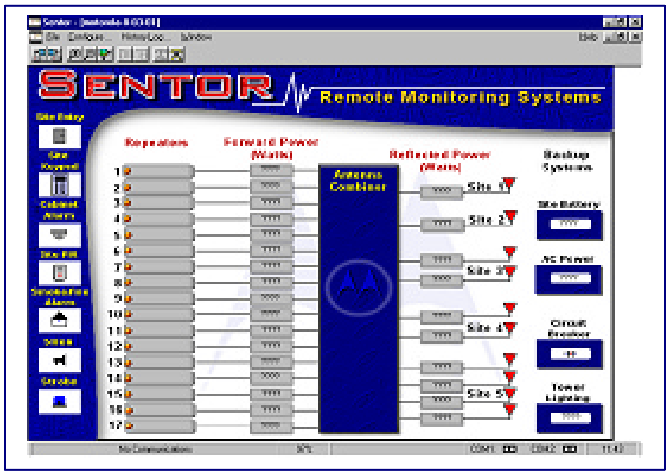
Sentor Remote Communications

Remote base site monitoring and control

Programming Sentor



Contents:

1 Introduction

Although you can control devices directly from a keypad, the real power of Sentor is its ability to be easily programmed to automatically look after your Site. This manual explains how you program Sentor by means of *scenarios*, which are simply collections of instructions.

Even if you have not done any conventional computer programming, you will find that understanding the basics of how to program Sentor is not difficult. As with any new system, you will go through a learning curve and your skill will grow with experience.

Keep Help open

It is a good idea to keep the on-line Help open while you program Sentor. You can then get instant information by clicking on the appropriate tab on the bottom of the screen. Use the Search facility to get information on topics.

Style convention

Text that you see on the screen is shown in a different font, **like this**. Code examples are shown like this.

About Scenarios

As mentioned above, a scenario is a collection of instructions. More properly, we should call it a collection of *lines*; in Sentor parlance a line contains between one and three parts or clauses:

- An optional Condition: 'If this condition is met, then …'
- An optional When: ' … at this time …'
- An Action: ' … do this action'.

We'll look at lines more closely in later sections, but for now you just need to remember that every line tells Sentor to carry out an action, possibly only if some condition is met, and possibly on a certain date/time (or within a period of time or randomly).

Typical Scenarios

Some typical scenarios are:

Site Disarmed	This could disarm the security, perhaps turn on air conditioning, switch on outside lights after dark etc.
Site Armed	Arm the security, switch off all lights, maybe turn off some equipment etc.
Keypad Scenarios	Control what happens when specified keypad inputs are pressed.
Arm Sentor	Set up the security. This scenario would be 'included' in other scenarios such **Site Armed**.

Processing

If you are familiar with common computer languages such as 'C' or BASIC, you may be surprised to learn that processing in Sentor is not sequential. That is, rather than each line being executed in sequence, all lines in a scenario are effectively processed simultaneously. This means that the order in which you put the lines in any scenario is not important because no line is dependent on the one before it.

The action specified in a line is carried out as soon as its condition clause(s) and when clause(s) – if any – become True. However, you may specify a delay before the action actually happens.

Once all the controlling clauses are True and the line is actioned, it will not be actioned again until one or more of the conditions have become False and then True again.

Including, Excluding and Selecting

As you would not want to have to write the same sets of instructions more than once, Sentor give you a mechanism for using any scenario from within other scenarios.

As in the example scenarios above, if your scenario called **Site Armed** contained the Ine ‘Include ‘Arm Sentor’’, it would function in exactly the same way as if you had typed in all the **Arm Sentor** lines. This is a very powerful feature, and those familiar with other computer languages will recognise it as being similar to calling a subroutine or a procedure.

You can **Exclude** a scenario in the same way. This would generally apply where you have included a scenario but you do not want it to be run if some condition is met.

Within one scenario you can have a line that selects another scenario if a condition is met. In that case, the selected scenario takes over completely and the one calling it is terminated.

More information on the use of these instructions will be given later.

2 Programming Philosophy

The great flexibility of Sentor is its major strength but it can be something of a liability. You need to be precise in your instructions and clear about what you want to achieve.

As almost every scenario is based on certain events occurring or conditions being met, you need to think about all the possible combinations of these and test your system to make sure it does what you expect it to do in all circumstances.

Keep it Simple

The simpler you make your scenarios, the more likely they will operate reliably and predictably. This does not stop you from carrying out quite sophisticated operations, but means you should try to break them up into small scenarios that are easy to understand, test and maintain.

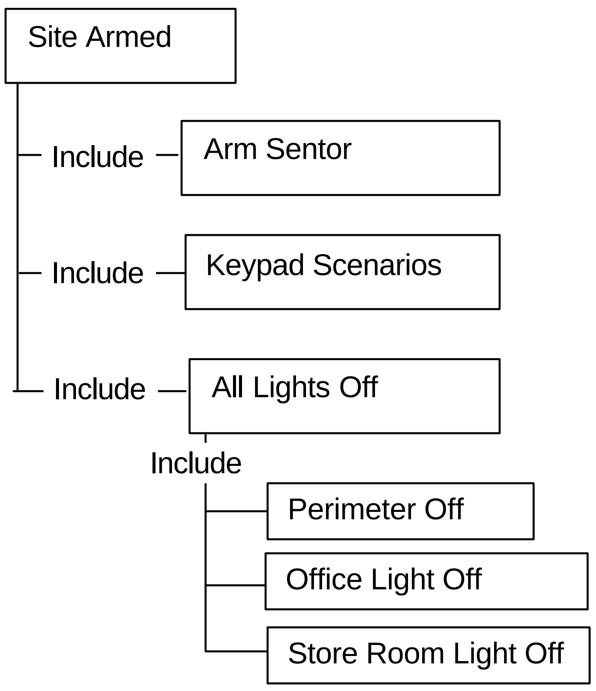
Use Comments

Sentor lets you put comments in your scenarios by starting a line with '…'. Use comments liberally to explain what is going on, so that when you go back to a scenario in the future you will quickly understand what it is doing. More importantly, someone else may need to work with your scenarios and this will help them even more.

