FUTURE VISION BIE

By K B Hemanth Raj

Visit : https://hemanthrajhemu.github.io

A Small Contribution Would Support Us.

Dear Viewer,

Future Vision BIE is a free service and so that any Student/Research Personal **Can Access Free of Cost**.

If you would like to say **thanks**, you can make a **small contribution** to the author of this site.

Contribute whatever you feel this is worth to you. This gives **us support** & to bring **Latest Study Material** to you. After the Contribution Fill out this Form (https://forms.gle/tw3T3bUVpLXL8omX7). To Receive a **Paid E-Course for Free**, from our End within 7 Working Days.

Regards

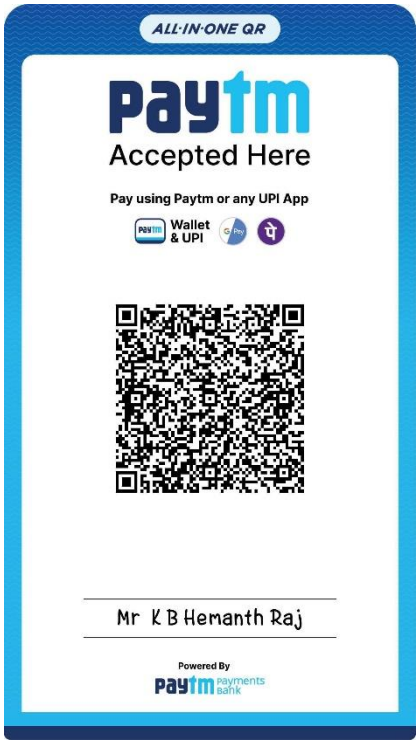
- K B Hemanth Raj (Admin)

Contribution Methods

UPI ID

1. futurevisionbie@oksbi
2. futurevisionbie@paytm

Scan & Pay



Account Transfer

Account Holder's Name: K B Hemanth Raj

Account Number: 39979402438

IFSC Code: SBIN0003982

MICR Code: 560002017

More Info: https://hemanthrajhemu.github.io/Contribution/

Gain Access to All Study Materials according to VTU,
CSE – Computer Science Engineering,
ISE – Information Science Engineering,
ECE - Electronics and Communication Engineering & MORE...

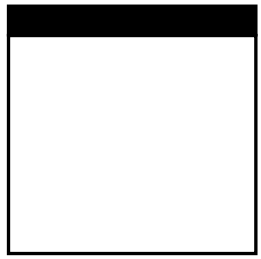
Stay Connected... get Updated... ask your queries...

Join Telegram to get Instant Updates: https://bit.ly/VTU_TELEGRAM

Contact: MAIL: futurevisionbie@gmail.com

INSTAGRAM: www.instagram.com/futurevisionbie/

WHATSAPP SHARE: https://bit.ly/FVBIESHARE



The Essential Guide to User Interface Design

An Introduction to GUI Design Principles and Techniques

Third Edition

Wilbert O. Galitz

Wiley Publishing, Inc.

STEP

5

Select the Proper Kinds of Windows

A window is an area of the screen, usually rectangular in shape, defined by a border that contains a particular view of some area of the computer or some portion of a person's dialog with the computer. It can be moved and rendered independently on the screen. A window may be small, containing a short message or a single field, or it may be large, consuming most or all of the available display space. A display may contain one, two, or more windows within its boundaries. In this step the following is addressed:

- A window's characteristics.
- A window's components.
- A window's presentation styles.
- The types of windows available.
- Organizing window system functions.
- A window's operations.
- Web system frames and pop-up windows.

Window Characteristics

A window is seen to possess the following characteristics:

- A name or title, allowing it to be identified.
- A size in height and width (which can vary).

- A state, accessible or active, or not accessible. (Only active windows can have their contents altered.)
- Visibility — the portion that can be seen. (A window may be partially or fully hidden behind another window, or the information within a window may extend beyond the window's display area.)
- A location, relative to the display boundary.
- Presentation, that is, its arrangement in relation to other windows. It may be tiled, overlapping, or cascading.
- Management capabilities, methods for manipulation of the window on the screen.
- Its highlight, that is, the part that is selected.
- The function, task, or application to which it is dedicated.

The Attraction of Windows

The value of windowing is best seen in the context of a task or job. A person performs a variety of tasks, often in a fairly unstructured manner. A person is asked to monitor and manipulate data from a variety of sources, synthesize information, summarize information, and reorganize information. Things are seldom completed in a continuous time frame. Outside events such as telephone calls, supervisor or customer requests, and deadlines force shifts in emphasis and focus. Tasks start, stop, and start again. Materials used in dealing with the tasks are usually scattered about one's desk, being strategically positioned in the workspace to make handling the task as efficient as possible. This spatial mapping of tools helps people organize their work and provides reminders of uncompleted tasks. As work progresses and priorities change, materials are reorganized to reflect the changes.

Single-screen technology supported this work structure very poorly. Because only one screen of information could be viewed at one time, comparing or integrating information from different sources and on different screens often required extensive use of one's memory. To support memory, a person was often forced to write notes or obtain printed copies of screens. Switching between tasks was difficult and disruptive, and later returning to a task required an extensive and costly restructuring of the work environment.

The appeal of windowing is that it allows the display workspace to mirror the desk workspace much more closely. This dramatically reduces one's short-term memory load. One's ability to do mental calculations is limited by how well one keeps track of one's place, one's interim conclusions and products, and, finally, the results. Windows act as external memories that are an extension of one's internal memory. Windows also make it much easier to switch between tasks and to maintain one's context, because one does not have to reestablish one's place continually. In addition, Windows provide access to more information than would normally be available on a single display of the same size. Overwriting, or placing more important information on top of that of less importance at that moment, does this.

While all the advantages and disadvantages of windows are still not completely understood, windows do seem to be useful in the following ways.

Presentation of Different Levels of Information

Information can be examined in increasing levels of detail. A document table of contents can be presented in a window. A chapter or topic selected from this window can be simultaneously displayed in more detail in an adjoining window. Deeper levels are also possible in additional windows.

Presentation of Multiple Kinds of Information

Variable information needed to complete a task can be displayed simultaneously in adjacent windows. An order-processing system window could collect a customer account number in one window and retrieve the customer's name and shipping address in another window. A third window could collect details of the order, after which another window could present factory availability of and shipping dates for the desired items. Significant windows could remain displayed so that details may be modified as needed prior to order completion. Low inventory levels or delayed shipping dates might require changing the order.

Sequential Presentation of Levels or Kinds of Information

Steps to accomplish a task can be sequentially presented through windows. Successive windows are presented until all the required details are collected. Key windows may remain displayed, but others appear and disappear as necessary. This sequential preparation is especially useful if the information-collection process leads down various paths. An insurance application, for example, will include different types of coverage. A requested type of coverage might necessitate the collection of specific details about that type of coverage. This information can be entered into a window presented to collect the unique data. The windows disappear after data entry, and additional windows appear when needed.

Access to Different Sources of Information

Independent sources of information may have to be accessed at the same time. This information may reside in different host computers, operating systems, applications, files, or areas of the same file. It may be presented on the screen alongside the problem, greatly facilitating its solution. For instance, a writer may have to refer to several parts of a text being written at the same time. Or, a travel agent may have to compare several travel destinations for a particularly demanding client.

Combining Multiple Sources of Information

Text from several documents may have to be reviewed and combined into one. Pertinent information is selected from one window and copied into another.

Performing More Than One Task

More than one task can be performed at one time. While waiting for a long, complex procedure to finish, another can be performed. Tasks of higher priority can interrupt less important ones. The interrupted task can then be resumed without the necessity to "close down" and "restart."

Reminding

Windows can be used to remind the viewer of things likely to be of use in the near future. Examples might be menus of choices available, a history of the path followed or the command choices to that point, or the time of an important meeting.

Monitoring

Changes, both internal and external, can be monitored. Data in one window can be modified and its effect on data in another window can be studied. External events, such as changes in stock prices, out of normal range conditions, or system messages can be watched while another major activity is carried out.

Multiple Representations of the Same Task

The same thing can be looked at in several ways — for example, alternate drafts of a speech, different versions of a screen, or different graphical representations of the same data. A maintenance procedure may be presented in the form of textual steps and illustrated graphically at the same time.

Constraints in Window System Design

Windowing systems, in spite of their appeal and obvious benefits, have failed to completely live up to their expectations. In the past, a windows user interface has been described as "chaotic" because of the great amount of time users must spend doing such things as pointing at tiny boxes in window borders, resizing windows, moving windows, closing windows, and so forth. The problems with windowing systems can be attributed to three factors: historical considerations, hardware limitations, and human limitations.

Historical Considerations

Historically, system developers have been much more interested in solving hardware problems than in user considerations. Because technical issues abound, they have received the most attention. There has been little research addressing design issues and their impact on the usability of window systems. Therefore, few concrete window design guidelines are available to aid designers.

This lack of guidelines makes it difficult to develop acceptable and agreeable window standards. While many companies have developed style guides, they are very general and limited in scope to their products. Standardization is also made more difficult by the complexity and range of alternatives available to the designer. Without user performance data, it is difficult to compare realistically the different alternatives, and design choices become a matter of preference.

Standardization of the interface is also inhibited by other factors. Some software developers, who are proud of their originality, see standards as a threat to creativity and its perceived monetary rewards. Some companies are wary of standards because they fear that other companies are promoting standards that reflect their own approach. Finally, some companies have threatened, or brought, legal action against anyone who adopts an approach similar to their own.

The result is that developers of new systems create another new variation each time they design a product, and users must cope with a new interface each time they encounter a new windowing system.

Hardware Limitations

Many of today's screens are not large enough to take full advantage of windowing capabilities. As a result, many windows are still of "Post-it" dimensions. As already mentioned, there is some evidence that many users of personal computers expand their windows to cover a full screen. Either seeing all the contents of one window is preferable to seeing small parts of many windows or the operational and visual complexity of multiple windows is not wanted.

Also, the slower processing speeds and smaller memory sizes of some computers may inhibit the use of windows. A drain on the computer's resources may limit feedback and animation capabilities, thereby reducing the system's usability. Poor screen resolution and graphics capability may also deter effective use of windows by not permitting sharp and realistic drawings and shapes.

Human Limitations

A windowing system, because it is more complex, requires the learning and using of more operations. Much practice is needed to master them. These window management operations are placed on top of other system operations, and window management can become an end in itself. This can severely detract from the task at hand. In one study comparing full screens with screens containing overlapping windows, task completion times were longer with the window screens, but the non-window screens generated more user errors. After eliminating screen arrangement time, however, task solution times were shorter with windows. The results suggest that advantages for

windows do exist, but they can be negated by excessive window manipulation requirements.

It is also suggested that to be truly effective, window manipulation must occur implicitly as a result of user task actions, not as a result of explicit window management actions by the user.

Other Limitations

Other possible window problems include the necessity for window borders to consume valuable screen space, and that small windows providing access to large amounts of information can lead to excessive, bothersome scrolling.

Where To?

In spite of their problems, windows do have enormous benefits and are here to stay. So, we must cope with their constraints for now and, in the meantime, enjoy the benefits they possess.

Components of a Window

A typical window may be composed of up to a dozen or so elements. Some appear on all windows, others only on certain kinds of windows, or under certain conditions. For consistency purposes, these elements should always be located in the same position within a window. Most windowing systems provide consistent locations for elements in their own windows. Some inconsistencies do exist in element locations between different systems, however, as do some differences in what the elements are named, or what graphic images or icons are chosen to identify them. What follows is a description of typical window components and their purposes, with emphasis on the most popular windowing system, *Microsoft Windows*. Specifically reviewed will be primary windows, secondary windows, and a form of secondary window called the dialog box. An illustration of a primary window is found in Figure 5.1. Illustrations of secondary windows and dialog boxes are illustrated in Figures 5.8 and 5.13. How these different types of windows are used will be described in a later section in this step. A summary of window components for these types of windows is also found in Table 5.1.

Frame

A window will have a frame or border, usually rectangular in shape, to define its boundaries and distinguish it from other windows. While a border need not be rectangular, this shape is a preferred shape for most people. Also, textual materials, which are usually read from left to right, fit most efficiently within this structure. The border comprises a line of variable thickness and color. This variation can be used as an aid in identifying the type of window being displayed. Windows filling an entire screen may use the screen edge as the border. If a window is resizable, it may contain control points for sizing it. If the window cannot be resized, the border coincides with the edge of the window.

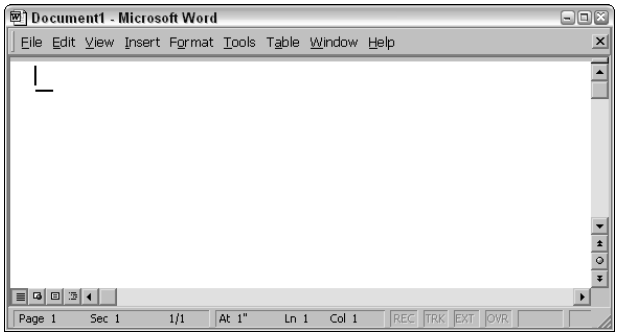


Figure 5.1: Microsoft Windows primary window.

