Fourth Edition

DESIGNING USER EXPERIENCE

A guide to HCI, UX and interaction design

Pearson

DAVID BENYON

Chapter 12
Visual interface design

Contents

Aims

The design of the interface that mediates the interaction of people with devices is a crucial characteristic of the overall UX design. This is often referred to as the user interface (UI) and it consists of everything in the system that people come into contact with, whether that is physically, perceptually or conceptually. In this chapter we discuss the issues of interface design, focusing on the visual aspects of the design. UX designers need to know what their options are for delivering a compelling UI that draws people in and engages them in the interaction. UX designers also need to know what works in interface design and what principles and guidelines there are for delivering usable and engaging UX. In the next chapter we focus on issues of design when multiple modalities are involved in an interface.

After studying this chapter you should be able to:

- Understand different types of interaction, command languages and graphical user interfaces (GUIs)
- Understand and apply interface design guidelines
- Understand the issues of information presentation
- Understand the key issues in designing for visualization.

12.1 Introduction

The design of the interface that mediates the interaction of people with devices is a crucial characteristic of the overall interaction design. This is often referred to as the user interface and it consists of everything in the system that people come into contact with, whether that is physically, perceptually or conceptually.

← We first encountered the interface in Chapter 2

Physically, people interact with systems in many different ways, such as by pressing buttons, touching a screen, moving a mouse over a table so that it moves a cursor over the screen, clicking a mouse button, rolling their thumb over a scroll wheel. We also interact physically through other senses, notably sound and touch, but we defer a discussion of these modalities until the next chapter.

Perceptually, people interact with a system through what they can see, hear and touch. The visual aspects of interface design concern designing so that people will see and notice things on a screen. Buttons need to be big enough to see and they need to be labelled in a way that people can understand. Instructions need to be given so people know what they are expected to do. Displays of large amounts of information need to be carefully considered so that people can see the relationships between data to understand its significance.

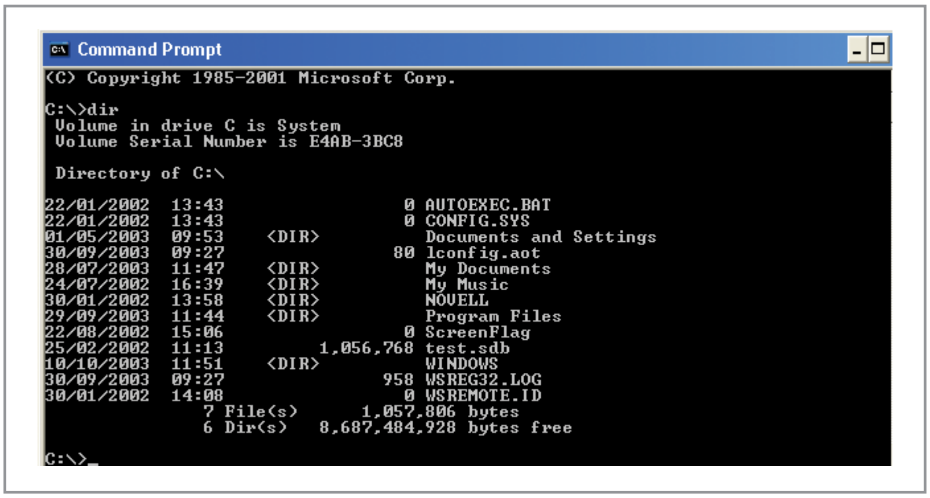
Conceptually, people interact with systems and devices through knowing what they can do and knowing how they can do it. People employ a 'mental model' of what the service or device is and how it works. They need to know that certain commands exist that will allow them to do things. They need to know that certain content is available and the form it takes. They need to find their way to particular pieces of content (undertake navigation). They need to be able to find details of things, see an overview of things and focus on particular areas.

Putting these three aspects together is the skill of the interface designer. Interface design is about creating an experience that enables people to make the best use of the system being designed. When first using a system people may well think, 'right, I need to do X so I am going to use this device which means I am going to have to press Y on this keyboard and then press Z', but very soon people will form their intentions in the context of knowing what the system or device does and how they can achieve their goals. Physical, perceptual and conceptual design get woven together into the experiences of people.

The vast majority of personal computers, phones and handheld and tablet devices have graphical user interfaces typically based on one of the main three software platforms: Apple (with its operating systems macOS and iOS), Microsoft Windows and Google's Android. However, underlying these GUIs are user interfaces without the graphical elements, known as command languages. A command language is simply a set of words with an associated syntax, the rules governing the structure of how commands are put together. To interact with a device using a command language, the user types a command such as 'send', 'print', etc. and supplies any necessary data such as the name of a file to be sent or printed. UNIX is the most common command language. Command languages suffer from the problem that people:

- Have to *recall* the name of a particular command from the range of literally hundreds of possibilities
- Have to *recall* the syntax of the command.

Prior to the creation of Microsoft Windows, the vast majority of personal computers ran the operating system MSDOS. On switching on their PC, people were faced with the user interface known as the c:\> prompt (Figure 12.1). People were then required to type in a


Command Prompt

```
(C) Copyright 1985-2001 Microsoft Corp.

C:\>dir
 Volume in drive C is System
 Volume Serial Number is E4AB-3BC8

 Directory of C:\

22/01/2002  13:43                    0 AUTOEXEC.BAT
22/01/2002  13:43                    0 CONFIG.SYS
01/05/2003  09:53    <DIR>          Documents and Settings
30/09/2003  09:27                   80 lconfig.aot
28/07/2003  11:47    <DIR>          My Documents
24/07/2002  16:39    <DIR>          My Music
30/01/2002  13:58    <DIR>          NOVELL
29/09/2003  11:44    <DIR>          Program Files
22/08/2002  15:06                    0 ScreenFlag
25/02/2002  11:13           1,056,768 test.sdb
10/10/2003  11:51    <DIR>          WINDOWS
30/09/2003  09:27                  958 WSREG32.LOG
30/01/2002  14:08                    0 WSREMOTE.ID
               7 File(s)      1,057,806 bytes
               6 Dir(s)   8,687,484,928 bytes free

C:\>_
```

Figure 12.1 The enigmatic c:\> prompt in MSDOS

command such as dir, which listed the contents of the current directory (or folder). Anyone who had never encountered MSDOS (or even those who had) was continually faced with the problem of having to recall the name of the command to issue next.

However, command languages are not all bad. They are quick to execute and, particularly if there are only a few of them, people using them frequently will remember them. Commands can be spoken, which makes for a very convenient interface, particularly if you are concentrating on something else. Spoken commands are convenient for in-car systems, for example. The search engine Google has a number of commands such as 'define:' to indicate particular types of search. There are gestural commands such as a three-fingered swipe on an Apple track pad to move to the next item. Spoken commands to search for items or to set reminders are available from the agent called Cortana in the Windows 10 operating system, from Google Now, Apple's Siri or Alexa from Amazon.

Challenge 12.1

In his piece in Interactions, *Don Norman (2007) argues that commands have a number of benefits. However, a key issue is that the system must be in the correct mode to recognize and react to the commands. For example, in the science fiction series 'Star Trek' people have to alert the computer when they wish to enter a command, e.g. the captain might say 'Computer. Locate Commander Geordie Laforge'. If they did not do this the computer would not be able to distinguish commands intended for it from other pieces of conversation. However, in the 'Turbo Lift' (the elevator), this is not necessary. Why is this?*

12.2 Graphical user interfaces

Graphical user interfaces, which are found on every personal computer, on smartphones, on touchscreen displays and so on, have had an interesting though brief history, starting in the 1980s. The Microsoft range of Windows GUIs were broadly based on the Macintosh, which in turn was inspired by work at Xerox PARC, which in turn was developed from and built upon early research at the Stanford Research Laboratory and at the Massachusetts Institute of Technology. During the 1980s and 1990s a number of GUI designs were produced, but gradually Windows and Apple Macintosh came to dominate the GUI operating system market. Google Chrome OS is starting to challenge them, but the basic functions and icons of a GUI are currently fairly well defined.

The fact that objects are represented as graphics means that people can *recognize* what they want to do rather than having to *recall* some command from memory. They can also reverse their actions, which means recovering from mistakes is much easier.

The most prevalent of the GUIs is the WIMP interface such as Windows or macOS. WIMP stands for windows, icons, menus and pointers. A **window** is a means of sharing a device's graphical display resources among multiple applications at the same time. An **icon** is an image or symbol used to represent an object or app. A **menu** is a list of commands or options from which one can choose. The last component is a **pointing device**, of which the mouse is most common for desktop interaction, but people use their fingers on smartphones and tablets. They can also use a pen or pencil or a stylus to point and select items.

BOX 12.1

Direct manipulation

A direct manipulation (DM) interface is one where graphical objects on the screen are directly manipulated with a pointing device. This approach to interaction was first demonstrated by Ivan Sutherland in the Sketchpad system. The concept of direct manipulation interfaces for everyone was envisioned by Alan Kay of Xerox PARC in a 1977 article about the Dynabook (Kay and Goldberg, 1977). The first commercial systems to make extensive use of direct manipulation were the Xerox Star (1981), the Apple Lisa (1982) and the Macintosh (1984). However, it was Ben Shneiderman at the University of Maryland who actually coined the term 'direct manipulation' in 1982.

He defined a DM interface as one where there is:

1 Continuous representation of the object of interest.
2 Physical actions or labelled button presses instead of complex syntax.
3 Rapid incremental reversible operations whose impact on the object of interest is immediately visible (Shneiderman, 1982, p. 251).

Direct manipulation depends upon having bitmapped screens, so that each picture element or pixel can be used for input and output, and a pointing device. Early mobile phones did not have such a display, so direct manipulation of objects was not possible. Currently many of them do, and DM is found on a wide range of devices.

Windows

Windows allow a device to be divided into areas which act like separate input and output channels that can be placed under the control of different applications. This allows people to see the output of several processes at the same time and to choose which one will receive input by selecting its window, using a pointing device, such as clicking on it with a mouse, or touching a touchscreen. This is referred to as changing the focus. Microsoft Windows 10 uses a tiled design where the windows do not overlap, Apple macOS uses overlapping windows.

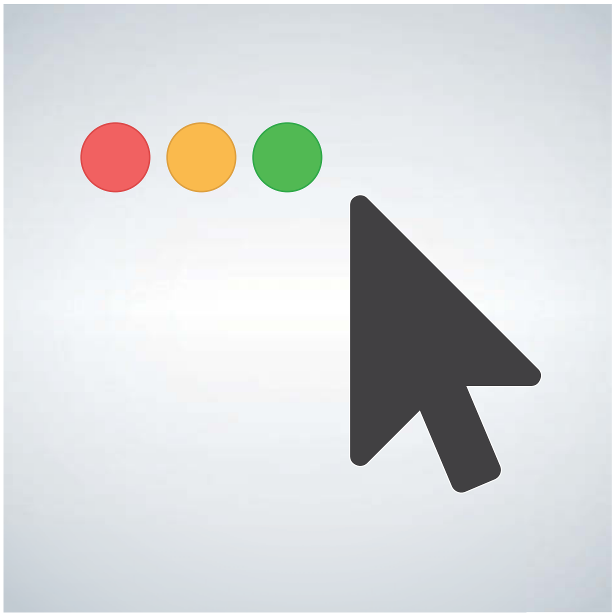
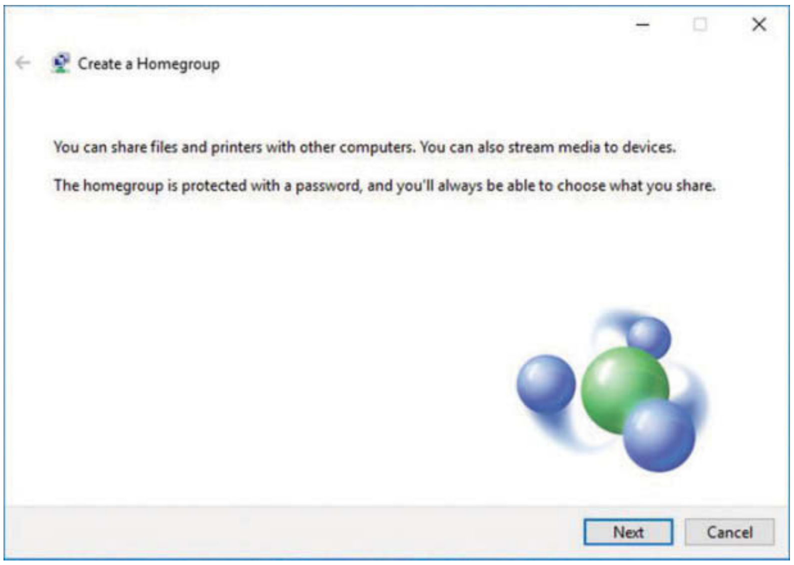


Figure 12.2 MacOS window

Windowing systems exist in a wide variety of forms but are largely variations on the same basic theme. Figures 12.2 and 12.3 show examples of an macOS window and a Microsoft Windows 10 window.

