Revia Assets Acq*Knowledge* for Unity

V1.0.5

Ref. RTD_ACQK_UM_20171004

Contact and support

For all support requests concerning Revia Assets Acq*Knowledge* for Unity, please visit http://acqku.support.reviatech.com

General support questions for Acq*Knowledge* or BIOPAC products should be addressed directly to BIOPAC support

Table of contents

Note: AcqKnowledge and NDT are products from BIOPAC Systems, Inc.

Introduction

Acq*Knowledge for Unity (AcqKU)* is a Unity3D package allowing Unity developers to access the Acq*Knowledge*'s network API (NDT) v5.0 from Unity scripts without having to deal with the details of network communication, threading and data access. AcqKU works on recent versions of Windows and MacOS.

First steps and install

In order to use *AcqKU*, you'll need some prerequistes:

- A computer with Window 8+ or MacOS X
- A Working install of Unity 3D Editor
- A working install of BIOPAC Acq*Knowledge* with NDT module
- The AcqKU deliverable (folder or ZIP file)
- A Licence Key (25 characters grouped by 5: XXXXX-XXXXX-XXXXX-XXXXX-XXXXX) that you'll need in order to activate the software on your computer.

It's also advised to have a hardware system with Input/outputs connected to Acq*Knowledge* if you wish to use this range of features.

Installation in Unity

To install the *AcqKU* in a Unity project

- Open a Unity Project or create a new one.
- Download or copy the folder corresponding to the AcqK deliverable and, if needed, unzip it
- Use **"Assets\Import Package\Custom Package…"** from the main Unity menu
- Choose the "**AcqKU_X_X.unitypackage"** from the AcqKU main folder
- Wait for the confirmation dialog box a few seconds, keep all checkboxes checked and click **"Import"**

Folder structure

You will find a new "**ReviaAssetsAcqKUnity**" folder in your project containing all *AcqKUnity* related scripts. It contains specific folders:

- **Prefabs** contains some basic prefabs you can integrate directly in your scenes
- **SampleScene** contains a scene with some basic scripts demonstrating the features
- **SampleScripts** contains the scripts used in the sample scene. You may reuse them or modify them for your own developments
- **Editor** and **Plugins** are folder containing the main libraries of AcqKU. Please do not rename or change the content of these folders

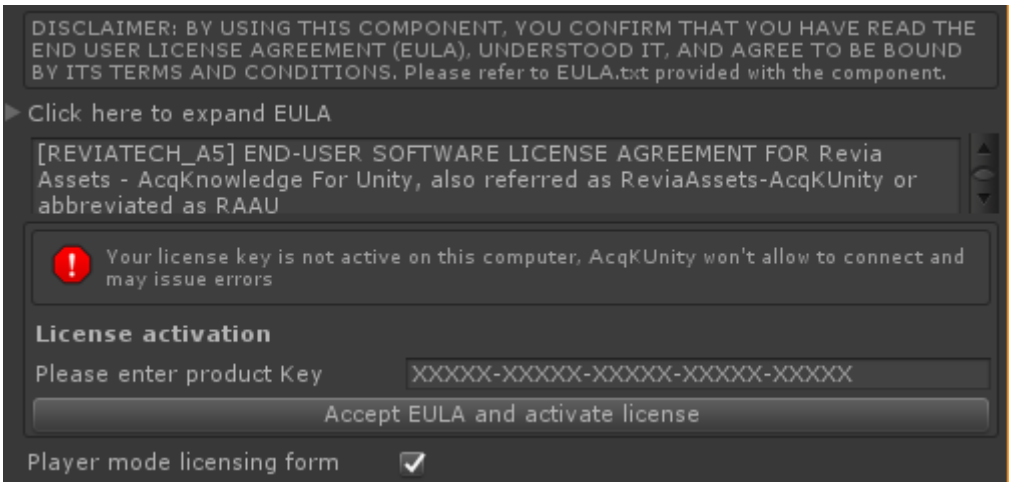
At the root, you'll find the **End User License Agreement (in the file EULA.txt)** that you must read and accept before using AcqKU.

