Table of Contents

- Table of Contents
- Meta-Data
  - Lesson Goals
  - Lesson Outcomes
  - Assessments
  - Lesson Plan
- Script
  - 2.6.1 Introduction
    - 2.6.1.1 Headshot Studio
  - 2.6.2 Mental Models
    - 2.6.2.1 Headshot Studio
  - 2.6.3 Mental Models and Education
    - 2.6.3.1 Headshot Studio
  - 2.6.4 Mental Models in Action
    - 2.6.4.1 David’s House (Volvo)
    - 2.6.4.2 David’s House (Car)
  - 2.6.5 5 Tips: Mental Models for Learnable Interfaces
    - 2.6.5.1 Headshot Studio
  - 2.6.6 Representations
    - 2.6.6.1 Tablet Studio
  - 2.6.7 Representations for Problem Solving 1
    - 2.6.7.1 Tablet Studio
    - 2.6.7.2 Exercise
    - 2.6.7.3 Tablet Studio
  - 2.6.8 Representations for Problem Solving 2
    - 2.6.8.1 Headshot Studio
    - 2.6.8.2 Exercise
    - 2.6.8.3 Headshot Studio
    - 2.6.8.4 Tablet Studio
  - 2.6.9 Characteristics of Good Representations
    - 2.6.9.1 Tablet Studio
  - 2.6.10 Design Challenge: Representations
    - 2.6.10.1 David’s House (Basement)
    - 2.6.10.2 Exercise
    - 2.6.10.3 David’s House (Basement)

2.6.11 Representations in Interfaces
2.6.11.1 Tablet Studio
2.6.12 Metaphors and Analogies
2.6.12.1 Tablet Studio
2.6.13 Exploring HCI: Metaphors and Analogies
2.6.13.1 Headshot Studio
2.6.14 Design Principles Revisited
2.6.14.1 Headshot Studio
2.6.15 New Functionality Meets Old Interfaces
2.6.15.1 Tablet Studio
2.6.16 Learning Curves
2.6.16.1 Tablet Studio
2.6.17 User Error: Slips and Mistakes
2.6.17.1 Tablet Studio
2.6.18 Exercise: Slips vs. Mistakes
2.6.18.1 Headshot Studio (Morgan)
2.6.18.2 Exercise
2.6.18.3 Headshot Studio (Morgan)
2.6.19 Learned Helplessness
2.6.19.1 Tablet Studio
2.6.20 Learned Helplessness and Education
2.6.20.1 David's House (Playroom)
2.6.21 Expert Blind Spot
2.6.21.1 Headshot Studio
2.6.21.2 Exercise
2.6.21.3 Headshot Studio
2.6.22 Reflections: Learned Helplessness and Expert Blindspot
2.6.22.1 David's House (Kitchen)
2.6.22.2 Exercise
2.6.22.3 David's House (Kitchen)
2.6.23 Conclusion
2.6.23.1 Headshot Studio

Meta-Data

Lesson Goals

- Students will understand the nature of mental models in feedback cycles.

- Students will understand the relationship between representations and strong mental models.
- Students will understand the criteria for a good representation for forming mental models.
- Students will understand the relationship between learning curves and representations.
- Students will understand the concepts of learned helplessness, expert blindspot, slips, and mistakes.

Lesson Outcomes

- Students will be able to describe interfaces and interface design in terms of the mental models they leverage.
- Students will be able to design representations that lead to strong mental models.
- Students will be able to critique interfaces from the perspective of their relationship with the user's mental model.
- Students will be able to differentiate slips from mistakes and use both in the design process.
- Students will be able to identify instances of learned helplessness and expert blindspot.

Assessments

- Students will complete exercises demonstrating the power of different mental models to problem-solving.
- 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 first be introduced to the idea of mental models and their role in users' simulation and prediction of what happens in interfaces.

- Mental models will be used to contextualize the role of representations: helping the user form strong mental models.
- Representations will then be used to set up several concepts regarding users learning to use new interfaces: learned helplessness, expert blindspot, and user error.

Script

2.6.1 Introduction

2.6.1.1 Headshot Studio

- [C] David talking
- [A] Clips of the lesson
- [B] Lesson; Mental Models and Representations
- Today, we're going to talk about mental models and representations.
- [B] Definition; Mental Model: An internal, simulatable understanding of external reality.
- A **mental model** is the understanding you hold in your head about the world around you.
- Simulating a mental model allows you to make predictions and figure out how to achieve your goals in the world.
- So, a good interface will give the user a good mental model of the system it presents.
- [B] Definition; Representations: Internal symbols for an external reality.
- In order to develop good mental models, we need to give users good **representations** of the system with which they're interacting.
- That way, we can help users learn how to use our interfaces as quickly as possible.
- So, that's what we'll talk about in this lesson: creating representations that help users develop accurate mental models of our systems.
- [B] Topic; Mental models
- We'll start by talking about **mental models** in general and how they apply to interfaces with which we're familiar.
- [B] Topic; Representations
- Then, we'll talk about how **representations** can make problem-solving easier or harder.
- [B] Topic; Metaphors and analogies
- After that, we'll talk about how **metaphors** and analogies can be used to create good representations that lead to accurate mental models.
- [B] Topic; User error: slips and mistakes

