# Table of Contents

[Table of Contents](#_s7oxk5s9x2o9)

[Meta-Data](#_wmw6vju0tvzt)

[Lesson Goals](#_y0f332pzpsd3)

[Lesson Outcomes](#_yuun61fn00lj)

[Assessments](#_qy4klj35a395)

[Lesson Plan](#_jhf5z8xjau2m)

[Script](#_x1hgyjwhsibz)

[2.4.1 Introduction](#_xgus6bu1y1vv)

[2.4.1.1 Headshot Studio](#_e72ot2dhuuu8)

[2.4.2 Information Processing](#_tyoo57dqnq2n)

[2.4.2.1 Tablet Studio](#_n4t9h53gw6eo)

[2.4.3 Sensation and Perception: Visual](#_de22vadn37yi)

[2.4.3.1 Headshot Studio (Morgan)](#_7dtkj85dumam)

[2.4.4 Sensation and Perception: Auditory](#_n58g73o2n4iw)

[2.4.4.1 Headshot Studio (Morgan)](#_qv2cwsfu0t9z)

[2.4.5 Sensation and Perception: Haptic](#_j2lgjauwhugl)

[2.4.5.1 Headshot Studio (Morgan)](#_9q3e332811z7)

[2.4.6 Design Challenge: Message Alerts](#_suc6d71wyc0o)

[2.4.6.1 Headshot Studio (Morgan)](#_9mxks0nfuigo)

[2.4.6.2 Exercise](#_m975qgq09whn)

[2.4.6.2 Headshot Studio (Morgan)](#_dufxljjx8t0b)

[2.4.7.1 Memory: Perceptual Store](#_bi03hgdashzt)

[2.4.7.1 Tablet Studio](#_qik0yfuthjgq)

[2.4.7.2 Exercise](#_ktp3bvxf3iha)

[2.4.7.3 Tablet Studio](#_wkt45se97n1u)

[2.4.8 Memory: Short-Term and Chunking](#_any85qbmxa7v)

[2.4.8.1 Tablet Studio](#_lqveh8iwf9ff)

[2.4.8.2 Exercise](#_yi1oodz0e909)

[2.4.8.3 Tablet Studio](#_hyaejeylrzfm)

[2.4.9 Memory: Short-Term and Recognition](#_q5z3rk9w7oc0)

[2.4.9.1 Tablet Studio](#_sp6h7nn50ift)

[2.4.9.2 Exercise](#_m4ppiphrgods)

[2.4.9.3 Tablet Studio](#_51yj0ltf6djn)

[2.4.10 Memory: Short-Term Takeaways](#_soyzrc6eqjdh)

[2.4.10.1 Headshot Studio](#_1dhdkepv9wrc)

[2.4.11 Memory: Long-Term Memory](#_n2fmkcb1rucb)

[2.4.11.1 Tablet Studio](#_i3scpj6x8c85)

[2.4.11.1 Tablet Studio (On Camera)](#_iqjrno4if0t0)

[2.4.12 Cognition: Learning](#_d2jeuayxmxuv)

[2.4.12.1 Tablet Studio](#_22cdbvihw8bg)

[2.4.12.1 Headshot Studio](#_j76btbtw68on)

[2.4.13 Cognition: Cognitive Load](#_pvaraeu2p16g)

[2.4.13.1 Tablet Studio](#_a3j2zwi1t2c5)

[2.4.14 Reflections: Cognitive Load](#_bpv22k9mhei1)

[2.4.14.1 David’s House (Office)](#_ju9ufic19xcb)

[2.4.14.2 Exercise](#_3sv6y4xdx7rt)

[2.4.14.3 David’s House (Office)](#_bg6nr4syxhgc)

[2.4.15 5 Tips: Reducing Cognitive Load](#_cww6wojgmr4s)

[2.4.15.1 Headshot Studio](#_jadoltu974x9)

[2.4.16 Motor System](#_kded3u5b0jfz)

