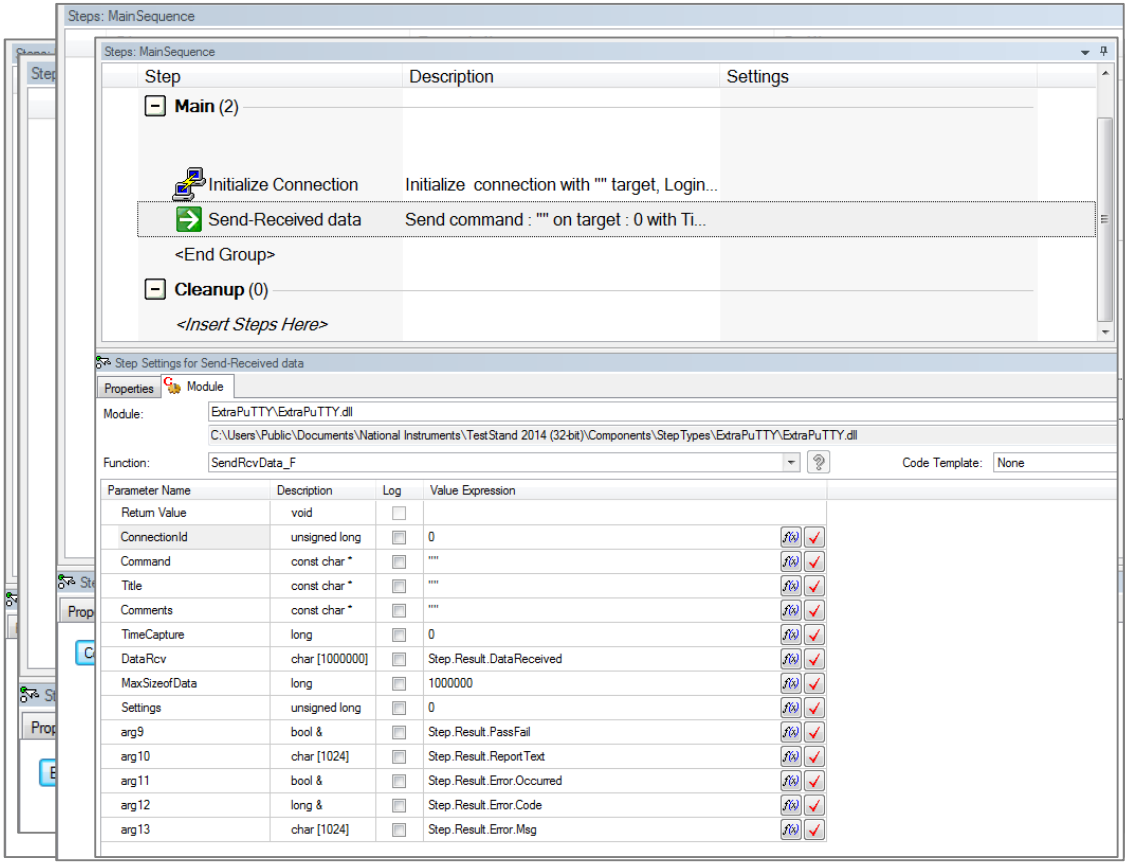
Post Actions

NATIONAL INSTRUMENTS

Common custom step types

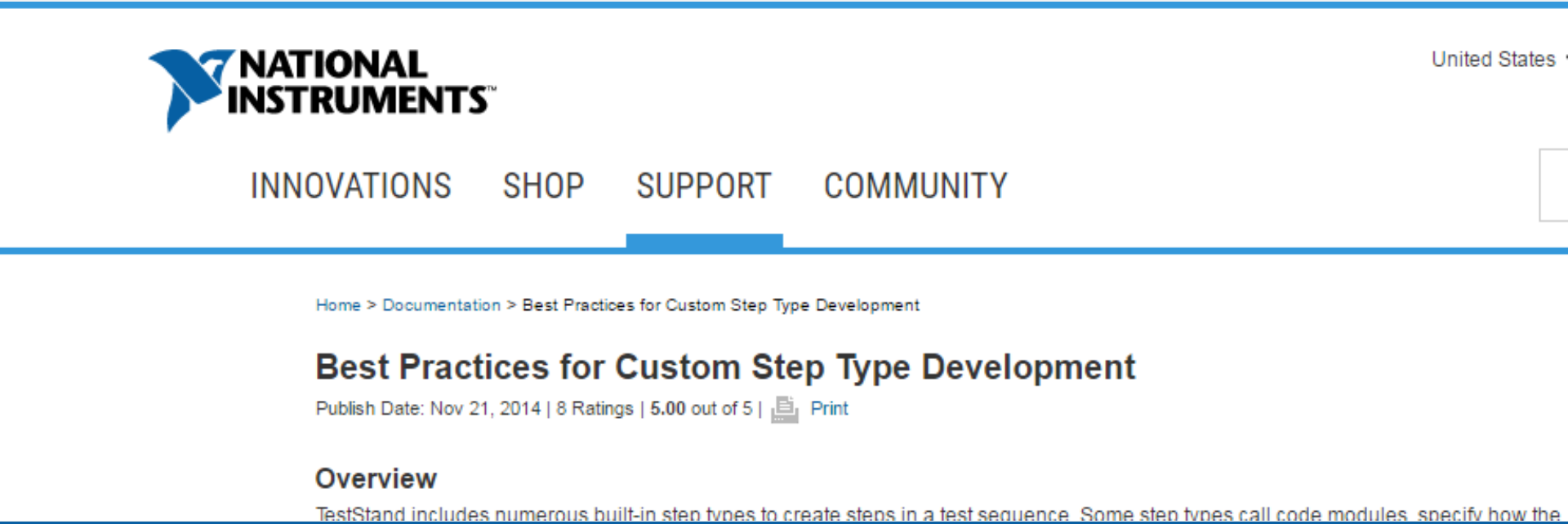
(Or built-in steps implemented as custom step types)

- Pre-TS 2016 Property Loader
- IVI Functions
- NI TCL Steps
- ExtraPutty



NATIONAL INSTRUMENTS

Bloomy EFT Module for TestStand