Icons

Icons are used to represent features and functions on everything from software applications, DVD players and public information kiosks to clothing and street signs. Icons are generally regarded as being useful in helping people to recognize some feature



Create a Homegroup

You can share files and printers with other computers. You can also stream media to devices.

The homegroup is protected with a password, and you'll always be able to choose what you share.

Figure 12.3 Windows 10 window

or function they need to access or know about. Icons first appeared on the Xerox Star (Box 12.2) and became an important research issue in the 1980s and early 1990s, though since then there has been considerably less interest.

The use of icons is now ubiquitous, but their design, apart from a small number of standard items, is rather arbitrary. Icons make use of three principal types of representation: metaphor, direct mapping and convention. **Metaphor** relies on people transferring knowledge from one domain and applying it to another. The use of metaphor can be seen in icons for such things as the cut and paste operations that exist in many applications.

← Metaphor is discussed in detail in Chapter 9

Direct mapping involves creating a more or less direct image of what the icon is intended to represent. Thus a printer icon looks like a printer. Finally, **convention** refers to a more or less arbitrary design of an icon in the first instance, which has become accepted as standing for what is intended over time. This can lead to anachronisms. For example, the icon representing the function *save* on the Mac that I am using to write this is a representation of a floppy disk (Figure 12.4) despite the fact that the machine is not fitted with a floppy disk drive and many people will never have heard of a floppy disk. Figure 12.5 shows examples of Window icons.

However, the two most important design issues for icons are legibility (whether or not one can discriminate between icons) and interpretation (what it is that the icon is intended to convey). The legibility aspect refers to icons not always being viewed under ideal conditions (e.g. poor lighting, screen resolution or the size of the icon itself). Research has indicated that under such conditions it is the overall global appearance of the icon that aids discrimination, so icons should not be designed so that they differ only with respect to one small detail.

The interpretation of the icon is a non-trivial issue. The icon may indeed be recognized as an object but remains opaque as to its meaning. Brems and Whitten (1987) for this reason caution against the use of icons which are not accompanied by a textual label. Do remember, however, that one reason why icons are used is that they are succinct and small (i.e. do not take up too much screen space); adding labels removes this advantage.

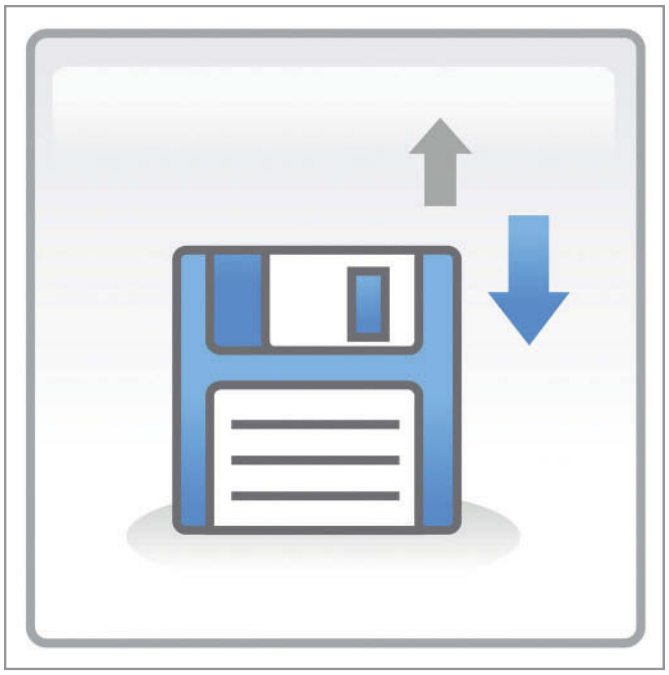


Figure 12.4 An icon representing a floppy disk
(Source: Ivary/Getty Images)

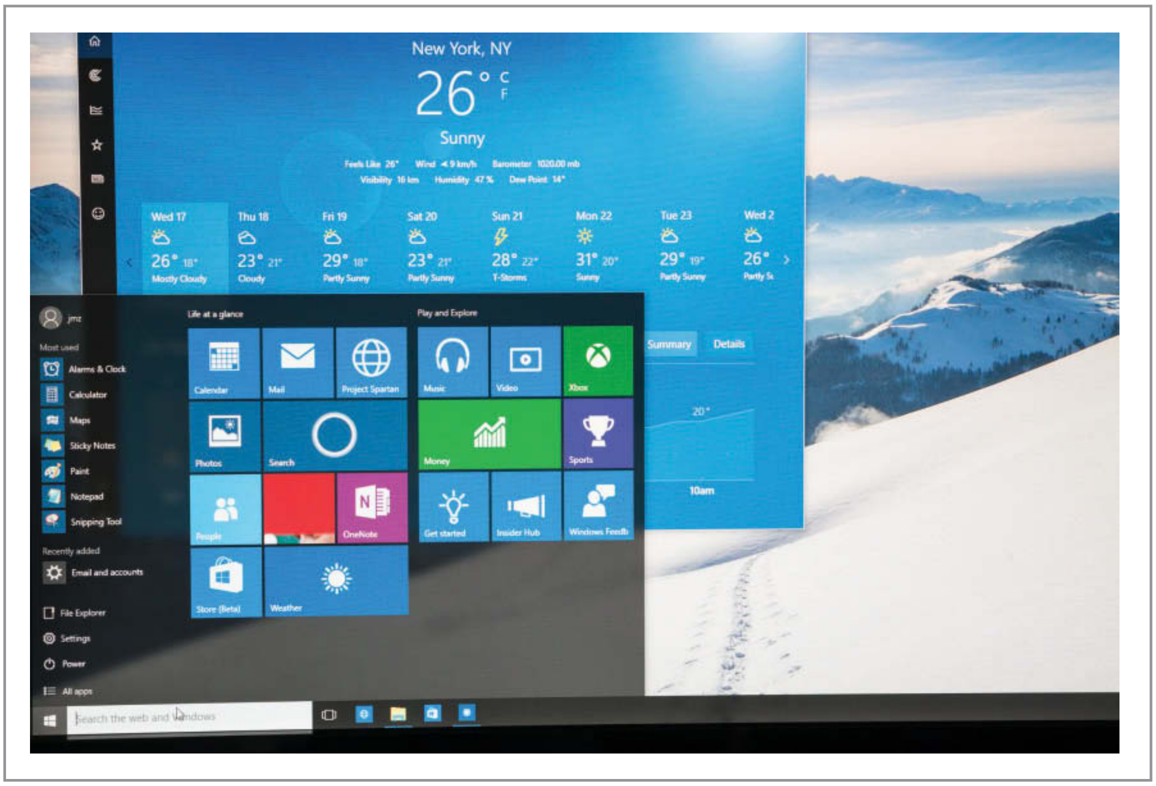


Figure 12.5 Examples of commonly used icons

BOX 12.2

The Xerox Star

It is widely recognized that every graphical user interface owes a debt to the Xerox Star workstation. Launched as the 8010 Star information system in April 1981, it was designed to be used by office workers and other professionals to create and manage business documents such as memos, reports and presentations. The Star's designers took the perspective that people were primarily interested in their jobs and not in computers *per se*. Thus, from its inception a central design goal was to make use of representations of objects that would be easily recognizable from an office environment (Figure 12.6).

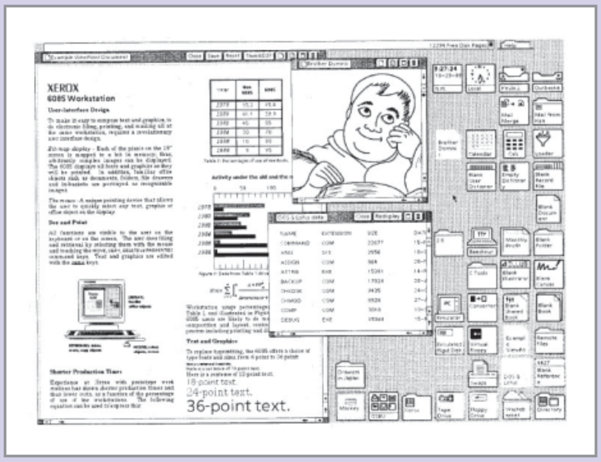


Figure 12.6 The Xerox Star user interface

(Source: Courtesy of Xerox Ltd)

Solutions to this problem include **balloon help** and **tool tips** which have appeared as effective pop-up labels if a mouse is moved over them. However, this is not a solution on a phone where the user has to touch an item to see what it does.

BOX 12.3

Horton's icon checklist

William Horton (of William Horton Consulting, Inc.) has produced a detailed checklist (1991) designed to help the icon designer avoid a whole raft of common mistakes. We reproduce his top-level headings here, together with a sample question for each issue.

Understandable	*Does the image spontaneously suggest the intended concept to the viewer?*
Familiar	*Are the objects in the icon ones that are familiar to the user?*
Unambiguous	*Are additional cues (label, other icon documentation) available to resolve any ambiguity?*
Memorable	*Where possible, does the icon feature concrete objects in action? Are actions shown as operations on concrete objects?*
Informative	*Why is the concept important?*
Few	*Is the number of arbitrary symbols less than 20?*
Distinct	*Is every icon distinct from all others?*
Attractive	*Does the image use smooth edges and lines?*
Legible	*Have you tested all combinations of colour and size in which the icon will be displayed?*
Compact	*Is every object, every line, every pixel in the icon necessary?*
Coherent	*Is it clear where one icon ends and another begins?*
Extensible	*Can I draw the image smaller? Will people still recognize it?*

?

Challenge 12.2

Using Horton's checklist for icon design and your own ideas, critique the design of the icons in Figure 12.5. Are they informative and easy to understand? What do you need to know in order to understand them?

Menus

Many applications of interactive systems make use of menus to organize and store the commands that are available. Items are chosen from the menu by selecting them with the pointer. Menus are also familiar on mobile phones, touchscreen kiosks and, of course, restaurants where the available options are listed on a menu.

When creating menus, commands should be grouped into menu topics, which are a list of menu items. When a command or option (menu item) is selected from the list, an action is performed. Menus are also used extensively on websites to structure information and to provide the main method of navigation of the site's content. While menus should be simple, there is little to prevent the over-zealous designer from creating complex and difficult-to-navigate menus. Windows 10 has a typical **hierarchically** organized menu.

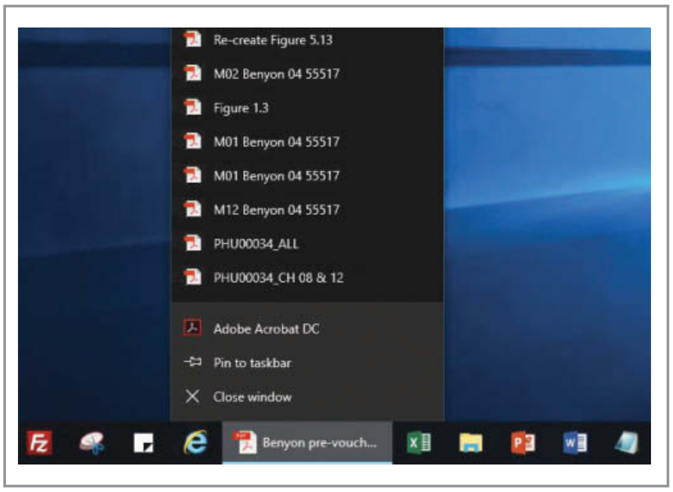


Figure 12.7 Example menu from Windows 10

The various options are arranged under a top-level topic (filter) and in turn have series of sub-menus. Hierarchical menus are also called **cascading menus**. In a cascading menu, the sub-menu appears to cascade out when a choice is made from the higher-level menu. Figure 12.7 is an example menu from Windows 10.

