Integration with SONOS

Integration with audio systems is one of the most desirable features of an intelligent building. This tutorial is the first of a series of tutorials that will present integration between the **DOMIQ** and **SONOS** products.

By reading this manual you will learn elementary functions required to integrate **DOMIQ** with **SONOS**. In addition you will find how to:

- Integrate **SONOS** with other subsystems of an intelligent installation, such as LCN automation system or **SATEL** Intrusion Detection System.
- Use events and timers in the **Base** to integrate with **SONOS**.
- Control **SONOS** devices via **DOMIQ** user interfaces. However the solution presented in this tutorial is not intended to replace a dedicated application provided by **SONOS**.

1. What is SONOS

SONOS is a mobile multiroom audio system. Its structure is based on mobile devices with built–in speakers and amplifiers.

Source: www.sonos.com

Devices communicate with audio sources via the IP network using built–in Ethernet or wirelessly using a special protocol in the 2.4GHz band. It is also possible to use **SONOS Bridge**, which converts a dedicated **SONOS** protocol to the **WiFi** standard.

The use of IP network allows easy integration of the **SONOS** with an existing intelligent installation without any changes in a building infastructure.

SONOS devices are independent, which means that each device can play music from different source. However, if necessary, devices can be freely grouped to play the same audio track in sync. This feature in combination with mobility of the system allows you to create music zones spanning multiple rooms in your home.

2. SONOS Integration

As with other subsystems available in the **DOMIQ/Base**, integration with other devices is accomplished by the use of number of identifiers, which can be divided in two primary groups: state identifiers and commands.

State identifiers can be used to display current state of all connected **SONOS** devices, while commands are used to change that state.

All **SONOS** identifiers use the following syntax: `UAV.function.device`, where `function` is indicating specific function, while `device` is the name assigned to **SONOS** device via **SONOS** user interface.

2.1. State Identifiers

The table below presents all information functions with use examples. Examples use *office* as a name assigned to a **SONOS** device. Using information funcitons you can display the most important information about audio source.

Function	Example/Syntax	Explanation
`album`	`UAV.album.office`	Name of currently played album
`creator`	`UAV.creator.office`	Name of an artist of currently played track
`duration`	`UAV.duration.office`	Track duration
`mode`	`UAV.mode.office`	Playback mode: • normal • loop • random
`state`	`UAV.state.office`	Current playback state: • stopped • paused • playing
`title`	`UAV.title.office`	Returns the track title
`trackno`	`UAV.trackno.office`	The number of the file being played
`tracks`	`UAV.tracks.office`	The total number of files in a folder
`uri`	`UAV.uri.office`	Path to a file/audio stream (e.g. Internet radio)
`volume`	`UAV.volume.office`	Current volume

In case whe given audio source doesn’t provide one of the above information, then identifier will be set to `0`.

2.2. Commands

Following are commands to control **SONOS** devices. All with all other commands, they are prefixed with ”`C.`”.

Function	Example/Syntax	Explanation
`control`	`C.UAV.control.office=play`	Playback control. Available values: • `stop` – Stops playback. Playback resumed after receiving `stop` will start from the beginning of a playlist. • `play` – Starts playback. • `pause` – Pauses playback. • `next` – Next track. • `prev` – Previous track.
`mute`	`C.UAV.mute.office=1`	Mutes playback. Available values: 1 or 0. Value of 1 means: mute on.
`uri`	`C.UAV.uri.office=<path>`	Command used to set path to a file you want to play. `Path` has to point to a file on local or network hard drive. It can be also an address of an Internet stream. `Path` can point to a single file as well to a playlist in the m3u format. Additionally the SONOS software allows to import playlists and library from iTunes. **NOTE:** Calling the `uri` function doesn’t starts playback.To start the playback you need to invoke `control` function with the `play` value. A detailed description can be found below.
`volume`	`C.UAV.volume.office=30`	A command to control the volume. Available range is 0–100.

Automatic playback:

Bear in mind, that calling `UAV.uri` just points to a file or audio stream and does not start the playback. Playback needs to be started by a separate command. To automate this process in any **SONOS** player, you just need to create a single event. The event will be triggered automatically in case any **SONOS** player changes its path. In the result, the `play` command will be send to a player that triggered the event.

The event definition is as follows:

1. In the **DOMIQ/Base** configurator choose the **Events** tab and add an event.
2. In the **Channel** field type: `E.UAV.uri.(.+)` . This means, that the event will be triggered in case **any** player changes its path.
3. Leave the **Data** field blank.
4. Click on the **Add command** button. In the window that appears, in the **Name** field enter: `C.UAV.control.$C1`. In the **Value** type: `play`. `$C1` parameter passes a name captured by the `(.+)` pattern, in our case the name of a **SONOS** device.

3. Integration with Other Subsystems

Functions presented in the chapter 2. allow to integrate **SONOS** with any subsystem of an intelligent building. Further in this chapter we focus on integration with **LCN** automation system and with **SATEL** Intrusion Detection System.

