Interfaces in G

Stephen Loftus-Mercer
(Aristos Queue)

LabVIEW R&D
Principal Software Architect

Coming soon...

... to both LabVIEW 2020

(or as soon as my team can make it)

& LabVIEW NXG

A. What is an interface type?

An introduction to G interfaces

Basic Definition

- In general context:
  software interface = the set of functions that one code module uses to request functionality from another module.

- In today's context
  interface = a new data type that defines a set of tasks that an object can do without specifying how those tasks must be done.

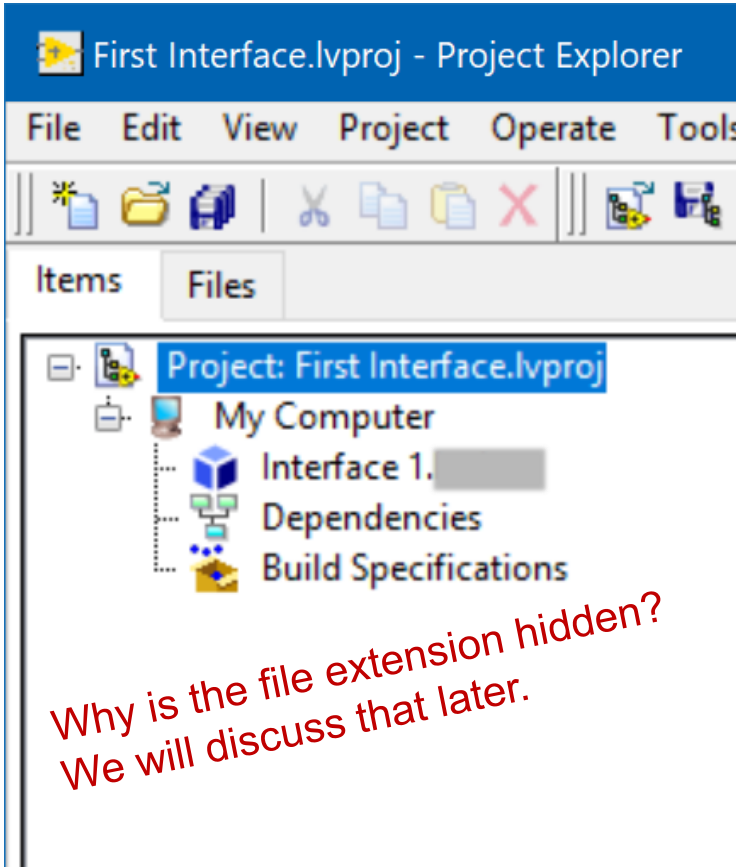
Basic Definition

Identity – ~~State~~ – Behavior

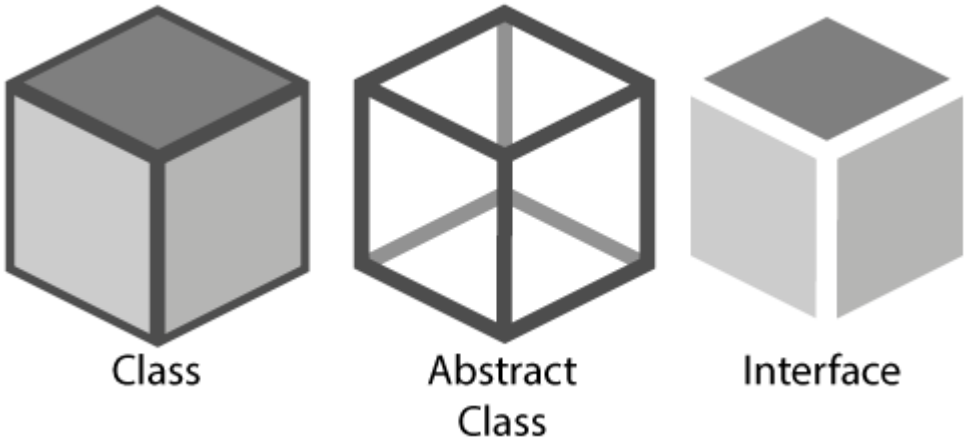
Interfaces define identity and behavior only.

An interface is similar to a class

- An interface is a user-defined data type with encapsulation and inheritance.
- An interface has all the parts of a class except the private data control and the Call Parent Class Method node.
- Classes inherit methods from interfaces and then provide their own implementations.



An interface is similar to a class



B. Why would users want that?

What use is a thing that looks like a crippled class?

There is a problem with inheritance.

