The .rc file, such as .mwmrc for the Motif Window Manager, is an important configuration location. The MWM resource specifies this file location:

```
Mwm*configFile: /<home directory>/.mwmrc
```

3.4.2.4 Server-related Resource Files

Resources for server-related settings are typically in the user’s home directory in a file called .Xdefaults. This file can contain resources for all clients which run on the ViewStation server, regardless of where the clients are executed.

3.4.2.5 Host-related Resource Files

Host-related resources give you a way to make configuration adjustments based on the host executing the application. The resource file in this case is in the user’s home directory with the name .Xdefaults-<hostname>.

3.4.2.6 Resource Specification from the Command Line

You can set a resource from the command line using the -xrm argument. This is often useful for test purposes. For example:

```
xterm -xrm ‘*background: LightSteelBlue’ &
```

3.4.2.7 ViewStation Local Client Resources

The HDSperx clients have resources that can be set in the normal way, typically in the .Xdefaults file. In addition, there are special client attributes which are not conventional resources, but special properties that are unique to ViewStation operations; these resources are set in the ViewStation’s clients.config file.

These attributes are things like paths and environments, as well as password protection, memory requirements, etc. These are discussed in detail in Section 2.1.5.

3.4.3 Loading Resource Files with Xrdb

Because there are so many application programs and so many places for resource files, the automatic reading and setting of resources for applications can be an uncertain process. You should verify that each application finds the correct resource files when it starts.

You can also manage your resources and ensure that they are read by explicitly reading them with xrdb, the X resource database manager. Xrdb is a host-based X utility that you can run in a startup file, like .xsession. The command line is:

```
xrdb -merge $HOME/.Xdefaults
```

This specifically merges the resources in the .Xdefaults file with resources from other locations, like app-defaults or .mwmrc.

```
!HDSTerm*screen.font: -adobe-courier-medium-r-normal--14-*


HDSview*background: aquamarine3

HDSview*toc.visibleItemCount: 40

HDSview*locator.width: 100


HDSDClock*foreground: red

HDSDClock*background: black

HDSDClock*geometry: 220x65-5+5

HDSDClock*date: %W, %M %d

HDSDClock*font: 10x20

HDSDClock*tails: Yes


HDSCalc*foreground: LightSkyBlue1

HDSCalc*background: SkyBlue4
```

This sample file indicates some of the resources you might use for some clients and the way they are all used together. Note the different syntax for the resources depending on each client. Note also that trailing spaces will create an error, so be careful how you write the file.

3.4.2.2 Application Resource Files

Each application program has resources set in its own resource file. This is the “applications-default” file, or app-defaults. For X clients, this is located in */usr/lib/X11/app-defaults.* For other applications, the app-defaults file is sometimes in the same directory as the executable file. For OpenWindows, it is in *$OPENWINHOME/lib/app-defaults*.

There are also environment specifications XUSERFILESEARCHPATH and XAPPLRESDIR that specify a directory for these files. These environments may or may not be set on your host. If XAPPLRESDIR has not been defined, the application will look in the user’s home directory for a resource file with the application’s class name.

3.4.2.3 Run Command Files for Resources

The UNIX method of entering these settings is with a run command file, or .rc file. This file is also used as a reference location for resources. These files can contain resources, and other variables, for each application. It is advisable to locate all the appropriate resources in a single file to simplify your system control and to insure that the files are read.

A resource file can also be an .rc file, such as .mwmrc, usually in a user’s home directory.

Each of these file types is discussed below. It is best to keep resources together in a consistent location. The actual location is not too important, but you must be sure the file is read when you start up.

A typical .Xdefaults file might contain:

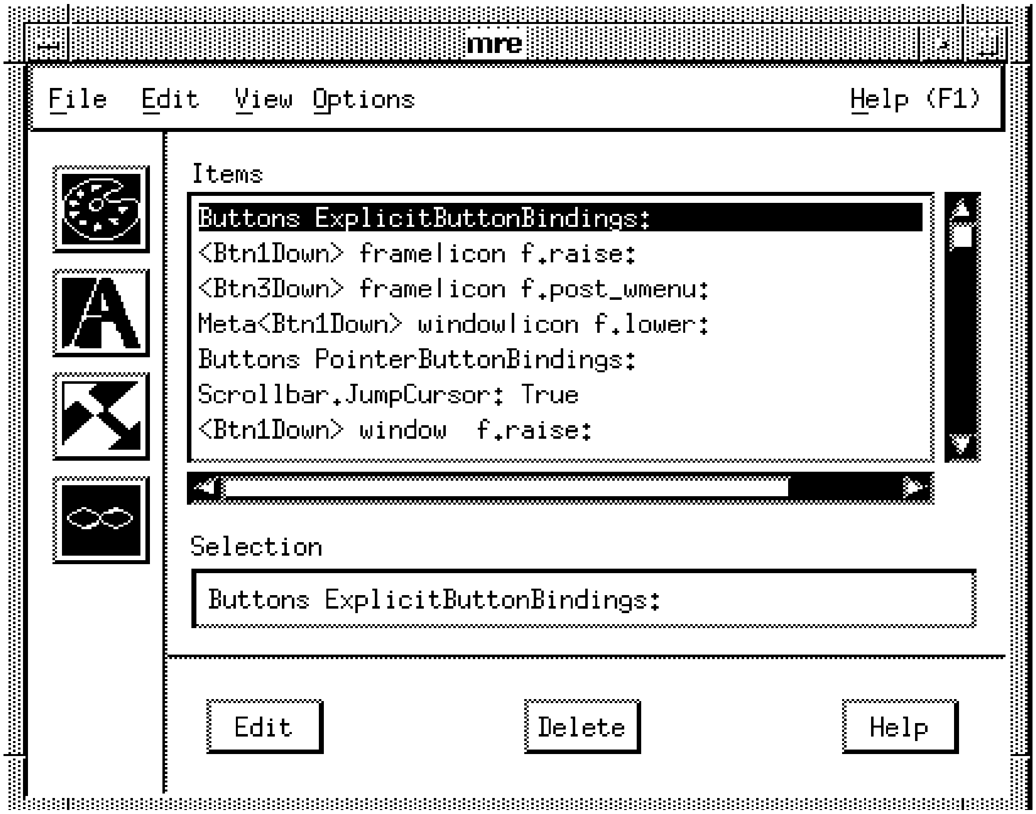
```
!
!X Resource specifications
!
xterm*scrollbar: true
xterm*background: SkyBlue
xterm*saveLines: 300
XTerm*Curses: true
XTerm*JumpScroll: true
XTerm*MultiScroll: true

xmanCommands*Geometry: 132x232+0+0
xmanSections*Geometry: 156x255+0+0
xmanSearch*Geometry: 316x48+0+0

Mwm*keyboardFocusPolicy: pointer
Mwm.autoClientMenu: false
Mwm.autoClientMenuName: HDS Clients
Mwm*buttonBindings: PointerButtonBindings
Mwm*windowMenu: WindowMenu
Mwm*keyBindings: DefaultKeyBindings
Mwm*fadeNormalIcon: True

HDSAudio*background: IndianRed1

HDSTerm*screen.SaveLines: 300
HDSTerm*statusLine: false
HDSTerm*bitmapDirectory: /HDS$$ROM
```



```
Buttons ExplicitButtonBindings:
<Btn1Down> frame|icon f.raise:
<Btn3Down> frame|icon f.post_wmenu:
Meta<Btn1Down> window|icon f.lower:
Buttons PointerButtonBindings:
Scrollbar.JumpCursor: True
<Btn1Down> window  f.raise:
```

```
Buttons ExplicitButtonBindings:
```

You can start the resource editor by typing:

mre &

This starts the Motif Resource Editor and reads the .Xdefaults file in your home directory. From that point, you can use the editor to change your Motif resources. The Motif Resource Editor screen is shown above.

3.4.2 X Resource Database

The resources you've set are put into a file and read by a resource manager on demand. There are several locations for these resource files, depending on their function.

3.4.2.1 The X Resource File

The X resource file itself is simply a list of resources indicating the user's preferences for various applications, usually located in the user's home directory.

Another resource file can also be an app-defaults file, which has resources specific to one client only and is generally located in a system-wide directory such as */usr/lib/X11/app-defaults*.

A resource file can also be an .Xdefaults file and it will typically have resources for many different clients.

the result will be a foreground text color of AntiqueWhite and a cursor color of PaleGreen.

3.4.1.4 Resource Bindings

In the hierarchy of resources, there can be many levels of variables. The number of levels and their connections are often unknown so resource specifications can be entered with a kind of "wildcard" (called a loose binding) to cover unknown levels. Variables in the resource are linked with "*" and "." for loose and tight bindings respectively.