Title Bar

The title bar is the top edge of the window, inside its border and extending its entire width. This title bar is also referred to by some platforms as the *caption*, *caption bar*, or *title area*. The title bar contains a descriptive title identifying the purpose or content of the window. In Microsoft Windows, the title bar may also possess, at the extreme left and right ends, control buttons (described later) for retrieving the system menu and performing window resizing. The title bar also serves as a control point for moving the window and as an access point for commands that apply to a window. For example, as an access point, when a user clicks on the title bar using the secondary mouse button, the pop-up or shortcut menu for the window appears. Pressing the Alt+Spacebar key combination also displays the shortcut menu for the window. Title bars are included on all primary and secondary windows. Title bar text writing guidelines are described in Step 8.

Microsoft recommends that one never place application commands or other controls in the title bar. Doing so may conflict with the special user controls Windows adds for configurations that support multiple languages.

Title Bar Icon

Located at the left corner of the title bar in a primary window, this button is used in Windows to retrieve a pull-down menu of commands that apply to the object in the window. It is a 16×16 version of the icon of the object being viewed. When clicked with the secondary mouse button, the commands applying to the object are presented. Microsoft suggests that

- If the window contains a tool or utility (that is, an application that does not create, load, and save its own data files), a small version of the application's icon should be placed there instead.
- If the application creates, loads, and saves documents or data files and the window represents the view of one of its files, a small version of the icon that represents its document or data file type should be placed there.
- Even if the user has not yet saved the file, display the data file icon rather than the application icon, and again display the data file icon after the user saves the file.

Table 5.1: Microsoft Windows Components

	WINDOWS CONTAINING COMPONENT		
COMPONENT	**PRIMARY**	**SECONDARY**	**DIALOG BOX**
Frame or Border • Boundary to define shape. • If sizable, contains control points for resizing.	X	X	X
Title Bar Text • Name of object being viewed in window. • Control point for moving window.	X	X	X
Title Bar Icon • Small version of icon for object being viewed. • Access point for commands that apply to the object.	X		
Title Bar Buttons • Shortcuts to specific commands.	X	X	X
Close	X	X	X
Minimize/Maximize/Restore	X		
What's This? • Displays context-sensitive Help about any object displayed on window.		X	X
Menu Bar • Provides basic and common application commands.	X		
Status Bar • *An area used to display status information about what is displayed in window.*	X		
Scroll Bar • Standard control to support scrolling.	X		
Size Grip • *Control to resize window, located at right side of status bar.*	X		

Window Sizing Buttons

Located at the right corner of the title bar, these buttons are used to manipulate the size of a window. The leftmost button, the *minimize* button — inscribed with a short horizontal line toward the bottom of the button — is used to reduce a window to its minimum

size, usually an icon. It also hides all associated windows. The *maximize* button — typically inscribed with a large box — enlarges a window to its maximum size, usually the entire screen. When a screen is maximized, the restore button replaces the maximize button, because the window can no longer be increased in size. The restore button — typically inscribed with a pair overlapping boxes — returns a window to the size it had before a minimize or maximize action was performed. A close button — typically inscribed with an X — closes the window. Minimize, maximize, and close buttons are shown in Figure 5.1. These command buttons are graphical equivalents to the actions available through the Title Bar icon.

Sizing buttons are included on primary windows only. All buttons on a primary window's title bar must have equivalent commands on the pop-up or shortcut menu for that window.

When these buttons are displayed, use the following guidelines:

- When a window does not support a command, do not display its command button.
- The Close button always appears as the rightmost button. Leave a gap between it and any other buttons.
- The Minimize button always precedes the Maximize button.
- The Restore button always replaces the Maximize button or the Minimize button when that command is carried out.

What's This? Button

The *What's This?* button, which appears on secondary windows and dialog boxes, is used to invoke the What's This? Windows command to provide contextual Help about objects displayed within a secondary window. When provided, it is located in the upper-right corner of the title bar, just to the left of the close button. It is inscribed with a question mark, as illustrated in Figure 5.2.

On a primary window this command is accessed from the Help drop-down menu. This command may also be included as a button on a toolbar or as a command on a pop-up menu for a specific object. This command is described more fully in Step 9.

Menu Bar

A menu bar is used to organize and provide access to actions. It is located horizontally at the top of the window, just below the title bar. A menu bar contains a list of topics or items that, when selected, are displayed on a pull-down menu beneath the choice. A system will typically provide a default set of menu actions that can be augmented by an application. In the past, some platforms have called the menu bar an *action bar*. Menu bar design guidelines were presented in Step 4. The contents of the menu bar and its pull-downs are determined by the application's functionality and the context in which the user is interacting with it.

Figure 5.2: What's This? button.

Status Bar

Information of use to the user can be displayed in a designated screen area or areas. They may be located at the top of the screen in some platforms and called a *status area*, or at the screen bottom. Microsoft recommends the bottom location and refers to this area as the *status bar*. It is also referred to by other platforms as a *message area* or *message bar*.

Microsoft Windows suggests using the status bar to display information about the current state of what is being viewed in the window, descriptive messages about a selected menu or toolbar button, or other noninteractive information. It may also be used to explain menu and control bar items as the items are highlighted by the user.

Scroll Bars

When all display information cannot be presented in a window, the additional information must be found and made visible. This is accomplished by scrolling the display's contents through use of a scroll bar. A scroll bar is an elongated rectangular container consisting of a scroll area or shaft, a slider box or elevator, and arrows or anchors at each end. For vertical scrolling, the scroll bar is positioned at the far right side of the work area, extending its entire length. Horizontal scrolling is accomplished through a scroll bar located at the bottom of the work area. Scroll bars are more fully described in Step 7.

Split Box

A window can be split into two or more pieces or panes by manipulating a *split box* located above a vertical scroll bar or to the left of a horizontal scroll bar. A split box is sometimes referred to as a *split bar*. A window can be split into two or more separate viewing areas that are called *panes*. Splitting a window permits multiple views of an object. A split window enables the user to

- Examine two parts of a document at the same time.
- Display different, yet simultaneous, views of the same information.

To support the splitting of a window that is not presplit by design, include a split box. The split box should be just large enough for the user to successfully target it with the pointer; the default size of a size handle, such as the window's sizing border, is a good guideline.

Toolbar

Toolbars, illustrated in Figure 5.3, are panels or arrays of choices or commands that must be accessed quickly. They are sometimes called *command bars*. Toolbars are designed to provide quick access to specific commands or options. Specialized toolbars are sometimes referred to as *ribbons*, *toolboxes*, *rulers*, or *palettes*. Toolbars may occupy a fixed position on a window, be movable, or be contained in a pop-up window. The design of toolbars is discussed in Step 7.

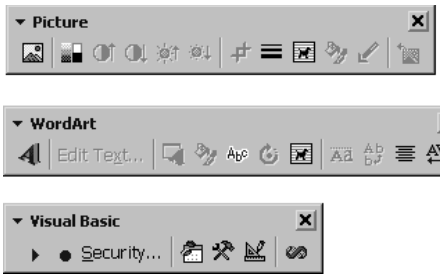


Figure 5.3: Toolbars.

Command Area

In situations where it is useful for a command to be typed into a screen, a command area can be provided. The desired location of the command area is at the bottom of the window. If a horizontal scroll bar is included in the window, position the command area just below it. If a message area is included on the screen, locate the command area just above it.

Size Grip

A size grip is a Microsoft Windows special handle included in a window to permit it to be resized. When the grip is dragged the window resizes, following the same conventions as the sizing border. Three angled parallel lines in the lower-right corner of a window designate the size grip. If the window possesses a status bar, the grip is positioned at the bar's right end. Otherwise, it is located at the bottom of a vertical scroll bar, the right side of a horizontal scroll bar, or the junction point of the two bars. A size grip is shown in the lower-right corner of Figure 5.1.

Work Area

The work area is the portion of the screen where the user performs tasks. It is the open area inside the window's border and contains relevant peripheral screen components such as the menu bar, scroll bars, or message bars. The work area may consist of an open area for typing, or it may contain controls (such as text boxes and list boxes) or customized forms (such as spreadsheets). The work area may also be referred to as the *client area*.

Window Presentation Styles

The presentation style of a window refers to its spatial relationship to other windows. There are two basic styles, commonly called tiled or overlapping. In early windowing days, most systems commonly used one or the other style exclusively, seldom using both at the same time. Now, the user is usually permitted to select the style to be presented on the display.

Tiled Windows

Tiled windows, illustrated in Figure 5.4, derive their name from the common floor or wall tile. Tiled windows appear in one plane on the screen and expand or contract to fill up the display surface, as needed. Most systems provide two-dimensional tiled windows, adjustable in both height and width. Some less-powerful systems, however, are only one-dimensional, the windows being adjustable in only one manner (typically the height). Tiled windows, the first and oldest kind of window, are felt to have these advantages:

- The system usually allocates and positions windows for the user, eliminating the necessity to make positioning decisions.
- Open windows are always visible, eliminating the possibility of them being lost and forgotten.
- Every window is always completely visible, eliminating the possibility of information being hidden.
- They are perceived as less complex than overlapping windows, possibly because there are fewer management operations or they seem less "magical."
- They are easier, according to studies, for novice or inexperienced people to learn and use.
- They yield better user performance for tasks where the data requires little window manipulation to complete the task.

Perceived disadvantages include the following:

- Only a limited number can be displayed in the screen area available.
- As windows are opened or closed, existing windows change in size. This can be annoying.
- As windows change in size or position, the movement can be disconcerting.
- As the number of displayed windows increases, each window can get very tiny.
- The changes in sizes and locations made by the system are difficult to predict.
- The configuration of windows provided by the system may not meet the user's needs.
- They are perceived as crowded and more visually complex because window borders are flush against one another, and they fill up the whole screen. Crowding is accentuated if borders contain scroll bars or control icons. Viewer attention may be drawn to the border, not the data.
- They permit less user control because the system actively manages the windows.

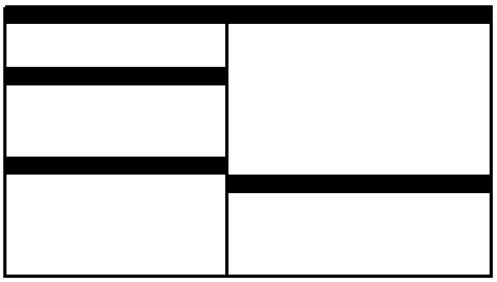


Figure 5.4: Tiled windows.

Overlapping Windows

Overlapping windows, illustrated in Figure 5.5, may be placed on top of one another like papers on a desk. They possess a three-dimensional quality, appearing to lie on different planes. Users can control the location of these windows, as well as the plane in which they appear. The sizes of some types of windows may also be changed. Most systems today normally use this style of window. They have the following advantages:

- Visually, their look is three-dimensional, resembling the desktop that is familiar to the user.
- Greater control allows the user to organize the windows to meet his or her needs.
- Windows can maintain larger sizes.
- Windows can maintain consistent sizes.
- Windows can maintain consistent positions.
- Screen space conservation is not a problem, because windows can be placed on top of one another.
- There is less pressure to close or delete windows no longer needed.
- The possibility exists for less visual crowding and complexity. Larger borders can be maintained around window information, and the window is more clearly set off against its background. Windows can also be expanded to fill the entire display.
- They yield better user performance for tasks where the data requires much window manipulation to complete the task.

Disadvantages include the following:

- They are operationally much more complex than tiled windows. More control functions require greater user attention and manipulation.
- Information in windows can be obscured behind other windows.
- Windows themselves can be lost behind other windows and be presumed not to exist.

- That overlapping windows represent a three-dimensional space is not always realized by the user.
- Control freedom increases the possibility for greater visual complexity and crowding. Too many windows, or improper offsetting, can be visually overwhelming.

Cascading Windows

A special type of overlapping window has the windows automatically arranged in a regular progression. Each window is slightly offset from others, as illustrated in Figure 5.6. Advantages of this approach include the following:

- No window is ever completely hidden.
- Bringing any window to the front is easier.
- It provides simplicity in visual presentation and cleanness.

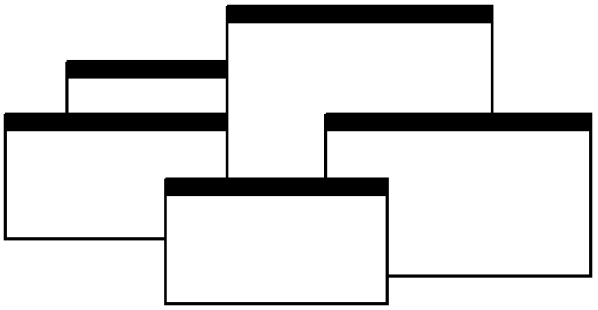


Figure 5.5: Overlapping windows.

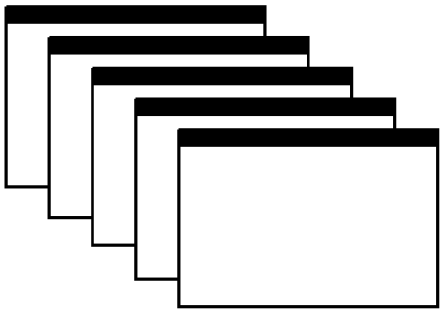


Figure 5.6: Cascading windows.

Picking a Presentation Style

- Use tiled windows for
  - — Single-task activities.
  - — Data that needs to be seen simultaneously.
  - — Tasks requiring little window manipulation.
  - — Novice or inexperienced users.
- Use overlapping windows for
  - — Switching between tasks.
  - — Tasks necessitating a greater amount of window manipulation.
  - — Expert or experienced users.
  - — Unpredictable display contents.