Another frequently encountered form of menu is the **pop-up**. A pop-up menu is distinguished from a standard menu in that it is not attached to a menu bar in a fixed location (hence the name). Once a selection is made from a pop-up menu, the menu usually disappears. Figure 12.8 is a screenshot of a pop-up menu. In this case it includes a number of

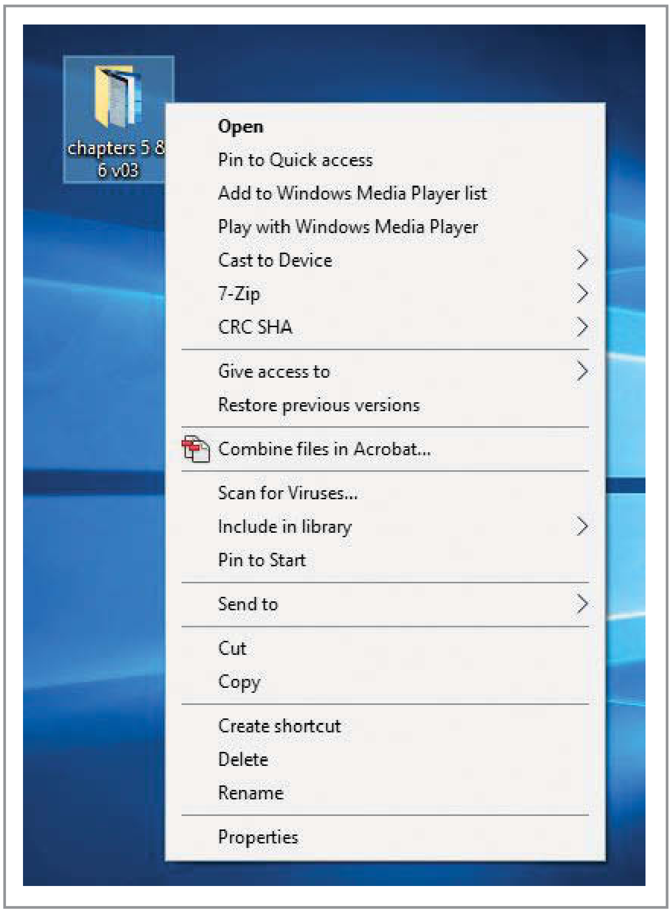


Figure 12.8 A screenshot of a pop-up menu (or panel) providing information on the file 'chapters 5 & 6 v03' after right clicking on the file and using the context menu

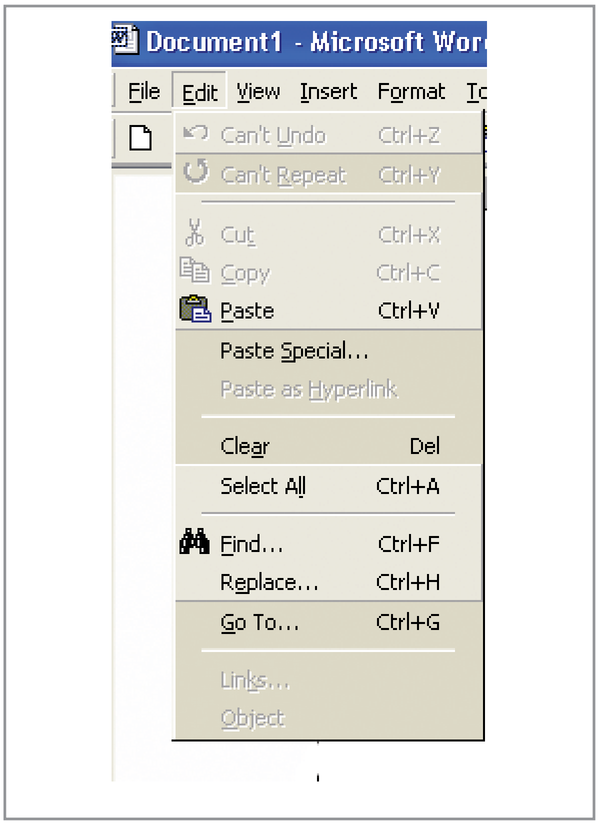


Figure 12.9 Shortcuts (Windows 10)

options that are not simple commands, so it is more usually referred to as a panel. In this case it is also a **contextual menu**. The make-up of contextual menus varies according to the context (hence their name) from which they are invoked. If a file is selected, the contextual menu offers file options. If a folder is selected instead, folder options are displayed.

Finally, to aid experts, it is common practice to associate the most frequently used items with keyboard shortcuts (also known as accelerators in MS Windows systems). Figure 12.9 illustrates shortcuts for the Windows 10 operating system.

Pointers

The final part of the WIMP interface is the pointer. The most common pointer is the mouse, but joysticks are also popular, for example in game controllers, and track pads control the pointer on laptop computers. On mobile phones and tablets, a stylus is often provided as the pointer and on touchscreen systems people use a finger. Remote pointers include the Wii wand and other infra-red pointers, for example for doing presentations. Gestures can be used with Microsoft Kinect.

→ Gestures are discussed in Chapter 13

12.3 Interface design guidelines

Modern GUIs as part of their make-up have a range of widgets, including buttons and radio buttons, sliders, scroll bars and checkboxes. These will often combine several aspects of the basic WIMP objects. Designing a GUI for an application does not guarantee that the finished system will be usable. Indeed, given the ease with which GUIs can be created with modern development tools, it is now very simple to create inelegant, unusable interfaces. This problem is well recognized and has resulted in the creation of **style guides** that provide a range of advice to the interface developer. Style guides exist for the three main operating systems, or platforms: Microsoft Windows 10, Mac macOS and iOS, and Android.

The Microsoft website offers abundant helpful advice on designing interfaces. Here is a sample:

> Grouping of elements and controls is also important. Try to group information logically according to function or relationship. Because their functions are related, buttons for navigating a database should be grouped together visually rather than scattered throughout a form. The same applies to information: fields for name and address are generally grouped together, as they are closely related. In many cases, you can use frame controls to help reinforce the relationships between controls.

Other advice on interface design operates at a much smaller level of detail, at the level of individual widgets. Interface consistency is an important result of using style guides, as is evident on devices such as the iPhone. The Apple guidelines for the iOS platform provide good advice and guidance on designing for standard items such as a toolbar, navigation bar, etc.

Android provides detailed advice about how large to make certain widgets and Apple says that any button should be no smaller than 44 pixels square. Android widgets are shown in Figure 12.10.

There are development environments available for creating apps on all three of the main platforms. These allow developers to use standard icons, menus and other features and to emulate how designs will look on a number of different devices such as a 5 inch smartphone or a 10 inch tablet.

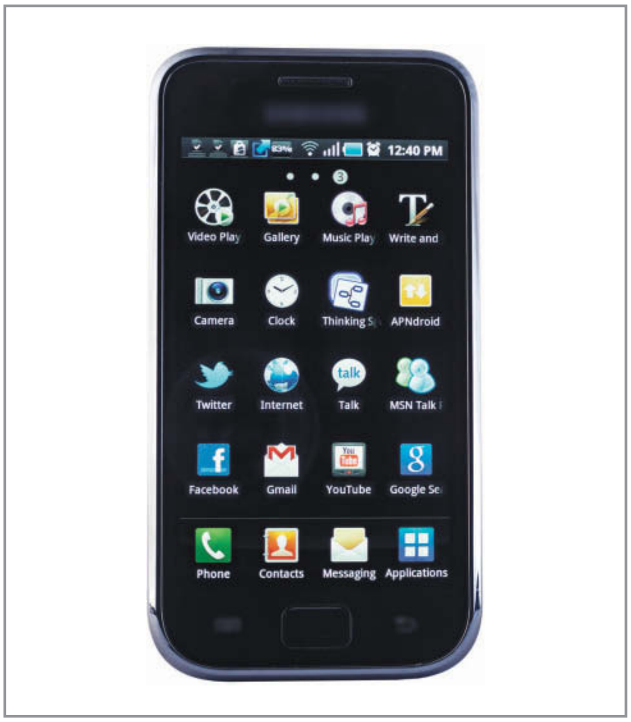


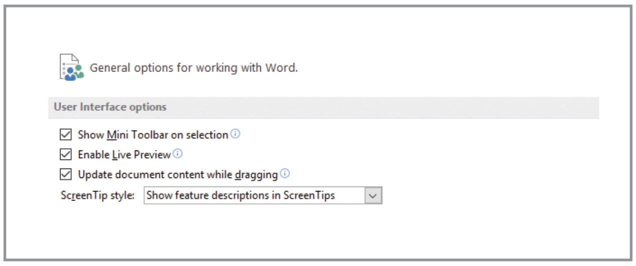
Figure 12.10 Android widgets

Radio buttons

A series of radio buttons can allow people to make *exclusive* choices – similar to the buttons on a radio: you can listen to FM or AM at any one time but not both.

Checkboxes

Checkboxes should be used to display individual settings that can be switched (checked) on and off. Use a group of checkboxes for settings that are not mutually exclusive (that is, you can check more than one box). Apple iOS uses switches in a similar way. An example is shown in Figure 12.11.



General options for working with Word.

User Interface options

☑ Show Mini Toolbar on selection

☑ Enable Live Preview

☑ Update document content while dragging

ScreenTip style: Show feature descriptions in ScreenTips

Figure 12.11 An example of a checkbox

Challenge 12.3

You are designing an email client which, among other things, allows people to:

- Set a series of preferences for incoming mail (download large files on receipt, display first two lines of message body, reject mail from senders not in address book, alert when new mail received. . .)
- Set a colour scheme for the email application (hot colours, water colours or jewel colours).

Would you use radio buttons or checkboxes for these?

Toolbars

A toolbar is a collection of buttons grouped according to function (in this respect they are conceptually identical to menus). The buttons are represented as icons to give a clue as to their function. Passing the mouse pointer over an icon will usually trigger the associated 'tool tip', which is a short textual label describing the function of the button. Toolbars are also configurable: their contents can be changed and one can choose whether or not they are displayed. Hiding toolbars helps make the best use of the display resources (usually described as the screen real estate). Figure 12.12 illustrates this.

List boxes

List boxes take a variety of forms and within these forms they offer different ways of viewing the contents – as lists (with more or less detail), as icons or as thumbnails (little pictures of the files' contents). A list box for the iPad is shown in Figure 12.13.

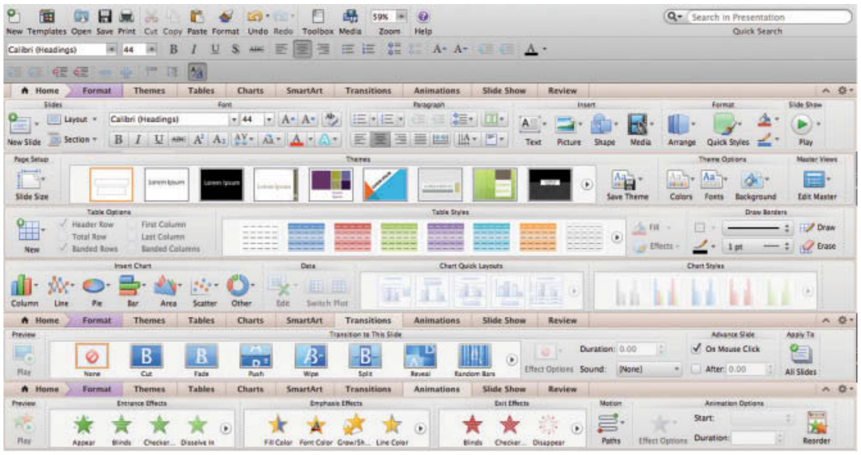


Figure 12.12 Why it is useful to be able to hide the full range of available toolbars (taken from MS PowerPoint)

Sliders

A slider is a widget that can return analogue values – rather than setting, say, the volume to 7 on a scale of 10, people can drag a slider to a position three-quarters of the way along a scale. Sliders (Figure 12.14) are ideally suited to controlling or setting such things as volume or brightness or scrolling through a document.

