Implementing a Device Property

You can implement a device property to provide a configuration network input to your application. Device properties are properties which apply to the device as a whole and not to specific block or datapoint. are not members of a block. Because blocks provide grouping of datapoints and properties into logical entities with well-defined behavior, device properties are not commonly used. However, there are cases when device properties can be useful, and they are often used for rapid prototyping or testing. See Implementing a Block for details about implementing blocks and datapoints or properties within blocks.

To implement a device property, use the **IML property** class. You can specify other attributes to support advanced controls over the property implementation, such as the **init()** attribute to initialize a property, and the standard **Const**, **DeviceSpecific**, **Disable**, **Manufacture**, **Offline**, and **Reset** property flags.

When you define a device property with IML and build your application, the IzoT Interface Interpreter creates a C definition for your datapoint declaration. Unlike device datapoints, the IzoT Interface Interpreter does not create an additional type definition for the property variable.

Example

This example implements two device properties, one implementing the SCPTlocation location property type (a structure containing a 31-byte ASCII string), and the other implementing a SCPTnwrkConfig network configuration source property type (an enumerated value used to indicate whether the device is commissioned by a network tool, or self-installed).

```
#include "IzoTDev.h"

SCPTlocation  myLocation;  //@Izot property
SCPTnwrkCnfg  netConfig;   //@Izot property init(CFG_EXTERNAL)
```

When you build this application, the IzoT Interface Interpreter generates the following C type definitions within the **IzotDev.h** file for the **myLocationdeclaration**. This type is defined as a structure containing an array of 31 bytes to hold a short nul-terminated string with alphanumeric location information such as **Room 101**.

```
typedef struct {
        unsigned char ascii_ [31];
} SNVT_str_asc;
typedef SNVT_str_asc SCPTlocation;
extern SCPTlocation myLocation;
```

You can access the property's current value directly from the **myLocation** variable.

Device properties are always implemented as property datapoints. A **global_index** value is not automatically published for a property, even if it is implemented as a configuration datapoint and therefore, as a datapoint, has a global index value. The index value of a property (implemented as a datapoint) is not typically used, but if your application requires that level of detail, you can find it using **IzotGetDatapointIndex()**.

See IML Syntax Summary for a complete list of modifiers and attributes supported with the device datapoint implementations.

See Data Types for more about the IML data types.