Activating the license

In order to use AcqK on a computer, you need to activate your license. You may do it:

- From the "**AckUnity\License Tool**" menu
- From the **RAAU connector inspector panel**
- In player mode from the mini-form on the lower-right corner, if you are in the scope of a standalone published application that was made with Unity3D.

In each case, you have to enter your product key and click "activate". You'll need Internet connection at the time of the activation and some periodic connections to the Internet for updates.

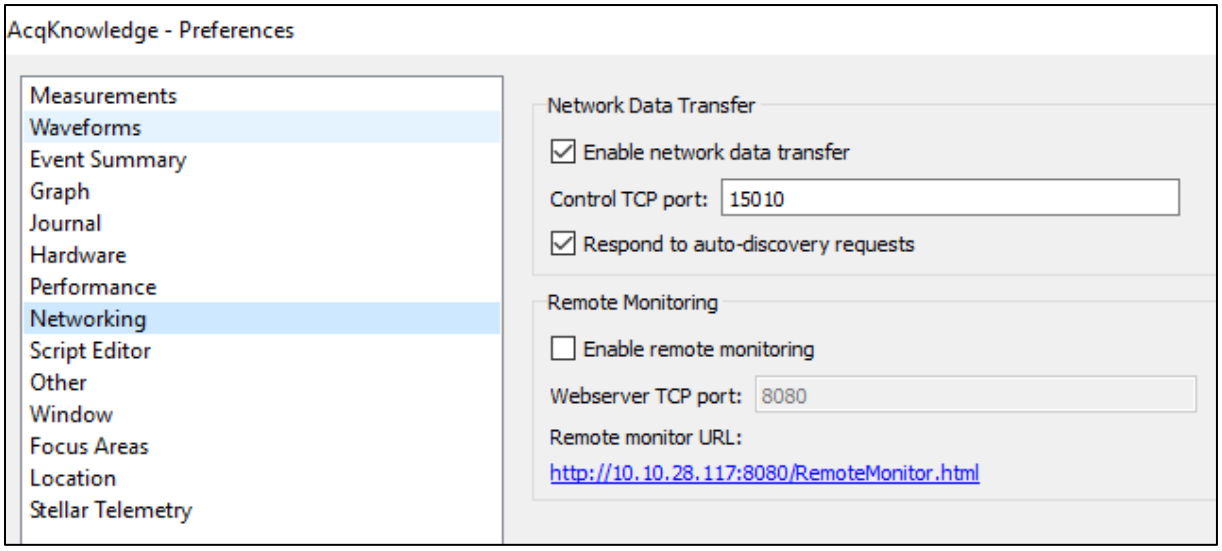


If you need to use your license on another computer, you can deactivate the software from the same panel and reuse the license key on the other computer.

In any case, you must keep you license key safe as it will be required to activate the software later if you require it.

Enabling NDT in Acq*Knowledge*

In Acq*Knowledge* preference panel, please activate Network Data Transfer as shown below:



Demonstration content and sample scripts

Preparation

Please activate your license from the "AckUnity\License Tool" menu and close the window.

We can now open the sample scene **SampleScene\AcqKUnity_SampleScene** and check the **Hierarchy:**