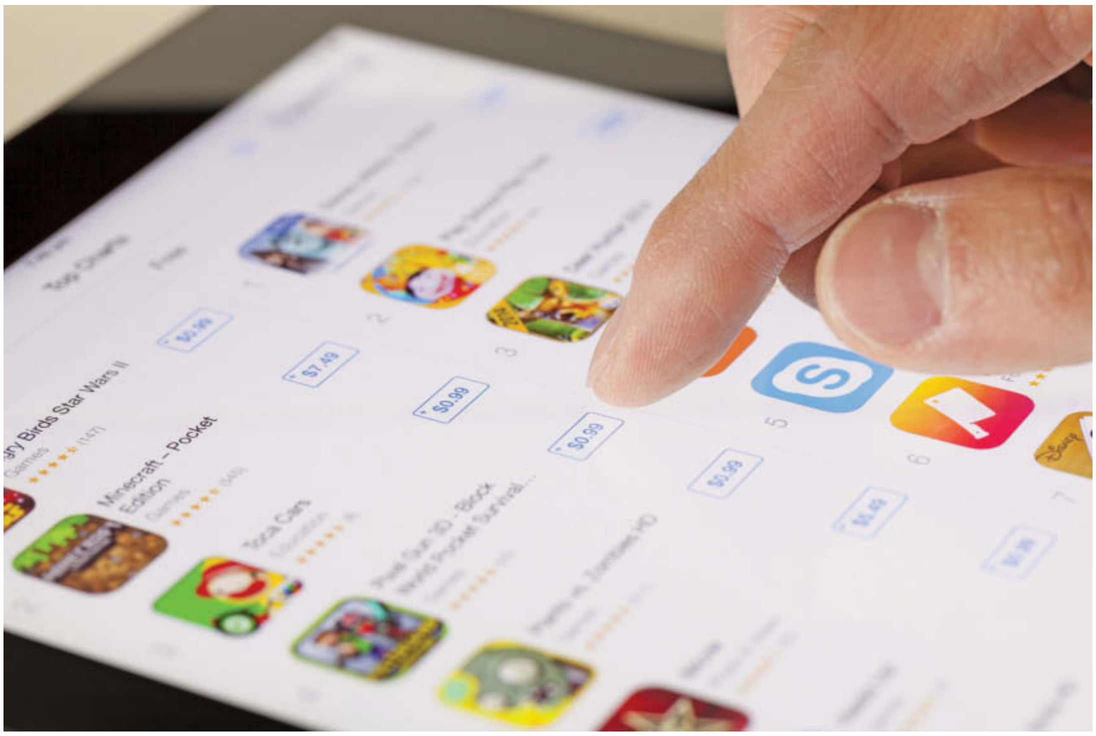


Figure 12.13 iPad list box

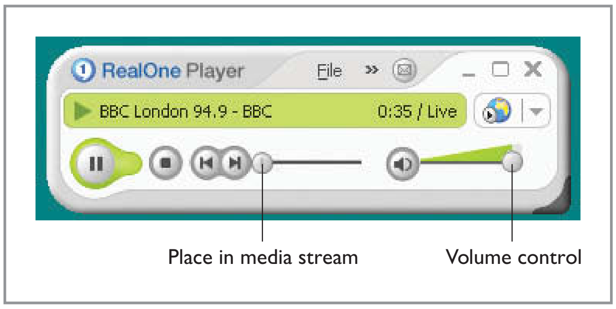


Figure 12.14 The RealOne Player® with two slider controls

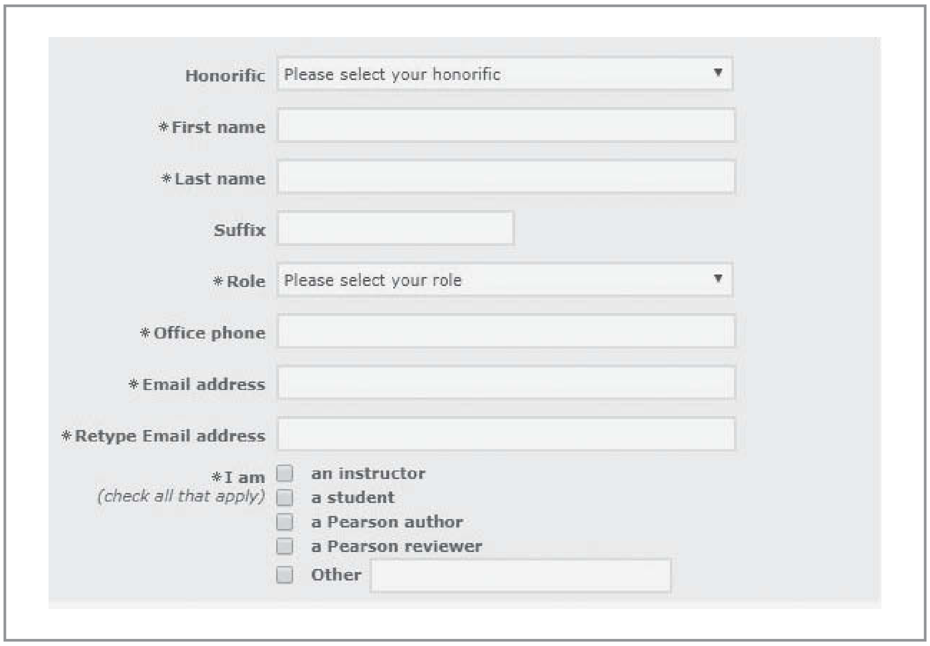
(Source: Courtesy of Real Networks, Inc.)

Form fill

Form filling is an interface style that is particularly popular with web applications. Form fill interfaces are used to gather information such as name and address. Figure 12.15 is a typical example of a form fill interface. This screenshot is taken from an online bookshop. The individual boxes are called **fields** and are frequently marked with an asterisk (*) to indicate that an entry is **mandatory**. This particular interface is a hybrid as it not only has form fill aspects but has other widgets too, including pull-down menus.

Form fill interfaces are best used when structured information is required. They can sometimes be automatically updated from a set of structured data stored on a personal computer. Examples of structured information include such things as:

- An individual's name and postal address required for mail order services
- Travel details, e.g. the airport from which one is flying, intended destination, time and date of departure
- Number and type of goods, e.g. ten copies of the DVD *The Sound of Music*.



Honorific	Please select your honorific ▼
*First name	
*Last name	
Suffix	
*Role	Please select your role ▼
*Office phone	
*Email address	
*Retype Email address	
*I am (check all that apply)	☐ an instructor
	☐ a student
	☐ a Pearson author
	☐ a Pearson reviewer
	☐ Other

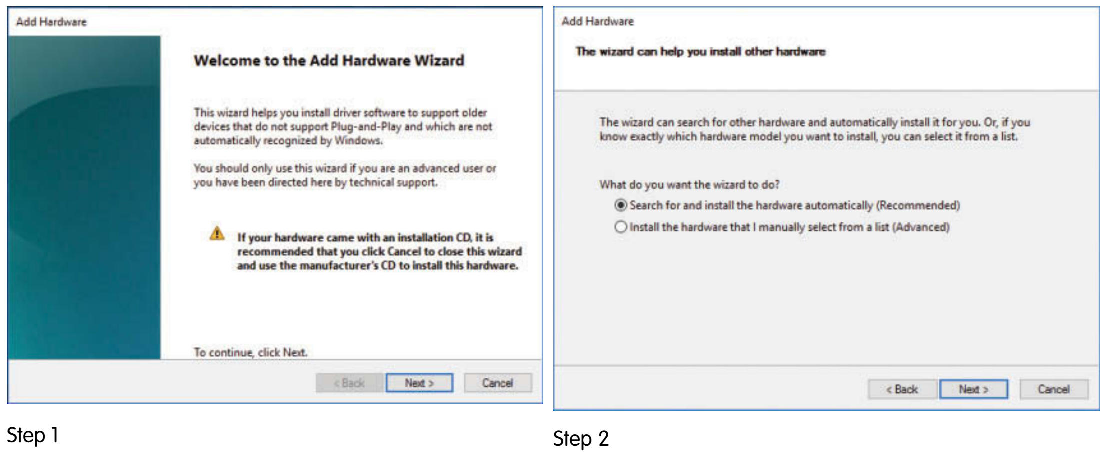
Figure 12.15 A typical form fill user interface

Wizards

Wizard is the name given to a style of interaction that leads people by the metaphorical hand (or pointer) step by step through a series of questions and answers, picklists and other kinds of widgets to achieve a task. Wizards are used to install hardware, applications and updates to operating systems. The great strength of wizards is that they present complex tasks in 'bite-sized' pieces. Figure 12.16 is a series of screenshots capturing the steps involved in installing a new item of hardware. This is only one possible route through the process of installing a new item of hardware; many others are possible.

Alerts

Figure 12.17 illustrates two different approaches (by the same software vendor) to alerting people to the presence of new mail. Figure 12.17(a) is the unobtrusive display of an envelope or mailbox symbol. In this instance one would expect the user to notice the message but in their own time. The second approach, in contrast, may interrupt people's work, so do not display this kind of alert box, which requires interaction, unless it is important, urgent or life-threatening. Allow the person to configure the application to turn off such alerts. Figure 12.17(b) is an illustration of an unobtrusive alert signalling the delivery of a new email message.



Add Hardware

Welcome to the Add Hardware Wizard

This wizard helps you install driver software to support older devices that do not support Plug-and-Play and which are not automatically recognized by Windows.

You should only use this wizard if you are an advanced user or you have been directed here by technical support.

If your hardware came with an installation CD, it is recommended that you click Cancel to close this wizard and use the manufacturer's CD to install this hardware.

To continue, click Next.

Add Hardware

The wizard can help you install other hardware

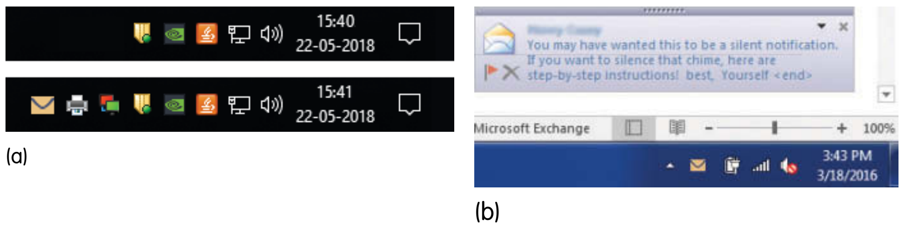
The wizard can search for other hardware and automatically install it for you. Or, if you know exactly which hardware model you want to install, you can select it from a list.

What do you want the wizard to do?

● Search for and install the hardware automatically (Recommended)

○ Install the hardware that I manually select from a list (Advanced)

Figure 12.16 The Microsoft Add Hardware wizard



[illegible]

You may have wanted this to be a silent notification. If you want to silence that chime, here are step-by-step instructions! best, Yourself <end>

Figure 12.17 Attracting attention

→ Chapter 21 discusses attention

Attracting attention is a simple enough matter – flash a light, use some other form of animation, ring a bell, and our attention is directed at that stimulus. However, the challenge of attracting and holding attention is to do so in a manner which:

- Does not distract us from the main task, particularly if we are doing something important, such as flying an aircraft or operating a complex or hazardous tool
- In certain circumstances *can* be ignored while in other circumstances *cannot* and *should not* be ignored
- Does not overwhelm the user of a system with more information than they can reasonably understand or respond to.

12.4 Psychological principles and interface design

As we mentioned above, there are many sites offering good guidelines to the interface designer. Apple, Android and Microsoft have style guides and many development environments will ensure that designs conform to the standards they are aiming at. There are also many issues applicable to the different contexts of design – websites, mobiles, etc. – that we discuss in Part III. In this section we present some guidelines deriving from the principles of psychology presented in Part IV.

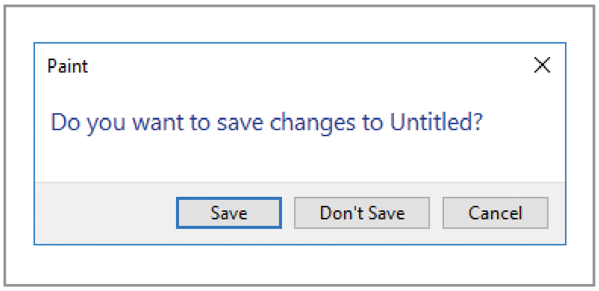
Cooper *et al*. (2007) argue that visual interface design is a central component of UX design as it combines graphic design, industrial design and visual information design. We deal with information design and the closely related area of visualizations in the next section. Designers need to know about graphic design, such as what shape, size, colour, orientation and texture screen objects should be. Designs should have a clear and consistent style. Recall the idea of a design language introduced in Chapter 3 and discussed in Chapter 9. The design language will be learned and adopted by people, so they will expect things that look the same to behave the same and, conversely, if things behave differently, make sure they look different. Cooper recommends developing a grid system to help to structure and group objects at the interface. In Chapters 8 and 14 we describe wireframes which are used to provide visual structure. However, we cannot hope to teach the whole of graphic design. We can, however, provide some guidelines that follow from our understanding of the psychology of people.

Guidelines from perception

Chapter 25 discusses perception and introduces a number of 'laws' of visual perception that have been developed by the 'gestalt' school of perception. Perception research also provides us with other fundamental aspects of people's abilities that should be considered when designing visual interfaces.

Using proximity to organize buttons

One of the *Gestalt* principles of perception is the observation that objects appearing close together in space or time tend to be perceived together. The usefulness of this law can be seen by contrasting the next two figures. Figure 12.18 is a standard Microsoft Windows 10 alert box with the buttons equally spaced. Figure 12.19 makes clear use of proximity. The **Cancel** and **Save** buttons are grouped away from the option **Don't Save**. This has the effect of seeing the two commands – **Save** and **Cancel** – as a pair and clearly separating from the potentially ambiguous **Don't Save**.


Paint

Do you want to save changes to Untitled?

Figure 12.18 Equally spaced buttons – Windows 10

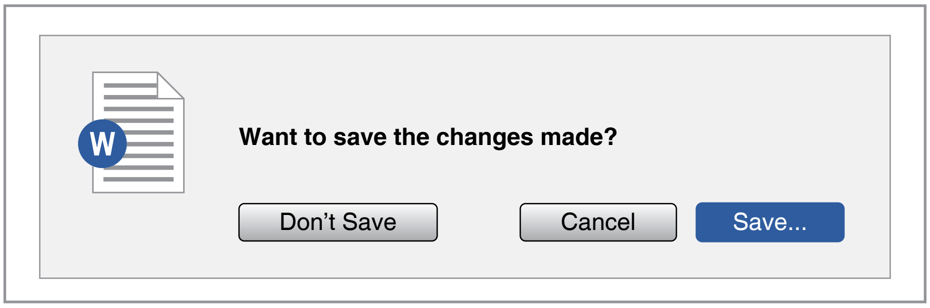

Want to save the changes made?