[2.4.16.1 Tablet Studio](#_p7g68h99r2sy)

[2.4.17 Exploring HCI: Human Abilities](#_16o23y2qyf5n)

[2.4.17.1 Headshot Studio](#_z3a09mt15k1l)

[2.4.18 Conclusion](#_7cutccqwj3zg)

[2.4.18.1 Headshot Studio](#_4uj3ntuwcgpi)

# Meta-Data

## Lesson Goals

* Students will understand the three main human abilities: perception, cognition, and action.
* Students will understand the strengths and weaknesses of the different types of perception in designing interfaces.
* Students will understand the limitations of human cognition and memory and its relevance to designing interfaces that strain working memory.
* Students will understand the role of the motor system in designing interactions.

## Lesson Outcomes

* Students will be able to design interactions within the limitations and opportunities of the human sensory system.
* Students will be able to identify interfaces that strain human cognition.
* Students will be able to design interfaces that reduce cognitive load during tasks.
* Students will be able to describe and design around the physical limitations of users during tasks in context.

## Assessments

* Students will complete exercises illustrating the limitations of human memory and cognition.
* Students will reflect on the application of the lesson’s concepts to their chosen area of HCI.
* Students will engage in a short design task based on the lesson’s concepts.
* Students will complete a short answer assignment in which they critique a provided interface from the perspective of the lesson’s concepts.
* Students will complete a short answer assignment in which they select an interface to critique from the perspective of the lesson’s concepts.
* Students will complete a short answer assignment in which they design a revision of one of the critiqued interfaces from the perspective of the lesson’s concepts.

## Lesson Plan

* Students will initially be reintroduced to the notion of humans as information processors for the purpose of this lesson.
* The information processing system of perception to cognition to motor system will be described.
* Students will then be introduced to the most relevant portions of the human perceptual system for interaction design.
* Students will then be introduced to the different parts of human memory.
* Working memory will be used to introduce concepts of learning and cognitive load.
* Students will finally be introduced to the limitations of the motor system and how to design with it in mind.

# Script

## 2.4.1 Introduction

### 2.4.1.1 Headshot Studio

* [C] David talking
* [A] Clips of the lesson to the right
* Human-Computer Interaction starts with human, so it’s important that we understand who the human is and what they’re capable of doing.
* [B] Topic; Psychology
* So, in this lesson, we’re going to bring up some **psychology** of what humans can do.
* We’ll look at three systems: input, processing, and output.
* [B] Topic; Input
* **Input** is how stimuli are sensed from the world and perceived in the mind.
* [B] Topic; Processing
* **Processing** is cognition, how the brain stores and reasons over the input it’s received.
* [B] Topic; Output
* **Output** is how the brain then controls the individual’s actions in the world.
* Now, we’re going to cover a lot of material at a high level.
* If you’re interested in hearing more, I’d recommend taking a psychology class focusing especially on sensation and perception.
* We’ll put some recommended courses in the notes.

## 2.4.2 Information Processing

### 2.4.2.1 Tablet Studio

* [V] Feedback cycle diagram
* In discussing human abilities, we’re going to adopt something similar to the ‘processing’ via of the human.
* For now, we’re interested in what they physically can do.
* [V] Zoom in on the left side, cutting the arrows in half and only showing the human with input and output.
* So, we’re going to focus exclusively on what’s going on over here.
* This is the information processing model, which involves three major systems: perception, cognition, and response.
* Notice that in this lesson, we’re going to be discussing the human similar to how we discuss the computer in most lessons: as something that takes input and produces output.
* For now, though, we’re entirely interested in how the human does this.