Top Down Design

Developing a top down design means that you identify high level scenarios that cover broad circumstances such as **Site Disarmed, Site Armed** etc. Before writing these, you think about other scenarios that can be **Included** in them. At a lower level still, it may be convenient to have more scenarios that can be included in the latter group (this is known as *nesting* scenarios).

The following diagram illustrates the concept.



Although all of the lines involved in **Site Armed** could be combined into one large scenario, it is obviously much easier to understand and test when the different types of activities are in their own scenarios. As well as this, any of the scenarios can be included in other high level scenarios with a single instruction.

Bottom up writing

You start out by outlining a top down design, but then you commence writing from the bottom up. In other words, having identified the lowest level scenarios, you write these first, testing each as you go along. This means that when you go up to the next level, the scenarios you are including have been written and tested. In effect, the lower level scenarios become building blocks for higher level scenarios.

Include is Simpler (and Better) than Select

When you use **Include**, program flow is fairly easy to follow. However, **Select** causes the program to 'jump' to another scenario and terminate the current one. This makes program flow harder to follow and should generally be avoided for the same reason that the instruction Go To (or its equivalent) in other languages is undesirable. The expression 'spaghetti code' arises from the use of Go To.

So wherever possible, use **Include** rather than **Select**.

Use a Single Top Level Scenario

A well programmed Sentor installation using the above guidelines will have a single 'main' or 'always' scenario at the highest level, with other scenarios included within it (and others within them and so on).

When the main scenario runs, changing conditions will determine the actions taken by Sentor at any given time. For example, when you leave to go off site and the site will be unoccupied, you would use a keypad to notify Sentor. One of the included scenarios will observe the keypad input and all the actions necessary to secure the site will take place, probably activating the **Include** of several of the lower level scenarios.

As another example, a smoke detector device may change state on being triggered by smoke. This will be immediately picked up by the relevant scenario which acts to **Include** the fire response scenario, causing the appropriate action (such as sound alarm, activate sprinklers, telephone fire brigade etc.).

3 Writing Scenarios

We now get to the 'nuts and bolts' of how to actually write the lines of code that make up a scenario.

Keep Help available

It is worth having the on-line Help running all the time you are writing code. You can then get immediate information as needed by using the Search facility.

If you have the Infra-red remote control option, be sure to fill in the chart in the to take note when you use remote control commands in your scenarios, so you have a record of what each of the buttons does.

The Sentor Language

Like any other computer language, that used by Sentor has its own syntax (way of constructing the lines) and words that have special meanings to Sentor. Fortunately the language is based on plain English and is not hard to understand.

The versatility of Sentor means that its language is necessarily quite extensive, but luckily you do not have to learn it before you can write real scenarios. As you would have discovered in the Guided Tour, the Sentor **Insert Line Help** facility gives you the full power of the language without making you actually write the code yourself.

A Language Reference chapter on page 22 provides details of syntax and reserved words etc. However, until you are very experienced in Sentor programming you will almost certainly find it much easier to use **Line Insert Help** as described in the following sections.

We use the term *device* to include both actual and virtual devices.

The Scenario Configuration Screen

The first step in working with scenarios is to access the Scenario Configuration Screen:

Select **Configure | Edit Scenario Configuration**.

Depending on how long it is since you last used this menu, you may need to enter your 4-digit password.

You will see a list of scenarios in the left hand pane. Although it may not be immediately apparent, you can make your scenario list hierarchical. That is, you can arrange it so that some items are indented under others to show the major **Include** dependencies.

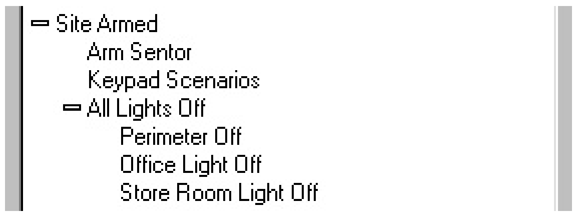
Arranging the List

There is a set of four arrows below the scenario list. When you select a scenario, you can use the Up and Down arrows to change the position in the list.

Indenting

You can use the Left and Right arrows to change the amount of indenting of an item by one or more positions depending on what is above it. A + sign means that an item has indented items below it. Click on the + to show the items and change the + to a – sign.

The figure below shows how the list could be arranged to illustrate the hierarchy shown previously.



```
– Site Armed
      Arm Sentor
      Keypad Scenarios
    – All Lights Off
          Perimeter Off
          Office Light Off
          Store Room Light Off
```

It is important to note that arranging a hierarchy is only for a visual representation; it does not affect the scenario operation.

Adding a Scenario

1. Click the **Add** button to bring up a **Program Scenario** window.
2. In the box at the top, replace the **New Scenario #xx** with the name you want to give the new scenario.
3. Follow the instructions in the next section to write your lines of code.

As we noted before, you could type in your code directly here, but you will certainly want to use the **Line Insert Help** until you are very familiar with the language.

Comments

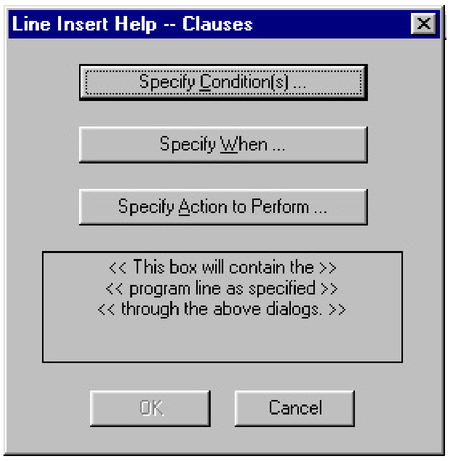
You may want to add a comment to this window before you insert the code, possibly to explain what the scenario does. To do so, put the cursor in the main text box, then type '**…**' followed by your comment. You can add comment lines at any time.

