ART 260

User Experience Design

Instructor

- Rich Rowan
- rrowan@digipen.edu
- Cell: 206-898-2955

Office Hours

- Tuesday 5pm-7:30pm
- Thursday 1pm-3:30pm

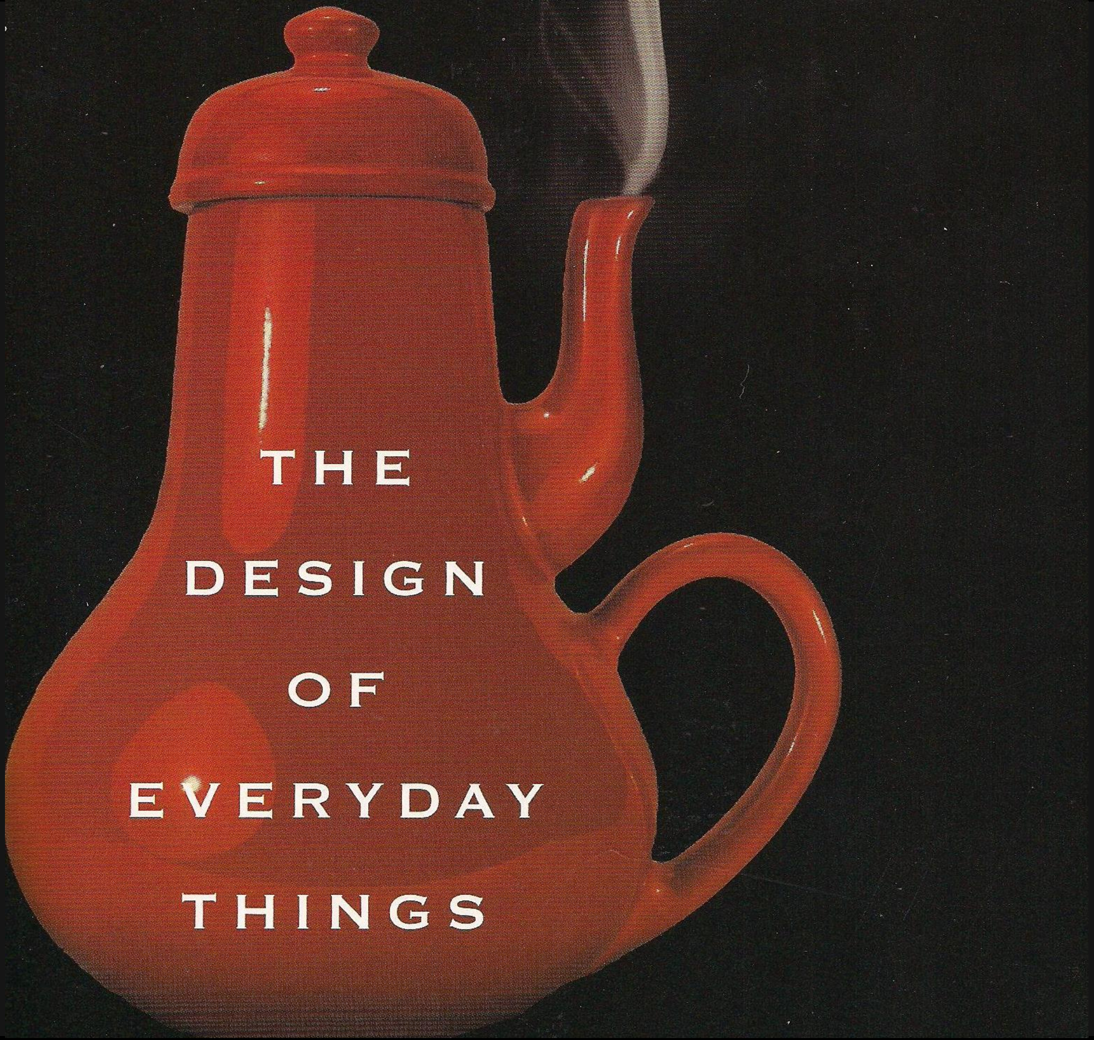
DigiPen INSTITUTE OF TECHNOLOGY

Core Principles of User Experience

- ❑ Core Principles of UX
- ❑ Conceptual Models
- ❑ Principles of Player Behavior