3.1. Integration with LCN

As the integration possibilities are almost unlimited, we will present some of the most (in our opinion) useful features. Similarly to the chapter 2., our **SONOS** player is named `office`.

3.1.1. SONOS and Light Scenes

SONOS system can react according to the programmed schedule when a light scene is loaded. Reaction can be any, for example: it can stop playback or play a specific playlist. Below is and example of the ”guests” scenario. Let’s assume that ”guests” scene, when called, turns on certain lights and starts playback of a specific **SONOS** playlist. **SONOS** players support `.m3u` playlist files (used in WinAmp, iTunes, etc.).

LCN configuration won’t be presented in this tutorial. **Note**: In order to notify the **Base** module that a LCN scene was loaded, you need to send a `light scene` command to the **Base** (assign this command to any key in any table and call it when the scene is loaded). As a reminder: the **Base** module distinguishes scene numbers in range from 1 to 100 (register 1 includes scenes from 1 to 10, register 2 includes scenes from 11 to 20 and so on.)

In the **Base** module configuration is as follows:

1. Add an event in the **Events** tab.
2. In the **Channel** field type: `E.LCN.scene`
3. In the **Data** field enter the number of the loaded scene.
4. Click on the **Add command** button, in the window that appears in the **Name** field enter: `C.UAV.uri.<device>=<path>`, where `<device>` is the name assigned to a **SONOS** device and `<path>` is a path to the playlist file. In order to check path syntax, play this playlist using dedicated **SONOS** software and then check the `UAV.uri.<device>` identifier in the **State** tab in the **Base** configurator.

Below are examples of paths to files on a network drive:

A playlist file: `x-file-cifs://192.168.10.20/Music/playlist.m3u`

A mp3 file: `x-file-cifs://192.168.10.20/Music/track1.mp3`

3.1.2. Using LCN wall–mounted buttons

With the integration of **DOMIQ** and **SONOS,** controlling music using LCN wall–mounted buttons became possible. Just program LCN buttons so when pressed send ”`send keys`” command to the **Base** module. **Base** module should react to this command by invoking certain (any) actions in the **SONOS** system.

Study the example below to learn how to configure the **Base** module. In this case, **Base** module will send `play` command to the **SONOS** system when `A5 hit` command is received.

1. Add an event in the **Events** tab.
2. In the **Channel** field type: `E.LCN.key.`
3. In the **Data** field enter: `A5 hit`. Note: There is a space character between `A5` and `hit`.
4. Click on the **Add command** button. In the **Name** enter: `C.UAV.control.<device>,` in our case: `C.UAV.control.office`. In the **Value** field type: `play`.
5. Similarily you can configure any **SONOS** player functionality.

Based on example presented above you can define more complex scenes such as a scene for exiting/entering home. Such scene for example can stop/start playback in whole **SONOS** system. The definition of this scene is almost the same as presented above. The only difference is the number of command sent to the **SONOS** system.

3.2. Integration with SATEL Intrusion Detection System

Integration with Satel Alarm System allows to achieve functionality similar as in the case of integration with LCN. For example arming the alarm can stop playback in whole **SONOS** system. Disarming the alarm can initiate playback of a selected playlist and so on. In the following, we will present how to implement this and describe how to use **SONOS** in case of alarm.

3.2.1. Scene for Exiting the Building

To achieve this functionality we will use arming event sent to the **Base** module by the alarm panel each time someone arms the alarm. In the **Base** you need to create an event that, when triggered will send commands to **SONOS** devices to stop playback. The definition of the event is as follows:

1. Add an event.
2. In the **Channel** field enter `E.IDS.armed.<zone>`, in the **Data** type: `1`. As the `<zone>` enter the number of a zone that is armed after you leave the building.
3. Click on the **Add command** button. In the **Name** field enter: `C.UAV.control.<device>` and in the **Value** type: `stop`.
4. Repeat step no. 3 for the remaining **SONOS** devices.

3.2.2. Scene for Entering the Building

Disarming the alarm can initiate playback of a selected playlist. To achieve that create an event based on procedure presented in section 3.1.1 SONOS and Light Scenes.

In the **Channel** field enter: `E.IDS.armed.<zone >` and in the **Data** type: `0`, where the `<zone>` is the number of a zone that is disarmed. The rest of the procedure is identical.

3.2.3. Alarms

SONOS can also be used for alarm purposes, contributing to increasing safety. In case of alarm **SONOS** can play sounds informing about fire, flooding, burglary or startle burglars, by playing the sound of barking dogs:)

In this case configuration is limited to a single event that will be triggered when the alarm goes off.