## 2.4.3 Sensation and Perception: Visual

### 2.4.3.1 Headshot Studio (Morgan)

* [C] Close-up on Morgan’s face
* Let’s start by talking a bit about what the average person can sense and perceive.
* Here we have Morgan again.
* Morgan has eyes.
* [A] As these sentences are read, icons might appear on the right representing each point.
* Morgan’s eyes are useful for lots of things.
* The center of Morgan’s eye is most useful for focusing closely on color or tracking movement.
* So, we can assume that the most important details should be placed in the center of her view.
* Morgan’s peripheral vision is good for detecting motion, but it isn’t as good for detecting color or detail.
* So, while we might use her periphery for some alerts, we shouldn’t require her to focus closely on anything out there.
* As a woman, Morgan is unlikely to be color blind -- she has about a 1 in 200 chance.
* Men have a much greater prevalence of color blindness, at about 1 in 12.
* Either way, that’s a significant body of people, so we want to avoid relying on color to understand an interface. We can use it to emphasize knowledge that is already present in the system, but using the system shouldn’t rely on perceiving color.
* Sight is directional.
* If Morgan is looking the wrong way or has her eyes closed, she’ll miss visual feedback.
* As Morgan gets older, her visual acuity will decrease.
* So, if we’re designing something with older audiences in mind, we want to be careful of things like font size. Ideally, these would be adjustable to meet the needs of multiple audiences.
* Altogether, though, Morgan’s visual system is hugely important to her cognition.
* The majority of concepts we discuss in HCI are likely connected to visual perception.

## 2.4.4 Sensation and Perception: Auditory

### 2.4.4.1 Headshot Studio (Morgan)

* [C] Close-up on Morgan’s face
* Morgan also has ears.
* [A] Icons appear on the right illustrating these points.
* Morgan can discern noises based on both their pitch and their loudness.
* Her ears are remarkably good at localizing sound as well.
* In fact, she can tell the difference between a nearby quiet sound and a faraway loud sound even if their relative pitches and loudnesses are the same when they reach her ear.
* Unlike vision, hearing isn’t directional.
* Morgan can’t close her ears or point her ears the wrong direction, so she can’t as easily filter out auditory information.
* That might be useful for designing alerts, but problematic for overwhelming her, or sharing too much with the people around her.

## 2.4.5 Sensation and Perception: Haptic

### 2.4.5.1 Headshot Studio (Morgan)

* [C] Close-up on Morgan’s face
* Morgan’s skin can feel things.
* It can’t feel at a distance, but it can feel when things are touching up against it.
* It can feel a variety of different types of input, like pressure, vibration, and temperature.
* Like listening, Morgan can’t easily filter out touch feedback.
* But unlike listening, touch feedback is generally only available to the person it’s touching, so it can be used to create more personal feedback.
* [C] Cut to Morgan typing on a computer
* Traditionally, touch feedback or haptic feedback has been very natural.
* Morgan feels the keys go down as she presses them.
* But with touchscreens, motion controls, and virtual reality, touch needs to more and more be designed explicitly into the system if we’re to use it.

## 2.4.6 Design Challenge: Message Alerts

### 2.4.6.1 Headshot Studio (Morgan)

* [C] Morgan sitting at the desk typing, with a phone next to her computer
* Let’s design something for Morgan real quick.
* Let’s tackle the common problem of being alerted when you’ve received a text message.
* Here are the constraints on our design for Morgan:
* It must be alert her whether the phone is in her pocket or on the table.
* It cannot disturb the people around her.
* <Phone vibrates>
* And yes, vibrating loudly against the table counts as disturbing the people around here.
* You’re not restricted to just one modality, though, but you *are* restricted to the sensors the phone has available.

### 2.4.6.2 Exercise

* “Click to continue” exercise

### 2.4.6.2 Headshot Studio (Morgan)

* [C] Morgan at her desk
* So, here’s one possible design.
* We know that smartphones have cameras and light sensors on them.
* We can use that to determine where the phone is and what kind of alert should trigger.
* If the sensor detects light, that means the screen is visible. So, it might alert her simply by flashing its flashlight or illuminating the screen.
* If the sensor does not detect light, it would infer that the phone is in her pocket, and thus would vibrate it instead.
* Of course this isn’t perfect: it could be in her purse, or she could have put it face down.
* That’s why we would iterate on a design like this, to improve it based on the user’s experiences.