Figure 12.19 Buttons organized by proximity

Using similarity to organize files

A second *Gestalt* law we consider is that of **similarity**. Figure 12.20 is a screenshot of the contents of a folder. All of the files are ordered alphabetically, starting at the top left. The PowerPoint files are perceived as a contiguous block. This stands in sharp contrast to the file icons in Figure 12.21.

Using continuity to connect disconnected elements

A third *Gestalt* law is **continuity**. Disconnected elements are often seen to be part of a continuous whole. Figure 12.22 illustrates part of an MS Windows scrollbar that indicates that there is more of the document to be seen below the current window. The length of the slider is an indication of how much of the total document is visible – about 80 per cent here.

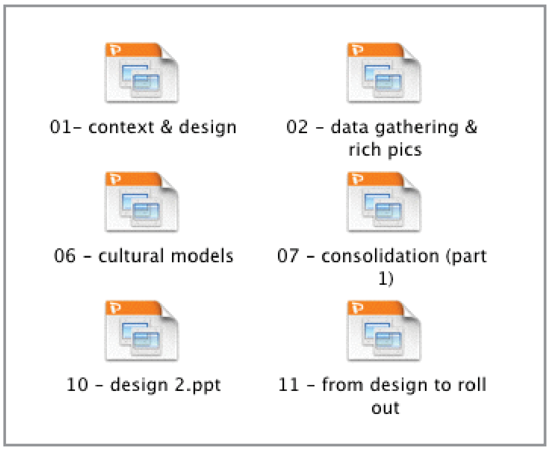

01– context & design
02 – data gathering & rich pics
06 – cultural models
07 – consolidation (part 1)
10 – design 2.ppt
11 – from design to roll out

Figure 12.20 Organizing files using similarity

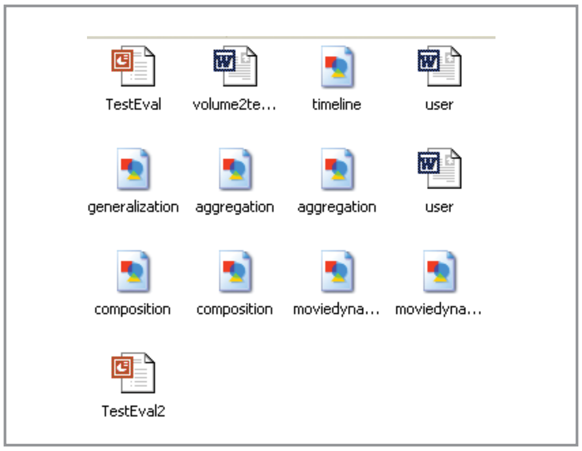


Figure 12.21 Disorganized files

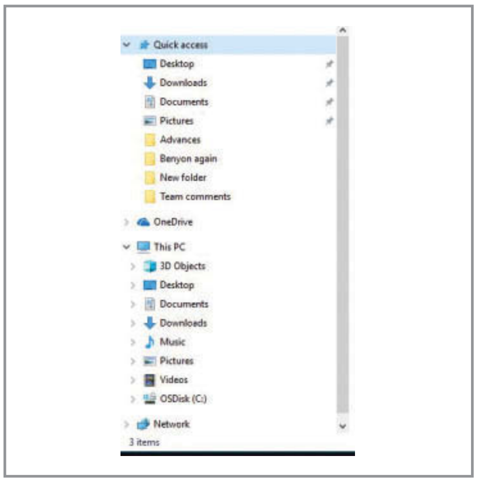


Figure 12.22 A Microsoft Windows 10 scrollbar

Principles from memory and attention

Our understanding of human abilities in remembering and attending to things also leads to a number of sound guidelines. Memory is usually considered in terms of our short-term or working memory and long-term memory. These are explained in detail in Chapter 21. Attention concerns what we focus upon.

Short-term (or working) memory

There is a widely quoted design guideline based on *Miller and his magic number*. George Miller (1956) found that short-term memory is limited to only 7 +/− 2 'chunks' of information. This principle has been used in HCI to suggest that menus should be restricted to about seven items, or web navigation bars should be seven items. While these are perfectly reasonable heuristics for designers to use, they do not derive from a limitation of short-term memory which is to do with how much most people can remember.

There is also an issue about how true this finding is. More recent work indicates that the real capacity of working memory is closer to three or four items; indeed, Cowan has argued for 4 +/− 1 (Cowan, 2002). However, the central observation that you should not expect people to remember lots of detail is well made.

Chunking

Chunking is the process of grouping information into larger, more meaningful units, thus minimizing the demands on working memory. Chunking is a very effective way of reducing memory load. An example of chunking at the interface is the grouping of meaningful elements of a task into one place (or dialogue). Think about setting up a standard template for a document. Among the things we have to remember to do are printing the document on the printer we wish to use, setting document parameters such as its size and orientation, setting the print quality or colour setting, and so on. Another example of chunking can be seen in Figure 12.23. Here a large number of options have been chunked into a single, expandable dialogue. The ▶ symbol indicates that the selection will expand if selected. Having clicked on the *Send to* button, the chunked dialogue expands to unpack a number of related options.

Time limitations

Memories, particularly those in short-term or working memory, are surprisingly short-lived, and even in ideal conditions they will persist for only 30 seconds. So, it is essential to make the presentation of important information persist (Figure 12.24) – do not flash

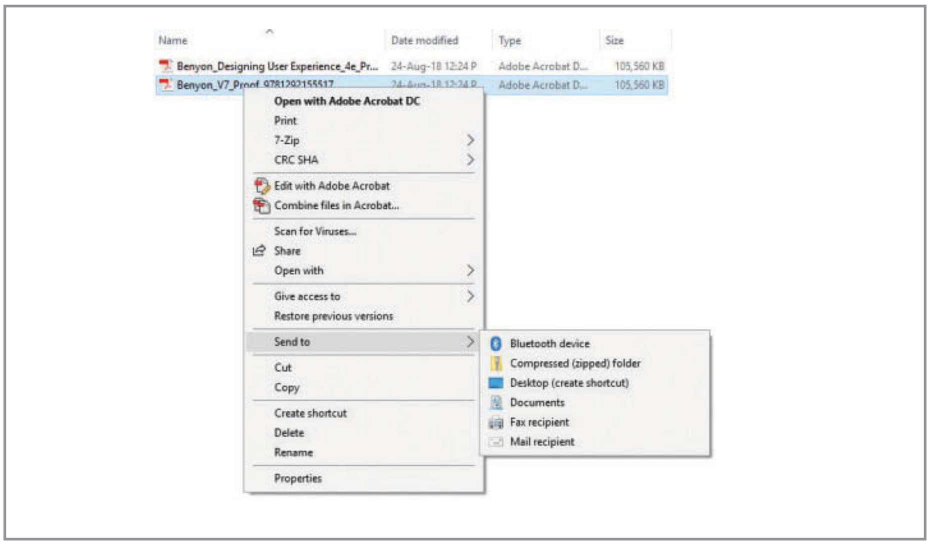
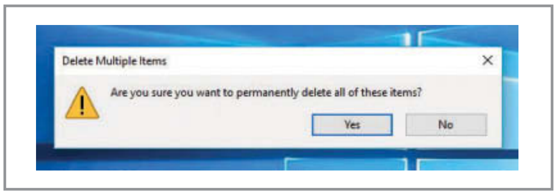

Name
Date modified
Type
Size
Benyon_Designing User Experience_4e_Pr... 24-Aug-18 12:24 P Adobe Acrobat D... 105,560 KB
Benyon_V7_Proof_9781292155517 24-Aug-18 12:24 P Adobe Acrobat D... 105,560 KB
Open with Adobe Acrobat DC
Print
7-Zip
CRC SHA
Edit with Adobe Acrobat
Combine files in Acrobat...
Scan for Viruses...
Share
Open with
Give access to
Restore previous versions
Send to
Cut
Copy
Create shortcut
Delete
Rename
Properties
Bluetooth device
Compressed (zipped) folder
Desktop (create shortcut)
Documents
Fax recipient
Mail recipient

Figure 12.23 A before and after chunked dialogue


Delete Multiple Items

Are you sure you want to permanently delete all of these items?

Figure 12.24 An example of a persistent alert box

an alert such as 'Cannot save file' onto a screen for a second or two and then remove it; insist that the user presses a button, typically 'OK'. 'OK' in this instance really means 'I acknowledge the message'.

Recall and recognition

Another guideline derived from our knowledge of memory is to design for recognition rather than recall. Recall is the process whereby individuals actively search their memories to retrieve a particular piece of information. Recognition involves searching your memory and then deciding whether the piece of information matches what you have in your memory store. Recognition is generally easier and quicker than recall.

→ Chapter 21 examines these issues more closely

Challenge 12.4

Find instances of designing for recall and recognition in software you use regularly. Hint: websites requiring form filling are often good sources of examples.

Designing for memory

Consider the drop down menu in Figure 12.25. This is an image of the formatting palette which is part of the version of Microsoft Word current at the time of writing. Microsoft has extensive usability laboratories and the design of this application will have benefited from a sound understanding of the capabilities of people. As such it is an excellent

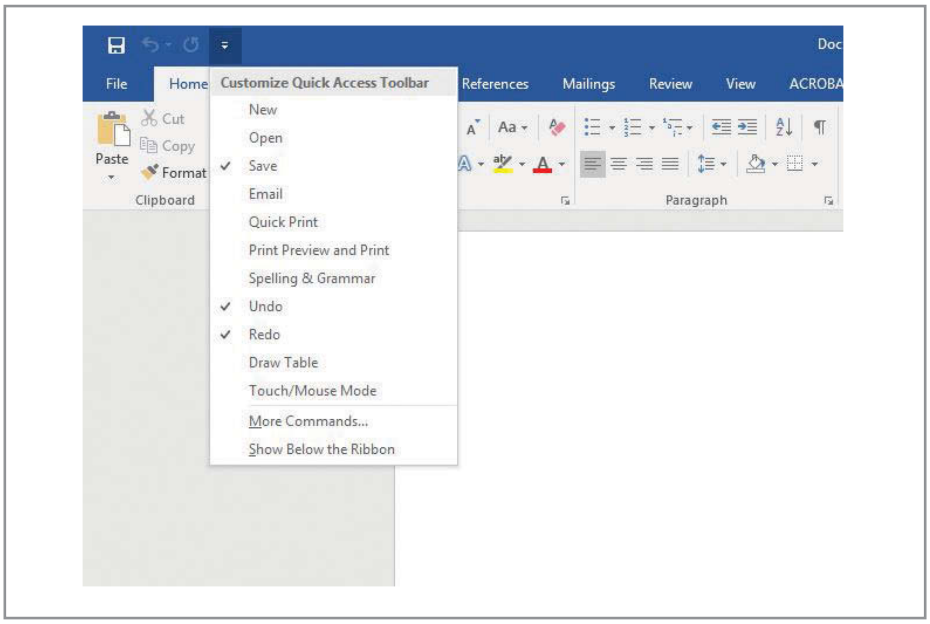


Figure 12.25 Drop down menu in Microsoft Word

example of designing for memory and embodies a whole series of design principles reflecting good design practice:

- The palette has been designed to use recognition rather than recall. The drop-down menus for style, name and size remove the need to recall the names of the fonts installed and the range of styles available. Instead the main memory mechanism is recognition. In addition to this, the burden on working memory is kept to a minimum using selection rather than having to memorize the name of a font (e.g. Zapf Dingbats) and then having to type it correctly in a dialogue box.
- The palette has been organized into four chunks – font, alignment and spacing, borders and shading, and document – which are logical groups or chunks of functions.
- Meaningful associations are used: **B** stands for bold, *I* for italic. It is good design practice to use these natural mappings.
- The palette relies on aspects of visual processing and the use of icons.

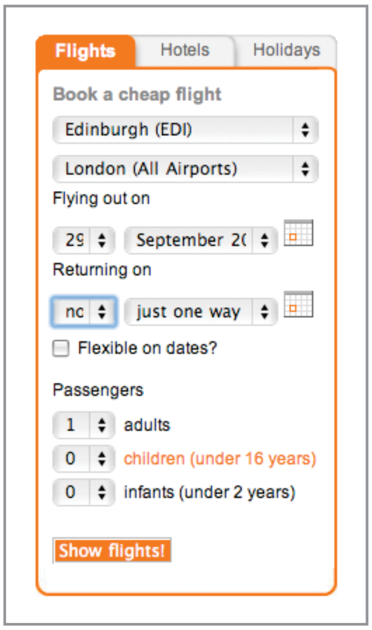
As we have seen, it is much easier to recognize something than to recall it. Novices prefer menus because they can scroll through the list of options until a particular command is recognized. Experts, however, have been reported as being frustrated in scrolling through a series of menus (particularly nested menus) and often prefer keyboard shortcuts instead (e.g. <alt>-F-P-<return> instead of select File menu/Print/OK). Interactive systems should be designed to accommodate both styles of working.

