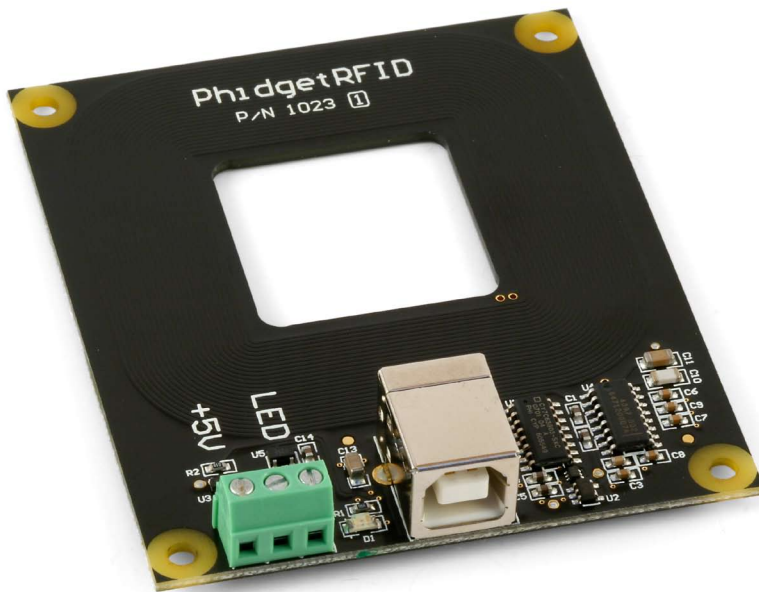


# Product Manual

## 1023 - PhidgetRFID



**Phidgets 9999 - Product Manual**  
**For Board Revision 1**  
**© Phidgets Inc. 2009**

# Contents

## 5 Product Features

- 5 Programming Environment
- 5 Connection

## 6 Getting Started

- 6 Checking the Contents
- 6 Connecting all the pieces
- 6 Testing Using Windows 2000/XP/Vista
  - 6 Downloading the Phidgets drivers
  - 6 Running Phidgets Sample Program
- 7 Testing Using Mac OS X
- 8 If you are using Linux
- 8 If you are using Windows Mobile/CE 5.0 or 6.0

## 9 Programming a Phidget

- 9 Architecture
- 9 Libraries
- 9 Programming Hints
- 9 Networking Phidgets
- 10 Documentation
  - 10 Programming Manual
  - 10 Getting Started Guides
  - 10 API Guides
- 10 Code Samples
- 10 API for the PhidgetRFID
  - 10 Functions
  - 11 Events

## 12 Technical Section

- 12 RFID
- 12 RFID Protocols
- 12 Communication and Effectiveness
- 12 Multiple Readers
- 12 Multiple Tags

- 13 Controlled Outputs
- 13 RFID Tags
- 14 Mechanical Drawing
- 14 Device Specifications

## **15 Product History**

## **15 Support**

# Product Features

---

- Reads tags brought within 3 inches of the reader
- Reads any tag with EM4102 protocol
- Returns the unique number contained in the tag
- Provides 2 Digital Outputs to drive LEDs, relays, etc.
- On-Board LED

## Programming Environment

**Operating Systems:** Windows 2000/XP/Vista/7, Windows CE, Linux, and Mac OS X

**Programming Languages (APIs):** VB6, VB.NET, C#.NET, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

**Examples:** Many example applications for all the operating systems and development environments above are available for download at [www.phidgets.com](http://www.phidgets.com) >> Programming.

## Connection

The board connects directly to a computer's USB port.

# Getting Started

---

## Checking the Contents

### You should have received:

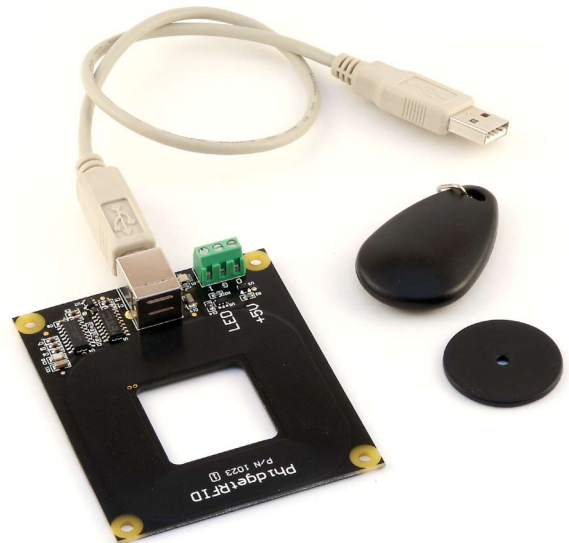
- A PhidgetRFID
- A USB Cable

### In order to test your new Phidget you will also need:

- A compatible RFID tag

## Connecting all the pieces

Connect the PhidgetRFID board to the computer using the USB cable.




