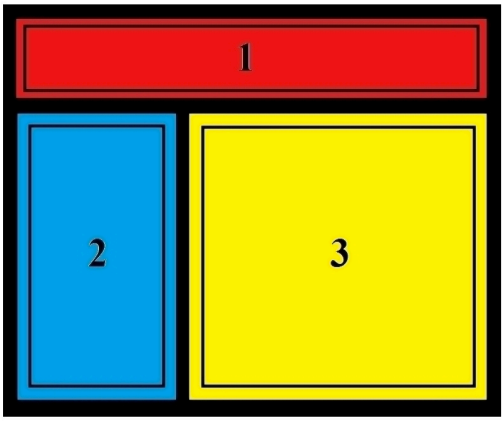
# Organising Content for the Best Results

When users arrive at a website they need to see where they are, what is available and where they can go. If they cannot determine these aspects within a reasonable length of time (which to most users is probably just a few seconds) then there is the serious threat of regular site abandonments. Users leaving a website within a few seconds without visiting any other pages are described as 'bounces' and a 'bounce rate' represents the percentage of visitors who enter the site and 'bounce' (leave the site) rather than continue viewing other pages within the same site.

**The Design Solution**

The key to solving this problem is to establish a visual hierarchy; one that is instantly visible and satisfies the initial questions a user has when they first arrive at a website. 3 key questions when visiting a website:

- 'Where am I?
- 'What's here?'
- 'Where can I go?'




The first rectangle along the top of the page should answer the question, 'Where am I?' to help orient the user, as they can arrive at the site from any number of possible avenues, which do not necessarily indicate explicitly, prior to arrival, where they are actually being directed.
The third panel is usually consuming the largest amount of screen space and is found in the centre of the web page and should address the question of, 'What's here?'.
The second panel, on the left-hand side of the web page, should instantly show where the user can go; answering the third design question.

The positioning of the areas, shown above, should not be considered as concrete and immovable, but they help to convey a general approach that answers the three essential webpage defining questions.

Web design should establish a clear, visual hierarchy on every page within a site. Visual hierarchies help to immediately inform the user of the relation between items within a page and what function each item affords the user. A number of established design norms also help to communicate function within a hierarchy, such as the use of imagery or icons to represent certain operations (e.g. baskets for 'storing' items to purchase at a later point in eCommerce). Design norms may seem fusty or restricting but common and familiar representations, styles, layouts and structures **help the user instantly recognise the 'what?', 'where?' and 'how?'** underpinning human-computer interactions.

**Why Choose a Visual Hierarchy?**

Visual hierarchies are used when you want to organise content into a logical and cohesive order. Without organisation content is simply a random assembly of items which requires the user to scan much larger areas of the display in order to identify target items. By arranging all of the 'Where am I?', 'What's here?' and 'Where can I go?' information into three distinct regions of the user interface the user only has to check the region of the display relevant to their current aims and objectives.

**Implementing Visual Hierarchies**

You will always need to include some clear homepage banner at the top of the site that informs the user exactly where they are. Placing lots of other information or visuals in this region can reduce the impact of your logo or company name, so add a little bit of dead space between the homepage banner and the rest of

the site contents. The company name or logo in the top banner should be maintained across all pages of the website; serving as a "You are here" signpost when they are moving through different pages. There are other types of signpost you can include in your design, such as titled sections, breadcrumbs, stacked card patterns/navigation tabs and simple headers. However, these should never be in place of a site name or logo in the top banner, as this serves as a single and constant navigation point when the user is on any other page in the website.
Review the contents you will be offering in your site. **Common themes should appear and these are what the visual hierarchy will be based on**.
Now determine what you will allow the user to do on your site; whether this involves navigating from page to page, viewing contents in the form of menus or two-panel selectors and so on.
Space and align the content, navigation devices and any other elements you wish to include using page margins, line spacing, clearly defined boundaries, spaces between distinct groups of items and different user interface design patterns, text and label justification etc.
Finally, position all of the content into the most logical arrangement

**Web design is a careful balancing act between sorting the contents of your site and satisfying the expectations of the user.**

## Progressive Disclosure

**'Progressive Disclosure'** involves the division of content into distinct, yet instantly accessible, regions of the user interface. By splitting information into separate pockets, the user can attend to their specific task demands without the presence of irrelevant information, graphics or any other unrelated content. Users are looking for simple designs, minus the clutter involved in having all options displayed at once, yet with the capacity to carry out a range of different tasks without being taken to different pages, windows or spatial locations within the user interface. **Progressive disclosure satisfies these demands by maintaining all the available options in view whilst providing them with the means to access the finer details of a task and a fixed panel, window or page to operate in.**

Restricting the display to only information the user has requested has a number of inherent benefits. Firstly, this gives the user a sense of control over events in the user interface; **they see what they want, when they want - no more, no less**. Secondly, when they are performing a task they will see only the related information; saving the user from assessing the user interface in an attempt to tease out relevant information from the array. Thirdly, the user can approach component tasks step-by-step, without the potential interference from information involved in preceding or proceeding tasks.

Maintaining all available options in view, whether in the form of titled sections, tabbed documents or some other data-division method, affords instant movement from one section to another. Furthermore, progressive disclosure does not involve moving to different regions of the user interface; therefore, the user does not have to re-orient themselves every time they go from one set of information to another, as they do with other user interface design patterns, such as 'wizards'. **Progressive disclosure ensures the user only has to attend to a confined region of the user interface, as opposed to the full screen**.

**A further benefit associated with using progressive disclosure is the ability to convey exactly what is important to the user, whether they are a novice or a seasoned pro.**
"Progressive disclosure improves three of usability's 5 components: learnability, efficiency of use and error rate".

### Implementing Progressive Disclosure

When implementing progressive disclosure in your user interface design a lot will depend on the information you are dealing with. For example, some designs usher users through a prescribed order of steps, restricting their proceeding options according to their selections, whilst others simply provide the user with all accessible options at once. **Step-by-step progressive disclosure designs**, such as those

used when filling out online order forms, begin by showing the user a small set of options. Depending on this initial selection the user is shown further options specific to that first choice; constraining the user to only those options relevant to the current task and in accordance with previous choices. In contrast, **all-category-options-at-once designs**, such as the speaker settings panel, allow the user to move between different sub-sections as and when they please. No one design is better than another; however, they must be used in the appropriate setting.

In progressive disclosure designs, where the user is afforded freedom of movement through the options, you must decide which options, tasks or set of data is/are most important to the user. Obviously importance is a subjective measure, but the frequency with which tasks are performed should direct your initial display choices.

It is also essential to make the means of moving from one section to another as explicit as possible. In web design this might involve clearly distinguishing interactive words, such as inline links, from other non-interactive words within the display or using some unambiguously labelled graphical object to skip from section to section (for further visualization methods). One thing is for sure those elements intended to afford clickability must appear to allow such interaction.
For both websites and applications the content labels or links must leave the user in no doubt of where they will be taken and what they will find when they arrive. Therefore, the words used to identify these sub-sections must be unambiguous, familiar to the intended users and consistent with their expectations (i.e. closely linked to the type of words they would use to identify the contents of a link or label).

**Potential Problems**
**One potential pitfall is using progressive disclosure when the user is meant to compare different sets of information**. If data sets are placed in different sections of a panel, window or web page, skipping between these locations interrupts the comparative process and forces the user to rely on what they can recall from the previous, now out of view, section. Short-term memory is very limited, with an approximate capacity of 7 distinct items and an estimated duration of 18 seconds. Having to maintain information in an active state whilst switching between different sections is a difficult task and one that is more often than not likely to lead to forgetting; requiring the user to constantly move back and forth to refresh their memory. Therefore, progressive disclosure must only be used where all of the necessary information for a particular task can be displayed in full view in one section of the user interface and there is no benefit to displaying information from different sections together.

## Two-Panel Selector

There are many occasions where the user needs to see a list of options, categories, commands or other related items but when they make a selection they still need the list in view. The user must be given the facility to review the contents of one of the list items whilst having access to all other items, or at the very least an overview of the list structure, as is the case with email applications where the user can see the message folders on the left-hand side of the screen.

**Two-panel selectors are commonplace in web and application design as they enable users to view the contents of individual files, emails and other content-carrying items in one panel and quickly switch between them in another**.

Whilst users must learn how to interact with the two panel selector in order to see target information and switch between different options**, the pattern is present in many different applications and web designs so they will likely be able to apply existing knowledge to any new instances.** Maintaining the individual items in view, whilst the user delves into one particular item in the list, means they can move seamlessly from one item to another. Positioning the two panels side-by-side or top-and-tail allows the viewer to switch their attention from the contents of one item to other potential selections, without having to carry out any intermediate steps or interactions, as is the case with one-window drilldowns and tabs. This

high level of cohesiveness between the list of options and their corresponding contents reduces the amount of physical 'travelling' required, both in terms of the number of interactions required and the distance the user must span when switching their attention from one panel to another.

Another benefit associated with using two-panel selectors is **the minimal cognitive expenditure required**, as the same visual framework is used for all objects and the same spatial location is preserved regardless of which list item is selected or how many selections the user makes. User interface designs that involve movement from one section, window or page to another usually involve multiple visual styles, formats or layouts; all of which demand more of the user as they must orient themselves every time they reach a new position and find the information relevant to their current task(s).

**Two-panel selectors relieve users of the burden of having to remember where things are as they are never removed from view, or placed in unseen regions of the user interface**. In the same way global navigation bars, maintained across all pages of a website or application, serve as a reminder of where the user is in the system two-panel selectors provide users with a stable visual framework that leaves them in no doubt where they are and how they can achieve their current aims.

**Implementing Two-Panel Selectors**
Users tend to look from left to right and top to bottom (as per the viewing direction followed during reading), therefore, the list - being the user's first port of call - should be placed either above or to the left of the panel in which the selected item contents are displayed. This provides a logical visual framework from which the user can switch from the broad list to specific contents in the anticipated fashion.

