



[ni.com](http://ni.com)

HOW DO I LOATHE THEE? LET ME COUNT THE WAYS.

# I hate about 10 things



Rob Humfeld

Certified LabVIEW Architect

Senior Project Engineer for Innoventor, Inc

August 16, 2005

# How This Talk Came About

## Three Observations:

1. Good programmers talk to themselves
2. Good programmers are detail-oriented
3. Fans want their team to be perfect

One day I noticed this guy sitting in my chair talking to me about some minor detail in LabVIEW saying:

***“I HATE it when LabVIEW does that”***

# The Purpose of This Talk

- Not to...
  - Bitch and moan
  - Create a Wish List
- But Instead to...
  - Find a larger context for our criticisms
  - Learn from our own expectations
  - Challenge NI with our ideas

# Who Am I to Say?

- Long-time user
- Professional LabVIEW programmer
- Certified Architect

But mostly...

- I'm a big fan of LabVIEW
- ...and fans want  
their teams to be perfect



# #1: Image, Marketing, and Expectations

## *Introduction*

How many times have you struggled to convince someone that LabVIEW is a "REAL" programming language?

- Programmers: "Tinkertoy" Language
- Scientists/Engineers: Data Logger Package

*"You mean LabVIEW can DO that?!?!?"*

# #1: Image, Marketing, and Expectations

## *Marketing*

- Target: scientists and engineers
  - It's all about the data
  - What about large program directors or plant managers?

LabVIEW delivers a powerful graphical development environment for signal acquisition, measurement analysis, and data presentation, giving you the flexibility of a programming language without the complexity of traditional development tools.



Acquire



Analyze



Present

# #1: Image, Marketing, and Expectations

## *Expectations*

- Customers expect fast and easy
- They don't realize requirements of good software
- Three-click solutions?
  - Which software development process steps did we skip?



LabVIEW is an open environment designed to make interfacing with any measurement hardware simple. With interactive assistants, code generation, and connectivity to thousands of devices, LabVIEW makes gathering data as simple as possible.

# #1: Challenge to NI

## *Better sell the POWER of LabVIEW*

- Opportunity coming: LabVIEW 8
- Encouraging find on ni.com:

### Looking for technical information on LabVIEW as a software development tool?

The National Instruments LabVIEW graphical development environment provides advanced functionality and performance that engineers and scientists can use to develop sophisticated applications. The technical resources below offer information on the built-in functionality, best practices, and add-on tools for developing powerful and maintainable NI LabVIEW programs.

Software Engineering and Code Reuse

Professional Application Design

LabVIEW as a Programming Language

## #2: Making Professional GUIs

### *Introduction*

When LabVIEW came out, just the fact that it allowed users to make user interfaces fairly easily was awesome.

Since then, more computer users have very high expectations for what a user interface should look and act like.

## #2: Making Professional GUIs

*Great LabVIEW features for creating GUIs:*

- Event Structure (and User Events)
- Grouping
- Locking
- Aligning
- Distributing
- Layering
- Grid (I turn off Panel Grid Alignment)
- Sizing tools and Dialog

## #2: Making Professional GUIs

### *Challenges*

- Positioning and Sizing window is annoying
- Powerful controls can be non-intuitive *(Treeview)*
- Properties list is long and confusing *(XYGraph)*
- Many controls have limited usefulness *(Path Control)*
- Meager choice of control styles
- Control editor limited and difficult

## #2: Lessons Learned

*Making professional GUIs is a skill to develop*

- A professional GUI does not happen by accident
- The user can tell if you designed a good GUI

Attend:

*"Beyond 'The Good, the Bad, and the Ugly'"*

*Wednesday 11:45AM*

## #2: Challenge to NI

*LabVIEW needs to make creating better GUIs easier*

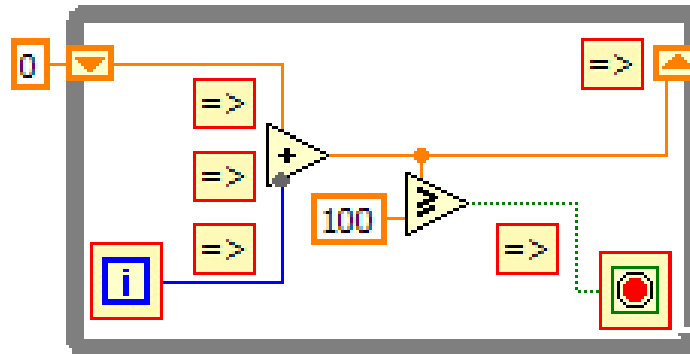
- Intelligent management of Active UI area
  - Allow user to create “active frame”
  - Re-locate and resize on run
  - Skip off-screen controls when tabbing
- Inherent support of customizable pop-up menus
- Review how users want to use complex controls
- Disable indicators by default
- Improve Control Editor

