

SunRiseSet

Syfte

Det här skriptbibliotekets syfte är att hämta lokal tid för soluppgång och solnedgång oavsett position(latitud/longitud) i decimalgrader.

Körkrav

Importerera skriptbiblioteket *scExtensions.zip* under skriptbibliotek i Ethis Admin.
Importerera skriptbiblioteket *scSunRiseSet.zip* under skriptbibliotek i Ethis Admin.
Skapa den globala *Boolean* variabeln, *bSun1*, för att testa användarexempel som beskrivs i detta dokument.

Licenskrav

Skriptbiblioteket fungerar på licensnivå Advanced, Universal nivå 4 eller högre.

Huvudskript med förklaring

Lägg till följande kod i mainskriptet för att testa funktionerna i *scSunRiseSet*.

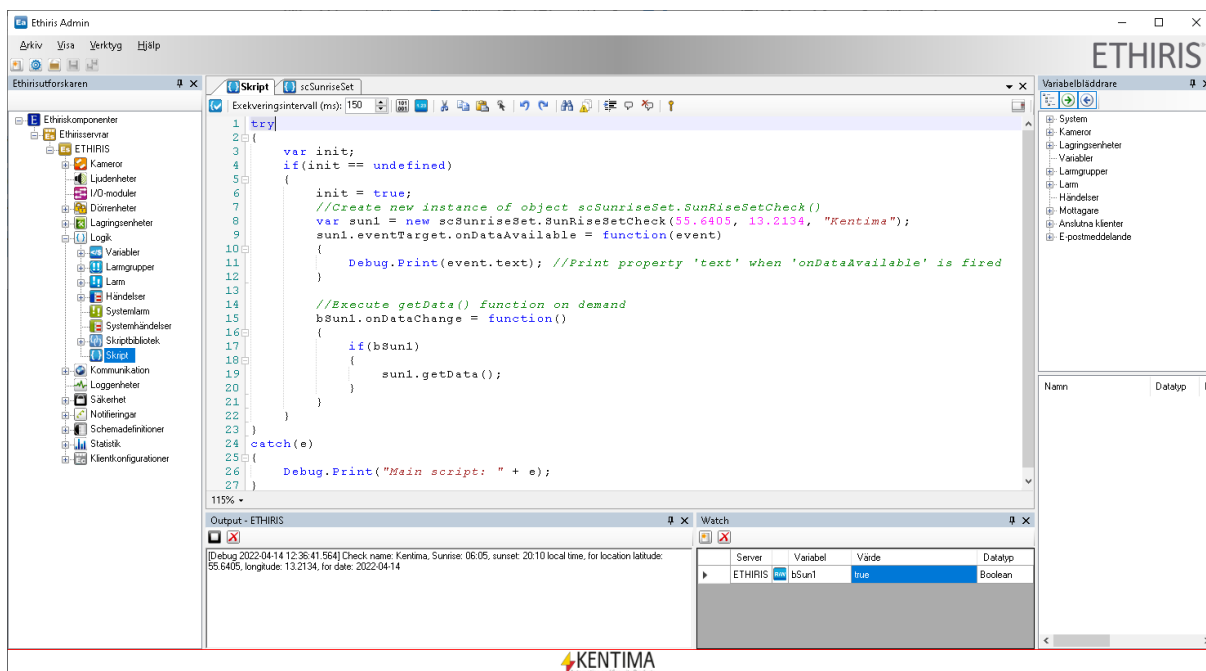
Notera! När *SunRiseSetCheck()* objektet skapas se till att endast göra det en gång, "loopa" inte skapandet av objektet. Som ett exempel kan en *if*-sats användas enligt nedanstående kod.

```
try
{
    var init;
    if(init == undefined)
    {
        init = true;
        //Skapa en instans av scSunriseSet.SunRiseSetCheck()
        var sun1 = new scSunriseSet.SunRiseSetCheck(55.6405,13.2134,"Kentima");
        sun1.getEventTarget().onDataAvailable = function(event)
        {
            Debug.Print(event.text); //Skriv ut egenskap 'text'.
        }
        //Exekvera getData()funktionen på begäran.
        bSun1.onDataChange = function()
        {
            if(bSun1)
            {
                sun1.getData();
            }
        }
    }
}
catch(e)
{
    Debug.Print("Main script: " + e);
}
```

Nu går det att lägga till variabeln *bSun1* i *Watch Panelen*. **Dubbelklicka** på *Value* för *bSun1*, för att sätta värdet till *true*. Då kommer funktionen *getData()* att exekvera. Output kommer att skrivas ut i *Debug Output Panel*.