To make the interactive and viewing experience as cohesive as possible, the contents of an item should appear immediately in the second panel, following interaction with the corresponding item in the first panel list**. Even brief delays can negatively affect user experience**, so every effort should be made to ensure the contents appear as quickly as possible. List options are generally arranged vertically and there are often times when the number of options exceeds the available space of the panel so scrolling must be enabled. To simplify movement through the options, provide users with the utility to scroll using the directional keys on their keyboard, to promote keyboard-only usability.

The selected option in the first panel must also be distinguished from the rest to provide the user with a means of instantly determining which item contents they are currently viewing. This can be achieved by using a different colour background for the selected option (see image above) or using a clear, unmissable marker, such as a black dot or some other eye-catching shape.

**Potential Problems**
**The appearance of the option list is constrained by the nature of the content and the intended tasks. This means a one-size- or one-design-fits-all approach will not work for all occasions.** For example, email clients employing a two-panel selector to display brief overviews of individual messages whilst other applications simply show category labels or titled sections. Following convention is a reasonably sound policy; if the application or site you are designing performs or affords similar functions to existing sites or applications, use these norms to avoid the mistakes and mis-designs that have gradually been filtered out over time.

## List Inlay

The user needs to view the contents of an item or multiple items within a list, but sending them to a new location, tab, link or window to do so increases the travel time and distance. They need a user interface design pattern that enables them to view the contents of one individual file, email or any other item without having to continually navigate back-and-forth to view the contents of other items.

**A list inlay allows the user to view the contents of all items in a list, from one screen, by simply clicking on the item**. Typically the list shows clear labels or brief descriptions of the contents, and then

**upon clicking an individual item more in-depth details are shown beneath the corresponding label**. The user is then provided with the utility to close the item by clicking on the general region of space occupied by the item contents, the brief list description/label or some designated button/icon, such as a plus/minus symbol.

**Why choose a list inlay?**
You want to provide users with a list of options, files, emails or some other group of items that would consume vast amounts of screen space if their full contents were displayed all at once. Using a small description or content label for each item enables you to show the user all of the available options in a relatively small region of the user interface. They can then access individual item contents by clicking on the brief description or label. Following interaction, the contents are almost invariably displayed beneath the label/description; pushing the proceeding item further down the panel, but not affecting the position of preceding items. This means **the user can usually see a large proportion of other list items even when the contents of one are displayed; affording freedom of movement from one option to another.**

A list inlay **is especially useful when items should be arranged in a particular order** (alphabetical, chronological etc) as the user can see the position of an item within the list whilst reading/analysing its contents.

List inlays can also be combined with other user interface design patterns, to make the user's job of switching from one item to another easier. For example, two-panel selectors work in tandem with list inlays in a number of websites and applications. Where these designs patterns are used together, the first panel includes a list of broad categories, as is the case with email clients where the user is presented with the 'inbox', 'sent mail' and other common categories. Then the second panel houses the individual contents of a broad category, usually accessed by simply clicking on the appropriate category label. The user then delves further by clicking on the individual items in the second panel; revealing the contents just below, as previously mentioned. **The benefit of using the two design patterns in combination is the reduction in the amount of travelling necessary**, both in terms of physical (e.g. movement of the mouse) and graphical distance (movement from the contents of one item back to a broad category). Again, maintaining the list of individual contents and the broad categories in view and on a fixed region of the user interface affords fluid movement.

**How to implement List Inlays**
First, **start by arranging the options into a vertical list**. Items should be arranged into a **clear and logical order**. For example, the list may be ordered according to the date files were last modified - as is the case with some file management applications - or they may simply be arranged alphabetically. Whichever method you adopt, ensure it is both instantly apparent and **the best order for the job**.
When the user has clicked on an individual item the contents should be displayed directly below the selected item and the same gesture should allow the user to close the contents; returning the list back to its original state. The **means by which list items are opened and closed should be explicit**, such as the 'close' label accompanied by a 'X' in the list inlay example above. This **closing control should be near to or in the same position as the opening control**. Some designs allow the user to open items by clicking on the general region occupied by the option label and they can then close it by clicking on the general region occupied by the contents. This has the benefit of reducing the level of precision required to quickly open, close and switch between items.
**Opening an item should cause no change to the rest of the list other than to push the proceeding items further down**, and then they should also return to their starting position when the item is closed. As list items may contain reams and reams of information or large images the list could become very long so there should be **some scrolling facility** allowing the user to gain access to items either further up or down the list whenever they wish.

**Potential Problems**

List inlays by their very nature contain lots of different items, which themselves generally contain vast amounts of information. Therefore, if more than one list item is open at a time, the length of a list could go on and on. This presents the user with the task of scrolling through significant amounts of information in order to find the desired item if more than one item is left open at a time. **One approach is to close an open item automatically**, on the user's behalf, when a further item is opened; thus ensuring the list stays a reasonable length and the user does not have to carry out any further interactions. However, there are occasions where the user will want or need to make comparisons between list items, which is obviously not possible if they can only maintain one open item at a time. Therefore, where you wish to afford comparison allow the user to scroll through the options or combine the list inlay with a two-panel selector, which enables swift movement back to the original list state. Using a two-panel selector also increases the chance all list items will be within view at all times, as the contents of open items are displayed on a separate panel. Therefore, if you wish to promote or allow comparison a combination of user interface design patterns might be best.

## Slideshows

You wish to show users a range of images without calling on them to scroll or flick through them manually. This may be on an eCommerce site, a personal website or any other situation in which there are a number of useful and/or interesting images you want to bring to the user's attention.

**Slideshows are often used to grab the user's attention and tease them with the promise of further images or products.** Slideshows enable a number of images, or products to be displayed within a fixed point on the screen. This helps **to conserve the amount of space consumed in a web page to display as many images as you would like** - although it is recommended that the number of items within the slideshow is not excessive as users may well just navigate away if they become bored of watching a procession of seemingly never-ending images.

### Why choose a slideshow?

Slideshows are used primarily in web design as a way of showing potential customers a range of products they might not ordinarily consider, or alternatively to show them recommendations chosen according to their previous purchasing habits. The automatic, revolving nature of images in slideshows relieves the user of having to progress through the images by interacting with the user interface. Keeping the number of interactions expected of the user to a minimum removes obstacles from them viewing these images or products; increasing the chance they will take the time to assess and consider these options. Some slideshows do, however, require interaction, through clicking arrows at either end of the panel or some other designated symbol or icon. **The benefit of including manual controls rather than implementing automatic movement through the images, is the user is given autonomy** so they can decide when to move forward or back; saving them from missing an image as the items move around. Many designs employ a combination of both automatic and user-controlled movement.

### Implementing Slideshows

A slideshow is a highly visual design element which can immediately grab the user's attention with interesting images or intriguing products. In addition, slideshows should not be used when the content they are linked to cannot be easily represented by some form of imagery.
Slideshows are usually arranged horizontally, with images fitting into uniformly sized and shaped sections. The number of images visible at any one time should be kept to an acceptable number, where it is easy to determine exactly what the image depicts. Usually, the number of images that can be seen at one point is between three and eight, so that users do not have to scan too much screen space and picture quality is not sacrificed. Manual navigation through the slideshow is achieved by clicking left or right arrows, or some other symbol that unambiguously conveys their specific function. Once users have reached the end of the slideshow they are returned back to the beginning.

Slideshows help to reduce screen 'clutter' and help consolidate images or products with a common factor, such as 'DVDs', 'electronic goods' or genres of music. By allowing user-controlled navigation through the

images the potential frustrations of slideshows (too fast or too slow) are avoided, and they can progress at their own desired speed. It is also a **seductive design pattern**, hooking users to view a number of products or items that bare some relation to a present interest, as is the case when slideshows are used to show site recommendations. Therefore, slideshows can help to enhance your web design, especially when the aim is to attract users to make purchases.

**Potential Problems**

The main problem facing the use of slideshows is perhaps something referred to as **banner blindness**, which is a form of selective attention following experience with other web designs where large advertisements are placed at the top of the display and the user learns to ignore such eye-catching elements. Many websites now include some bold advert on the banner and usually they are of no interest to the user, simply included as a means of generating revenue for the website owner. Therefore, over time the **user learns, to some extent unconsciously, that items in this region should be ignored,** especially if they match the appearance of previous advertisements. However, by placing the slideshow within the region of space below the top-level navigation bar (see image above) there is less chance of the user viewing the scrolling images as nuisance advertising. Additionally, experience with eCommerce sites also teaches users that slideshows are a common user interface design pattern, so they are more likely to attend to large banners, especially if they include interesting images or pictures of products they have generally shown some interest in.

A further problem with slideshows is **the speed with which images move,** as previously stated. If the images are links to product pages you will want to make sure the user has ample time to click on the image in the slideshow. Having to chase after an image is annoying and places a hurdle in the user's way when trying to investigate a product further. Therefore, ensure automatic slideshows move not too fast. However, the flip side of this is if they move too slowly, which can also be frustrating. Whilst including manual controls affords freedom of movement at the user's own speed, if there is an automatic function it is best to trial the slideshow on some real users and find out from them what the best speed is.

## Wizard

A wizard is a series of web pages or dialogue boxes that guide users through a sequence of component tasks to the point of overall completion. When the user has satisfied the particular demands of a component they either click a button, such as a right-facing arrow, 'continue' or 'next', or the system automatically takes them to the next step. When they have completed all the steps a declarative message should appear leaving the user in no doubt all phases of the overall task have been fulfilled.

**Why choose a Wizard?**

Wizards are best used in situations where the **task is long, complicated, novel to most users or infrequently performed as they provide a step-by-step instructional walkthrough,** which relieves the user of having to undertake lengthy training or having to commit task-based information to memory. Wizards are especially useful in eCommerce as they help simplify what can often be a long-winded process and remove the responsibility from users of having to remember which elements of a task they have or have not completed. This does, however, remove control from the user but they are unlikely to be resentful if it means they simply have to follow the instructions to complete the task without any mishaps along the way.

**Implementing Wizards**

**The overall task must be broken down into logical chunks**. There is usually an order that best suits the overall task; however, there might be occasions where it is not as apparent. When this is the case, carry out a trial with a sample of users and see which order provides the smoothest transition from step to step and fastest completion time.