# #3: Great Idioms Inconsistently Applied

## *Introduction*

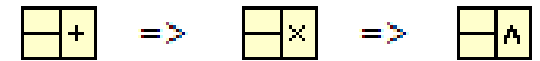
Idiom (n): a manner of speaking that is natural to native speakers of a language

A good program teaches its user the proper idioms



# #3: Great Idioms Inconsistently Applied

*Example: Compound arithmetic "change mode"*



What about numeric functions?



Comparisons?



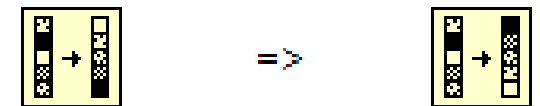
Numeric conversions?



Sort 1-D Array?



String constant?



# #3: Great Idioms Inconsistently Applied

- Array Functions (row and column tip strips)
  - What about Array Size output?
    - Multiple I32 terminals
    - Cluster/unbundle by name
- Shift registers
  - Replace sequence locals?

# #3: Great Idioms Inconsistently Applied

- String Functions
  - Special characters not in Context Help
- Format Date/Time String
  - No scan function?
- Array function index inputs resize to array dimensions
  - Why doesn't Format Into String?
- Format Into String doesn't always require format
  - Why does Array to Spreadsheet String?

# #3: Great Idioms Inconsistently Applied

- “NaN” for numerics is very useful
  - Not possible for Time Stamp datatype
  - What about similar auto-conversion for string constants
    - Convert to “” for Empty String, to Tab const for “\t”, etc.
- “Find All Instances” is great, but none = trouble
  - I don’t want to look in vi.lib
  - Skip locked block diagrams
  - Add these options to search results dialog

## #3: Great Idioms Inconsistently Applied

- When loading a VI, can “skip” a missing subVI
  - Remember when the next VI looks for the same one
- Can hide a subpanel frame or LED decal
  - Why not for all controls?

## #3: Lesson Learned

*Create Great Idioms, Teach Them, Be Consistent*

*Review how your users use your software*

# #4: Expected Functionality That's Missing

## *Introduction*

Whether it is fair or not, users come to the table with certain expectations.

These expectations are formed in various ways before exposure to a given application, but they shape the user's perception of the application.

# #4: Expected Functionality That's Missing

## *Expectations set by NI*

- Debugging features for subpanels & cloned VIs
- Properties for decorations
- Wire error into logic functions

## *Expectations set by the Operating System*

- Cut and Paste on pop-up menu
- High-resolution icons
- Customization of environment

# #4: Expected Functionality That's Missing

## *Expectations set by other programs*

- Edit all properties within one dialog
  - Many missing from control properties dialog
  - Text formatting via dialog
- Copy/Paste text formatting (Word)
- Edit properties of multiple controls (Excel)
- Copying and pasting array data (Excel)

## #4: Challenge to NI and Lesson Learned

*The user will be disappointed  
if you fail to meet their expectations*

- Know your user
  - Expectations
  - Goals
  - Abilities
  - Previous experience

# #5: Memory and Graphics Performance

## *Introduction*

LabVIEW provides programmers with a valuable service: it makes memory management transparent.

Unfortunately, the cost is ignorance about what is really going on inside of the computer.

Programmers may not know how their implementation determines a program's performance.