- Then, we'll discuss how **user** error can arise either from inaccuracies or mistakes in the user's mental model, or from accidental slips despite an accurate mental model.
- [B] Topic; Learned helplessness
- [B] Topic; Expert blindspot
- Finally, we'll close by discussing **learned** helplessness, one of the repercussions of poor interface design, as well as **expert** blindspot, which is one of the reasons poor design can occur.

2.6.2 Mental Models

2.6.2.1 Headshot Studio

- [C] David talking, with basketball
- [B] Definition; Mental Model: An internal, simulatable understanding of external reality.
- A **mental model** is a person's understanding of the way something in the real world works.
- It's an understanding of the processes, relationships, and connections in real systems.
- Using mental models, we generate expectations or predictions about the world, and then check whether the outcomes match our mental model.
- So, I'm holding this basketball because generally, we all probably a model of what will happen if I bounce this ball.
- <David bounces the ball>
- It comes back up.
- You didn't have to see it come up to know what would happen.
- You use your mental model of the world to simulate the event, and then use that mental simulation to make predictions.
- When reality doesn't match our mental model…
- <David bounces the ball, but it doesn't come back up>
- It makes us uncomfortable.
- We want to know why our mental model was wrong. Maybe it makes us curious.
- But when it happens over and over, it can frustrate us. It can make us feel that we just don't and will never understand.
- As interface designers, this presents us with a lot of challenges.
- We want to make sure that the user's mental model of our systems matches the way our systems actually work.
- [B] Designing systems that act the way the user expects them to act
- [B] Designing systems that teach the user how they react

- We can do this in two primary ways: by designing systems that **act** the way people already expect them to act, and by designing systems that by their design **teach** people how they will act.
- That way, we can minimize the discomfort that comes from systems acting in ways that users don't expect.
- <Ball comes back into David's hands… somehow>

2.6.3 Mental Models and Education

2.6.3.1 Headshot Studio

- [C] David talking
- Mental models are not a uniquely HCI principle.
- In fact, if you search for mental models online, you'll probably find just as much discussion of them in the context of education as in HCI.
- And that's a very useful analogy to keep in mind.
- When you're designing an interface, you're playing very much the role of an educator.
- Your goal is to teach your user how the system works through the design of the interface.
- But unlike a teacher, you don't generally have the benefit of being able to stand here and explain things to your user.
- Most users don't watch tutorials or read documentation.
- You have to design interfaces that teach users while they're using them.
- That's where representations will come in. Good representations show the user exactly how the system actually works.
- It's an enormous challenge, but also incredibly satisfying when done well.

2.6.4 Mental Models in Action

2.6.4.1 David's House (Volvo)

- [C] David in the Volvo
- Let's check out mental models in action by comparing the climate control systems between my 27-year-old Volvo 240 and my 1-year-old Nissan Leaf.
- So generally speaking, my model of my climate control system is that it has two variables to control: air temperature and fan speed.
- I can make the air hotter or colder, and I can make it come out faster or slower.
- [A] Cut to the climate control area

- So, it's a hot day: how would you make the air temperature colder and the air come out faster?
- <pause>
- The natural thought is to turn the fan up here and the air temperature down over here.
- But this doesn't actually make the temperature and colder, it just disables the heat.
- This dial over here has to be turned on to make the air conditioning actually work.
- So, to make it colder, you have to both slide this lever to the left, and turn this dial to… the red area. Yes, the red area indicates maximum coldness.
- This also means you can turn on both the heat and the AC at the same time, and have neither of them blowing out.
- None of these controls really match my mental model of how this system works, and the colors used here do nothing to correct my mental model.
- [C] Back to David
- So, the Volvo 240 was a great car, but it had a lot to learn in terms of usability in its interface.

2.6.4.2 David's House (Car)

- [C] David in the Leaf
- 26 years later, have we learned to do this better? Definitely.
- [A] Show controls
- In this car, I can simply turn on the heat or AC, use this dial to control the fan speed, and use this dial to control just how hot or cold it is.
- The numbers here don't actually mean much to me. I just know to turn it up if I'm too cold or down if I'm too warm.
- It also has an automatic feature that will take control of the fan speed to try to bring the car's temperature as a whole in line with what I want.
- Note, though, that this interface does still have some issues.
- For example, the number indicated here has different meanings based on whether I'm in auto mode or manual mode.
- In one it's my desired car temperature, in the other it's my desired incoming air temperature -- subtly different things.
- Or, at least, I think that's what it represents. I could be wrong, in which case the interface still has issues, just different issues.
- More importantly, though, these buttons don't always behave as expected.
- The control system as whole needs to be on for the fans to run, and yet if I press to turn the AC off, sometimes it turns the system as a whole off. Turning it back on turns the AC back on.
- [C] Back to David

- But, that said, this system matches my mental model of how climate control works much more closely than the Volvo system.

