SensIDL Quickstart Guide

Getting Started

To create a SensIDL project, first create a new Java project. After that create a new file in that project with the filename extension *.sidl*. The file will be from now on opened in the SensIDL Editor. Now you can start using the SensIDL Grammar to define the sensor.

SenIDL language Elements

1. *SensorInterface*

The Model starts with a SensorInterface which contains all the information.

```
sensorInterface name /**Description*/
{

	// EncodingSettings

	// DataDescription

}
```

2. *EncodingSettings*

A SensorInterface contains encoding settings which set the alignment, endianness and the encoding of the sensor

```
encoding: SENSIDL_BINARY, endianness:
BIG_ENDIAN, alignment: 1 BIT
```

3. *DataDescription*

Furthermore, a SensorInterface contains also Data Description, indicated by the Keyword sensorData. The Data Description contains all the data.

```
sensorData {

	// DataSets

}
```

4. *DataSets*

The Data Description contains any amount of DataSets. A DataSet can also use other DataSets. Furthermore, the DataSets contain the Data declaration

```
dataSet DataSet1 uses DataSet2,
DataSet3 /**Description*/ {

	// Data

}
```

5. *Data*

Data can be either be MeasurementData, NonMeasurementData, Methods or Data Lists.

a) *MeasurementData*

MeasurementData is data that is gained during a measurement process

```
voltage as INT16 in V /**Description*/
```

MeasurementData can also be adjusted (as long as it's type is not string or Boolean). Adjustments can either be with a scaling factor and an offset or by linear mapping. Furthermore, there is the option to adjust MeasurementData with a range.

aa) Adjustment with scaling factor and offset

```
voltage as INT16 in V adjusted by
linear mapping [0;255] => [0;1] as
FLOAT /**Description*/
```

bb) Adjustment by linear mapping

```
voltage as INT16 in V adjusted with
scaling factor: 5 and offset: 1.5
/**Description*/
```

cc) Adjusted with range

```
voltage as INT16 in V adjusted with
range [0;1] /**Description*/
```

Note that a MeasurementData can't be adjusted with a) and b) at the same time, however it can have a) and c) or b) and c) at the same time.

b) NonMeasurementData

NonMeasurementData is not gained by a measurement.

```
deviceType as UINT8 constant value="1"
/**Description*/
```

c) Method

Furthermore, there is the Option to declare Methods for the device. Methods can have Parameters and return values.

```
method1(INT8 parameter1, STRING
parameter2):BOOLEAN as Method
/**Description*/
```

d) Lists

There is the Option to declare Lists of a specific data type with the grammar.

```
list1 as list of INT8 /**Description*/
```

e) Type System and Units

SensIDL supports many different datatypes, that are from int8 to int64 and also the unsigned version of them, Boolean, string, double and float.

Furthermore, the units are based on JScience and so SensIDL supports all Units that JScience supports, that are all SI Units and many non SI-Units.

Generate Code

When you finished declaring your Sensor with the SensIDL Grammar, you want to generate code for it with the SensIDL Wizard. To do so make a right click either anywhere in the Editor or in the Package Explorer on your .sidl file. This will open the SensIDL Wizard. In the wizard specify the path and the language you want to generate the Code in and klick Generate Code. The Code is now generated.

Complete Example

```
sensorInterface name /**Description*/ {

	encoding: SENSIDL_BINARY,
	endianness: BIG_ENDIAN,
	alignment: 1 BIT

	dataSet DataSet1 uses DataSet2,
	DataSet3 /**Description*/ {

		powerFactor as INT16 in
		Dimensionless adjusted by
		linear mapping [0;255] =>
		[0;1] as FLOAT

		read() as Method

	}

	dataSet DataSet2 uses DataSet3 {

		power as INT16 in W

		current as INT16 in A

	}

	dataSet DataSet3 {

		voltage as INT16 in V

		list as list of INT8

	}

}
```

Further Reading

Link to the SensIDL – GitHub Repository:
https://github.com/SENSIDL-PROJECT/SensIDL

How to install SensIDL:
https://github.com/SENSIDL-PROJECT/SensIDL/wiki/Download

SensIDL Release Update Site:
https://sdqweb.ipd.kit.edu/eclipse/sensidl/releases/latest/

SensIDL Website:
http://sensidl-project.github.io/SensIDL/