PLEASE SILENCE
ALL ELECTRONIC DEVICES

THANK YOU



CORE PRINCIPLES

Affordances

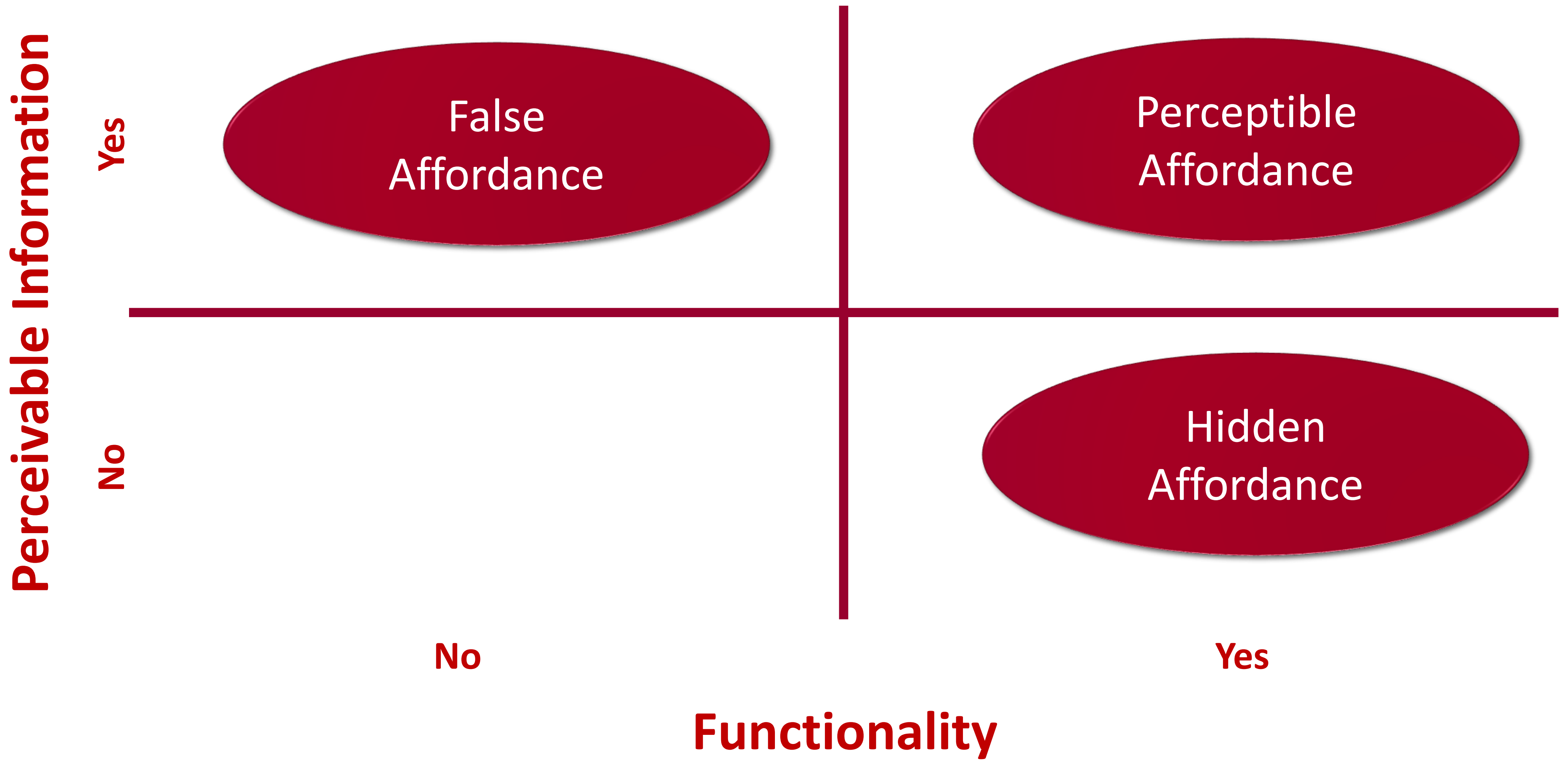
"[T]he term *affordance* refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used…" – Don Norman, *The Design of Everyday Things*

- The term was introduced by James J. Gibson, a psychologist, in the 1977 article *The Theory of Affordances* and elaborated on in *The Ecological Approach to Visual Perception.*
- Affordances give us clues about how something might work.
  - Plates are for pushing
  - Knobs are for turning
  - Slots are for inserting things into
  - Etc.

