

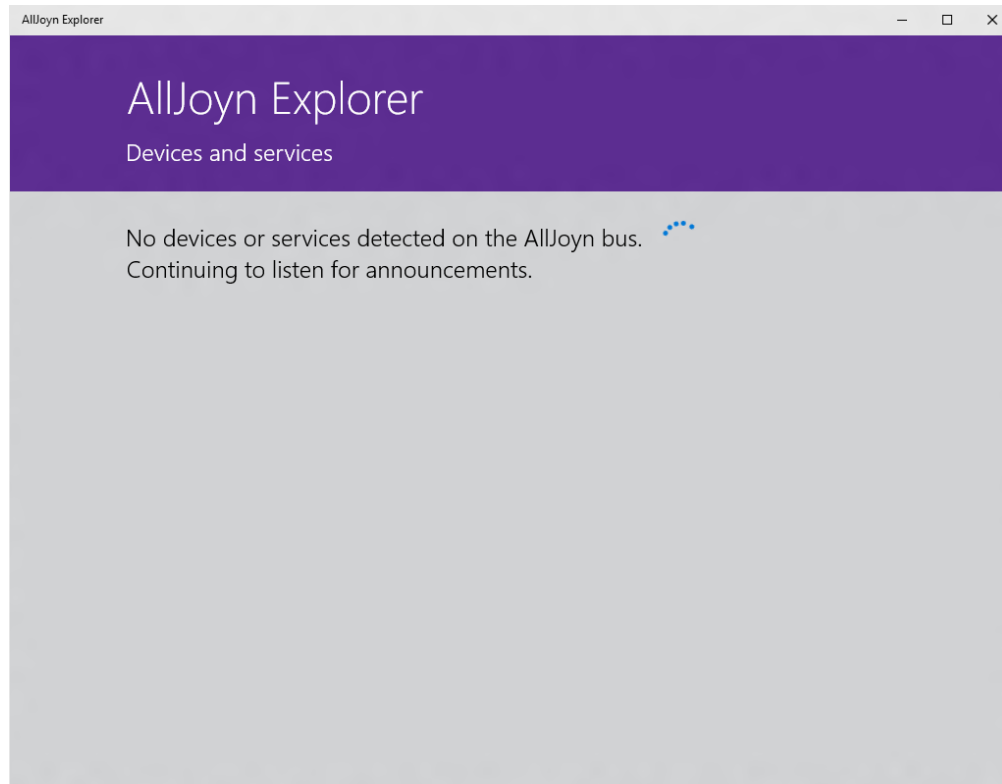
# AllJoyn Explorer

## User Guide

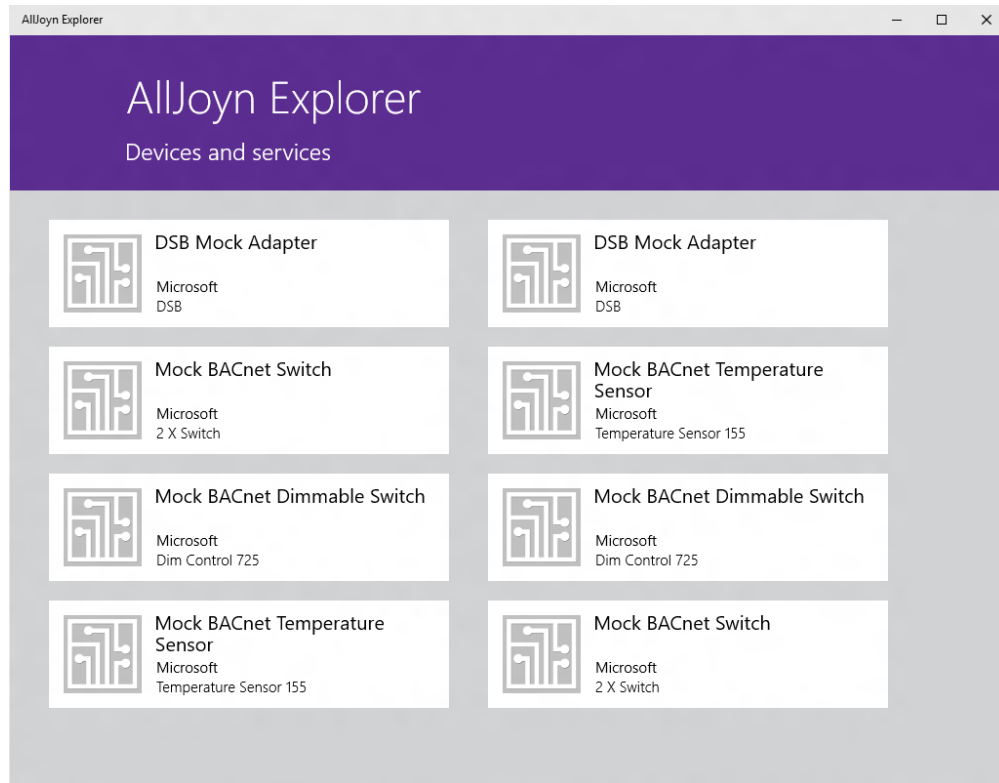
This document describes how the AllJoyn Explorer (AJX) application can be used to browse and interact with services exposed on the AllJoyn bus.

