

```

if the end of the resource string has been reached
if '**' is the last separator then set the value over this
widget and all its descendents else if '.' then set the
value over this widget only
else
descend the widget tree, parse off the resource
segment, determine the children of this widget and call
__set_and__search procedure for each child widget
else
if '.' is the resource separator
continue looping - call __set_and__search procedure to
look at the rest of the siblings of this widget
if '**' is the resource separator
descend the widget tree but don't parse the resource
string and continue looping call __set_and__search
procedure to look at the rest of the siblings
end loop

```

When a resource string and one or more widgets have been matched, the resource value must be set to the widgets. The procedure for setting resource values first determines if the resource type is valid for each widget. For example, labelString is a valid resource for a label widget but not for a bulletin board widget. In order to determine whether the resource can be set a particular widget, a list of all valid resources for the widget is obtained from the application's resource database. These resources are matched against the value portion of the user-edited resource string. If the user-edited resource is not a valid resource for the widget, or if the resource name or class are NULL, the next widget is tested. The next step in the procedure is to build from the user-edited resource a full resource name and class specification for the widget of interest which can be used to query the resource database.

In the query process, the resource and its value are placed in the application's resource database and then queried back out again using specific name and class resource strings that that were built from the widget information. This ensures that the resource editor's on-the-fly paradigm is followed: an application that is instantaneously updated should look the same as one that is restarted using the .Xdefaults file. The resource editor should not violate the precedence rules of the resource database manager. For example, consider the resource specifications:

```
*background: red
```

```
app*mywidget.background: blue
```

Both resource strings would apply to "mywidget". However, because the second string is more specific, it would override the first string so the resulting value would be blue. By calling the resource manager function XrmPutStringResource() with the resource and value, and then querying the resource database manager with XrmGetResource(), the resource database manager is requested to apply its precedence rules and return the proper value.

Once the proper resource value is determined, the Xt function XtVaSetValues() is called with the resource name and the value. By using the XtVaTypedArg argument in the XtVaSetValues() function, the resource value, which is a character string type is automatically converted to a type known for the resource. Prior to this point, no manipulation is performed on the value string. The Xt intrinsics handle type conversion just as they would for string values read from the .Xdefaults file and it is therefore unnecessary to keep track of resource data types. Moreover, internationalization is not a concern; Xt and the X toolkit will automatically handle the resource value properly even if it is a multi-byte string.

Accordingly, a graphical resource editor for customizing software applications in a data processing system has been

disclosed. Although preferred embodiments of the invention have been shown and described, it will be understood that many modifications thereof may occur to persons skilled in the relevant art. For example, it would be possible to provide editors or other graphical interface mechanisms for editing other resources types in addition to those described above. Such resource types could include keyboard editors, mouse editors, screen saver editors and many others. The invention therefore, is not to be limited except in accordance with the spirit of the appended claims and their equivalents.

We claim:

1. In an application resource editor for customizing graphical resources of a software application executing concurrently with the application resource editor in a data processing system, a method for dynamic customization of the software application comprising the steps of:

displaying a list of resources in said software application that are capable of modification;

determining from user input an application resource to be modified from said list of resources and a resource value representing an attribute of said resource to be modified;

searching a list of resources in a resource database maintained by said software application and identifying matching resources in said resource database corresponding to said resource to be modified; and

applying said resource value to each matching resource in said resource database in order to change the values of said matching resources and thereby customize said software application.

2. In a data processing system, the data processing system having a data processing unit, a video display terminal, a keyboard, a cursor control system including a screen cursor and a user actuatable cursor positioner for providing control inputs to the data processing system, a data storage resource, and a graphical user interface system through which software applications running on the data processing system generate one or more graphical window objects on the video display terminal to provide user interaction functionality for the software applications, the appearance and behavior of the graphical window objects being partially determined by resources, a graphical resource editor for dynamically modifying selected resources in a software application running concurrently with the graphical resource editor in the data processing system, comprising:

means for displaying a list of resources in said software application that are capable of modification;

means for determining from user input an application resource to be modified from said list of resources and a resource value representing an attribute of the said resource to be modified;

means for searching a list of resources in a resource database maintained by said application and identifying matching resources in said resource database corresponding to said resource to be modified; and

means for applying said resource value to each matching resource in said resource database in order to change the values of said matching resources and thereby customize said software application.

3. A graphical resource editor computer program product for dynamically modifying selected resources in a software application running concurrently with the graphical resource editor product in a data processing system, the data processing system having a data processing unit, a video display terminal, a keyboard, a cursor control system including a screen cursor and a user actuatable cursor positioner for