Tiled windows. Tiled windows seem to be better for single-task activities and data that must be seen simultaneously. A study found that tasks requiring little window manipulation were carried out faster using tiled windows. They also found that novice users performed better with tiled windows, regardless of the task.

MAXIM **An activity will be pursued only if its benefits are equal to or greater than the cost.**

Overlapping windows. Overlapping windows seem to be better for situations that necessitate switching between tasks. A research study concluded that tasks requiring much window manipulation could be performed faster with overlapping windows but only if user window expertise existed. For novice users, tasks requiring much window manipulation were carried out faster with tiled windows. Therefore, the advantage to overlapping windows comes only after a certain level of expertise is achieved. Overlapping windows are the preferred presentation scheme.

Types of Windows

People's tasks must be structured into a series of windows. The type of window used will depend on the nature and flow of the task. Defining standard window types is again difficult across platforms because of the varying terminology and definitions used by different windowing systems, and changes in terminology for new versions of systems. For simplicity, the Microsoft Windows windowing scheme will be described. Summarized are a description of the window, its purpose, and its proper usage. Any other platform's windows may not behave exactly as presented, and some platform windows may exhibit characteristics common to more than one of the described window types.

Primary Window

- Proper usage
  - — Use to represent an independent function or application.
  - — Use to present constantly used window components and controls.
    - Menu bar items that are
      - Used frequently.
      - Used by most, or all, primary or secondary windows.
    - Controls used by dependent windows
  - — Use for presenting information that is continually updated, such as date and time.
  - — Use for providing context for dependent windows to be created.
  - — Do not
    - Divide an independent function into two or more primary windows.
    - Present unrelated functions in one primary window.

The *primary* window is the first one that appears on a screen when an activity or action is started. It is required for every function or application, possessing a menu bar and some basic action controls, as previously described. It should present the framework for the function's commands and data, and provide top-level context for dependent windows. It has also been variously referred to as the *application* window or the *main* window. In addition, it may be referred to as the *parent* window if one or more *child* windows exist. A Microsoft Windows primary window is shown in Figure 5.7.

The primary window is the main focal point of the user's activities and should represent an independent function. Avoid dividing an independent function into two or more primary windows, and avoid presenting unrelated functions in a single primary window. This tends to confuse people.

Independent functions should begin in a primary window. A primary window should contain constantly used window components such as frequently used menu bar items and controls (for example, control bars) used by dependent windows. Also include in a primary window continually updated information such as the date and time. The components of a primary window are summarized in Table 5.2.

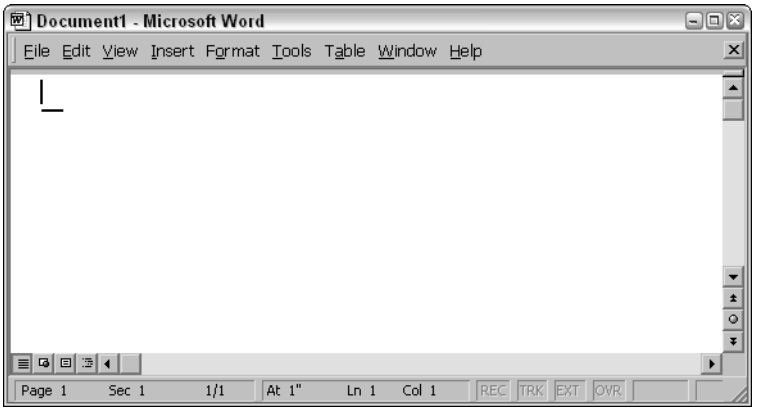


Figure 5.7: Microsoft Windows primary window.

Secondary Windows

- Proper usage:
  - — For performing subordinate, supplemental, or ancillary actions that are
    - Extended or more complex in nature.
    - Related to objects in the primary window.
  - — For presenting frequently or occasionally used window components.
- Important guidelines:
  - — Should typically not appear as an entry on the taskbar.
  - — A secondary window should not be larger than 263 dialog units x 263 dialog units.

Secondary windows are supplemental windows. Secondary windows may be dependent upon a primary window or displayed independently of the primary window. They structurally resemble a primary window, possessing some of the same action controls (Close button) and possibly a What's This? button.

A *dependent* secondary window is one common type. It can only be displayed from a command on the interface of its primary window. It is typically associated with a single data object, and appears on top of the active window when requested. It is movable, and scrollable. If necessary, it uses the primary window's menu bar. Most systems permit the use of multiple secondary windows to complete a task. In general, dependent secondary windows are closed when the primary window closes, and hidden when their primary window is hidden or minimized.

An *independent* secondary window can be opened independently of a primary window — for example, a property sheet displayed when the user clicks the Properties command on the menu of a desktop icon. An independent secondary window can be typically closed without regard to the state of any primary window unless there is an obvious relationship to the primary window.

A Microsoft Windows secondary Window is illustrated in Figure 5.8.

Proper usage. Although secondary windows share many characteristics with primary windows, they also differ from primary windows in behavior and use. Secondary windows are used to perform supplemental or subordinate tasks, or tasks that are extended in nature. Frequently and occasionally used window components should also be presented in secondary windows. Microsoft Windows possesses several types of secondary windows called *dialog boxes*, *property sheets*, *property inspectors*, *message boxes*, *palette windows*, and *pop-up* windows.

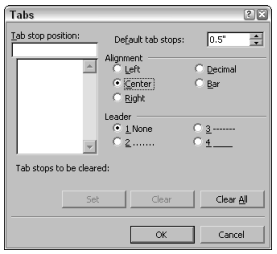


Figure 5.8: Microsoft Windows secondary window.

Guidelines. A secondary window should typically not appear as an entry on the taskbar. Secondary windows obtain or display supplemental information that is usually related to the objects that appear in a primary window.

A secondary window is typically smaller than its associated primary window and smaller than the minimum display resolution. Microsoft recommends not displaying any secondary window larger than 263 dialog units x 263 dialog units. Microsoft defines size and location of user-interface elements not in pixels but in *dialog units* (DLUs), a device-independent unit of measure.

- One horizontal DLU is equal to one-fourth of the average character width for the current system font.
- One vertical DLU is equal to one-eighth of the average character height for the current system font.

These sizes keep the window from becoming too large to display at most resolutions. However, they still provide reasonable space to display supportive information, such as Help information, that applies. The components of a secondary window are summarized in Table 5.2.

Table 5.2: Microsoft Window Types and Components

PRIMARY WINDOW	
Purpose:	To perform a major interaction.
Components:	Frame or border. Title bar. —Access point for commands that apply to the window, with commands displayed in a pop-up menu. Title Bar icon. —Small version of the icon of the object being viewed. —Access point for commands that apply to the object being displayed in the window, with commands displayed in a pop-up window. Title bar text. Title bar buttons to close/minimize/maximize /restore a window. Menu bar. Status bar. Scroll bar. Size grip.
SECONDARY WINDOWS	
Purpose:	To obtain or display supplemental information related to the objects in the primary window.
Components:	Frame or border. Title bar. Title bar text. Close button. What's This? button. —Context-sensitive Help about components displayed in the window; this is optional.

Table 5.2 *(continued)*

SECONDARY WINDOWS	
Kinds:	Modal and modeless.
Dialog Boxes	
Purpose:	To obtain additional information needed to carry out a particular command or task.
Description:	Secondary window. Contains the following common dialog box interfaces: — Open/Replace/Find. — Save As /Print/Print Setup. — Page Setup/Font/Color.
Property Inspectors	
Purpose:	To display the most common or frequently accessed properties of a current selection, usually of a particular type of object.
Description:	A modeless secondary window. Typically modal with respect to the object for which it displays properties.
Usage:	Displayed when requested from selected object.
Property Sheets	
Purpose:	For presenting the complete set of properties for an object.
Description:	A modeless secondary window. Typically modal with respect to the object for which it displays properties.
Usage:	Displayed when requested from selected object.
Message Boxes	
Purpose:	To provide information about a particular situation or condition.
Description:	Secondary window. Types of message boxes: — Information/Warning/Critical.
Palette Windows	
Purpose:	To present a set of controls such as palettes or toolbars.
Description:	Modeless secondary window.
Pop-Up Windows	
Purpose:	To display additional information when an abbreviated form of the information is the main presentation.
Description:	Secondary window. Does not contain standard secondary window components such as title bar and close button. Example: ToolTip.

Modal and Modeless

- Modal:
  - — Use when interaction with any other window must not be permitted.
  - — Use for
    - Presenting information; for example, messages (sometimes called a message box).
    - Receiving user input; for example, data or information (sometimes called a prompt box).
    - Asking questions; for example, data, information, or directions (sometimes called a question box).
  - — Use carefully because it constrains what the user can do.
- Modeless:
  - — Use when interaction with other windows must be permitted.
  - — Use when interaction with other windows must be repeated.

A secondary window can be modal or modeless.

Modal. Most secondary windows will be *modal*. Modal windows will not permit interaction with another window until the current dialog is completed. It remains displayed until the appropriate action is taken, after which it is removed from the screen. Modal dialog boxes typically request critical information or actions that must be reacted to before the dialog can continue. Because modal dialog boxes constrain what the user can do, limit their use to situations in which additional information is required to complete a command or when it is important to prevent any further interaction until satisfying a condition.

Modeless. A *modeless* dialog box permits the user to engage in parallel dialogs. Switching between the box and its associated window is permitted. Other tasks may be performed while a modeless dialog box is displayed, and it may be left on the screen after a response has been made to it. Actions leading to a modeless dialog box can be canceled, causing the box to be removed from the screen.

Use a modeless dialog box when interaction with a primary window or another secondary window must be permitted, for example, during the accessing of the Help function. Also, use a modeless dialog box when interaction with other windows must be repeated; for example, in a word search operation.

Cascading and Unfolding

- Cascading
  - — Purpose:
    - To provide advanced options at a lower level in a complex dialog.
  - — Guidelines:
    - Provide a command button leading to the next dialog box with a "To a Window" indicator, an ellipsis (. . .).
    - Present the additional dialog box in cascaded form.

- Provide no more than two cascades in a given path.
- Do not cover previous critical information.
  — Title Bar.
  — Relevant displayed information.
- If independent, close the secondary window from which it was opened.

■ Unfolding
— Purpose:
- To provide advanced options at the same level in a complex dialog.

— Guidelines:
- Provide a command button with an expanding dialog symbol (>>).
- Expand to right or downward.

Access to additional options can be accomplished by inclusion of a command button that opens another secondary window. These multiple secondary windows needed to complete a task may be presented in two forms, cascading or expanding.

Cascading. A *cascading* window keeps the original window displayed, with the dependent window displayed on top, offset slightly to the right and below the original secondary window. A cascade, illustrated in Figures 5.9 and 5.10, is generally used when advanced options at a lower level in a complex dialog must be presented. An indication that the dialog will be cascading is signaled by an ellipsis placed in the command button used to display the additional dialog box. Because of the confusion that can develop with too many cascades, restrict the number of cascades to no more than two in a given path. Do not cover information on the upper-level dialog boxes that may have to be referred to, such as box title bars and other critical or relevant information. If the cascaded window is independent in its operation, close the secondary window from which it was opened and display only the new window.

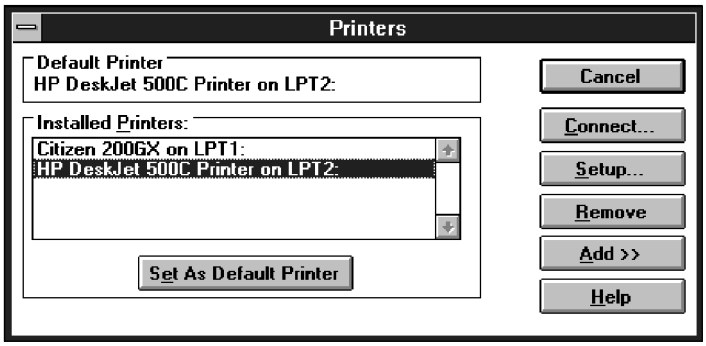


Figure 5.9: Printers secondary window with Connect cascade button.

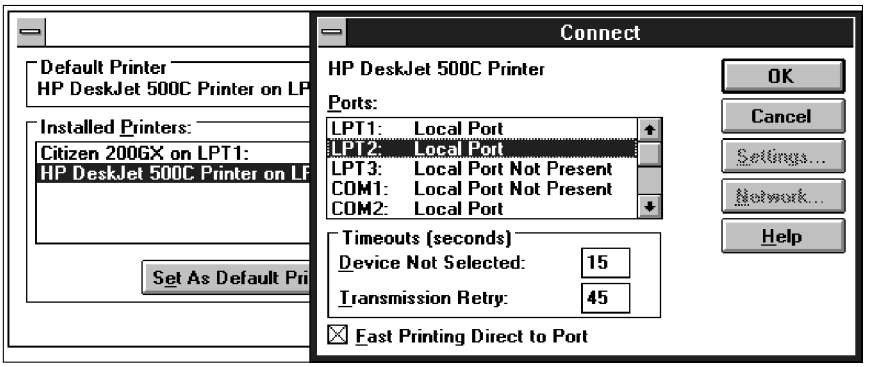


Figure 5.10: Cascading Connect secondary window.

Unfolding. An *unfolding* secondary window expands to reveal additional options, a form of progressive disclosure. Unfolding windows, sometimes called *expanding* windows, are generally used to provide advanced options at the same level in a complex dialog. They are good alternatives when the interface contains a fixed set of options or controls that seldom need to be accessed. An unfolding window is illustrated in Figures 5.11 and 5.12. An indication that the dialog will be expanding is signaled by a double arrow (>>) placed in the command button used to display the additional dialog box. Expand the box to right, preferably, or downward if screen space constraints exist. As an option, the same button can be used to "refold" the additional part of the window.

