**Software Interfaces**

While the OS gives you tools to work with the computer (logging on and off, file management, configuring components, etc), application software is designed to perform specific tasks not usually related to the computer.  Application software needs the operating system in order to run.  All operations and tasks that the application allows you to perform need to talk to the OS before it gets to the computer hardware.  In this instance the OS is functioning as an interface between the application and the computer hardware.

Computer applications are interfaces between the actions you are performing and the responses that the computer provides to those actions.  Well designed and well behaved programs take their interactive and visual cues from the operating system.  Icons, menus, and labels are used identically to the way the OS uses them.  The uniqueness in an application comes from the nature of the tasks it is meant to perform, and the features/limitations that the application programmers implemented.  The logic of the program, how it is organized, how well it emulates or supports the way you want to perform the task, and how difficult it is to learn all affect the usability of the interface.

For example, if the task you want to perform is "write an essay", how does the application support that process?  Does it auto-correct your spelling as you type?  Can you speak and the program types your words? Does it assess your writing style and make suggestions?  Does it give you an easy way to do a table of contents?  Are the options obvious?  Is it easy to change fonts?  Does it use styles or do you have to manually change formatting for paragraphs?  These are a few of the questions you need to ask when assessing a writing program.

**Form Follows Function**

This tenet of the Bauhaus is equally true for software design.  The form, or visual, structural, and interactive design of software is based on the purpose of the software, on the tasks and actions users will perform using the software.

The following programs illustrate software interface design for programs with different purposes.  These programs emulate practices that were established before the programs were invented when people did this sort of work manually.  Designing a program this way is called using a metaphor.  The program is designed to visually and mechanically emulate the normal practice in use prior to the program.  Of course, the programs provide many new options not available when the work was done manually.  In some cases they allow you to do entirely new things.  To make the images fit on these pages, each program was restricted to about 500 pixels wide.  When you are using these programs they are usually much wider.

* **Word processing programs**.  Word processors are writing tools.  They evolved from the typewriter.  Everything is tied to the page.  All content has to be on the page, and it has to be connected and sequenced.  The image below of Word illustrates this.

**
Figure Word using a Page Metaphor**

Nothing can exist outside the edges of the page.  Word processors have sophisticated features and are sometimes used to publish bigger documents, but they are primarily writing tools.

* **Desktop publishing programs**.  Desktop publishing (DTP) programs are used to lay out and publish documents.  They are not good as writing tools or drawing tools, although they have features to do both.  They are exceptionally good for assembling existing materials into sophisticated documents like magazines and books.   DTP programs evolved from the publishing industry where pages and whole documents were 'laid-up' on a desktop by gluing cut out pieces of text and images onto a page.  It should not be surprising to learn, then, that DTP programs use a desktop metaphor.   The working space is a 'desktop', on which you can place pages and 'boxes' containing text and graphics.  These boxes, or objects, can be moved onto and off the page, and placed in any order you want.  PageMaker, shown below, is typical of DTP programs.

**
Figure PageMaker using a Desktop Metaphor**

Features to note in PageMaker are the tools on the top left, the palettes on the top right, and the page selector on the bottom left.  Pages 2-3 are selected.  PageMaker is showing a 2 page spread.  As in all publications, the even-numbered page is on the left, and the odd-numbered page is on the right.  You should also note that the basic tools include a text tool, and drawing tools.

* **Drawing programs**. Drawing tools are also called vector drawing tools, because drawings are made up of lines, circles, boxes, and other drawing 'primitives'.  These objects are all stored as mathematical equations, resulting in drawings that have very small file sizes.  Vector drawings may be scaled up or down without loss of image quality.  These tools evolved from the technical drawing and technical illustration industries.
	+ Technical drawing tools, or CAD programs (Computer Assisted Drawing) specialize on features that make it easier to do technical drawings for houses, vehicles and so on.
	+ Technical illustration tools are programs that specialize on features to make illustrations easier to do.  Illustration tools also use a desktop metaphor.  Corel Draw is an example of this sort of drawing program.

**
Figure CorelDraw using Desktop Metaphor.**

The page is the rectangle in the center with the dropped shadow.  Notice that some drawings are on the page and some are off.  The page is configured in pixels for creating web drawings.  You may recognize the drawings from an earlier lesson.  Note the drawing tools on the left, and the colour palettes on the right. The left palette contains web-safe colours, and the right one has standard printing colours (yes, there is a difference!—remember RGB vs CMYK colour palettes).