Writing Scenario Code

As mentioned earlier, a line of code in Sentor contains an instruction (an action clause), optionally preceded by a condition that must be met and/or a time when the instruction should be performed. Line Insert Help does all the hard work for you by offering the available choices at each step, so you only need to point and click to generate your code.

We recommend that you read the Language Reference chapter even if you intend to mainly use Line Insert Help. This will give you a better understanding of just what is going on, and there are several useful hints and explanations.

Click the **Line Insert Help** button to bring up the dialog box shown below:



<< This box will contain the >>
<< program line as specified >>
<< through the above dialogs. >>

We will now go through the choices offered by each of the buttons. Remember that the **Condition** and **When** clauses are optional, but you must always have an **Action**.

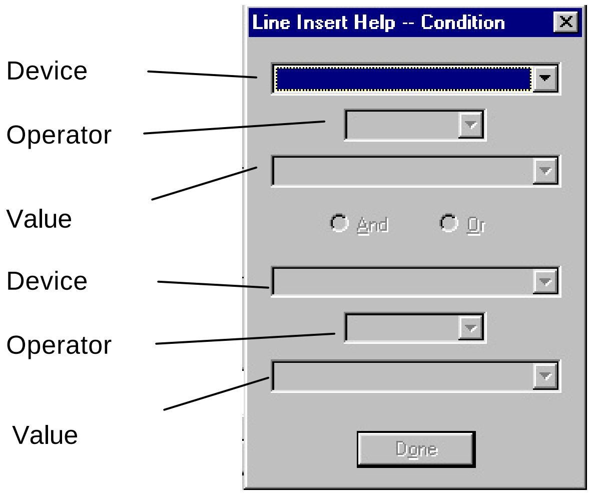
Specify the Condition(s)

A condition clause can have up to four comparisons, although you will rarely need more than two. **Line Insert Help** lets you specify one or two conditions; if you need more you will need to type in the extra code manually.

There are three parts to a condition:

- A device whose value is being tested.
- An operator indicating the type of comparison to perform. Examples are 'is', 'is not', 'matches', 'doesn't match' etc. For numeric tests you can use algebraic operators such as '<' (less than), '>', '=' and so on.
- A value (or device) that you are making the comparison against.

Clicking the **Specify Condition(s)** button brings up a dialog in which you select the conditions under which your action will be carried out, as shown next.



1. Select the device whose condition you want to test.
2. Select an operator. The choices offered will depend on the type of device as Sentor will only let you make a meaningful comparison.
3. Select or type in a value.
4. If you want to add a second condition , click on **And** or **Or** and repeat the above using the lower three drop down lists.
5. Click **Close**

The clause you have just created will now appear in the **Line Insert Help** dialog.

Specify When

You can specify the dates, days and/or time(s) an action should be performed. If there is a condition, that condition must be met as well. In the absence of a **Condition** clause, the action will occur as soon as the **When** clause(s) become True. A couple of simple examples may help here:

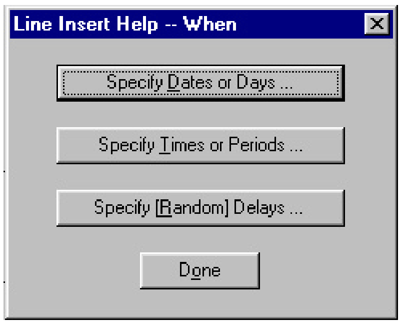
If ‘Door’ is ‘Open’, on Tuesday, then set ‘Site Lights’ to ‘On’

The above means that when the device called ‘Door’ is set to Open, during any Tuesday, the device called ‘Site Lights’ will be set to ‘On’.

On Tuesday, set ‘Site Lights’ to ‘On’

The above means the device called ‘Office Light’ will be set to On at 00:00:00 every Tuesday (which is midnight on Monday).

Clicking on **Specify When** brings up the dialog shown below:


Line Insert Help -- When
Specify Dates or Days ...
Specify Times or Periods ...
Specify [Random] Delays ...
Done

There are three factors you can specify here, as indicated by the three buttons.

Dates or Days

The **Dates or Days** dialog lets you choose a day of the month (or Any), a month (or Any), a year (or Any) and one or more days of the week.

For example, you could specify that the action can occur on any Sunday in April of any year. Or you could leave Day, Month and Year at Any and select Monday to Friday so that the action occurs on all weekdays.

Take care If you choose an impossible day (such as 30 February or a date and day of the week that do not match), the action will never be performed.

Times or Periods

The **Times or Periods** dialog lets you select one of two things:

- Setting a **Between This Time** value only, causes the action to occur at that time (the **And This Time** is automatically set to the same value). Note that you can choose **Any** for a setting. For example, if you set **Hour** to **Any** and **Minute** to 30, the action will occur at 30 minutes after each hour.

Or

- Setting a later time for **And This Time** causes the action to occur once at a random time during the period between the two time settings.

Note that you should not specify both a **Period** and a **Delay** (see next item).

Delay

The **Delay** dialog allows you to specify a delay between the time a condition is met, or a specified date/time occurs, before the action takes place.

Use the left hand buttons to select one of two types of delay:

- **Delay for** means that the delay period starts when other conditions are satisfied, and the action takes place at the end of that time, regardless of whether the conditions have changed in the meantime.
- **Ignore for** means the delay starts when other conditions are satisfied, and is cancelled if the conditions change to not true during the delay period. So the action will occur only if the conditions remain true for the entire delay period.

  A typical example of the use of **Delay** would be to allow (say) five seconds for you to open a door and disable an alarm. If you used **Ignore** in this case, an intruder would avoid setting off the alarm if he happened to close the door within five seconds.

Select one of the four delay times. The first three are as configured for your system, and the last one you specify via the two boxes below, as follows:

- The **Specified Time** sets the delay period.
- Setting **Random Variance** to a value other than zero means the delay time will vary randomly around (before and after) the **Specified Time**. For example, if you specify a time of one hour and a variance of 20 minutes, the delay will vary randomly between 40 minutes and one hour 20 minutes.