## Testing Using Windows 2000/XP/Vista/7

### Downloading the Phidgets drivers

Make sure that you have the current version of the Phidget library installed on your PC. If you don't, do the following:

Go to [www.phidgets.com](http://www.phidgets.com) >> Drivers


Download and run Phidget21 Installer (32-bit, or 64-bit, depending on your PC)

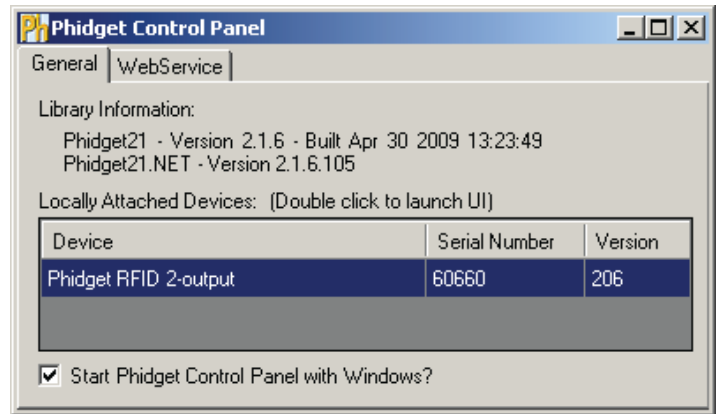
You should see the  icon on the right hand corner of the Task Bar.

### Running Phidgets Sample Program

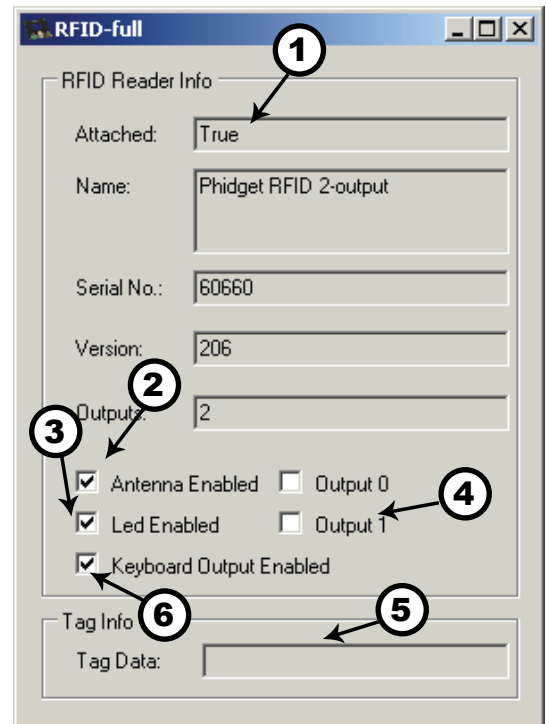
Double clicking on the  icon loads the Phidget Control Panel; we will use this program to make sure that your new Phidget works properly.

The source code for the PhidgetRFID-Full sample program can be found under C# by clicking on Phidget.com > Programming.

Double Click on the  icon to activate the Phidget Control Panel and make sure that **Phidget RFID** is properly attached to your PC.



1. Double Click on **Phidget RFID** in the Phidget Control Panel to bring up RFID-full and check that the box labelled Attached contains the word True.
2. Click on the Antenna Enabled box to enable the antenna.
3. Click on the LED Enabled box to turn the LED on or off.
4. The Output 0 box controls the +5V digital output and the Output 1 box controls the external LED digital output.
5. Bring an RFID tag close to the PhidgetRFID board and check that the identification string is displayed. Make sure that the Antenna is enabled for this step.
6. When the Keyboard Output is Enabled, the identification string will also be displayed at the cursor point in an application running in active window on the PC (such as a word document, for example).



## Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the **Phidget RFID** is properly attached.
- Double Click on **Phidget RFID** in the Phidget Preference Pane to bring up the RFID-full Example. This example will function in a similar way as the Windows version.

