

01

Elements of the Graphical User Interface

"While it is the user's job to focus on tasks, the designer's job is to look beyond the task to identify the user's goals. Therein lies the key to creating the most effective software solutions. Software designed to achieve purely business goals will fail, but if it is designed with the personal goals of the user in mind, it will also achieve its business goals. Of course, the program must satisfy the business problem at hand, but the people who use it will not behave like invoices, database records or modules of code."

- Alan Cooper

B.1 INTRODUCTION

GUI applications are not just character-based programs with better-looking graphics. GUIs represent a paradigm shift from programs in which the user interface is a practical afterthought to a realization that user interface is equally, if not more, important to the success of the system. As end users grow to expect applications to be friendly, the burden upon the developer increases - it is not enough for an application simply to be functional.

Modern applications must ultimately make themselves useful and intuitive to the end user. In other words, the success of an application is based not upon its technical merit, but on how useful it is. An application is a failure if the user cannot figure out how it works, regardless of its stability or performance. Since the user interface is the single most important factor that will make or break an application, it is important that all application developers should be familiar with the principles of GUI design as well as the technical aspects of building an application.

B.2 KNOW THY USERS

Knowing who the end users of an application are and how they work is crucial for the designer. While this may seem obvious, many application designers make the mistake of considering the system more important than the user. This trend usually starts with the requirements definition, which does not include the user. For example: "We need to keep better track of the inventory and so we need an application to record it." An application however is not capable of recording data on its own. Data will not be recorded unless somebody actually uses the application. Therefore, the primary goal is to get the users to use the application to record inventory data. So the requirements definition statement can be improved: "We need to keep better track of the inventory and so we need a system which will help our storekeepers enter the inventory data easily." The difference is the distinction between designing an application for the user and an application for the business. The latter approach rarely succeeds, and the former rarely fails.

Knowing the end users is not an easy task. It goes beyond learning what data they would like to keep track of and what features they would like in the system. There are often concepts for specific end users that will outweigh general principles—for example if you know that the application that you are developing is going to be used along with an application (say like Lotus Notes) where the double click of the right mouse button closes the currently open window, you would want to consider implementing the same in your application also. If you know your users are visually impaired, you may want to use larger fonts and blinking status indicators, etc. Ideally, the end users should be actively involved in developing the 'look and feel' of an application.

2 ▲ Essentials of Database Management Systems

The next most common mistake that a developer makes is assuming that the end user is just like the developer. While this can be true in some rare occasions, it is more often the case that the developer has great deal more technical knowledge and familiarity with computer systems than the end user does, and the end user tends to know a whole lot more about what they would like to accomplish than the developer. So why not have the end users develop the application? Because developers are better than end users at translating end user needs or tasks into usable applications and they are in a better position to say what is possible and what is not. So for developing a good and successful application, the application designer should develop a good knowledge of the tasks or the user requirements.

B.3 CONSISTENCY

The only rule that must not be broken under any circumstances is consistency. Nothing is more confusing and frustrating than an application that violates its own rules. When a function is available on a button or key on one screen, the end user will expect to find the same function in the other screens also. For example, if you have assigned 'F1' key for context-sensitive help in one screen, the standard should be followed throughout the application. End users will feel much better about an application if they feel they are learning how to use it and how it works rather than learning its quirks. Consistency also helps in habit formation and the user will be able to do the routine tasks without any conscious effort over a period of time.

Consistency applies to all aspects of an application. If the user knows nothing about the application, computers or their platform, they will still be able to easily identify errors in consistency. Since Oracle allows code reuse and inheritance of properties, inconsistencies are inexcusable.

Consistency can be easily promoted using Oracle Developer/2000 in a team environment by defining objects, property classes and visual attributes to be used by an entire team. Many development teams put together template forms and reports, containing common code, parameters and object definitions. The template forms and reports are used instead of developing each form or report from scratch. This approach reduces the programming effort and ensures consistency and improves maintainability.

B.4 WHEN TO USE WIDGETS OR CONTROLS

There are two common mistakes made by application designers new to designing GUI applications: overuse of widgets and under use of widgets. When widgets are overused, the application has an amateurish, toy-like appearance and actually gets in the way of using the application. On the other hand, without any widgets, the GUI application might as well be a mainframe application running on a dumb terminal.

Both problems come from a misunderstanding of the purpose of a widget. A widget indicates to the user what can be done and how to do it. For example a button with a label indicates that something can be accomplished by clicking the cursor on the button—the user is not required to remember a keystroke or type in a code or know the menu to perform the task - the button is right there on the screen, whenever he needs it. The presence of the button communicates to the user that the function is available and it is immediately apparent what must be done to perform the function. In other words, the widget's purpose is to couple the visual and functional aspects of an application - one major advantage of the GUI.

Now let us examine some of the most commonly used widgets. Wherever possible, we have used examples from the applications that you are familiar with (like MS-Word) rather than showing the controls in isolation.

B.4.1 Button Control (Push Button)

Button control allows clicking it to perform an action. Text and images can be placed on a button. The button has a specific name and allows you to write an event handler to control the actions performed when the button is clicked.



Figure 1 Push Buttons with Images (Opera)