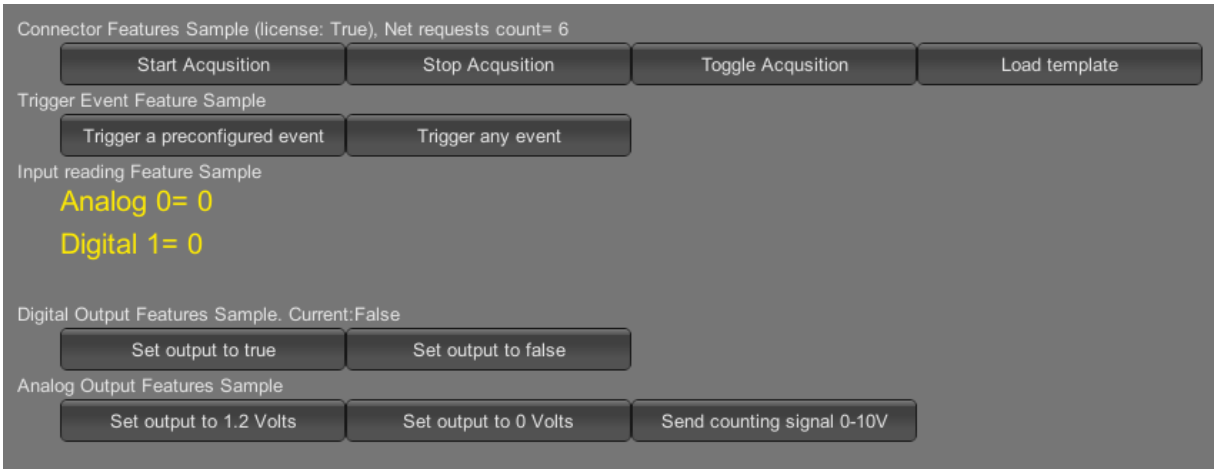
- Camera and Directional Light are default GameObjects from Unity
- **RAAU_Connector** contains the connector script, the main component creating the link with Acq*Knowledge*. You'll need one instance of this component in your scene.

- **RAAU_Components** are the AcqKU components used for the demonstration.
- **SampleScripts** are the user-scripts using the components

Each sample script demonstrates a scope of features:

- Connector features
- Trigger event Features
- Input reading features
- Ouptut driving features (analog and digital)

By starting the project with the **"Play"** button, you should see on your **Game** screen a set of sample buttons in order to help you on your first experimentations. Don't hesitate to look at the code in SampleScripts for inspiration.

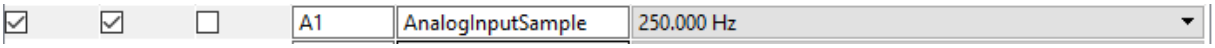


Configuring sample acquisition in Acq*Knowledge*

Please open Acq*Knowledge* and configure an empty project with some Input/Outputs

In the "Set up Data Acquisition…" panel:

- On first tab, setup analog channel A1 with checkboxes "acquire", "plot" and "value". You can also add a label and tune the update rate.



- On second tab, setup digital channel D1 with checkboxes "acquire", "plot" and "value". You can also add a label and tune the update rate.

In the samples, we will also use some outputs:

- Analog channel 0 (#1) as output
- Digital channel 2 as output

Type	Name	Set up in GUI	How to connect on hardware
Analog Input	A1	yes	Connect a jack plug on "Analog Input 1"
Digital Input	D1	yes	Connect a button with wires between screw terminals for "Digital I/O 1" and "GND"
Analog output	N/A	No	Connect a jack plug on "Analog Output 0" and connect a voltmeter to its wires
Digital output	N/A	no	Connect a voltmeter between screw terminals for "Digital I/O 2 " and "GND"

Note: Analog input A1 is labelled 1 on the acquisition hardware but referred as ID 0 in both Acq*Knowledge* and AcqKU. So Analog inputs A1-A16 corresponds to ids 0 to 15.

Testing samples

You can try the following actions:

- Start Acquisition / Stop acquisition / Toggle Acquisition to perform the action In Acq*Knowledge*. This can be useful if you want to relate a capture to some events in the 3D scene
- Trigger an event (preconfigured or "any"). This adds a marker on the Acq*Knowledge* timeline. The only difference in the two buttons is the way that the event was prepared (design-time or run-time)
- Display The current input of the configured channels
- Trigger a preconfigured digital output (0V or 5V)
- Trigger a preconfigured analog output (0V, 5V or a steppe curve)

Some other features, as the template loading or the device identification are not demonstrated here.

A note about network performance

IMPORTANT: AcqKU uses the NDT network API to drive Acq*Knowledge* remotely, with XML RPC network calls. According to the performance of the machine, the speed of the network and other parameters, we can expect to perform 10 to 100 calls per second on this API.