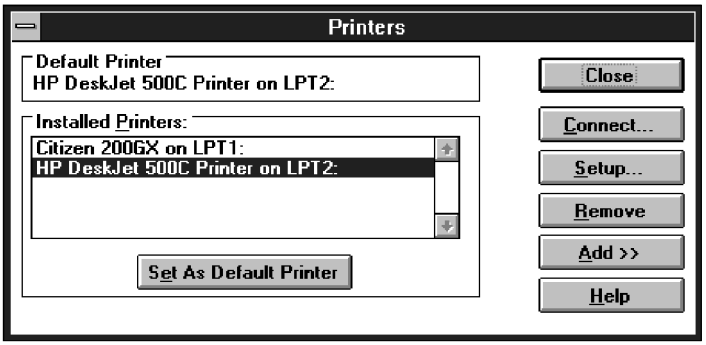


Figure 5.11: Printers secondary window with Add >> unfolding button.

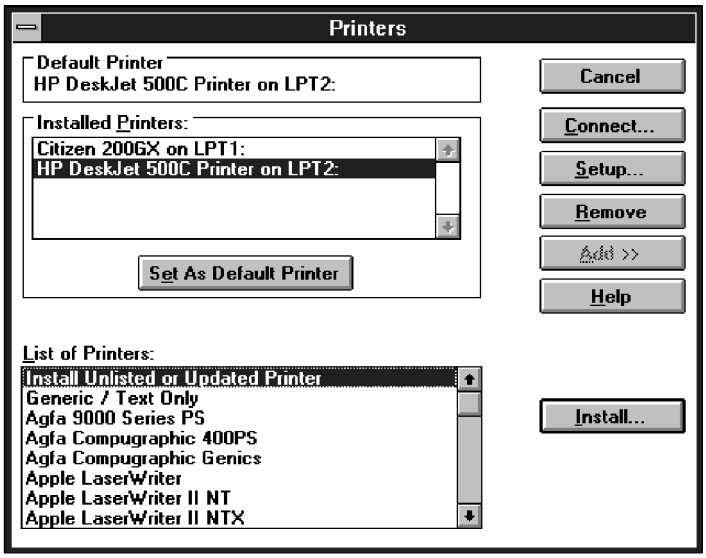


Figure 5.12: Unfolded Printers secondary window.

Dialog Boxes

- Use for presenting brief messages.
- Use for requesting specific, transient actions.
- Use for performing actions that
  - — Take a short time to complete.
  - — Are not frequently changed.
- Command buttons to include
  - — OK.
  - — Cancel.
  - — Others as necessary.

Dialog boxes are used to extend and complete an interaction within a limited context. Dialog boxes are always displayed from another window, either primary or secondary, or another dialog box. They may appear as a result of a command button being activated or a menu choice being selected, or they may be presented automatically by the system when a condition exists that requires the user's attention or additional input. They may possess some basic action controls (Close button and possibly a What's This? button), but do not have a menu bar. A Microsoft Windows dialog box is illustrated in Figure 5.13.

Most windowing systems provide standard dialog boxes for common functions, some examples being Open, Save As, and Print. Many platforms also recommend a set of standard command buttons for use in the various kinds of dialog boxes, such as OK, Cancel, and so on. Dialog boxes are of two types, modal and modeless, as recently described. They may also cascade or unfold.

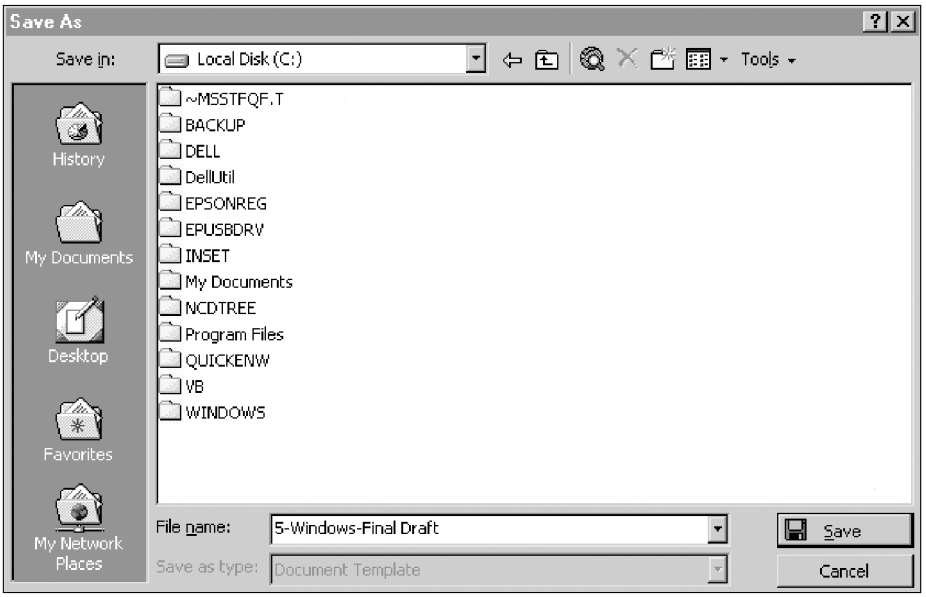


Figure 5.13: Microsoft Windows dialog box.

Uses. Dialog boxes are used for presenting brief amounts of information or to request specific transient actions. Dialog box actions will usually be those that do not occur frequently.

Command buttons. Dialog boxes commonly include OK and Cancel command buttons. OK and Cancel buttons work best in dialog boxes that allow the user to set the parameters for a particular command. OK is typically defined as the default command button when the dialog box window opens. Other command buttons may be included in a dialog box in addition to or instead of the OK and Cancel buttons. Follow the design conventions for command buttons found in Step 7.

Property Sheets and Property Inspectors

The properties of an object in an interface can be displayed in a variety of ways. For example, the image and name of an icon on the desktop reflect specific properties of that object, as do other interface components such as toolbars, status bars, and even scroll bars. Secondary windows provide two other techniques for displaying properties, *property sheets* and *property inspectors.*

Property Sheets

- Use for presenting the complete set of properties for an object.
- Categorize and group within property pages, as necessary.
  — Use tabbed property pages for grouping peer-related property sets.

— The recommended sizes for property sheets are
  - 252 DLUs wide×218 DLUs high
  - 227 DLUs wide×215 DLUs high
  - 212 DLUs wide×188 DLUs high

— Command buttons to include
  - OK.
  - Cancel.
  - Apply.
  - Reset.
  - Others as necessary.

— For single property sheets, place the commands on the sheet.
— For tabbed property pages, place the commands outside the tabbed pages.

Use. A property sheet is the most common way to present an object's complete set of properties in a secondary window. A property sheet is a modeless secondary window that displays the user-accessible properties of an object, properties that may be viewed but not necessarily edited. A single page property sheet is illustrated in Figure 5.14.

Property pages. Because there can be many properties for an object and the object's context, the categorization and grouping of properties within sets may be necessary. A technique for supporting navigation to groups of properties in a property sheet is a tabbed *property page*, where each set of properties is presented within the window as a separate page. Each page tab is labeled with the name of the set, as shown in Figure 5.15. Use tabbed property pages for grouping peer-related property sets. Tabs are described in Step 7.

Size. The sizes recommended for property sheets by Microsoft shown earlier will create a window smaller than its associated window and smaller than the minimum display resolution.

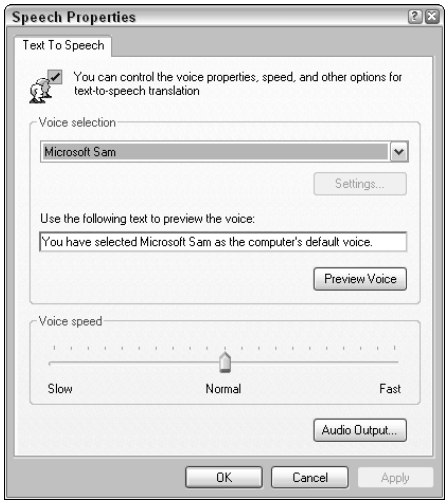


Figure 5.14: Microsoft Windows property sheet.

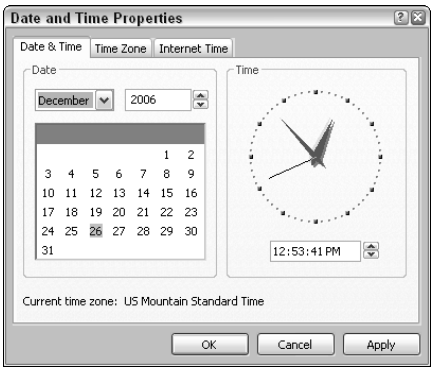


Figure 5.15: Microsoft Windows property sheet tabbed pages.

Command buttons. Property sheets typically allow the values for a property to be changed, and then applied. Include the following common command buttons for handling the application of property changes. For common property sheet transaction buttons, use OK, Cancel, and Apply. A Reset command button to cancel pending changes without closing the window can also be included. Other command buttons can be included in property sheets. Avoid including a Help command button. If a Help button seems necessary, the best solution is to simplify the window.

Command buttons on *tabbed property* pages should be located outside of the tabbed page but still within the window. Buttons placed on a page imply that the action being performed applies *only* to that page. Buttons outside the pages imply the action performed applies to *all* pages. This is the desired positioning because, most often, the tabs are considered by the user as simple grouping or navigation techniques. If the properties are to be applied on a page-by-page basis, however, then place the command and buttons on the property pages, and always in the same location on each page. When the user switches pages without selecting a command button, any property value changes for that page are applied. In these situations, it is useful to prompt the user by displaying a message box that asks whether to apply or discard any changes made.

Property Inspectors

- Use for displaying only the most common or frequently accessed object properties.
- Make changes dynamically.

Use. Display only the most common or frequently accessed properties in a property inspector. Properties of an object are displayed by using a dynamic viewer or browser that reflects the properties of the current selection. A property inspector differs from a property sheet. Even when a property sheet window is modeless, the window is typically modal with respect to the object for the properties being displayed. If the user selects another object, the property sheet continues to display the properties of the original object. A property inspector, on the other hand, always reflects the current selection.

Figure 5.16: Microsoft Windows property inspector.

A palette window (described shortly) or a toolbar is used to create a property inspector, as shown in Figure 5.16. An even better alternative is to use a palette window that the user can also configure. Another control in a property inspector can be used to enable the user to display the properties of various objects in the primary window. For example, as the first control in the property inspector, include a drop-down list box that displays the name of the object being viewed. To view another object's properties within the inspector, the object is selected in the drop-down list box.

Dynamic changes. Changes a user makes in a property inspector should be made dynamically. That is, the property value in the selected object should be changed as soon as the user makes the change in the related property control.

Property inspectors and property sheets are not exclusive interfaces. Both can be included in an interface. The most common or frequently accessed properties can be displayed in a property inspector and the complete set in the property sheet. Multiple property inspectors can also be included, each optimized for managing certain types of objects. An interface's behavior can also be changed between that of a property sheet and that of a property inspector. A control can be provided that "locks" its view, making it modal to the current object, rather than tracking the entire selection.

Message Boxes

- Use for displaying a message about a particular situation or condition.
- Command buttons to include
  - — OK.
  - — Cancel.
  - — Help.
  - — Yes and No.
  - — Stop.
  - — Buttons to correct the action that caused the message box to be displayed.
- Enable the title bar close box only if the message includes a cancel button.
- Designate the most frequent or least destructive option as the default command button.

Use. A message box, as illustrated in Figure 5.17, is a secondary window that displays a message about a particular situation or condition.

Figure 5.17: Microsoft Windows message box.

Command buttons. Typically, message boxes contain only command buttons with the appropriate responses or choices offered to the user. The command buttons used should allow the message box interaction to be simple and efficient. Microsoft suggests providing the following:

- If a message requires no choices to be made but only acknowledgment, include an OK button and, optionally, a Help button.
- If the message requires the user to make a choice, include a command button for each option.
- Include OK and Cancel buttons only when the user has the option of continuing or stopping the action.
- Use Yes and No buttons when the user must decide how to continue.
- If these choices are too ambiguous, label the command buttons with the names of specific actions, for example, Save and Delete.

Command buttons to correct the action that caused the message box to be displayed can also be included in a message box. For example, if the message box indicates that the user must switch to another application window to take corrective action, a button that opens that application window can also be included.

Stop. If Cancel is used as a command button in a message box, remember that to users, cancel implies that the state of the process or task that started the message is being restored. If you use Cancel to interrupt a process and the state cannot be restored, use Stop instead.

Help. A Help button can be included in a message box for messages needing more detail. This allows the message text to be more succinct.

If other command buttons are needed, consider the potential increase in complexity that their inclusion will cause.

Close box. Enable the title bar Close box only if the message includes a Cancel button. Otherwise, the meaning of the Close operation may be ambiguous.

Default. Designate the most frequent or least destructive option as the default command button.

A definition of message types and guidelines for writing messages are found in Step 8.

Palette Windows

- Use to present a set of controls.
- Design as resizable.
  — Alternately, design them as fixed in size.

Use. Palette windows are modeless secondary windows that present a set of controls, as shown in Figure 5.18. Palette windows are distinguished by their visual appearance, a collection of images, colors or patterns. The title bar for a palette window is shorter and includes only a close button.