Best Practices for Custom Step Type Development

http://www.ni.com/product-documentation/8300/en/

Joe Spinozzi – Cyth Systems LLC

Why make a Custom Step Type?

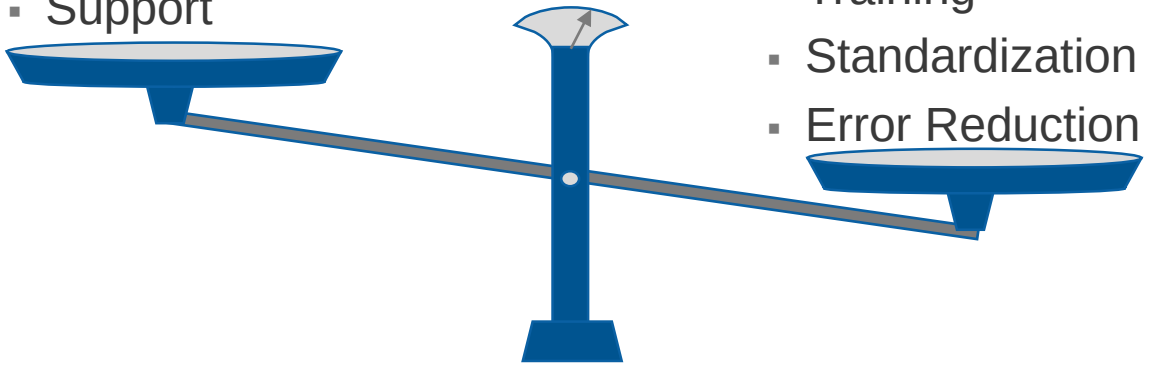
- Simplify
- Standardize
- Speed Up

Sequence Development

When should you make a Custom Step Type?

