

# Getting started with Phidgets in LabVIEW

## Environment and Libraries

First, we need to set up the proper environment and get the necessary files off the Phidgets website. Visit the drivers section at [www.phidgets.com](http://www.phidgets.com) and get the latest:

- Phidget Framework

You will need files installed with the Phidget Framework to program or use Phidgets. We also recommend that you download the following reference materials from the programming section:

- Examples in LabVIEW
- COM API Manual
- Programming Manual
- The Product Manual for your device

The COM API manual contains calls and events for every type of Phidget and can be used as a reference. You can find a high level discussion about programming with Phidgets in general in the Programming Manual. The Product manual for your device also contains an API section that describes limitations, defaults, and implementation details specific to your Phidget. You may want to have the manuals and examples open while working through these instructions.

## Setting up a Phidgets Project

The Phidget examples were written using LabVIEW 8.2 and this tutorial assumes its use. Other versions such as LabVIEW 7 are supported and would be set up in a similar manner. This tutorial will be using ActiveX objects to work with a PhidgetInterfaceKit.

To get started, in the front panel add an Automation Refnum (from Controls palette | Modern | Refnum) to store a reference to a Phidget ActiveX object. Right click the Automation Refnum and choose the "Select ActiveX Class option", then browse to the installation directory for the Phidget Framework. Open the Phidget21COM.dll, and then check the box labeled "Show Creatable Objects Only". Next, choose the IPhidgetInterfaceKit from the list. The project is now ready to work with the Phidget.



Note that if a non-creatable object of the same name is chosen by accident, you may have to delete and replace the Automation Refnum, (or set it to some other target before changing it to the creatable object) to ensure LabVIEW does not use its cached target.

## Coding For Your Phidget

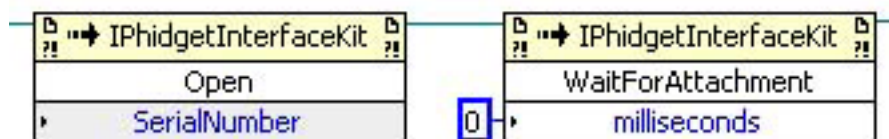
Methods belonging to the Phidget can be called on the block diagram using appropriately wired Invoke Nodes. Some values can also be accessed directly using Property Nodes. These, along with other useful ActiveX functions, can be found under Functions Palette | Connectivity | ActiveX sub-palette, and wired to the IPhidgetPhidgetInterfaceKit automation refnum.



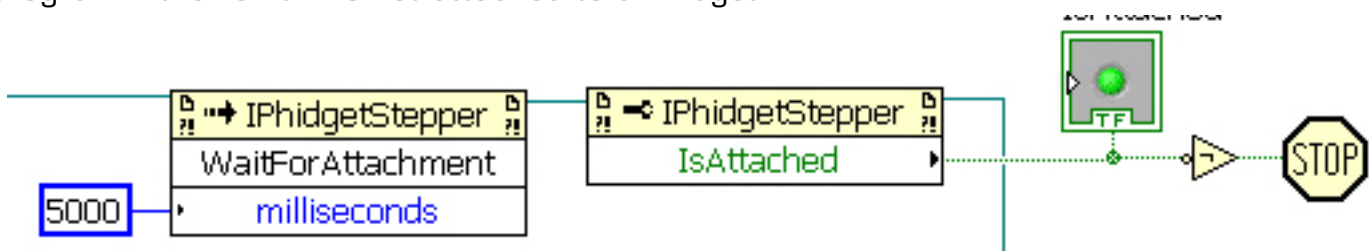
Initially a new instance of the IPhidgetInterfaceKit class needs to be created through an Automation Open. Wire the IPhidgetInterfaceKit automation refnum to the Automation Open object, and set the open new instance to true. The program flow can be continued by wiring with the Automation Refnum.

## Connecting to the Phidget

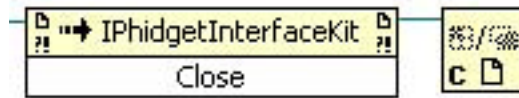
Next, we need to tell the program to try and connect to the Phidget through an open call. This Phidget open call is different from the Automation Open. It tells the program to continuously try to connect to a Phidget, based on the parameters given, even trying to reconnect if it gets disconnected. This means that simply calling open does not guarantee you can use the Phidget immediately. We can handle this by using event driven programming and tracking the AttachEvents and DetachEvents, or by calling waitForAttachment. WaitForAttachment will block indefinitely until a connection is made to the Phidget, or an optional timeout is exceeded.



Here the open method is called with an Invoke Node after the Automation Open. Use waitForattachment immediately afterwards to ensure the Phidget is attached before other methods are invoked. You can set the wait for attachment time to 0 for an indefinite wait. If you are using a value other than 0 for your timeout (for example, 5000 for a 5 second timeout), once that timeout has been reached, WaitForAttachment will continue even if it has not gotten an attachment. A good idea is to check IsAttached afterwards, and to stop the program if the refnum is not attached to a Phidget.



