This description is valid for version 0.4.8 of the Tellstick plugin. It will only work with OSA version 4.8 and up.

This plugin will connect a Tellstick (Tx) or Tellstick Duo (Rx and Tx) unit to OSA. It will handle both sending and receiving of messages from devices supported by the Tellstick controller.

*TellstickDuo Tellstick*

The Tellstick controllers are handled via a dll, TelldusCore.dll. Telldus also have a GUI, Telldus Center, to configure and control the supported devices.

To be able to use the Tellstick plugin you should first configure and check the function of all devices from Telldus Center.

The Telldus software allows that you use the OSA plugin and the Telldus Center at the same time. This is very practical as it allows you to simulate devices from Telldus Center when developing functionality in OSA.



telldusCore.dll

Telldus Center

OSA  
Tellstick plugin

*Tellstick controller*

OSA  
Objects

OSA  
Objects

OSA  
Objects

OSA  
Objects

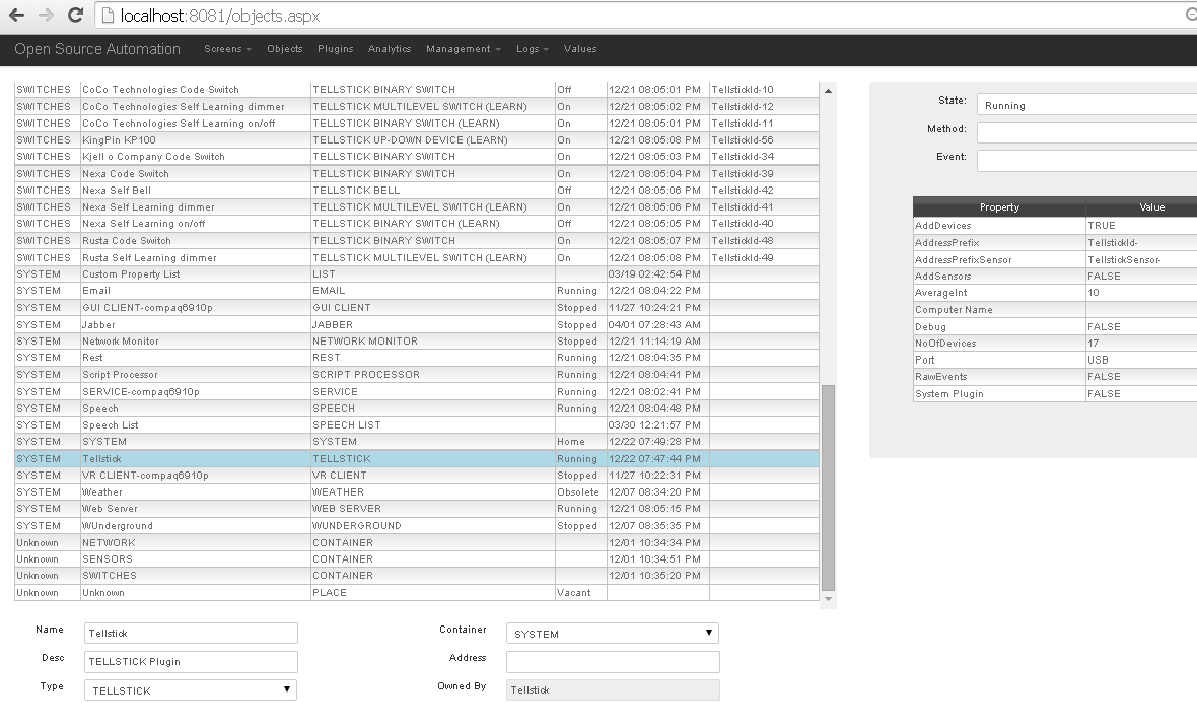
# Installation and configuration

1. Download and install Telldus Center as described on [www.Telldus.com](http://www.Telldus.com). This will also install the necessary drivers and the TelldusCore dll.
2. Download and install the Tellstick.osapd file according to the plugin installation instructions on [www.opensourceautomation.com](http://www.opensourceautomation.com).  
   After installation you will have the following files in the AddIns\Tellstick folder:

Per Tobiasson

* **OSAE.Tellstick.dll**This is the Tellstick plugin run by OSA.
* **Tellstick.osapd**  
  This is the plugin description needed by OSA to run the plugin.
* **Screenshot.jpg**This is a picture of a Tellstick unit used by OSA Manager.
* **Install.sql**  
  This is a file with all SQL commands needed to create the different Object types in the MySQL database during installation.
* **TelldusNetWrapper.dll**This is a wrapper around the TelldusCore.dll to adopt it to .NET.
* **Tellstick OSA plugin\_YYYY-MM-DD.pdf**This manual.