Sizing. A palette window can be defined as fixed in size, or, more typically, resizable by the user. Two techniques should indicate when the window is resizable: changing the pointer image to the size pointer, and placing a *size* command in the window's shortcut menu. Preserve the window's size and position so the window can be restored if it, or its associated primary window, is closed and then reopened.

Pop-Up Windows

- Use pop-up windows to
  — Display additional information when an abbreviated form of the information is the main presentation.
  — Collect secondary information.
  — Display textual labels for graphical controls.
  — Display context-sensitive Help information.
- Present the pop-up at the front of the screen.
- Make the pop-up one-quarter to one-third of the window size.
- Provide OK or Save and Cancel buttons.
- Never display unsolicited pop-up windows.

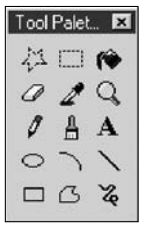


Figure 5.18: Microsoft Windows palette window.

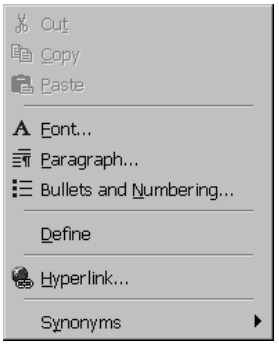


Figure 5.19: Microsoft Windows pop-up window.

Pop-up windows can be used to display additional information when an abbreviated form of the information is the main presentation technique. Other examples of pop-up windows used to display contextual information are *ToolTips* and *balloon tips* that provide the names for controls in graphical toolbars. Pop-up windows are also used to collect secondary information whenever the user's flow through an application should not be interrupted, and to provide context-sensitive Help information.

Always present a pop-up at the front of the screen so it will not be missed, especially if it is reused. The pop-up should be a quarter to a third of a window size. If it is too small it may not be seen; too large and it will cover too much of the screen. Always provide OK or Save and Cancel buttons to remind people of the methods for getting rid of the pop-up. Displaying unsolicited pop-up windows are aggravating and distracting to people. Never display unsolicited windows. Wait for people to take actions necessitating the pop-up.

Organizing Window Functions

Information and functions must be presented to people when and where they need them. Proper organization and support of tasks by windows will be derived only through a thorough and clear analysis of user tasks.

Window Organization

- Organize windows to support user tasks.
- Support the most common tasks in the most efficient sequence of steps.
- Use primary windows to
  - — Begin an interaction and provide a top-level context for dependent windows.
  - — Perform a major interaction.
- Use secondary windows to
  - — Extend the interaction.
  - — Obtain or display supplemental information related to the primary window.

- Use dialog boxes for
  — Infrequently used or needed information.
  — "Nice-to-know" information.

People most often think in terms of tasks, not functions or applications. Windows must be organized to support this thinking. The design goal is to support the most common user tasks in the most efficient manner or fewest steps. Less frequently performed tasks are candidates for less efficiency or more steps.

Mayhew (1992) suggests that poor functional organization usually occurs because of one of, or a combination of, these factors:

- Emphasis on technical ease of implementation rather than proper analysis of user tasks.
- Focus on applications, features, functions, or data types instead of tasks.
- Organization of the design team into applications, with little cross-team communication.
- Blindly mimicking the manual world and carrying over manual inefficiencies to the computer system.

Emphasis on implementation ease puts the needs of the designer before the needs of the customer. Focusing on tasks conforms to the model of how people think. Application orientation imposes an unnatural boundary between functions, and lack of cross-team communication seldom yields consistent task procedures. Mimicking "what is" will never permit the capabilities of the computer system to be properly exploited.

MAXIM **The information to make a decision must be there when the decision is needed.**

Recommended usages for the various window types are summarized in the above guidelines. These recommendations were discussed more fully earlier in this chapter.

Number of Windows

- Minimize the number of windows needed to accomplish an objective.

A person does not work with windows simply for the joy of working with windows. Windows are a means to an end, a method of accomplishing something. Multiple windows on a display, as discussed elsewhere in this text, can be confusing, can increase the load on the human visual system, or may be too small to effectively present what needs to be contained within them.

Guidelines that appeared in early stages of window evolution concerning the maximum number of windows that a person could deal with were quite generous, a limit of seven or eight being suggested. As experience with windows has increased, these numbers have gradually fallen. One study found the mean number of windows maintained for experienced users was 3.7. Today, based on expressions of window users, a recommendation of displaying no more than two or three at one time seems most realistic. The guidelines on limitations for items like cascades (1–2) reflect today's feelings. The exact number of windows a person can effectively deal with at one time will ultimately depend on both the capabilities of the user and the characteristics of the task. Some users and situations may permit handling of more than three windows; for other users and situations, three windows may be two too many.

The general rule: Minimize the number of windows used to accomplish an objective. Use a single window whenever possible. Consider, however, the user's task. Don't clutter up a single window with rarely used information when it can be placed on a second, infrequently used, window.

Sizing Windows

- Provide large-enough windows to
  - — Present all relevant and expected information for the task.
  - — Avoid hiding important information.
  - — Avoid crowding or visual confusion.
  - — Minimize the need for scrolling.
    - But use less than the full size of the entire screen.
- If a window is too large, determine
  - — Is all the information needed?
  - — Is all the information related?
- Otherwise, make the window as small as possible.
  - — Optimum window sizes:
    - For text, about 12 lines.
    - For alphanumeric information, about 7 lines.

Larger windows seem to have these advantages:

- They permit displaying of more information.
- They facilitate learning: Data relationships and groupings are more obvious.
- Less window manipulation requirements exist.
- Breadth is preferred to depth (based on menu research).
- More efficient data validation and data correction can be performed.

Disadvantages include the following:

- Longer pointer movements are required.
- Windows are more crowded.
- More visual scanning is required.

- Other windows more easily obscure parts of the window.
- It is not as easy to hide inappropriate data.

Always provide large enough windows to present all the relevant and expected information for the task. Never hide important or critical information, and minimize the need for scrolling. A study has found that very small windows requiring a significant amount of scrolling appear to increase decision-making time. Scrolling is also a cumbersome operation. To avoid scrolling, consider using unfolding dialog boxes, cascading windows, or a tab control. Avoid, however, making a window's default size the full size of the display. Doing so leads to any underlying windows being completely hidden from the user's view. The option to maximize primary and secondary windows always exists.

If, through analysis and design, a window appears to be too large, determine the following:

- Is all the information needed?
- Is all the information related?

Important, critical, or frequently used information must be maintained on a screen, but perhaps information exists that is needed or used infrequently, for example, only 10 to 20 percent of the time. This kind of information is a good candidate for placement on another window or dialog box. Perhaps information is included on a screen that is not related to the task being performed. This kind of information should be located with the information to which it is related. As a last resort, consider shortening some window control captions or other included window text to achieve a proper fit.

At least two studies have looked at optimum window sizes. Procedural text in window sizes of 6, 12, and 24 lines were evaluated by one study. Fastest and most accurate completion occurred with the 12-line window. The retrieval of alphanumeric information was compared in 7-, 13-, and 19-line windows in another study. A 7-line window was found to be more than adequate.

Window Placement

- Considerations:
  - — In placing a window on the display, consider
    - The use of the window.
    - The overall display dimensions.
    - The reason for the window's appearance.
- General:
  - — Position the window so it is entirely visible.
  - — If the window is being restored, place the window where it last appeared.
  - — If the window is new, and a location has not yet been established, place it
    - At the point of the viewer's attention, usually the location of the pointer or cursor.
    - In a position convenient to navigate to.
    - So that it is not obscuring important or related underlying window information.

— For multiple windows, give each additional window its own unique and discernible location.
  • A cascading presentation is recommended.
— In a multiple-monitor configuration, display the secondary window on the same monitor as its primary window.
— If none of the above location considerations apply, then
  • Horizontally center a secondary window within its primary window just below the title bar, menu bar, and any docked toolbars.
— If the user then moves the window, display it at this new location the next time the user opens the window.
  • Adjust it as necessary to the current display configuration.
— Do not let the user move a window to a position where it cannot be easily repositioned.

■ Dialog boxes:
— If the dialog box relates to the entire system, center it on screen.
— Keep key information on the underlying screen visible.
— If one dialog box calls another, make the new one movable whenever possible.

Considerations. In placing a window on the display, consider how the window is used in relation to other windows, the overall dimensions of the display, and the reason that the window is being presented.

General. First, locate the window so it is fully visible. If the window is being restored, locate it where it last appeared. If the window is new and the location has not yet been established, place it at the point of the viewer's attention. This will usually be the location of the pointer or cursor. Also, place the window in a position where it will be convenient to navigate to, and where it will not obscure important underlying screen information. Preferred positions are essentially below and right. The suggested order of placement is below right, below, right, top right, below left, top, left, top left.

In a multiple-monitor configuration, display the secondary window on the same monitor as its primary window. If none of these situations applies, horizontally center a secondary window within the primary window, just below the title bar, menu bar, and any docked toolbars. Give each additional window its own unique location. A cascading presentation, from upper left to lower right is recommended. If the user then moves the window, display it at this new location the next time the user opens the window, adjusted as necessary for the current display configuration. Do not let the user move a window to a position where it cannot be easily repositioned.

Dialog boxes. If a dialog box relates to the entire system, center it on display, keeping key information on an underlying window visible. If one dialog box calls another, make the new one movable whenever possible.

The Web and the Browser

An entity called the browser is the user interface for the World Wide Web. In structure it resembles a standard window in many aspects, but differs in others. The most popular browser is Microsoft's Internet Explorer, which, as of this writing, is the one utilized by about 83 percent of Web users. The second most popular is Mozilla's Firefox, currently being used by about 11 percent of users. Other available browsers include Netscape, Lynx, Opera, and Safari. This section describes the major components of a browser using Internet Explorer as the model. Some unique features of Firefox will also be described.

Browser Components

In appearance, Internet Explorer presents the same visual style found in Windows. It also contains many of the same standard components, including a title bar, menu bar, toolbar, sizing buttons, size grip, command field (called URL address bar), status bar, enclosing frame border, and if necessary, a scroll bar. It also includes a content area for displaying information, buttons, data fields, and so forth.

Standard toolbar actions include the following:

- Back - Displays the previous page viewed.
- Forward - Displays the next page in the viewing sequence if already viewed.
- Stop - Stops page from being loaded.
- Refresh - Refreshes and redisplays the page being viewed.
- Home - Redisplays the Web site homepage.
- Search - Displays a Search field.
- Favorites - Displays a listing of favorite URLs that have been saved.
- History - Displays a history of viewed pages.

Mozilla Firefox, the second most popular browser, visually resembles Internet Explorer but provides one unique feature called *tabs*. With the tabs feature, multiple Web pages can be displayed on different tabs, resulting in fast and easy movement between each page.

Content Area

The content area of a window is discussed in Step 3, and will be further addressed in Step 13. By way of a brief summary, the content area will normally contain the following:

- A global navigation panel at the top.
- A local navigation panel on the left.
- A bottom navigation panel.
- Information, data fields, buttons, and so forth.

The content area may also contain limited windowing capabilities. The frame concept does provide window-like ability, and JavaScript does provide pop-up windows.

Frames

- Description:
  - — Multiple Web screen panes that permit the displaying of multiple documents on a page.
  - — These documents can be independently viewed, scrolled, and updated.
  - — The documents are presented in a tiled format.
- Proper usage:
  - — For content expected to change frequently.
  - — To allow users to change partial screen content.
  - — To permit users to compare multiple pieces of information.
- Guidelines:
  - — Use only a few frames (three or less) at a given time.
  - — Choose sizes based upon the type of information to be presented.
  - — Never force viewers to resize frames to see information.
  - — Never use more than one scrolling region on a page.

Description. Historically, the Web is essentially a single page (or, by analogy, a single window) entity. While providing significant interface benefits, it is also a reversal of the interface evolution process that led from single-screen technology to windowing. To counteract this shortcoming, *frames* were created. A frame is an independent pane of information presented in a Web page, or, again by analogy, as multiple windows. Frames, however, are presented as tiled, with no overlapping capability. The interaction richness, support, and contextual cues provided by overlapping windows are lacking. Frames, then, allow the displaying of multiple documents on a single Web page. These multiple documents can be independently viewed, scrolled, and updated.

Proper usage. Frames are useful in situations where portions of the page content are expected to change frequently. The volatile information can be separated from other page content and placed within a frame, thereby requiring only a portion of the page's content to be modified. Frames are also useful for allowing the user to change page content; navigation links can be placed in one frame and the resulting content displayed within another frame. As different links are selected, the content in the related frame changes. Frames more effectively allow users to compare multiple pieces of related information displayed within the different frames.

Advantages and disadvantages. Frames, like most interface entities, have advantages and disadvantages. At this moment in their existence, the disadvantages seem to outweigh the advantages. These disadvantages, however, are being whittled away as Web technology advances.

Frames, *advantages* mostly cluster around their ability to reduce the user's content comprehension mental load. These include the following:

- They decrease the user's need to jump back and forth between screens, thereby reducing navigation-related cognitive overhead.
- They increase the user's opportunity to request, view, and compare multiple sources of information.
- They allow content pages to be developed independently of navigation pages.