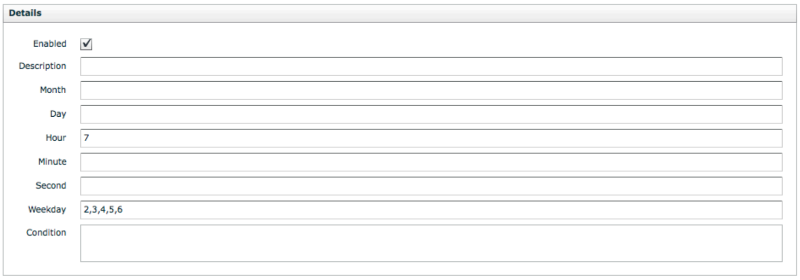
As an example we will present scenario for burglary alarm. Procedure is very similar to the scene for entering the building:

1. Add an event. In the **Channel** field enter: `E.IDS.alarm.(%d+)`. Thanks to the `(%d+)` pattern, the event will be triggered by the alarm in any zone.
2. In the **Data** field enter: `1`.
3. To make sure that the sound will be clearly audible, set the volume to an appropriate level. For this purpose, **Add command** in the **Name** field, type: `C.UAV.volume.<device>`, in the **Value** field enter the volume level you want to set.
4. Next, add another command that points the file you want to play – the definition is analogous to step 4 in section 3.1.1 SONOS and Light Scenes. Remember to change the path to a file.

4. Music Alarm Clock

Using timers you can create musical alarm clock. You just need to define a timer that at a certain time, on selected days of the week will play your favourite playlist or radio. Below is an example of a timer that starts playlist on weekdays at 7 a.m.

1. Add a timer and set its properties according to the picture below:



Details

Enabled	☑
Description	
Month	
Day	
Hour	7
Minute	
Second	
Weekday	2,3,4,5,6
Condition	

2. In the **Action** section define commands to be executed when the timer is triggered – setting the volume and point the path to the playlist file.

5. Control from Visualization

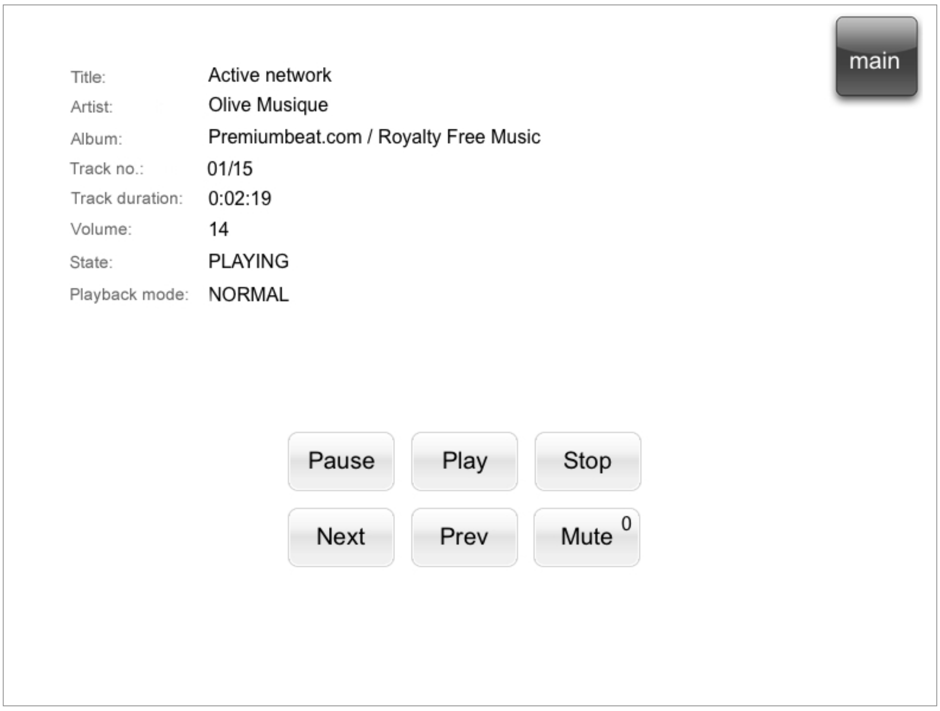
By using integration functions described in the chapter 2. you can create a simple user interface that will allow you to control playback and display key information about the played audio.

5.1. Control

To build the interface we will use **Switch** elements. The example below limits to a single button, as the other buttons are created the same way. The only difference is a command assigned to a button. The example below presents the **Play** button.

1. Add a **Switch** to a visualization.
2. In the **Command** field enter: `UAV.control.<device>=play`, in our case: `UAV.control.office=play`.
3. Repeats steps 1 and 2 for the remaining buttons.
4. Use **Text** elements in combination with proper information functions presented in section 2.1 to display information about played audio.

Examplary interface layout is presented in the picture below.



Title:	Active network
Artist:	Olive Musique
Album:	Premiumbeat.com / Royalty Free Music
Track no.:	01/15
Track duration:	0:02:19
Volume:	14
State:	PLAYING
Playback mode:	NORMAL