OWconfig Database 3

In This Chapter

To provide for maximum flexibility in adding software modules, your module is dynamically loaded as a shared object at run time. The list of loadable objects (modules) is maintained in a system configuration database file called `OWconfig`. `OWconfig` must be edited using scripts to add entries for modules that will be used with Calibrator Tool.

By default, Calibrator Tool reads `OWconfig` from the `/usr/openwin/server/etc` directory. If it does not find the `OWconfig` file or the modules in it that it is looking for, it reads the `OWconfig` file from `/etc/openwin/server/etc`.

This chapter describes `OWConfig` and its format. The chapter explains how you can create entries for the calibration modules you want to add to this file. The chapter also describes how the server uses the information in `OWconfig` entries to locate a dynamically loadable module.

Note – `OWconfig` is a database for various dynamically loadable objects. It not only can contain entries for calibration but also can include entries for other extensions and X Window System modules. The guidelines in this chapter pertain specifically to modules that interact with Calibrator Tool.

3

OWconfig Entry

Code Example 3-1 shows an `OWconfig` text file entry for two dynamically loadable calibration modules.

You create a text file that contains an `OWconfig` entry for your dynamically loadable module(s). Then you execute a script to append your entry to the system `OWconfig` file. You also can remove your entry with a script. This chapter describes both of these procedures and provides templates you can use for adding and removing entries.

Code Example 3-1 `OWconfig` File Entry

```
# Start SUNWkcsrt
################################################
# This is database for the calibrator tool
################################################

# Kcs calibrate class and the name of the class.

# Module that contains handlers for Kodak ColorSense

class="KCMS_CALIBRATE" name="colorsense"
    kcmsCalDeviceType="monitor"
    kcmsCalLoadableModule="kcmscSUNWcolorsense.so.1";

# Module that contains handlers for the XSolarisVisualGamma

class="KCMS_CALIBRATE" name="XSolarisVisualGamma"
    kcmsCalDeviceType="monitor"
    kcmsCalLoadableModule="kcmscSUNWgamma.so.1";

# End SUNWkcsrt
```

Formatting Guidelines

`OWconfig` has some basic formatting guidelines you should be aware of. These are described below.

Any characters following the pound character (#) through the end of a line are treated as a comment and are disregarded when the file is read.

Quotation marks around value strings are required only if the string contains delimiters such as white space or a semicolon (;).

The back slash character (\) can be used as an escape character. For example, \" is used to include the double quotation mark character as part of a string value.

Parsing routines strip the quotation marks surrounding string values and pass the string only to the underlying software. The parsing software treats all values as strings; interpretation of the string value is up to the module.

The strings 'Start' and 'End' indicate the beginning and end of the database definitions for a set of loadable modules. In Code Example 3-1, for example, the string 'SUNWkcsrt' identifies the Kodak Color Management System (kcs) loadable modules. The contents of the string you use for your entry are up to you. You will use the string in a script file to append or delete your entry from the `OWconfig` file. (See "Using Script Files" on page 16, for details.)

Between the 'Start' and 'End' strings are one or more loadable module 'resource' entries for calibration. Each calibration module description consists of a set of four attribute=value pairs separated by white space (including tabs, spaces, and new line characters) and terminated by a semicolon.

3

Table 3-1 describes the meaning of each attribute=value pair.

Table 3-1 Calibration Module Attributes and Meanings

Attribute=	Meaning
class=	This value string is always KCMS_CALIBRATE, which uniquely identifies the class of loadable modules that interact with Calibrator Tool.
name=	This value is the name of the loadable module. It uniquely identifies the instance of the class object in the `OWconfig` file and must be different for each loadable module. This name string will appear in the Calibrator Tool Calibrator:Devices window. (See Figure 2-2 on page 10 in Chapter 2, "Calibrating A Monitor. ")
kcmsCalDeviceType=	This value is accessed when the end user chooses a particular device type to calibrate. It identifies the type of the device and is used to create a list of loadable modules for this device. The names of all the modules of the specified device type are displayed in the Calibrator:Devices window. (See Figure 2-1 on page 8 in Chapter 2.) There can be several monitor device type entries. Each is uniquely identified by the name= value.
kcmsCalLoadableModule=	This value is the name of the shared object that will be loaded when the end user clicks on Load. It has all the information necessary to start its own application to measure the monitor data. The module is expected to be installed in the `/usr/openwin/etc/devhandlers/` directory.

Using Script Files

Before your module can be loaded, you need to add an entry for it in `OWconfig`. To add (or delete) `OWconfig` entries, you create an executable script file.

Appending an Entry

You add an entry to `OWconfig` by appending the entry to the file. The following procedure illustrates this process.