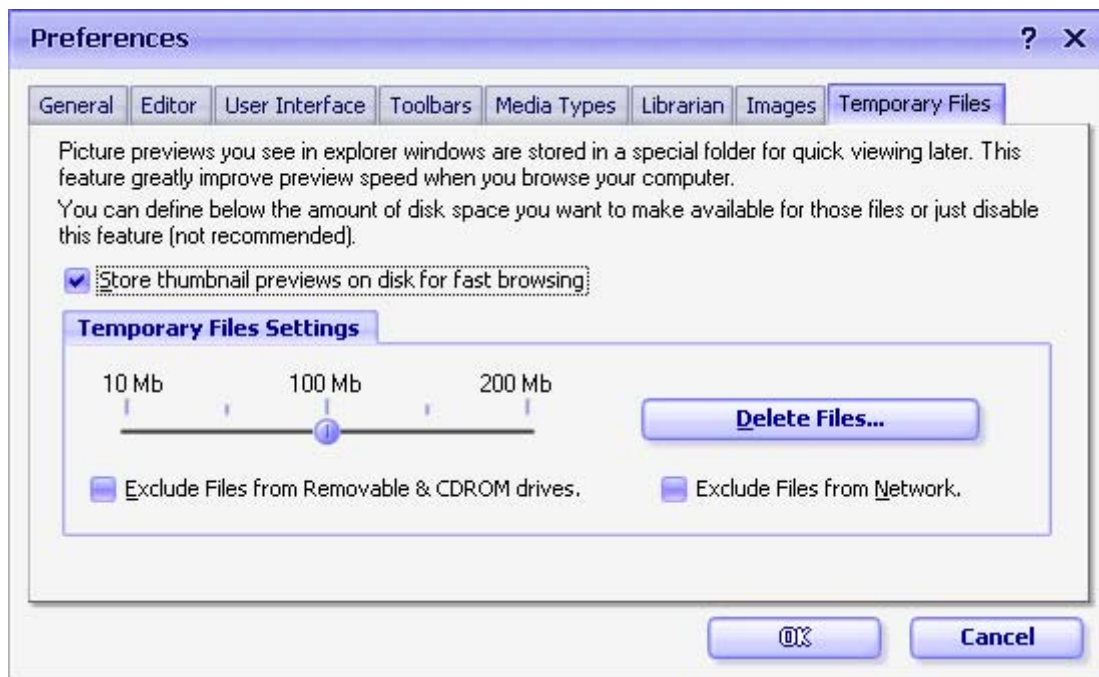


Figure 2 Push Buttons (Delete Files, OK, Cancel) in the Preferences Dialog Box (Icon Workshop)

B.4.2 Radio Button

Radio button is a button used to turn mutually exclusive settings on and off. Users can set only one radio button at a time. Radio button allows the user to choose only one of a number of options. When you choose one option, any previously selected option is unselected. Radio buttons are used when you want to let the user select one—and just one—option from a set of alternatives. Only one button in a group of radio buttons can be selected at one time.

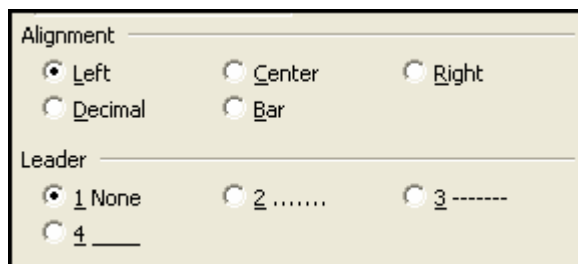


Figure 3 Alignment and Leader Option Radio buttons in the Tabs Dialog Box (Word 2002)

Radio buttons are ideal for selecting only one setting from two to six possible settings and letting users toggle between two states when the states are not opposites or easily inferred from one another (for example (Male, Female)). Radio buttons are not suitable when there are more than six or seven settings at a time (use a single-selection list or a drop-down list box instead) or for starting programs or opening dialog boxes (use pushbuttons instead) and for switching between two views of the same data (use tabbed windows or dialog boxes instead).

B.4.3 Check Box

Check box is most commonly rendered as a box when the element is off and a box with a check when the element is on. The user can switch the state of the check box by clicking it with the mouse. A check box is a graphical component that can be in either "on" (true) or "off" (false) state. Clicking on a check box changes its state from "on" to "off," or from "off" to "on." Checkboxes are used to allow user to select several items with the click of a mouse. Groups of checkboxes can be used to create checklists. Users can set any number of check boxes, including none.

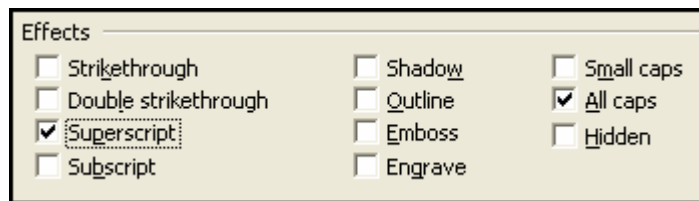


Figure 4 Check Boxes in the Font Dialog Box (Word 2002)

Check boxes are square (radio buttons are round) and can have either text or iconic (picture) labels. The "on" setting is usually indicated by a check mark or an X inside the box. However, **Motif** and **iconic** check boxes (see Figure 5) simply look pushed-in or grayed out.



Figure 5 Motif (iconic) Check Boxes (Formatting Toolbar of Word 2002)

B.4.4 Command Line

Command line is an entry area from which users can run commands and searches.

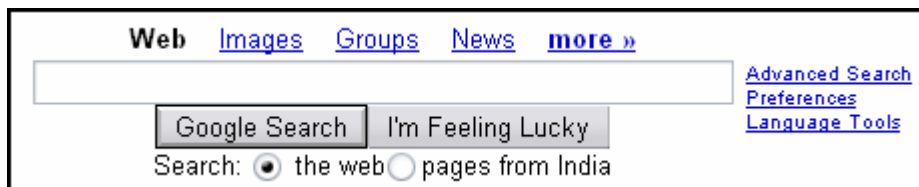


Figure 6 A Command Line for Searching in Google

B.4.5 Combo Box

A combo box is a combination of a **list box** and a **text-entry area**. It lets the user to choose one among several choices. The list box allows only single selections. The text-entry area has two functions: searching and data-entry. There are two very different forms of combo box component. Default form is uneditable combo box which is represented as button and drop-down list of values. The second form, called the editable combo box, features a text field with a small button abutting it. Depending on the type of combo box, the text-entry area may simply show whatever item was selected from the drop-down list, let users type in search items, or let them add new entries to a database.

There are three types of combo boxes—simple, drop-down combo, and drop-down list. Users can add new items to the simple or drop-down combo boxes, but not to a drop-down list. Users can search all three types by typing the search item in the entry area. The **simple combo** box (Figure 7) is an entry area with a list below it. Use this style whenever you have enough space—being able to see the list items is always good.

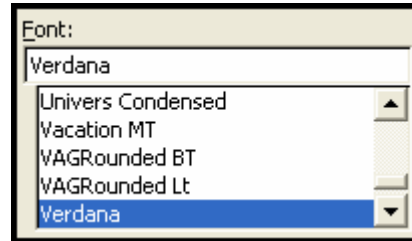


Figure 7 Simple Combo Box

The **drop-down combo box** is an entry area with a down-arrow button to its right. Use this style when space is limited—for example, on a toolbar. Both drop-down combo box and drop-down list box look the same, but users can add values to the drop-down combo box, but not to the drop-down list box. The **drop-down list box** (Figure 8) looks the same as a drop-down combo box, but is does not permit the users to enter new items. They can only select from the items already on the list.