Further key evidence of the advantage of recognition over recall can be seen in the use of **picklists**. Picklists have two clear advantages over simply asking someone to recall a specific name, or any other piece of data. They offer help when we are faced with trying to recall something that is on the tip of our tongue or something that is ambiguous (as in the example of Figure 12.26, which identifies one of several London airports) or which may be difficult to spell. Consider the next two examples: imagine you are trying to book a flight from Edinburgh to London. You know that the target airport is not London Heathrow but one of the others and are confident that you will be able to recognize the specific airport without difficulty from the list more easily than from unaided memory. Figure 12.26 is an image of a standard web pull-down picklist.

London Stansted is easier to recognize than trying to remember (a) how to spell it – *Stanstead, Standsted* or *Stansted*? – and (b) the official airline abbreviation (STN).

The use of a picklist can also significantly improve the spelling of the documents we produce. Current versions of Microsoft Word identify misspelled words by underlining them with a red wavy line. Left-clicking on the word drops down a picklist of alternative spellings. This approach has also been adopted by modern visual (software) development environments where not only are misspelled commands identified but the syntax of commands is checked. Figure 12.27 is an illustration of this.

The recent use of **thumbnails** is another example of how recognition is more effective than recall. Figure 12.28 is a screenshot of the *My Pictures* folder on a computer running the Windows 7 operating system. The folder contains a number of thumbnails, that is, very small (thumbnail-sized) images of the contents of the files in the folder. Each is immediately recognizable and reminds the person of the original content.



Book a cheap flight

Edinburgh (EDI)

London (All Airports)

Flying out on

29 September 2(

Returning on

nc just one way

Flexible on dates?

Passengers

1 adults

0 children (under 16 years)

0 infants (under 2 years)

Figure 12.26 Flying to London
(Source: www.easyjet.co.uk/en/book/index.asp)

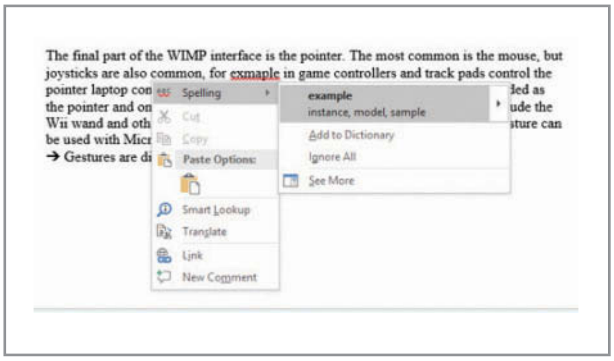
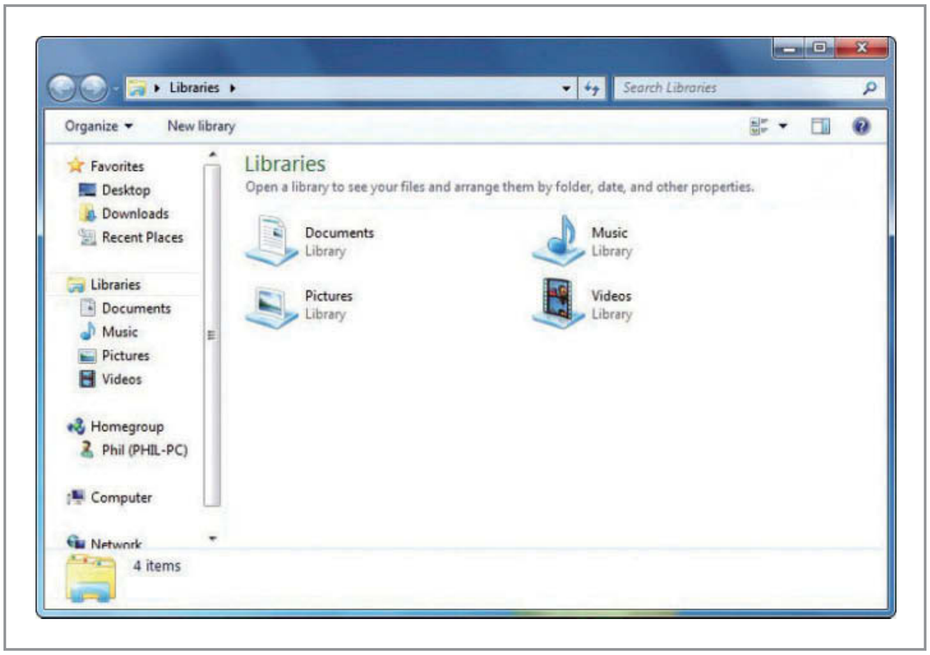

The final part of the WIMP interface is the pointer. The most common is the mouse, but joysticks are also common, for exmaple in game controllers and track pads control the pointer laptop con ded as the pointer and on ude the Wii wand and oth sture can be used with Micr
➔ Gestures are di

Figure 12.27 Spelling checker



Libraries

Open a library to see your files and arrange them by folder, date, and other properties.

Figure 12.28 Use of thumbnails

BOX 12.4

Colour blindness

The term colour blind is used to describe people with defective colour vision. Red–green colour blindness (i.e. the inability to distinguish reliably between red and green) is the most common form, affecting approximately 1 in 12 men (8 per cent) and 1 in 25 women (4 per cent). It is a genetic disorder with a sex-linked recessive gene to blame – hence the greater number of men being affected. A second and rarer form of colour blindness affects the perception of the colours blue–yellow. The rarest form of all results in monochromatic vision in which the sufferer is unable to detect any colour at all.

Designing with colour

Colour is very important to us. To describe someone as being colourless is to say that they are without character or interest. The design language adopted by Microsoft (discussed in Chapter 9) makes use of colourful tiles for its design and Apple has long favoured smooth blue and graphite grey as its livery.

Aaron Marcus's excellent book *Graphic Design for Electronic Documents and User Interfaces* (Marcus, 1992) provides the following rules:

Rule 1	Use a maximum of 5 +/− 2 colours.
Rule 2	Use foveal (central) and peripheral colours appropriately.
Rule 3	Use a colour area that exhibits a minimum shift in colour and/or size if the colour area changes in size.
Rule 4	Do not use simultaneous high-chrome, spectral colours.
Rule 5	Use familiar, consistent colour codings with appropriate references.

Table 12.1 illustrates a number of Western (Western Europe, the United States and Australia) denotations as identified by Marcus. These guidelines are, of course, just that – guidelines; they may not suit every situation but should at the very least provide a sound starting point. One final caveat – colour connotations can vary dramatically even within a culture. Marcus notes that the colour blue in the United States is interpreted differently by different groups – for healthcare professionals it is taken to indicate death; for movie-goers it is associated with pornography; for accountants it means reliability or corporateness (think of 'Big Blue' – IBM).

Table 12.1 Some Western colour conventions

Red	Danger, hot, fire
Yellow	Caution, slow, test
Green	Go, okay, clear, vegetation, safety
Blue	Cold, water, calm, sky
Warm colours	Action, response required, proximity
Cool colours	Status, background information, distance
Greys, white and blue	Neutrality

Source: After Marcus, A. *Graphic Design for Electronic Documents and User Interfaces*, 1st edition, © 1992. Printed and electronically reproduced by permission of Pearson Education, Inc., Upper Saddle River, New Jersey

Error avoidance design guidelines

The following design guidelines have been drawn (and edited) from Reason and Norman's design principles for minimizing error (*cf*. Reason, 1990, p. 236):

→ Human error is covered in Chapter 21

- Use knowledge both in the world and in the head in order to promote a good conceptual model of the system; this requires consistency of mapping between the designer's model, the system model and the user's model.
- Simplify the structure of tasks so as to minimize the load upon vulnerable cognitive processes such as working memory, planning or problem solving.
- Make both the execution and the evaluation sides of an action visible. Visibility in regard to the former allows users to know what is possible and how things should be done; visibility on the evaluation side enables people to gauge the effects of their actions.
- Exploit natural mappings between intentions and possible actions, between actions and their effects on the system, between the actual system state and what is perceivable, and between the system state and the needs, intentions and expectations of the user.
- Exploit the power of constraints, both natural and artificial. Constraints guide people to the next appropriate action or decision.
- Design for errors. Assume that they will happen, then plan for *error recovery*. Try to make it easy to reverse operations and hard to carry out non-reversible ones. Exploit forcing functions such as wizards that constrain people to use a limited range of operations.
- When all else fails, standardize actions, outcomes, layouts, displays, etc. The disadvantages of less than perfect standardization are often compensated for by the increased ease of use. But standardization for its own sake is only a last resort. The earlier principles should always be applied first.

Error message design guidelines

- Take care with the wording and presentation of alerts and error messages.
- Avoid using threatening or alarming language in messages (e.g. fatal error, run aborted, kill job, catastrophic error).
- Do not use double negatives as they can be ambiguous.
- Use specific, constructive words in error messages (e.g. avoid general messages such as 'invalid entry' and use specifics such as 'please enter your name').
- Make the system 'take the blame' for errors (e.g. 'illegal command' versus 'unrecognized command').
- DO NOT USE ALL UPPERCASE LETTERS as it looks as if you are shouting – instead, use a mixture of uppercase and lowercase.
- Use attention-grabbing techniques cautiously (e.g. avoid over-using 'blinks' on web pages, flashing messages, 'you have mail', bold colours, etc.).
- Do not use more than four different font sizes per screen.
- Do not over-use audio or video.
- Use colours appropriately and make use of expectations (e.g. red = danger, green = ok).

Principles from navigation

Navigation is discussed in Chapter 25, highlighting the importance of people having both survey knowledge and route knowledge in understanding and wayfinding through an environment. Apple user experience guidelines agree:

> **Give People a Logical Path to Follow.** People appreciate knowing where they are in an app and getting confirmation that they're on the right path. *Make the path through the information you present logical and easy for users to predict.* In addition, be sure to provide markers – such

as back buttons – that users can use to find out where they are and how to retrace their steps. *In most cases, give users only one path to a screen*. If a screen needs to be accessible in different circumstances, consider using a modal view that can appear in different contexts.

Source: See Further thoughts box

FURTHER THOUGHTS

FURTHER THOUGHTS

Apple's user experience guidelines for iOS apps

Focus on the Primary Task
Elevate the Content that People Care About
Think Top Down
Give People a Logical Path to Follow
Make Usage Easy and Obvious
Use User-Centric Terminology
Minimize the Effort Required for User Input
Downplay File-Handling Operations
Enable Collaboration and Connectedness
De-emphasize Settings
Brand Appropriately
Make Search Quick and Rewarding
Entice and Inform with a Well-Written Description
Be Succinct
Use UI Elements Consistently
Consider Adding Physicality and Realism
Delight People with Stunning Graphics
Handle Orientation Changes
Make Targets Fingertip-Size
Use Subtle Animation to Communicate
Support Gestures Appropriately
Ask People to Save Only When Necessary
Make Modal Tasks Occasional and Simple
Start Instantly
Always Be Prepared to Stop
Don't Quit Programmatically
If Necessary, Display a License Agreement or Disclaimer

For iPad:

Enhance Interactivity (Don't Just Add Features)
Reduce Full-Screen Transitions
Restrain Your Information Hierarchy
Consider Using Popovers for Some Modal Tasks
Migrate Toolbar Content to the Top

Source: http://www.designprinciplesftw.com/collections/ios-user-experience-guidelines

12.5 Information design

In addition to designing screens and individual widgets for people to interact with a system or device, UX designers need to consider how to lay out the large amounts of data and information that are often involved in applications. Once designers have worked out how best to structure and organize the information, they need to provide people with methods to interact with it. The tools and techniques for navigating through large amounts of information have a big impact on the inferences people will be able to make from the data and the overall experience that people will have.

→ See also the discussion of information spaces in Chapter 18

Jacobson (2000) argues that the key feature of information design is that it is design dealing with meanings rather than materials. Information design is essentially to do with sense-making, with how to present data (often in large amounts) in a form that people can easily understand and use. Information designers have to understand the characteristics of the different media being used to present data and how the medium affects how people move through structures.

Information design is traditionally traced back to the work of Sir Edward Playfair in the eighteenth century and to the work of French semiologist Jacques Bertin (1981). Bertin's theories of how to present information and the different types of visualizations have been critical to all work since. The work of Edward Tufte (1983, 1990, 1997) shows just how effective good information design can be (see Box 12.5). He gives numerous examples of how, in finding the best representation for a problem, the problem is solved. Clarity in expression leads to clarity of understanding. Tufte describes various ways of depicting quantitative information, such as labelling, encoding with colours or using known objects to help get an idea of size. He discusses how to represent multivariant data in the two-dimensional space of a page or a computer screen and how best to present information so that comparisons can be made. His three books are beautifully illustrated with figures and pictures through the centuries and provide a thoughtful, artistic and pragmatic introduction to many of the issues of information design.