## Default View

When you first launch the application the AJX will search for available AllJoyn producers (devices) on the same subnet. You may see the following screen for a couple of seconds.



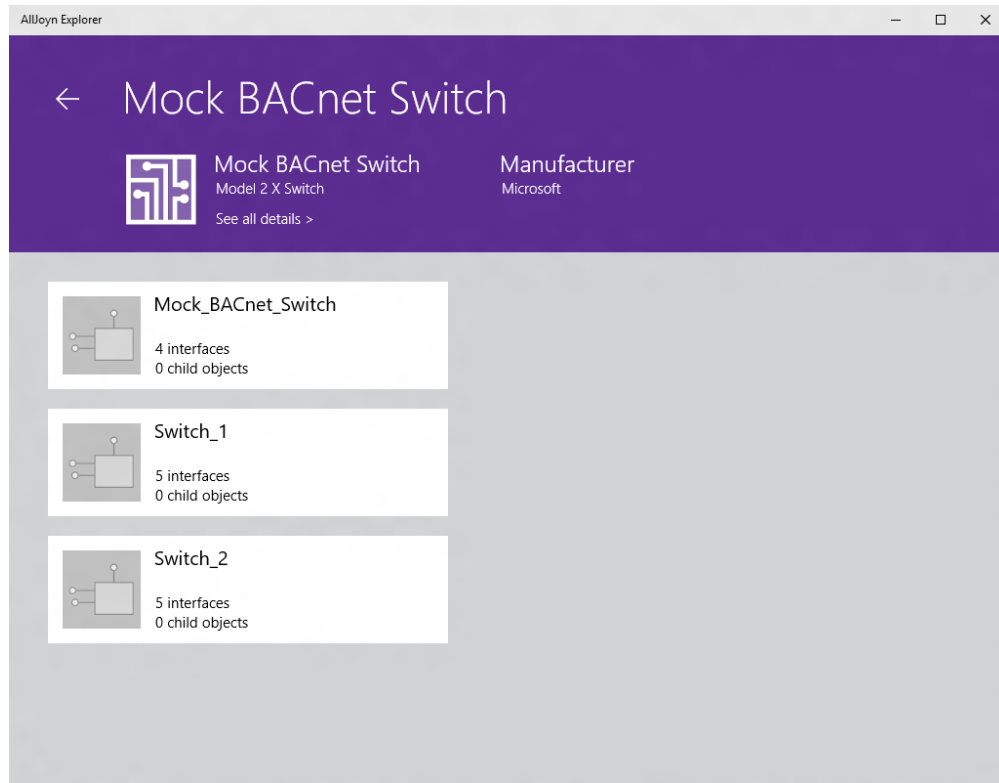
This screen indicates that AJX does not see any AllJoyn services that have broadcasted an “About” announcement. To see AllJoyn devices and services, start an AllJoyn producer application on the same subnet where you are running AJX. As an example, here is what you should see when running the Mock DSB sample alongside AJX:



AJX will dynamically listen for AllJoyn services to announce themselves and add them to this view. Likewise, if a service disappears, it will be removed from the starting view automatically.