## 2.4.7.1 Memory: Perceptual Store

### 2.4.7.1 Tablet Studio

* [V] Left half of feedback cycle appears
* After the perception portion of this model comes the cognition portion, starting with memory.
* [V] Timeline appears across the top with three different-color segments, each ~5x wider than the last, representing perceptual store, short-term memory, and long-term memory.
* [V] Segments are labeled with arrows downward.
* There are lots of different models of memory out there.
* For our purposes, we’re going to talk about three different kinds of memory: the perceptual store, the short-term memory, and the long-term memory.
* Some scientists argue that there are other types as well, like an intermediate “back of your mind” kind of memory, but the greatest consensus is around the existence of at least these three kinds.
* Let’s start with the first, perceptual store or working memory.
* [V] Labels fade
* The perceptual store is a very short-term memory, lasting less than a second.
* One of the most common models of this working memory came from Alan Baddeley and Graham Hitch in 1974.
* They described it as having three parts.
* [V] Sketchpad appears
* The visuospatial sketchpad holds visual information for active manipulation.
* Picture a pencil, for example. The visuospatial sketchpad is where you’re seeing that pencil.
* [V] Phonological loop appears
* [B] Also known as: the articulatory loop or phonological store
* The phonological loop is **similar**, but for verbal or auditory information. It stores the sounds or speech you’ve heard recently.
* [V] Episodic buffer appears
* [B] The episodic buffer was added to the model in 2000.
* The episodic buffer takes care of integrating information from the other systems, as well as chronological ordering to put things in place.
* [V] Central executive appears.
* The final component, the central executive is responsible for coordinating these various systems.
* So, let’s try a simple example of this.
* I’m going to very quickly show you a picture, then ask you a question about it.
* Don’t focus on any particular portion of the picture, try just to view it as a whole.
* [V] Flash baseball game picture from KBAI for one second.
* What was the score on the scoreboard?

### 2.4.7.2 Exercise

* [E] What was the score on the scoreboard?
* [E] A. 0-0
* [E] B. 2-2
* [E] C. 5-2
* [E] D. 5-3

### 2.4.7.3 Tablet Studio

* [V] Baseball game image
* As you can see, the score was 0-0.
* Now as you tried to reason over that, what you probably did was picture the image in your mind.
* That was trying to reason over what was stored in the perceptual buffer, and it decayed very quickly.
* However, if you’re a fan of baseball, you probably had a better chance of getting that right.
* That’s because when you have domain expertise, you’re able to better process images about that domain more quickly.
* You might, for example, have recognized that most of the innings weren’t marked, so the odds are that the score was low.
* [V] Image of a chess board comes up
* This idea is actually the foundation of a fascinating study about chess experts vs. novices and recognizing the configurations of chess boards.
* [B] (On the left) This is the final board from the first game between Kasparov and IBM’s Deep Blue!
* The study found that experts are much better at recalling realistic chess boards that are only flashed for a second, but are no better than novices at recalling unrealistic chess **boards**.
* So, expertise or rehearsal delays the decay of the perceptual buffer.

## 2.4.8 Memory: Short-Term and Chunking

### 2.4.8.1 Tablet Studio

* [V] Show short-term memory label
* When we’re designing interfaces, short-term memory is one of our biggest concerns.
* It’s important we avoid requiring the user to keep too much stored in short-term memory.
* [V] Show “4-5 chunks at a time”
* The current state of the art says that people can only hold 4 to 5 chunks in short-term memory at one time.
* There are two principles we need to keep in mind here, though.
* The first is just that idea of chunking.
* To illustrate this, let’s try it.
* We’re about to show you six combinations of five letters.
* Try to memorize them, then enter them.
* [V] Show three columns of two five-letter combinations. On the left are non-words, like “FSDVW” and “POBXC”. In the middle are word-like ‘words’, like “PODAP” and “REXIC”. On the right are real words, like “PUPPY” and “WATCH”.
* [V] Hide after ~5 seconds.
* Now to keep you from just rehearsing it in your perceptual store until you can re-enter it, I’m going to stall and show you some pictures of my cats.
* Ok, now fill in the words.