As the user interface will only be used to display instructions for one sub-task at a time **you must provide some graphic depicting task progress**, such as a fill bar or a percentage showing how much of the overall task has been completed. **Informative feedback is an essential part of user interface design** as the user must know whether their actions have been successful or not. It also plays a role in reducing the

user's sense of powerlessness over events in the user interface. The inclusion of progress bars have been shown to reduce perceived loading times, reflecting the importance of feedback in minimising the user's sense of how long tasks take. **This feedback should be proportional to the seriousness of the action**, with minor incidents flagged by undisruptive feedback and major system events indicated by eye-grabbing feedback, such as serious error messages. Therefore, if a user has failed to complete a sub-task sufficiently you must draw their attention with some prominent, unmissable graphic or highlighted text, whilst less significant events, such as leaving an input field blank that does not have to be filled, must not be brought to the user's attention.
**The aim of splitting the components of a task into separate sections is to minimise the amount of information a user must contend with at any one time.** Therefore, you must keep the amount of extraneous information and the number of navigation options in the periphery to a minimum.
**The user is travelling through the wizard 'blind';** they are unable to see what lies ahead and what they have already completed. Therefore, **it is essential that the design contains some form of summary**, showing the user where they are within the sequence of tasks and which component has just completed. This might be in the form of breadcrumbs, a user interface design pattern in which a short label for each of the preceding steps is shown in order, or another type of brief summary. Ensure the selected method is not so eye-catching as to be distracting or so small/uninspiring that it is easily missed.
The user may wish to abort the overall task at any given step so it is important that **you implement some means of quitting the wizard**, such as a cancel button, or a 'return to shopping basket' button, for instance.
Whilst it might not always be possible, particularly with complex tasks, you must attempt to **keep the number of steps to a manageable number**. The usual rule-of-thumb is between 3-7 screens; bear in mind user's will want to complete their tasks as quickly as possible, especially when they are perfunctory, such as completing online forms or entering personal details.

**Potential Problems**
As the user is merely expected to respond to a series of instructional prompts, they lose all control over movement through the user interface, other than determining how quickly they progress to the following stage. Due to the constraining nature of Wizards they can be **particularly frustrating for expert users** and those who wish to learn the mechanics of what is actually going on. This annoyance is magnified in software that is meant to foster creativity, such as art, coding or writing packages. Furthermore, **the user is unaware of what their actions have actually done**; simply serving to usher them to the next step. A general loss of control can be extremely aggravating as the user might want to know exactly what has happened and why they are at their current step and how to complete the stages appropriately. Being taken to the next step might seem sufficient to inform the user they have satisfied all elements of the previous step, but to many users, especially those new to a website or application, they might think they have missed something important. Therefore, it is **essential to implement wizards where each sub-task logically fits into a multi-step instructional wizard and the intended users would not benefit from, or want, freedom of movement through the component stages or more progress information**.

Frequently within wizards**, users are unable to go back through the steps**; sometimes leaving the user unsure of whether they have completed the necessary input fields, for instance. This is at odds with various user interface design guidelines which call for websites and applications to **'Permit easy reversal of actions'**. Many users are anxious about using systems because they fear causing an irreversible problem; avoid this anxiety by allowing them to backtrack and make corrections. This should be permitted at various points along a goal path whether after a single action, a data entry or a whole sequence of actions. As 'Wizards' are often used in eCommerce, when the customer is filling out order details and entering personal information, easy reversal is frequently prohibited, with back movement leading to the loss of previously entered information. This is a security measure to prevent details being stored when the user has aborted a task. Therefore, there are times when guidelines must be flexed to suit the current design, **especially when there are major security concerns associated 'permitting reversal'**.

However, your design could include reminders, prompts and indicators to help reinforce that all elements of a sub-task have been completed and the user can proceed. You could also include a small map of the component stages along the top of the display, showing the user which stages have been completed appropriately and the steps that lay ahead.

**As control is removed from the user you must make every effort to inform them at each stage that the preceding step has been completed** to the requisite level, and there are no important blank fields. **If backward steps would not cause any problems, provide the user with this capacity**; it will at least give them the peace of mind that they can check for themselves no element has been overlooked or completed incorrectly.

Therefore, wizards can be an excellent time-saving addition to the user interface, especially with long-winded tasks that involve a series of steps that must be completed in a prescribed order. However, wizards are not always appropriate, as they remove control from the user, preventing them from learning exactly what is going on and they can stifle the creative process. They must not be employed when the user would benefit from the capacity to move through the sub-tasks in their own order, or on occasions where the user needs to see the contents of each section together, for comparison purposes, or to cross-check input information.

## Archive List

You have a long list of items that represent events over a period of time which the user will need to see in chronological order. However, displaying all of these items at once would complicate the task of identifying relevant items, as the user would then have to scan hundreds, if not thousands, of possible selections. Therefore, the user needs a list of broad categories, arranged chronologically, so they can drill down to more specific content.

**Archive lists satisfy these demands by consolidating data sets into broad categories, listed by the day, week, month or year in which the contents were uploaded, edited or took place.**
Archive lists typically involve a series of dated sections, each of which contains a column of categories, labelled according to their contents, such as 'links', 'images' or, in some cases, personalised titles. The user then clicks on one of these labelled categories which opens the category contents in a new page. Depending on the nature of the information, files or items in the list the user might be shown more sub-categories which, when selected, lead to another page, unless another user interface design pattern is implemented, such as a dropdown menu or two-panel selector, in which case the user will be able to scan the choices from one page.

### Why Choose an Archive List?

Archive lists help guide the user to target information with the use of labelled categories which keep the number of options on the screen at any one time to an acceptable or manageable level.
Channelling the user in this way **saves them from having to scan large numbers of items to identify their desired information.**

### Implementing Archive Lists

- decide whether to arrange the archive list in days, weeks, months or years.
- decide which display method to use when a category is selected. Most designs take the user to a new page where the contents are further broken down into smaller categories or more lists; however, when the broad categories are split into further categories you could use a list inlay, where the sub-lists appear underneath the broad category label when clicked. The user can then drill down into more specific options from one page; saving them from having to locate the necessary information on a new page or window.

- ensure the category labels appear 'clickable' so the user knows which elements can and cannot be interacted with. Additionally, it can be useful to change the colour of category label text when the user has successfully made their selection. 2 major benefits: firstly, the user knows their click has hit the target and secondly, by changing the colour of previously selected options the user is aware of categories they have already checked when searching for an item; saving them time and effort.

Some dates are more important to a user than others. It is difficult to reflect these personal differences in a rigid user interface design. Therefore, in websites and applications where the user would benefit from making certain dates more prominent you should provide them with the utility to personalise the display.

**Potential Problems**
As previously stated, finding the appropriate division into days, weeks, months or years can have a significant impact on the user's ability to extract target items quickly. The reverse holds true when dealing with a small number of items, as a large number of categories needlessly increases the points a user must scan when searching for an item.

If the category labels are ambiguous, inappropriate or unfamiliar to the user, they will be uncertain of where they will be taken when they select an item. Poor labels also fail to reveal the contents of a category; leaving the user unsure of where specific items are located. Therefore, item labels should contain language familiar to the intended users and commonly associated with their purpose.

## Navigation Tabs

When there are a number of clearly defined sections, which do not suffer as a result of being placed in separate regions of the user interface, **navigation tabs** meet the users' needs. Navigation tabs are almost exclusively placed along the top-level navigation bar of websites; offering small clickable labels that open content in a new webpage. This is in contrast to **Module Tabs** which present information on the same page, usually within a fixed panel, allowing the user to operate without having to navigate to other separate sections of the user interface.
Navigation tabs act as the highest order sections or categories; they are purposefully broad so the user can gradually channel into the user interface to access more specific content without having to scan all of the available contents a site has to offer at once.

**Why choose Navigation Tabs?**
Perhaps the most compelling argument for using navigation tabs is simply that they are so common in user interface design and, specifically, web design. This means even **novice users will likely have come across the pattern and know how to use them.**
A further benefit to using navigation tabs is they **help keep the user interface free of clutter**. Each tab navigates the user to a separate web page, which can contain an enormous amount of information; they may also contain links or other user interface design patterns that take the user to further content. Therefore, if all the contents from these distinct sections were combined on one page the user interface would be flooded and the user would face the unenviable task of scanning the gargantuan display to try and identify specific items or elements of interest to them. Therefore, whilst tabs provide a satisfying means of navigating to, from and between different sections they also **help establish order within the user interface.**

As tabs are organised together within the same region of space the user can hop from one section to another with the click of a button. Affording movement through the different pages like this means the user does not have to perform multiple interactions when they have finished on a particular page, or if they simply want to investigate the contents of another tab. Reducing the amount of effort the user must expend to complete their tasks saves time, improves usability and it can promote exploration. When the user has to carry out a number of different steps to reach content it contributes to a poor 'Return on Click Investment'

(ROCI), which refers to the amount of effort expended versus the pay-off when they arrive at their intended location. The ideal ROCI is achieved where the user has to carry out the fewest number of interactions possible, whilst the new location provides them with all of the information they seek in an easy to digest fashion.

Implemented appropriately, navigation tabs provide **an ever-present means of changing location.** Consistent design allows the user to operate in the same fashion regardless of the page they are on. No matter where the user is, they can navigate to another broad category by simply clicking on another tab. The maintenance of navigation tabs in the top-level of the user interface can also help orient the user and provide them with important contextual information. If the selected tab is highlighted/distinguished from unselected options the user knows where they are, where they can go and how they can get there.

**Implementing Navigation Tabs**
Establish how the site contents will be divided, which involves clubbing related information, images and other items together under the same umbrella term. As a general rule navigation tabs should only be used when you are dealing with at least three and no more than nine or ten distinct sections.

After you have established which groups all the available contents fit into you must assign a logical and unambiguous tab label to each category. All of these tabs must then be arranged horizontally along the top-level navigation bar, abutting one another to reduce the amount of space the user must traverse when skipping from tab to tab and to ensure there is enough room to fit all of the necessary information onto each tab. **The usual convention is to make sure the tab bar spans the width of the screen**, as this helps establish the first level in the visual framework - with the navigation tabs serving as the top tier and providing the broadest channels to explore.