2.6.5 5 Tips: Mental Models for Learnable Interfaces

2.6.5.1 Headshot Studio

- [C] David talking
- Matching our interface design to users' mental models is a valuable way to create interfaces that are easily learnable by users.
- Here are five tips, or principles, to leverage for creating learnable interfaces. These principles of learnability were proposed by Dix, Finlay, Abowd, and Beale in their book Human-Computer Interaction.
- [B] 1. Predictability.
- **1. Predictability.** Looking at an action, can the user predict what will happen? For example, graying out a button is a good way to help the user predict that clicking that button will do nothing.
- [B] 2. Synthesizability.
- **2. Synthesizability.** Not only should the user be able to predict the effects of an action before they perform it, they should also be able to see the sequence of actions that led to the current state. This can be difficult in graphical user interfaces, but the log of actions through the undo menu can make it easier. Command lines make this easier, actually, as they give a log of the commands in order.
- [B] 3. Familiarity.
- **3. Familiarity.** This is similar to Norman's principles of affordances. The interface should leverage actions with which the user is already familiar from real-world experience. For example, if you're trying to indicate something is either good or bad, you'd likely want to use red and green rather than blue and yellow.
- [B] 4. Generalizability.
- **4. Generalizability.** Similar to familiarity and to Norman's principle of consistency, knowledge of one user interface should be generalize to others. If your interface has tasks that are similar to other interfaces' tasks, like saving, copying, and pasting, it should perform those tasks in the same way.
- [B] 5. Consistency.
- **5. Consistency.** Slightly different than Norman's principle of consistency, this means that similar tasks or operations within a single interface should behave similarly. For example, you wouldn't want to have Ctrl+X cut text if text is selected, but close the application if no text is selected. Its behavior should be consistent.

- Using these principles can help the user leverage their existing mental models of other designs, as well as develop a mental model of your interface as quickly as possible.

2.6.6 Representations

2.6.6.1 Tablet Studio

- [V] Bad text-heavy furniture assembly plans
- The most powerful tool in our arsenal to help ensure users have effective mental models of our system is representation.
- We get to choose how things are visualized to users, and so we get to choose some of how their mental model develops.
- Using good representations can make all the difference between effective and ineffective mental models.
- So, to take an example, let's look at some instructions for assembling things.
- <walk through bad example>
- [V] Completed furniture
- This representation of how this fits together leads to little to no mental model of how the finished product will actually look.
- [V] Good assembly plans
- This, on the other hand, is a good plan.
- I can look at that arrangement and get a feel for how the pieces are meant to fit together.
- This representation helps me understand the problem in a way the other did not.

2.6.7 Representations for Problem Solving 1

2.6.7.1 Tablet Studio

- [V] A "hill" appears with a hiker at the bottom
- A good representation for a problem makes the solution self-evident.
- Let's take a classic example of this.
- [V] Text of problem appears: "A hiker starts climbing a mountain at 7:00 am and takes his usual path, arriving at the top at 7:00 pm. The following morning he starts from the mountaintop at 7:00 am and returns to the base following the same route, arriving at 7:00 pm. Is there guaranteed to be a point on this path where the monk is at the exact same point at the exact same time on both days?"
- [V] Monk animates while reading the problem
- <read problem>

2.6.7.2 Exercise

- [E] “Is there guaranteed to be a point on this path where the monk is at the exact same point at the exact same time on both days?”
- [E] Yes
- [E] No

2.6.7.3 Tablet Studio

- [V] Replay animation
- Described that way, it might seem odd that there’s a point where the monk is at the same place at the same time both days.
- But what if we tweak the representation a little?
- Instead of one monk on two days, what about two monks on one day?
- If we represent the problem that way, we see…
- [V] Animation
- That of course the answer is yes because they have to cross paths at some point.
- Changing the representation made the problem far easier to solve.

2.6.8 Representations for Problem Solving 2

2.6.8.1 Headshot Studio

- [C] David talking
- For simple problems, identifying a good representation can be easy, but what about for more complex problems?
- For those problems, we need some examples of what makes a good representation.
- So, let’s try a complex example.
- We’ll use a problem you, might be familiar with.
- For now, I’ll call it the circles and squares problem.
- On one side of the table, I have three circles and three squares.
- My goal is to move the three circles and three squares to the other side of the table.
- I can only move two shapes at a time, and the direction of the moves must alternate, starting with a move to the right.
- The number of squares on either side can never outnumber the number of circles, *unless* there are no circles at all on that side.
- How can you accomplish this?
- Try it out, and enter the number of moves it takes in the box -- or, just skip if you give up.