Specify Action

When you specify an action you have to tell Sentor what instruction to carry out (the action verb), the device on which to act, and in some cases the result or state you want to achieve.

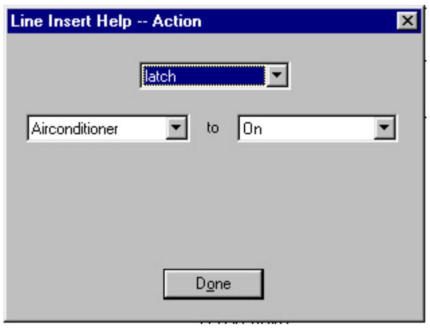
Sentor has a lot of action verbs (see Language Reference). The type of device you can act on depends on the action, and **Line Insert Help** only offers those that make sense for the action specified.

1. Click **Specify Action to Perform** to bring up the Action dialog.
2. From the top drop down list, choose the action you want to perform. The box will change appearance depending on your choice.

We will now describe the main classes of actions and their dialogs. For detailed information on a particular action, search for ‘action’ in the Help system then select the one you are interested in.

Set, Latch

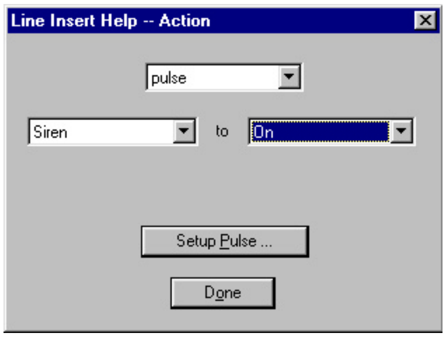
Each of these presents a list of devices. The selected device can then be set to **On, Off** or the value of another device. The verb **set** is included for backward compatibility and should generally be used only for non-digital devices. A **latch** example is shown in the following figure.



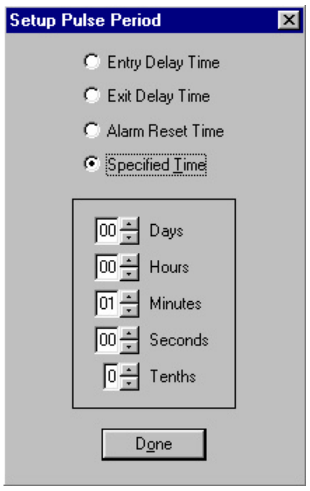
Pulse

A pulse action operates for a specified period. For example, you could pulse a siren on for 30 seconds. If the scenario that initiated a pulse is terminated before the pulsing period ends, the pulse period is not affected, and the device will stay in the same state until the end of the period (unless it is explicitly set by some other action).

Selecting **Pulse** offers a list of digital devices, plus a **Setup Pulse** button as follows:



Clicking on **Setup Pulse** brings up the dialog illustrated below.


Setup Pulse Period

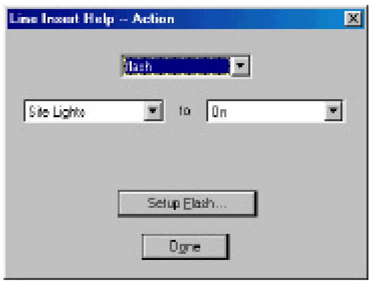
( ) Entry Delay Time
( ) Exit Delay Time
( ) Alarm Reset Time
(●) Specified Time

00 Days
00 Hours
01 Minutes
00 Seconds
0 Tenths

The first three choices offer the times set for the System Parameters. If you changed the names from the defaults, those names will be shown.

Flash

A device that is flashed alternates its state (at specified intervals) until it is explicitly stopped. If you terminate the scenario that initiated flashing, this has no effect on the device, which will continue to flash until stopped by some other action.

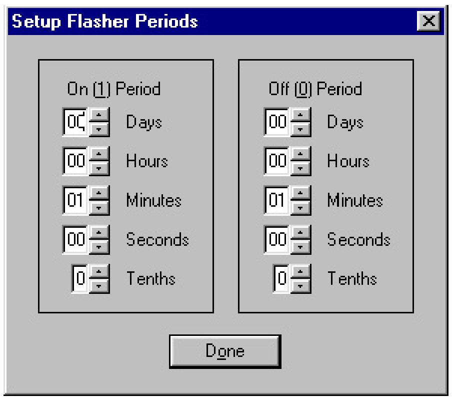

Line Insert Help -- Action

Flash

Site Lights to On

This offers a list of digital devices, plus a **Setup Flash** button as illustrated below.

Clicking on **Setup Flash** brings up the dialog shown next, in which you can set the **On** and **Off** periods.


Setup Flasher Periods

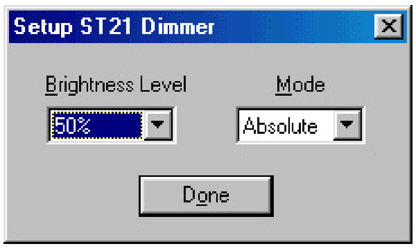
On (1) Period
00 Days
00 Hours
01 Minutes
00 Seconds
0 Tenths

Off (0) Period
00 Days
00 Hours
01 Minutes
00 Seconds
0 Tenths

Dim

This offers a list of devices that can be set to a digital output level (X–10 and C-Bus devices). The dialog box is similar to **Flash**, but there is now a **Setup Dim** button (instead of a **Setup Flash** button) that gives the dialog shown below. You can set brightness level in 5% increments from 5% to 95% for X-10 devices, and **Mode** may be **Absolute** or **Relative**.

Use **Absolute** if it is possible that the brightness has been changed by something other than Sentor. Otherwise, you can use **Relative** to change the brightness relative to its previous level (which will be known if only Sentor changes it).