The *disadvantages* mostly cluster around navigational shortcomings, including

- The difference between a single Web page and a page with frames is not always obvious to the user.
- They suffer some of the shortcomings of tiled screens.
- Only a limited number can be displayed in the available screen area.
- They are perceived as crowded and more visually complex because frame borders are flush against one another and they fill up the whole screen. Crowding is accentuated if the borders contain scroll bars and/or control icons. Viewer attention may be drawn to the border, not the data.
- Frames-based pages behave differently from regular Web pages.
- Page-printing difficulties and problems can exist.
- Page interaction can be clumsy.
- URLs cannot be e-mailed to other users.
- Frames will not work on older browsers.

Past problems, now being addressed and mostly solved, have included difficulties in bookmarking pages, difficulties in creating browser history lists, and inconsistencies in behavior of the browser's Back button.

Guidelines. Guidelines to consider in using frames are the following: Use no more than three frames at a time. Using more will shrink each frame's usable area to the point where little space will be available for presenting content. Then, users will not be able to see much and be forced to scroll. Choose frame sizes based upon the type of information you want to present. Navigational links, for example, should be presented in a small frame and the corresponding information in a larger adjacent frame. Never force people to resize frames to see information. If people feel they must resize frames, the page design is poor. Do not use more than one scrolling region in frames contained on a page. This may be confusing to many users.

Technological advances in frames will continue to occur. Knowledge related to frame usability will also advance. Always be aware of the latest developments.

Pop-Up Windows

- Be extremely cautious in the use of pop-up windows.

JavaScript pop-up windows began appearing on the Web in 1996. Their use is multiplying and, in the view of almost all Web users, polluting screens. Because they are most frequently used in advertising, they have become a source of great aggravation to almost every user. Anecdotal evidence suggests that when a pop-up window begins to appear, most people close them before they are rendered. So, if a pop-up window is used, it may never be completely seen or read by the user. Use them with extreme caution.

Step 5 Exercise

An exercise for Step 5 can be found on this book's companion Web site, www.wiley.com/college/galitz.

STEP

6

Select the Proper Interaction Devices

Interaction devices are the input mechanisms or devices through which people communicate their needs and desires to the computer, and the output mechanisms or devices through which the computer responds to people. The fast-paced evolution of computer systems has seen greatly expanded families of devices to assist and enhance this communication. The distinction between input and output devices is not always clear-cut, however. The common keyboard, for example, is essentially a device for keying information, commands, and so forth. But the keyboard also has an output component. A properly designed keyboard provides feedback when a key is pressed; an audible click and a tactile resistance change when it activates or bottoms-out. For this discussion, devices will be categorized and reviewed based upon their primary purpose — input or output.

Input Devices

For years the device of choice in display-based systems was the standard keyboard and some human-engineered variations. As graphical systems evolved, emphasis shifted to another device, the mouse and some of its cousins: the trackball and joystick. These new mechanisms are most commonly referred to as pointing devices. A few other devices have also been around and have seen extended service through the years: the light pen and the graphic tablet. Some unique human devices also exist: touch and voice. A finger has been used in conjunction with touch-sensitive screens. Our vocal chords are being harnessed to speak meaningfully to the computer, not simply to shout words of exasperation. These various alternatives have both strengths and

weaknesses. Selecting the proper device-based control to do the required job is critical to system success. A good fit between user and control will lead to fast, accurate performance. A poor fit will result in lower productivity, produce more errors, and increase user fatigue. The characteristics and capabilities of various input devices will be reviewed including the following:

- Trackball
- Joystick
- Graphic tablet or trackpad
- Touch screen
- Light pen
- Voice
- Mouse
- Keyboard

We'll begin by reviewing the kinds of input tasks being performed. We'll discuss each device and identify its advantages and disadvantages. Then, we'll focus on the most popular control, the mouse, describing it in more detail and presenting a series of design guidelines for its use. The keyboard, because of its versatility and usefulness for text entry tasks, will also be examined in more detail. Finally, pertinent research will be reviewed and guidelines presented to aid in selecting the proper device.

Characteristics of Input Devices

Several specific tasks are performed using today's systems:

- To point at an object on the screen.
- To select the object or identify it as the focus of attention.
- To drag an object across the screen.
- To draw something free-form on the screen.
- To track or follow a moving object.
- To orient or position an object.
- To enter or manipulate data or information.

The various devices vary in how well they can perform these actions. Among the considerations to be reviewed are two very important factors. First, is the mechanism a direct or indirect pointing device? *Direct* devices are operated on the screen itself. Examples include the light pen, the finger, and voice. *Indirect* devices are operated in a location other than the screen, most often on the desktop. Examples include the mouse, trackball, and keyboard. The psychomotor skills involved in learning to use, and using, a direct device are much simpler than those required for an indirect device. Most of these direct device skills were instilled in our formative years.

Input devices are either discrete or continuous in their action. *Discrete* devices are used to enter individual bits of information — letters, numbers, or commands. *Continuous* input devices operate sequentially in nature — best exemplified by tasks such as dragging or drawing.

Another consideration is what is the relationship between movement of the hand-operated device and the corresponding pointer movement on the screen in terms of *direction*, *distance*, and *speed*? Does the pointer movement track control movement exactly or does it not? The mouse achieves a coupled relationship in all three aspects of direction, distance, and speed: The pointer on the screen moves in the direction the mouse is pushed, at the speed the mouse is pushed, and the distance the mouse is pushed (there may be some ratios applied). A trackball does not achieve this relationship in all three aspects. The pointer moves the direction the trackball is turned and the speed the ball is turned, but not the distance the ball is moved because the ball does not move forward or backward; its socket is stationary. Devices possessing coupled relationships in these three aspects require less psychomotor skill learning than those not possessing a coupled relationship in all three aspects.

Trackball

- Description:
  - — A spherical object (ball) that rotates freely in all directions in its socket.
  - — Direction and speed is tracked and translated into cursor movement.
- Advantages:
  - — Direct relationship between hand and pointer movement in terms of direction and speed.
  - — Does not obscure vision of screen.
  - — Does not require additional desk space (if mounted on keyboard).
- Disadvantages:
  - — Movement is indirect, in a plane different from the screen.
  - — No direct relationship exists between hand and pointer movement in terms of distance.
  - — Requires a degree of eye-hand coordination.
  - — Requires hand to be removed from keyboard keys.
  - — Requires different hand movements.
  - — Requires hand to be removed from keyboard (if not mounted on keyboard).
  - — Requires additional desk space (if not mounted on keyboard).
  - — May be difficult to control.
  - — May be fatiguing to use over extended time.

Description. Commonly used with notebook PCs, the trackball is a ball that rotates freely in all directions in its socket. Essentially it is an inverted mouse. The ball is rotated with one's fingertips, and its direction and speed are tracked and translated into equivalent screen cursor movement. Trackballs are well suited for navigational control, as in video games or exploration of 3-D environments. In these tasks, smooth movement is more important than fine target acquisition. A miniature trackball may also be mounted on a mouse, as is done by Apple.

Advantages. In terms of direction and speed, a trackball possesses a direct relationship between how it is rolled and how the cursor moves on the screen. The cursor moves in the same direction and speed ratio as the ball is rotated. Many trackballs

are mounted on the keyboard itself, permitting the user's hands to remain close to the keys. Trackballs on the keyboard do not require additional desk space, although the keyboard must often be expanded to allow for their inclusion. People with limited finger movement may find it easier to manipulate a trackball than a mouse. Trackballs are available in a variety of sizes and shapes to satisfy varying user needs.

Disadvantages. Trackballs share a common problem with several other controls: Control movement is in a different plane from the screen, or indirect. The cursor, or pointer, is separated from the control itself — the pointer being on the screen, the control on the keyboard. To effectively use a trackball requires learning the proper psychomotor skills, fine finger movements for accurate pointing, and gross hand movements for moving longer distances. The fine finger movements necessary to use them can be difficult to perform. Over longer periods of use, they can be fatiguing. When paired with keyboard tasks, they require a shift in motor activity from keystrokes to finger/hand movement.

Joystick

- Description:
  - — A stick or bat-shaped device anchored at the bottom.
  - — Variable in size, smaller ones being operated by fingers, larger ones requiring the whole hand.
  - — Variable in cursor direction movement method, force joysticks respond to pressure; movable ones respond to movement.
  - — Variable in degree of movement allowed, from horizontal-vertical only to continuous.
- Advantages:
  - — Direct relationship between hand and pointer movement in terms of direction.
  - — Does not obscure vision of screen.
  - — Does not require additional desk space (if mounted on keyboard).
- Disadvantages:
  - — Movement indirect, in plane different from screen.
  - — Indirect relationship between hand and pointer in terms of speed and distance.
  - — Requires a degree of eye-hand coordination.
  - — Requires hand to be removed from keyboard keys.
  - — Requires different hand movements to use.
  - — Requires hand to be removed from keyboard (if not mounted on keyboard).
  - — Requires additional desk space (if not mounted on keyboard).
  - — May be fatiguing to use over extended time.
  - — May be slow and inaccurate.

Description. A joystick, like its aircraft namesake, is a stick or bat-shaped device usually anchored at the bottom. They come in variable sizes, smaller ones being operated by fingers, larger ones requiring the whole hand. The smaller joysticks require fine motor coordination, the larger ones more gross coordination. Some,

called *force* joysticks, are immovable, responding to pressure exerted against them. The direction and amount of pressure is translated into pointer movement direction and speed. Others, called *movable* joysticks, can be moved within a dish-shaped area. The direction and distance of the movements create a similar pointer movement on the screen. Some kinds of joysticks permit continuous movements, others only horizontal and vertical movements. Some joysticks may be mounted on the keyboard. They are also well suited for navigational control where smooth movement is most important. Small joysticks, sometimes called "eraser heads" may be found on Notebook computers.

Advantages. Joysticks typically possess a direct relationship between hand and cursor movement in terms of direction. They do not obscure vision of the screen and, when mounted on the keyboard, do not require additional desk space.

Disadvantages. Joysticks are also indirect devices, the control and its result being located in different planes. They require developing a skill to use and can be slow and inaccurate. Use over extended time may also be fatiguing. When paired with keyboard tasks, they require a shift in motor activity from keystrokes to finger/hand movement.

Graphic Tablet or Trackpad

- Description:
  - — Pressure-, heat-, light-, or light-blockage-sensitive horizontal surfaces that lie on the desktop or keyboard.
  - — May be operated with fingers, light pen, or objects like a stylus or pencil.
  - — Pointer imitates movements on tablet.
- Advantages:
  - — Direct relationship between touch movements and pointer movements in terms of direction, distance, and speed.
  - — More comfortable horizontal operating plane.
  - — Does not obscure vision of screen.
- Disadvantages:
  - — Movement is indirect, in a plane different from screen.
  - — Requires hand to be removed from keyboard.
  - — Requires hand to be removed from keyboard keys.
  - — Requires different hand movements to use.
  - — Requires additional desk space.
  - — Finger may be too large for accuracy with small objects

Description. A graphic tablet, also called a "trackpad," "touch tablet," "touchpad," or simply "tablet," is a device with a horizontal surface sensitive to pressure, heat, light, or the blockage of light. It may lie on the desk or may be incorporated on a keyboard, and it is operated with fingers, light pen, or objects like a pencil or stylus. The screen pointer imitates movement on the tablet.

Advantages. With graphic tablets, a direct relationship exists between touch movements and pointer movements in terms of direction, distance, and speed. The screen mimics the tablet. When used with objects like styluses, light pens, or pencils, the operational angle, horizontal, is more comfortable than those vertically oriented.

Disadvantages. Tablets are also indirect controls, creating coordination problems. To use them requires moving one's hand from the keyboard and, if using another device, picking it up. If the finger is the tablet-activation object, accuracy with small objects is difficult. Tablets may also require desk space.

Trackpads operated with one's finger are increasingly found on Notebook PCs. Learning to use a touchpad is more difficult than learning to use a mouse and people with limited finger dexterity frequently find them difficult to operate.

Touch Screen

- Description:
  — A special surface on the screen sensitive to finger or stylus touch.
- Advantages:
  — Direct relationship between hand and pointer location in terms of direction, distance, and speed.
  — Movement is direct, in the same plane as screen.
  — Requires no additional desk space.
  — Stands up well in high-use environments.
- Disadvantages:
  — Finger may obscure part of screen.
  — Finger may be too large for accuracy with small objects.
  — Requires moving the hand far from the keyboard to use.
  — Very fatiguing to use for extended period of time.
  — May soil or damage the screen.
- Design Guidelines:
  — Screen objects should be at least ¾ × ¾ inches in size.
  — Object separation should be at least ⅛ inch.
  — Provide visual feedback in response to activation. Auditory feedback may also be appropriate.
  — When the consequences are destructive, require confirmation after selection to eliminate inadvertent selection.
  — Provide an instructional invitation to begin using.

Description. A touch screen is a screen that consists of a special surface sensitive to finger or stylus touch. Objects on the screen are pointed to and touched to select them.

Advantages. Touch screens possess a direct relationship between hand and pointer movement in terms of direction, distance, and speed. This relationship is direct, not indirect, because the control (finger or stylus) is on the same plane as the

pointer. Another significant advantage of a touch screen is that it does not require any additional desk space.

Disadvantages. A disadvantage of touch screens is that they are fatiguing to use over an extended period of time. If a finger is the touch mechanism, it may obscure part of the screen and be too large to be accurate with small objects. A stylus is usually more accurate than the finger. Fingers may also soil the screen, and a stylus may damage it. Both finger and stylus require moving a hand from the keyboard, and if a stylus is used, it must also be picked up.

