Device.GetCharacteristics

to get the service characteristics

Syntax

```
IR.CreateDevice(IR.DEVICE_BLE_DEVICE).GetCharacteristics()
```

input	**sample**	**description**
`ID`	IR.DEVICE_BLE_DEVICE	type: Number the identifier of the BLE scanner
output		
`Array of objects`		type: Array the array of characteristics (characteristic - object)

Example:

```
var device = IR.CreateDevice(IR.DEVICE_BLE_DEVICE);
device.GetCharacteristics();
```

The fields of the characteristic object (the OPTIONAL marker means you are not required to fill in the field, TO_DELETE - the field can be deleted later as it is almost useless):

- Uuid: (String) - The characteristic identifier;
- Permissions: (Signed) - TO_DELETE, the characteristic permissions, partly duplicate data from Properties, there is no one for reading on iOS;
- Properties: (Signed) - The permissions for communication with the characteristic;
- Value: (Array) - OPTIONAL, the array of bytes (container for Read / Write). It can be read or edited if * Properties allow doing so. The syntax is determined by the manufacturer;
- Descriptors: (Array) - OPTIONAL, the array of descriptors.

Properties of the characteristic. They determine communication with the characteristic (they are summed up bitwise. A separate characteristic can contain several properties but at that present only one number). To extract a separate property you are required to use 'bitwise AND' Prop & IR.CharacteristicPropertyBroadcast.
The list of properties:

```
IR.CharacteristicPropertyBroadcast = 0x01;
IR.CharacteristicPropertyRead = 0x02;
IR.CharacteristicPropertyWriteWithoutResponse = 0x04;
IR.CharacteristicPropertyWrite = 0x08;
IR.CharacteristicPropertyNotify = 0x10;
IR.CharacteristicPropertyIndicate = 0x20;
IR.CharacteristicPropertyAuthenticatedSignedWrites = 0x40;
IR.CharacteristicPropertyExtendedProperties = 0x80;
```