

## SunRiseSet

### Purpose

The purpose of this script library is to get local times for sunrise and sunset, for any position(latitude/longitude), decimal degrees.

### Prerequisites

Import the script library, *scExtensions.zip*, under script libraries in Ethis Admin.  
Import the script library, *scSunRiseSet.zip*, under script libraries in Ethis Admin.  
Add the global *Boolean* variable, *bSun1*, if you want to test the examples in this documentation.

### License requirements

The solution works on license level Advanced, Universal level 4, or higher.

### Main script with explanation

Add the following code to your main script, if you want to test the functions in the script library.

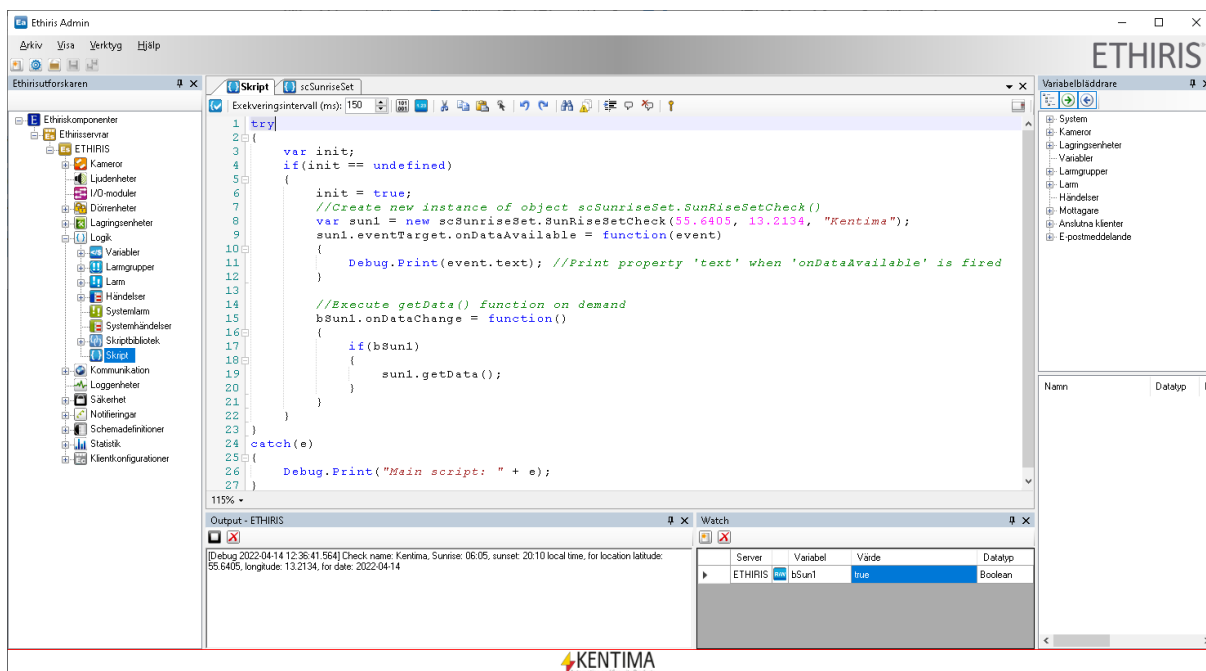
**Note!** When you create the *SunRiseSetCheck()* object, make sure you only do it once, and not on every script loop. You can for example use an *if*-statement like below.

```
try
{
    var init;
    if(init == undefined)
    {
        init = true;
        //Create new instance of scSunriseSet.SunRiseSetCheck()
        var sun1 = new scSunriseSet.SunRiseSetCheck(55.6405,13.2134,"Kentima");
        sun1.getEventTarget().onDataAvailable = function(event)
        {
            Debug.Print(event.text); //Print property 'text'.
        }
        //Execute getData() function on demand
        bSun1.onDataChange = function()
        {
            if(bSun1)
            {
                sun1.getData();
            }
        }
    }
}
catch(e)
{
    Debug.Print("Main script: " + e);
}
```

You can now add the variable *bSun1* to the *Watch Panel*. If you **double-click** *Value* of *bSun1*, to set the value to *true*, the function *getData()* will execute. Output will be printed to the *Debug Output Panel*.

The function *getData()* will trigger an event which will trigger event function *onDataAvailable*. The event contains the properties *sunrise*(string), *sunset*(string), *latitude*(number), *longitude*(number), *date*(string), *name*(string) and *text*(string).

In the example below we print the property *text*, using *Debug.Print()*.



The screenshot shows the Ethisis Admin interface with the following components:

- Script Editor:** A JavaScript script named `scSunriseSet` is shown. It includes an initialization function `init` that sets `init = true` and creates a `sun1` object. The `onDataAvailable` event handler calls `Debug.Print(event.text)`. The `onDataChange` event handler calls `sun1.getData()` if `bSun1` is true. The main script prints `"Main script: " + e`.
- Watch Panel:** A table showing the state of variables. The variable `bSun1` is highlighted, and its value is `true`.
- Debug Output Panel:** A log entry showing the output of the script: `[Debug 2022-04-14 12:36:41.564] Check name: Kentima, Sunrise: 06:05, sunset: 20:10 local time, for location latitude: 55.6405, longitude: 13.2134, for date: 2022-04-14`

Server	Variabel	Värde	Datatyp
ETHIRIS	bSun1	true	Boolean

## Script library with explanation

```

9 // Check for existence of script library 'scExtensions'
10 if(typeof scExtensions == "undefined")
11 {
12     throw new Error("Script library 'scExtensions' is missing."); //Script library is missing, we can't continue
13 }