## If you are using Linux

There are no sample programs written for Linux.

Go to [www.phidgets.com](http://www.phidgets.com) >> Drivers

Download Linux Source

- Have a look at the readme file
- Build Phidget21

There is no Control Panel written for Linux, but there are C/C++ and Java code samples available for all Phidgets which will compile and run on Linux without modification.

Notes:

Many Linux systems are now built with unsupported third party drivers. It may be necessary to uninstall these drivers for our libraries to work properly.

Phidget21 for Linux is a user-space library. Applications typically have to be run as root, or udev/hotplug must be configured to give permissions when the Phidget is plugged in.

## If you are using Windows Mobile/CE 5.0 or 6.0

Go to [www.phidgets.com](http://www.phidgets.com) >> Drivers

Download x86, ARMV4I or MIPSII, depending on the platform you are using. Mini-itx and ICOP systems will be x86, and most mobile devices, including XScale based systems will run the ARMV4I.

The CE libraries are distributed in .CAB format. Windows Mobile/CE is able to directly install .CAB files.

The most popular languages are C/C++, .NET Compact Framework (VB.NET and C#). A desktop version of Visual Studio can usually be configured to target your Windows Mobile Platform, whether you are compiling to machine code or the .NET Compact Framework.



# Programming a Phidget

---

Phidgets' philosophy is that you do not have to be an electrical engineer in order to do projects that use devices like sensors, motors, motor controllers, and interface boards. All you need to know is how to program. We have developed a complete set of Application Programming Interfaces (API) that are supported for Windows, Mac OS X, and Linux. When it comes to languages, we support VB6, VB.NET, C#.NET, C, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

## Architecture

We have designed our libraries to give you the maximum amount of freedom. We do not impose our own programming model on you.

To achieve this goal we have implemented the libraries as a series of layers with the C API at the core surrounded by other language wrappers.

## Libraries

The lowest level library is the C API. The C API can be programmed against on Windows, CE, OS X and Linux. With the C API, C/C++, you can write cross-platform code. For systems with minimal resources (small computers), the C API may be the only choice.

The Java API is built into the C API Library. Java, by default is cross-platform - but your particular platform may not support it (CE).

The .NET API also relies on the C API. Our default .NET API is for .NET 2.0 Framework, but we also have .NET libraries for .NET 1.1 and .NET Compact Framework (CE).

The COM API relies on the C API. The COM API is programmed against when coding in VB6, VBScript, Excel (VBA), Delphi and Labview.

The ActionScript 3.0 Library relies on a communication link with a PhidgetWebService (see below). ActionScript 3.0 is used in Flex and Flash 9.

## Programming Hints

- Every Phidget has a unique serial number - this allows you to sort out which device is which at runtime. Unlike USB devices which model themselves as a COM port, you don't have to worry about where in the USB bus you plug your Phidget in. If you have more than one Phidget, even of the same type, their serial numbers enable you to sort them out at runtime.
- Each Phidget you have plugged in is controlled from your application using an object/handle specific to that phidget. This link between the Phidget and the software object is created when you call the .OPEN group of commands. This association will stay, even if the Phidget is disconnected/reattached, until .CLOSE is called.
- For full performance, the Phidget APIs are designed to be used in an event driven architecture. Applications that require receiving all the data streaming from the device will have to use event handlers, instead of polling.

## Networking Phidgets

The PhidgetWebService is an application written by Phidgets Inc. which acts as a network proxy on a computer. The PhidgetWebService will allow other computers on the network to communicate with the Phidgets connected to that computer. ALL of our APIs have the capability to communicate with Phidgets on another computer that has the PhidgetWebService running.

The PhidgetWebService also makes it possible to communicate with other applications that you wrote and that are connected to the PhidgetWebService, through the PhidgetDictionary object.

# Documentation

## Programming Manual

The Phidget Programming Manual documents the Phidgets software programming model in a language and device unspecific way, providing a general overview of the Phidgets API as a whole. You can find the manual at [www.phidgets.com](http://www.phidgets.com) >> Programming.

## Getting Started Guides

We have written Getting Started Guides for most of the languages that we support. If the manual exists for the language you want to use, this is the first manual you want to read. The Guides can be found at [www.phidgets.com](http://www.phidgets.com) >> Programming, and are listed under the appropriate language.