Features for Input Channels reads and Output Channels writes could specifically cause a lot of network traffic that have high performance repercussions. These components integrate a "polling frequency" or "limiting frequency" parameter to set a maximum rate of updates.

At any time, you can use "GetActiveNetworkRequestCount()" on the RAAU Connector to get the number of network requests currently opened to NDT. This count should be around under 10 in good performance situations. If the count get higher, or even goes up continuously, the network traffic is too high and should be limited with the frequency limiters parameters. If the count reach a maximum value (depending on the computer), network connections will overflow and fail and Unity and/or Acq*Knowledge* will have to be restarted.

Details about the RAAU Connector

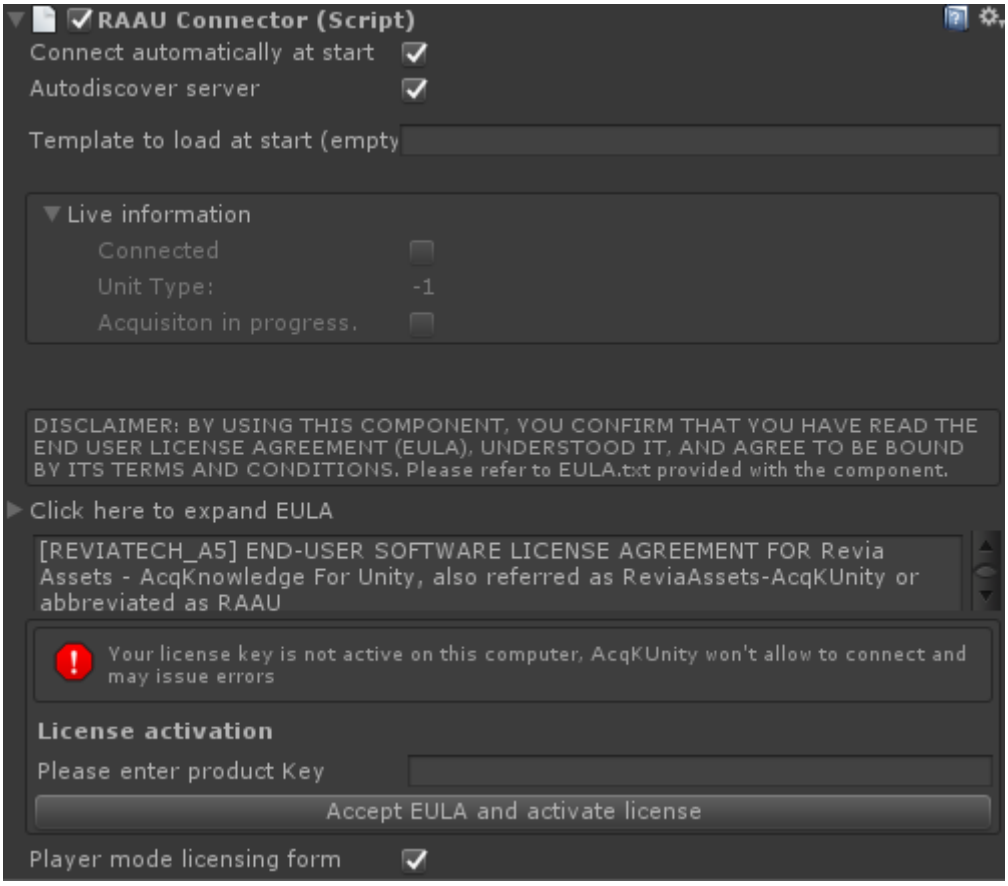
The RAAU_Connector is the main component of AcqKU and must be added to any scene using the other features.

Inspector panel

Following features are available:

- Connect automatically at start: if activated, the connection will be immediate in play mode. If not, you can use `ConnectToServer()` when required.
- Autodisconnect server: if activated, it will use the built-in discovery protocol to find the local Acq*Knowledge* server, if deactivated, you may enter the IP and port to use to connect to your own Acq*Knowledge* server.
- Template to load at start: allows to load a GTL template file from the beginning. Keep empty to not load any template. You may use `LoadTemplate(string template)` later to load a template at any time.