Funktionen *getData()* triggar ett event som i sin tur kommer att trigga event-funktionen *onDataAvailable*. Eventet innehåller egenskaperna ***sunrise***(sträng), ***sunset***(sträng), ***latitude***(nummer), ***longitude***(nummer), ***date***(sträng), ***name***(sträng) och ***text***(sträng).

I exemplet nedan skrivs egenskapen ***text*** ut med hjälp av *Debug.Print()*.



The screenshot shows the Ethisis Admin interface. The main window is titled "Ethisis Admin" and contains a "Skript" (Script) editor. The script is for "scSunriseSet" and has an execution interval of 150 ms. The script code is as follows:

```
1 try
2 {
3     var init;
4     if(init == undefined)
5     {
6         init = true;
7         //Create new instance of object scSunriseSet.SunRiseSetCheck()
8         var sun1 = new scSunriseSet.SunRiseSetCheck(55.6405, 13.2134, "Kentima");
9         sun1.eventTarget.onDataAvailable = function(event)
10        {
11            Debug.Print(event.text); //Print property 'text' when 'onDataAvailable' is fired
12        }
13
14        //Execute getData() function on demand
15        bSun1.onDataChange = function()
16        {
17            if(bSun1)
18            {
19                sun1.getData();
20            }
21        }
22    }
23 }
24 catch(e)
25 {
26     Debug.Print("Main script: " + e);
27 }
```

The "Watch" panel on the right shows the following data:

Server	Variabel	Värde	Datotyp
ETHIRIS	bSun1	true	Boolean

The "Output" panel at the bottom shows the following debug message:

```
[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
```

Skriptbiblioteket SunriseSet förklarar

```

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 }

```

Rad 10: Verifierar att skriptbiblioteket *scExtensions* är importerat. *scExtensions* är nödvändigt för att kunna köra *SunRiseSet* skriptbiblioteket.

```

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:

```

Rad 16: Definerar en funktion som beräknar tider för soluppgång och solnedgång för en specifierad plats med latitud och longitud som 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;

```

Rad 86: Definerar ett objekt som innehåller tiderna i form av formaterade strängar.

Rad 90: Returnerar ett objekt till "caller funktionen".

```

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

```

Rad 94: Definerar en funktion ("Class" objekt) som håller utkik efter soluppgång och soldnedgång i ett specifikt intervall.

Skapa nya instanser av det här objektet i huvudskriptet enligt nedanstående kod:

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

Rad 98: Lagrar värdet för latitude på objektet. Om latitude saknas eller har ett inkorrekt format så kommer standardvärdet '55.6405' att användas.

Rad 99: Lagrar värdet för longitud på objektet. Om longitud saknas eller har ett inkorrekt format så kommer standardvärdet '13.2134' att användas.

Rad 100: Lagrar värdet för namn på objektet. Om namnet saknas eller har ett inkorrekt format, så kommer standardvärdet '<NONAME>' att användas.

Rad 101: Objektet *eventTarget* användas för avsändande och inkommande event på det här objektet.

Kan användas i huvudskriptet enligt nedanstående kod:

```

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

```

Rad 104: Definerar en funktion *sendEvent()* som är ansvarig för att skicka events i specifika intervall.

Eventet innehar nedanstående egenskaper:

sunrise(sträng), Lokal tid för soluppgång

sunset(sträng), Lokal tid för solnedgång

latitude(nummer), Värde för latitud

longitude(nummer), Värde för longitud

date(sträng), Datumintervall för giltiga datum

name(sträng), Värde för namn om namn används som input för objektet

text(sträng), Formaterad text som innehåller alla ovanstående värden.

```

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 }

```

Rad 104: Definerar en funktion *getData()* som hämtar alla nuvarande tider för soluppgång och solnedgång. Funktionen anropas cykliskt eller manuellt via huvudskriptet.

Rad 140: anropar funktionen *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 }

```

Rad 149: Definerar ett timerobjekt

Rad 151: Funktionen *onTimeout()* anropas i specifika intervall. En sekund efter att Ethisis Server sparas/startas om och en gång varje midnatt.

Rad 155: Anropar funktionen *getData()*.

Rad 156: Startar om timern.