Cost()
Manufacturer()
Test()

Tool

Turn()

Rotating Tool

Pound()

Pounding Tool

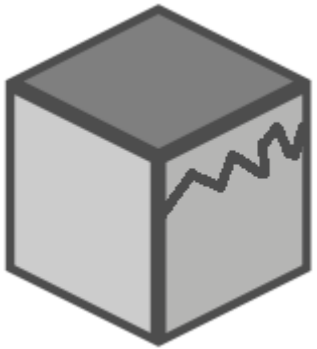
Pry()

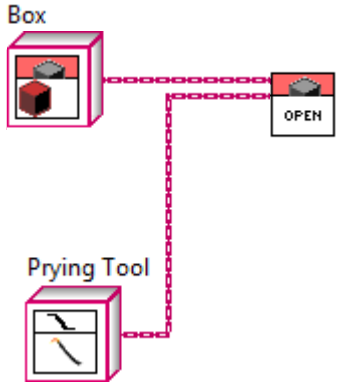
Prying Tool



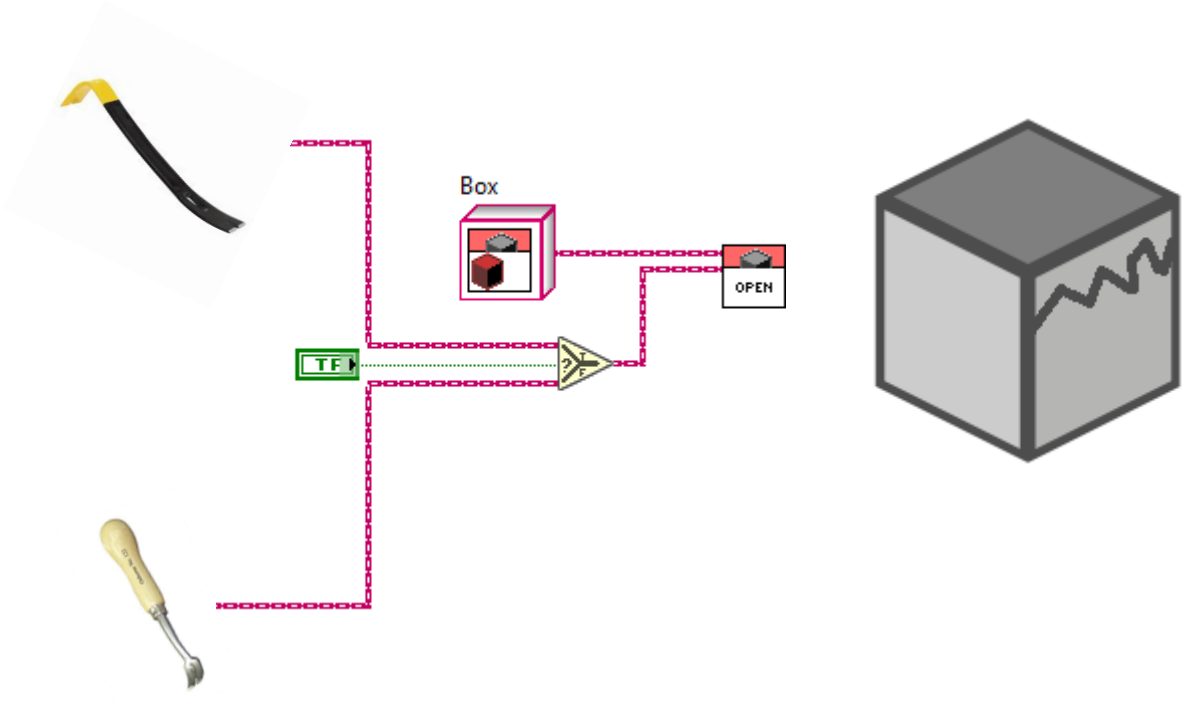



Phillips Flathead Nailer Mallet Prybar Claw





We can use either tool to pry open the box.



But tools have many uses.

Cost()
Manufacturer()
Test()

Tool

Turn()

Rotating Tool

Pound()

Pounding Tool

Pry()

Prying Tool

Phillips

Flathead

Pry()

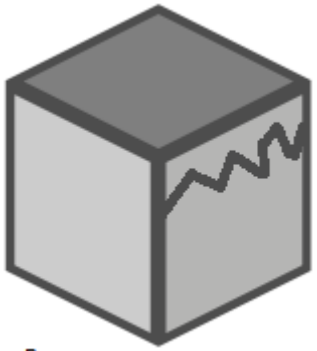
Nailer

Pry()

Mallet

Prybar

Claw



But tools have many uses.

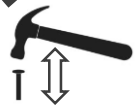
Cost()
Manufacturer()
Test()
~~Pry()~~

We all know this is a bad idea, right?

