

L14: Graphic Design

Spring 2013

6.813/6.831 User Interface Design and Implementation

1



Google is an outstanding example of **simplicity**. Its interface is as simple as possible. Unnecessary features and hyperlinks are omitted, lots of whitespace is used. Google is fast to load and trivial to use.

But maybe Google goes a little too far! Let's take the perspective of a completely novice user coming to Google for the first time. What are the learnability obstacles to forming a mental model of what Google actually does and how to use it?

Today's Topics

- Simplicity
 - reduction
 - regularity
 - double-duty
- Contrast
 - visual variables
 - associativity & selectivity
 - squint test

Spring 2013

6.813/6.831 User Interface Design and Implementation

3

Today's lecture is the first in a series about graphic design for graphical user interfaces, and I want to explain why we're doing this.

We've made a point (earlier) that this class is focused on usability, but many of the guidelines in the upcoming lectures are more about aesthetics than pure usability. Serious *mistakes* in graphic design certainly affect usability, however, so we're trying to help you avoid those pitfalls. There's also a phenomenon, sometimes called the Aesthetic Effect, that attractive user interfaces (like attractive people) are *perceived* as more usable, whether they are or not.

But there's a larger question here: in practice, should software engineers have to learn this stuff at all? Shouldn't you just hire a graphic designer and let them do it? Some people think that the most important lesson a software engineer can learn from a course like 6.813 is "UI design is *hard*; leave it to the *experts*." The person who told me that was a high-level designer at Microsoft Research. I was tempted to retort that *designers* shouldn't bother learning to *program* either, but I don't actually believe that so I held my tongue.

But there's some substance to the argument: a little knowledge can be a dangerous thing, and that a programmer with a little experience in UI design but too much self-confidence can be just as dangerous as an artist who's learned a little bit of HTML and thinks they now know how to program. But I prefer to believe that a little knowledge is a step on the road to greater knowledge. Some of you may decide to *become* UI designers, and this course is a step along that road.

In a commercial environment, you *should* hire experienced graphic designers, just as you should hire an architect for building your corporation's headquarters and you should contract with a licensed building firm. Big jobs for big bucks require experts who have focused their education and job experience on those problem. One reason this course is useful is that you can appreciate what UI experts do and evaluate their work, which will help you work on a team with them (or supervise them).

But it's also worth learning these principles because you can apply them yourself on smaller-scale problems. Are you going to hire a graphic designer for every PowerPoint presentation you make, every chart you draw, every web page you create, every blog post you write? Those are all user interfaces. Many interactions and communications in life have a user interface, and many of them are up to you to do-it-yourself. So you should know when to leave it to the experts, but you should be able to do a creditable job yourself too, when the job is yours to do.

SIMPLICITY

Spring 2013

6.813/6.831 User Interface Design and Implementation

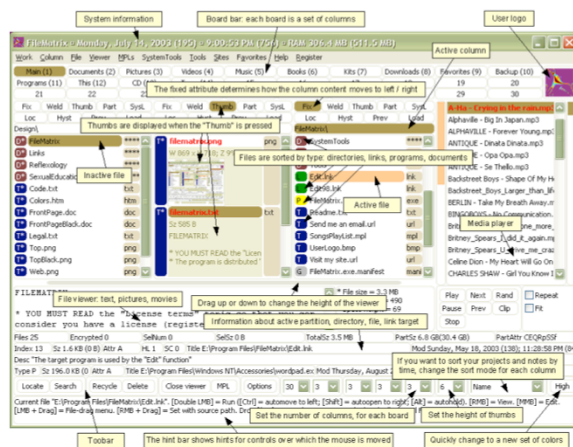
4

Simplicity

- “Perfection is achieved not when there is nothing more to add, but when there is nothing left to take away.”
 - Antoine de St-Exupery
- “Simplicity does not mean the absence of any decor... It only means that the decor should belong intimately to the design proper, and that anything foreign to it should be taken away.”
 - Paul Jacques Grillo
- “Keep it simple, stupid.” (KISS)
- “Less is more.”
- “When in doubt, leave it out.”

