API documentation for the Infrared Proximity Module, DBSU10, Group 6

The Infrared Proximity Module is a module which is able to receive and process data from 3 separate infrared sensors. The main component that the module will provide to the user is if the sensors are being triggered.

Module setup

The module's behaviour on how to process incoming signals from the sensors relies on a set of customizable variables. These variables can be altered through OOCSI calls using the following call request; **setIR6**:

```
OOCSICall call = oocsi.call("setIR6", 1000).data("variable",
value);
call.sendAndWait();
if (call.hasResponse()) {
  val = call.getFirstResponse().getBoolean("variableChanged",
false);
}
```

Generally, the response from the call would be in the form of:

```
response.data("variableChanged", true); \\ example:
response.data("pinAmountChanged", true);
```

However, for the clear variable this is slightly different as it responds with an integer representing the sensor that was cleared:

```
response.data("TriggerCleared", 0);
```

Variable	Type	Default	Description
pinAmount	Integer	3	Amount of active sensors.
channelName	String	null	Name of the channel over which the module will send trigger messages.
delay	Integer	1000	Time delay in ms for which a sensor needs to be active to be triggered.
respondOnTrigger	Boolean	true	Let the module send messages on trigger.
stick	Boolean	false	Stick trigger values. If true, a sensor will stay triggered once it triggers.
clear	Integer	0	Clear trigger value of a sensor; 0 will clear all.
reset	-	-	Resets all the variables back to their default values.

Getting Values

If *repondOnTrigger* is set to true and a *channelName* is defined, then the module will send trigger messages over the channel using this format:

```
oocsi.channel(channelName).data("IR6sensor", triggeredSensor);
```

With *triggeredSensor* representing the sensor that is being triggered (1, 2 or 3). To retrieve these messages, the user must be subscribed to the corresponding channel name with an eventhandler defined.

Trigger values and setup values can also be manually requested using the call; **getIR6**:

```
OOCSICall call = oocsi.call("getIR6", 1000).data("triggered",
value);
call.sendAndWait();
if (call.hasResponse()) {
  arr = (boolean[]) call.getFirstResponse().getObject("triggers");
}
```

Variable	Description

triggered	Responds with the trigger array, containing boolean values.
	To retrieve this from a call, use:
	`(boolean[])`
	`call.getFirstResponse().getObject("triggered");`

setup	Responds with the setup values and their corresponding types.
	To retrieve this from a call, use one of the following:
	`call.getFirstResponse().getInt("pinAmount", 0);`
	`call.getFirstResponse().getString("channelName", "");`
	`call.getFirstResponse().getInt("delay", 0);`
	`call.getFirstResponse().getBoolean("stick", false);`