1. Run Telldus Center and configure all your Tellstick devices. Check that you can use Telldus Center to control all your devices. Give all Tellstick devices unique names. It is allowed to have the same name for different devices in Telldus Center, but that will create conflicts and inconsistence in OSA.
2. Run OSA Manager and start the service. The plugin will now automatically create:  
   \* The Tellstick OSA object needed by the plugin  
   \* OSA Objects for all Tellstick devices that were previously configured in Telldus Center.



Properties of the Tellstick object

* 1. ‘**AddDevices**’ can be used to turn on or off the feature to automatically add new devices. Default is TRUE. When you have objects for all your devices this property can be set to FALSE
  2. **‘AddressPrefix**’ should be set to something that will be used as a prefix for the address of all found Tellstick devices. When the plugin connects to the Tellstick unit it will automatically create OSA Objects for all found devices. The address of the Tellstick devices is a simple Integer number why there is a risk of address conflicts with other plugins. The ‘AddressPrefix’ property should be used to create unique addresses for all devices. The default value is ‘TellstickId-‘ and that should work fine.
  3. ‘**AddressPrefixSensor**’ is used in the same way, but it is used for all sensors. Default is ‘TellstickSensor-‘.
  4. ‘**AddSensors**’ can be used to turn on or off the feature to automatically add new sensors. Default is TRUE. When you have objects for all your own sensors this property can be set to FALSE. Then you will not have unnecessary ghost objects when the Tellstick catches signals from your neighbors.
  5. ‘**Author**’ name of the plugin writer.
  6. ‘**AverageInt**’ the plugin will calculate an average value for each sensor. This property defines the length of the time period for the average calculation. The average values is a good way to reduce the number of events written to the database.
  7. ‘**ComputerName**’ can be used to run the Tellstick plugin on more than one computer.
  8. ‘**Debug**’ when set to TRUE additional logging will be written in the log table. Default is FALSE.
  9. ‘**NoOfDevices**’ will be automatically filled in when the plugin connects to the Tellstick unit, and will tell how many devices TelldusCore has found. This value does not include the number of sensors.
  10. ‘**Port**’ will be set to the used port, i.e. will always be set to USB.
  11. ‘**RawEvents**’ the Tellstick device can when set to TRUE send raw data to OSA. Default is FALSE. The only action when set to TRUE is that the raw events are written in the log table.
  12. ‘**System Plugin**’ should be set to FALSE.
  13. ‘**TrustLevel**’ for this plugin.
  14. ‘**Version**’ version number of the plugin.

# Object types

There are several OSA Object types (automatically inserted during the plugin installation) that are used for the Tellstick devices. All Objects will automatically be created with the correct Object type depending on the device capabilities. The following object types exist:

* **TELLSTICK**  
  This is the Object type for the Tellstick plugin.
* **TELLSTICK BELL**  
  Object type for door bells. Supports BELL method and event.
* **TELLSTICK BINARY SWITCH**  
  Object type for binary switches. Supports ON and OFF methods and events.
* **TELLSTICK DEVICE**  
  This object type supports all methods and events. This type will not be used on any automatically created devices.
* **TELLSTICK GROUP DEVICE**  
  This object type will be used for group devices that are configured in Telldus Center. As it is possible to create groups consisting of devices with different capabilities, this Object type will support all methods and events. But don’t worry; if you try to use a method on a device that does not support it, it will be stopped in the plugin.
* **TELLSTICK HUMIDITY SENSOR**  
  This Object type will be used for Humidity sensors.
* **TELLSTICK MULTILEVEL SWITCH**  
  Object type for dim switches. Supports ON, OFF and DIM methods and events.
* **TELLSTICK TEMPERATURE SENSOR**  
  This Object type will be used for temperature sensors.
* **TELLSTICK TEMP-HUMIDITY SENSOR**  
  This Object type will be used for sensors supporting both temperature and humidity.
* **TELLSTICK UP-DOWN DEVICE**  
  Object type for projector screens etc. Supports UP, DOWN and STOP methods and events.

# Objects

### Devices

When the plugin starts it will pick up all devices from the Telldus-service. If there already is an object with the same address or name that object will be used, otherwise a new object will be created. If a device is removed from Telldus Center, the corresponding OSA object will change state to DISABLED, but it will not be automatically deleted.