2.6.8.2 Exercise

- [E] How many moves did it take?
- [E] (box for answer)

2.6.8.3 Headshot Studio

- [C] David talking
- If you answered that problem, well done.
- If you skipped it, I don't blame you.
- It's not an easy problem.
- But, it's even harder when the representation is so poor.
- There were lots of weaknesses in that representation.
- Let's step through how we would improve it.

2.6.8.4 Tablet Studio

- The first thing we could do is simply write the problem out.
- Audio is a poor representation of complex problems.
- [V] Text appears of the problem
- Simply being able to see the rules of the problem would probably help a lot with solving it.
- But we can still do better than this.
- [V] Visual representation comes up, changing with the following description
- Instead, we can represent the problem visually.
- Here we have the shapes, and we can imagine actually moving them back and forth around the frame.
- The arrow in the center indicates the direction of the next move.
- But we can still do better. Right now, we have to work to compare the number of squares to the number of circles.
- So, let's line up the shapes, so we can directly visibly compare.
- Now, the only remaining problem is that we have to keep in working memory the rule about more squares than circles.
- There's no natural reason why we need more squares than circles, it's just an arbitrary rule.
- So, let's make it more self-evident.
- Let's make the squares wolves and the circles sheep.
- As long as the sheep outnumber the wolves, they can defend themselves.
- But if the wolves ever outnumber the sheep, they'll eat them.
- But if there are no sheep, there's nothing for the wolves to eat, so that's okay.

- So, now, we have a new representation of the problem, one that will make the problem much easier to solve. The rules are more obvious and easy to evaluate.
- We can finally visualize this by showing the movements between multiple states. This way, we can clearly see that for any state, there's a finite number of follow-up states, and we can explore that tree.

2.6.9 Characteristics of Good Representations

2.6.9.1 Tablet Studio

- [V] Two states of final wolf-and-sheep representation
- [V] Characteristics appearing on other side as we go through them
- What are the characteristics of a good representation?
- [B] Makes relationships explicit
- First, good representations **make** relationships explicit.
- Laying these out this way makes it easy to tell whether there are more sheep or wolves.
- [B] Bring objects and relationships together
- Second, good representations **bring** objects and relationships together.
- We're visualizing the objects, but in a way that exposes the relationships among them.
- [B] Excludes extraneous details
- Third, a good representation **excludes** extraneous details.
- For example, sometimes this problem is described in terms of a river and a boat, but those details actually aren't relevant to solving the problem, so they're absent from here.
- [B] Expose natural constraints
- Fourth, good representations **expose** natural constraints.
- We describe these as sheep and wolves because it makes it easier to think in terms of the rule that sheep must outnumber wolves.
- That isn't the best rule, of course, because we know that sheep can't actually defend themselves against wolves.
- [B] David; You might recognize this example from Knowledge-Based AI...
- However, if we visualized these as **guards** and prisoners instead, for example, it involves holding in working memory that the prisoners inexplicably won't run away.
- So, I think the wolves and sheep metaphor is better.
- However, perhaps the original name of the problem is even better: the cannibals and missionaries problem. It makes more sense that a missionary can defend himself against a cannibal than that a sheep can defend itself from a wolf.
- But that's a little dark, so we'll stick with our wolves and sheep.

2.6.10 Design Challenge: Representations

2.6.10.1 David's House (Basement)

- [C] David by the circuit breaker
- So, let's take an example of redesigning a representation to create a better mapping with the task.
- Here we have my circuit breaker.
- On the left, we have a list of the breakers, and what they control.
- On the right, we have the breakers themselves.
- To reset a breaker, I need to go down this list on the left, find the right breaker, count down on the right, and switch it.
- How can we make this representation of what each breaker corresponds to better?

2.6.10.2 Exercise

- [E] How can we make this representation better?
- [E] (box for answer)

2.6.10.3 David's House (Basement)

- There are a number of things we can do here.
- The simplest change we could make would be to make these breakers writeable.
- Instead of writing a list on the left to match to the right, we could write what each breaker corresponds to directly on it.
- But then I still have to manually scan through.
- We could further augment this by having a floorplan over here on the side with numbers written for each breaker area.
- That way, instead of having to surf through the list, I can jump straight to the area of the house I want to reset.
- Then, that can send me over to the right breaker here, where the name is written to confirm my selection.
- Now if we wanted to get really crazy, we could actually lay out the breakers themselves to correspond to the different areas of the house.
- Or, we could put the breakers in the actual rooms.
- But there, we're starting to encounter some of the other constraints on the problem, so it's probably best to stick to what we can control without requiring that the physical device be built differently in the first place.