* **Image editing programs**. Image editing programs work with bitmap images.  These programs emulate photographic processes, but add many tool and techniques not possible with traditional photo processing. These images are made up of dots.  Each dot is assigned a colour value.  These pictures do not scale up or down easily.  Making them bigger makes bigger dots.  Making them smaller eliminates dots.  File sizes can be very large.  These programs are based around the image, which can be any size you want.  Work can only occur within the boundaries of the image.  Photoshop CS, shown below, is typical of image editing tools.

**
Figure Photoshop as Image-based Metaphor**

Like many programs, Photoshop has 'tear away' menus and tools.  Note the toolbar on the left.  A few of the many palettes are shown.  They can be customized by dragging the tabs from one palette to another to put the ones you use most into a single grouping.

**Common Interface Features**

The four examples above also share a number of common interface features.  Menus, for example, are in the same place and some menu items are common to all. Only the first two items seem to be kept in the same place in all programs.  Many programs use View and Window menus but put them in different places.  The rest of the menu items reflect the purpose of the program.

**
Figure Menus**

**Common Icons**

Other standards are also at work.  Icons used to represent different actions and tools have a common meaning regardless of the program.  This is part of the standardization of the graphical user interface.  These standards are imposed more rigorously on the Mac platform than on the Windows platform, and much more so than on Linux.  A few common examples are shown below.

*     Save the current page
*     Save everything that is open
*     Print
*     Cut the current selection
*      Copy the current selection
*      Magnify tool
*      Text Tool
*      Rectangle drawing tool

**Common Methods of Work**

Although each of the programs illustrated are for fundamentally different purposes, the interface design transfers methods of work across all of them.  For example, all programs let you create and edit 'content', whether it is text, images, or both.  Regardless of the type of content, each program treats the content as collections of objects.  Each object has attributes assigned to it that you can change.  Text, for example has a font face (i.e., Ariel), a type size (i.e., 12 point), and specified spacing between characters, words and lines of text.  Drawings have line thickness, line length, line colour and fill colours.  The respective program lets you change the attributes.  You can make text larger, for example, or change the colour of a box.  While you can make these changes one at a time, most programs use a concept called styles.  As you will learn, Visual Basic implements Properties, which are versions of styles.

A style lets you assign all the possible attributes to an object at the same time.  If you make a text style called Heading, for example, you can assign it the font Helvetica, font size 12 points, left justified, and bolded.  Then every time you want a heading with these attributes, you just tell it to be Heading.  One click and you are done.  Imaging the time saving for large documents with a few hundred pages.  Similarly, styles can be used to apply attributes to graphics, or copy attributes from one graphic to another.

The examples below show how styles are used in the 4 sample programs.

* **Styles used in Word**.  Word has dozens of pre-designed styles.  A few are shown below.  You can modify the existing ones, or create you own.  You can also save a set of styles as a stylesheet and use them with new documents.  Select the text, and then select a style from the drop down list.  Want to reformat the document?  Just go back and change the attributes assigned to the style and the whole document reformats.  Very powerful.

**
Figure Styles applied in Word**

* **Styles used in PageMaker**.  PageMaker relies exclusively on styles to format all the text in the document.  It also uses master pages with layout of columns and other features pre-defined by you.  When you add text, it flows through the layout you've created.  Clicking on the text and selecting a style formats the entire paragraph with the attributes of that style.  Documents are designed using the visual design elements and principles by designing the master pages and style sheets.  Just add content and format it with the stylesheet and you have a masterpiece.

**
Figure Styles in PageMaker**

* **Styles used in Draw**.  Draw makes extensive use of object attributes to keep track of how an object is formatted, and makes use of styles to make it easier to use the same attributes on different objects.  For example, Corel Draw lets you copy the attributes of graphics from one image to another

**
Figure Copying Graphic Styles in Corel Draw (from the top circle to the bottom one)**

Note that you can copy the outline of a shape as well as the fill.  The object being copied has the outline removed, has a radial fill, and has a dropped shadow.  To copy all of them to the new object, you check all three boxes.  To create all-new graphic or text styles, use the graphics and text docker, shown next.

**
Figure Text Styles in Corel Draw (The graphic and text styles Docker Window is shown)**

The default styles may be changed and saved so that when you start a new document, the new styles apply.  An unlimited number of new styles can be created.  Styles make it possible to use Draw as a desktop publishing program.

* **Styles used in Photoshop**.   Photoshop allows you to work in layers.  Layers are used to apply all sorts of effects to the final image.  Layers can be assigned styles using a variety of attributes.

**
Figure Photoshop Pre-designed Layer Styles**