Okay, we'll shout some slogans at you now. You've probably heard some of these before. What you should take from these slogans is that designing for simplicity is a process of *elimination*, not accretion. Simplicity is in constant tension with other design guidelines that might otherwise encourage you to pile more and more elements into a design, “just in case.” Simplicity forces you to have a good reason for everything you add, and to take away anything that can't survive hard scrutiny.

Simplicity



Source: Alex Papadimouli

Spring 2013

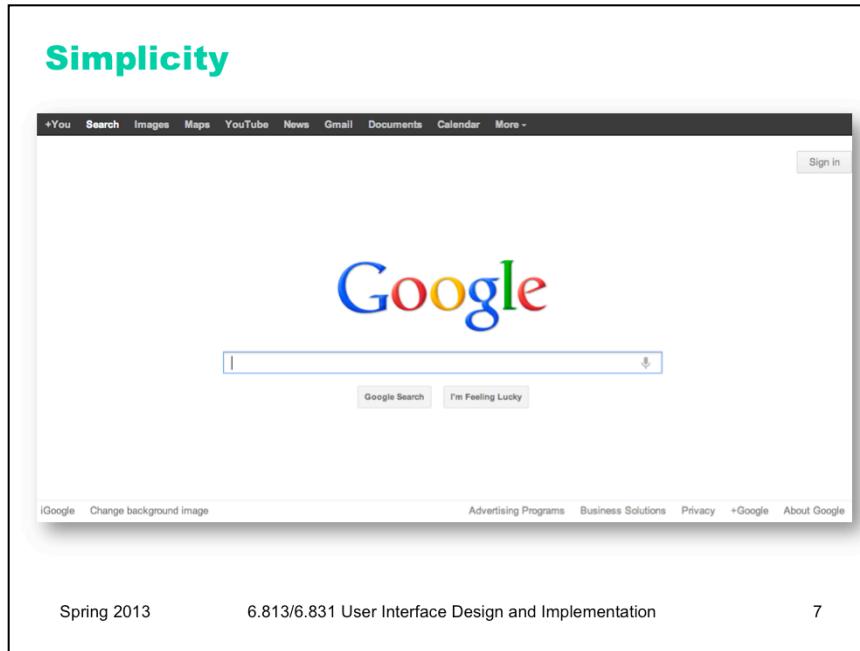
6.813/6.831 User Interface Design and Implementation

6

Although we'll largely be looking at simplicity in graphic design today, the value of simplicity to user interface design is much broader than that. Keeping a design simple, with as few parts and pieces as you can, tends to improve a design on all our usability dimensions: learnability, efficiency, and safety.

Here's a counterexample to illustrate. This is a program called FileMatrix. I have no idea what it does, but it seems to do it all. The complexity of this interface actually interferes with a lot of our usability goals: it's less learnable (because there are so many things you have to learn), less efficient (because cramming all the functions into the window means that each button is tiny), and less safe (because so many things look alike, hence description slips are easy).

Incidentally, this may be a good example of designing for yourself, rather than for others. The programmer who wrote this probably understands it completely, and maybe even uses a significant fraction of those features; but few other users will need that much, and it will just interfere with their ability to use it.



In contrast to the previous example, here's Google's start page. Google is an outstanding example of simplicity. Its interface is as simple as possible. Unnecessary features and hyperlinks are omitted, lots of whitespace is used. Google's page is fast to load and simple to use.

Simplicity certainly makes an interface easier to learn, because there's less to learn. But it can also make an interface faster and safer to use, even for an expert with that interface. Consider a remote control with 20 buttons on it. Suppose you're an expert, so you know which button to push. But you only use 3 of those buttons regularly. Which would be faster -- an interface with 3 fat buttons, or an interface with 20 little buttons? Which would be less error prone?