In order to help the user identify where they are within the website the currently selected tab should be distinguished from the rest. Providing context is an important element of user interface design as the user might not remember where they are, where they have already been or where they can go (in general or from their current position).

Ensure **the same design is maintained across all the different pages to allow free and easy movement around the contents of the site and to prevent errors**. Users are more likely to make selection mistakes if the tabs are constantly changing place. Furthermore, inconsistent design forces the user to consciously engage with the visual display rather than rely on experience to help guide their interactions.

**Potential Problems**
When dealing with more than ten tabs the top-level navigation bar is stretched to its limits and the visibility and clarity of tab labels starts to suffer. Whilst experienced users will learn where their favoured tabs are located, the time new or novice users take to identify their desired tab will increase proportionally as the number of alternatives increases.
As you are grouping contents together it should become apparent exactly what you should call each section. **Tab labels should be short and sweet**; it is recommended that they should not exceed two words and they must convey to the user exactly where they will be taken when clicked.
Further to this, **you must never abbreviate tab labels** as the user then has to either rely on what they can deduce from the visible portion or remember where certain tabs are positioned along the navigation bar.

**There are a number of occasions where you should not use navigation tabs:**
One such occasion is when the user would benefit from seeing the contents of multiple sections on the same page. For example, a user might need to compare different data sets to derive an understanding of the overall thrust of a topic or they may want to compare the prices offered by different sellers on an ecommerce site. By placing such information on separate pages the user is required to switch back and forth, increasing the amount of time and effort involved in making comparisons. Additionally, this forces the

user to remember information from at least one of the sections as they will never be able to have the two groups of content side-by-side.
Navigation tabs are used as broad categories, so they should not be employed when dealing with data that is content-specific, such as newsfeeds or dated articles. On these occasions archive lists are probably more likely to satisfy your needs and the users'.
If there is only a small amount of information in a number of tabs there might be a user interface design pattern better suited to the content of your site. Tabs should be used to consolidate distinct groups of information, but if each one is only a few lines of text long or contains one or two images the user might benefit from having them available all on one page in some logical order.

## Module Tabs

Module tabs provide a simple solution, especially when **dealing with a limited amount of space and you want to avoid continual page refreshes**. Almost exclusively placed at the top of a page or panel, module tabs are arranged in a horizontal line, each containing one or two content-defining words, and positioned immediately above the panel or space where contents will be displayed. When the user places the cursor over a tab or clicks it - depending on the design - the contents of that particular tab are displayed below. **In contrast to navigation tabs, the user operates from one web page**, with the contents of each tab displayed within a fixed space or in the form of a dropdown menu. The user can then move the cursor down and select the new options, which will generally take them to a new page, or simply read the contents of the tab. Moving the cursor away from the tab or content panel then causes the dropdown menu or list of contents to disappear. The user can then repeat the process by moving the cursor back over the module tabs.

**Why choose Module Tabs?**
Module tabs help organise the user interface; restricting the amount of information on the screen at any one time. **As the content of module tabs are displayed within a fixed and generally small area of the display the user does not have to scan large amounts of screen space in order to find items and/or options of interest.**
Unlike navigation tabs the user can operate from one webpage, saving them from moving between different sections of a website. The content of module tabs can be accessed immediately and usually without the need for any interaction/clicking.
The user can skip between the contents of different tabs by simply moving the cursor a little distance to the left or right.
You want to keep the user interface free of clutter without sacrificing the user's ability to move freely through the distinct groups of information, links, options etc contained within the tabs.
Module tabs provide valuable contextual information; informing the user of where they are, where they can go and what they can see.

**Implementing Module Tabs**
Organise content so the appropriate information, links, images etc are grouped together under the correct (i.e. most logical) category. These categories then form the basis of your tabs.
Now you must choose a fitting tab label that defines the associated content and serves to inform the user of exactly what they will find when they select a specific tab.
Arrange these tabs in a horizontal line, ensuring they are neither too long for the available space nor so short that the user struggles to see or make sense of the content-defining term.
Ensure there is vacant or data-poor space (i.e. an area where there is very little useful information) directly beneath the row of tabs so the contents can be displayed without obscuring key regions of the user interface.
Now decide whether to use the automatic display method (where tab contents are shown when the user simply hovers the cursor over a tab) or the manual display method (where users must click on tabs to see their contents). The benefit of using the former method is the user does not have to interact with the user interface; saving them the effort of clicking on a tab means they can move from one to another by simply

moving the cursor along the row. This does, however, increase the error rate as the user has to be mindful of where they move the cursor; moving it a little too much to the left or right can lead to a frustrating switch between tabs.
Allow users to move through the tabs by clicking the shift/tab key; this might mean the user has to skip through items (e.g. moving from tab one to tab nine would require eight mouse clicks) but it requires less precision than moving the cursor to the correct tab with the mouse.

**Module tabs should not:**
Open content in a new page or in different points of the user interface
Cause any other change to the user interface when selected
Lead to a whole page refresh
Be stacked on top of one another; if you have exceeded the width of the tab bar then an alternative user interface design pattern, that can contend with a large number of categories, might be best. Alternatively, your tab labels might be too long, in which case go back to the drawing board and come up with some content-defining terms that are more concise.
Be used when the user would gain some benefit from seeing the contents of different tabs together at the same time. For example, where the user needs to compare and contrast different data sets, images or other items.
House content that would be just as effective displayed on its own.

**Potential Problems**
A common complaint with navigation tabs, when using the automatic display method, is their sensitivity to cursor movements. Although the user can move from one tab to another faster than they would if they had to click the tabs, using the cursor location to open and close tabs can be a nuisance. If the user wants to see content in one tab they must make sure the cursor is not positioned over another tab, otherwise undesired content will be displayed. Therefore, whilst the user does not have to click on a tab to view its contents, automatic content display can be frustrating as they must ensure the cursor is not flicked over or left on another tab. Furthermore, if the tab contents are revealed in a dropdown menu they can obscure the user's view of content on the rest of the page, so they must be mindful not to move the cursor over the module tabs when attending to content on the rest of the display. One way around this problem is to introduce a slight delay between moving the cursor away from a tab and removal of the dropdown menu or content panel. This means if the user moves the cursor around the screen it is unlikely to cause the contents of a tab to close unintentionally, unless it is left outside the region of space occupied by the menu or panel for a prescribed length of time (typically 200-300 milliseconds). An alternative method of preventing the unintentional closure of a tab is to make the menu or panel wider than the contents require. This increases the user's margin for error when moving the cursor over the dropdown menu or panel. The final method is to alter the image of the cursor when positioned over a tab (e.g switching from an arrow to a pointing hand); this helps to inform the user that they are within an interactive area of the user interface and moving away from this region will cause the removal of associated content.

## Vertical Dropdown Menus: Organising Content for the Best Results

You have a number of different categories of options, but displaying them all at once would clutter the user interface and impede the user's ability to quickly identify and make their selections.
**A vertical dropdown menu allows you to display a list of contents, navigation points and functions without flooding the user with lots of options all at once.** A solitary dropdown menu can be used when you are dealing with a single group of options. This is in contrast to tabs, which must only be used for multiple (i.e. two or more) groups of options, data sets, categories and most other groups of related content. Like tabs, dropdown menus are employed when dealing with a maximum of ten different sections; although this figure might increase or decrease depending on the length of your menu description labels and the width of the panel or screen on which the dropdown menus are arranged.

Options are presented to the user in the form of a vertical list when the menu label is clicked or the user hovers the cursor over it - depending on which interaction has been enabled in the design. The user can

then select one of the dropdown menu options by clicking on the appropriate content name. These options might take the user to another page or they might lead to the presentation of a new menu or panel. Unlike module tabs the content of these dropdown menu items is not presented in a fixed location or on one specific panel. Instead the new location is determined by the nature of the content, such as a dialog box for help and documentation or a new page for a menu list item on a website. This means dropdown menus can be used to display lists of options for a more flexible range of content, unlike module tabs which constrain the user to a limited and fixed space on the user interface.
In dropdown menus each list of options is assigned a content-defining title and these are placed horizontally along the top of the screen or panel. As the name suggests, the list of options drops down when the user places the cursor over or clicks a menu label. They can see all of the contents in that specific menu ensuring they only ever have to attend to one list of options at a time. The user can then make their selection from the vertical list of options by clicking on the appropriate point on the menu. Typically, dropdown menus are designed so the list item under the cursor is highlighted; serving to inform the user which option will be selected.

**Why choose Vertical Dropdown Menus?**
As they are traditionally used in desktop applications, located at the very top of the screen, **nearly all users, no matter what their level of experience, will be familiar with this means of organising content** and how they must interact in order to see the available options and make their selections.

Consolidating the options in this way also saves the user from having to scan large numbers of options across a sizeable amount of the screen, in order to identify their desired item/option.

A list of dropdown menus can act as the first tier in the visual framework of a user interface. Placed at the highest point of the screen or panel and offering the user the broadest categories of options, they can instantly see what they can do and how they can drill down into more specific contents or options.

**Implementing Vertical Dropdown Menus**
Identify what your categories or dropdown menu titles are, which involves reviewing your contents to establish common themes and links between items, options, functions and site/application contents. As previously stated, (generally) the number of sets of options should be no greater than ten, due to the arrangement of menu titles side-by-side and spanning the width of the user interface. This restricts the amount of space available for presenting the menu titles and as, by their very nature, they dropdown you cannot stack them one on top of the other.
After arranging these options into their appropriate groups you should assign a logical and fitting title to each category. This label must not be too long - so as to conserve the available space along the panel or screen - nor so short that the clarity of the content-defining term is sacrificed.
Now place the menu titles in a row.
As the cursor will begin on the selected category - which is always at the top of the list - **you should order the options from most frequently selected (at the very top) to least frequently selected** (at the very bottom). Obviously you will not know for certain which options will be the most popular before trialling your design, so you should try to see things from the user's point of view, or conduct some usability testing and measure the frequency with which items are chosen.
Now you have decided what will be in the menus and the order items will be displayed you must add some superficial features that help the user choose the intended option. There are two main features that assist the user: firstly, the use of background colour to inform them which option will be selected according to the position of the cursor, and secondly, the implementation of a clear boundary around the dropdown menu to help the user isolate the group of available options from the rest of the user interface. Without some visual indicator the user might assume the cursor is over their desired option, when in fact it is over a neighbouring option.
Finally, when the user clicks on an option the menu should automatically disappear and the associated content should open immediately.