Guidelines. When using touch screens, larger screen objects should always be provided to foster accuracy in use. Objects should be ¾ inches square at a minimum and separated by at least ⅛ inch. Visual, and perhaps auditory, feedback should be provided in response to activation. When the consequences of selection are destructive, require a confirmation to avoid inadvertent selection. Observational research indicates that touch screen devices placed in public places, for use by the general public, should possess an instructional invitation to begin their use.

Today other forms of touch screen devices are being used. One type allows placement of a finger on the screen without item selection, selection being accomplished by lifting the finger off the screen. This may allow more accurate item selection. Another method involves placing a cursor on the screen directly above one's finger and moving the cursor as the finger is moved. The cursor permits better target visibility, as well as the detection of smaller targets.

Light Pen

- Description:
  — A special surface on a screen sensitive to the touch of a special stylus or pen.
- Advantages:
  — Direct relationship between hand and pointer movement in terms of direction, distance, and speed.
  — Movement is direct, in the same plane as screen.
  — Requires minimal additional desk space.
  — Stands up well in high-use environments.
  — More accurate than finger touching.
- Disadvantages:
  — Hand may obscure part of screen.
  — Requires picking it up to use.
  — Requires moving the hand far from the keyboard to use.
  — Very fatiguing to use for extended period of time.

Description. A light pen, or pen, also utilizes a touch screen, but one that is sensitive in a specific way to one kind of pen or stylus. Advantages and disadvantages are similar to those of the touch screen.

Advantages. Light pens possess a direct relationship between hand and pointer movement in terms of direction, distance and speed, and are also classified as

direct pointing devices because the control (pen or stylus) is on the same plane as the pointer. Another advantage of a light pen is that it does not require any additional desk space, except for a place for the pen to rest. A light pen or stylus is usually more accurate than the finger. Stylus pens are useful for the very small screens found on personal digital assistants (PDAs) because of their small pointer size.

Disadvantages. A disadvantage is that they are also fatiguing to use over an extended period of time. Light pens require moving a hand from the keyboard to pick up and use.

Voice

- Description:
  — Automatic speech recognition by the computer.
- Advantages:
  — Simple and direct.
  — Useful for people who cannot use a keyboard.
  — Useful when the user's hands are occupied.
- Disadvantages:
  — High error rates because of difficulties in
    - Recognizing boundaries between spoken words.
    - Blurred word boundaries because of normal speech patterns.
  — Slower throughput than with typing.
  — Difficult to use in noisy environments.
  — Impractical to use in quiet environments.

Description. Automatic speech recognition technology has been under development for more than a quarter of a century. Its progress has been hindered by the following disadvantages listed. Recently, however, it has evolved to the point where telephone-answering systems can now handle simple voice input.

Advantages. Speech is a simple and direct communication medium. It is very useful for people who cannot use a keyboard, or whose hands are otherwise occupied.

Disadvantages. Speech recognition errors are fundamentally different from keying errors. Most keying errors result from a user's inability to always press the correct key. Most speech recognition errors result from the computer speech recognizers' inability to correctly recognize words. People can dictate to a computer at a fairly fast rate, about 105 words per minute (Karat et al., 1999; Lewis, 1999). After making the required corrections, the input rate becomes about 25 words per minute when transcribing the input. New users had even lower transcribing rates. As summarized in Step 1, typists, even those of the two-finger variety, have much higher keying rates. Error correction also takes much longer with a speech recognition system. The most commonly used correction methods are deleting and repeating the last phrase; deleting and repeating a specific word; deleting and selecting a correct word from a list of alternative words; and retyping the selection.

Several research studies have shown that correcting voice recognition errors using a method other than additional voice recognition speeds up the correction process. Suhm, Myers, and Waibel (1999) found that fast typists made almost three times more corrections per minute than people who made corrections by voice only. Lewis (1999) and Karat et al. (1999) uncovered very similar results.

Speech recognition is also, of course, difficult to utilize in an improper environment. Noise can hinder the process, and it is very impractical, and disturbing, to try and use it in a very quiet location such as a library.

Mouse

- Description:
  - — A rectangular or dome-shaped, movable, desktop control containing from one to three buttons used to manipulate objects and information on the screen.
  - — Movement of screen pointer mimics the mouse movement.
- Advantages:
  - — Direct relationship between hand and pointer movement in terms of direction, distance, and speed.
  - — Permits a comfortable hand resting position.
  - — Selection mechanisms are included on mouse.
  - — Does not obscure vision of the screen.
- Disadvantages:
  - — Movement is indirect, in a plane different from screen.
  - — Requires hand to be removed from keyboard.
  - — Requires additional desk space.
  - — May require long movement distances.
  - — Requires a degree of eye-hand coordination.

Description. A mouse is a rectangular or dome-shaped, movable, desktop control typically possessing one, two, or three buttons, but may include more. It is used to manipulate objects and information on the screen. The movement of the screen pointer mimics the mouse movement. It is also common for a mouse to possess a scroll wheel and, in the case of Apple, a miniature trackball that acts as an omnidirectional scroll device. In the mid 1960s, Doug Engelbart, a researcher at the Stanford Research Institute, invented what became the mouse. While using a trackball, he was inspired to turn it upside down and let the ball become the bottom of a control that, attached to a cord, was moved across the desk. It was patented as the "x-y position indicator," and finally christened the "mouse" by a colleague of Engelbart's (a colleague whose name is lost in time). In 1997, Engelbart was at long last rewarded for his invention when he received the annual Lemelson-Mit Prize for American Innovation, including a well-deserved and very substantial monetary reward (cnn.com, 1997).

Advantages. There is a direct relationship between hand and pointer movement in terms of direction, distance, and speed. The mouse itself contains some basic controls (buttons) useful for manipulating screen objects. The hand position when

using the mouse is generally fairly comfortable, and the mouse does not obscure the screen.

Disadvantages. Disadvantages are that they are also indirect devices, the control and its result being located in different planes. They require developing a skill to use and, when paired with keyboard tasks, they require movement away from the keyboard and a shift in motor activity from keystrokes to finger/hand movement. The mouse also requires extensive additional desk space and long positioning movements. The mouse comes in a variety of configurations, performs some basic functions, and is operated in several ways.

Configurations. A mouse may possess one, two, or three buttons. Most, but not all, windowing systems permit operation using all configurations. Buttons are used to perform three functions to be described. When three mouse buttons are not available, the pointer location or keyboard qualifiers must be used to determine the function to be performed. A multibutton mouse permits a more efficient operation, but a person must remember which button to use to perform each function. A multibutton mouse may usually be configured for left- or right-hand use.

Functions. The functions performed by a mouse are Select, Menu, and Adjust. The *Select* function is used to manipulate controls, to select alternatives and data, and to select objects that will be affected by actions that follow. Select is a mouse's most important function and is the function assigned to a one-button mouse. For a multibutton mouse, it is usually assigned to the leftmost button (assuming a right-handed operation).

The *Menu* function is typically used to request and display a pop-up menu on a screen. A menu appears when the button is depressed within a particular defined area of the screen. This area may be, for example, the entire screen, within a window, or on a window border. This button eliminates the need for a control icon, which must be pointed at and selected. The user, however, must remember that a menu is available. The *Adjust* function extends or reduces the number of items selected. It is the least used of the three functions and is usually assigned last and given the least prominent location on a mouse.

Operations. Several operations can be performed with a mouse. The first, point, is the movement and positioning of the mouse pointer over the desired screen object. It prepares for a selection or control operation. To press is to hold the button down without releasing it. It identifies the object to be selected.

To *click* is to press and immediately release a button without moving the mouse. This operation typically selects an item or insertion point, operates a control, or activates an inactive window or control. To *double-click* is to perform two clicks within a predefined time limit without moving the mouse. It is used as a shortcut for common operations such as activating an icon or opening a file.

To *drag* is to press and hold the button down, and then move the pointer in the appropriate direction. It identifies a range of objects or moves or resizes items. To *double-drag* is to perform two clicks and hold the button down, and then move the pointer in the appropriate direction. It identifies a selection by a larger unit, such as a group of words.

Mouse Usage Guidelines

- Provide a "hot zone" around small or thin objects that might require extremely fine mouse positioning.
- Never use double-clicks or double-drags as the only means of carrying out essential operations.
- Do not use mouse plus keystroke combinations.
- Do not require a person to point at a moving target.

If an object is very small and might require fine mouse positioning, provide a large "hot zone" around it. This will increase target size and speed selection. Do not require double-clicks or double-drags as the only way to carry out essential operations. Rapid double-pressing is difficult for some people. Do not use mouse plus keystroke combinations to accomplish actions. This can be awkward. One exception: multiple selections of items in a list. Do not require a person to point at a moving target, except, of course, for a game.

Keyboard

- Description:
  — Standard typewriter keyboard and cursor movement keys.
- Advantages:
  — Familiar.
  — Accurate.
  — Does not take up additional desk space.
  — Very useful for
    - Entering text and alphanumeric data.
    - Editing text and alphanumeric data.
    - Keyed shortcuts — accelerators.
    - Keyboard mnemonics — equivalents.
  — Advantageous for
    - Performing actions when less than three mouse buttons exist.
    - Use with very large screens.
    - Touch typists.
- Disadvantages:
  — Slow for non-touch-typists.
  — Can be over-elaborate.
  — Slower than other devices in pointing.
  — Requires discrete actions to operate.
  — No direct relationship between finger or hand movement on the keys and cursor movement on screen in terms of speed and distance.

Description. A keyboard is a discrete input device. Christopher Latham Sholes invented the standard typewriter keyboard in 1870. Commonly called the QWERTY layout, Sholes' placement of letters was intended to slow down a

typist's keying movements so that the potential for key jams was minimized. From a strictly human-engineering perspective, its layout inadequacies included a dominance of the left hand in making keystrokes, frequent successive keystrokes with the same hand, frequent movement between keyboard key rows, and frequently used letter pairs being placed far from each other. In 1936, August Dvorak created a revised and well-human-engineered keyboard that overcame many of these deficiencies. The advantages of the Dvorak layout, as it came to be called, included a right-hand dominance in keying, much less frequent row changes, and more systematic alternation between the right and left hand. With this new layout, finger travel distances were reduced by at least one order of magnitude. Acceptance of this new keyboard was, and continues, to be slow. Most people have seemed unwilling to invest the time and effort to change.

In the 1980s IBM performed a series of studies comparing the QWERTY keyboard with various sequential key layouts such as ABCDEF or JIHGFE (starting from the upper-left or QWERTY location). IBM wanted to determine if a sequential layout was better for users who professed to be non-touch-typists. Their findings were surprising. Non-touch-typist performance results were as good, or better, using the QWERTY layout as using the various systematic layouts. IBM's conclusion: Why change? So they didn't. IBM researchers could only speculate about why the new systematic layouts fared so poorly. Perhaps, they said, while non-touch-typists profess no knowledge of the QWERTY layout, through experience they have learned the layout, at least well enough to permit effective two-finger typing to be accomplished. Another possibility, they said, was that perhaps some characteristic of the QWERTY layout makes it easy to scan and find needed keys. Speculation number one seems to be the most reasonable explanation, but it may never be known for sure.

Advantages. The standard keyboard is familiar, accurate, and does not consume additional desk space. It is useful and efficient for entering or inserting text or alphanumeric data. For tasks requiring heavy text or data entry, shifting the hands between a keyboard and an alternative control, such as a mouse, can be time-consuming and inefficient, especially for a touch typist. The keyboard is flexible enough to accept keyed shortcuts, either keyboard accelerators or mnemonic equivalents. Some systems also permit navigation across a screen through use of keyboard keys such as the space bar, arrows, Tab, and Enter. Inefficiencies in using other graphical device-based controls can occur, making it advantageous to use a keyboard. A mouse with a limited number of buttons will require use of the keyboard to accomplish some functions, possibly causing frequent shifting between devices. When operations are being performed on very large screens, the user may also find keyboard window management preferable to the long mouse movements frequently required. Therefore, to compensate for these possible inefficiencies, many windowing systems provide alternative keyboard operations for mouse tasks.

Disadvantages. Disadvantages of a keyboard include its requiring discrete finger actions to operate instead of the more fine positioning movements. As a result, no direct relationship exists in terms of speed and distance between finger or hand movement on the keys and cursor movement on the screen. Depending on the

layout of the keyboard cursor control keys, direct-relationship direction problems may also exist, because fingers may not move in the same direction as the cursor. Keyboards will also be slower for non-touch-typists and slower than other controls in pointing tasks. Unfortunately, many keyboards are over-elaborated in design, possessing many keys that are rarely, if ever, used by most people.

Keyboard Guidelines

- Provide keyboard accelerators.
  - — Assign single keys for frequently performed, small-scale tasks.
  - — Use standard platform accelerators.
  - — Assign Shift+key combinations for actions that extend or are complementary to the actions of the key or key combination used without the Shift+key.
  - — Assign Ctrl+key combinations for
    - Infrequent actions.
    - Tasks that represent larger-scale versions of the task assigned to the unmodified key.
- Provide keyboard equivalents.
  - — Use standard platform equivalents.
  - — Use the first letter of the item description.
  - — If first letter conflicts exist, use
    - Another distinctive consonant in the item description.
    - A vowel in the item description.
- Provide window navigation through use of keyboard keys.