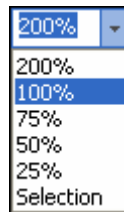


Figure 8 The Drop-down List Box

The combo box is ideal for presenting a list of suggested choices. Combo boxes let the user:

- Select from a list
- Type in a new entry
- Search for and jump to an item on the list by typing its first character or first few characters

But is not a good idea to use a combo box in the following situations:

- Selecting more than one item at a time. Combo boxes allow only single selections.
- Fewer than five items (unless you expect to add more items later). Use radio buttons instead.
- Restricting users to predefined items (except for the drop-down list box). Use a list box instead

B.4.6 Dialog Box

A dialog box is a window used to hold settings or secondary information and to gather information from the user. The main window contains the user's actual task (Find and Replace, for example), dialog boxes let users change how the application itself works (to find what and replace it with what).

Dialog boxes have three equally important functions:

- Transactional – Gathering the details needed to complete a command—for example, which file to save or which file to open.
- Tools – Holding tools such as spelling checkers, floating toolbars, and palettes. Tools usually float on top of main windows, are always visible, and are not confined to any secondary window or document in a multiple-document interface (MDI).
- Messages – Delivering messages and providing feedback. Message boxes are used to ask questions, confirm actions, and warn of problems. These are also called message boxes or alert boxes depending on the nature of the message displayed in the box.

6 ▲ Essentials of Database Management Systems

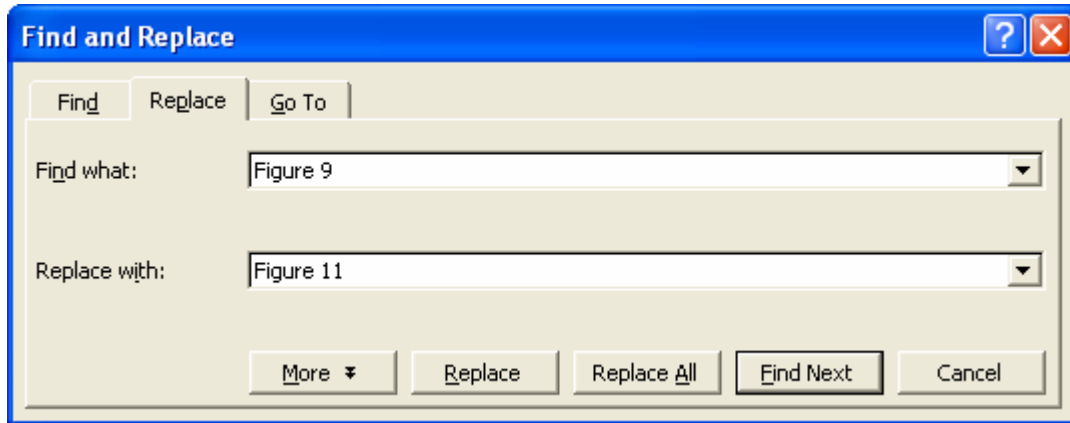


Figure 9 The Find and Replace Dialog box (Word 2002)

Dialog boxes can be classified into three types: standard, expanding and tabbed. The **standard dialog** box displays all the information it is supposed to display. For example the Change Case dialog box (See Figure 10) displays all the information it is supposed to display and is an example of a standard dialog box.

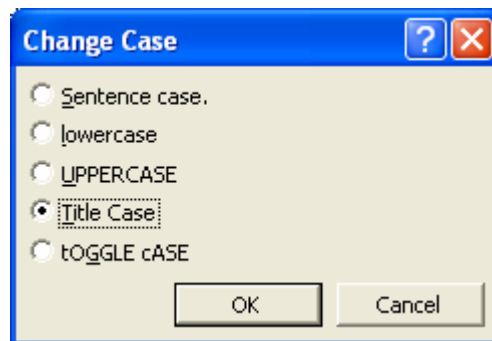


Figure 10 Change Case Dialog Box (Word 2002)

The **expanding dialog box** does not display all the information when it first appears. It usually will contain the most frequently used information. There will be a "More" or "Details" button on the dialog box. When the user presses the More button, the dialog box expands to show additional details or options. So, when we click on the More button of the Find and Replace dialog box (Figure 9) we can see that it expands to reveal a host of other information and options that we can use to customize the 'Find and Replace' process. The expanded Find and Replace dialog box is shown in Figure 11. Notice that the More button has changed to Less button. Once the user has selected the options that are required, clicking on the Less button will take the button to its original form (the non-expanded form).

The **tabbed dialog box** is one in which settings are grouped into sections. Each section has its own tab with a tab name. Users switch among the groups by clicking on the tabs. For example, the Find and Replace dialog box is a tabbed dialog box. It contains three sections and therefore three tabs: Find, Replace and Go To. If you only want to find something, then you can click on the Find tab and so on. The tabbed dialog box is an excellent control for displaying large amounts of information that otherwise would have needed larger dialog boxes.

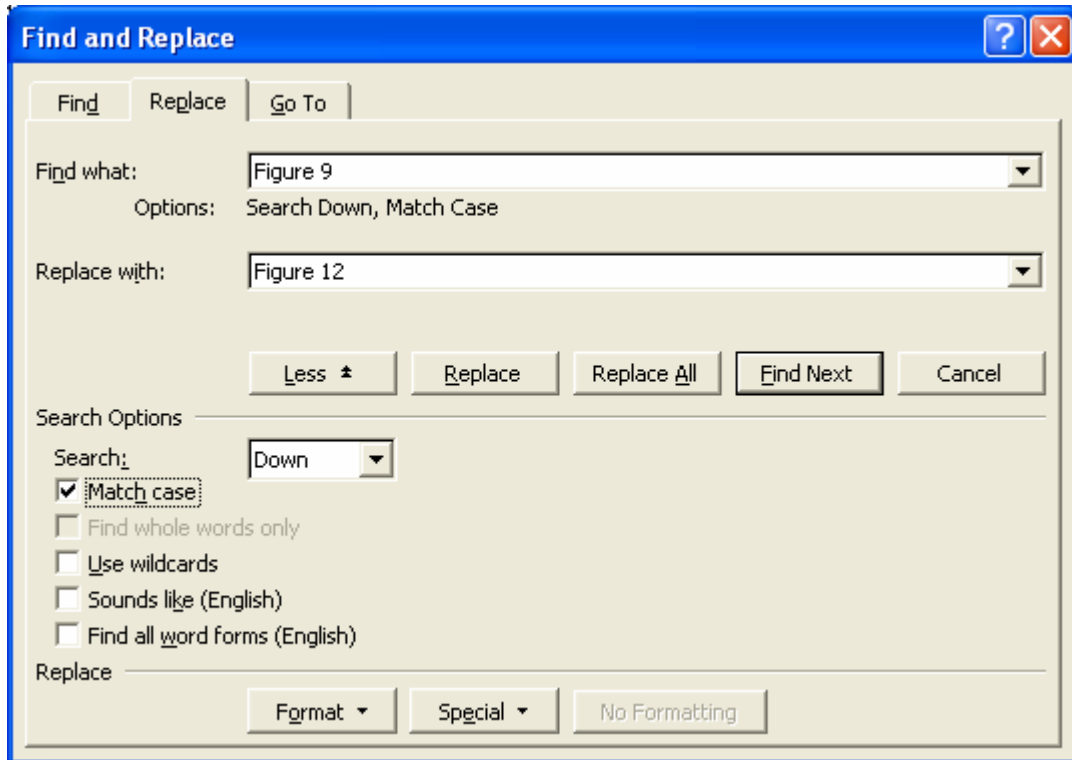


Figure 11 Expanded Find and Replace Dialog box (Word 2002)