**Potential Problems**
Automatically removing the menu from the screen when the cursor is moved to another region of the user interface can be infuriating, but there are a number of possible ways around this problem. Firstly, you could implement a time lag of approximately 200-300 milliseconds between the user moving the cursor from the space occupied by the list of options and the removal of the menu. Secondly, you could make the menu larger than the options necessitate, so unintentional or inaccurate movements are less likely to cause the menu to disappear. Thirdly, **alter the image of the cursor when positioned over a tab** (e.g switching from an arrow to a pointing hand); this helps to inform the user that they are within an interactive area of the user interface and moving away from this region will cause the removal of associated content. The first two approaches allow a larger margin for error when the user is moving the cursor around the screen and the last provides the user with important feedback. Therefore, the three techniques combined can help reduce the potential frustrations associated with the automatic removal of a dropdown menu when the cursor is moved away.

**Do not use dropdown menus when:**
You wish to show the user where they are within the website or application. When it is important to provide the user with such contextual information use navigation tabs.
The user would benefit from seeing the content of the individual menus together.
When a category contains only one item.

## Horizontal Dropdown Menus

Horizontal dropdown menus help save the user from having to go to separate sections of the user interface at each stage of an option selection sequence. As you can see from the example above, when the user places the cursor over an option in the first dropdown menu, with an east-facing arrow to the right of the item, a further dropdown menu appears alongside the original, which contains more specific selections.

**Why choose Horizontal Dropdown Menus?**
The reasons for choosing horizontal dropdown menus are largely the same as those for Vertical Dropdown Menus. However, **using a horizontal dropdown menu allows the user to view a large number of options from one page, that are progressively more specific, whilst helping keep the number of page refreshes to a minimum.** By displaying all of the sub-categories of options within the same general space occupied by the highest order of categories, the user simply has to switch their view a little to the right every time a horizontal menu is selected. If the user were taken to another page to see these sub-categories they would have to re-orient themselves at each new page refresh, which adds time to the process.

**Implementing Horizontal Dropdown Menus**
Again, the stages involved in the implementation of horizontal dropdown menus are much the same as those for Vertical Dropdown Menus. **The one major difference is the inclusion of some visual indicator, like the east-facing, black arrows in the example above, informing the user that hovering the cursor over an option with extra options will reveal an associated sub-category in a further dropdown menu**. This new menu then appears to the right, with the first possible selection now in line with the corresponding item in the original dropdown menu.

**Potential Problems**
The same problems affect the usability of Vertical Dropdown Menus as do horizontal dropdown menus. However, one problem is more of an issue in the latter, as a larger region of the user interface is occluded due to the sprawling nature of the open menus. As you can see in the example above, the two open menus conceal a large amount of the user interface and some designs even contain a third or even a fourth set of sub-categories (on rare occasions). Therefore, it is important to use the space in which dropdown menus appear for filler content or to simply leave it vacant.

# Affording Fluid Navigation

## Global Navigation

Global navigation is a region of the graphical user interface reserved for buttons, links, search bars or any other design element affording movement from one set of content to another. This region is maintained across all different pages to provide a consistent means of travelling to anywhere in the application or website. This area may include titled sections, dropdown menus, buttons, module and/or navigation tabs, search bars and homepage links.

Maintaining a fixed set of navigation points in one region at all times allows the user to switch from one page or list of contents to another with ease, and saves them having to learn the whereabouts of such information for every new page, window or panel. This pattern is also frequently used in web design, which means the majority of users will be familiar with and come to expect this means of navigation.

A bar along the top of the window/panel/page containing the broadest categories provides users with the facility to drill-down into more specific content, or revert back to more general information, through the simplest interaction; a single click. The top of the screen is now, after a number of decades of GUI design, well-established as the springboard from which users can dive into further content and they now come to expect links, buttons or category titles that enable them to navigate a site or application to be positioned here. Arranging content or category labels along the top helps give the user an overview of the user interface and informs them of where they can go and how they can get there; affording free movement and promoting exploration.

### Implementing Global Navigation

The key to global navigation is consistency, both in terms of appearance and location. Inconsistent design interrupts the application of knowledge from one setting to another; calling on the user to learn new representations for the same information or function across separate instances. The user should only have to encode and learn one set of representations and one overall spatial location (i.e. at the top of every page, panel or window). Whilst consistency is essential, it is not the whole answer to effective global navigation. If the design is ugly or unhelpful, it does not matter how or when you apply it; its negative effects are likely to endure.

## Top-Level Navigation

You wish to present the user with a number of groups of content that fall into distinct categories, but displaying everything on one page or window would place great strain on the user. Using one page or window would also give the wrong impression that information lower down the page/window is less important than content nearer the top. Therefore, the user requires that the content is divided into separate regions of the user interface and they are provided with some means of navigating between different groups of content.

### The Solution

Top-level/Global Navigation is a common solution to the above problem, allowing the user to switch between sections of the user interface regardless of their position. The navigation bar will include broad category titles that navigate the user directly to the associated content when clicked. The user is able to drill-down into more specific content within the contents of these categories but they provide an ever present means of switching between the broadest categories. Top-level navigation bars are essentially the highest point in the architecture of the user interface; enabling the user to dive into a broad group of contents placed together on the basis of some shared relationship. These categories are assigned short labels that encapsulate the meaning of the group of contents (e.g. 'Sports' for football, rugby, cricket, hockey etc content). The category labels serve as links to the contents and they are arranged along the top of the window, page or panel in a row. Whichever section of the user interface the user is on, the top-level

navigation bar is located in the same position, ensuring they are able to quickly move from one region to another without having to make any backward steps.

### Why choose top-level navigation?

well known patter, can accommodate as much as 12 categories without using too much of the user interface, provide navigation from anywhere in the interface.

### How to Implement Top-Level Navigation

Establish your groups of content; if the number of categories exceeds twelve then you might be best with another user interface design pattern, such as a mega dropdown menu.
Assign each of these groups a fitting title (e.g. from the top-level navigation bar above: 'Home', 'Search Full Text', 'Site List', and 'Explore Data').
Link each title to the appropriate contents, so when clicked the user is navigated directly to the associated information.
Arrange these category titles in a logical order. For each group of categories there will likely be an appropriate order, but the first option is usually the homepage, as this represents the first tier in the whole user interface.
When used in combination with tabs, top-level navigation bars serve as the second tier of the user interface. In these instances the first tab should be used to take the user back to the homepage.
Implement the same top-level navigation bar in all other regions of the user interface.
Inform the user of their current position by changing the appearance of the individual category label. In the example, above the background colour of the selected category label changes from light blue to white, whilst all other options remain the same colour.

### Potential Problems

Whilst top-level navigation bars are common and most users are aware of how to interact to move from category-to-category, unless they are visible and apparent in each section, the user could overlook this design pattern. For example, when users arrive at websites they seek strongly scented visual clues to direct their attention to the most personally relevant information. This strategy, referred to as **'satisficing',** means the user could overlook the top-level navigation bar if it is poorly designed or buried among a mass of other user interface design patterns and content. **Satisficing is a cognitive heuristic (rule-of-thumb**) where the user scans the available options until an acceptability threshold is reached, rather than using a directed, logical approach. Therefore, whilst we might assume that the user will, first of all, look for a broad overview of their options, instead their approach is heavily based on the strength of the scent of information on the first page, panel or window.

Potential overlap between categories represents another threat to the usability of top-level navigation bars. By this we mean that when two or more categories appear to represent the same content, the user is left to use trial-and-error in order to determine the exact location of a particular source of information. For example, whilst they appear in dropdown menu, Amazon presents the user with 'Music,Games, Movies and TV', which the user might assume contains computer games; however, the category that actually contains such products is listed under 'electronics and computers'. This labelling issue creates ambiguity and slows the user down when they want to hop to a specific group of products.

### Progressive Disclosure

Users want different things, but presenting all of the available features, actions, commands and options at once slows them down and over-complicates the user interface. Additionally, users need the display to be simple, as they do not have the time to learn which options are available and where they are within a mass of different elements on the screen. For this reason, the user needs a pattern that can consolidate the options into easily accessed different regions or sections within the user interface, so they can promptly make their selections without having to consider a range of other options that are not optimal for their needs.

## The Solution

Information is presented gradually to the user when they actively seek it, rather than displayed all at once. When using progressive disclosure the user is shown a small set of important and broad options, then upon selecting one of these a further set of more specific options are presented. Like this the user is in control of the user interface and the screen is restricted to only options with some connection to the user's current needs. Progressive disclosure helps to reduce the complexity of the user interface so the user is never overloaded with options; making it easier for them to determine how to progress through their tasks.

## Why choose progressive disclosure?

The major benefit of using progressive disclosure in design is that at every stage the user knows the screen contents are always important and relevant to their aims and objectives. They determine where they go and what options are on the screen at any one time, by selecting a particularly category. Furthermore, this pattern serves novice users just as well as the more experienced because they only ever have to consider options that bear relation to their present intentions. Take the accordion menu above, for example, if the user wishes to narrow their search they can select a particular artist and review the specific songs and albums available. By placing information into separately accessed groups, the user can conserve their attention as they never have to consider options they do not need or have no relation to their needs.

The same benefit is experienced by the advanced users as they only have to consider a small set of options when they first arrive at the user interface, rather than being forced to scan through a long list of various features and other graphical elements that they rarely make use of.

Progressive disclosure is, perhaps, even more important now computer devices are smaller, as the designer and user are dealing with a much more restricted amount of screen space, making the selection process harder and harder if there are lots of different option displayed at once. Jakob Nielsen, author, researcher and consultant on user interfaces at the Nielsen/Norman Group, states that as websites have become more and more complex - affording a wide range of different functions and providing an array of information and products - the need for organisation is of ever greater importance to the user experience. Ecommerce sites have, largely, embraced the notion of progressive disclosure; listing a small amount of product information at first, then when the user wants it, they can access more detailed product specification details by clicking on some designated region of the page (usually the image of the product itself).