2.6.11 Representations in Interfaces

2.6.11.1 Tablet Studio

- [T] Desktop
- Representations are all around us in the real world, but they play a huge role in interfaces as well.
- Designing representations of the system state is actually one of the most common tasks you might perform as an interface designer.
- Let's take a look at a few.
- [T] Bring up Google Calendar
- Here we have Google Calendar. This is a representation of my week.
- Notice how it uses space to represent blocks of free time.
- <switch to alternative visualization>
- An alternative visualization just lists the appointments, which struggles to represent the actual pace and structure of my day.
- <select someone else's calendar>
- Clicking on some other individuals' calendars allows me to visualize what they're up to as well, easily equipping me to visually find times for us to meet.
- Similarly, this representation helps me easily detect conflicts in my schedule.
- [T] Bring up Powerpoint from previous slide
- Here's another example of this. The PowerPoint animation pane is a representation of when animations will take place. The length of the bars represent how long the animation will take, and the placement indicates their relative timing. Here, for example, we see that all these shapes fade in together, that this animation takes longer than this one, and that this animation waits until after this one is complete.
- And these are just two of the many, many representations you use whenever you use a computer.
- Scrollbars, for example, are representations of your relative position in a document.
- Highlighting markers are representations of what is currently selected.
- All of these representations work together to help your mental model of the state of the system.
- Representations, when used correctly, can make many tasks trivial or even invisible, and we as interface designers have a lot of control over representations in our designs.

2.6.12 Metaphors and Analogies

2.6.12.1 Tablet Studio

- [T] Previous comparison of WSJ paper vs. web site
- Analogies and metaphors are powerful tools for helping users understanding your interface.
- If you can ground your interface in something they already know, you can get a solid start in teaching them how to use your interface.
- For example, the Wall Street Journal's web site heavily leverages an analogy to the Wall Street Journal print edition.
- Headlines, grids, text all appear similarly, so someone familiar with the print version doesn't have a lot to learn about the web version.
- If you've ever tried to explain a tool you use to someone that's never seen it, you've probably encountered this.
- [T] Switch to Slack
- For example, both at Udacity and in the Georgia Tech OMSCS program, we use Slack, for communicating with each other.
- If you've never used Slack, it's a chat app for organizations to talk in different public and private rooms.
- Listen to what I just said: it's a chat app. In my description, I leveraged an analogy to something you've already seen.
- Slack is a pretty easy example because it *is* a chat app, but what about something harder?
- [T] Switch to Medium
- How about Medium?
- Medium is a writing platform that's kind of like a blogging service, but kind of also like a publishing service, but kind of also like a news feed.
- You write articles kind of like WordPress blog posts, but you can publish them through organizations, kind of like traditional newspapers or news aggregators like the Huffington Post.
- Then, articles are published to interested people more like a newsfeed, like Facebook or Twitter.
- Notice that my entire explanation of Medium was based on analogies to other services like WordPress, Huffington Post, and Facebook.
- Analogies and metaphors have a downside, though.
- When you choose to use them, users don't know where the analogy ends.

- When I describe Medium's newsfeed as kind of like Facebook and Twitter, users might wonder where the 'retweet' or 'share' options are.
- So while analogies are powerful ways to help users understand our interfaces, we also have to pay attention to what misconceptions they might introduce.

2.6.13 Exploring HCI: Metaphors and Analogies

2.6.13.1 Headshot Studio

- [C] David talking, holding smartphone
- One of the challenges encountered by every new technology is helping the user understand how to use it.
- Smartphones may be pretty ubiquitous by now, but we're still figuring out some elements of how to best use these things.
- Typing efficiency on a touch screen still hasn't caught up to efficiency with a full keyboard.
- But, typing on a phone was also a pretty straightforward transition from a regular keyboard because the onscreen keyboard was designed as an analogy to a real keyboard.
- There are probably more efficient ways to enter text into a phone, but they wouldn't be as easily learnable as this straightforward analogy to a physical keyboard.
- This illustrates both the positive and negative sides of using analogies in design: analogies make the interface more learnable, but they also risk restricting the interface to outdated requirements or constraints.
- So take a moment and think about how this applies to your chosen area of HCI.
- If you're looking at things like gestural interfaces, virtual reality, touch-based interfaces, or other types of emerging technologies, what analogies can you draw to other interfaces to make your designs more learnable?
- At the same time, what do you risk by using those analogies?

2.6.14 Design Principles Revisited

2.6.14.1 Headshot Studio

- [C] David talking
- [A] Visuals from consistency morsel of design principles lesson
- In our lesson on design principles, we touched on a number of principles that are relevant to these ideas of mental models, representations, and metaphors.

- First, the idea that people reason by analogy to past interfaces or metaphor to the real world is one of the reasons the principle of consistency is so important.
- We want to be consistent with the analogies and metaphors that people use make sense of our interfaces.
- [A] Visuals from the affordances morsel of design principles lesson
- Second, when we say that an interface should teach the user how the system works, we're echoing the idea of affordances.
- The way the system looks should echo how it's used.
- Just by observing the system, the user should be actively learning how to interact with it.
- [A] Visuals from the mapping morsel
- Third, representations are important because they map the interface to the task at hand.
- A good representation is one that users can use to predict the outcomes of certain actions.
- In other words, a good representation lets users predict the mapping between their actions in the interface and outcomes in the world.