## Service Details View

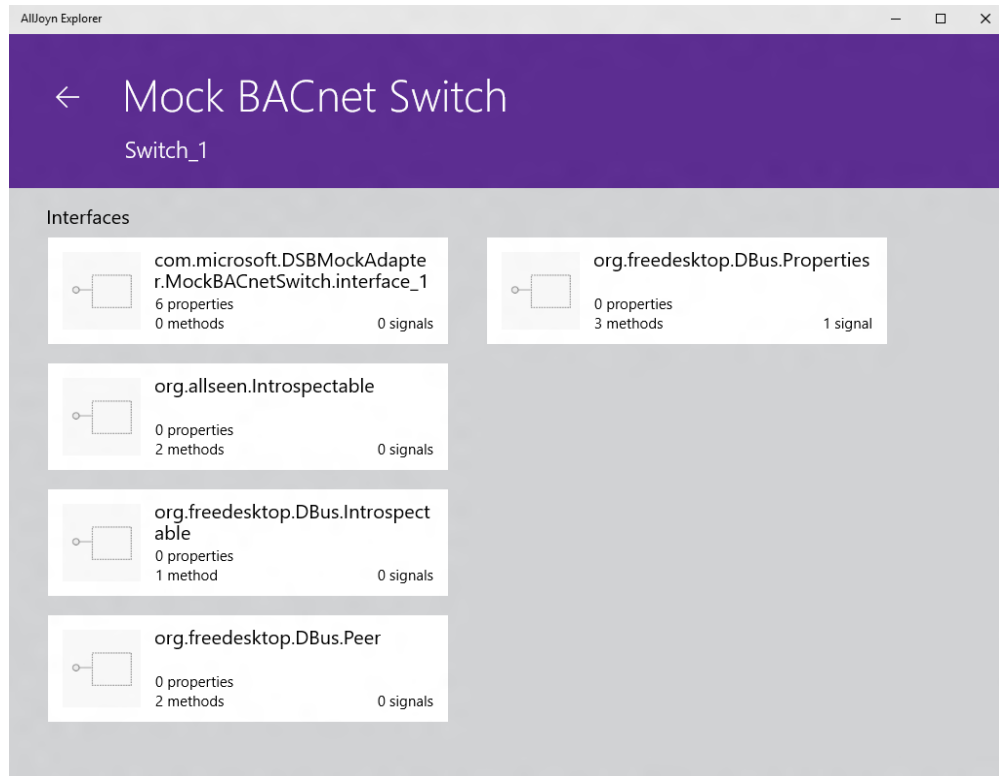
By clicking on any of the service tiles, AJX shows more detailed information about that service. For example, clicking on the service for the “Mock BACnet Dimmable Switch” above takes us to this view:



Here we see a more detailed view of the service, including the broadcasted “About” information, as all of the bus objects exposed by the service. Each tile shown on this page represents a single bus object that can be interacted with using AJAX.

## Bus Object View

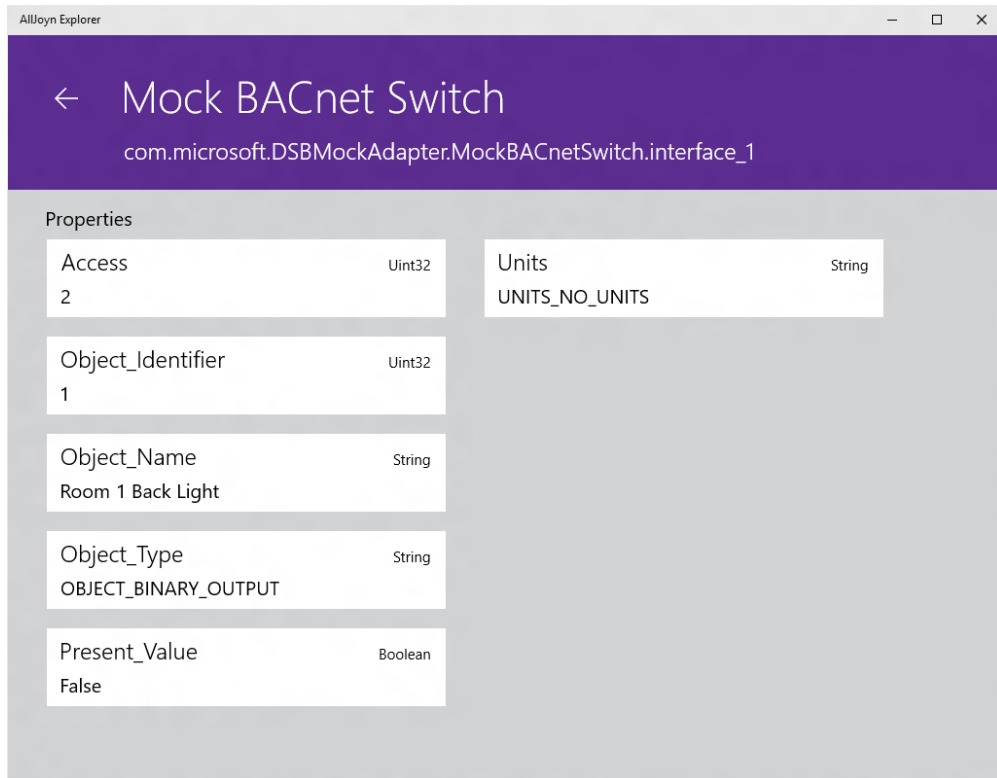
Just like above, we can get a more detailed information about a bus object by clicking on the corresponding tile. For example, clicking on the “Switch\_1” tile from the above screen takes us to the following page:



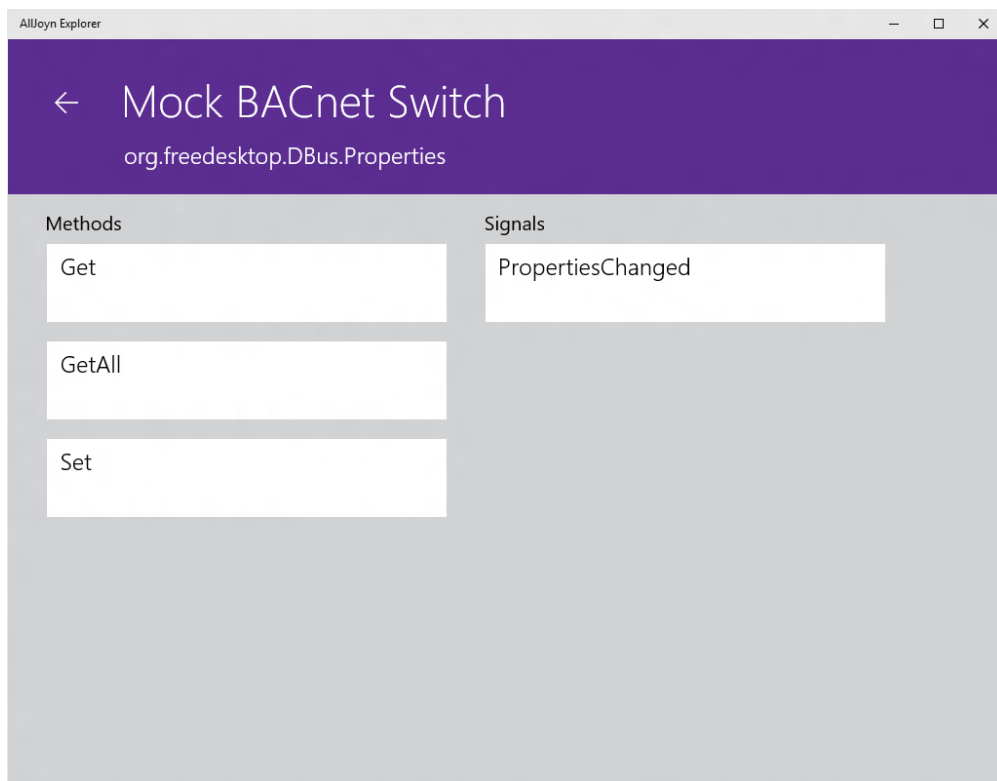
At the top of the page, we see the AllJoyn service name and bus object name, and at the bottom of the page we see tiles corresponding to each of the interfaces that this bus object exposes. Each interface can expose a number of properties, methods, and signals for this bus object.

## Interface View

Clicking on any of the interface tiles lets you see more information about the properties, methods, and signals that they expose. A couple of examples are below:



The interface shown above only exposes properties. The tiles show the property names, types, and current values. If those properties change, the view will be updated live.