Techniques for Simplicity: Reduction

- Remove inessential elements



- Remove inessential features



Spring 2013

6.813/6.831 User Interface Design and Implementation

8

Here are three ways to make a design simpler. Again, we're focusing on graphic design (visual appearance here), but you can generalize these rules easily to other aspects of a design: the features that your application offers, the labels for buttons or form fields, the steps in a process, etc.

Reduction means that you eliminate whatever isn't necessary. This technique has three steps: (1) decide what essentially needs to be conveyed by the design; (2) critically examine every element (label, control, color, font, line weight) to decide whether it serves an essential purpose; (3) remove it if it isn't essential. Even if it seems essential, try removing it anyway, to see if the design falls apart.

Icons demonstrate the principle of reduction well. A *photograph* of a pair of scissors can't possibly work as a 32x32 pixel icon; instead, it has to be a carefully-drawn picture which includes the bare minimum of details that are essential to scissors: two lines for the blades, two loops for the handles. The standard US Department of Transportation symbol for handicapped access is likewise a marvel of reduction. No element remains that can be removed from it without destroying its meaning.

We've already discussed the minimalism of Google. The Tivo remote is another notable example, about minimalizing **functionality**. It's much simpler than comparable remote controls, which tend to be dense arrays of tiny rectangular buttons, all alike. Tivo's designers aggressively removed functions from the remote, to keep it as simple as possible ("Now Preening on the Coffee Table", *New York Times*, Feb 19, 2004, <http://query.nytimes.com/gst/fullpage.html?res=9c0de2d6123df93aa25751c0a9629c8b63>).

Techniques for Simplicity: Regularity

- Use a regular pattern
- Limit inessential variation among elements



Spring 2013

6.813/6.831 User Interface Design and Implementation

9

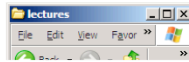
For the essential elements that remain, consider how you can minimize the unnecessary differences between them with **regularity**. Use the same font, color, line width, dimensions, orientation for multiple elements. Irregularities in your design will be magnified in the user's eyes and assigned meaning and significance. Conversely, if your design is mostly regular, the elements that you do want to highlight will stand out better.

PowerPoint's Text Layouts menu shows both reduction (minimalist icons representing each layout) and regularity. Titles and bullet lists are shown the same way.

Techniques for Simplicity: Double-Duty

- Combine elements for leverage
 - Find a way for one element to play multiple roles

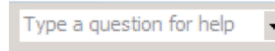
title bar



scrollbar thumb



help prompt



breadcrumbs

[Travel](#) > [Guides](#) > North America

pagination

Results Page:
1 2 3 4 5 6 7 8 9 10 ▶ [Next](#)

Spring 2013

6.813/6.831 User Interface Design and Implementation

10

Another technique for simplicity is to **double duty**, in which you try to combine elements to make them serve multiple roles in the design. Desktop and web interfaces have a number of patterns in which elements have multiple duties. For example, the “thumb” in a scroll bar actually serves three roles. It affords dragging, indicates the position of the scroll window relative to the entire document, and indicates the fraction of the document displayed in the scroll window. Similarly, a window’s title bar plays several roles: label, dragging handle, window activation indicator, and location for window control buttons. In the classic Mac interface, in fact, even the activation indicator played two roles. When the window was activated, closely spaced horizontal lines filled the title bar, giving it a perceived affordance for dragging.

The breadcrumbs pattern and the pagination pattern also do double duty, not only showing you where you are but also providing an affordance for navigating somewhere else. Pagination links, like a scrollbar, may also show you how many pages there are.

Picoquiz

Consider the evolution of a hyperlink design through three successive iterations, shown on the right. Each design's hover feedback is shown as well. Which of the following statements are true? (**choose all good answers**):

- A. The process shows the reduction technique in action.
- B. Hyperlinks are an example of the double-duty technique.
- C. Affordances can conflict with simplicity.

Visit [our store](#).

[our store](#)



Visit [our store](#).

[our store](#)



Visit our store.