# #5: Memory and Graphics Performance

*The Great Mystery: What's really going on?*

- Large arrays
- Giant clusters
- Continually resizing strings and arrays
- Screen drawing
- Graphical controls
- IMAQ / Video playback

## #5: Challenge to NI

*Improve LabVIEW's graphics and memory performance*

*Continue to expand the number and utility of tools available for understanding and evaluating resource use*

# #5: Lesson Learned

*Expert users should understand:*

- How LabVIEW uses memory
- How to evaluate program performance
- How to take advantage of multi-threading
- How to evaluate memory performance
- How to improve memory performance
- LabVIEW's graphics limitations
- LabVIEW's tools for evaluating resource use

# #6: Application Builder

## *Introduction*

With the Application Builder, programmers can create executables and even executable installers from LabVIEW code.

Sweet.

Unfortunately, using it effectively can be difficult.

# #6: Application Builder

## *Complications*

- Path specification
- MAX configurations
  - Installation
  - Security
- Support file locations/structure
- Support drivers

## #6: Lesson Learned (and Challenge to NI)

*Continuously evaluate and improve  
the usability of your software*

*For powerful or complex applications,  
protecting the users from errors  
is a major challenge*

# #7: The “This Should be Easier” Feeling

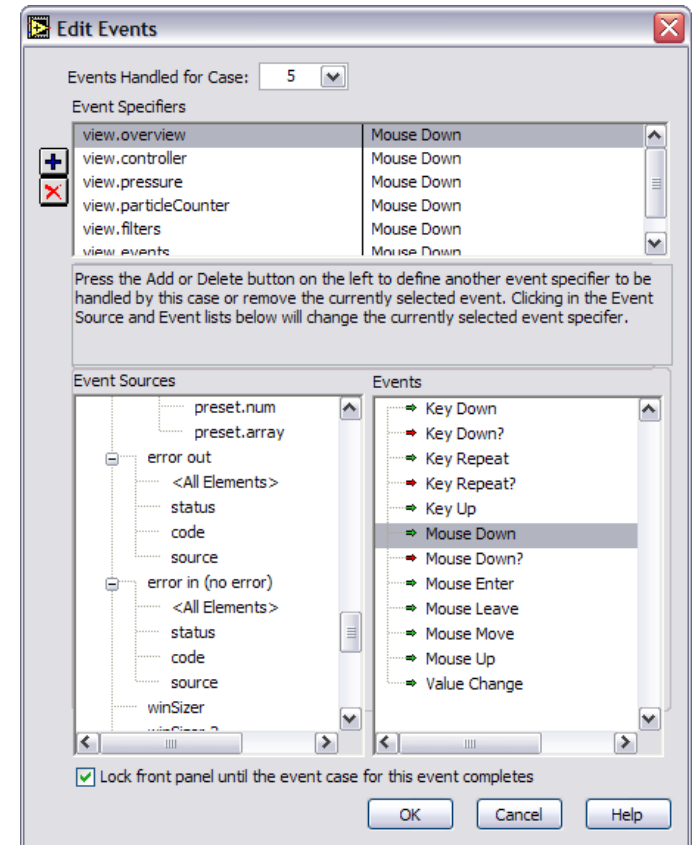
## *Introduction*

Once you have learned LabVIEW's paradigms it is incredibly easy to understand how to use most functions.

That makes it more frustrating when you come upon something that “just shouldn't be that hard...”

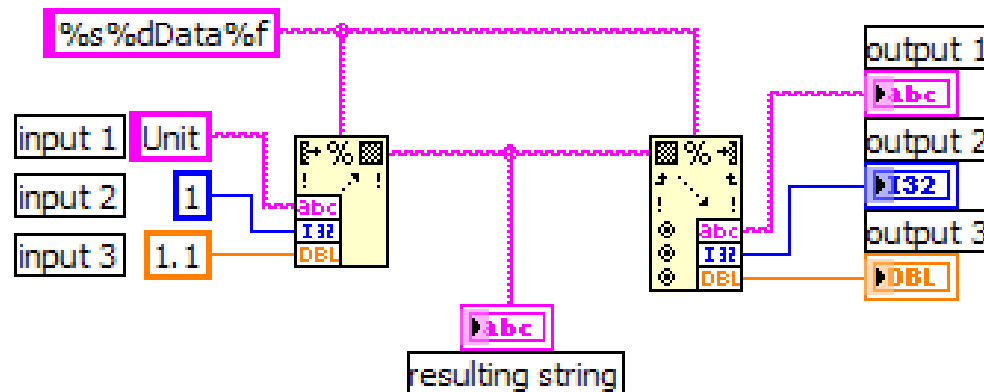
# #7: The “This Should be Easier” Feeling

- Editing Event Handler
  - Why have clusters expanded by default?
  - Why is “Value Change” on bottom?
  - What about events for Graph Cursors?
  - Multiple Events per case non-intuitive



# #7: The “This Should be Easier” Feeling

- String parsing



Why are the outputs different from the inputs?