### 2.4.8.2 Exercise

* [E] “What were the six words? Enter one per line.”
* [E] Text box to enter the words.

### 2.4.8.3 Tablet Studio

* [V] Blank exercise image
* What happened?
* Well, what likely happened is you had little trouble remembering the two real words
* You might have had a bit more trouble remembering the two fake words that nonetheless looked like words
* And you probably had a lot of trouble remembering the things that didn’t look like words at all.
* Why is all that?
* Well when it came to memorizing these words, you were just calling up a chunk you already had. You didn’t see these as collections of letters, you saw them as words.
* For the ones in the middle, you’ve never seen these combinations before, but you can nonetheless pronounce them. So, you likely saw them as fake words rather than collections of letters.
* For these, though, you had to memorize five individual letters.
* So, that means that while these four were individual chunks to memorize, these two each represented five individual chunks.

## 2.4.9 Memory: Short-Term and Recognition

### 2.4.9.1 Tablet Studio

* [V] Exercise from below
* However, there is a way we can make this easier.
* Which of these six sequences of letters did you see earlier?

### 2.4.9.2 Exercise

* [E] Which of these six sequences of letters did you see earlier? Check two.
* [E] Six checkbox answers, two from the previous set.

### 2.4.9.3 Tablet Studio

* [V] Exercise from above
* Even if you had trouble earlier naming those series of letters, you probably succeed at this.
* Why? Because it’s far easier to recognize something you know than to recall it independently.
* That will be a useful takeaway for us as we design interfaces: we can minimize the memory load on the user by relying more on their ability to recognize things than recall them.

## 2.4.10 Memory: Short-Term Takeaways

### 2.4.10.1 Headshot Studio

* [C] David talking
* [A] Ten numbers appear spaced out on the right or above David’s head
* So, what are the implications of short-term memory for HCI?
* We do not want to ask the user to hold too much in working memory at a time.
* 4 to 5 chunks is all.
* Asking the user to hold ten numbers in short-term memory, for example, would probably be too much.
* However, we can increase the user’s effective short-term memory capacity by helping them chunk things.
* [A] Numbers form into three chunks, mimicking a phone number.
* For example, this is probably far easier to remember even though it’s the same content.
* We’ve chunked ten items into three, and we’ve used a format with which you’re familiar if you’re in the United States.
* If you’re from outside the US, then you might be familiar with a different grouping, but the same principle applies.
* [A] Numbers disappear
* And finally, when possible, we should leverage recognition over recall.
* For example, if I asked you to recite the number, maybe you could.
* Go ahead, try it.
* <pause>
* [A] Four 10-digit numbers reappear
* Whether or not you could do that, you almost certainly can pick it from this list
* This is one of the reasons why menu bars and toolstrips are so ubiquitous in software design.
* The user doesn’t have to remember the icon for a command or the name of an option, they just need to recognize it when they see it.

## 2.4.11 Memory: Long-Term Memory

### 2.4.11.1 Tablet Studio

* [V] Timeline highlighting long-term memory
* Finally, we have long-term memory.
* Long-term memory is a seemingly unlimited store of memories, but it’s harder to put something into long-term memory than to put it into short-term memory.
* In fact, to load something into long-term memory, you generally need to put it into short-term memory several times.