[our store](#)



To answer the picoquiz questions in this lecture, go to:
<http://courses.csail.mit.edu/6.831/2013/picoquiz?lectureId=14>

CONTRAST

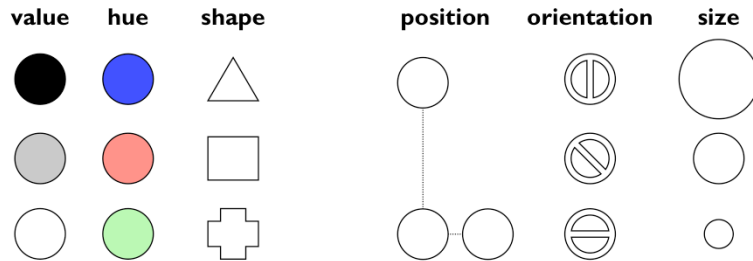
Spring 2013

6.813/6.831 User Interface Design and Implementation

12

Contrast & Visual Variables

- Contrast encodes information along visual dimensions



Spring 2013

6.813/6.831 User Interface Design and Implementation

13

Contrast refers to perceivable differences along a visual dimension, such as size or color. Contrast is the irregularity in a design that communicates information or makes elements stand out. Simplicity says we should eliminate **unimportant** differences. Once we've decided that a difference is important, however, we should choose the dimension and degree of contrast in such a way that the difference is salient, easily perceptible, and appropriate to the task.

Crucial to this decision is an understanding of the different visual dimensions. Jacques Bertin developed a theory of *visual variables* that is particularly useful here (Bertin, *Graphics and Graphics Information Processing*, 1989). Visual variables identified by Bertin are shown above. Bertin called these dimensions *retinal variables*, in fact, because they can be compared effortlessly without additional cognitive processing, as if the retina were doing all the work.

Each column in this display varies along only one of the six variables. Most of the variables need no explanation, except perhaps for hue and value. **Hue** is pure color; **value** is the brightness or luminance of color. (Figure after Mullet & Sano, p. 54).

Characteristics of Visual Variables

- Scale = kinds of comparisons possible
 - Nominal (can compare only for equality)
 - Ordered (can compare <, >)
 - Quantitative (can compare amount of difference)
- Length = number of distinguishable levels

	Value	Hue	Shape	Position	Orient	Size
Nominal	✓	✓	✓	✓	✓	✓
Ordered	✓			✓		✓
Quantitative				✓		✓
Scale	~10	~10	very long	very long	~4	~10
Spring 2013	6.813/6.831 User Interface Design and Implementation					14

The visual variables are used for communication, by encoding data and drawing distinctions between visual elements. But the visual variables have different characteristics. Before you choose a visual variable to express some distinction, you should make sure that the visual variable's properties match your communication. For example, you could display a temperature using any of the dimensions: position on a scale, length of a bar, color of an indicator, or shape of an icon (a happy sun or a chilly icicle). Your choice of visual variable will strongly affect how your users will be able to perceive and use the displayed data.

Two characteristics of visual variables are the kind of **scale** and the **length** of the scale.

A **nominal** scale is just a list of categories. Only comparison for equality is supported by a nominal scale. Different values have no ordering relationship. The shape variable is purely nominal. Hue is also purely nominal, at least as a *perceptual* variable. Although the wavelength of light assigns an ordering to colors, the human perceptual system takes no notice of it. Likewise, there may be some cultural ordering imposed on hue (red is "hotter" than blue), but it's weak, doesn't relate all the hues, and is processed at a higher cognitive level.

