

Getting started with Phidgets in AppleScript

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 and get the latest:

- Phidget Framework

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

- Programming Manual
- The Product Manual for your device
- Example Programs written in AppleScript

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 these manuals open while working through these instructions.

For a list of supported classes and commands, please refer to the PhidgetsOSA dictionary. In the AppleScript editor, this dictionary can be viewed by selecting PhidgetsOSA from File->Open Dictionary.

Setting up a Phidgets Project

The Phidget examples were written using AppleScript 2.1.2 under AppleScript Editor 2.3, and this tutorial assumes their use. Other versions and development environments should work as well and would be set up in a similar manner.

- First, open the AppleScript editor to create a new script.
- Coding with Phidgets is made possible by the interaction of Apple Events between AppleScript and the PhidgetsOSA application. Type the following to interact with PhidgetsOSA:

```
tell application "PhidgetsOSA"  
end tell
```

The project now has access to Phidgets and we are ready to begin coding. For the rest of this document, unless otherwise stated, it will be assumed that all code will be typed inside this tell block

Coding For Your Phidget

A Phidget object will need to be declared. For example, we can declare a PhidgetInterfaceKit:

```
set ifkit to make new phidget interfacekit
```

The object name for any type of Phidget is listed in the PhidgetsOSA dictionary. Every type of Phidget also inherits functionality from the Phidget base class.

Connecting to the Phidget

Next, the Phidget object needs to be initialized and the program needs to try and connect to the Phidget through a call to `open`. `open` will tell 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 Attach Events and Detach Events, or by specifying the `wait` parameter. The `wait` parameter will block for a certain amount of time until a connection is made to the Phidget. For example, we can connect to a `PhidgetInterfaceKit` with:

```
tell ifkit to open
```

The different types of `open` can be used with parameters to try and get the first device it can find, `open` based on its serial number, or even `open` across the network. For more information on connecting across a network, please see the "Working with Phidget WebService" guide on the programming section at www.phidgets.com. The PhidgetsOSA dictionary lists all of the available modes that `open` provides. Examples of the usage of different types of `open` are listed below:

We can tell the `ifkit` to block for 5000ms until a connection has been made to the `PhidgetInterfaceKit` with:

```
tell ifkit to open wait 1000 --wait for 5000ms
```

We can also connect to a `PhidgetInterfaceKit` over the `WebService` with a serial number of 99999 on a server with an IP Address of 192.168.3.180 and a port that is opened on 5001 with:

```
tell ifkit to open serial number 99999 server address "192.168.3.180" server port 5001
```

It is also possible to connect using the server id. For example, we can connect to a `PhidgetInterfaceKit` on a password protected server with:

```
tell ifkit to open server id "TestMac" password "pw"
```

At the end of your script, don't forget to call `close` to free any locks on the Phidget.

```
tell ifkit to close  
delete ifkit
```

Phidgets can also be freed from the PhidgetsOSA menu bar. For more information, please see the "PhidgetsOSA Menu bar" section of this document.

Event Driven Programming

We recommend the use of event driven programming when working with Phidgets. We can hook an event handler at loading with the following code:

```
tell ifkit to make new interfacekit sensor change handler with properties {script  
file: thisScript}
```

And after the tell block at end of the script, the callback method is defined as follows:

```
using terms from application "PhidgetsOSA"
  on interfacekit sensor changed ind to val on ifkit
    log "Sensor Index: " & ind & ", Sensor Value: " & val
  end interfacekit sensor changed
end using terms from
```

With this function, the code inside the `interfacekit sensor changed` handler will get executed every time the PhidgetInterfaceKit reports a change on one of its analog inputs. 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 PhidgetsOSA dictionary and the AppleScript examples for a list of events and their usage.

Working directly with the Phidget

Some values can be read and sent directly to the Phidget. For example, sensor values from the PhidgetInterfaceKit can be read with:

```
log "The first sensor has a value of: " & first interfacekit sensor's value
```

These functions can be used inside a polling loop as an alternative to event driven programming.

Using the same Phidget in more than one Application

One important thing to remember is that when working with Phidgets, a call to open will reserve the device until closed. This prevents any other instances from retrieving data from the Phidget, including other programs. If your goal is to use the same Phidget among multiple AppleScripts and/or other applications, there are two approaches.

1. Implement logic in the script that will use the same Phidget object if the script detects that a Phidget of the same type has already been initialized. For example,

```
if first phidget interfacekit exists then
  set ifkit to the first phidget interfacekit
else
  set ifkit to make the phidget interfacekit
  open ifkit
end if
```

2. The one connection per device limit does not apply when exclusively using the open Phidget Webservice. For more information, please see the "Connecting to the Phidgets" section of this document.

Working with multiple Phidgets

Multiple Phidgets of the same type can easily be run inside the same program. In our case, it requires another InterfaceKit instance to be defined and initialized. The new instance can then be set up, opened and used in the same process as the previous one.

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 PhidgetInterfaceKit, you would declare an RFID object instead of an InterfaceKit. The methods and events available would change but they can be accessed in a similar manner.

PhidgetsOSA Menu Bar



The PhidgetsOSA menu bar appears whenever AppleScript accesses the PhidgetsOSA application. The menu bar is used to monitor the status of any accessed Phidgets. Additionally, it can be used to free the lock on a Phidget or quit the PhidgetsOSA application.

The menu bar can be disabled by clicking 'Disable Menu', after which it will not appear again, even after restarting PhidgetsOSA. To re-enable the menu bar, run the following command in Terminal:

```
sudo defaults write com.phidgets.PhidgetsOSA showmenu -bool YES
```