- The next fields are for information (read-only) during runtime:
  - Connected: is checked if connected to the server
  - Unit Type: type of device connected, 160 for MP160 for example
  - Acquisition in progress: is the acquisition currently active
- You'll find below the licensing panel presented in the "Activating the license" paragraph. Use the "Player mode licensing form" allows to automatically display a GUI in player mode to enter a licence key if required.



Runtime access and API Reference

You need to keep a reference to the component to use its features

bool	**autoconnectAtStart = false**
	configure the component to automatically connect at start
string	**templatetoLoadAtStart = ""**
	configure the component to automatically load a template file at start
bool	**autodiscovery = true**
	configure the component to load autodetect IP/port when connecting
string	**host = "127.0.0.1"**

	If autodiscovery is disabled, specifies the IP address to use for connection
int	**port = 15010**
	If autodiscovery is disabled, specifies the port address to use for connection
bool	**playerExtension = true**
	Activates the player GUI for license activation
bool	**IsAcquisitionInProgress**
	Returns if acquisition is currently in progress
bool	**Connected**
	Returns if the Acq*Knowledge* is currently connected
int	**UnitType**
	Returns the hardware unit type as a number
bool	**IsLicenseActive**
	Returns if the license is correclty activated

---

void	**ConnectToServer** ()
	Starts the connection to the Acq*Knowledge* server using autodetected or predefined IP and port

void	**DisconnectFromServer** ()
	Diconnects from server

void	**LoadTemplate** (string template)
	Loads the template file in Acq*Knowledge* **Parameters** **template** filename to load. Use full path preferably as default path can change according to your computer configuration

void	**ToggleAcquisition** ()
	Toggle acquisition state

int	**GetActiveNetworkRequestCount ()**
	Returns the number of requests currently active.
void	**SetAcquisitionActivation** (bool on)
	Sets acquisition state. Requests the current state and uses Toggle if required. **Parameters** **on** true to activate, false to deactivate

Sample script

Please refer to **Sample1_ConnectorFeatures.cs**

For example:

```
// Starting acquisition
connector.SetAcquisitionActivation(true);

// Toggling acquisition
connector.ToggleAcquisition();

// Loading template file
connector.LoadTemplate("SampleTemplate.gtl");

// Checking activated license and availability of the features)
bool isActive = connector.IsLicenseActive;
```

General notes

- You must use only one Connector component in your scenes
- It is advised to set the component as "Do Not Destroy OnLoad" if you plan to load scenes dynamically in Unity, this way, you won't have temporary disconnections when the scene is resetting
- If the connector is not yet connected or active, the other features will return errors. Please check the connection state before starting custom scripts
- Please make sure that the Template file name and path you specify in the "Template to load at start file" will be available when using a built version of your application. You can edit the path to make it local to your application if needed.