Keyboard accelerators. Accelerators provide a way to access menu elements without displaying a menu. They are useful for frequent tasks performed by experienced users. Keys assigned for accelerators should foster efficient performance and be meaningful and conceptually consistent to aid learning. Use standard accelerators as shown in Table 4.2 in Step 4. Microsoft suggests that frequently performed, small-scale tasks should be assigned single keys as the keyboard alternative. Actions that extend or are complementary to the actions of a key (or key combination) should be assigned a Shift key in conjunction with the original action. Microsoft, for example, uses a single key, F6, as the key to move clockwise to the next pane of an active window. To move counterclockwise to the next pane, use Shift+F6. Infrequent actions, or tasks that represent larger-scale versions of the task assigned to the unmodified key, should be assigned Ctrl+key combinations. The left arrow key in Microsoft Windows, for example, moves the cursor one character; Ctrl+left arrow moves it one word.

Keyboard equivalents. Keyboard mnemonics enable the selection of a menu choice through the keyboard instead of by pointing. This enables a person's hands to remain on the keyboard during extensive keying tasks. Keyboard mnemonics should be chosen in a meaningful way to aid memorability and foster predictability of those things that may be forgotten. Mnemonics need only be unique within a menu. A simple rule is always to use the first letter of a menu item description. If the first letter of one item conflicts with that of another, choose

another distinctive consonant in the item description, preferably, but not always necessarily, the second in the item word (occasionally another consonant may be more meaningful). The last choice would be a vowel in the item description. If standard platform equivalents are available, use them. Standard equivalents are shown in Table 4.1 in Step 4.

Window navigation. Also provide ways of navigating through windows by the use of keyboard keys.

MAXIM **The greater the effort to accomplish a task, the less likely that the task will be accomplished successfully.**

Other Input Devices

As technology advances, other types of input devices are in various forms of development. *Gesture* recognition involves understanding the gestures people use in day-to-day communication. Through hand movements and facial expressions people can communicate moods, feelings, and actions. (A smile is recognized as OK, a frown as cancel. A downward pointed finger means stop. A turn of the head left means back.) Imagine the possibilities.

Eye tracking can be used to control an interaction. The cursor is moved to where the user is looking. An action is performed through looking at the proper button. *Iris* and *fingerprint* recognition can be used to verify that you are who you say you are. *Handwriting* is accurately recognized and can be used to enter data or information. Some day these devices may be as common as the mouse or keyboard.

Selecting the Proper Input Device

Many studies have been performed comparing the various input devices for assorted office tasks. Significant findings include the following:

Keyboard versus Mouse

Why do many skilled typists prefer a keyboard to a mouse? Speed is obviously one reason. An experienced typist, through kinesthetic memory, has memorized the location of keyboard keys. The keying process becomes exceptionally fast and well learned. The mouse is slower, and it has a tendency to move about the desk. Its location cannot be memorized. The keyboard keys always remain in the same spot.

Consider the following: When using the mouse, the time to move one's hand from the keyboard, grasp the mouse, and point at a screen object ranges from 1.5 to 2 seconds. A very skilled typist can type 13 to 15 characters in that amount of time; an average typist can type 4 to 6 characters. No wonder the keyboard is often preferred.

Control Research

Which devices work better for which tasks and under what conditions has been addressed by many investigators. A survey of the research literature comparing and

evaluating different devices yields the following summarization concerning tasks involving pointing and dragging:

- The fastest tools for pointing at stationary targets on screens are the devices that permit direct pointing: the touch screen and light pen. This is most likely because of their high level of eye-hand coordination and because they use an action familiar to people.
- In terms of positioning speed and accuracy for stationary targets, the indirect pointing devices — the mouse, trackball, and graphic tablet — do not differ greatly from one another. The joystick is the slowest, although it is as accurate as the others. Of most importance in selecting one of these devices will be its fit to the user's task and working environment.
- A separate confirmation action that must follow pointer positioning increases pointing accuracy but reduces speed. The mouse offers a very effective design configuration for tasks requiring this confirmation.
- For tracking small, slowly moving targets, the mouse, trackball, and graphic tablet are preferred to the touch screen and light pen because the latter may obscure the user's view of the target.

Another common manipulation task is dragging an object across the screen. Pointing and dragging using a mouse, graphic tablet, and trackball for this task was studied by MacKenzie et al., (1991). They report the following:

- The graphic tablet yielded best performance during pointing.
- The mouse yielded best performance during dragging.
- The trackball was a poor performer for both pointing and dragging, and it had a very high error rate in dragging.

Guidelines for Selecting the Proper Input Device

- Consider the characteristics of the task.
  - — Provide keyboards for tasks involving
    - Heavy text entry and manipulation.
    - Movement through structured arrays consisting of a few discrete objects.
  - — Provide an alternative pointing device for graphical or drawing tasks. The following are some suggested best uses:
    - Mouse — pointing, selecting, drawing, and dragging.
    - Joystick — selecting and tracking.
    - Trackball — pointing, selecting, and tracking.
    - Touch screen — pointing and selecting.
    - Graphic tablet — pointing, selecting, drawing, and dragging.
  - — Provide touch screens under the following conditions:
    - The opportunity for training is minimal.
    - Targets are large, discrete, and spread out.
    - Frequency of use is low.
    - Desk space is at a premium.
    - Little or no text input requirement exists.

- Consider user characteristics and preferences.
  — Provide keyboards for touch typists.
- Consider the characteristics of the environment.
- Consider the characteristics of the hardware.
- Consider the characteristics of the device in relation to the application.
- Provide flexibility.
- Minimize eye and hand movements between devices.

Selection of the proper device for an application or system, then, depends on a host of factors.

Task characteristics. Is the device suited for the task? Standard typewriter keyboards are always necessary for tasks requiring text entry and manipulation; keyboards (cursor control keys) are usually faster when moving through structured arrays consisting of a few discrete objects. For graphical and drawing tasks, alternative pointing devices are easier and faster. Use a mouse, joystick, trackball, or graphic tablet for pointing, selecting, drawing, dragging, or tracking. The devices best suited for each kind of task are summarized in the preceding list.

Provide touch screens when the opportunity for training is minimal; targets are large, discrete, and spread out; frequency of use is low; desk space is at a premium; and little or no text input requirement exists. The pointing device should also reflect the degree of accuracy needed to perform the task. Touch screens are generally inaccurate, unless the target is large enough, comprising space availability.

Consider how much time the user will spend on the system. Using a touch screen for a long period of time can be fatiguing. Devices where the forearm can be rested on a table are best for long periods of use.

User characteristics and preferences. Will the user be able to easily and comfortably operate the control? Are the fine motor movements required by some devices capable of being performed? Some children and people with disabilities such as hand tremor may find it difficult to use devices requiring a high degree of accuracy. A study found elderly adults enjoy significant improvements in their interaction accuracy with a touch screen relative to a mouse, whereas younger adults do not do any better with a touch screen (Iwase and Murata, 2002).

Is the user familiar with the standard keyboard? What are the user's preferences? While preferences do not always correspond to performance, it is important that the user be comfortable with the selected device. Also consider device-learning requirements. Direct pointing devices (light pen, stylus, and touch screen) are intuitive and easy to learn. They are best in situations where people cannot be expected to spend time learning, and for infrequent system users.

Environmental characteristics. Will the device fit easily into the work environment? If desk space is necessary, does it exist and is it large enough? Some devices require very little room (trackball, joystick, and graphic tablet). A mouse requires significantly more space. Public access systems will require a sturdy device such as the touch screen. A pen or stylus can easily be broken or stolen. Touch screens work well when the usage environment is dirty.

Hardware characteristics. Is the device itself of a quality that permits easy performance of all the necessary tasks? Joysticks, for example, are quite variable in their movement capabilities.

The device in relation to the application. Is the device satisfactory for the application?

Flexibility. Often task and user needs will vary within an application. Providing more than one kind of device will give the user choices in how to most efficiently accomplish whatever tasks must be performed. A keyboard paired with another kind of pointing device is almost always necessary.

Minimizing eye and hand movements. When multiple devices are used, eye and hand movements between them must be minimized. Structure the task, if possible, to permit the user to stay in one working area. If shifts must be made, they should be as infrequent as possible. It is estimated that, for a good typist, it costs 3 to 8 keystrokes for each jump between the keyboard and a mouse. The general rule is that more than 80 percent of the tasks should be doable using only one device.

Pointer Guidelines

- The pointer
  - — Should be visible at all times.
  - — Should contrast well with its background.
  - — Should maintain its size across all screen locations and during movement.
  - — The hotspot should be easy to locate and see.
  - — Location should not warp (change position).
- The user should always position the pointer.
- The shape of a pointer
  - — Should clearly indicate its purpose and meaning.
  - — Should be constructed of already defined shapes.
  - — Should not be used for any purpose other than its already defined meaning.
  - — Do not create new shapes for already defined standard functions.
- Use only as many shapes as necessary to inform the user about current location and status. Too many shapes can confuse a person.
- Be conservative in making changes as the pointer moves across the screen.
  - — Provide a short "time-out" before making noncritical changes on the screen.
- Animation should not
  - — Distract.
  - — Restrict one's ability to interact.

The focus of the user's attention in most device operations is usually the pointer. Therefore, the pointer image should be used to provide feedback concerning the function being performed, the mode of operation, and the state of the system. For example, the pointer shape image can be changed when it is positioned over a selectable object, signaling to the user that a button action may be performed. When an action is being

performed, the pointer can assume the shape of a progress indicator such as an hourglass, providing an indication of processing status.

A pointer should contrast well with its background and be visible at all times. The user should always be in control of its location on the screen. The shape of a pointer should clearly indicate its purpose and meaning. Always use predefined shapes provided by graphical systems. Microsoft Windows, for example, provides about two dozen standard shapes. To aid learning and avoid user confusion, never create new shapes for already defined standard functions or use a shape for any purpose other than its previously defined meaning. Also, use only as many shapes as absolutely necessary to keep the user informed about current position and status. Too many shapes can also confuse a person.

Be conservative in making changes as the pointer moves across the screen. Excessive changes can be distracting to a person. To avoid frequent changes while crossing the screen, establish a short time-out before making noncritical pointer changes. Any pointer animation should not distract the viewer or restrict one's ability to interact with the system.

Output Devices

The computer communicates to the user through output devices. The most common is the display screen.

Screens

Screens are very useful for presenting a wide range of visual elements and complex data. They have been the workhorse of computer systems for decades. For many years the visual display terminal, or VDT (as it was called before the term monitor became popular when the keyboard and monitor were detached), has been constructed using a raster-scan cathode ray tube (CRT). In its earlier years CRT's were very prone to flickering, but technological improvements have resulted in flicker-free CRTs with excellent resolution. Their one drawback is their size.

Recent years have seen tremendous improvement in an alternative display, the liquid crystal display (LCD). LCDs are much smaller, thinner, and lighter than CRTs while possessing the same viewing area. They consume less power than a CRT and are now much more affordable. Stone et al. (2005) suggests the following should be considered in choosing a screen:

Image. How detailed does the image need to be? For highly graphic applications involving images and photographs, a high-resolution screen will be desirable. For text work and larger letter sizes, a lower resolution may be satisfactory.

Colors. How many colors are needed? The number of colors can range from monochrome to millions. The application will determine this requirement.

Size. Like televisions, screen sizes are measured across the diagonal. Common desktop sizes are 17, 19, and 21 inches. The larger the screen, the more advantages exist. More information can be displayed, as can larger text and images. Size will

depend upon the needs of the application and the needs of the user. A visually impaired person, for example, may well want larger text and images. Much smaller screens are required for hand-held devices.

Portability. Does the screen need to be portable? CRTs are heavy and not portable. LCDs are lightweight and very portable.

Usage space. CRTs have a much larger desktop footprint. LCDs take up much less space. Crowded desks and environments will find benefits in an LCD.

Speakers

Computer sounds have advanced from a simple beep to the reproduction of speech, music, and sound effects. The quality of the speaker should reflect the quality of the sound being presented.

Step 6 Exercise

An exercise for Step 6 can be found on this book's companion Web site, www.wiley.com/college/galitz.

FUTURE VISION BIE

By K B Hemanth Raj

Visit : https://hemanthrajhemu.github.io

Quick Links for Faster Access.

CSE 8th Semester - https://hemanthrajhemu.github.io/CSE8/

ISE 8th Semester - https://hemanthrajhemu.github.io/ISE8/

ECE 8th Semester - https://hemanthrajhemu.github.io/ECE8/

8th Semester CSE - TEXTBOOK - NOTES - QP - SCANNER & MORE

17CS81 IOT - https://hemanthrajhemu.github.io/CSE8/17SCHEME/17CS81/

17CS82 BDA - https://hemanthrajhemu.github.io/CSE8/17SCHEME/17CS82/

17CS832 UID - https://hemanthrajhemu.github.io/CSE8/17SCHEME/17CS832/

17CS834 SMS - https://hemanthrajhemu.github.io/CSE8/17SCHEME/17CS834/

8th Semester Computer Science & Engineering (CSE)

8th Semester CSE Text Books: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Text-Book.html

8th Semester CSE Notes: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Notes.html

8th Semester CSE Question Paper: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Question-Paper.html

8th Semester CSE Scanner: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Scanner.html

8th Semester CSE Question Bank: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Question-Bank.html

8th Semester CSE Answer Script: https://hemanthrajhemu.github.io/CSE8/17SCHEME/Answer-Script.html

Contribution Link:

https://hemanthrajhemu.github.io/Contribution/

Stay Connected... get Updated... ask your queries...

Join Telegram to get Instant Updates:
https://telegram.me/joinchat/AAAAAFTtp8kuvCHALxuMaQ

Contact: MAIL: futurevisionbie@gmail.com

INSTAGRAM: www.instagram.com/futurevisionbie/