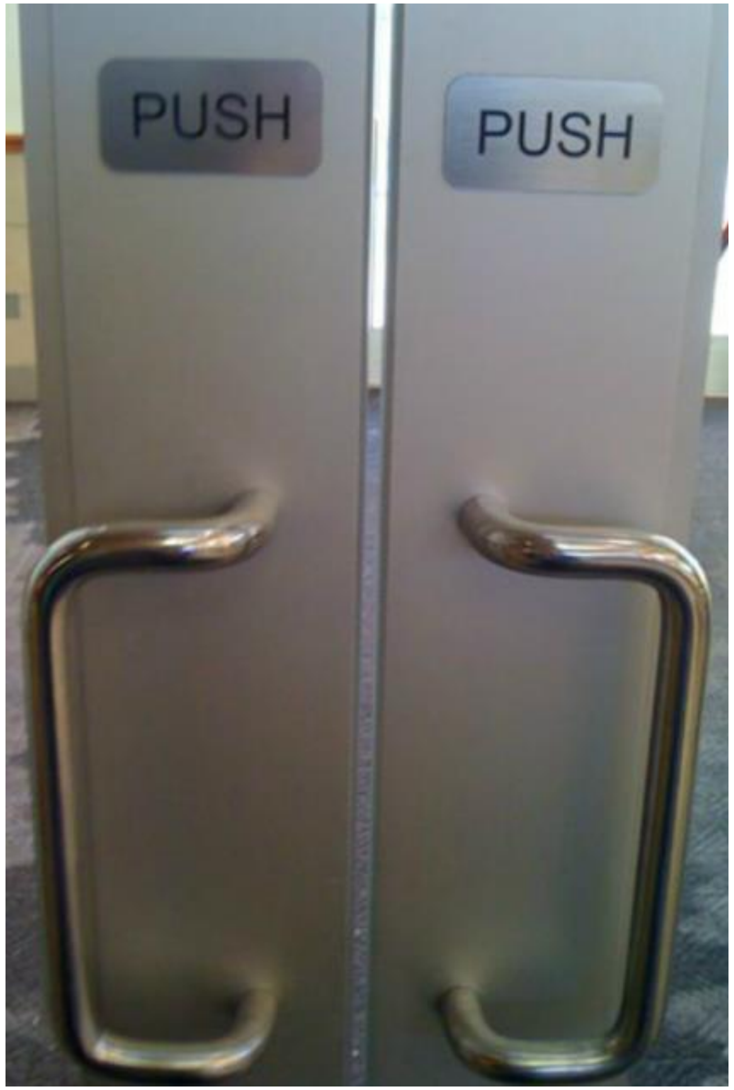
Affordance Types

- Perceptible
  - Information about an object that the player can perceive and then act upon.
- False
  - A perceptible affordance that does not have any function, e.g. a placebo button.
- Hidden
  - There are available functions but they are not perceived by the player.