Setup ST21 Dimmer

Brightness Level: 50%

Mode: Absolute

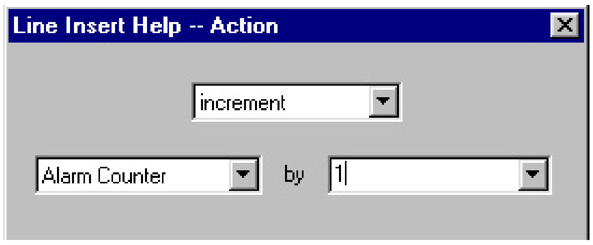
For C-Bus, the brightness levels are in increments of 1/255% (about 0.4%) and you can select one of 16 rates that determine over what period the dimming occurs.

Reset, Enable, Disable, Toggle

Each of these presents a list of devices. The selected device will be reset, enabled, disabled or toggled. There are no other parameters.

Increment, Decrement

Each of these presents a list of counters. The selected counter can be incremented or decremented by a typed-in value as illustrated in below . Alternatively, you can select another device, and the counter will start counting the off to on transitions of that device.


Line Insert Help -- Action

increment

Alarm Counter by 1

Select, Include, Exclude

Each of these presents a list of scenarios. The chosen scenario will be selected, included or excluded as follows:

- **Select** is followed by either a scenario name (in quotes) or the special name "**no scenario**". In either case the current scenario will be terminated (together with any **Included** scenarios). Actions already performed will remain (so devices turned on, flashed or pulsed will stay in that state). However, actions that are pending expiration of a delay period will not happen.
- **Include** causes the lines of the chosen scenario to be performed as if they were part of the current scenario.
- **Exclude** causes the chosen scenario to be terminated in the same way as described above under **Select**.

Stop

This offers a list of counters. The selected counter will be stopped, meaning it will stop counting the transitions of the device it was set to Increment or Decrement on. If the counter is already stopped, this instruction has no effect.

Log, Say, Remark, Message

Each of these provides a text box in which you can type the appropriate string.

Log	adds the text to the History Log.
Say	sends the text to a speech synthesiser.
Remark	adds the text as a remark to the scenario.
Message	displays the text in a message box.

Run

This presents a box in which you can type the program to be run as a pathname. There is also a browse button that operates in the standard Windows fashion so you can browse to the program location without needing to type in the pathname.

As file type associations are understood by this action, you could put in the path for a document (for example) and the corresponding word processor will automatically start with that document.

Maximise, Call 'Sentor', Hangup

These have no other parameters to be set. They maximise the window, call the Sentor GUI or hang up the connection to the Sentor GUI respectively.

Report

This presents a list of devices. The device description and current value will be recorded in the History Log when the action occurs.

Updating the Scenario

When you have finished writing your scenario, you need to send it to the Controller.

Click on **Update**, (If you did any manual editing, check that Sentor has interpreted your input correctly) then click **Yes**.

You will now see the **Scenario Configuration** window again, and your scenario will be stored. It will not be selected automatically (whatever scenario(s) are currently running will stay that way).

Editing a Scenario

To edit an existing scenario, select its name in the **Scenario Configuration** window and click the **Edit** button at the bottom.

The **Program Scenario** window will appear, with the lines of code that comprise the scenario. You can put the cursor where you want to make any changes and either type in code directly or use **Line Insert Help**.

When you have finished, update the scenario as described above.

Running and Managing Scenarios

You will want to use these commands while you are developing your scenarios so you can test them. The Select, Include and Exclude actions are as described on page 14.

Running a Scenario

The currently running scenario is shown at the top of the screen

You can run a scenario by selecting it in the **Scenario Configuration** window and clicking the **Select** button at the bottom. The scenario will then run until you **Exclude** it (see below) or select another one.

Including a Scenario

With a scenario running, you can include another scenario by selecting it as above and clicking the **Include** button below. The scenario will run until you **Exclude** it or **Select another scenario**.

Note that including another scenario will not affect any scenarios currently included.

Excluding a Scenario

To exclude or end a scenario, select it and click on **Exclude**.

Deleting a Scenario

To delete a scenario, select it and click on **Delete**. A dialog will ask if you are sure.

Changing a Password on a Keypad

If you want to allow users to change a password via a keypad, you can take advantage of a built-in facility, together with a scenario that you write.

On an LCD keypad, a user must enter the sequence:

On On <Old password> <New password> <New password>

On an LED keypad, start with * * or # # instead of **On On**.

The currently active scenario must have a line starting with one of the following, where **My Keypad** is the name you assigned to keypads:

If ‘My Keypad’ changes ‘Main Password’, …
If ‘My Keypad’ doesn’t change ‘Main Password’, …
If ‘My Keypad’ changes ‘Any Password’, …
If ‘My Keypad’ doesn’t change ‘Main Password’, …

The term ‘Main Password’ can be replaced by a particular password name. Using ‘Any Password’ means that any password can be changed.

The change will only happen if the sequence is entered correctly (and the two entries of the new password match), and a currently included (i.e. active) scenario has a suitable line as shown.

You would generally follow the condition clause with something like a message to tell the user that the password has been changed (by pulsing the appropriate message to On for a few seconds).

If the user makes an error (first password does not exist or the two new password entries do not match), this would make true a condition that started with “ If ‘My Keypad’ doesn’t change …”. You would probably want to show a suitable message to tell the user that the password has not been changed in that case.

There is a further facility here that lets you nominate a particular keypad for password changing. It is not accessible via Line Insert Help, so you need to manually insert a bit of code. You can have a condition like:

If ‘My Keypad @ Place’ changes ‘Main Password’, …

‘Place’ will be what you entered for the Location of the keypad. This only applies to LCD keypads.

Scenario Examples

These scenarios demonstrate some of the techniques and principles described.

More examples will be made available on our web site at http://sentor.cc

Basic Security and Utilities

This group of scenarios illustrates a way of controlling basic security, plus a few utility functions.