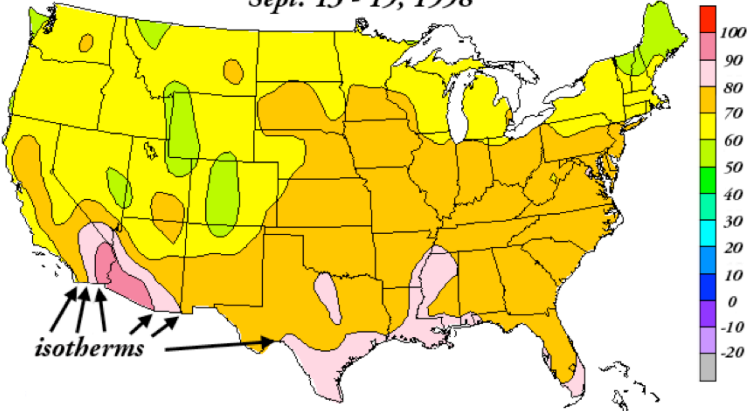
An **ordered** scale adds an ordering to the values of the variable. Position, size, and value are all ordered.

With a **quantitative** variable, you can perceive the *amount* of difference in the ordering. Position is quantitative. You can look at two points on a graph and tell that one is twice as high as the other. Size is also quantitative, but note that we are far better at perceiving quantitative differences in one dimension (i.e., length) than in two dimensions (area). Value is not quantitative; we can't easily perceive that one shade is twice as dark as another shade.

The **length** of a variable is the number of distinguishable values that can be perceived. We can recognize a nearly infinite variety of shapes, so the shape variable is very long, but purely nominal. Position is also long, and particularly fine-grained. Orientation, by contrast, is very short; only a handful of different orientations can be perceived in a display before confusion starts to set in. The other variables lie somewhere in between, with roughly 10 useful levels of distinction, although size and hue are somewhat longer than value.

Hue Is Not Ordered or Quantitative

Average Temperature (°F)
Sept. 13 - 19, 1998



Selectivity & Associativity

- Selective perception
 - Can attention be focused on one value of the variable, excluding other variables and values?
 - Shape doesn't "pop out"
- Associative perception
 - Can variable be ignored while looking at other variables?
 - Small size and low value interfere with ability to perceive hue, value, and shape

	Value	Hue	Shape	Position	Orient	Size
Selective	✓	✓		✓	✓	✓
Associative		✓	✓	✓	✓	

Spring 2013

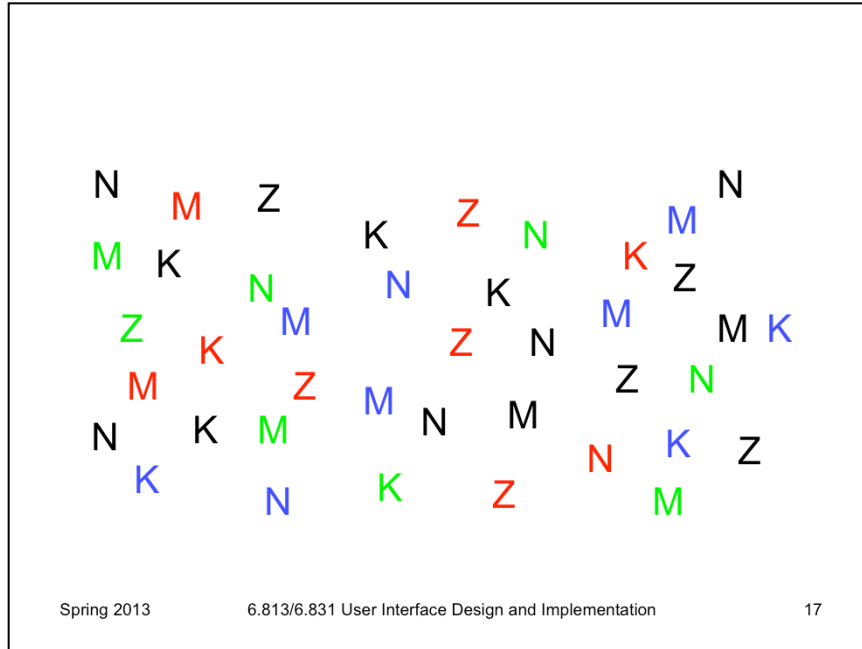
6.813/6.831 User Interface Design and Implementation

16

There are two ways that your choice of visual variables can affect the user's ability to attend to them.