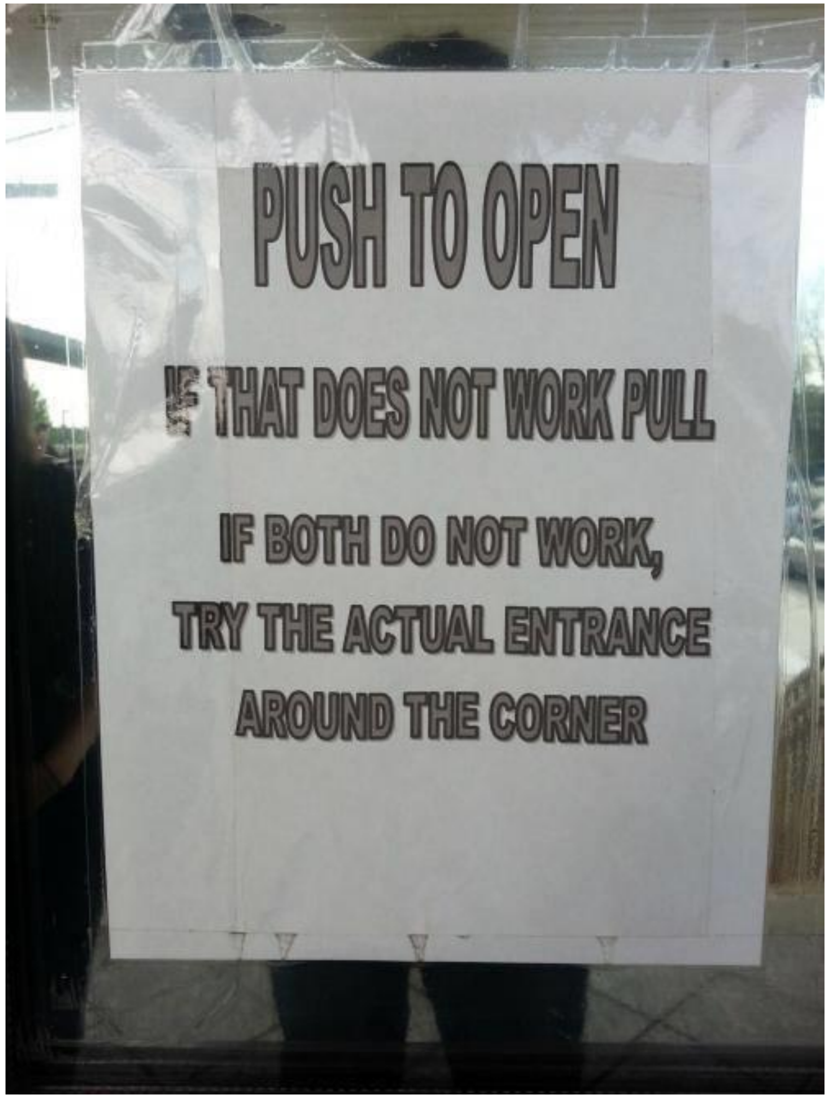
Affordance Types



False Affordance



False Affordance

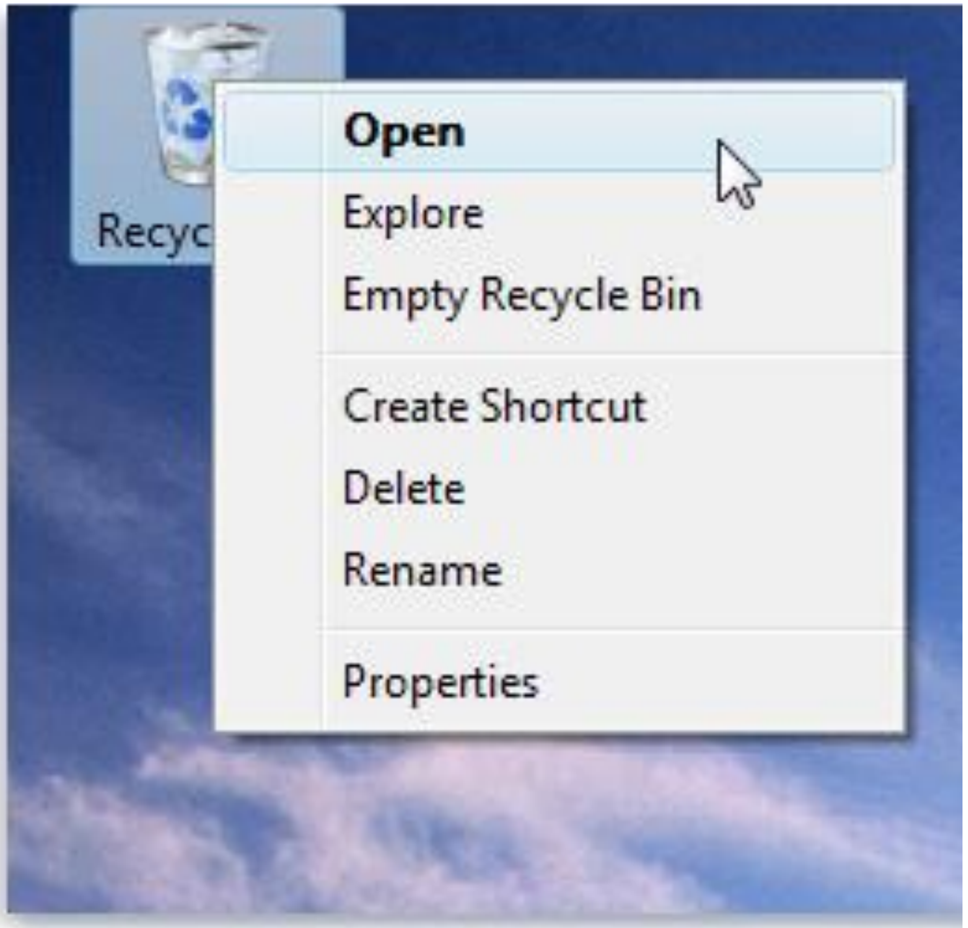


PUSH TO OPEN

IF THAT DOES NOT WORK PULL

IF BOTH DO NOT WORK,
TRY THE ACTUAL ENTRANCE
AROUND THE CORNER

Hidden Affordance



CLASS DISCUSSION

Affordances in Games

Common Game Affordances

- Button
- Slider
- Toggle/Checkbox
- Dropdown Menu
- Text Entry Fields

Visibility

"Visibility acts as a good reminder of what can be done and allows the control to specify how the action is to be performed. The good relationship between the placement of a control and what it does makes it easy to find the appropriate control for the task." – Don Norman, *The Design of Everyday Things*

- To make functionality obvious and apparent, you must make them visible to the player and in such a way that it bears an obvious relationship to the action.
- In a perfect world, all functions would have a visible control, but we can't make all controls visible in most games. This leads to the need for Information Architecture.

Mapping

"Mapping is a technical term meaning the relationship between two things, in this case between the controls and their movements and the results in the world." – Don Norman, *The Design of Everyday Things*

- Physical Analogies
  - Analog Sticks – push right to move right
  - Spatial mapping – control location corresponds to the physical location in the space
- Cultural Standards
  - Up means more, down means less
  - Reading direction – right to advance/continue, left to go back

Constraints

"The physical properties of objects constrain possible operations: the order in which parts can go together and the ways in which an object can be moved, picked up, or otherwise manipulated." – Don Norman, *The Design of Everyday Things*

- By providing good constraints, you limit the possible alternatives for interaction.
- Consider the slider…