Definitions

The following device definitions are assumed:

‘Main Door’	A digital input from a reed switch.
‘Site PIR’	A digital input from a PIR detector.
‘Panic Button’	A digital input from a pushbutton.
‘Siren’	Siren output to the Horn speaker terminals.
‘Entry Lights’	X-10 controlled lights.
‘Office Lights’	X-10 controlled lights.
‘Store Lights’	X-10 controlled lights.
‘Sound’	Stereo controlled via infra-red.
‘Air con’	Air conditioner controlled via X-10.
‘Arming’	Keypad message (Sentor Arming)
‘Disarming’	Keypad message (Sentor Disarming)
‘Intrusion Alarm’	Monitoring message.
‘Panic Alarm’	Monitoring message
‘Light Flag’	General flag.
‘Away Flag’	General flag.
‘Alarms’	Alarm counter.

Scenarios

Wonderful Sentor

… This is the top level scenario

Include ‘Utilities’

… Turn on security when we leave

If ‘LCD Keypad’ matches ‘Our password’ and ‘Away Flag’ is Off’, delay for ‘Exit Delay Time’, then include ‘Away’.

If ‘LCD Keypad’ matches ‘Our password’ and ‘Away Flag’ is ‘Off’, pulse 'Arming' to 'On' for 00:05

… Turn off security when we get to site

If ‘LCD Keypad’ matches ‘Our password’ and ‘Away Flag’ is ‘On’ then include ‘Disarm’.

Away

Log ‘Site Armed Included’

If ‘Site PIR’ is not ‘Off’ and ‘Alarm Counter’ < 1 Counts, delay for ‘Entry Delay Time’ then include ‘Intrusion’.

If ‘Main Door’ is ‘Open’ and ‘Alarm Counter’ < 1 Counts, delay for ‘Entry Delay Time’ then include ‘Intrusion’.

Disarm

Log ‘Disarm Included’

Pulse ‘Disarming’ for ‘Entry Delay Time’.

If ‘Panic Flag’ is ‘On, then reset ‘Panic Flag’.

Set ‘Away Flag’ to ‘Off’.

Set 'Alarm Counter' to 0 Counts.

…

Reset ‘Intrusion Alarm’.

Reset ‘Panic Alarm’.

Reset ‘Siren’.

Reset ‘Notify Group’.

…

Reset ‘Entry Lights’.

…

Exclude Away

Exclude 'Intrusion'

Exclude 'Panic'

Exclude 'Disarm'

Panic

Log ‘Panic Included’.

Set ‘Away Flag’ to ‘On’.

Flash ‘Entry Lights’ to ‘On’ for 00:05 on and 00:03 off.

Set ‘Panic Alarm’ to ‘Active’.

Increment ‘Alarm Counter’ by 1 Counts.

…

Exclude ‘Panic’.

Intrusion

Log ‘Intrusion Included’.

Set ‘Away Flag’ to ‘On’.

Pulse ‘Siren’ to ‘On’ for ‘Alarm Reset Time’.

Set ‘Intrusion Alarm’ to ‘Active’.

Include ‘Lights On’

Increment ‘Alarm Counter’ by 1 Counts.

…

Exclude ‘Intrusion'

Lights On

Latch ‘Site Lights’ to ‘On’.

Latch ‘Office Lights’ to ‘On’.

Latch ‘Store Lights’ to ‘On’.

…

Exclude ‘Lights On’.

Lights Off

Reset ‘Site Lights’.

Reset ‘Office Lights’’.

Reset ‘Store Lights’.

…

Exclude ‘Lights Off’.

Utilities

Log ‘Utilities Included’.

…Keypad control of some devices

If ‘LCD Keypad’ matches ‘3 On’, then latch ‘Air Con’ to ‘On.

If ‘LCD Keypad’ matches ‘3 Off’, then set ‘Air Con’ to ‘Off’.

If ‘LCD Keypad’ matches ‘4 On’, then latch ‘Sound’ to ‘On.

If ‘LCD Keypad’ matches ‘4 Off’, then set ‘Sound’ to ‘Off’.

… Panic button actions

If ‘Panic Button’ is ‘On’ and ‘Light Flag’ is ‘Off’, delay for 00:01 then include ‘Lights On’.

If ‘Panic Button’ is ‘On’ and ‘Light Flag’ is ‘On’, delay for 00:01 then include ‘Lights Off’.

If ‘Panic Button’ is ‘On’, then include ‘Panic’.

Site Entry

Assume you have a remote site, with access via a door. You want an alarm to sound if an unauthorised person goes into the site area and an authorised person is not present. Authorised people can override the alarm via a fingerprint scanner.

If the alarm sounds, it should continue until 30 seconds after the door has been closed. An authorised person should be able scan their print before opening the door, which will disable the alarm for 15 seconds to allow entry. An authroised person should also be able to stop a sounding alarm by scanning their print.

The following group of scenarios assume that you have set up:

- A digital input device called ‘Site Door’, connected to a switch that detects when the door is opened. It is configured to report the state as ‘Open’ (=1) or ‘Closed’ (=0).

- A digital input device called ‘Print Scanner, wired to a fingerprint scanner and configured to report ‘Off’ or ‘Scanned’.
- A flag called ‘Site door disabled’.
- A speaker connected to the Horn output.

You can achieve the required results with a scenario called ‘Monitor Site Entry’, which has two other scenarios included as described below.

MONITOR SITE ENTRY

Include 'Sound the Site Alarm'.

Include 'Disable Site Alarm'.

SOUND THE SITE ALARM

If ‘Site Door’ is 'Open', then set 'Siren 1' to 'On'.

If 'Site Door' is 'Closed', delay for 00:30, then set ' Siren 1' to 'Off'.

DISABLE SITE ALARM

If 'Print Scanner’ is 'Scanned', then pulse 'Site door disabled' to 'On' for 00:15.

If 'Print Scanner’ is 'Scanned', then reset 'Siren 1'.

If ‘Site door disabled' is 'On', then exclude 'Sound the Site Alarm'.