Selectivity is the degree to which a single value of the variable can be selected from the entire visual field. Most variables are selective: e.g., you can locate green objects at a glance, or tiny objects. Shape, however, is not selective in general. It's hard to pick out triangles amidst a sea of rectangles.

Associativity refers to how easy it is to ignore the variable, letting all of the distinctions along that dimension disappear. Variables with poor associativity interfere with the perception of other visual dimensions. In particular, size and value are dissociative, since tiny or faint objects are hard to make out.



Ask yourself these questions:

- find all the letters on the left edge of the page (**position**)
- find all the red letters (**hue**)
- find all the K's (**shape**)

Which of these questions felt easy to answer, and which felt hard? The easy ones were **selective** visual variables.

Techniques for Contrast

- Choose appropriate visual variables
- Use as much length as possible
- Sharpen distinctions for easier perception
 - Multiplicative scaling, not additive
 - Redundant coding where needed
 - Cartoonish exaggeration where needed
- Use the “squint test”

Spring 2013

6.813/6.831 User Interface Design and Implementation

18



















Once you’ve decided that a contrast is essential in your interface, **choose the right visual variable** to represent it, keeping in mind the data you’re trying to communicate and the task users need to do with the data. For example, consider a text content hierarchy: title, chapter, section, body text, footnote. The data requires an ordered visual variable; a purely nominal variable like shape (e.g., font family) would not by itself be able to communicate the hierarchy ordering. If each element must communicate multiple independent dimensions of data at once (e.g., a graph that uses size, position, and color of points to encode different data variables), then you need to think about the effects of associativity and selectivity.

Once you’ve chosen a variable, use as much of the **length** of the variable as you can. Determine the minimum and maximum value you can use, and exploit the whole range. In the interests of simplicity, you should minimize the number of distinct values you use. But once you’ve settled on N levels, distribute those N levels as widely across the variable as is reasonable. For position, this means using the full width of the window; for size, it means using the smallest and the largest feasible sizes.

Choose variable values in such a way as to make **sharp, easily perceptible distinctions** between them. Multiplicative scaling (e.g., size growing by a factor of 1.5 or 2 at each successive level) makes sharper distinctions than additive scaling (e.g., adding 5 pixels at each successive level). You can also use redundant coding, in several visual variables, to enhance important distinctions further. The title of a document is not only larger (size), but it’s also centered (position), bold (value), and maybe a distinct color as well. Exaggerated differences can be useful, particularly when you’re drawing icons: like a cartoonist, you have to give objects exaggerated proportions to make them easily recognizable.

The **squint test** is a technique that simulates early visual processing, so you can see whether the contrasts you’ve tried to establish are readily apparent. Close one eye and squint the other, to disrupt your focus. Whatever distinctions you can still make out will be visible “at a glance.”

Choosing Visual Variables for a Display

Subject	Sender	%	Date
 Содействие в трудоустройстве.	chao		10/15/2004 4:26...
 Автовладелец	АвтоГранд		10/15/2004 4:45...
 Обучение теннису	eliot		10/15/2004 7:16 AM
 PITTSBURGH PA Silverton Home Services for...	Erica Gallenbeck		10/15/2004 7:21...
 156 - 00 - 00 наш ...	XjXFxLxmXgX@tdb.com		10/15/2004 10:4...
 156-00-00	huckster@EOPIN		10/15/2004 11:12 ...
 A Library A Dream...	Arthur GuoBin Yin		10/15/2004 6:38...
 SAVE 20% on holiday cards by shopping early	Snapfish		5:18 AM
 How are you	АНИСИНОВ К.И.		11:24 AM

Spring 2013

6.813/6.831 User Interface Design and Implementation

19

Let's look at an email inbox to see how data associated with email messages are encoded into visual variables in the display. Here are the data fields shown above, in columns from left to right:

Spam flag: nominal, 2 levels (spam or not)