Step Type

- Development
- Debug
- Deployment
- Maintenance
- Support

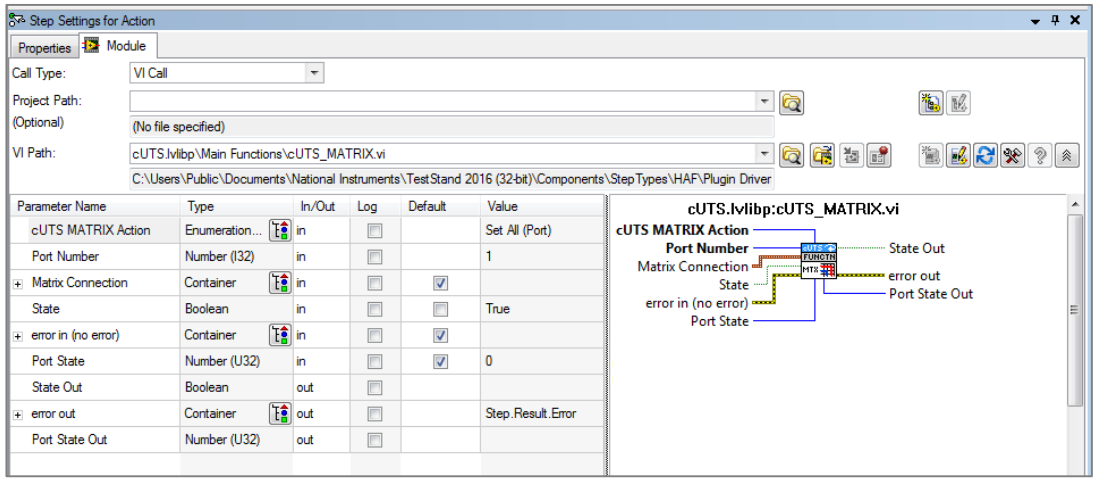


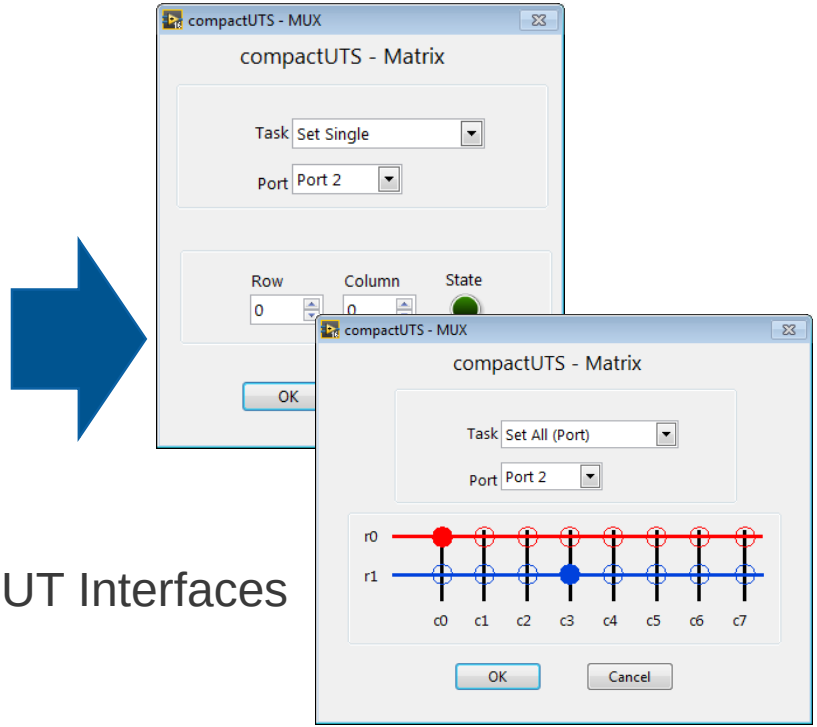
Time Savings

- Test Development
- Test Deployment
- Training
- Standardization & Reuse
- Error Reduction