**Name** of the object will be set to the name received from Telldus-service. You should not manually change the name of the object, as this will create an inconsistence with Telldus Center. Instead change the name from Telldus Center.

**Description** will be set to the model and protocol received from Telldus-service. It can be manually changed on the object if you want another description.

**Type** will automatically be set to an object type that corresponds to the capabilities of the Tellstick device. It makes no sense to change this object type.

**Container** will be set to SWITCHES when the object is created. You can change it to something suitable.

**Address** will be set automatically. It will be constructed from the AddressPrefix property of the TELLSTICK plugin object, and the DeviceId property from Telldus-service. If you change it, it will create an inconsistence with Telldus Center.

**Min Trust Level**, trust level needed to change this object. Default 50.

**Owned By**, object must be owned by ‘Tellstick’ otherwise methods on the object will not work.

**Enabled** will be checked when the object is created.

### Sensors

Objects will be created for each sensor the Tellstick device sends data for. They will not be created when the plugin starts, you will have to wait until the sensors have sent data. If you don’t want data to be updated and events triggered for a sensor (it could be your neighbor’s sensor!), you can set the status to DISABLED. When objects have been created for all your sensors you can set the property AddSensors on the Tellstick object to FALSE, then no more sensors will be added. You can also delete eventual sensors from your neighbors.

**Name** of the object will be set to the AddressPrefixSensor property from the Tellstick plugin object, plus the device id received from Telldus-Service. You can change the name manually. This name will not change automatically if you change the name of the sensor in TelldusCenter.

**Description** will be set to the model and protocol received from Telldus-Service. It can be manually changed on the object if you want another description.

**Type** will automatically be set to an object type that corresponds to the capabilities of the Tellstick sensor. It makes no sense to change this object type.

**Container** will be set to SENSORS when the object is created. You can change it to something suitable.

**Address** will be set automatically. It will be constructed from the AddressPrefixSensor property of the TELLSTICK plugin object, and the DeviceId property from Telldus-Service. If you change it, it will create an inconsistence with Telldus Center.

**Min Trust Level**, trust level needed to change this object. Default 50.

**Owned By**, object must be owned by ‘Tellstick’ otherwise methods on the object will not work.

**63Enabled** will be checked when the object is created.

## Properties for devices

The different Object types use the following properties:

* **Debug** if set to TRUE this object will write extra debug information to the log table.
* **DeviceAddress** (string)  
  This property will automatically be set to the address used for the device. It is read from Telldus Center when the plugin starts. It consists of three parts separated with /, house code, code and unit. All three parts are not used for all types of devices. The device address is not used for anything inside the plugin; it is only there for information.
* **DeviceChangeEvents** (Boolean)  
  If set to TRUE, the plugin will update the object when the Telldus-Service sends device change information. For example if you change the device address of the device in Telldus the DeviceAddress property of the OSA Object will automatically be updated.
* **DeviceId** (Integer)  
  This is a unique number for the device received from Telldus-Service. It will automatically be set when the plugin starts. It is used within the plugin when a method should send a command to a Tellstick device.
* **DeviceType** (string)  
  This property is automatically set when the plugin starts. It can be any of “Device”, “Group” or “Scene”. (Scene is not implemented in current version of Telldus-Service.)
* **LastSentCommand** (string)  
  This property will automatically be set to the last command sent or received for the device.
* **Level** (string)   
  This property will automatically be set to the last dim level that was sent or received for a multilevel switch. It will be a value between 0 and 100. The Level will also be updated to 0 or 100 for a binary switch.
* **Methods** (string)  
  This property will automatically be set to a string indicating what methods this device supports. For example a multilevel switch will have “ON-OFF-DIM”.
* **Model** (string)  
  This property will automatically be set to the model of the device, received from Telldus-Service.
* **Protocol** (string)  
  This property will automatically be set to the protocol used by the device, received from Telldus-Service.