B.4.7 Text Box

Text box is a control used to either display text, to request text, or to do both. It is provided as a rectangular area with borders. Usually there will be a label by the side of the text box indicating what you information you should enter into the text box.

If the text box is used for displaying information, then the label will indicate what information is provided in the text box. By default, an text box is used to display or request a single line of text. A text box is referred to as **multi-line** when it can display more than one line text.

Full Name:	<input type="text"/>
Company:	<input type="text"/>
Email:	<input type="text"/>

Figure 12 Text Boxes



Figure 13 Multi-line Text Box

B.4.8 Label

A label is a description, usually text but sometimes pictorial, of fields, buttons, and areas of windows or dialog boxes. Labels are good for indicating expected content for fields and indicating the purpose of a button or other control. They are not suitable for showing detailed instructions or offering instructions, status-bar messages, or context-sensitive on-line help.

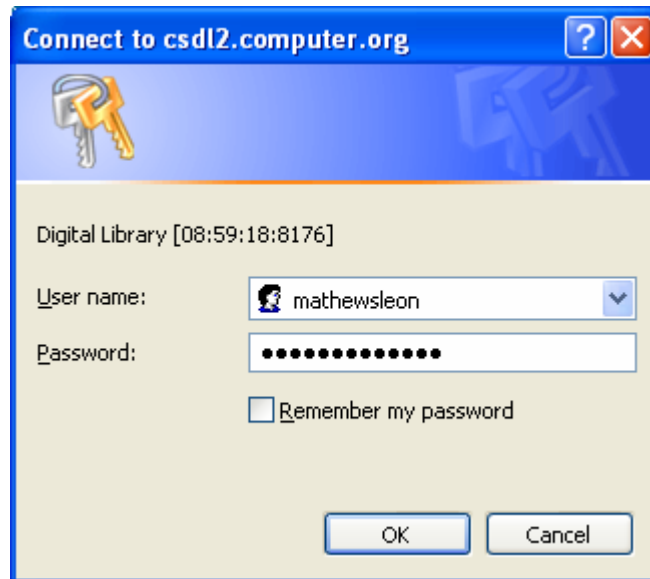


Figure 14 Pictorial and Button Labels

B.4.8.1 Iconic Label

An iconic label is a pictorial description of a tool or function that generally appears on toolbars and palettes. It is ideal for identifying tools that require a mouse or pen to be effective. Drawing tools—paintbrushes, erasers, and so on—are typical examples (Figure 15).



Figure 15 Iconic Labels for Drawing Tools (Pencil, Brush, Spray Can, Fill, Color Dropper)

It is also used for identifying mouse shortcuts—for example, Save, Cut, Copy, Paste. Use standard images for these types of options (Figure 16).



Figure 16 Mouse Shortcuts for Cut, Copy and Paste.

Iconic labels are not suited for abstract functions for which it is difficult to find visual metaphors and for tasks that do not require or benefit from mouse use.

B.4.9 Group Box

Group box is a static control used to create "physical" limits or sections of a dialog boxes. It is a rectangular box drawn around a group of controls to indicate that the controls are related and to provide a heading for the group. Group boxes are used to associate, isolate, and distinguish groups of related items in a dialog box. You can embed other controls, such as radio buttons, checkboxes, and pop-up menu buttons, within group boxes.

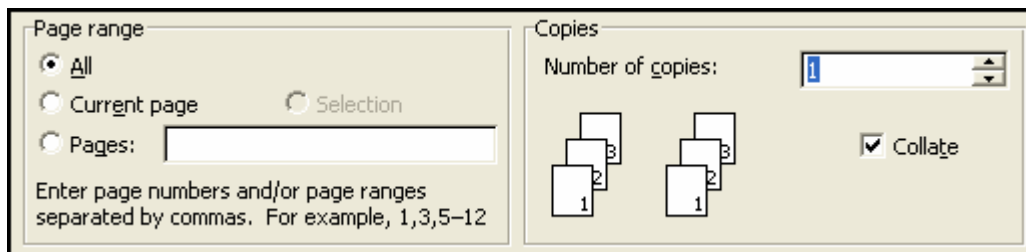


Figure 17 Page range and Copies Group Boxes in a Print Dialog Box

B.4.10 List box

List box is used to create a list of items. There are two types of list boxes—single selection and multiple selection list boxes. In the case of single selection list boxes, the user will be able to select only one item from the list of values. A **single selection list box** is ideal for making the user select one item from a list of six or more items (Figure 18). It can be used as an alternative to radio buttons. A single selection list box is not ideal for situations in which the list is not yet complete or well-defined.

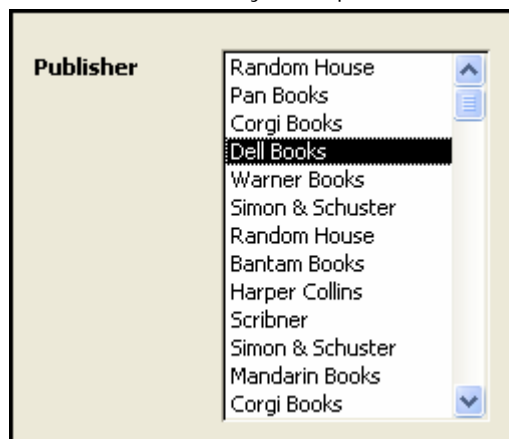


Figure 18 Single Selection List Box

A **multiple selection list box** (Figure 19) is a scrollable list from which users can select more than one item. A multi-selection list box is can be used as an alternative to check boxes. It is also ideal for letting users select more than one item from a long and possibly dynamic list.