2.6.15 New Functionality Meets Old Interfaces

2.6.15.1 Headshot Studio

- [V] Clips of the thermostat comparison from earlier
- In designing interfaces, we want to leverage analogies to the real world and principles from past interfaces wherever possible to help the user learn the new interface as quickly as possible.
- But there's a challenge here.
- Why are we designing technology if we're not providing users anything new?
- It's one thing to take the technology they're already using and make it more usable, but generally, we also want to enable people to do things they've never done before.
- That means there are no analogies, no expectations, no prior experiences to leverage.
- How do you tell someone that's used to controlling their thermostat that they don't need to anymore?
- So while we need to leverage analogies and prior experiences whenever possible, we also need to be award that eventually, if we're doing something interesting, they're going to break down.
- Eventually, we're going to have to teach the user to use the unique elements of our interface.

2.6.16 Learning Curves

2.6.16.1 Tablet Studio

- [V] Learning curve axis: expertise on the vertical, experience on the horizontal
- Every interface requires the user to do some learning to understand how to use it.
- Very often, we visualize these as learning curves. A learning curve plots expertise against experience.
- Generally, as the user gains more experience, they also gain more expertise.
- Here, our user starts with no experience, and ends with an experience above this line.
- However, the shape and steepness of this curve can vary.
- [V] True steep learning curve appears
- Ideally, we want a learning curve that grows quickly in expertise with little experience.
- This is a steep learning curve, although typically you'll hear 'steep' used in the opposite sense. Technically, 'steep' is good, but it is often used to mean difficult.
- That's because 'steep' calls to mind connotations of high difficulty. So, 'steep' is actually a poor representation of this concept.
- Instead, let's call this a rapid learning curve, meaning that our expertise grows rapidly with experience.
- [V] Slow learning curve appears
- Interfaces that are more difficult to use would have slower learning curves.
- The user needs a lot more experience to reach the same level of expertise.
- [V] Lines disappear
- So how do we reach proficiency faster?
- For one, if we're consistent with existing conventions and use analogies that users understand, we can actually start them off with some initial expertise.
- [V] Dot for initial expertise rises along the vertical axis
- For example, when you download a new smartphone app, you know that three horizontal lines probably indicate a menu.
- The design of the interface gives you some initial expertise.
- From there, we want to make the ascension as rapid as possible.
- [V] Line appears
- One way we can do that is by using representations and affordances that help the user immediately understand how to use the interface.
- So, good design is about making a user achieve expertise as quickly as possible, either through starting them off higher or lifting them up quicker.

2.6.17 User Error: Slips and Mistakes

2.6.17.1 Tablet Studio

- [V] ‘Slip’ and ‘Mistake’ appear on screen
- As we design interfaces, we will no doubt encounter instances where the user makes mistakes.
- Sometimes this might be because our users are stressed, busy, or distracted, and other times it might be because our users don’t understand our interfaces or even their own goals.
- [B] Elsewhere in HCI: Constraints from Lesson 2.5: Design Principles and Heuristics are one way of preventing errors.
- As designers, though, we know that there’s no such thing as user error: any user error is a failure of the interface to properly **guide** the user to the right action.
- In designing interfaces, there are two kinds of user error that we’re interested in avoiding.
- The first are slips.
- [V] Slips definition: “The user has the right mental model, but does the wrong thing anyway.”
- Slips occur when the user has the right mental model, but does the wrong thing anyway.
- [V] ‘Saved’ prompt box comes up
- Take this box prompting a user closing a program on whether to save their work.
- In all likelihood, the user knows what they want to do, and it’s probably to save their work.
- If you asked them to explain what they should do, they would say ‘Click yes’.
- Yet, notice that the order of these buttons is reversed from what you would expect -- a violation of the consistency principle.
- Notice also that the ‘No’ button appears to be selected by default.
- This is a slip waiting to happen.
- The user knows what they want to do, but there’s a high likelihood they’ll do the wrong thing.
- [V] Mistake definition: “The user has the wrong mental model, and does the wrong thing as a result.”
- A mistake, on the other hand, happens when the user’s very model is wrong.
- [V] ‘Revert’ prompt box comes up
- Take this prompt, for example.
- The user is asked if they want to revert to the original file.