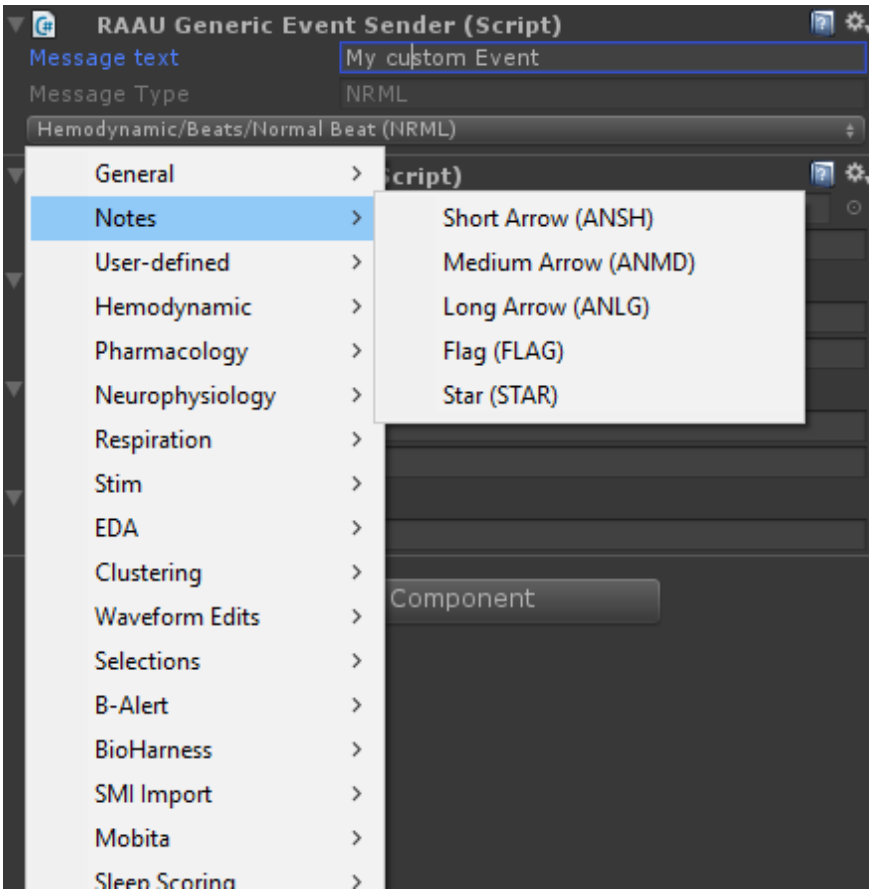
Details about the RAAU Generic Event Sender

This component allows to pre-configure an event that can be sent later by a script call. It also allows to send custom events with no pre-configuration.

Inspector panel

- Message text: the text for the predefined event
- Message type: the type for the predefined event

You can use the special dropdown to select a type by category and name:



Runtime access and API Reference

You need to keep a reference to the component to use its features

string	**message**
	Default message to use

AcqKGlobalEventType	**eventType**
	default event type to use

---

void	**InsertGlobalEventUsingDefaults** ()
	Insert a global event using the default values configured in the component

void	**InsertGlobalEvent (string message, string eventType)**
	Insert a global event using provided values
	Parameters

	message text of the message **eventType** type of event to add as string (letter code)
void	**InsertGlobalEvent** (string **message**, RAAcqK.AcqKGlobalEventType **eventType**)
	Insert a global event using provided values **Parameters** **message** text of the message **eventType** type of event to add as ENUM

Sample script

Please refer to **Sample2_TriggerEventFeatures.cs**

For example:

```
// Sending the preconfigured event
genericEventSender.InsertGlobalEventUsingDefaults();

// Sending a custom event (style 1)
genericEventSender.InsertGlobalEvent("Second event", RAAcqK.AcqKGlobalEventType.APC);

// Sending a custom event (style 2)
genericEventSender.InsertGlobalEvent("Second event", "APC");
```

General notes

- If you have only a few events in your application, it is advised to create a few preconfigured GenericEventSender components and trigger them as required. If your events have a dynamic part, using only one component and triggering custom events is prefered.

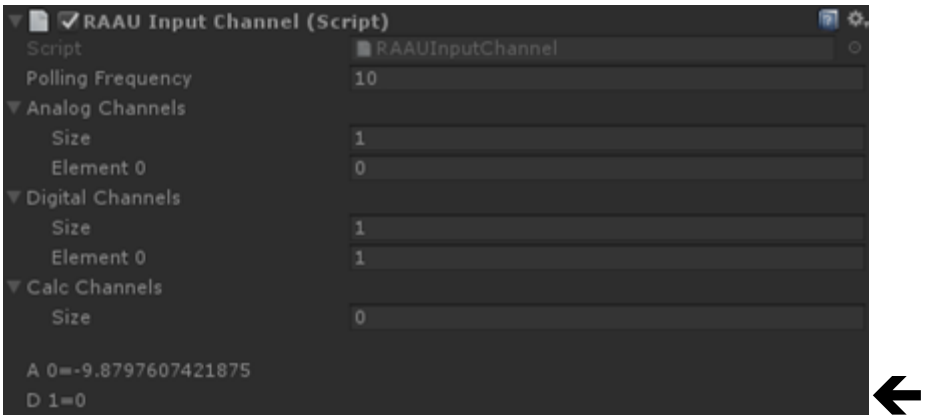
Details about the RAAU Input Channel

This component allows to configure the input channels in the inspector panel, in order to access the values on demand later. Values are automatically updated in the background.