# #7: The “This Should be Easier” Feeling

- “Create SubVI” function
- “Commenting out” code
- Copying and pasting array data
- Connector pane for new VIs
- Location of new controls on front panel
- Location of new control terminals on diagrams

## #7: Lessons Learned (and Challenge to NI)

*Continuously evaluate and improve  
the usability of your software*

*Making an existing feature work better  
is sometimes more useful  
than adding a new feature*

*Continually solicit and consider user feedback*

# #8: The Express VI Mentality

## *Introduction*

Express VIs.

The idea is that LabVIEW is going to automatically do stuff for me.

Great! I like to be served as much as the next guy...  
...except when it doesn't do what I want, the way that I want, or as well as I expect.

# #8: The Express VI Mentality

- Express VIs
- Express VI code generation
- Auto-routing wires
- Automatic error handling
- Terminals as icons
- DAQmx tasks
- Automatic numeric formatting
- See item #1: Image, Marketing, and Expectations

## #8: Lessons Learned

*Sometimes easier functionality is a tradeoff  
of user power, control, and efficiency*

*Newbies are people too.  
Your software must serve people with diverse abilities.*

*God, I am old and crotchety.  
I need to try something new once in a while.*

# #9: I Have So Much Still to Learn

## *Introduction*

LabVIEW used to be so simple...

- Basic lab testing applications
- DAQ and GPIB
- Acquire, Analyze, Present

Everyone thought I was a genius!

# #9: I Have So Much Still to Learn

- The “World of LabVIEW” is HUGE
  - LabVIEW RT, DSC
  - LabVIEW Toolkits
  - LabVIEW Application Builder
  - Measurement and Automation Explorer
  - Motion, Vision, Simulation, PDA, FPGA, etc...
- New versions can be difficult adjustments for user
- And don't even talk about the hardware updates...

# #9: Challenge to NI

*Be aware that you are swamping us*

- Develop good, user-relevant examples
- Explain new features in Help (*Timed-Loop*)
- Continue excellent level of support
- Teach the AEs about the new features
- Continue webcast learning opportunities
- Improve web-searching
  - Indicate age of hit in index of results

# #9: Lessons Learned

*It is incumbent upon us to...*

- Put in the time to learn new features
- Volunteer for beta-testing
- Participate in peer-to-peer forums
- Take advantage of NI resources
  - Seminars
  - Webcasts
  - Discussion Forum

*NI provides an awesome level of support*

# #10: Other Items

## *Introduction*

Some things defy categorization

## #10: Other Items

- Why do we still have icons in triplicate?
- Execution highlighting is slow
  - Add a speed wheel control
- Error clusters
- Complex data types
  - Waveform, DDT, Timestamp, Variants
- Distribute software on DVD

## #10: Lessons Learned

*There'll always be some room for improvement*

*Don't expect your customers  
to just accept whatever you want to give them*

# #10: Challenge to NI

*Keep on keeping on*

LabVIEW ROCKS!

We may bitch about LabVIEW occasionally,  
but NI does a killer job making LabVIEW  
the best programming tool on the market.

*...and LabVIEW 8 already addresses a number of  
the items in this talk*

# Contributors

*Thank you to everyone who contributed*

Andy Marshall

Sam Hammond

David Asher

Mark Balla

Ed Dickens

Bob Duet

Phillip Brooks

Brent Goldberg

Bob Young

Norman Kirchner

Andrew Kay

David Hakey

Petar Rakocevick

Sheldon Stokes

Marita Ogden

Daniel Press

Steve Coughlan

Devin Maxey-Billings

Joe Zoller

Parker Russell

Paulo Martins

Dan Ponce

Mike Valentino

Didier Jud

*Updated Presentation: [10Things@innoventor.net](mailto:10Things@innoventor.net)*