### 2.4.11.2 Tablet Studio (On Camera)

* [C] David sitting at the desk
* To demonstrate this, I’m going to describe something called the Leitner System.
* The Leitner System is a way of memorizing key value pairs, or in other words, a way of memorizing flashcards.
* Those could be words and definitions, countries and capitals, laws and formulas, etc.
* Anything where you’re given a key and asked to return the value.
* So I have some flashcards here that are for the capitals of the world.
* What I do is go through each one, read the country, and try to remember the capital.
* For each one I can remember, I put it in the pile on the right.
* For each that I can’t remember, I put it in the pile on the left.
* Tomorrow, I’m going to just do the pile on the left.
* Any that I remember tomorrow, I’ll add to the pile on the right.
* Any that I don’t will stay in the pile on the left.
* Thus, I’m focusing my attention on the ones that I don’t yet know.
* In four days, I’ll go through the pile on the right.
* Any that I don’t remember this time get moved back to the left, any that I do remember stay on the right.
* Thus, the things that I remember least are loaded into short-term memory most often, solidifying them in my long-term memory.
* In practice, you would do this not with just two piles, but with four or five, reviewing the fuzzier ones monthly or even yearly, just to check whether they’ve decayed.

## 2.4.12 Cognition: Learning

### 2.4.12.1 Tablet Studio

* [T] A Microsoft Word, PowerPoint, etc. window
* Now let’s talk a little about cognition.
* One of the most important cognitive processes to consider is learning.
* When we design interfaces, we are in some ways hoping the user has to learn as little as possible to find the interface useful.
* However, our interfaces should also teach the user over time how to use them most efficiently.
* We can take Microsoft PowerPoint as an example of this.
* Let’s pretend this is the first time I’ve used PowerPoint and I want to copy something.
* The application doesn’t assume I know anything yet.
* If I poke around, I’ll find the Copy button on this menu.
* However, it also lists a hotkey over on the right.
* That’s helping me learn to interact with the application more efficiently, through hotkeys instead of clicking around in menus.
* [V] Return to slide, showing “procedural learning” on the left and “declarative learning” on the right.
* There are two kinds of learning we’re most interested in: procedural and declarative.
* [V] “How to do something” appears under ‘procedural’
* Procedural is how you *do* something
* [V] “Knowledge about something” appears under ‘declarative’
* Declarative is what you know *about* something
* So, if I asked you: what’s the hotkey for ‘Paste’, I’m asking for declarative knowledge.
* If I asked you: please paste your clipboard here, I’m asking for procedural knowledge.

### 2.4.12.2 Headshot Studio

* [C] David talking
* What’s interesting is that while declarative knowledge is how we generally communicate with one another, procedural knowledge is generally what we do in HCI.
* When you have strong procedural knowledge, you may forget how you’re doing what you’re doing because it’s so second-nature.
* You’re unconsciously competent with what you’re doing.
* When you’re in that state, it can be difficult to explain to someone who lacks that competence because you aren’t sure what makes you good at it.
* It’s difficult to translate your subconscious procedural knowledge into explicit declarative knowledge, but declarative knowledge is how we communicate.
* That’s what we use to communicate with novice users.
* This is important because as the designers of interfaces, we’re the experts in our domains. That means we’re prone to design things that are easy for us to use, but hard for anyone else.

## 2.4.13 Cognition: Cognitive Load

### 2.4.13.1 Tablet Studio

* [V] Visualization on the right of a “meter” of resources: probably a black rectangle with green fill, with 0% at the bottom, 50% at the middle, and 100% at the top. An icon of a brain can be on the left of the meter.
* To talk about cognitive load, let’s think for a moment of the brain like a computer.
* The community is actually divided on whether or not the brain actually operates that way, but for the purposes of this explanation, it’s a useful metaphor.
* So, your brain has a certain number of resources available to it.
* Each thing the brain is working on saps those resources.
* [V] The bottom 60% of the green changes color, and an arrow points to that portion reading “Calculus problem”
* Let’s say that you’re at home in a quiet area working on a calculus problem that requires 60% of your cognitive resources.
* In that setting, you have plenty of resources to solve the problem.
* [V] The top half of the rectangle also changes color, overlapping with the Calculus problem portion.
* However, then you go to take the test.
* Now you’re worried about the test, how much effect it has on your grade, etc.
* That stress commands 50% of your cognitive resources.
* Now, you don’t have sufficient resources to complete the test successfully.
* I hypothesize that’s why test-taking anxiety can have such a negative effect: it takes resources away from problem-solving to use on anxiety.
* You can apply these same principles to the presence of distractions, anxiety disorders, and more.
* [V] Return to original black rectangle and green fill.
* Cognitive load has two major applications to our work in designing interfaces.
* [V] Green rectangle subdivides into areas marked “Task” and “Interface”.
* One, we want to reduce the cognitive load posed by the interface so the user can dedicate more of their resources to the task itself.
* [V] Replace “Task” and “Interface” with more rectangles: “Monitoring own driving”, “Monitoring surround cars”, “Route planning”, “Using GPS”
* Second, we want to understand for our context what other tasks are competing for cognitive resources.
* If we’re designing a GPS, for example, we want to be aware that the user will have relatively few cognitive resources to devote to interacting with the interface.