Subject: nominal (but can be ordered alphabetically), infinite (but maybe only ~100 are active)

Sender: nominal (but can be ordered alphabetically), infinite (but maybe ~100 people you know + everybody else are useful simplifications)

Unread flag: nominal, 2 levels (read or unread)

Date: quantitative (but maybe ordered is all that matters), infinite (but maybe only ~10 levels matter: today, this week, this month, this year, older)

This information is **redundantly** coded into visual variables in the display shown above, for better contrast. First, all the fields use position as a variable, since each is assigned to a different column. In addition:

Spam: shape, hue, value, size (big colorful icon vs. little dot)

Subject: shape

Sender: shape

Unread: shape, hue, value, size (big green dot vs. little gray dot) and value of entire line (boldface vs. non)

Date: shape, size (today is shorter than earlier dates), position (list is sorted by date)

Exercise: try designing a visualization with these encodings instead:

Spam: size (this takes advantage of dissociativity)

Subject: shape

Sender: position

Unread: value

Date: position

Designing Information Displays

Title: HCI Bibliography : Human-Computer Interaction / User Interface ...
Summary: The HCI Bibliography (HCIBIB) is a free-access bibliography on Human-Computer Interaction, with over 20000 records in a searchable database. ... Learn about HCI ...
Keywords: HCI
URL: www.hcibib.org/
Size: 14k

[HCI Bibliography : Human-Computer Interaction / User Interface ...](#)

The HCI Bibliography (HCIBIB) is a free-access bibliography on Human-Computer Interaction, with over 20000 records in a searchable database. ... Learn about HCI. ...
www.hcibib.org/ - 14k - [Cached](#) - [Similar pages](#)

[Human-Computer Interaction Resources on the Net](#)

... This is a collection of information related to Human-Computer Interaction (HCI). ...
Collections of resources for HCI researchers and practitioners. ...
www.ida.liu.se/labs/aslab/groups/um/hci/ - 9k - [Cached](#) - [Similar pages](#)

Spring 2013

6.813/6.831 User Interface Design and Implementation

20

Here's another example showing how redundant encoding can make an information display easier to scan and easier to use. Search engine results are basically just database records, but they aren't rendered in a simplistic caption/field display like the one shown on top. Instead, they use rich visual variables – and no field labels! – to enhance the contrast among the items. Page titles convey the most information, so they use size, hue, and value (brightness), plus a little shape (the underline). The summary is in black for good readability, and the URL and size are in green to bracket the summary.

Take a lesson from this: your program's *output displays* do not have to be arranged like *input forms*. When data is self-describing, like names and dates, let it describe itself. (This is yet another example of the **double duty** technique for achieving greater simplicity – data is acting as its own label.) And choose good visual variables to enhance the contrast of information that the user needs to see at a glance.

Contrast in Publication Styles

Title

Heading

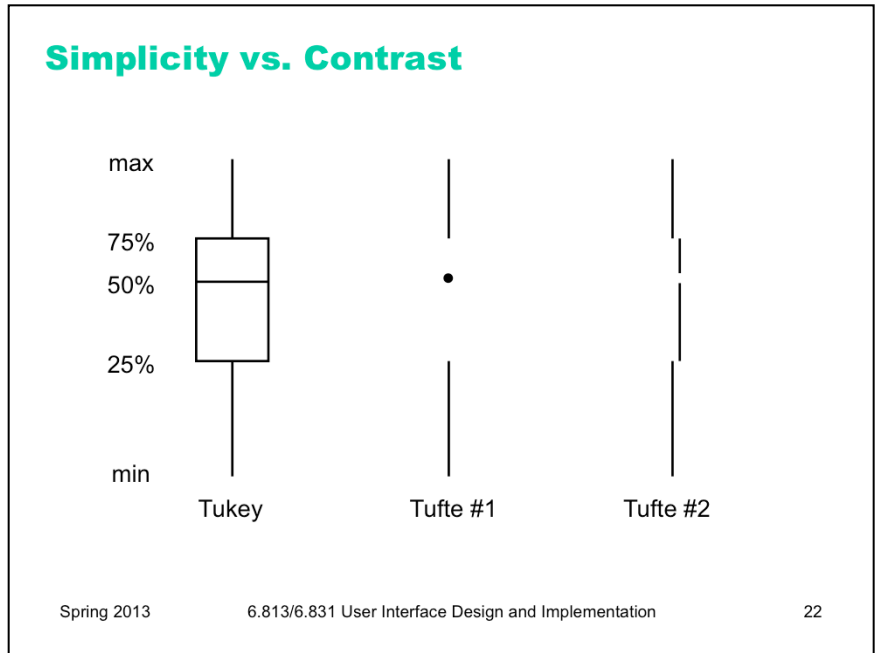
This is body text. It's smaller than the heading, lighter in weight, and longer in line length. We've also changed its shape to a serif font, because serifs make small text easier to read. Redundant encoding produces an effective contrast that makes it easy to scan the headings and distinguish headings from body text.¹