BOX 12.5

Edward Tufte

In the introduction to *Visual Explanations*, Tufte (1997, p. 10) writes:

> My three books on information design stand in the following relation:
>
> *The Visual Display of Quantitative Information* (1983) is about pictures of numbers, how to depict data and enforce statistical honesty.
>
> *Envisioning Information* (1990) is about pictures of nouns (maps and aerial photographs, for example, consist of a great many nouns lying on the ground). Envisioning also deals with visual strategies for design: color, layering and interaction effects.
>
> *Visual Explanations* (1997) is about pictures of verbs, the representation of mechanism and motion, or process and dynamics, or causes and effects, of explanation and narrative. Since such displays are often used to reach conclusions and make decisions, there is a special concern with the integrity of the content and the design.

Figure 12.29 is one of Tufte's designs and shows a patient's medical history involving two medical and two psychiatric problems.

Harry Beck's map of the London Underground is often cited as an excellent piece of information design. It is praised for its clear use of colour and its schematic structure – not worrying about the actual location of the Underground stations but instead concerned with their linear relationships. The original map was produced in 1933 and the style and concepts have remained until now. However, it is interesting to note how nowadays – with the proliferation of new lines – the original structure and scheme are breaking down. With only a few Underground lines, the strong visual message could be conveyed with strong colours; with a larger number of lines the colours are no longer easily discernible from each other. Figure 12.30 shows the map from 1933 contrasted with a recent version.

Another key player in the development of information architecture and information design is Richard Saul Wurman. His book *Information Architects* (Wurman, 1997) provides a feast of fascinating images and reflections on the design process by leading information designers. Wurman's own contribution dates from 1962 and includes a

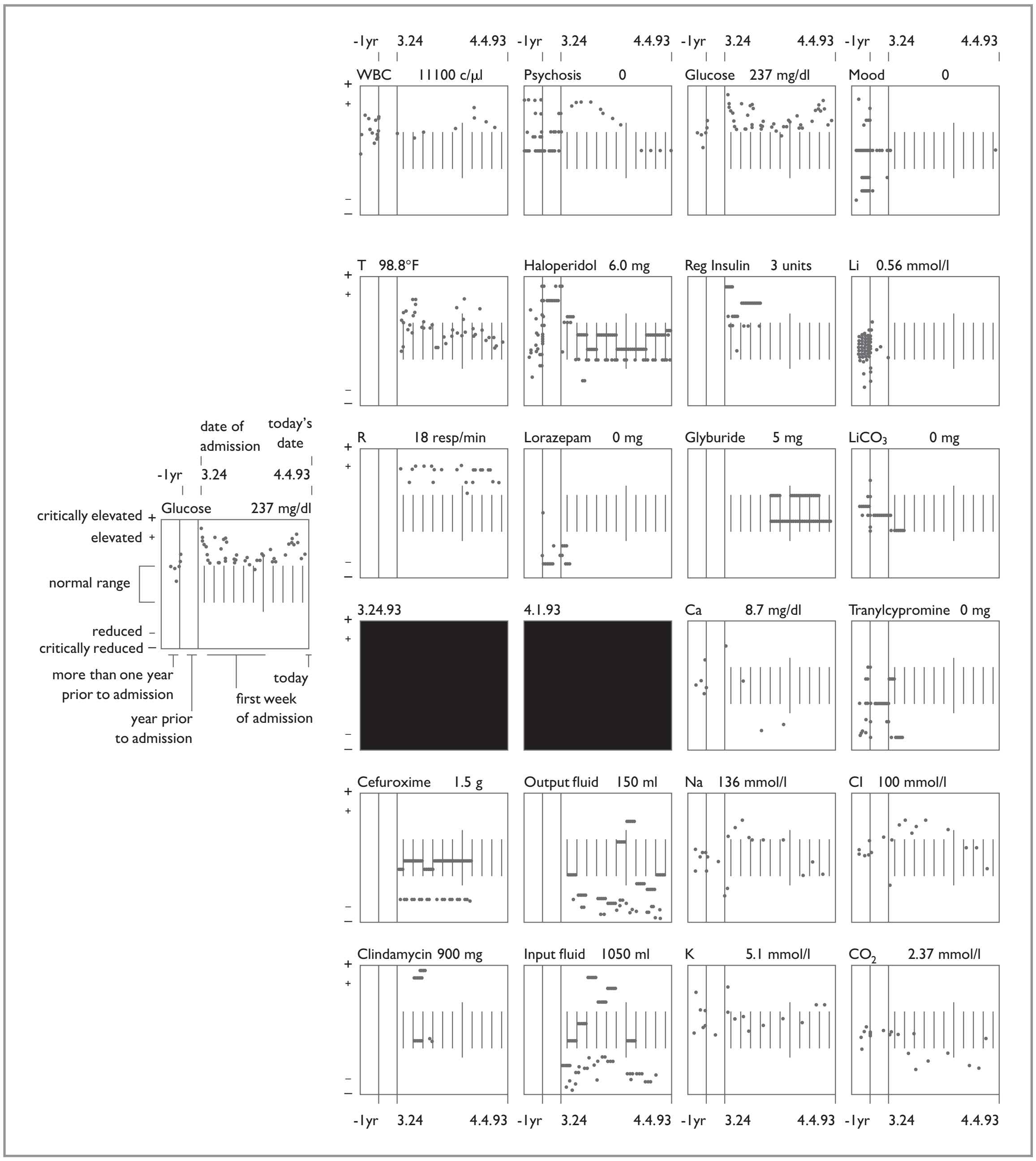


Figure 12.29 Examples of Tufte's work

(Source: After Tufte (1997), p. 110 and p. 111. Courtesy of Edward R. Tufte and Seth M. Powsner)

wide variety of information design cases, from maps comparing populations to books explaining medical processes, to his *New Road Atlas: US Atlas* (Wurman, 1991), based on a geographical layout, with each segment taking one hour to drive. Figure 12.31 shows an example from his *Understanding USA* book (Wurman, 2000).

A number of authors are keen to ground information design in theory – particularly theories of perception and cognition. Indeed, some of these theoretical positions, such as the *Gestalt* principles described above, are useful. General design principles of avoiding clutter, avoiding excessive animations and avoiding clashing colours also help make displays understandable. Bertin's theories and modern versions such as that of Card (2012) are useful grounding for anyone working in the area. Card (2012) offers a detailed taxonomy of the

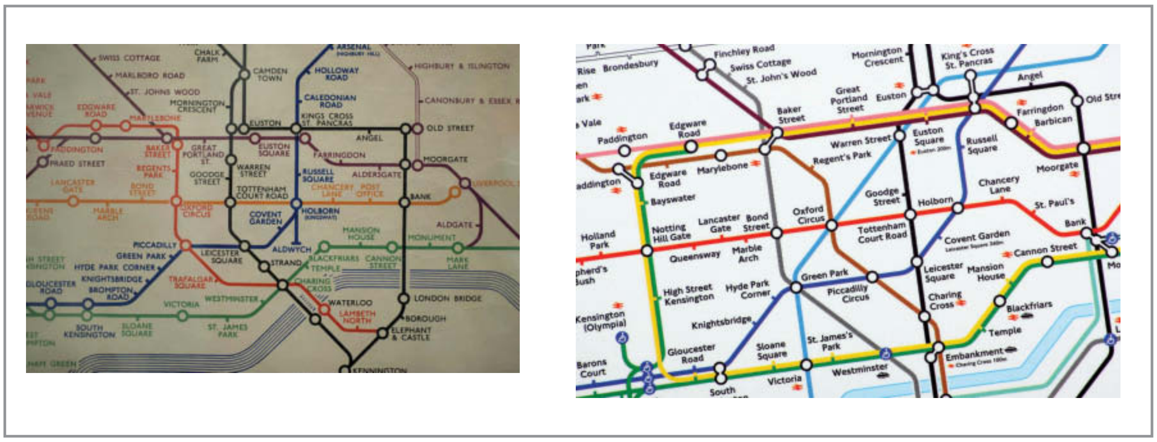


Figure 12.30 Maps of the London Underground rail network: left, in 1933; right, now

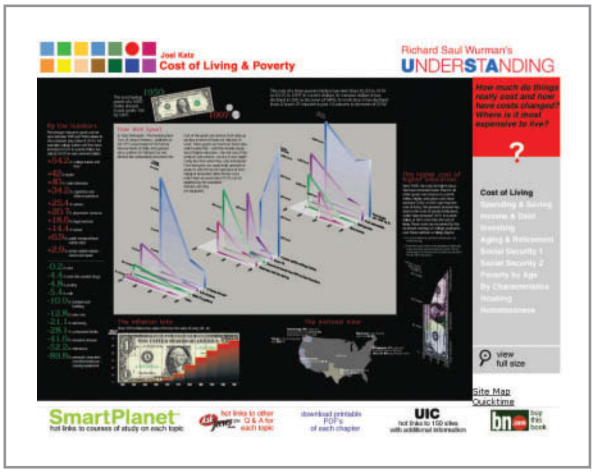


Figure 12.31 Illustration from Richard Saul Wurman's book Understanding USA

(Source: Wurman, 2000, designed by Joel Katz)

various types of visualization and provides details on different types of data that designers can deal with. He also discusses the different visual forms that can be used to represent data.

Essentially, though, information design remains a design discipline rather than an engineering one. There are many methods to help designers understand the problems of information design in a particular context (and taking a human-centred view is the most important), but there can be no substitute for spending time critiquing great designs and looking at the reflection of designers on their creative and thinking processes. Readers are encouraged to follow up the references at the end of this chapter.

When developing a scheme of information design in a given context, designers should realize that they are developing visual 'languages'. The visual language of information design is an important part of these. Designers will imbue colours, shapes and layouts with meanings that people have to come to understand.

← Design languages are discussed in Chapter 9

12.6 Visualization

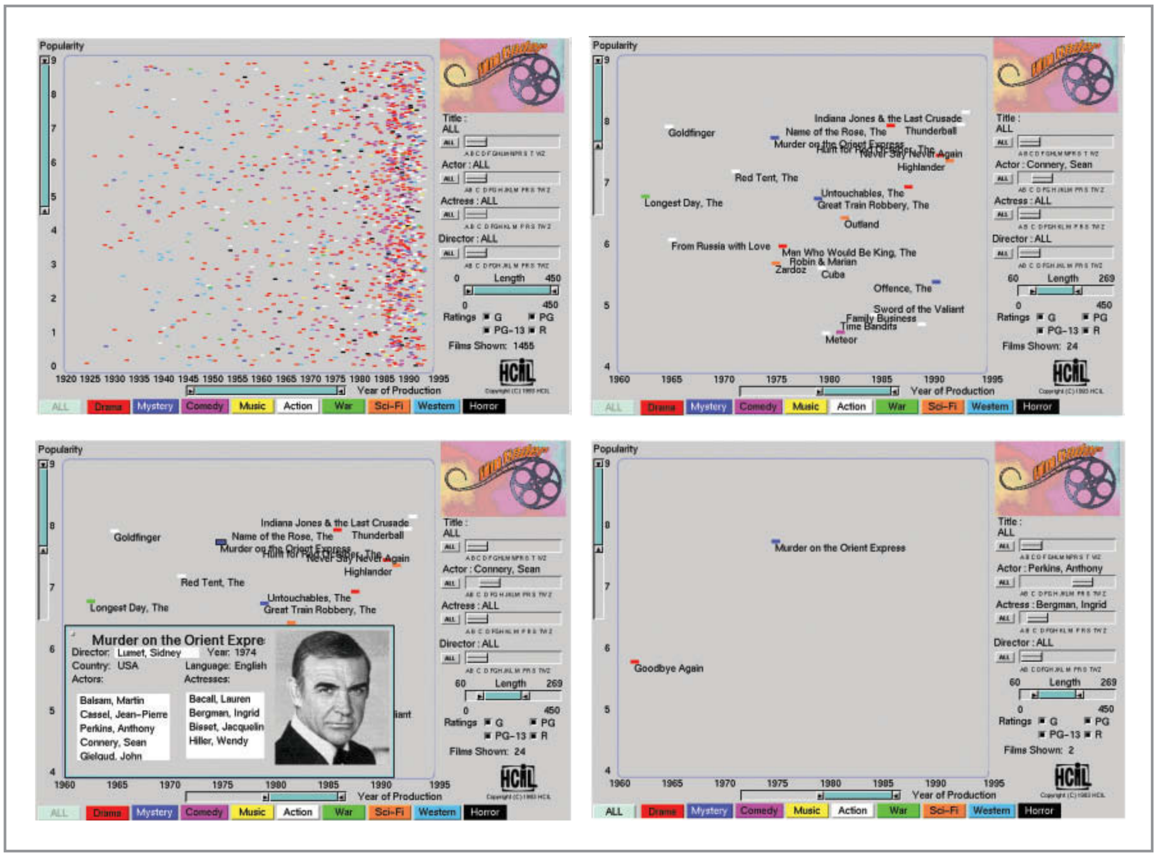
The other key feature of information design that the modern information architect or UX designer might get involved with is interactive visualization. With the vast amounts of data available, novel ways of presenting and interacting with these are necessary. Card *et al*. (1999) is an excellent set of readings covering many of the pioneering systems. Spence (2001) provides a good introduction to the area, while Card (2012) offers a thorough and accessible treatment of the options. Interactive visualizations are concerned with harnessing the power of novel interactive techniques with novel presentations of large quantities of data. Indeed, Card (2012) argues that visualization is concerned with 'amplifying cognition'. It achieves this through:

- Increasing the memory and processing resources available to people
- Reducing the search for information
- Helping people to detect patterns in the data
- Helping people to draw inferences from the data
- Encoding data in an interactive medium.