Tool

Turn()

Rotating Tool



Pound()

Pounding Tool

Pry()

Prying Tool

Phillips

Pry()

Flathead

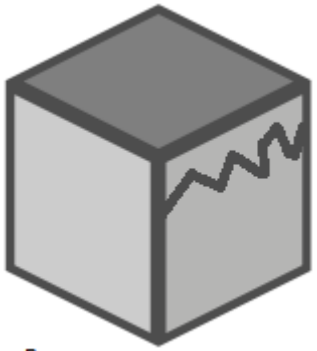
Pry()

Nailer

Mallet

Prybar

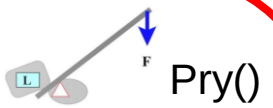
Claw



We can use an interface.

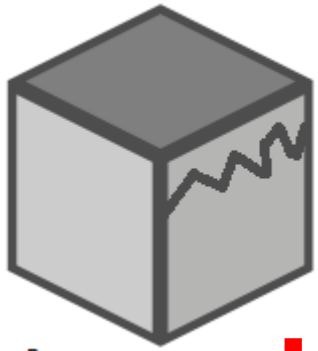
Cost()
Manufacturer()
Test()

Tool



Pry()

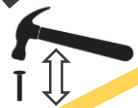
Lever



Box

Turn()

Rotating Tool



Pound()

Pounding Tool

Pry()

Prying Tool

Phillips

Pry()

Flathead

Pry()

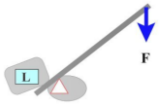
Nailer

Mallet

Prybar

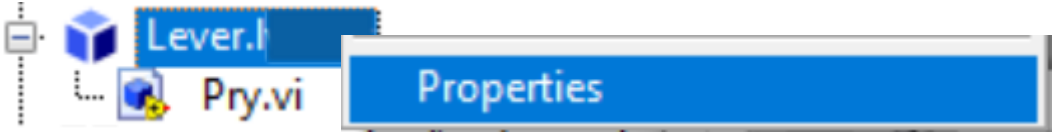
Claw

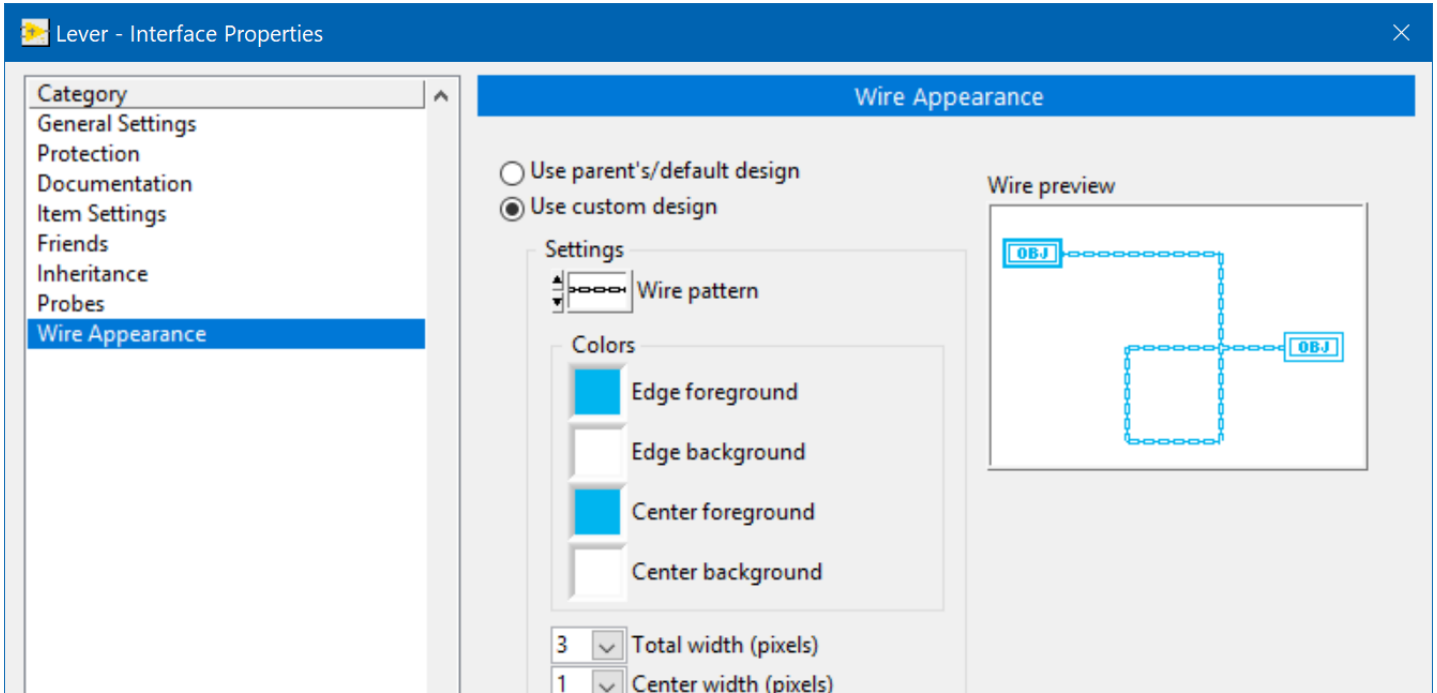
Change the wire appearance.