Constraint Types

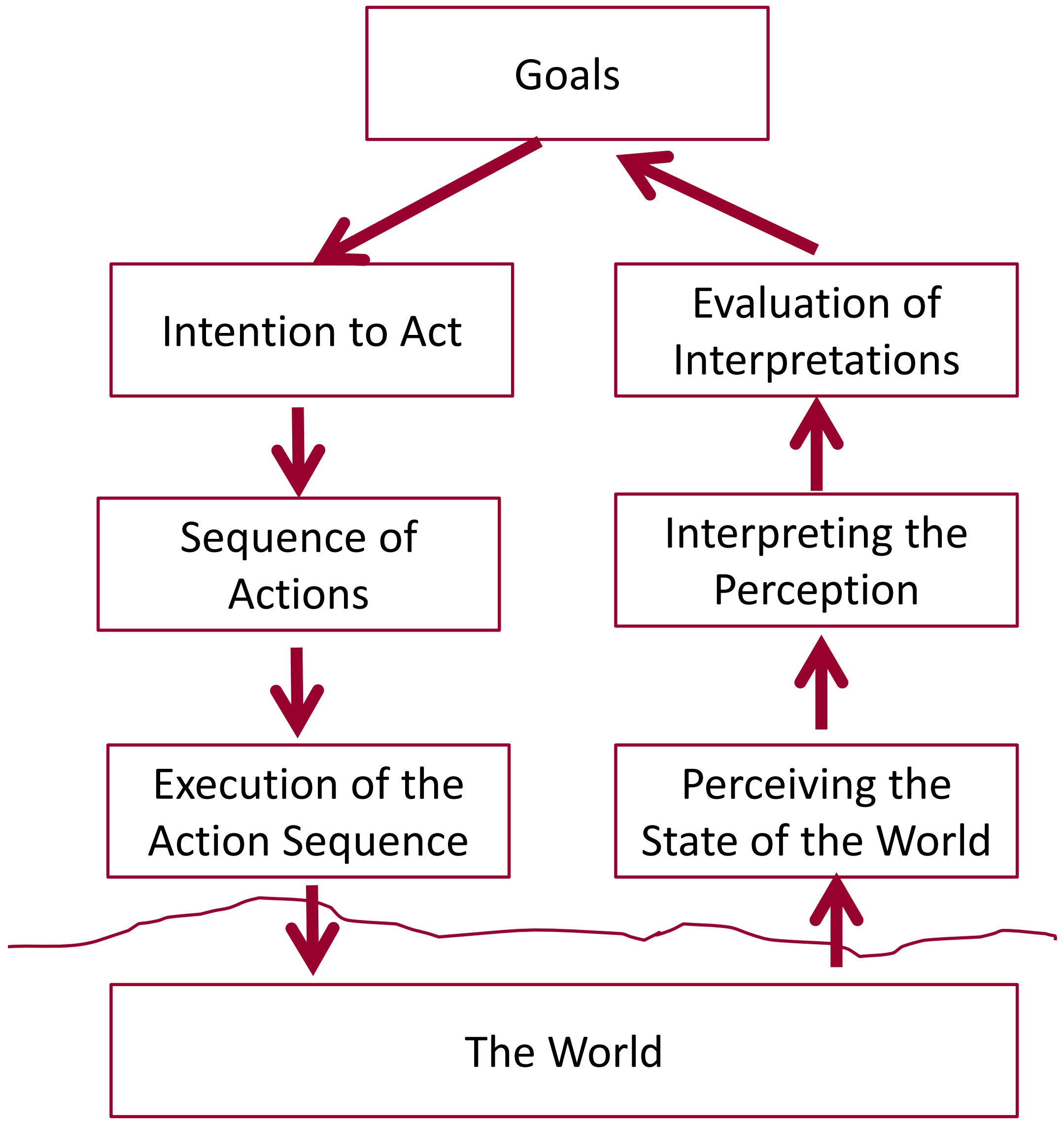
- Physical Constraints
  - The physical layout constrains inappropriate behavior.
  - Example: Modal dialogue covering the buttons that allow you to leave a UI state.
- Semantic Constraints
  - Relies on the meaning of the situation to control possible actions.
  - Example: When wielding a gun, a trigger is the logical method of control
- Cultural Constraints
  - Relies upon accepted cultural conventions. This is common in color meaning, where to look for controls, etc.
- Logical Constraints
  - This is what makes physical mappings work. If there is a switch next to a door, it is logical to assume that it controls that door.

Feedback

Feedback is the process of sending information back to the player about what action has actually been done or what result has been accomplished.

- Feedback has two primary outcomes:
    - Evocative – initiates an emotional “charge” associated with using an established neural pathway
    - Functional – establishes new neural pathways or reinforces neural pathways that can be used in the future

Feedback


Goals
Intention to Act
Sequence of Actions
Execution of the Action Sequence
Evaluation of Interpretations
Interpreting the Perception
Perceiving the State of the World
The World