Figure 1. This is a caption, which is smaller than body text, and set off by position, centering, and line length.

¹This is a footnote. It's even smaller, and positioned at the bottom of the page.

Titles, headings, body text, figure captions, and footnotes show how contrast is used to make articles easier to read. You can do this yourself when you're writing papers and documentation. Does this mean contrast should be maximized by using lots of different fonts like Gothic and Bookman? No, for two reasons – contrast must be balanced against simplicity, and text shape variations aren't the best way to establish contrast.



Conversely, here's a case where simplicity is taken too far, and contrast suffers. Simplicity and contrast seem to fight with each other. The standard Tukey box plot shows 5 different statistics in a single figure. But it has unnecessary lines in it! Following the principle of simplicity to its logical extreme, Edward Tufte proposed two simplifications of the box plot which convey exactly the same information – but at a great cost in contrast. Try the squint test on the Tukey plot, and on Tufte's second design. What do you see?

Contrast Problems

The screenshot shows a web form with several fields and labels. The labels and their corresponding fields are:

- Form Title: J&D Software Development Order Desk
- Form Heading: Q&D Software Development Order Desk
- E-Mail responses to: dversch@q-d.com
- Text to appear in Submit button: Send Order
- Text to appear in Reset button: Clear Form
- Background Color: FFFBFD
- Text Color: 000080
- Background Graphic: (empty)
- Mailto: (radio button selected)
- Scrolling Status Bar Message: ""WebMania 1.5b with Image Map Wizard is here!""

Source: Interface Hall of Shame

Spring 2013

6.813/6.831 User Interface Design and Implementation

23

Here's an example of too little contrast. It's important to distinguish captions from text fields, but in this design, most of the visual variables are the same for both:

- the **position** is very similar: the box around each caption and text field begins at the same horizontal position. The text itself begins at different positions (left-justified vs. aligned), but it isn't a strong distinction, and some of the captions fill their column.
- the **size** is the same: captions and text fields fill the same column width
- the background **hue** is slightly different (yellow vs. white), but not easily differentiable by the squint test
- the background **value** is the same (very bright)
- the foreground **hue** and **value** are the same (black, plain font)
- the **orientation** is the horizontal, because of course you have to read it.

The result is that it's hard to scan this form. The form is also terribly crowded, which leads us into our next topic...

Picoquiz

Look at the title of this slide. Which visual variables are being used to make it contrast with the other graphical elements on the slide? (**choose all good answers**):

- A. Hue
- B. Size
- C. Orientation
- D. Position
- E. Font
- F. Value

To answer the picoquiz questions in this lecture, go to:
<http://courses.csail.mit.edu/6.831/2013/picoquiz?lectureId=14>

Summary

- Strive for simplicity
 - **Reduce** features and data displayed
 - **Regularize** visual properties that aren't important
 - Make elements perform **double-duty**
- Enhance contrast
 - Choose visual variables carefully
 - Use the squint test