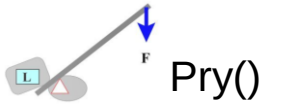
Pry()

Lever





We can use the interface with unrelated classes.



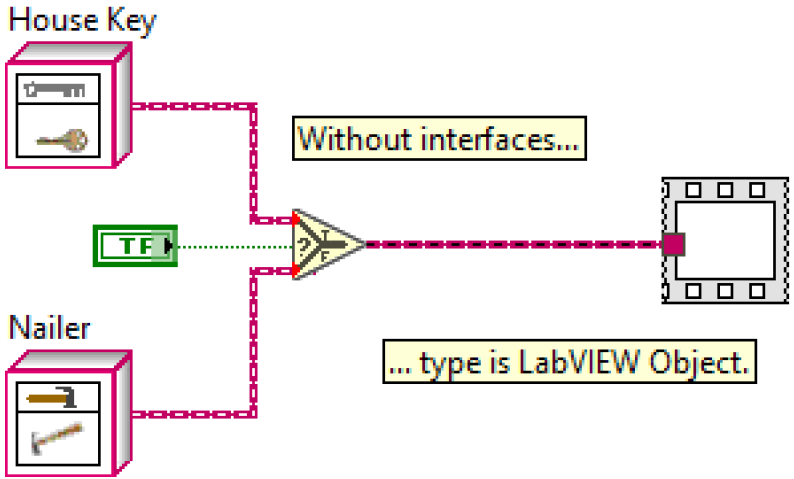
Pry()

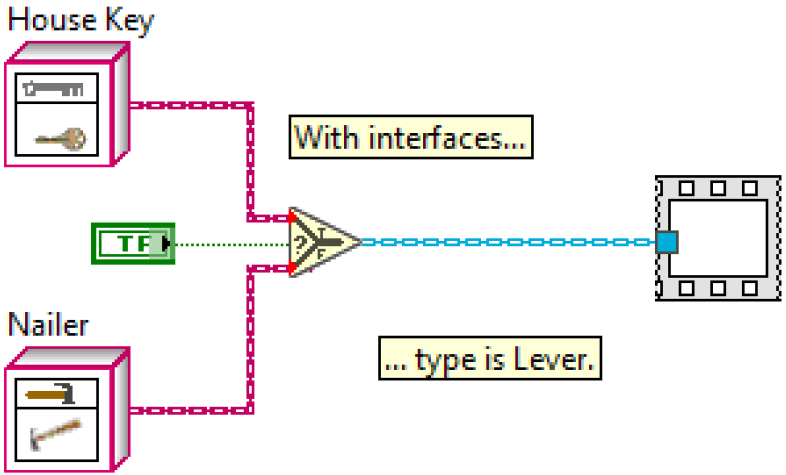
Lever

Pry()