## 2.4.14 Reflections: Cognitive Load

### 2.4.14.1 David’s House (Office)

* [C] David talking
* Let’s take a second and reflect on cognitive load.
* Try to think of a task you’ve encountered where you’ve experienced a high cognitive load.
* What different things did you have to keep in mind at the same time?
* How could an interface have helped this problem?

### 2.4.14.2 Exercise

* “Click to continue” exercise

### 2.4.14.3 David’s House (Office)

* [C] David talking
* Computer programming is one task with an incredible high cognitive load.
* At any given time you’re likely holding in working memory your goals for this line of code, your goals for this function, your goals for this portion of the program as a whole, the variables you’ve created, and a lot more.
* That’s why there are so many jokes about how bad it is to interrupt a programmer.
* But there are ways good IDEs can help mitigate those issues.
* For example, in-line automated error checking is one way to reduce the cognitive load on programmers because it lets them focus more on what they’re trying to accomplish than on low-level syntax mistakes.
* In that way, the IDE offloads some of the responsibility from the user onto the interface.
* We could phrase this a little differently, too. We could describe this as distributing the cognitive load more evenly between the different components of the system: myself and the computer.
* That’s a perspective we discuss when we talk about distributed cognition.

## 2.4.15 5 Tips: Reducing Cognitive Load

### 2.4.15.1 Headshot Studio

* [C] David talking
* Here are five quick tips for reducing cognitive load in your interfaces.
* [B] 1. Use multiple modalities.
* **1. Use multiple modalities.** Most often, that’s going to be both visual and verbal. When only one system is engaged, it’s natural for it to become overloaded while the other system is bored. So, describe things verbally, and also present them visually.
* [B] 2. Let the modalities complement each other.
* **2. Let the modalities complement each other.** Some people will take that first tip and use it as justification to present different content in the two modalities. That actually increases cognitive load because the user has to try to process two things at once… as you just noticed with Amanda putting up some irrelevant material while I said that.
* [Amanda] Sorry!
* Instead, focus on letting each modality support, illustrate, or explain the other instead of competing with the other.
* [B] 3. Give the user control of the pace.
* **3. Give the user control of the pace.** That’s more pertinent in educational applications of cognitive load, but often times interfaces have built-in timers on things like menus fading away or selections needing to be made. That dictates the pace, increases stress, and raises cognitive load. Instead, let the user control the pace.
* [B] 4. Emphasize essential content and minimize clutter.
* **4. Emphasize essential content and minimize clutter.** The principle of discoverability says we want the user to be able to find the functions available to them, but that could also raise cognitive load if we just present a list of 500 options to them. To alleviate that, we can design our interfaces in a way that emphasizes the most common actions while still giving access to the full range of possible options.
* [B] 5. Offload tasks.
* **5. Offload tasks.** Look closely at what a user has to do or remember at every stage of the interface’s operation, and ask if you can offload part of that task. For example, if a user needs to remember something they entered on a preview screen, show them what they entered. If there’s a task they need to do manually that can be triggered automatically, trigger it automatically.