## Implementing Progressive Disclosure

The implementation of progressive disclosure is not specific to any one user interface design pattern; it is simply a means of reducing the amount of information presented to the user at each stage of their tasks. Therefore, the nature of each design problem will determine the nature of progressive disclosure; however, the basic steps will largely remain the same. For example:

- Establish the highest order options or categories
- Place the contents of each of these different options/categories into a secondary region of the user interface
- Provide the user with some means of accessing this information promptly, such as simply clicking the category label
- At each stage present the user with more and more specific information, ensuring to keep the number of these stages to a reasonable amount (e.g. between two and four)
- Provide the user with efficient means of returning back to the previous stage and broader set of options

## Potential Problems

There are three significant areas for limiting the usability of a design involving progressive disclosure:

- Dividing the options and/or features into two distinct regions means the most important information must be presented first. The user should never be able to miss things relating to their current goal(s) because they are hidden in the secondary region of the user interface. Nor should they have to move to the second tier every time they visit a site or use an application.
- The initial list of options must not be too big; otherwise, this would defeat the object of using progressive disclosure as the user would be unable to focus their attention on only information bearing strong relation to their current needs. The option labels must also be clear and unambiguous so the user is confident that clicking a particular option will navigate them to their desired content.
- The means of moving through the different tiers of a design must be obvious and conform to the guidelines for visualising links. The information scent of the initial set of options must also be strong so the user able to accurately predict where they will be taken, and what they will find, when they select each individual option.

In order to determine which options the user needs in the initial display you may need to conduct a usability test, task analysis of field study. In usability testing a small sample of intended users interact with a prototype design and the results of their experiences are used to separate primary from secondary features/options. Task analysis involves measuring the frequency-of-use when these sample users are using a design so that the most commonly selected options are given a correspondingly high and visible position within the user interface. Field studies are similar to usability tests but they are carried out in the environment where the design will be used when completed.

Finally, whilst progressive disclosure can accommodate a large number of different tiers, once it exceeds two or three separate regions or sets of options the user might forget where certain features and options are. Therefore, the progressive disclosure design must be simple too.

**Mega Dropdown Menus:**

When you are dealing with a website, application or system that contains a large number of different subpages or sections, you will need to provide the user with some means of navigating across all of these sources of content. However, a simple dropdown menu would not be sufficient to contain all of the sub-levels within the site or system and you want to allow the user to consider all of the potential areas for exploration from one user interface design pattern.
Mega dropdown menus (also referred to as **'fat menus'**) satisfy these requirements by consolidating all of the different sections of the site, application or system into one long list, which is further broken down into sub-categories and accessed from a navigation bar, such as the Asos homepage example above.

**Tabbed Document Interface**

You are dealing with a multi-level site, system, application, dialog box, panel or window and you want to allow users to navigate to these various levels from one point on the user interface. Each level contains different sets of contents that need to be grouped together a) so the user knows they are connected b) to prevent the potential confusion from having unconnected options arranged together c) so they can compare their options before making a selection and d) so the user can go straight to a specific set of options according to their current aims and objectives.
A common solution to this design problem is to use a tabbed document interface, also referred to as a 'card stack pattern'. In this pattern, labelled 'cards' or tabs help to structure content into separate sets of links.

**Breadcrumbs**

When the user is moving through a multi-level website there are often occasions where they would benefit from being able to retrace their steps. Therefore, the user needs a user interface design pattern that provides direct links to return them to the various levels of the site they had previously visited.

'Breadcrumbs' satisfy the user's needs in this respect, as each visited level in the site hierarchy is represented with a link, allowing the user to instantly jump to a particular page of contents or options. The user interface design pattern 'breadcrumbs' is so-called as it works in a similar way to the path-marking method employed by Hansel and Gretel, a German fairytale, of laying bread crumbs along their route through the woods, to ensure they would be able to find their way home. By listing the different levels of the site hierarchy visited by the user, they can trace their steps and navigate directly to any one of these levels by sampling clicking one of the 'breadcrumb' links. The following breadcrumbs are shown in the example above: 'Home' > 'Help and advice' > 'Help and advice home' > 'Living rooms' > 'Create the look for wall hung fires'. The first breadcrumb represents the highest level within the site architecture, whilst the last breadcrumb represents the user's current position and all of the links in between, are the various stages of the user's journey, in order, through the site.

## Scrolling Lists

You want to provide the user with a long list of options, whilst ensuring they do not occupy too much of the screen. If all of the available options were presented at once they might encroach on other important regions of the window, page or panel. Therefore, the user needs a user interface design pattern that condenses the options into a smaller region of space than they would consume if all were displayed in full view. This pattern must also accommodate all of the options without sacrificing the clarity of each option label and without adding too many extra steps to the process of accessing each individual option.

A scrolling list presents a satisfying solution to this problem, as all options can be made available on the page, window or panel, with items further down the list accessed by scrolling. Scrolling lists are often used in user interface designs where there are a number of options that open in a separate section of the dialog box, window, page, or in another panel. Typically, scrolling lists consist of a column of options, arranged in alphabetical order, or some other logical/meaningful sequence. The user can move back-and-forth through these options by clicking the south-facing arrow (to go down) or the north-facing arrow to return back to previous options. These arrows are placed at the top and bottom of a scrolling bar positioned either on the right- or left-hand side of the scrolling list. By providing the user with the scroll bar they can then access options that are out of view at first, by clicking the south-facing arrow. Alternatively, if one of the list items has already been clicked, the user can move through the options with the directional (up and down) keys on their computer keyboard.

## Search Boxes

The user needs to find the answer to a query, locate a particular word or item within a large body of text, locate a file within a system, or jump to a particular option within a list. Manually scanning through the available options or sections of a website, system application or a large database to find an answer, word, item, file, option or any other element would take a long time, and the user might not even be able to find what they are looking for. Therefore, the user needs something that will cut out the manual search process. A search box or search field is a common graphical user interface design element, which allows the user to enter letters, words, terms either in a web search engine, database, website, archive or a list options to return content or options directly related to their input. The search box typically appears somewhere along the top of a search engine, website, application or database, to ensure it is visible within the overall design. It is important to include the search field in a prominent position, as the user will often treat this as their first port of call when visiting a website or any other program that contains a large number of sections with contents buried deep within the architecture.

## Homepage Link

When users navigate to various sections and pages within a website they need some means of returning back to their starting position. There are many occasions when site visitors simply want to leave a specific page of contents to consider the broadest array of options and navigation points offered on the homepage. Most websites now hyperlink their logo or company, business or group name so users can immediately switch back to the homepage. The website name is usually located in the top left-hand corner of every

page, providing the ever present facility to go from a narrow area of interest to the broadest group of options.

1) The hyperlinked logo or name must appear on every page within the website. Maintaining consistent design is essential to the user

2) Homepage links are usually an implicit navigation point, with no label expressly stating that clicking on the logo or name will divert the user back to the homepage. But Visitors know that from experience so they will automatically look to the top-left logo or name when they want to jump back to the homepage. For this reason, it is important to conform to the established method of placing the homepage link in the top-left corner and hyperlinked to the company/business/site logo or name.

3) When your hyperlinked logo or name does not expressly state that it also operates as a homepage link, you should include a number of other small touches to help the user identify this functionality.

4) Make sure the whole region of space occupied by the name or logo is hyperlinked. When only a portion of the element is 'clickable' a higher level of precision is required

## Sitemap Footers

When users reach the end of a webpage, if there is no means of navigation in the footer they will have to return to the top of the page.
Include a sitemap in the footer of every webpage, so when the user reaches the end of the contents there is a highly visible and instantly accessible navigation tool for moving freely around the website, or at the very least to jump to another level within the site hierarchy.

## Pagination

You are dealing with a large set of ordered data that is difficult to display on one page, but the user may need to view specific items from this set.
Pagination is the process of splitting the contents of a website, or a section of contents from a website, into discrete pages. This user interface design pattern is used so site visitors are not overwhelmed by a mass of data on one page, which would also require lengthy scrolling and scanning in order to identify target items.

Many users like to see all options on a single page, rather than clicking from page to page looking at products.  View All option is helpful to some users. More important, the View All option doesn't bother users who don’t use it.

Many sites let users choose how many items they’ll see on each page. This is often overkill, as when pop-up menus let users View 10, 20, 30, 40 items per screen. It’s usually better to offer a single default number — such as 10 or 20 — and supplement it with View All for people who want more. Instead of a pop-up menu, this design requires only a simple button and is thus much faster to operate.

## Tag Clouds

Website users want to navigate to certain pages or groups of content that are particularly popular with other visitors. Therefore, site visitors need a user interface design pattern that represents user behaviour, in terms popular pages and search terms, and provides them with the means to quickly navigate to these pages and contents.
A tag cloud is a visual representation of text data, depicting keyword metadata from a website as a means of visualising free form text from all of the various pages to allow users to identify popular and/or prominent terms. The key words and terms, within the tag cloud, are hyperlinked, so the user can click on one of these tags and jump directly to the contents associated with the word or term. The relative importance or prominence of a keyword on the website is represented with font size or colour. Therefore, as terms and words increase in importance or prominence they are larger within the cloud than other less important or

prominent tags. Likewise, when using colour the most important/prominent tags are represented with bolder and more vibrant colours than less important/prominent keywords.
Tag clouds have all but disappeared from web designs in the last five years.

### Inline Linking

You want to allow users to investigate keywords or terms without overloading your website with all of the related information.
When clicked, hyperlinked words and terms divert users to another page containing related information. This design feature is referred to as 'Inline Linking', 'Hotline Linking' or 'Piggybacking', among other terms. Inline links are employed so that the information linked from the highlighted words do not clutter the user's current webpage. Instead, users can follow the link if they are interested to find out more. This is a handy design element as it ensures clutter in the user interface is kept to a minimum, whilst providing interested users with immediate means of following links and accessing relevant and more detailed information.