Inspector panel

- Polling frequency: the frequency of input update. 20Hz corresponds to "twenty-times per second".
- 3x channel arrays: standard Unity array editor to add a list of the channel IDs you require access to.

In play mode, you can set the values from the inspector for testing:



Runtime access and API Reference

You need to keep a reference to the component to use its features.

uint	**PollingFrequency**= 15
	Polling frequency, in Hz, to get data updates
int []	**AnalogChannels** = { }
	Configured analog channels
int []	**DigitalChannels** = { }
	Configured digital channels
int []	**CalcChannels** = { }
	Configured calc channels

Dictionary< int, double >	**AnalogRawSamples** `[get]`
	Getter for analog raw samples
Dictionary< int, double >	**DigitalRawSamples** `[get]`
	Getter for digital raw samples
Dictionary< int, double >	**CalcRawSamples** `[get]`
	Getter for calc raw samples

Dictionary< int, double >	**AnalogScaledSamples** `[get]`
	Getter for analog scaled samples

Dictionary< int, double >	**DigitalScaledSamples** `[get]`
	Getter for digital scaled samples

Dictionary< int, double >	**CalcScaleSamples** `[get]`
	Getter for calc scaled samples

Note:

- Dictionary getters use the channel id as key to get the value. The channel id must be defined in the inspector to be available through the accessor
- All data are provided as double type for standardised interface, you can cast them as integers (0 or 1) when using the digital channel
- At runtime some live data will be displayed below the configuration in the inspector (reselect the object to refresh)

Sample script

Please refer to **Sample3_InputReading.cs**

For example:

```
//Reading analog value on channel 1 (-99 is used if no data is available)
analogValue = -99;
inputChannel.AnalogRawSamples.TryGetValue(1, out analogValue);
```

General notes

- It is advised to keep "polling frequency" to a sensible value according to your usage. This feature can overflow the network if the frequency is too high. On a recent computer with fast network stack, a value of 15-30 can be used.

Details about the RAAU Output Channel

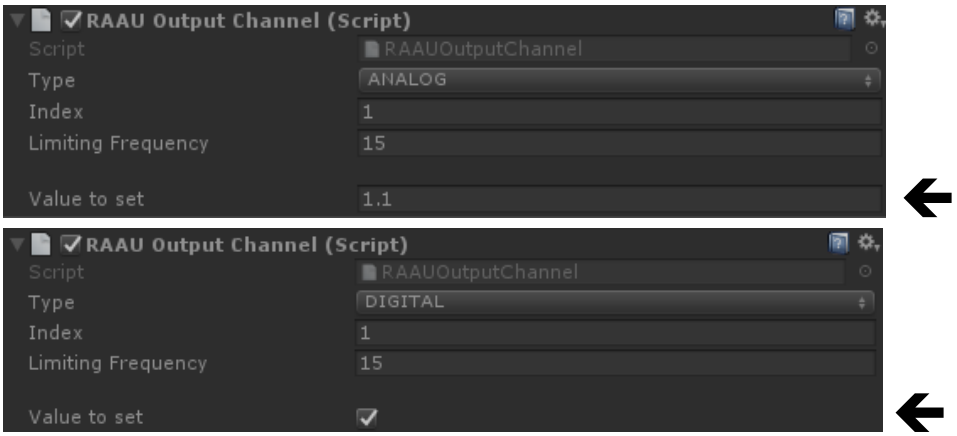
This component allows to preconfigure an output channel on which we can write.