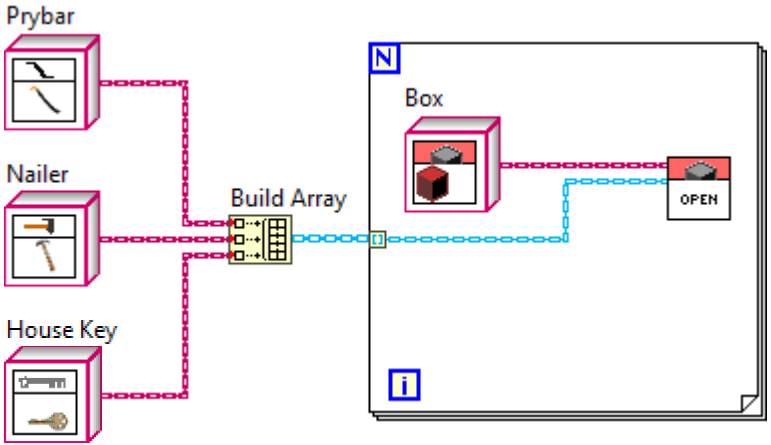
Key

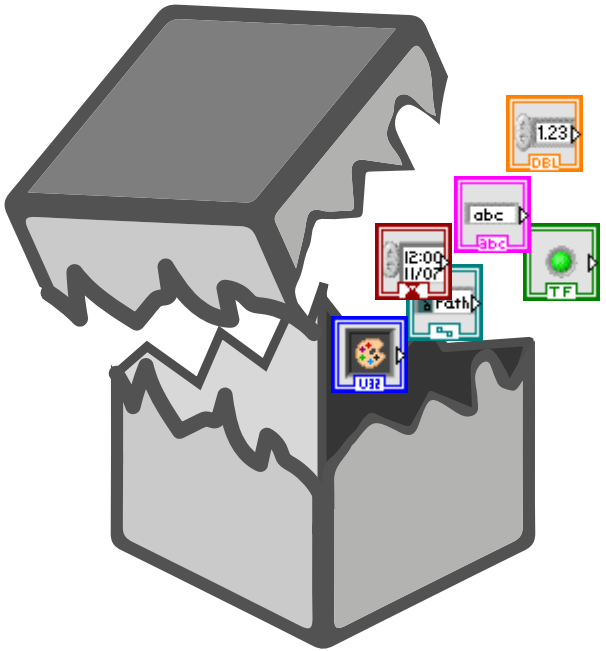
And now, this is possible...





And this...





1 *For the record, this is totally how generic probes work behind the scenes.*

(Just kidding.)

And that is why users want interfaces.

- Hardware\measurement abstraction layers
- Plug-in systems
- Loosely coupled messaging
- Mock testing

...and more.

Demonstration

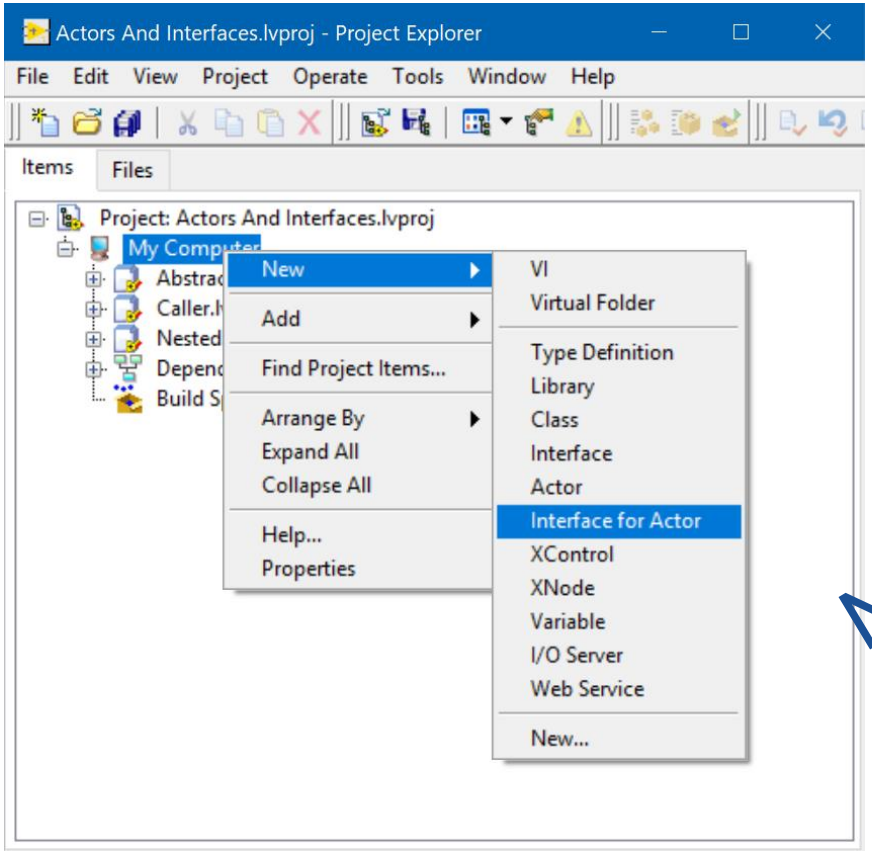
Interfaces in LabVIEW

And let’s talk about that file extension...

C. Interfaces & Actors

How interfaces enable decoupling of actors.

No more abstract message classes



Not enough time!!! But, trust me, there's good stuff here!

G Interfaces

Interfaces are essentially classes without data.

Interfaces enable a form of multiple inheritance.

Interfaces decouple modules and enable dependency inversion (“D” in SOLID).

Interfaces make nicer code in many use cases.

ni.com/techpreview