## Properties for sensors

The different Object types use the following properties:

* **AverageHum** (float)  
  This is a calculated average value for humidity. The value is calculated over the period given by the property AverageInt on the Tellstick object. This can be used to reduce the number of events that will be written to the database. Un-tick ‘Track Changes’ for the Humidity property on the Object type, and log only the events for the average value and you will have significantly less loggings.
* **AverageTemp** (float)  
  Same but for temperature.
* **Battery** (string)  
  Currently not handled by the Telldus-Service.
* **DeviceAddress** (string)  
  For sensors this will not be automatically set to any value, and is not used within the plugin. Can be freely used.
* **DeviceId** (Integer)  
  This is a unique number for the sensor received from Telldus-Service. It will automatically be set when the plugin starts. It is used within the plugin to find the object when Telldus-Service sends updated values for the sensor.
* **DeviceType** (string)  
  This property is automatically set to “Sensor” when the object is created.
* **Humidity** (float)  
  This property will be set to the last humidity value received from a Tellstick sensor.
* **MaxHum** (float)  
  This will be set to the max humidity value since the last reset. Reset can be done with the Reset MaxMin method. If you want the max and min values for each day, run the Reset MaxMin method from a script.
* **MaxTemp** (float)  
  Same but for max temperature.
* **MinHum** (float)  
  Same but for min humidity.
* **MinTemp** (float)  
  Same but for min temperature.
* **Model** (string)  
  This property will automatically be set to the model of the device, received from Telldus-Service. Currently not received for sensors.
* **OFF TIMER** (Integer)  
  When this property has a value a timer will be started every time the sensors status is set to ON. The timer value is in seconds. When the timer reach zero, the status will be set to OFF. In this way the status will reflect if the sensor is sending data or not. Useful to indicate sensors that has stopped working. Default is 600 seconds.
* **Protocol** (string)  
  This property will automatically be set to the protocol used by the device, received from Telldus-Service. Currently not sent for sensors.
* **Status** (string)  
  Not implemented. Is here just to be compatible with the same object type for RFXCOM.
* **Temperature** (float)  
  This property will be set to the last temperature value received from a Tellstick sensor.
* **Unit** (string)  
  Will not be automatically set, but can manually be set to indicate the unit for temperature values i.e. Celsius or Fahrenheit.

## Methods

The following methods are available on the Tellstick object types

* **BELL**Sends a BELL command to the device.
* **DEBUGOFF**Turn off debug logging for this object.
* **DEBUGON**Turn on debug logging for this object. The Debug property on the Tellstick object must be set to true, otherwise no debug will be logged.
* **DIM**Sends a DIM command to the device. You will be asked about the Level to send.
* **DISABLE**  
  This method will change the state of the OSA object to DISABLED. Other methods cannot be used on the object until the ENABLE method has been used.
* **DOWN**Sends a DOWN command to the device.
* **ENABLE**Will change the state of the object to UNKNOWN. This means that other methods can be used on the object.
* **EXECUTE**Will send an EXECUTE command to the device. Currently no devices support this.
* **LEARN**Will send a learn command to the device.
* **OFF**Will send an OFF command to the device. Level will be set to 0.
* **ON**Will send an ON command to the device. Level will be set to 100.
* **STOP**Will send a STOP command to the device.
* **TOGGLE**Will send a TOGGLE command to the device. Currently no devices support this.
* **UP**Will send an UP command to the device.
* **RESETMAXMIN**Will reset the max/min values for a sensor.

## Events

There are corresponding events to all methods (except ENABLE and DISABLE). The events will be fired when a deviceEvent is received from Telldus-Service, i.e. a signal from Telldus Center or from a remote control. Telldus-Service will automatically send back a deviceEvent after a command has been sent to the device with a method. The only exception to this is the BELL method. Telldus-Service does not send device events for BELL.

# Source code

The plugin is written in C#. The following files are in the source directory OSAE.Tellstick:

* **Tellstick.cs**This is the C# source code for the plugin.
* **Install.sql**  
  SQL script with commands for inserting the necessary OSA Object types into the MySQL database.
* **OSAE.Tellstick.csproj**  
  Visual Studio project file.
* **OSAE.Tellstick.dll**  
  The compiled plugin.
* **OSAE.Tellstick.sln**Visual Studio solution file.
* **OSAE.Tellstick.suo**  
  Visual Studio Solution User Options.
* **Screenshot.jpg**JPG picture of the ThellstickDuo.
* **TelldusNetWrapper.dll**  
  This is a .Net wrapper around the TelldusCore.dll. This wrapper is modified from the one supplied by Telldus. The modified source can be found in the TelldusNetWrapper folder.
* **Tellstick OSA plugin\_YYYY-MM-DD.docx**  
  This manual. It is created with Microsoft Word 2007.
* **Tellstick OSA plugin\_YYYY-MM-DD.pdf**  
  This manual in pdf format.
* **Tellstick.osapd**The plugin description file for OSA.