- That's really just a backwards of asking whether they want to 'save', but that's foreign terminology for many users.
- Their mental model of what saving means doesn't tell them what to do in this instance.
- [B] Elsewhere in HCI: This is a violation of the tolerance principle from Lesson 2.5: Design Principles and Heuristics.
- What's more, they have no choice here: without a cancel button, they're forced to choose, knowing one option could mean **losing** their changes.
- Here, the problem is a mismatch between their internal model and the way the system is working -- or at least, the way it describes itself.
- Norman further divides slips into two categories.
- [V] Action-based comes up
- Action-based slips are places where the user performs the wrong action, or performs the right action on the wrong object, even though they knew the correct action. They might click the wrong button or right click when they should left click.
- [V] Memory lapse comes up
- Memory lapse slips are places where the user forgets something they knew to do. Forgetting to start a timer or initiate logging might be examples of memory lapse slips.
- [V] Dialog appears
- In this dialog, clicking 'no' when you mean 'yes' would be an action-based slip.
- This dialog's existence prevents memory lapse slips: it corrects for times when you forget to save before closing.
- Norman also divides mistakes into three categories.
- [V] Rule-Based appears
- The first is rule-based, where the user correctly assesses the state of the world but makes the wrong decision based on it.
- [V] Knowledge-Based appears
- The second is knowledge-based, where the user incorrectly assesses the state of the world in the first place.
- [V] Memory lapse appears
- The third is memory lapse, similar to slips, but here this is about failing to fully execute a plan because of distractions or other memory lapses.
- [V] Dialog appears
- If the user clicks the wrong button on this dialog, it could be for different reasons.
- Maybe they correctly knew that they wanted to save their changes, but didn't realize that clicking 'no' would save. That would be a rule-based mistake.
- Or, perhaps they didn't realize they wanted to save in the first place. That would be a knowledge-based mistake.
- If they were to shut down their computer and never come back and answer this dialog in the first place, that might be considered a memory lapse mistake.

- In our designs, we want to do everything we can to prevent all these kinds of errors.
- [B] Elsewhere in HCI: The consistency principle from Lesson 2.5: Design Principles and Heuristics.
- We want to help prevent routine errors by leveraging **consistent** practices.
- [B] Elsewhere in HCI: Lesson 2.4 on Cognitive Load and Lesson 2.8 on Distributed Cognition.
- We want to let our interface **offload** some of the demands on working memory from the user to avoid memory lapse errors.
- And we want to leverage good representations to help users develop the right mental models to minimize rule-based and knowledge-based errors.
- [B] Elsewhere in HCI: The tolerance principle from Lesson 2.5: Design Principles and Heuristics.
- And while errors are inevitable, we should make sure to leverage the **tolerance** principle to make sure their repercussions can never be too bad.

2.6.18 Exercise: Slips vs. Mistakes

2.6.18.1 Headshot Studio (Morgan)

- [C] Morgan on her phone, David narrating
- When you're looking to improve an interface, user errors are powerful places to start.
- They're indicative either of weaknesses in the user's mental model, or places where the system isn't capturing the user's correct mental model.
- So, let's try to address an error Morgan is encountering.
- Morgan usually chats with her boyfriend on {whatever hip chat app you kids are using nowadays}, but she chats with some other people, too.
- But she finds she's often sending the wrong messages to the wrong people.
- The app by default brings up the last-open conversation, and usually that's her boyfriend -- but sometimes it's someone else, and she accidentally messages them instead.
- First, is this a slip or is this a mistake?

2.6.18.2 Exercise

- [E] Is this a slip or is this a mistake?
- [E] A. Slip
- [E] B. Mistake

2.6.18.3 Headshot Studio (Morgan)

- [C] Morgan on her phone, David narrating
- I would argue this is a slip.
- Morgan knows who she means to message, but the phone's behavior tricks her into sending things to the wrong people.
- What's more, this might be either an action-based slip or a memory lapse slip. Maybe Morgan is tapping the wrong person, or maybe she's forgetting to check who she's messaging.
- So, take a second and brainstorm a design for this app that can prevent this from happening in the future, without overcomplicating the interaction too much.
- <pause>
- I would argue the best way to do this would be to simply show more pervasive reminders of who Morgan is messaging.
- We could show the recipient's picture on the send button, for example.
- That way, the interaction is no more complex, but Morgan also has to directly acknowledge who she's messaging to send a message.

2.6.19 Learned Helplessness

2.6.19.1 Tablet Studio

- [V] Feedback cycle diagram
- The feedback cycle on HCI is reliant on a relationship between the user's input and the interface's output.
- The idea of this cycle is that the user learns from the output what they should input.
- If they encounter an error, they receive feedback on how to avoid it next time.
- If they do something correctly, they see that the goal was accomplished.
- This is the principle of feedback.
- [V] Some kind of break appears between the destination of the input arrow and the source of the output arrow, on the right
- But what happens when there is no discernible interaction between the input and the output?
- What happens when the user acts in the system over and over and over again, but never receives input that helps them? Never even receives input that makes them think they're making progress?
- That's when something called learned helplessness sets in.
- [V] Replace human with a sad human.

- The human working with the interface learns that they're helpless to actually use the system.
- If you've ever tried to explain to someone how to use a system only to find them very resistant, this is likely a major reason why.
- They've learned, through the poor feedback given by the interface, that they're helpless in the context of that system.
- That's a very difficult hurdle to overcome.