Common applications of Custom Step Types

Action engine front-end



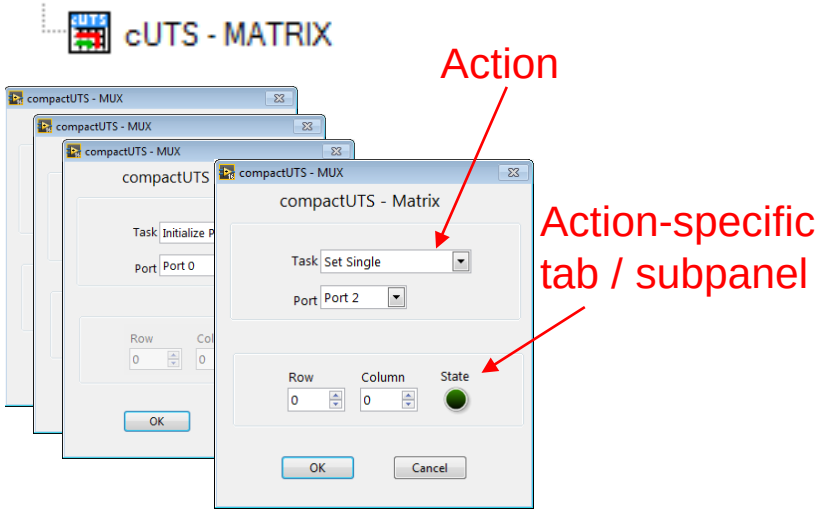


- Custom view for each action
- Show only relevant fields
- Reduce ambiguity
- Instrument & UUT Interfaces
- Custom file IO

Common applications of Custom Step Types

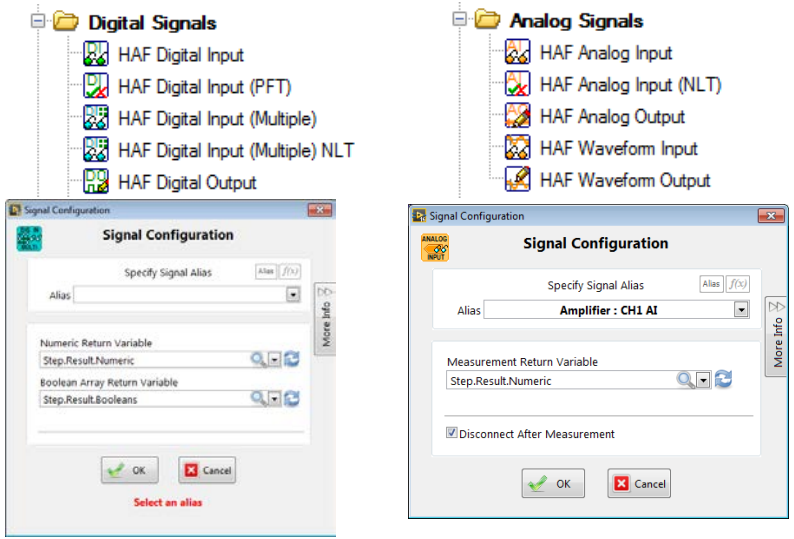
Action engine front-end implementation options