## API Guides

We maintain API references for COM (Windows), C (Windows/Mac OSX/Linux), Action Script, .Net and Java. These references document the API calls that are common to all Phidgets. These API References can be found under [www.phidgets.com](http://www.phidgets.com) >> Programming and are listed under the appropriate language. To look at the API calls for a specific Phidget, check its Product Manual.

## Code Samples

We have written sample programs to illustrate how the APIs are used.

Due to the large number of languages and devices we support, we cannot provide examples in every language for every Phidget. Some of the examples are very minimal, and other examples will have a full-featured GUI allowing all the functionality of the device to be explored. Most developers start by modifying existing examples until they have an understanding of the architecture.

Go to [www.phidgets.com](http://www.phidgets.com) >> Programming to see if there are code samples written for your device. Find the language you want to use and click on the magnifying glass besides "Code Sample". You will get a list of all the devices for which we wrote code samples in that language.

## API for the PhidgetRFID

We document API Calls specific to this product in this section. Functions common to all Phidgets and functions not applicable to this device are not covered here. This section is deliberately generic. For calling conventions under a specific language, refer to the associated API manual. For exact values, refer to the device specifications.

### Functions

#### **int OutputCount () [get] : Constant**

Returns the number of digital outputs available on this PhidgetRFID. These are the outputs provided by the terminal block.

#### **bool OutputState (int OutputIndex) [get,set]**

Sets/Returns the state of an output. True indicates activated, False deactivated. False is the default state.

#### **bool AntennaOn() [get,set]**

Sets/Returns the state of the antenna. True turns the antenna on, False turns it off. The antenna is by default turned off, and needs to be explicitly activated before tags can be read.

#### **bool LEDOn() [get,set]**

Sets/Returns the state of the onboard LED. True turns the LED on, False turns it off. The LED is by default turned off.

#### **string LastTag () [get]**

Returns the last tag read. This method will only return a valid tag after a tag has been seen. This method can be used even after a tag has been removed from range of the reader

#### **bool TagStatus() [get]**

Returns the state of whether or not a tag is being read by the reader. True indicates that a tag is on (or near) the reader.

## Events

### **OnOutputChange(int OutputIndex, bool State) [event]**

An event issued when an output has changed.

### **OnTag(string) [event]**

An event issued when a new tag is seen by the reader. The event is only fired one time for a new tag, so the tag has to be removed and then replaced before another OnTag event will fire.

Note: it is very important not to block in this event, or you will receive extra attach / detach events.

### **OnTagLost(string) [event]**

An event issued when a tag is removed from the reader.

# Technical Section

---

## RFID

RFID (radio frequency identification) systems use data strings stored inside RFID tags (or transponders) to uniquely identify people or objects when they are scanned by an RFID reader. These types of systems are found in many applications such as passport protection, animal identification, inventory control systems, and secure access control systems.



## RFID Protocols

In order for an RFID reader like the PhidgetRFID to communicate with an RFID tag, they must share a common protocol. This protocol acts as a set of rules for the way data is transmitted wirelessly between the reader and tag. The PhidgetRFID (as well as RFID tags sold by Phidgets) uses the EM4102 protocol. Any other tags that also use the EM4102 protocol can be used with the PhidgetRFID.

## Communication and Effectiveness

RFID tags come in two main varieties: passive and active. Active tags have their own power supply which they use to power an antenna to communicate and transmit data. Passive tags derive the power they require to operate directly from the RF output of the RFID reader, and no other power supply is necessary. This makes passive tags cheaper to produce and easier to implement.

Because passive tags require a strong RF field to operate, their effective range is limited to an area in close proximity to the RFID reader. In the case of the PhidgetRFID, tags brought within approximately 3" of the reader can be read. The distance over which the RFID tag is usable is affected by such things as the tag shape and size, materials being used in the area near the reader, and the orientation of the reader and tag in respect to each other and in their operating environment. The smaller a tag, the closer it must be to the reader to operate.



## Multiple Readers

Multiple PhidgetRFIDs within 1 to 2 meters may interfere with each other. This can be overcome in software by enabling the antennae of individual PhidgetRFID readers in sequence.

Starting with all readers disabled, enable the antenna of the first PhidgetRFID reader.

Wait for 100ms or more to detect any tags.

Disable the antenna of the first reader and enable the antenna of the second, and perform another wait cycle.