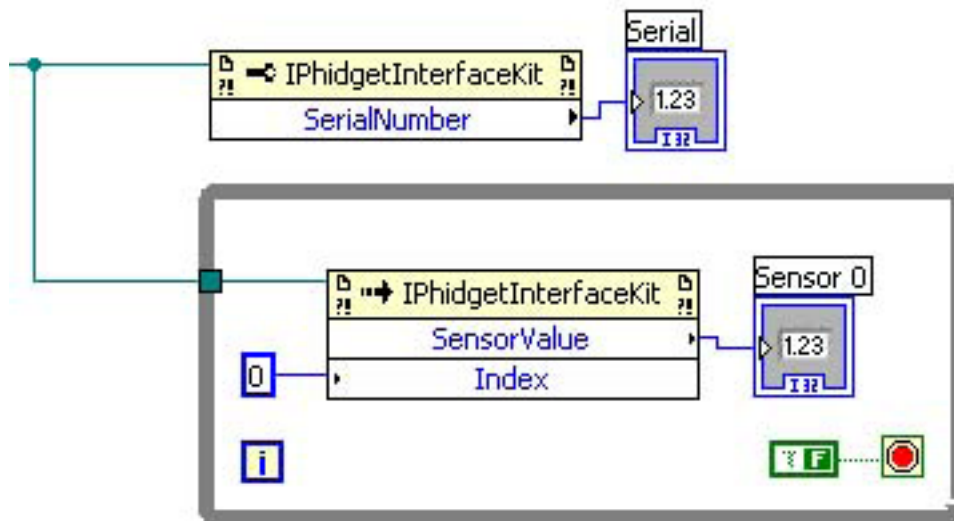
One important thing to remember is that when working with Phidgets, a local connection will reserve the device until closed. This prevents any other instances from retrieving data from the Phidget, including other programs. The one connection per device limit does not apply when exclusively using the Phidget Webservice. At the end of your program, don't forget to invoke close and then close the instance at the end of the program to free any locks on the Phidget.



## Working directly with Phidgets

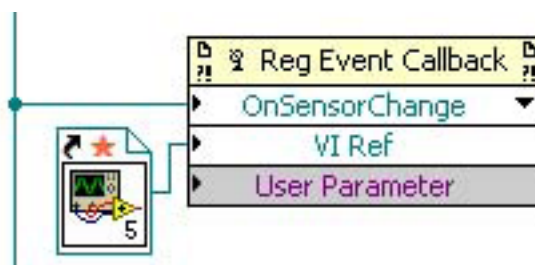
Two approaches for retrieving data can be taken when working with Phidgets. You can continuously poll the device or handle its events when triggered. This section will cover how to read directly from and poll the device while the next section will detail the event handling.

Getting or setting values directly from the Phidget can be done by invoking the Phidget object members such as OutputState(put), or using a property node. By putting these nodes inside a loop structure, polling the Phidget is straight forward task.

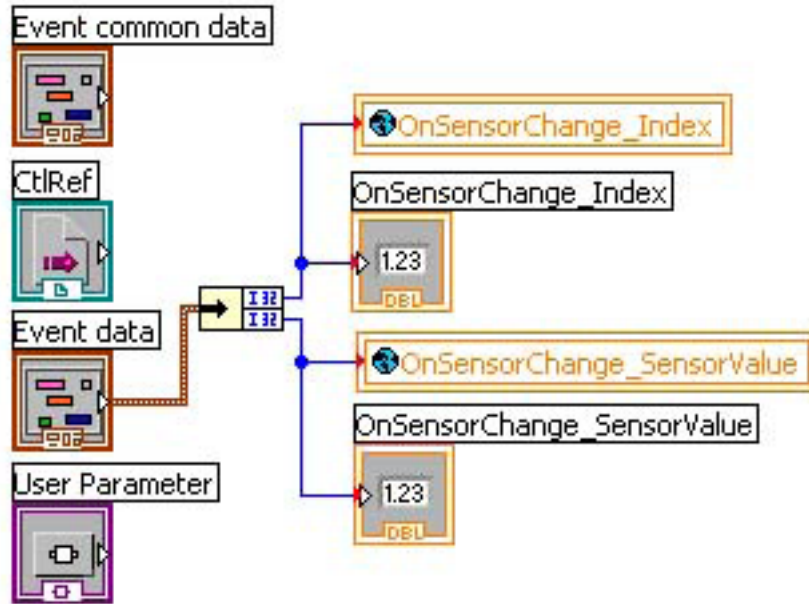


## Event Driven Programming

Phidgets also supports event driven programming in LabVIEW. We hook an event handler with Register Event Callbacks in the following setup:



By right clicking “VI Ref” on the Register Event Callback function, “Create Callback VI” can be used to create a VI with the all necessary parameters already inserted.



With the above OnSensorChange.vi event handler, the code will get executed every time the InterfaceKit reports a change on one of its analog inputs. The values from the report can be accessed by unbundling the Event Data, and then fed back into the main program with global variables.

Some events such as Attach and Detach belong to the base Phidget object and thus are common to all types of Phidgets. Please refer to the API manual for a full list of events and their usage.

## Working with multiple Phidgets

Multiple Phidgets of the same type can easily be run inside the same program. It only requires another Automation Open to be placed and wired to the IPhidgetInterfaceKit refnum. The new Automation Open can then be initialized, opened and used in the same fashion as before.

If the application needs to distinguish between the devices, open can be called with the serial number of a specific Phidget.

## Other Phidgets

The design given in this document can also be followed for almost all Phidgets. For example, if you were using a PhidgetRFID instead of an Interfacekit, you would place an IPhidgetRFID object instead of an IPhidgetInterfaceKit. The methods and events available would change but could be accessed in a similar manner.