## 2.4.16 Motor System

### 2.4.16.1 Tablet Studio

* [V] Diagram of left side of feedback cycle
* So, the user has received some input.
* It’s entered memory, they’ve cognitively processed it.
* Now, it’s time for them to act in the world.
* In designing interfaces, we’re also interested in what is physically possible for users to do.
* This includes things like how fast they can move and how precisely they can tap.
* [V] Visual of two Spotify screens comes up, [this one](http://cdn02.androidauthority.net/wp-content/uploads/2013/06/Spotify-notification-controls-2-2-645x449.jpg) and [this one](http://cdn2.ubergizmo.com/wp-content/uploads/2013/06/spotify-android-538x640.jpg).
* For example, here are two versions of the Spotify control widget that appears on Android.
* On the left is the version that sites in the tray at the top of the screen, on the right is the version on the lock screen.
* [B] Elsewhere in HCI; The ‘consistency’ principle, from the Design Principles and Heuristics lesson.
* In each case, the ‘x’ closes, which is **consistent** with a lot of other applications.
* The forward, back, and pause buttons are similar consistent with their usual meanings.
* [B] Elsewhere in HCI; The ‘mapping’ principle, from the Design Principles and Heuristics lesson.
* I don’t actually know what the plus sign does: it doesn’t have a clear **mapping** to an underlying function.
* Now, note on the left, we have the close button in the top right, while on the right, the “Next” button right next to the “Close” button.
* I can speak from personal experience and say that the level of specificity required to tap forward instead of the x is difficult, especially if you’re using this while running or driving.
* The precision of tapping involved under those situations is much lower, meaning that people were often closing the widget when they meant to skip a song.
* This wasn’t an error in their perception of the screen, their memory of the controls, or their cognition about the task: it was simply an error in the motor system because the interface relied on them having more precision than they had in that circumstance.
* This design doesn’t take into consideration the motor system of the user.
* This is one example of how we need to be aware of the constraints on the user’s motor system; what they physically can do, how precise or accurate they can be, and so on.
* Now, there might be other constraints around this. There might be a reason why these buttons have to be placed here.
* [B] Elsewhere in HCI; The ‘tolerance’ principle, from the Design Principles and Heuristics lesson.
* In that case, we would need to make our interface **tolerant** of errors.
* Maybe we require a double-tap to actually close the app, or maybe we mute it when it’s pressed but give the user five seconds to confirm their choice.

## 2.4.17 Exploring HCI: Human Abilities

### 2.4.17.1 Headshot Studio

* [C] David talking
* We’ve talked a lot about different human abilities in this lesson.
* Depending on the domain you chose, the human abilities in which you’re interested may vary dramatically.
* If you’re looking at gestural interfaces or wearable devices, then the limitations of the human motor system might be very important.
* On the other hand, if you’re interested in educational technology, you’re likely more interested in some of the cognitive issues around designing technology.
* For virtual reality, your main concern will likely be perception, although there are interesting open questions about how to physically interact with virtual reality as well.
* So, take a few moments and reflect on what the limitations of human ability are in the domain of HCI that you chose to explore.

## 2.4.18 Conclusion

### 2.4.18.1 Headshot Studio

* [C] David talking
* [V] Clips on the right
* Today we’ve gone through a crash course on human abilities, perception, and universal design.
* [B] Topic; Perception
* We started off by talking about the main ways people **perceive** the world around them: sight, sound, and touch.
* [B] Topic; Cognition
* Then we discussed some of the components of **cognition**, especially memory and learning.
* [B] Topic; Motor System
* Then we discussed the **motor system**, how the person then interacts with the world in return.
* In this single lesson, we’ve only scratched the surface of human perception.
* [B] Examples of other courses
* There are entire courses, even entire degree programs, that focus on these principles -- we’ll give you some suggestions in the notes.
* So, don’t think we’ve given you the full view of the field.
* Instead, we hope we’ve given you enough to start to keep human abilities in mind, and enough to know what to research as you start to learn to design interfaces.