One step for all actions



Best when result data is the same for all actions

One step for each action

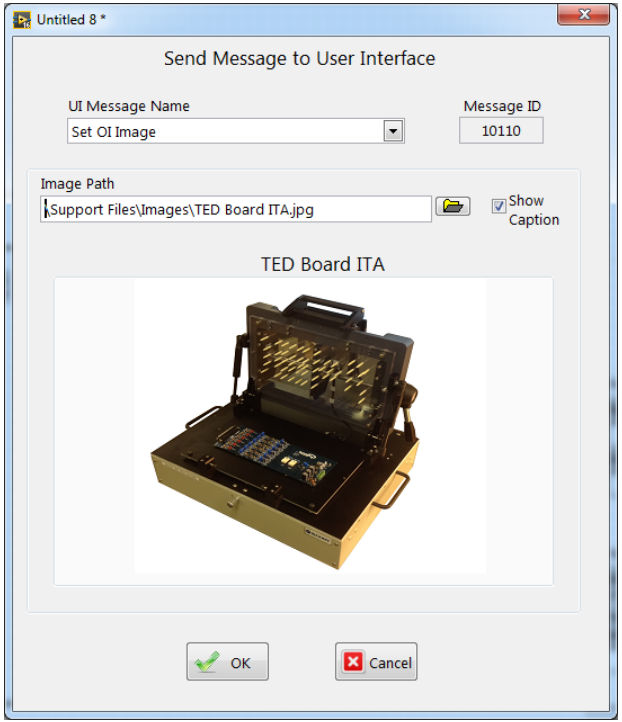


Best when result data is different from action-to-action

Common applications of Custom Step Types

Advanced TestStand Interactions

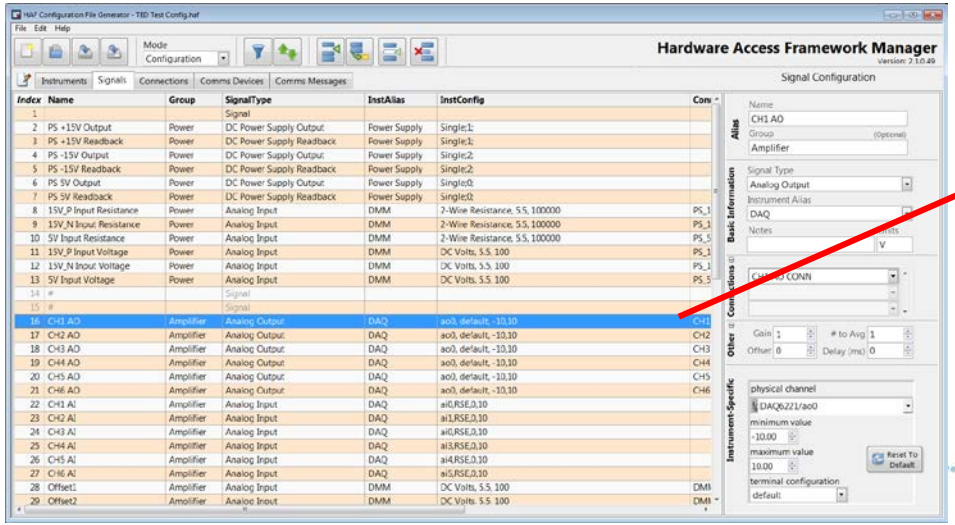
- Custom UI Messages
- Retrieving execution data
  - Serial & part numbers
  - Socket info
- Special process model interfaces

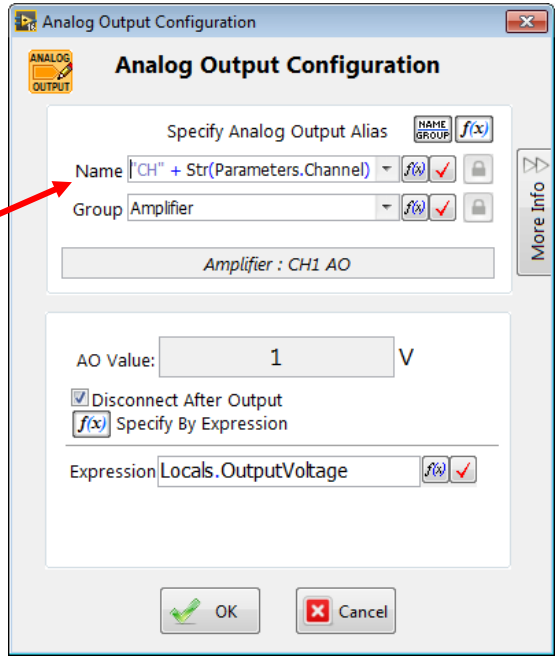


Common applications of Custom Step Types

High Level Architecture Interface

- Multi-IO interactions (Files + Hardware + TestStand API)
  - Hardware Abstraction Frameworks
  - Specialty Tests (eg. multi-limit waveform analysis)