If 'Site door disabled' is 'Off', then include 'Sound the Site Alarm'.

Note that the use of a flag (Site door disabled) allows an adult to press the button again before the 15 seconds has expired, thus extending the siren disable for a further 15 seconds. This illustrates how an event can be triggered and then re-triggered.

Battery Charging

The backup battery (if fitted) is automatically ‘float’ charged while AC power is available. This keeps the battery in good condition, but if the controller runs for any time on backup power, you need to boost charge the battery for quick recovery when AC power is restored.

The following scenario would usually be included in your ‘Always’ scenario. After a power failure, it boost charges the battery for two hours in every 24 hours until the battery test indicates full charge.

BATTERY CONTROL

If ‘A.C. Power’ is ‘Lost’ then reset ‘Battery Boost’.

If ‘A.C. Power’ is ‘Okay’ and ‘Battery Power’ is ‘Lost’, then flash ‘Battery Boost’ to ‘Enabled’ for 02:00:00 on and 22:00:00 off.

If ‘Battery Power’ is ‘Okay’ and ‘Battery Boost’ is ‘Enabled’, ignore for 10:00, then reset ‘Battery Boost’.

4 Language Reference

This chapter defines the Sentor programming language. You will need to follow the syntax detailed here if you type in lines of code directly (rather than using **Line Insert Help**).

Error checking

Sentor checks the syntax of your code when you click the **Update** button and will indicate an error if you make a mistake such as specifying an unknown action, using an invalid operator or referring to a device that has not been configured.

Lines are made up of clauses, which in turn comprise keywords (that have special meanings to Sentor), device names and possibly values. In addition, you can include punctuation and other words to make the code clearer – Sentor will generally ignore these.

Abbreviations

In many cases you can use abbreviations (such as Jan for January etc.) When you press **Update**, Sentor will expand the abbreviations and display its interpretation for you to confirm.

When you enter a line manually in the editing window, the clauses can be in any order. Sentor will rearrange them in the standard order when you click on **Update**.

Condition Clause

A condition clause is optional (you can enter it anywhere in a line, but it will be displayed at the beginning of the line when Sentor shows it for confirmation). It must begin with the word **if** and can be followed by one to four conditions separated by **and** or **or**. If you use both **and** and **or** in a condition clause, the **and** takes precedence. In the following example, the condition clause will be true if both A and B are true or both C and D are true.

If A and B or C and D

The above could be represented more clearly as follows, and Sentor would still interpret it correctly:

If (A and B) or (C and D)

Each of the conditions within a clause comprises a *device identifier* (the name you called it during configuration), a comparison *operator* and a comparison *value* (which in most cases will be a device identifier).

Enclose device identifiers within single quotes.

There are many valid operators, and the ones you can use in a particular condition depend on the type of device. Examples are **is, is not, matches, doesn't match, >, <, <=, >=**. Sentor accepts additional alternatives, so feel free to use what seems reasonable because Sentor will let you know what is not acceptable.

Values other than numerics should be enclosed in single quotes.

Keypad comparison values should be **'Any Password'** or a specific password name or a command string such as '**123 On**'. Use of a password number instead of a name is not recommended because Sentor could not distinguish between configuration password numbers and access password numbers.

You should express a digital value as the text description used for setting up the device, e.g. **'On', 'Open'**, **'Closed'** etc. Avoid using numeric values because their meaning is not always obvious.

The 'value' part may also be another device reference (in single quotes). For example, you could test to see if a light is in the same state as a switch. Although this facility is not often used, it can be very powerful when you need it.

Example condition clauses follow:

If 'Keypad' matches 'Any Password'	If the keypad (called **Keypad)** matches any of the defined passwords.
If 'Room temp' < 22 **°**C	If the room temperature is less than 22 °C. (Sentor adds the °C symbol).
If 'Site PIR' is 'On'	If the sensor called **Site PIR** is on – that is, has detected motion.

When Clauses

There are five clauses that can specify when an action is to take place. These are optional.

It is important to understand the two main ways in which the **Date, Day of Week** and **Time** clauses will act in a scenario:

- If there is no other modifying condition, the action will occur as soon as the specified date, day and/or time occurs. So a line that is simply: On Wednesday set 'This Light' to 'On' will cause the light to be turned on at midnight every Wednesday morning.
- If there is a modifying condition, the action will occur when that condition is true, provided the date, day and/or or time clause is also true (or vice versa). So the line:

  If 'HW Flag' is 'Off' on Wednesday, set 'This Light' to 'On' will cause the lamp to be turned on if **HW Flag** changes from **On** to **Off** at any time during Wednesday. Alternatively, if **HW Flag** is turned off on Monday (say), the light will be turned on at 00:00:00 on Wednesday.

It is helpful to look at these as comprising an enabling condition and a triggering condition, and which is which depends on what happens.

In the first case, the date is the enabling condition – so as soon as it is Wednesday, the triggering condition (**HW Flag** changing to **Off**) can cause the lamp to turn on.

In the second case, the **HW Flag** changing to **Off** is the enabling condition, so the start of Wednesday becomes the triggering condition.

In the examples given later, a date, time etc. is shown as True for a certain period when an * is used to mean **any**. Bear in mind that the period applies only if there are other conditions that determine the action. Of course this is generally the case; if you want something to occur at a specific time you would specify that time without using **any**.

Date Clause

This defines a date on which the action should occur. If it is the first (or only) **When** clause, it should start with the keyword **on**, or you can use **at**. A date clause may be preceded by a **Day of Week** or **Time** clause, in which case you can omit the starting keyword or replace it with a comma.

A date is expressed as **d/m/y** or **d-m-y**. A month can be specified as a number, name or abbreviation, and the year may contain two or four digits.

Day, month and year can be specified as *, meaning **any**.

If you make the year **any**, the clause will be True for the specified month and day every year.

If you make the day **any**, the clause will be True for every day of the specified month and year.

Examples follow:

On */Jan/*	True for all dates in January of any year.
On */*/00	True for all dates in 2000.
On 21/*/*	True for 21st of any month of any year.