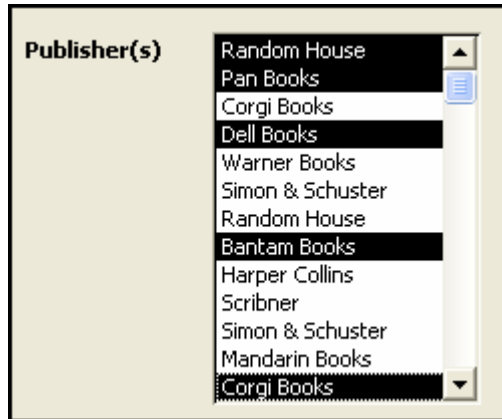


Figure 19 Multiple Selection List Box

B.4.11 Menu bar

Menu bar is the area on the main or secondary windows, the area containing the labels of the drop-down menus. A menu bar is ideal for showing the high-level structure of the application (Figure 20).

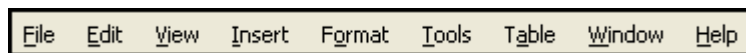


Figure 20 Menu Bar

B.4.12 Drop-down Menu

Menu is a list of options displayed to the user by a data processing system, from which the user can select an action to be initiated. A drop down menu is a list of application-related activities and settings that opens downward and, if there are submenus, usually to the right and down (cascades), when accessed with the mouse or from the keyboard. Dropdown menus are ideal for accessing secondary tasks and selecting settings that affect the entire application or window.

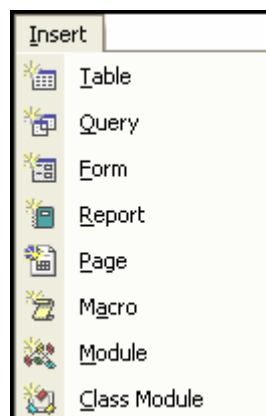


Figure 21 Drop-down Menu

B.4.13 Pop-up Menu

A pop-up menu is a context-sensitive menu accessed by pressing the secondary mouse button. Pop-up menus are ideal for expert computer users who want shortcuts, minimizing mouse travel by making

options available at the current pointer or cursor location and accessing properties of selected objects. But the pop-up menu should not be used for options that appear nowhere else in the system and for novice computer users who are unaware of pop-up menus.

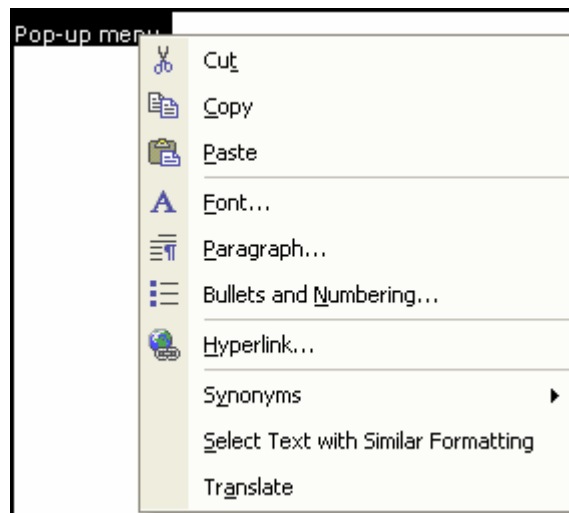


Figure 22 Pop-up Menu

B.4.14 Progress Bar

Sometimes a task running within a program might take a while to complete. It can be needed to know what task is occurring, how long the task might take, and how much work has already been done. One way of indicating work, and perhaps the amount of progress, is to use an animated image.

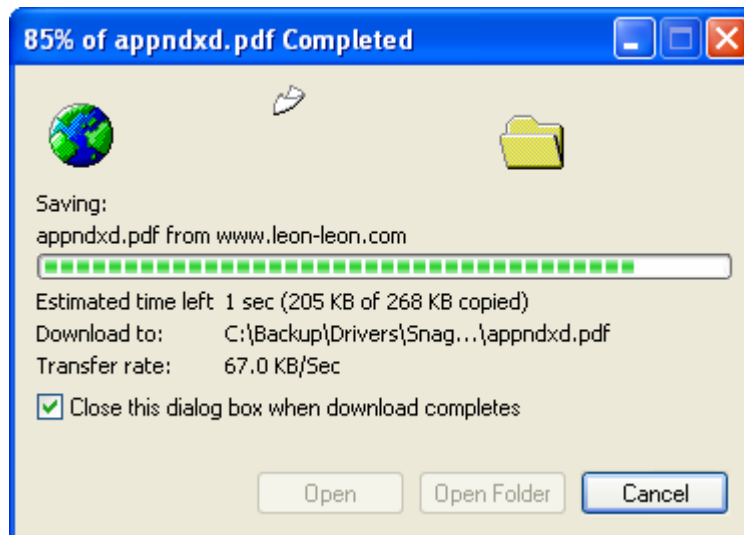


Figure 23 Progress Bar indicating the Progress of a File Download (Internet Explorer)

Progress bar offers a simple way to graphically show a process completion progress. As the process progresses a bar extends across the component until the job is completed and the bar is filled. The movement of the bar is usually part of some multithreaded task filled with a color.

B.4.15 Scrollbar

Scrollbar is a familiar user-interface object and a graphical device used to change a user's view of the contents of a window. A Scrollbar is a narrow rectangular area consisting of a scroll area (trough), a scroll box (slider), and scroll arrows or anchors at either end that is used to represent the user's relative position in a document, file, or list. Scroll bars can be vertical or horizontal.

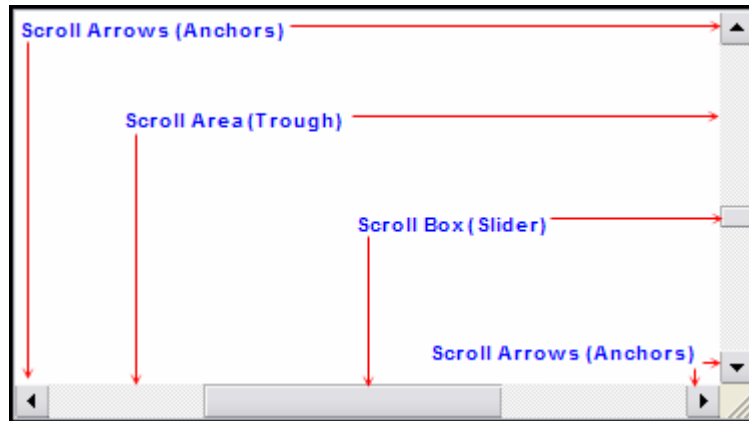


Figure 24 Horizontal and Vertical Scrollbars

Scroll bars are ideal for viewing information that is beyond the edge of the scrollable object (list, window, or dialog box).