Custom Step Type Guidelines

- Practical
- Intuitive
- Accessible
- Deployable
- Maintainable

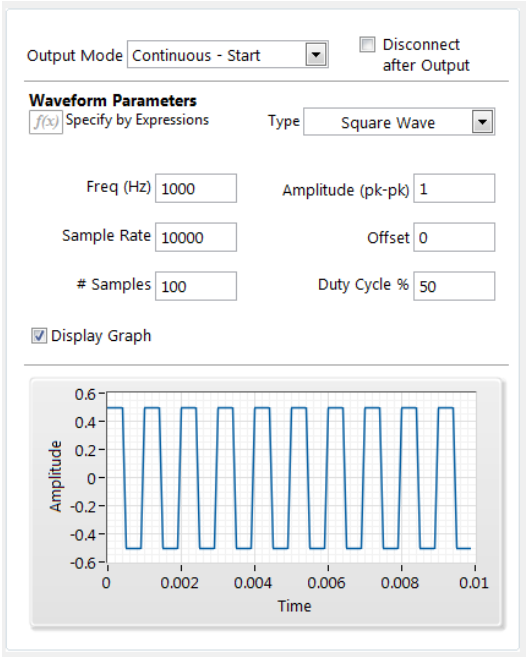
Custom Step Type Guidelines

- **Practical**
  - Time saved > time spent
    - Return on investment
  - Will it make developers lives easier?
  - Not every code module should be a CST
- Intuitive
- Accessible
- Deployable
- Maintainable

Custom Step Type Guidelines

- Practical
- **Intuitive**
- Accessible
- Deployable
- Maintainable

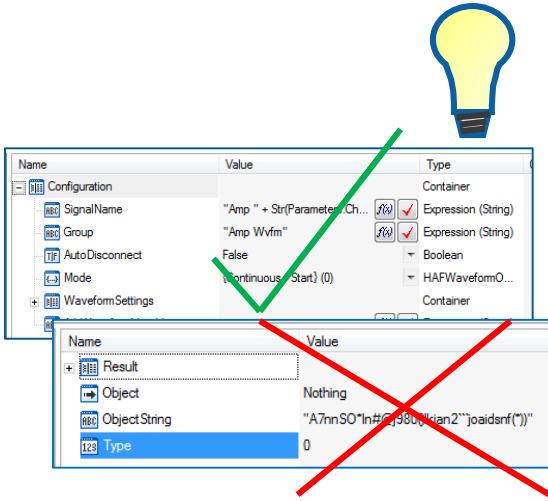
- Reduce ambiguity
  - Only show pertinent information
  - Include live feedback
- Tip strips and in-dialog help
- Use pre-existing conventions
  - Standard Variables
    - Eg. Step.Result.Numeric
  - Drop-down boxes

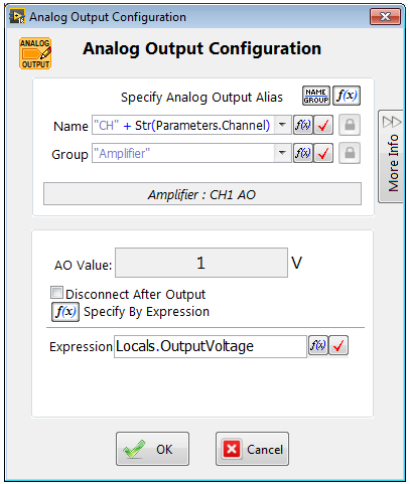


Custom Step Type Guidelines

- Practical
- Intuitive
- **Accessible**
- Deployable
- Maintainable

- Allow programmatic access
  - Store individual variables in properties
  - Avoid using flattened string data
- Step.Configuration for Setup
- Step.Result for return data
- Interface with TestStand API
  - SequenceContext
  - ExpressionEdit ActiveX object
  - Access variables (eg. locals, parameters)





Custom Step Type Guidelines

- Practical
- Intuitive
- Accessible
- **Deployable**
- Maintainable

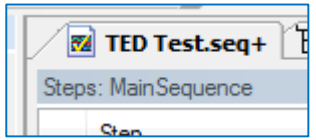
- Use Packed Project Libraries for source code

- Use TestStand Public directories
  - Relative paths, no changes to search directories
- Create installer using TestStand Deployment Utility

