

# Getting started with Phidgets in Max/MSP

## Environment and Libraries

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

- Phidget 21 Installer
- Max/MSP Code Samples - Mac/Windows

You will need the externals found in the code samples to use Phidgets with Max/MSP. You will also need to install the latest Phidget21 libraries for your Platform. The .help files included in the Max/MSP examples are working examples, which show all the supported function calls for that Phidget in Max. We also recommend that you download the following reference materials from the programming section:

- Programming Manual
- The Product Manual for your device

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 libraries were written to support Max/MSP version 4.5 and greater. This tutorial assumes the use of Max 5 but other versions can be set up and used in the same way.

To install the Phidgets to the Max environment, just add the Phidget Max/MSP examples folder to the Max search path. This can be done in Options | File Preferences, by adding the Phidget Max/MSP examples directory to the list. The project is now ready to use Phidgets and you can begin by creating a new Patcher (File | New Patcher).

## Coding For Your Phidget

Phidgets can be placed inside the patcher using Max objects, and functions can be called on them using appropriately connected messages. The Max/MSP libraries may not implement the full Phidget API - some function calls and Phidget classes may not be supported. The help files included with the Phidget externals show all the supported function calls for their type of Phidget.

Data is accessed either by polling, or at a fixed rate via internal timers. The Phidget webservice is supported.

This tutorial uses a PhidgetInterfaceKit and a new instance will be created. It can be done by



PhidgetInterfaceKit

placing a new “object” object, and naming it PhidgetInterfaceKit.

## Connecting to the 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 using the Phidget Webservice. Be aware that every Phidget object in Max will automatically try to connect to and reserve the Phidget for itself. It will also continuously try to connect to a Phidget, even trying to reconnect if it gets disconnected.

When the instance is created, normally it will make a connection to the first device of its type it can find. The Phidget object can also be declared with a serial number to open a specific Phidget instead:

```
PhidgetInterfaceKit 37560
```

We can also open remote Phidgets over the Phidget Webservice, for example:

Open a remote interface kit using ServerID:

```
PhidgetInterfaceKit remote "Some remote serverID"
```

Open a remote interface kit using IP address and port:

```
PhidgetInterfaceKit remoteip 192.168.2.5 5001
```

Open a remote interface kit with serial number 35569 on serverID "Patrick's PC", with password "pass":

```
PhidgetInterfaceKit 35569 remote "Patrick's PC" pass
```

Open a remote interface kit with serial number 35569, on any remote server

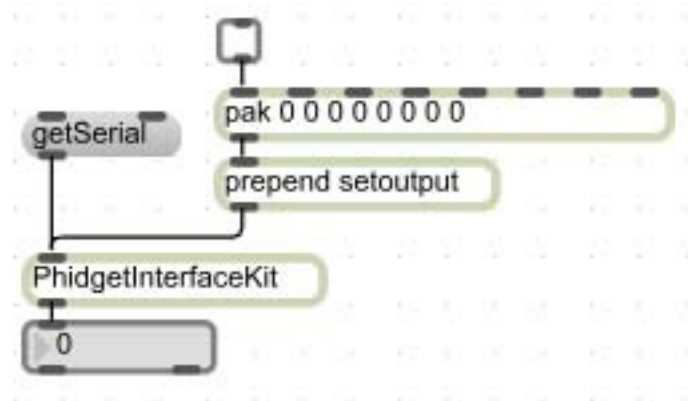
```
PhidgetInterfaceKit 35569 remote
```

Open the InterfaceKit on a PhidgetSBC:

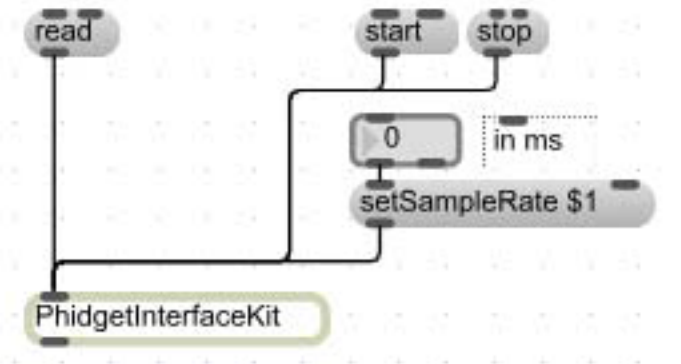
```
PhidgetInterfaceKit remote phidgetsbc
```

## Reading and Setting Data

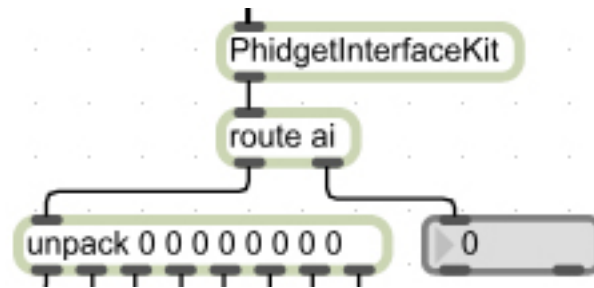
Getting or setting values directly on the Phidget can be done through messages linked to the inlet. The object's inlet can be wired to send commands to the device, and the outlet used to retrieve the results. Setting values on the Phidget is achieved by using the set messages, and some properties can be read with get messages.



There are two main approaches for retrieving data when working with Phidgets. One is to use the read message, and the other is to setSampleRate and use the start and stop message. Read will read the data off the Phidget once, while using start will trigger data to be sent in periodic intervals. If the sample rate is set to -1, then data output is only triggered on a change.



Phidget specific data is always given a prefix in Max to allow for their routing. For instance, the digital input states are given the prefix “di” and the analog inputs similarly use “ai”. The specific prefixes used for each Phidget is listed in their respective help file. Data common to all Phidgets, such as the serial number, is not prefixed. The following picture would be an example of how to route and split some of the data for the PhidgetInterfaceKit.



## Working with multiple Phidgets

Multiple Phidgets of the same type can easily be used inside a single program, it only requires another Phidget object placed. If two of the same type of Phidget object are placed, the serial number argument should always be specified to ensure that the correct Phidget gets associated with the correct object.

## 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 a PhidgetRFID object instead of a PhidgetInterfaceKit. The messages and data available would change but could be used in a similar manner.