You can use the accessor to set the value whenever required. A frequency limiter must be set to avoid creating too much network traffic that would destabilize Unity and Acq*Knowledge*.

Inspector panel

- Type: Analog or digital
- Index: index of the channel
- Limiting Frequency: maximum frequency possible for output updates

In play mode, you can set the values from the inspector for testing:



Runtime access and API Reference

You need to keep a reference to the component to use its features.

RAAcqK.AcqChannelType	**type**
	Type of channel (Analog/digital)

int	**index**
	Index of the channel
uint	**limitingFrequency = 15**
	Maximum frequency to send data

---

double	**valueAsDouble** `[get, set]`
	Sets the value of a digital channel as a decimal (double)
bool	**valueAsBool** `[get, set]`
	Sets the value of a digital channel as a boolean

Sample script

Please refer to **Sample4a_OutputDriving_Digital.cs** & **Sample4b_OutputDriving_Analog.cs**

For example:

```
//Setting values (use either accessor according to the type)
GetComponent<RAAUOutputChannel>().valueAsBool = true;

GetComponent<RAAUOutputChannel>().valueAsDouble = 2.2;
```

General notes

- It is advised to keep "limiting frequency" to a sensible value according to your usage. This feature can overflow the network if the frequency is too high, moreover if you need to set

several outputs in parallel in different parts of your code. On a recent computer with fast network stack, a value of 15 can be used.

- Try to set the values only when they change and by rounding the value before setting it in order to remove unneeded decimals.

Using the package in a new scene

For installation in Unity, please follow the "First steps and install" previously explained in the document.

If you create a new scene, you'll find in the **ReviaAssetsAcqKUnity\Prefabs** assets folder a prefab **RAAU_Connector** installing the basic configuration.

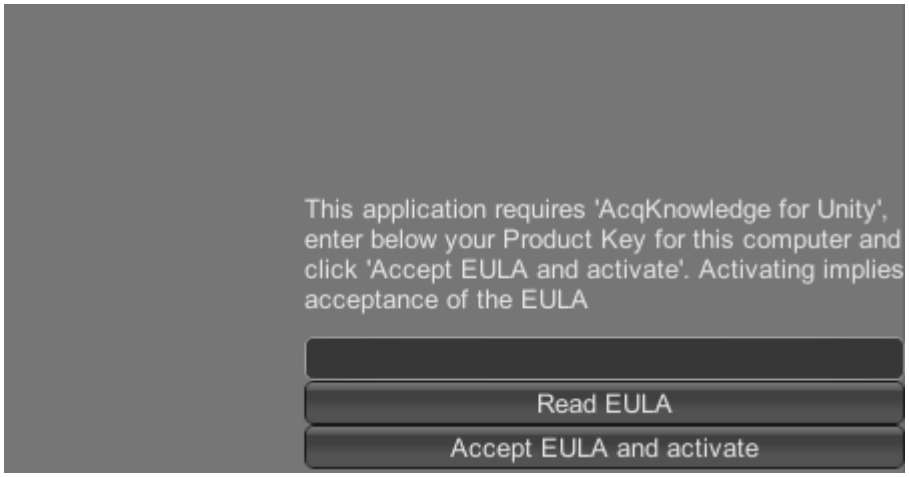
You can then add AcqKUnity components and your user scripts to build your application.

Building an application with Unity

Once developed, you can build your application for x86 or x64 in Windows and x64 on MacOS. No other specific configuration is necessary.

Note that for execution purposes on MacOS, libraries **libLexActivator.dylib** and **libLexActivator64.dylib** will be copied in the same folder than your build, please keep them next to your application to allow execution.

If you use the "Player mode licensing form" checkbox on the RAAU Connector the player will display automatically a small license activation form in the lower-right corner so the final user can activate his license at this stage if needed.



Troubleshooting and more resources

In order to access Reviatech Help centre for more documentation, Knowledge base, FAQ and support, please visit http://acqku.support.reviatech.com.