B.4.16 Slide Bar

Slide bar is a very narrow version of a scroll bar. It does not have arrows at both ends, like the scrollbar. Slide bar is suitable for situations in which space is tight, either because the window is very dense or because the main window has to hold many sub windows or panels.

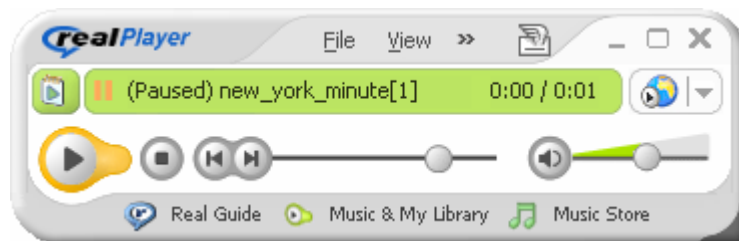


Figure 25 Real Player's Control Panel which has a Slide Bar

B.4.17 Slider

Sliders are controls for choosing a value from a range of values. Common uses are volume controls, seekers for movie and sound files as well as color pickers. Once a slider is focused the arrow keys and Page Up / Page Down keys can be used to change the value. Slider can be either horizontal or vertical. Sliders are ideally suited for selecting one value or point from an infinite number of values or points. It should not be used for precise entries, situations in which the range is not continuous or if the number of choices is fewer than 10.

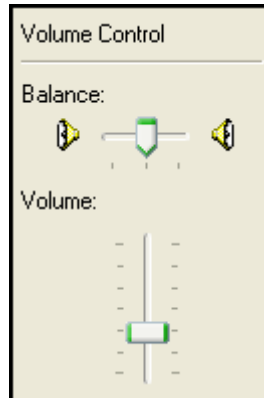


Figure 26 Horizontal (Balance) and Vertical (Volume) Sliders

B.4.18 Spin Control

There are many occasions where it is nice to have spin button control with auto disabling arrow buttons. Spin control is a pair of arrows that user can click to increment or decrement a value displayed in a companion text control. User can also type in a value into text control or edit box directly. The value associated with a spin button control is called its current position.

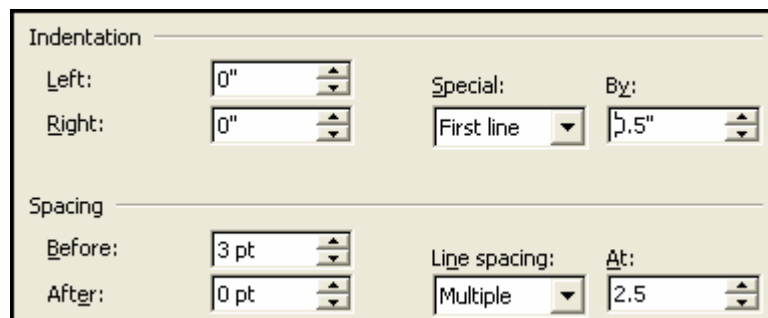


Figure 27 Spin Controls in the Format Dialog Box of Word 2002

Spin control is suited for applications with limited screen space, setting predictable, customary, or consecutive values (numbers, days of the week, and so on) with the mouse. It should not be used in situations in which users need to compare the choices and therefore need to see them or when the users have limited patience or manual dexterity as the arrows are small targets and therefore hard to hit.

B.4.19 Status Bar

Status bar is an area at the bottom, or occasionally the top, of a main window that displays information about the current state of what is being viewed in the window or any other contextual information, such as keyboard state (OVR vs. INS, for example). It may also contain progress messages and screen component definitions. Status bar is intended for a visual recognition of the user's actions.

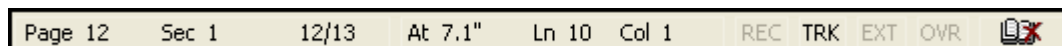


Figure 28 Status Bar of Word 2002

B.4.20 Tab Control

Tab control (tabbed dialog box) is analogous to the dividers in a notebook or the labels in a file cabinet. By using a tab control, an application can define multiple pages for the same area of window or dialog box. Each page consists of a certain type of information or a group of controls that application displays when user selects the corresponding tab. A tab control is a container window that organizes several controls in multiple pages.

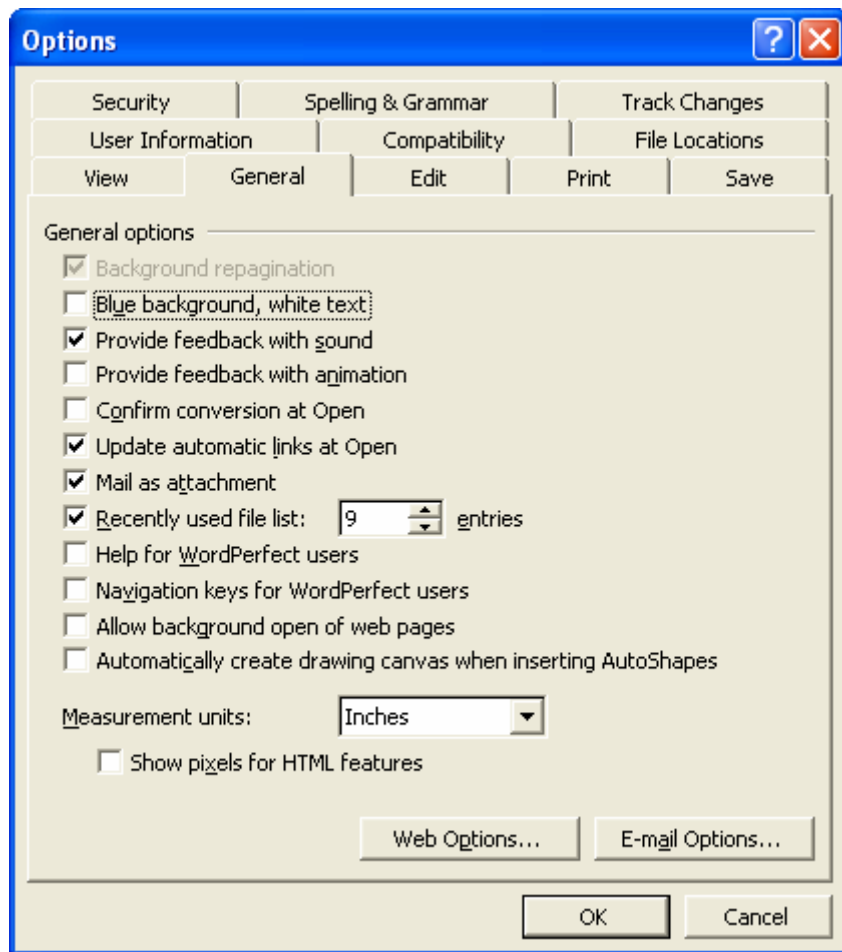


Figure 29 A Tab Control (Options Dialog Box of Word 2002)

B.4.21 Toolbar

Navigating a menu with a stylus can soon become tedious, especially if user selects the same menu item over and over again. The tool bar control appeals to reduce the number of clicks on menu items and performs executive menu options.