1. **Create an entry.**
   Use Code Example 3-1 on page 14 as a template to create your entry, filling in values for your module. Replace the string associated with ‘Start’ and ‘End’ (‘SUNWkcsrt’) with a string of your choice. Code Example 3-2 is exactly the same as Code Example 3-1 except ‘Start’ and ‘End’ are associated with the string ‘Morrismodule’.

2. **Save your entry.**
   Code Example 3-2 is saved locally as `OWconfig`.

Code Example 3-2 Using the `OWconfig` File Entry as a Template

```
# Start Morrismodule
################################################
# This is database for the calibrator tool
################################################

# Kcs calibrate class and the name of the class.

# Module that contains handlers for Kodak ColorSense

class="KCMS_CALIBRATE" name="colorsense"
    kcmsCalDeviceType="monitor"
    kcmsCalLoadableModule="kcmscSUNWcolorsense.so.1";

# Module that contains handlers for the XSolarisVisualGamma

class="KCMS_CALIBRATE" name="XSolarisVisualGamma"
    kcmsCalDeviceType="monitor"
    kcmsCalLoadableModule="kcmscSUNWgamma.so.1";

# End Morrismodule
```

3. **Create a script file to append your entry.**
   Use the script file shown in Code Example 3-3 as a template. Note the line associating ‘Start’ and ‘End’ with ‘Morrismodule’. It is repeated below:

   ```
   sed -e "/# Start Morrismodule/,/# End Morrismodule/d" \
   ```

   You need to replace this string with the one for your own entry. Then save the script file with a name (for example `ins.OWconfig)` and use the `chmod(1)` command to make it executable.

Code Example 3-3 Script for Appending an Entry

```
#! /bin/ksh
#
# Installation script for the OWconfig class
# If an OWconfig file existed, remove any entry belonging to
# this package, and append a new entry
#
echo $1
echo $2
chmod 644 $2
    if [ -r $2 ]
    then
    #An OWconfig file already exists
    if [ -w $2 ]
    then
        #It's editable by this script, edit it
        cp $2 /tmp/$$OWconfig || exit 2
        sed -e "/# Start Morrismodule/,/# End Morrismodule/d" \
        /tmp/$$OWconfig > $2 || exit 2
        cat $1 >> $2 || exit 2
        rm -f /tmp/$$OWconfig
    else
        #An OWconfig file exists that's not editable
        exit 2
    fi
    else
    #An OWconfig file was not present
    cat $1 >> $2 || exit 2
    fi
chmod 444 $2
exit 0
```

4. **Execute the script file.**
   First you need to become a superuser. Code Example 3-4 illustrates how to do this. The executable script file `ins.OWconfig` takes two arguments: the relative path to the local text file entry (`OWconfig`) and the full path to the system `OWconfig` file (`usr/openwin/server/etc/OWconfig`).

As the script file executes, it displays the values (`OWconfig` and `/usr/openwin/server/etc/OWconfig`) of the two arguments passed to it.

Code Example 3-4 Appending An Entry

```
morris-120% su
Password:
# ./ins.OWconfig OWconfig /usr/openwin/server/etc/OWconfig
OWconfig
/usr/openwin/server/etc/OWconfig
# ^D
```

Removing An Entry

To remove the entry inserted by the above procedure, use the following procedure.

1. **Use the script in Code Example 3-5 as a template.**
   Replace the string 'Morrismodule' in Code Example 3-5 with the string for the entry you want to remove. Save the script with a name such as `rm.OWconfig`, and make the file executable.

Code Example 3-5 Script For Removing an Entry

```
#! /bin/ksh
#
# Removal script for the OWconfig class
# Remove any extries that belong to this module
# Delete the file if it's empty
#
echo $1
chmod 644 $1
    sed -e "/# Start Morrismodule/,/# End Morrismodule/d" $1 > \
    /tmp/$$OWconfig || exit 2
    if [ -s /tmp/$$OWconfig ]
    then
    mv /tmp/$$OWconfig $1 || exit 2
    else
    rm $1 || exit 2
    fi
chmod 444 $1
exit 0
```

2. **Execute the script.**
   First you need to be a superuser. Code Example 3-6 illustrates how to so this. In this case, the script file `rm.OWconfig` takes one argument: the path to the system `OWconfig` file. As the script file executes, it displays the value (`/usr/openwin/server/etc/OWconfig`) of the single argument passed to it.

Code Example 3-6 Removing an Entry

```
morris-125% su
Password:
# ./rm.OWconfig /usr/openwin/server/etc/OWconfig
/usr/openwin/server/etc/OWconfig
# ^D
```

Locating and Opening the Module

Calibrator Tool supplies the `OWconfig` values as arguments to the `OWconfig` library routines. Using the values, the `OWconfig` routines find and dynamically open the loadable module. Then, using `dlsym(3X)`, they execute the module functions.

3