In most resource specifications that a user would use and change, a "loose" binding using the "*" is appropriate. This is a general specification of the resource hierarchy and indicates that any number of intermediate steps may be present.

A "tight" binding uses the "." to link levels of the hierarchy. This tight binding is used when there must be an exact match of the specified resource and the widget hierarchy. Tight bindings override loose bindings.

From a user's perspective, loose bindings are preferred, both from their ease of use and their low level of priority. Any application program that needs a particular resource setting will use a tight binding to insure that it is in place and the user's settings should not interfere. For example, you might use a loose binding to select a large font for an application window but the application, by using tight bindings, would continue to use its own font sizes for the window's buttons and menus.

3.4.1.5 Resource Precedence

There are several rules for using resources:

- Most recent entries override older entries. This means you can append a resource to the end of a resource file and it will override earlier entries.
- Instance specifications take precedence over class specifications.
- Explicit specifications take precedence over partial specifications.
- Tight bindings take precedence over loose bindings.

3.4.1.6 X Client - editres

The editres client is included with the MIT X distribution. It displays the client's resource hierarchy and lets you see what the names and connections are. Note that editres may not work smoothly with X11R4 Motif clients. It works better with X11R5 and more recent Motif releases.

3.4.1.7 X Client - mre

The Motif resource editor is called mre. This is an editing utility program provided with the Motif Window Manager which lets you edit your .Xdefaults file. The Motif Resource Editor may be installed on your system, or you may need to have your system administrator set it up and put it in your path.

3.4.1.1 X Resource Syntax

The syntax of these resources is very specific and you should be very careful with making entries. The man pages for each client have lists of resources available for that client and will show you the exact resource name and its permitted values.

Note that there may be other resources not specifically mentioned there, but are included as part of the toolkit which was used to build the client. For instance, if a client (such as HDSterm) uses the Motif toolkit for its GUI, it will support Motif resources in addition to its own listed resources.

3.4.1.2 Resource Classes and Resource Instances

The terms Class and Instance indicate the generality of a resource. A Class may have several Instances. By convention, class names use an uppercase letter as their first character and instance names use a lowercase letter. For example:

```
xterm*background: AntiqueWhite
xterm*Background: PaleGreen
```

are instance and class resources, respectively.

A resource instance is always associated with a class. Usually you can tell what the association is because the names are similar. However, there are cases where the names may not appear to have any connection, for instance in xterm, the resource cursorColor belongs to the class Foreground (because the cursor is part of the foreground).

3.4.1.3 Class and Instance Hierarchy

There is a hierarchy of execution for classes and instances. Resource instance settings override resource class settings. This may seem backward, since users often think that classes are more powerful than instances. It is more intuitive to think of classes as “general” settings and instances as “specific” settings.

This mechanism allows client and user settings to work together. If you, as a user, specify a resource with a class name, and an application program specifies an resource instance, the application’s value will be used. In this way, specific settings are preserved and preferences honored if possible.

For example, suppose you had set the Instance resource:

```
xterm*foreground: AntiqueWhite
```

Because this is a resource instance, it will apply to the foreground text only and not to other instances of the Foreground class (such as cursorColor).

If you set the Class resource:

```
xterm*Foreground: AntiqueWhite
```

it would apply to all instances of that class (foreground text and cursorColor).

If you set both Class and Instance resources:

```
xterm*foreground: AntiqueWhite
xterm*Foreground: PaleGreen
```

Xrdb - X Resource Database Operations

This section describes X resources, some of their customizing capabilities, and using Xrdb to manage them.

Section 3.0 - Network and Host System Operations

3.4 X Resource Operations

Resources are variables that define a window's or a client's properties. They are important to the way you use the ViewStation because they control the way a window looks and works. They also control client keyboard mappings.

In most cases, the resources as set by the client application's designers or your system administrator are sufficient for most applications and environments. However, the many "look and feel" issues of a graphical user interface can be customized by appropriate selection of X resources.

3.4.1 X Resource Format

Resources are specific for each client and typically start with the client name, such as xterm or hdsclock. The next entries in the resource line are typically some component of the client, such as scrollbar, backgroundcolor, or title. These entries can be several levels deep, such as scrollbarLocation or scrollbarWidth. Each of these variables then has an associated value, either true/false or some entry like a color or a font name.

Some examples of resources are:

```
xterm*scrollbar: true
xterm*scrollbarLocation: left
xterm*background: AntiqueWhite
```