## Implementing Page Structure

### Visual Framework

You are designing a website or a multi-window system; in order to help simplify the user experience, the superficial design and methods of navigation must be consistent across all the different sections of the user interface
When you are dealing with a complex user interface it is essential to implement a visual framework, as the user should be able to apply knowledge of how the website or system appears and behaves from page to page or window to window.

### Centre Stage

The most important information, panel, window or set of tools should be represented as such in the user interface. When the user is performing a task they need a central area for carry out actions and operations, whilst other subsidiary panels, tool sets etc need to be instantly accessible.
Place the most important element of the graphical user interface into the largest subsection and put the supporting tools, sets of actions etc into other, less prominent panels. By placing the most important information or frequently used panel at Centre Stage you draw the user's attention and allow them to maintain their focus when carrying out their tasks. Centre Stage is a common user interface design pattern, appearing in most applications, such as website, spreadsheets, and software packages

### Movable Pieces

There are a number of windows or panels that do not have to be arranged in a specific configuration and the user might benefit from being able to change the position of one or more of these pieces.
Where appropriate, allow users to resize and move panels around the user interface.

### Titled Sections

You are dealing with a page, widow or panel that contains a large amount of information. If everything must be displayed at once the user needs some means of isolating information of current interest from all other contents, to reduce the amount of scanning required.
Titled sections help the user skim contents and identify interesting and/or useful information.

### Responsive Enabling

At certain stages in a task, sub-components are either necessary or redundant. However, you do not want to take the user to a different location for all the different elements within a task, as that adds extra time to the process and the user would benefit from seeing all of the components of a task at once. Conversely, presenting all of the components of a task at once could be confusing, especially if redundant elements are enabled.

The gradual process of enabling users to interact with certain user interface elements as and when they need them is referred to as responsive enabling. Initially, the user is shown all of the information and user interface elements, such as checkboxes, radio buttons and input fields, in one panel, window or page, but only the those items necessary for the first sub-component of the task are enabled (i.e. active or 'interactable'). As the user makes their selections more options are enabled, whilst other, redundant options are de-activated (but still visible)

# Simplifying Data Entry

### Input fields

You require input from your users, but their responses cannot be predicted nor are they required to select an option from a constrained and preset list.
Provide users with editable input fields that are clearly distinguished from the rest of the display, so they know the region in which data can be entered is different from the non-editable 'background'.

**Why choose input fields?**
Input fields are an essential user interface design element; providing users with the means to enter non-standardised responses. They are used in many different situations, but most people will have come across them when entering personal details and delivery addresses on eCommerce web forms or sending online queries (see example image above).

Editable **input fields are now a ubiquitous element in graphical user interfaces**, so almost every user will be familiar with them and how they should interact with them. However, you must make sure to implement these input fields in a way that enables the user to see data can be entered into these regions; otherwise, the user might overlook them or struggle to see where they must enter their responses. Therefore, editable input fields are an absolute must in many different situations, but improper design can adversely affect the user experience.

If you have a constrained group of possible answers then you should probably use a dropdown menu or scrolling list, as this will save the user from having to manually enter their choice. Sometimes designers allow users to enter their response and provide a list of possible selections; like this the user can enter their choice manually if they do not need help remembering or choosing an option or they can scan the list if they do need help.

### Autocomplete

Data entry can be laborious; every effort should be made to simplify the job of inputting data. When the user is entering information that might be hard to remember, ambiguous or easily mistyped the system should support the data entry process.
Sometimes users may input data or enter search terms that can be easily matched to information already in the system. On these occasions the user would benefit from 'autocomplete', which either fills the input field automatically or provides the user with a list of possible, matching selections to choose from.

**Why choose 'Autocomplete'?**
Quite simply, by offering the user a complete search term or piece of data when they have only entered a portion of their intended selection, you make their life easier. If they only need to enter one letter then select the autocomplete option they have saved time and effort, especially if the whole term is long-winded, as is often the case with email addresses.

Autocomplete combines the flexibility of editable input fields and the speed of selection afforded by predetermined sets of options, such as in scrolling lists or dropdown menus. By using an input field the user can enter any letter, word, term, symbol etc, whilst the system helps them by providing a set of possible options that fit their data or search criteria .

An example 'autocomplete' can be seen in action above; the user has input a portion of the intended search term (Interaction design fo...) and the system has provided them with the whole term (Interaction Design Foundation) and a number of other possible suggestions, which can be selected to fill the data entry box. Autocomplete can also be used when there is a level of ambiguity, so that terms can be entered in a number of different ways. Allowing users to choose from a number of suggestions when only part of the term has been entered increases data entry speed. However, if the number of suggestions exceeds a manageable amount the dropdown list of suggestions may be more of a distraction. Therefore, it is best to set a maximum number of matching items displayed in the dropdown option list. This limit may be set according to the screen size - often an issue with android devices - or simply as a product of the number of items you feel can be immediately scanned.

Autocomplete also assists the user by providing them with continual feedback as they add characters to their search term or input data. With each new character the autocomplete list becomes more and more specific to their criteria, helping to zone in on the correct choice.

## Refining Search

When searching for a particular item, piece of information or body of content within a complex user interface, with a number of different pages and levels, the process might take a long time, especially if there is no way of searching a specific category or section of the user interface. The user does not want search results to appear from sections of the user interface where they know for certain their target item or piece of information does not reside. Therefore, the user needs some means of constraining the search engine, so results are specific to the area of the user interface they are interested in.
When dealing with a user interface with clear sections or levels, allowing the user to refine their searches according to these specific regions can help to reduce the number of irrelevant items or options they must consider, saving their time in the process. As you can see from the example above, the is able to select one of three different search refinement categories: 'In This Issue', 'In This Title' and 'All Content'.

**Why choose refining search?**

As stated, by including a 'refine search' facility the user can narrow in on their desired content with greater speed than if the search engine produced results from all levels and categories within the user interface. If results from every category and section were displayed at once there would, more likely, be a large number of irrelevant options to consider, which slows the user down and might be frustrating.

The simple inclusion of the refine search facility increases the user's sense of control of events in the user interface; ensuring the options are more and more specific to their current aims and objectives. If the user feels as though they have control over the things that happen when interacting with the system, application or website they will feel supported, which will benefit their productivity and user experience as a direct result. These positive experiences will encourage the user to continue using your design, rather than abandoning it for a design that does allow them to perform tasks in the shortest time possible.

## Forgiving Format

Users are unpredictable, which is reflected in how they enter data into input fields, such as search bars. For instance, they often leave blank spaces, abbreviate words, add capital letters and sometimes spell words incorrectly. However, you do not want to punish them for their irregularities and force them to correct things before providing them with the expected or desired response(s).
User interfaces should be simple and allow the user to enter data quickly and, where possible, without the need to correct their mistakes. Therefore, there must be a 'Forgiving Format' in your user interface design that allows the user to make mistakes, whilst correcting them on their behalf or performing the desired function without the need for corrective measures.

**Why choose 'Forgiving Format'?**

There are many small tasks in the majority of user experiences; if the user were forced to correct all of the mistakes or slip-ups they made along the way they would soon abandon the user interface for a much less demoralising experience. The user interface design must allow the user to carry out their tasks in a free and easy fashion; cleaning up after them as they go along to ensure they do not have to make continual backward steps.

As you can see from the example above, forgiving format is not just shown in the system's ability to contend with the mistakes users make. The user interface should allow the user to take a variety of different approaches to a problem, so that in the event they do not know one particular approach they can resort to another. For example, the user might not know the post/zip code for their current location, but they are aware of the city they are in; like this they can enter the latter information into the input field and carry on unhindered, rather than reaching a dead end and having to seek an alternative method.

## Event Calendar

The user is required to enter a date or a data range. For example, when purchasing train tickets or arranging a holiday.

**Why choose 'Event Calendars'?**
Event calendars (see example image above) allow the user to select a date from list, rather than input the information manually; saving them time and effort. The user might be trying to submit a date, track an order, arrange content according to a specific range of dates or filter results. By providing them with an event calendar the process for each of these occasions is faster - if the calendar has been designed in a user friendly way - as they do not have to enter the numbers nor do they have to abide by any particular format to make their selection.

For example, in the designs from some countries the user is requested to enter the date as day/month/year, such is the case in the UK, whilst other countries' designs require the user to enter the date as month/day/year. In addition, some designs request users space the day, month and year using a forward slash, whilst others use full stops. All of these formats constrain the user and force them to consider how they must input the data, as opposed to allowing them the freedom to simply select an option from a list, as they can in an event calendar.

However, people are creatures of habit and whilst the provision of an event calendar can speed up the process, many still prefer to input the data manually. For this reason, it is best to use event calendars in combination with an input field, so the user can enter their preferred date manually or select it from the calendar. In order to support the user further, you could use three separate boxes, one each for the day, month and year, so they do not have to format their data (i.e. place any punctuation mark in between), but make sure to clearly mark each box (e.g. 'D', 'M' and 'Y') so the user knows which order to enter each component of the date.

## Input Hints

Entering data can be boring and there might be times when the user needs a short prompt to help them identify exactly what they are meant to put in a particular input field.
Input hints help users establish what information should be entered into an input field.

**Why choose input hints?**
Input hints can support the user when they are filling out forms or entering data into input fields. Small hints can lead the user through a laborious task; by showing them exactly what they have to do in a particular situation they can merely follow the computer's lead and devote as little time and mental effort to the process as possible.

# Social Aspect of UI Design (03/07)

## Achievements

People often like to see or they enjoy being shown how they are performing in a particular domain. Providing users with information relating to their performance levels, such as the number of positive reviews you have gained from your peers or other users, can give them the incentive to persevere with a task or a range of tasks, depending on the nature of the experience. In addition, you could offer users the opportunity to gain collectible achievements, such as 'medals' or an increased status within an online community, as is the case on the eCommerce site Amazon where users can become a 'vine voice' member, a 'top 1000 reviewer' or even enter the 'hall of fame reviewers'.