Figure 30 Portion of the Standard Toolbar of Word 2002

A toolbar is ideal for accessing often-used (Save, Cut, Copy, Paste, etc.) or repeatedly used (bullets, numbering) functions with the mouse. It is also ideal for finding important functions easily. Because toolbars make options visible, they also remind users to do important actions (for example, Save).

B.4.22 Tooltip

Tooltip is a small window which displays some text when user hovers the mouse over a control giving a hint about what should be done with control. ToolTips are hidden most of the time. They appear automatically, or pop up, when the user pauses the mouse pointer over a tool.



Figure 31 A Tooltip

The Tooltip appears near the pointer and disappears when the user clicks a mouse button or moves the pointer away from the tool. A tooltip is a good solution for finding out what a screen component does, especially when it has no text label.

B.4.23 Message Box (Alert Box)

Message boxes are used to provide information, confirmation, warning or alerts to the users. A message or alert box is a specialized, usually moveable, type of dialog box used to present information or warnings and ask questions.

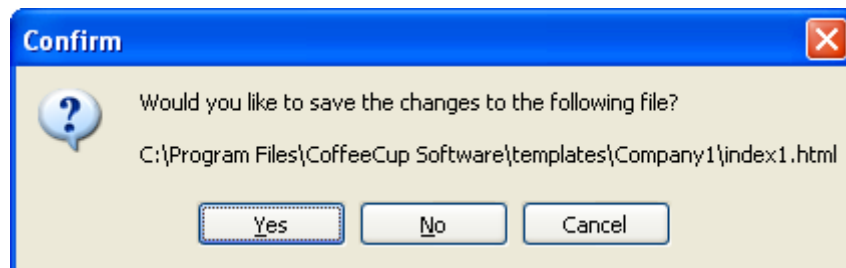


Figure 32 A Message Box Asking the User for a Response

A message box is good for conveying information to the users (for example, informing the user once the files are deleted), asking for a user response (for example, whether to continue with a file deletion), explaining an error (for example, size of the source and target cells should be the same) or confirming a potentially dangerous choice (like warning if the user is accidentally proceeding to format the disk), etc.



Figure 33 A Message Box Informing the User about the Options Available

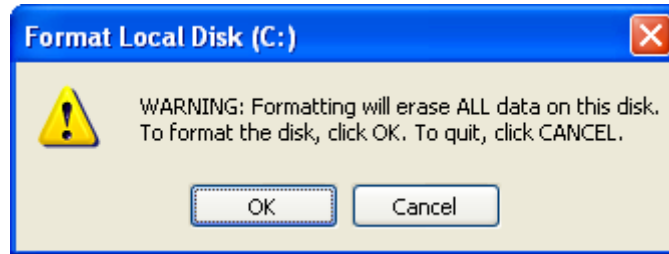


Figure 34 A Warning Message

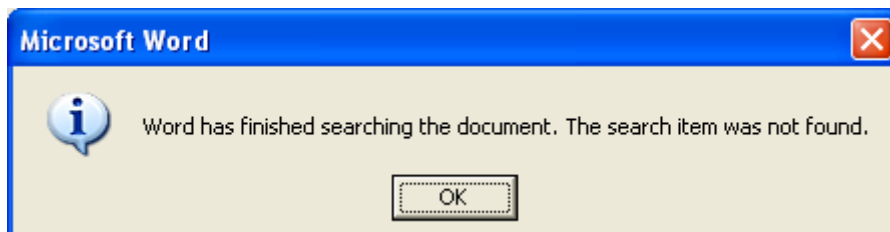


Figure 35 A Message Box Conveying Information

B.5 MESSAGES

There are many circumstances under which messages should be presented to the user. The messages can be informational, warnings or error messages. When presenting any kind of message to the user, two questions must be answered in the message:

- Exactly what happened?
- Exactly what do I do about it?

Traditionally, applications have not been very good about answering either question, resulting in cryptic, useless and often rude messages like "Error # 303". This happens because error handling is left until the end of the development process, which is a very bad strategy. This gives little consolation for the users who are left to wonder what they have done and what they should do.

Even if the first question is answered adequately, the second question is still imperative. A message such as "Error, the application's Help cannot be accessed" adequately informs the user what happened, but not what to do. Should the application be reinstalled? Is there something I can check? Can I fix this myself? Do I try again later? Answering questions isn't really that much a burden on the developer. It involves a little more typing, but not much more effort to change, "A disk error has occurred" to "There is not enough space on drive C:. Please specify the new location for the data files". It is perfectly reasonable to ask the user to refer the user manual or help file, if the questions "So what?" and "Now what?" cannot be adequately addressed in the message.

Messages should be displayed so that the user gives them attention they deserve. Vitaly important messages should be placed in alert boxes, status messages should be displayed in the status bar, etc. If every trivial message pops up on the screen to interrupt the user's work, the user will quickly ignore all messages, including the important ones.

B.6 THE MOUSE AND THE KEYBOARD

Although the mouse is usually considered as the cornerstone of GUI applications there will be always situations in which the user needs to perform the functions without using the mouse. The most common reason can be the machine does not have a mouse. In all cases, it is best to have keyboard alternatives for each mouse operation so that an application user is not lost without a mouse. Experienced users will also appreciate that it generally takes less time to press a key than move a mouse.

Buttons and widgets indicate to the user that a function is available, while keystrokes are generally faster way to accomplish it. Anything that could be done with the mouse should always have a keyboard equivalent of some kind. Any application developed should always be tested both with and without mouse to ensure that the application behaves appropriately in both situations.

B.7 FONTS AND TEXT

Many of the rules and conventions that are used when designing the printed page apply for designing the screen layout. Following guidelines will make the applications more readable and usable.