2.6.20 Learned Helplessness and Education

2.6.20.1 David's House (Playroom)

- [C] David with Lucy
- Just like mental models, learned helplessness is also a topic related as much to education as it is to HCI.
- If you've ever spent anytime in a teaching role, you've very likely encountered students that are very resistant to being taught, and the reason is that they've learned that no matter what they do, they never succeed.
- They've learned to be helpless based on their past experiences.
- In all likelihood, you've also been in situations where you've been the one learning that you're helpless.
- In fact, if you're a parent, I can almost guarantee you've been in that situation.
- There are times when your child was crying, inconsolable, and you had no clue why.
- Nothing you did helped.
- And you learned that you were helpless to figure out what your child wanted.
- So if you're a parent and you're dealing with learned helplessness as an interface designer, just imagine that you are the user and the interface is your screaming child.
- What feedback would you need from your child to figure out how to help them, and how can you build that kind of feedback into your interface?

2.6.21 Expert Blind Spot

2.6.21.1 Headshot Studio

- [C] David talking, with a basketball
- Generally, when we're developing interfaces, we're going to be experts in the domain.
- It's rare that you design an interface to help people do something you don't know how to do yourself.
- But as a result, there's risk for something called expert blind spot.

- When you're an expert in something, there are parts of the task that you do subconsciously, without even thinking about them.
- For example, a professional basketball player naturally knows where to place her hands on the ball when taking a shot.
- I know exactly what to do when I walk into the studio.
- Amanda knows exactly what to do when she gets behind the camera.
- And yet if we were suddenly asked to train someone else, there are lots of things we'd forget to say, or lots of things we would think would be obvious.
- And that's exactly what you're doing when you're designing an interface.
- You're teaching the user how to use what you designed.
- You're teaching them without the benefit of actually talking to them, explaining things to them, or demonstrating things for them.
- You're teaching them through the design of your interface.
- So you have to make sure that you don't assume they're an expert, too.
- You have to overcome that expert blind spot because we are not our users.
- We are not the user.
- That could be the motto of all of HCI: I am not my user.
- Say it with me: I am not my user.
- One more time: I are not my user.
- Now type it.

2.6.21.2 Exercise

- [E] "Write 'I am not my user'."
- [E] (box to write it)

2.6.21.3 Headshot Studio

- [C] David talking
- Now, write it on a post-it note and stick it to your monitor.
- If you wear glasses, write it on the inside of the lens.
- Record yourself saying it on your phone and set that as your ringtone.
- Do whatever you have to do to remember: I am not my user.

2.6.22 Reflections: Learned Helplessness and Expert Blindspot

2.6.22.1 David's House (Kitchen)

- [C] David talking

- In order for us to really sympathize with users suffering from the effects of learned helplessness and expert blindspot, it's important for us to understand what it's like to be in that position.
- We've all experienced these things at some point in life, although at the time we might not have understood what was happening.
- So, take a second and reflect on a time when you experienced learned helplessness and the effects of expert blindspot from someone trying to teach you.
- It might be in a class, it might be learning a new skill, or it might be in doing something that everyone else seems to accomplish day to day just fine.

2.6.22.2 Exercise

- "Click to continue" exercise

2.6.22.3 David's House (Kitchen)

- [C] David talking
- The fact that I'm filming this in the kitchen probably tells you where I experience this.
- Anything related to cooking, I feel completely helpless.
- I've given myself food poisoning with undercooked meat lots of times.
- I once forgot to put the cheese on a grilled cheese sandwich.
- I accidentally made toast, and it wasn't even good toast.
- And I've always heard it's so easy: just follow the recipe!
- But no! It's not that easy because many recipes are written for experts.
- For example, here's a recipe.
- It calls for a medium saucepan.
- Is this a medium saucepan?
- I have no idea.
- 1 egg beaten with a splash of water.
- A splash? Like a splash when you overfill a water bottle or a splash when your sibling soaks you at the pool?
- Pulse to combine.
- Cook until edges are golden brown.
- What's golden brown?
- Give me a color code and I'll compare it, but otherwise I don't know where on the spectrum from 'golden' to 'brown' golden brown lies.
- These are examples of places where the directions are given in a way that assumes I already have some expertise.

2.6.23 Conclusion

2.6.23.1 Headshot Studio

- [C] David talking
- [A] Clips of the lesson
- [B] Topic; Mental models
- In this lesson, we've talked about mental models.
- We discussed what mental models are and how users use them to make sense of a system.
- [B] Topic; Good representations
- We've discussed how good representations can help users achieve strong models.
- [B] Topic; User error
- We've talked about how issues with interfaces can lead to two different kinds of user error.
- [B] Topic; Learned helplessness
- We've discussed learned helplessness, the undesirable outcome of poor feedback on user errors.
- [B] Topic; Expert blindspot
- We've discussed expert blind spot, and the importance of understanding that you are not your own user.