Collectible achievements can be a persuasive addition to your design, but their effectiveness is greatly dependent on the quality of the incentives minus the level of effort required to achieve them. If the achievement rewards are arbitrary, inappropriate for the application or setting, difficult to access (i.e. they do not appear immediately in the user's dashboard or next to/in their profile) or simply lack visibility then you have fallen at the first hurdle and in all likelihood the user will feel no more engaged than if there were no achievements available. Therefore, whilst the achievements might require the user to carry out a task a certain number of times there should be minimal effort involved when accessing, determining or displaying these achievements.

## Language

You wish to convey a message to the user with brevity, but you do not want the meaning to get lost in translation.

The language you employ to help users understand what is available, what they can do, how they can carry out actions, where they can go and any other purpose of your design is a vital ingredient in the user experience. Complicated, convoluted or long-winded messages can confuse the user, slow them down, lead them to make mistakes and ultimately lead them to abandon a task, an element within your design or the whole design altogether. Therefore, it is vital to use appropriate language for each situation. For example, when you are conveying important messages the seriousness must be reflected in the tone and nature of the language used, so direct and informative is best on such occasions. As you can see from the error message descriptions above, the language is formal, clear and concise, ensuring the user knows exactly what each error codes means and the underlying problem.

In contrast, contents that are aimed at improving the fun aspects of a design should be correspondingly 'light' and laid-back. Social media sites tend to use 'friendly' language, which is intended to give the impression of light-heartedness and familiarity; consistent with the overall intentions of the service. However, this can come across as over-familiar, cynical or cringe-worthy, especially when the language is overly pally. **Therefore, it is important to maintain a consistent tone and style of language that conveys the necessary information concisely and without resorting to obscure or colloquial dialects.**

On most occasions, perhaps, the ideal is to strike a balance between robotic, precise and formal language and speaking to users as if they are your best friend. Therefore, direct conversational language offers a happy medium; offering the perfect means to convey information without seeming like computer-generated human-speak or you are trying to be best buddies. Using a conversational style plays on the user's real-world experiences and encourages them to think how to reply as they would do so during human-to-human discourse.

Users must be able to understand exactly what your are trying to say; if the language you have chosen is a barrier to this you must change it without question. The language used in your design is not an opportunity for you to flex your creative muscles; you want the user to grasp the meaning of the text immediately, so consider the user, what do they need to know? What type of language will they expect? Are your users from all over the world? Approach the task of choosing the language for your design from the user's perspective; otherwise, the purpose of the text may will complicate matters and limit the effectiveness and clarity of the underlying message.

## Person Perspective

You are either conveying information to the user or displaying information relating to their use of your product, service, application or website. You are unsure of whether to take the first, second or third person perspective to represent the user's relationship to the contents.
Use the first person perspective when you want the user to feel in control of their content without outside involvement, such is the case in social media sites where users have a personal profile, and use the second person perspective when you are conveying messages to the user, such as when referring to what the user can do in your design. In addition, if you want the user to feel open to connections, using the second person perspective (i.e. referring to things in the user's dashboard, profile or panel as 'Your ....') encourages them to think of themselves as part of something, rather than an isolated individual user. Using the word 'Your' instantly creates the sense of being involved in discourse, so the user is immediately entered into a social setting.

**Why choose person perspectives?**

The use of person perspectives helps to encourage the user to view a product, service, application etc as personally relevant and meaningful. This is particularly important in social media sites, dashboards, control panels and any other occasion where the user is meant to be in charge of the user interface. By using the **first person perspective (e.g. 'My Profile')** you give the impression that the user is in control, reinforcing the personal aspects of the design. In contrast, using the s**econd person perspective (e.g. 'Your Dashboard')** encourages the user to think of the experience of using your design as a social one.

## Leaderboards

You have a highly competitive community where current standings would give users the incentive to try harder and use your product or service more frequently.
A leaderboard, containing a list of rankings representing performance in a particular category or overall standings, helps to show users exactly how they are performing in relation to others. This can increase competitiveness and give users clear indication of how others are performing, which would not necessarily be apparent without a leaderboard.
Examples from user interface design where leaderboards are implemented include online gaming communities, online courses (such as the Leaderboard in the IDF courses 'Meet Your Peers' section), Amazon's list of top reviewers and the top tweeters, as seen in the image above.

**Why choose leaderboards?**

In competitive communities, leaderboards can encourage users to continue using your product or service as they strive to improve their standing. It is important to make it clear to the user how they are performing compared to other competitors; imagine playing a game where you never see the other person's score, you would never know when to push yourself harder or whether you are playing better than the other users. Therefore, leaderboards provide a visible and concise way of displaying relative performance levels, which can motivate users to try harder and persevere with your product or service.

## Importing Connections

The user needs to establish a list of contacts, like an address book, but entering all of them manually would be time-consuming.
Allowing the user to import contacts from one mail client, social media site or address book to another saves them time and effort.

**Why choose 'import contacts'?**

For the simple reason that it is a great means of saving the user from manually entering all of the contacts they have accumulated over time. In addition, if the user has already established a list of contacts in another application it would frustrate them if they had to go through the process all over again.

## Sign-In Reminder

Users might enter a website or attempt to engage from a number of different positions. Therefore, the experience of signing in to leave comments, make purchases, check personal profiles etc must be consistent across all of the different points and levels within the website or user interface. Furthermore, when the user attempts to engage with your design they will need a prompt to inform them that they cannot do so unless they are signed in.

At the point a user attempts to engage with a website or application, which requires them to sign in first, remind them of this requirement and provide them with the means to do so. A sign-in reminder or prompt, in combination with direct means of logging in to the website or application, helps minimise the amount of time and effort the user must invest before they are able to resume their intended activity (e.g. posting a comment within a forum).

**Why choose a sign-in reminder**

Sign-in reminders inform the user why they cannot carry out certain actions, such as leaving comments in a discussion forum or leaving seller/buyer feedback in eCommerce sites. Without a short message, alert, pop-up or re-direction to a sign-in page, the user will wonder why they cannot submit replies, feedback etc. Therefore, a sign-in reminder, in combination with the facility to immediately log in, is an important element in creating a cohesive user experience. The user should be able to carry out their activities with a few barriers as possible; a simple sign-in reminder acts as a quick and easy method of removing the obstacle blocking the user from engaging with your design.

### Update Alerts

When using websites and applications, users might like to see what new things have taken place since they last visited. For example, a user might be interested to see what their friends have been doing on a social media website or you may need to inform users of important changes that have been made to their account or the terms and conditions of their use.

Short update alerts can be a highly visible and effective means of informing users of important and personally relevant changes. Updates, also referred to as vitality or activity streams, can be used on all pages of a website and the content can be engineered according to the user's current position. Therefore, update alerts can be used to inform users of general changes (i.e. those that relate to the overall user experience) or of context-specific changes, such as bulletins showing the user what status updates their friends have made when they log in to their personal homepage.

**Why choose update alerts?**

Update alerts provide the user with interesting and/or important information immediately, saving them from expending the time and effort actively seeking it out. For example, a social media site might provide an update alert consisting of the 'wall' activity of their friends (e.g. status updates, news stories, new pictures, and new connections). If the user were to 'manually' find this information it would take them much longer; having to go from profile to profile and scrolling through each different user's contents.

### Starred Reviews

When viewing products online users would benefit from seeing/hearing the experience(s) of those who have previously bought and used these items, products or services. However, they might not want or have time to read through long reviews, so they need a quick overview of the user's experience(s).

Starred reviews provide a highly visual and instantly digestible way of conveying how previous users rate a product or service.

**Why choose starred reviews?**

Firstly, people need a rating system to help them make an informed decision when searching for products. Secondly, a starred review rating system allows people to leave quick reviews to show how they feel about a product/service. Thirdly, providing a transparent review system increases the perceived trustworthiness of a site; if you are prepared to show users what others have made of things you have for sale, potential customers will feel safe and confident when following through with a purchase.

## Dark Pattern: Methods of Coercing, Manipulating and Tricking Users

6.1 question 5 à 25 points.

### Using Dark Patterns

There are obvious reasons why designers use dark patterns (e.g. increased revenue and higher numbers of subscribers), but there is a clear downside; users will quickly lose their trust in a website if they are

constantly being tricked. For this reason, designers tend to use dark patterns sparingly; opting to hide their box of tricks in among a range of user friendly design patterns and features. A pre-selected checkbox here and a message in small print there might go unnoticed, but a website littered with dark patterns will soon lead to a mass of hasty abandonments. However, as the outcomes associated with a dark pattern become more serious, just one can turn users away from ever using your website again. For example, if you sneak expensive items into the shopping basket as users reach the checkout, they will probably try to cancel their order, take their business elsewhere and recommend others do the same. Therefore, when using dark patterns there is a careful balancing act; if the outcomes are too eye-catching or deleterious for the user the website will be dismissed as untrustworthy and if you are completely transparent you cannot trick, manipulate or coerce the user to make decisions that suit you.

**Examples:**

- Prioritising advertisements
- Automatically checked boxes
- Terms and conditions linked to command button
- Forced registration
- Using Colour to confuse
- Automatic Opt-in
- Opt-in opt-out dance
- Monthly charge
- Small print
- Sneak into basket
- Implied consent

## The Complete User Interface

Org Content
Visual Hierarchy
Progressive Disclossure
Two Pannel Selector
List Inlay
Slideshow
Wizard
Archive List
Navigation Tabs
Vertical Dropdown

Fluid Navigation
Global Navigation
Top Level Navigation
Progressive Disclossure
Mega Dropdown
Tabbed Document Interface
Breadcrums
Scrolling List
Search Boxes
Homepage Link
Sitemap Footer
Pagination
Tag Cloud
Inline Linking

Implementing Frame Structure
Visual Framework
Centre Stage
Moveable Pieces
Titled Sections
Responsive Enabling

Simple Data Entry
Input Field
Auto Complete
Refine Search
Forgiving Format
Event Calendar
Input Hints

Social Aspects of UI design
Achievements
Language
Person Perspective
Leaderboard
Importing Connections
Sign In Reminder
Update Alert
Stared Reviews

Dark Patterns
Prioritizing Advertising
Automatically Checked Boxes
Terms and Conditions Linked
Forced Registration
Color to Confuse
Automatic Opt In
Opt In Opt Out Dance
Monthly Charge
Small Print
Sneak Into Basket
Implied Consent