Note that the following two additional **When** clauses can be used with (or without) the Date specification.

Day of Week Clause

This is simply one or more names of days of the week, separated by or. If it is the first (or only) **When** clause it must start with **on** or **at**. If you specify all seven days, this is the same as not specifying a **Day of Week** clause.

Examples are:

On Wednesday	True every Wednesday .
On Saturday or Sunday	True every weekend.
On */Feb/*, Tuesday	True every Tuesday in February of any year.

Time Clause

A **Time** clause specifies the time at (or during) which an action occurs. If it is the first (or only) **When** clause it must start with on or at.

Specify time in the form **hh:mm:ss,** where the hour uses a 24-hour clock. You can specify * for one or more of the three parts, meaning **any**. Examples:

At 15:*:*	True between 3:00:00 pm and 3:59:59 pm
At * :15:*	True from 15 minutes past the hour to the second preceding 16 minutes past the hour.
At 12:00:00	True precisely at noon each day.

Between Clause

A **Between** clause specifies an interval within which an action can occur (once) at a random time. Define it as two times, each in the same form as a **Time** clause, separated by the keyword **and**.

You can use an * as a 'wild card'. If it follows a specific number it is taken to mean zero. Both the time specifications must be in the same form (if you have a * for hour in one, it must also be in the other). Here are some examples:

Between 17:00:00 and 18:00:00	means between 5 pm and 6 pm.
Between *:50:* and *:10:*	means between 10 minutes before and 10 minutes after each hour.

Where there is also a condition clause, that determines whether the action will occur during the interval, but only if the condition is **True** at the (random) instant that the **Between** clause is **True**. Take care to interpret this correctly. For example:

> If 'switch' is 'on' between 10:00:00 and 12:00:00 then …

In the above case, the action will occur at some random time between 10:00:00 and 12:00:00 only if the switch is on at that particular moment.

A typical example of how you might use the between clause is to turn on the site lights at some random time between 19:00 and 21:00 when you are away, so the site looks occupied:

> Between 19:00:00 and 21:00:00, set 'Site lights' to 'On'

Avoid the trap of trying to use **between** in the wrong way. For example, if you want an alarm to sound if a PIR is activated during the night, you may be tempted to try something like:

> Between 20:00:00 and 6:00:00 if 'PIR 1' is 'Active', then …

The above will not work because there is only a random instant during the period that the PIR activation would trigger the action.

To achieve the aim, you could define a flag that is set to On between 20:00 and 6:00:

> If 'Current Time = 20:00, then set 'Night Time' to 'On'
>
> If 'Current Time = 6:00, then set 'Night Time' to 'Off'

Then use the following line:

> If 'Night Time' is 'On' and 'PIR 1' is 'Active', then …

Delay Clause

A **Delay** clause specifies a delay before an action is performed (after any conditions are met). The clause starts with one of two keywords:

- **Delay** means apply the delay from when the conditions are true, and perform the action at the end of the period even if the condition(s) change during that time.
- **Ignore** means apply the delay from when the conditions are true, but perform the action *only* if the conditions remain true for the whole of the period. If the conditions become false, the operation is cancelled. This case includes both **if** conditions and **date/time/day of week** conditions.

You can specify the period to be one of three configured delays (using either **delay** or **ignore**):

delay for ‘entry delay time’

delay for ‘exit delay time’

delay for ‘alarm reset time’

The delays are as set for the System Parameters, and if you changed the default names, those you specified must be used.

Alternatively, you can specify an actual delay period in days, hours, minutes, seconds and tenths of a second in the form:

dd:hh:mm:ss.t

Note that days, hours and tenths are optional, but minutes and seconds must be specified. The maximum delay you can use is about 19 days.

You can optionally follow the above with the sub-clause **vary by** followed by a time specification in the same form. This randomly adjusts the delay by plus and minus the amount specified.

Examples:

ignore for 5:00	delay for 5 minutes (condition must remain True)
delay for 5:15, vary by 0:15	delay between 5 and 5 1/2 minutes (condition may change).
ignore for 1:00, vary by 2:00	delay between 0 and 3 minutes (condition must remain True).

You cannot combine a **between** cause with a delay or **ignore clause**. The reason is that a **between** clause is actually implemented as a **delay, vary by** clause.

Action Clause

An Action clause begins with one of the verbs listed below, generally followed by the name of the device to be acted on and in some cases the value to be applied. The actual form depends on the verb; to see more details, search for the verb in the Help system.

set	Set an output to on, off or other appropriate value (if available).
reset	Reset an output to its ‘0’ state, usually called Off.

latch	Latch (don't pulse, dim or flash) a digital output to on (usually) or off.
toggle	Toggle a digital output to on if it was off or off if it was on.
pulse	Pulse (don't latch, dim or flash) a digital output temporarily to on (usually) for a specified period, or off.
flash	Flash (don't latch, dim or pulse) a digital output on and off (usually) or off.
dim	Dim (don't latch, pulse or flash) a digital output to partially on (usually) or off.
increment	Increment counter based on a digital "device" or adjust a counter.
decrement	Decrement a counter based on a digital "device" or adjust a counter.
stop	Stop a counter from incrementing or decrementing based on a digital "device".
enable	Enable a device.
disable	Disable a device.
calculate	Add, subtract, multiply or divide two analog values.
log	Log a message to the History Log.
report	Report the current status of a device to the History Log.
select	Select a scenario to be activated, terminating the running scenario. If it is the same scenario, all processes will be stopped and the scenario re-started.
include	Include a scenario within the current scenario processing.
exclude	Exclude a scenario from the current scenario processing.
say	Say a text string via a suitable speech synthesiser.
run	Run another Windows program.
remark	Add a remark or comment to a scenario.
maximise	Maximise the Sentor window.
call 'Sentor'	Establish communications with Sentor (on a PC)
hangup	Terminate communications with Sentor.
message	Display text in a message box.

Index