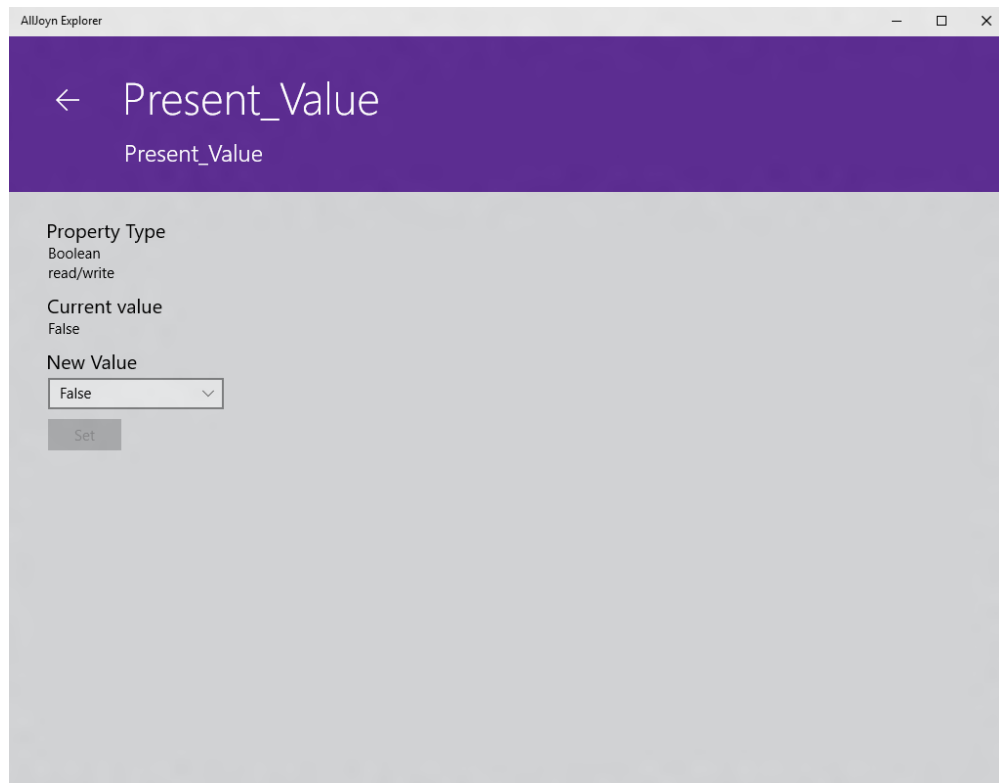


This interface exposes no properties, but it does expose three methods and a signal. The tiles show the names for each.

## Interacting with the Bus Objects

### Property Details Page

When you click on a property tile, you will be taken to a more detailed view of that property. You will be able to see the type, the current value of the property, and if the property is writable, you will be able to set a new value.



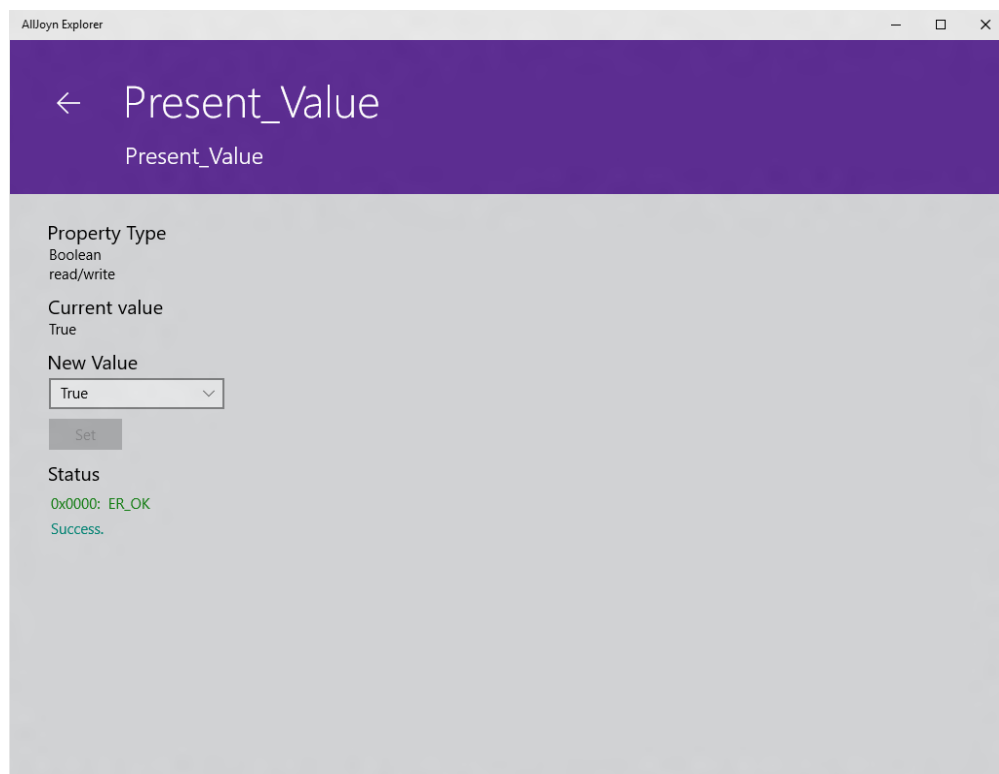
The screenshot shows a web application window titled "AllJoyn Explorer". The main header is purple with a back arrow and the text "Present\_Value". Below the header, the property details are displayed on a light gray background. The "Property Type" is "Boolean" and "read/write". The "Current value" is "False". The "New Value" is set to "False" in a dropdown menu. A "Set" button is visible below the dropdown.