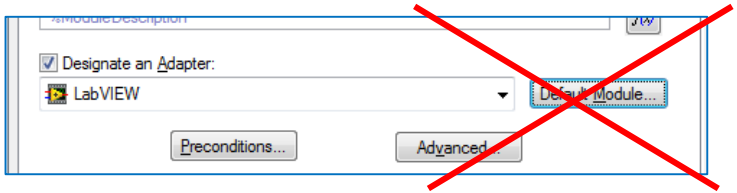
Custom Step Type Guidelines

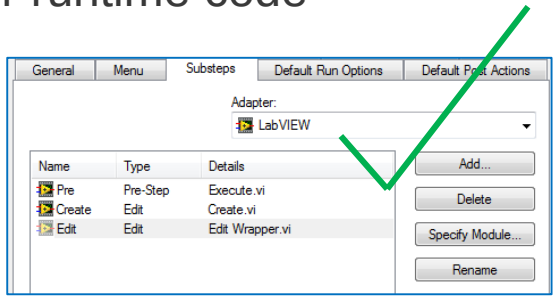
- Practical
- Intuitive
- Accessible
- Deployable
- **Maintainable**

- Use version numbers to update old sequences



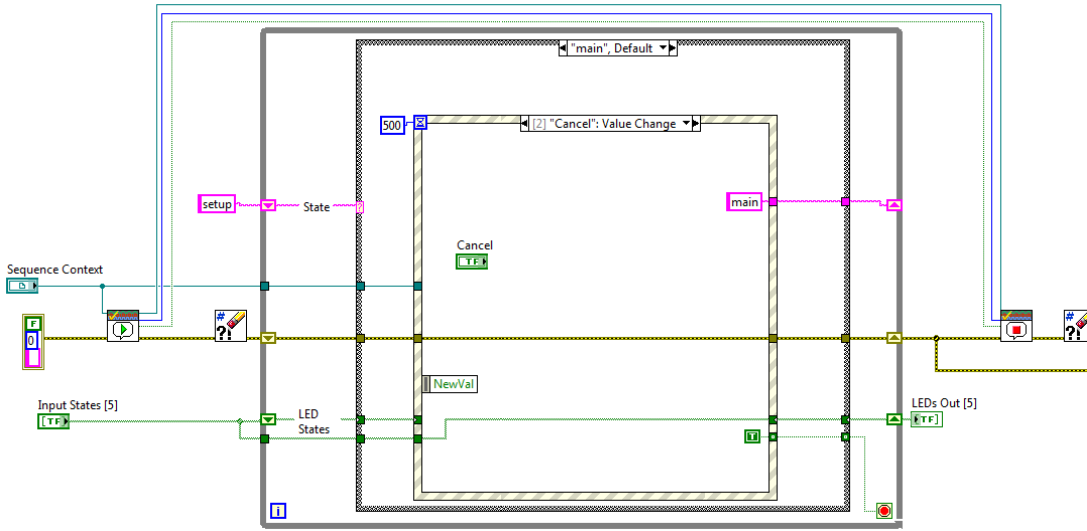
- Keep backwards compatibility, or change StepType Name
  - Don’t change property names
- DO NOT use the default module for runtime code
  - Use pre- or post- substeps





Additional Tips

- Debugging
  - Can’t Step Into Pre/Post Steps
  - Open VI first, set breakpoint
- Close References
  - Avoid TestStand object leaks
  - Avoid using API if unsure
- Use TestStand modal dialog and termination monitor functions
- Allow user to “Cancel”
  - Revert config to original



References

- Creating a Waveform Custom StepType and adding information to the ASCII report
  - http://www.ni.com/tutorial/3184/en/
- Best Practices for Custom Step Type Development
  - http://www.ni.com/product-documentation/8300/en/
- Bloomy EFT Module for TestStand
  - www.bloomy.com/eft-module-teststand
  - http://sine.ni.com/nips/cds/view/p/lang/en/nid/215009

Questions?

Grant Gothing

grant.gothing@bloomy.com

508 271-7340

Stop by the Bloomy pavilion at **Booth #205**

Follow our blog at www.bloomy.com/support/blog

Stay Connected During and After NIWeek

ni.com/niweekcommunity

facebook.com/NationalInstruments

twitter.com/niglobal

youtube.com/nationalinstruments

Please provide feedback on this session via the NIWeek Mobile App