```

**Line 10:** Check to see if script library `scExtensions` exists, which is required for this script library to function correctly.

```

15 //Calculate local time for sunrise and sunset, based on coordinates for latitude and longitude
16 this.calculateSunRiseSet = function(lat, lon)
17 {
18     var altit = -35.0/60.0;
19     var upper_limb = 1;
20
21     var veear = System.Clock.Year:

```

**Line 16:** Defines a function to calculate sunrise and sunset times, for a specific location, with latitude and longitude as input.

```

85 //Create object to return
86 var sunTime = new Object();
87 sunTime["rise"] = ("0" + sunRiseHour).slice(-2) + ":" + ("0" + sunRiseMinute).slice(-2);
88 sunTime["set"] = ("0" + sunSetHour).slice(-2) + ":" + ("0" + sunSetMinute).slice(-2);
89
90 return sunTime;

```

**Line 86:** Defines an Object which will contain the times as formatted strings.

**Line 90:** Return object to the caller function.

```

93 //Create Sunrise sunset object
94 this.SunRiseSetCheck = function(latitude, longitude, name)
95 {
96     try
97     {
98         this.latitude = (typeof latitude == "undefined" || typeof latitude != "number") ? 55.6405 : latitude; //Set default latitude value to 55.6405, if
99         this.longitude = (typeof longitude == "undefined" || typeof longitude != "number") ? 13.2134 : longitude; //Set default longitude value to 13.2134,
100        this.name = (typeof name == "undefined" || typeof name != "string" || name == "") ? "<NONAME>" : name; //Set default name
101        this.eventTarget = new EventTarget(); //Create EventTarget on the object

```

**Line 94:** Defines a function ("Class" object) to check for sunrise and sunset at a certain interval.

You create new instances of this object in your main script using the following construction

```
var _name_ = new scSunriseSet.SunRiseSetCheck(_lat_, _long_, _name_);
```

**Line 98:** Store value for latitude on the object. If latitude is missing or of an incorrect format, default value '55.6405' will be used.

**Line 99:** Store value for longitude on the object. If longitude is missing or of an incorrect format, default value '13.2134' will be used.

**Line 100:** Store value for name on the object. If name is missing or of an incorrect format, default value '<NONAME>' will be used.

**Line 101:** The object `eventTarget` will be used for dispatching and receiving events on this object.

Can be used in the main script using the following construction

```

_SunRiseSetCheck_.eventTarget.onDataAvailable = function(event)
{
    //Process event
}

```

```

103 //Create function for sending an Event
104 this.sendEvent = function(sunrise, sunset, latitude, longitude, date, name)
105 {
106     try
107     {
108         var event = new Event(); //Create new Event object
109         event.initEvent("DataAvailable", true, true);
110
111         event.sunrise = sunrise; //Store sunrise in the event
112         event.sunset = sunset; //Store sunset in the event
113         event.latitude = latitude; //Store latitude in the event
114         event.longitude = longitude; //Store longitude in the event
115         event.date = date; //Store date on the event
116         event.name = name; //Store name on the event
117         event.text = "Check name: " + name + ", Sunrise: " + sunrise + ", sunset: " + sunset + " local time, for
118
119         if(this.eventTarget.willTrigger("DataAvailable")) //Check if EventListener exists
120             this.eventTarget.dispatchEvent(event);
121         else
122             throw new Error("Event listener 'DataAvailable' is missing"); //Throw Error if EventListener is missing.
123     }
124     catch(e)
125     {
126         Debug.Print("scSNMP.SunRiseSetCheck.sendEvent(): " + e);
127     }
128 }

```

**Line 104:** Defines a function `sendEvent()` which will be responsible for sending events on a specific interval.

The event contains the following properties

**sunrise**(string), Local time for sunrise

**sunset**(string), Local time for sunset

**latitude**(number), Latitude value

**longitude**(number), Longitude value

**date**(string), Date value for when times are valid

**name**(string), Name value, if used as input to the object

**text**(string), Formatted text containing all of the values above.

```

130 //Function For getting new data
131 this.getData = function()
132 {
133     try
134     {
135         var name = this.name; //Get name
136         var date = new Date().toLocaleDateString(); //Get todays' date
137         var lat = this.latitude; //Store latitude
138         var lon = this.longitude; //Store longitude
139         var sun = scSunriseSet.calculateSunRiseSet(lat, lon); //Function will return
140         this.sendEvent(sun.rise, sun.set, lat, lon, date, name); //Send event to Ever
141     }
142     catch(e)
143     {
144         Debug.Print("scSNMP.SunRiseSunSetCheck.getData(): " + e);
145     }
146 }

```

**Line 104:** Defines a function `getData()` which will get current times for sunrise and sunset. Will be called cyclically each midnight or manually from the main script.

**Line 140:** Calls function `sendData()`.

```

148 //Create Timer object
149 this.timer = new Timer(1000, false); //Create new Timer object
150 this.timer.parent = this; //Create reference to this object, to get access from Timer object
151 this.timer.onTimeout = function()
152 {
153     try
154     {
155         this.parent.getData(); //Execute getData() function
156         this.start(scExtensions.msLeftUntilMidnight()+60000); //Restart the timer to get an update at midnight
157     }
158     catch(e)
159     {
160         Debug.Print("scSNMP.SunRiseSunSetCheck.timer.onTimeout(): " + e);
161     }
162 }

```

**Line 149:** Define a timer object.

**Line 151:** Function `onTimeout()` will be called at certain intervals. One second after save/restart of Ethisis Server, then again each midnight.

**Line 155:** Call function `getData()`.

**Line 156:** Restart timer.