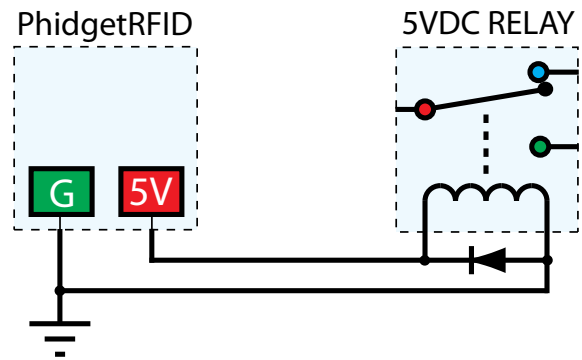
## Multiple Tags

The PhidgetRFID offers no capability for collision detection or collision avoidance. If two tags are brought within the read field of a PhidgetRFID reader at the same time, neither tag will be read. An RFID tag should be removed from the read field before a second tag is introduced.

## Controlled Outputs

The PhidgetRFID has four outputs - two of which are available to the user, and two of which are for internal control of the Phidget board only.

Output 0 is a +5V source from the USB bus through a P-Channel MOSFET with less than one ohm impedance. This can be used to switch a TTL or CMOS device, or it can be used to drive a 5VDC relay such as the Aromat JS1-5V. Output 1 is an LED drive output at 5VDC with maximum 15mA of available current (250 ohm CMOS output). Both Output 0 and 1 are available in hardware at the terminal blocks on the PhidgetRFID board. If Output 0 is used to drive a relay, a fast clamping diode must be placed across the relay drive pins as shown in the diagram on the right. Not doing so can result in permanent damage to the PhidgetRFID board.



Output	Function	Connection
0	+5VDC Source	Terminal Block
1	External LED Drive	Terminal Block
LED	Internal LED Drive	Internal Only
RF Enable	RF Antenna Enable	Internal Only

The PhidgetRFID comes equipped with an on-board LED that can be controlled using the LED property in software. Additionally, the on-board antenna can be enabled or disabled in software by using the RF Enable property. The antenna must be enabled for the PhidgetRFID to detect and read an RFID tag.

## RFID Tags

The PhidgetRFID can be used with any RFID tag designed for the EM4102 protocol. RFID tags come in a variety of shapes and sizes to suit various applications. All RFID tags sold by Phidgets are guaranteed to be unique, and are available as:

- 30mm Disc Tags (these can be sewn into garments, attached to objects)
- Credit Card Sized Tags (good for security identification applications)
- Key Fob Tags (attach easily to key rings)

## Device Specifications

Characteristic	Value
Antenna Output Power (max, far field)	< 10 $\mu$ W
Antenna Resonant Frequency	125kHz - 140kHz
Communication Protocol	EM4102
Read Update Rate	30 updates / second
External +5V Supply Voltage	5VDC
External +5V Supply Current Limit	400mA
External LED Supply Voltage	5VDC
External LED Supply Current Limit	16mA
External LED Output Resistance	250 Ohms
Recommended Terminal Wire Size	16 - 26 AWG
Terminal Wire Strip Length	5 - 6mm (0.196" - 0.236")
USB-Power Current Specification	500mA max
Device Quiescent Current Consumption	16mA
Device Active Current Consumption	100mA max
Typical Read Distance - Credit Card Tag	11cm (5")
Typical Read Distance - Disk Tag	6cm (3")
Typical Read Distance - Key Fob Tag	7cm (3.5")
Operating Temperature	0 - 70°C

## Product History

Date	Board Revision	Device Version	Comment
June 2002		100	Product Release.
April 2004		200	Onboard LED, 2 Digital Outputs added. Ability to enable/disable Antenna added.
Jan 2005		201	RF Circuitry upgraded to improve reliability.
Jan 2006		202	Onboard microprocessor upgraded to Flash version.
Jun 2006		204	Low Voltage Reset set at 4.7 Volts
April 2007	0	205	Protocol parsing bug fixed which garbled top 3 bits of RFID Tag.
July 2007	1	206	Unused internal I/O pulled high out of an abundance of caution, bus current characterized. Terminal block moved to edge of PCB, center of antenna routed out, digital output transients on startup eliminated.

## Support

Call the support desk at 1.403.282.7335 9:00 AM to 5:00 PM Mountain Time (US & Canada) - GMT-07:00  
or

E-mail us at: [support@phidgets.com](mailto:support@phidgets.com)