Font families should be used sparingly in an application, as if it were a printed page. As a general rule, one to three font families per screen is plenty. Different sizes, effects and colors can be used to indicate emphasis or function without distracting the user or disturbing the coherence of the application. With thousands of fonts at the developer's disposal, it may be tempting to overuse fonts or to use fancy and esoteric fonts in an application. While this is questionable from the aesthetic standpoint, it also runs the risk that an esoteric font will not be available in the end user's machine or platform and a substitution will be made - which can be very ugly and undesirable. Specialty fonts are problematic for deployment and should be avoided entirely in favor of a bitmapped image. The bitmapped image will display consistently on all graphical platforms while the specialty fonts will usually appear as something unexpected and confusing.

Proportionally spaced fonts are easier to read and better looking than fixed pitch fonts. Because many application development tools can create applications that operate on fixed pitch terminals, this can be problematic for the developer whose application targets both those with graphical and non-graphical displays. It is far easier to guess what an application will look like on a character terminal when using similarly pitched and sized fonts, but unfortunately, this will result in an application that looks like it was ported from a character-terminal, which rarely makes a good impression. If the developer does not have the luxury of ignoring character based terminals altogether, although it can be a difficult task to support both types of environments without resorting to a character grid and fixed pitch fonts, it is not impossible. Whether, the trial-and-error visual design is worth the effort is a decision the developer has to make.

The amount of text that can be displayed in editable fields will vary depending on the font and size used. For a given item, the displayable text is specified by the Item/Width property, while the number of characters is specified independently by the Item/Maximum Length property. Thus means that a field can appear to be too narrow to display the data it contains or that the field can appear too wide and look like it can accept more data, but no more characters will fit. When using a proportional font, either situation can occur, depending on the data in the field. While neither is really desirable, scrolling can be prevented by making the field large enough to accept all the 'W's, the widest character in almost any font.

The use of all upper case letters should be avoided for both text and data. All uppercase is the visual equivalent of SHOUTING and is harder to read. Text can be emphasized using color, special effects like bold, italics, underline, size and different fonts. The use of all uppercase is a poor alternative. When allowing mixed case entry of data, it can be argued that the database treats lower and upper cases differently and it poses additional burden for the developer to ensure that the application treats them equally. The developer must make a decision whether the convenience to the developer is worth the inconvenience to the user—ideally the user's convenience is considered more important.

B.8 COLOR

Color should be used sparingly and with taste. It is important to note that the liberal use of color is further complicated by variations between display terminals. While the better displays boast millions of simultaneous colors, some are limited to as few as sixteen colors. The subtle pastels employed on one display system could translate to an incomprehensible miasma of dots on a lesser display. The effect is not always disastrous, but it is important to at least view the display in sixteen colors if a user will be using it on a terminal with those capabilities. In Windows, it is easy to change the display driver to a driver that only supports sixteen colors. Many laptop machines, and some desktop machines, are only capable of displaying black and white or grey scale. It is generally a good practice to test a color scheme on a monochrome monitor to ensure that there is enough contrast for the display to be readable.

Regardless of the display, the same principles apply. Color can be extremely useful when used sparingly. It can be effectively used to draw the user's attention, as a way of grouping similar items or as

18 ▲ Essentials of Database Management Systems

indicator. It can also be used to indicate the field properties. For example all the mandatory fields in a screen can be in one color while the optional fields can be in another. An application consisting of black text on white background is very readable but looks almost unfinished. A dash of color in a logo or image in the background, some colorful buttons and keywords or titles will lend a lot of character to an application and will help to distinguish it from other applications.

When overused, color can be distracting and annoying. While fluorescent green and pinks will catch people's attention initially, it will also cause fatigue to the eye and people will get tired of the application very easily. If the application is used for flashy demonstrations, it is highly recommended that two color schemes be used—one for demonstrations and one for daily use.

To avoid providing potentially undesirable color scheme, one solution adopted by many developers is to give the users the option of configuring all colors used by the application. While this may be a good idea, most users do not want to spend their time changing the colors of their applications. Users expect such things to be taken care of by the developers. Experimenting with colors is a necessary evil, but once a reasonable color scheme has been established, then use it consistently across the application. The color palette can be used to customize the colors available to an application, such as creating a large number of shades of grey for three dimensional effects, etc. However, if all parts of an application do not share the same palette, a significant amount of time will be spent by the drawing engine in optimizing the palettes used.

B.9 MENUS

On many platforms, pull-down menus follow a simple noun-verb or object-action paradigm, where nouns(objects) are located along the top level and the actions (verbs) that are performed on the objects such as File -> open or File ->close are given as pull-down menus. This design helps in eliminating redundancy and make clear to the user exactly what the menu accomplishes. When the user selects a verb from the menu, he will expect something to happen.

Generally using the noun-verb combination is an excellent way to approach customized application menus, because it helps to clarify exactly what each menu option does. It is always better to follow the Windows menu convention starting with File and ending with Help. Any deviation from this will be irritating to the user. Short-cut key should be provided for most frequently used actions and here also it is better not to deviate much from the standards. For example Ctrl+X is for cut and Ctrl+V is for paste. Changing this will be very annoying to a user who has been using these short-cut keys in all other applications. When arranging menu items, group menu items together that are associated logically. If there are menu options that are not available in some places, and are not shown (instead of disabling it) then place them at the bottom, where they will not disrupt the menu. Users will learn to navigate the through the menu items by its position, so here also consistency should be followed.

Menus should not contain not more than 20 items and that also if they are grouped into blocks of three to five each using a menu separator. If the number of items is more another level can be used to reduce the menu size and there should not be more than three levels of menu items because it will be impossible for the user to find the item they want.

B.10 SUMMARY

We have seen some of the fundamental of graphical user interface design and the various controls used in building the user interface. You can get more information on GUI design from books about GUI design like the following:

- Cooper, A., *About Face 2.0: The Essentials of Interaction Design*, Wiley, 2003.
- Johnson, J., *GUI Bloopers: Don'ts and Do's for Software Developers and Web Designers*, Morgan Kaufmann, 2000.
- Shneiderman, B., Plaisant, C., *Designing the User Interface: Strategies for Effective Human-Computer Interaction (4th Edition)*, Addison Wesley, 2004.
- Spolsky, J., *User Interface Design for Programmers*, Apress, 2001.
- Stone, D., Jarrett, C., Woodroffe, M., and Minocha, S., *User Interface Design and Evaluation*, Morgan Kaufmann, 2005.