Ben Shneiderman has long been a designer of great visualizations (see www.cs.umd.edu/nben/index.html). He has a 'mantra', an overriding principle for developing visualizations:

Overview first, zoom and filter, then details on demand.

The aim of the designer is to provide people with a good overview of the extent of the whole dataset, to allow zooming in to focus on details when required, and to provide dynamic queries that filter out the data that is not required. Card (2012) includes retrieval by example as another key feature. So rather than having to specify what is required in abstract terms, people request items similar to one they are viewing. Ahlberg and Shneiderman's (1994) Film Finder is an excellent example of this (Figure 12.32).



Murder on the Orient Expre
Director: Lumet, Sidney Year: 1974
Country: USA Language: English
Actors: Balsam, Martin; Cassel, Jean-Pierre; Perkins, Anthony; Connery, Sean; Gielgud, John
Actresses: Bacall, Lauren; Bergman, Ingrid; Bisset, Jacqueline; Hiller, Wendy

Figure 12.32 Film Finder

(Source: Ahlberg, C. and Shneiderman, B. (1994) Visual information seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays, Proceedings of the CHI'94 Conference, pp. 313–317. © 1994 ACM, Inc. Reprinted by permission.)

In the first display we see hundreds of films represented as coloured dots and organized spatially in terms of year of release (horizontal axis) and rating (vertical axis). By adjusting the sliders on the right-hand side, the display zooms in on a selected part of the first display, allowing names to be revealed. Effectively the sliders provide dynamic queries on the data, allowing people to focus in on the part that is of interest. Clicking on a film brings up its details, allowing this to be used for retrieval-by-example-style further searches.

Another classic example of a visualization is ConeTree (Figure 12.33). Various facilities are available that allow people to 'fly' around the display, identifying and picking out items of interest. Once again the interactive visualization allows for overview first, zoom and filter, and details on demand. The key thing with visualizations is to facilitate 'drilling down' into the data.

Figure 12.34 shows the display of the stock market at SmartMoney.com. This display is known as a 'tree map'. The map is colour-coded from red through black to green, indicating a fall in value, through no change to a rise in value. The brightness of colour indicates the amount of change. Companies are represented by blocks, the size of the block depicting the size of the company. Running a mouse over the block brings up the name and clicking on it reveals the details.

Figure 12.35 shows a different type of display in which connections are shown by connecting lines. It is an online thesaurus that demonstrates the 'fish-eye' capability, which again allows for the focus and context feature required by visualizations. This

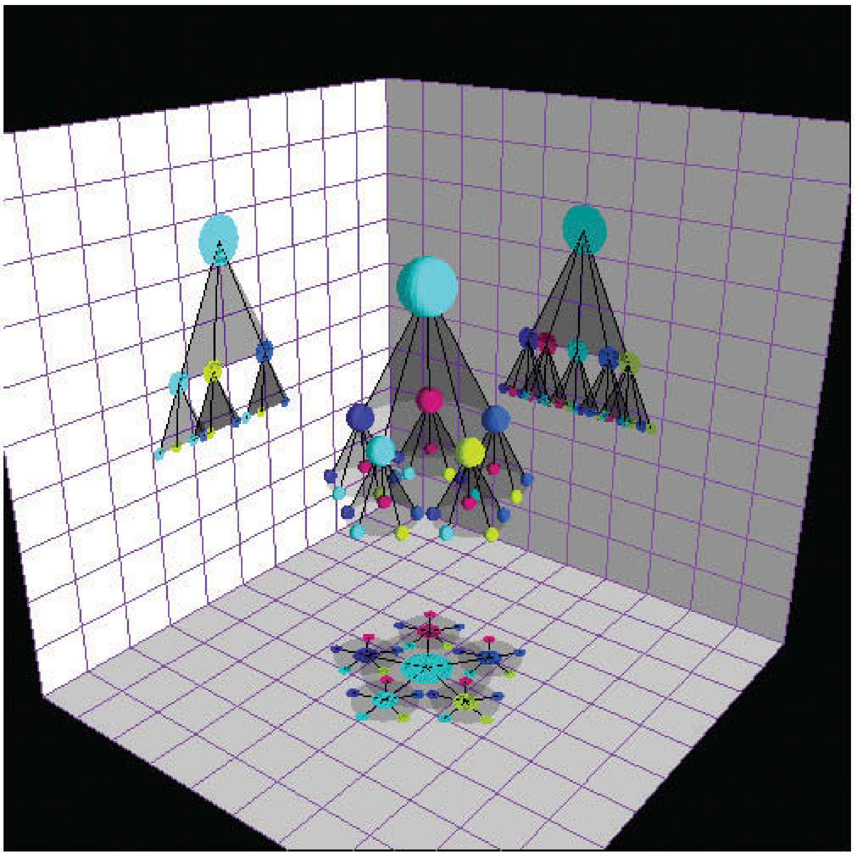
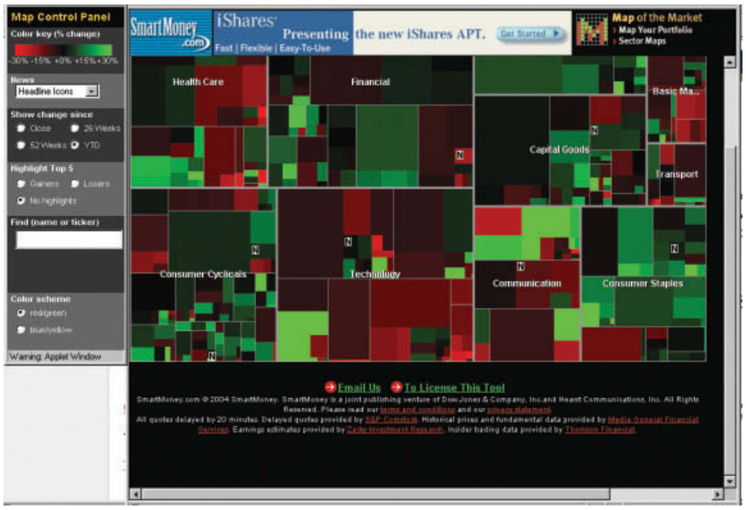


Figure 12.33 ConeTree



SmartMoney.com © 2004 SmartMoney. SmartMoney is a joint publishing venture of Dow Jones & Company, Inc.and Hearst Communications, Inc. All Rights Reserved. Please read our terms and conditions and our privacy statement.
All quotes delayed by 20 minutes. Delayed quotes provided by S&P Comstock. Historical prices and fundamental data provided by Media General Financial Services. Earnings estimates provided by Zacks Investment Research. Insider trading data provided by Thomson Financial.

Figure 12.34 SmartMoney.com

(Source: www.smartmoney.com/map-of-the-market © SmartMoney 2004. All rights reserved. Used with permission. SmartMoney is a joint venture of Dow Jones & Company, Inc. and Hearst Communications, Inc.)

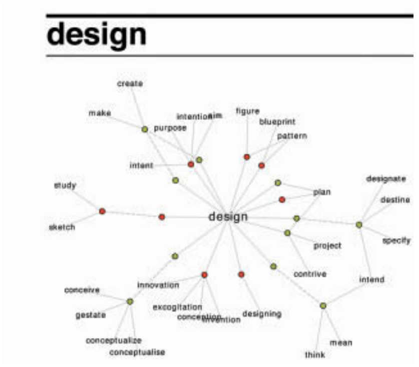


Figure 12.35 The Visual Thesaurus™

(Source: www.visualthesaurus.com/app/view)

allows users to see what is nearby and related to the thing on which they are focusing. There are many more exciting and stimulating visualizations built for specific applications. Card (2012) lists many and Card *et al*. (1999) discuss specific designs and their rationale.

Card (2012) argues that the key decision in any visualization is to decide which attributes of an object are to be used to spatially organize the data. In Film Finder it is rating and year; in SmartMoney.com it is the market sector. Once this has been decided, there are relatively few visual distinctions that can be made. The designer can use points, lines, areas or volumes to mark different types of data. Objects can be connected with lines or enclosed inside containers. Objects can be distinguished in terms of colour, shape, texture, position, size and orientation. Other visual features that can be used to distinguish items include resolution, transparency, arrangements, the hue and saturation of colours, lighting and motion.

There are a number of novel visualization applications that are available to view certain websites and other large datasets such as collections of photos. Cool Iris is one such application, facilitating panning, zooming and moving through the data in an extremely engaging way. DeepZoom is a zoomable interface based on Silverlight from Microsoft and Adobe market Papervision, which provides similar functionality based on Flex.

Summary and key points

The design of visual interfaces is a central skill for interactive system designers. There are principles of aesthetics to consider (we covered aesthetics in Chapter 5), but mostly designers need to concentrate on understanding the range of 'widgets' that they have available and how they can be best deployed. It is how the overall interaction works as a whole that is important.

- Graphical user interfaces use a combination of WIMP features and other graphical objects as the basis of their design.
- Design guidelines are available from work in psychology and perception and from principles of graphic design.
- In information design, interactive visualizations need to be considered when there is a large amount of data to be displayed.

Exercises

1 Examine the tabbed dialogue widget shown in Figure 12.25. Which of the major components of human cognition are being addressed in the design?

2 (*Advanced*) I pay my household credit card bill every month using my debit card (which is used for transferring money from my bank account). The procedure is as follows:

- I have to phone the credit card company on a 12-digit telephone number.
- Then from the spoken menu I press 2 to indicate I wish to pay my bill.
- I am instructed to enter my 16-digit credit card number followed by the hash key.
- I am then told to enter the amount I want to pay in pounds and pence (let's imagine I wish to pay £500.00 – seven characters).
- Then I am told to enter my debit card number (16 digits) followed by the hash key.
- Then I am asked for the debit card's issue number (2 digits).
- Then the system asks me to confirm that I wish to pay £500.00 by pressing the hash key.
- This ends the transaction. The number of keystrokes totals 12 + 1 + 16 + 7 + 16 + 2 + 1 = 55 keystrokes on a handset which does not have a backspace key.

What design changes would you recommend to reduce the likelihood of making a mistake in this complex transaction?

Further reading

Card, S. (2012) Information visualizations. In Jacko, J.A. (ed.) *The Human–Computer Interaction Handbook*, 3rd edn. CRC Press, Taylor and Francis, Boca Raton, FL, pp. 515–548.

Marcus, A. (1992) *Graphic Design for Electronic Documents and User Interfaces*. ACM Press, New York.

Getting ahead

Cooper, A., Reiman, R. and Cronin, D. (2007) *About Face 3: The Essentials of Interaction Design*. Wiley, Hoboken, NJ. *Provides a wealth of detailed interface design guidance and numerous examples of good design.*

Web links

For further information on Horton's approach to icon design see **www.horton.com**

The accompanying website has links to relevant websites. Go to
www.pearsoned.co.uk/benyon

Comments on challenges

Challenge 12.1

Saying 'Computer' puts the computer into the correct mode to receive commands. In the lift the only commands the system responds to are instructions on which deck to go to. Thus the context of the interaction in the lift removes the need for a command to establish the correct mode.

Challenge 12.2

There are a number of issues with the icons. Some seem very old fashioned (such as the old telephone icon) but have become so ingrained that it is difficult to change them. Several of them require you to know what the application is. Some are a bit too similar and easily confused. It is interesting to notice just how much effort goes into designing things such as icons, and yet designs can still be sub-optimal.

Challenge 12.3

Radio buttons for the colour scheme – only one option can be chosen. The incoming mail preferences use checkboxes since multiple preferences can be selected.

Challenge 12.4

Again, instances abound. An example of design for recognition is the provision of a drop-down list of all airports for a particular city destination in a flight booking site rather than expecting customers to recall which airports exist and then to type in the exact name.