Property Type  
Boolean  
read/write

Current value  
False

New Value  
False

Set



The screenshot shows the same web application window after the value has been updated. The "Current value" is now "True". The "New Value" dropdown is also set to "True". Below the "Set" button, a "Status" section shows "0x0000: ER\_OK" in green text, followed by "Success." in blue text.

Property Type  
Boolean  
read/write

Current value  
True

New Value  
True

Set

Status  
0x0000: ER\_OK  
Success.

When changing property values you will get some feedback indicating if your change was successful. If it is unsuccessful, there will be an error message that may indicate why.

### Method Details Page

When you click on a tile corresponding to a method, you are taken to a page with more details about that method. The page will indicate what input parameters the method takes, including their types and names (if available). It also gives information about the method's return parameters, included type(s) and name(s) when available. After entering the input parameters in the provided fields, you can invoke the method with the "Invoke" button.

The screenshot shows a web application window titled "AllJoyn Explorer". The main header is purple with a back arrow and the text "Get". Below the header, the page is divided into two main sections: "Method input arguments" and "Method output".

**Method input arguments**

- interface**  
String  
Input field: `com.microsoft.DS8MockAdapter.MockBACnetSwitch.interface_1`
- propname**  
String  
Input field: `Present_Value`
- Invoke** button

**Method output**

- <unnamed>**  
Variant

And after clicking “Invoke”:

The screenshot shows the AllJoyn Explorer interface for a 'Get' method. The title bar reads 'AllJoyn Explorer'. The main header is purple with a back arrow and the text 'Get' and 'Get'. Below this, the 'Method input arguments' section contains two input fields: 'interface' (String) with the value 'com.microsoft.DS8MockAdapter.MockBACnetSwitchInterface\_1' and 'propname' (String) with the value 'Present\_Value'. An 'Invoke' button is located below these fields. The 'Method output' section shows the result: '<unnamed>' (Variant) with the value 'True'. The 'Status' is '0x0000: ER\_OK' and 'Success'.

As with properties, if for any reason the method invocation is unsuccessful, you will receive an error message indicating why. For example, if I try to invoke this method but do not provide valid arguments (they are both left empty), this is the result:

The screenshot shows the AllJoyn Explorer interface for a 'Get' method. The title bar reads 'AllJoyn Explorer'. The main header is purple with a back arrow and the text 'Get' and 'Get'. Below this, the 'Method input arguments' section contains two empty input fields: 'interface' (String) and 'propname' (String). An 'Invoke' button is located below these fields. The 'Method output' section shows the result: '<unnamed>' (Variant) with the value 'True'. The 'Status' is '0x9003: ER\_BUS\_BAD\_VALUE\_TYPE' and 'Read an invalid value type'.



## Signal Details Page

When you click on a tile corresponding to a signal, you get more detailed information about that signal. You are shown details about the parameters that will be signaled, included their types and their names.