Conceptual Models

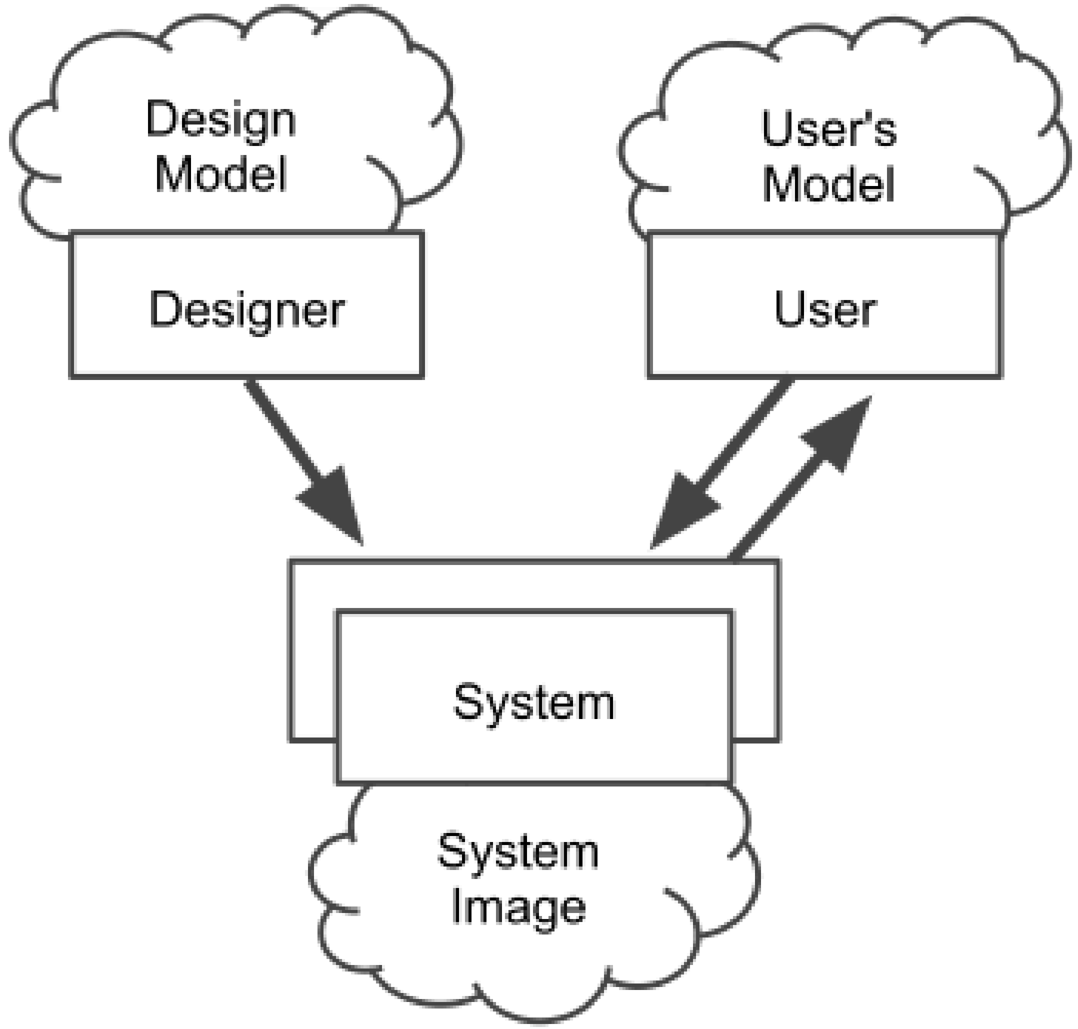
Transmitting a Conceptual Model from Designer to Player

Conceptual Models

"A good conceptual model allows us to predict the effects of our actions. Without a good model we operate by rote, blindly; we do operations as we were told to do them; we can't fully appreciate why, what effects to expect, or what to do if things go wrong." – Don Norman, *The Design of Everyday Things*

- People form conceptual models through experience, training, and instruction.
- The conceptual model of a device is formed largely by interpreting its perceived actions and its visible structure, called the system image.

System Image



Psychology of Causality

- If something happens immediately after taking an action, the action will appear to have caused that action.
- The psychology of causality results in many strange superstitions in technology.
- People will often ascribe failures to actions they just took, even if the failure is coincidental.
- The reverse is also true – if people take an action, they expect to see an effect of their action or they will most likely believe the action failed and attempt the action again, which could lead to an error state.

PRINCIPLES OF PLAYER BEHAVIOR

Safe Exploration

- When an interface keeps a player from experiencing negative consequences, they are more willing to explore and learn more of the interface because they feel safe doing so.
- Negative consequences don't even have to necessarily be dire, just annoying; for example, closing away undesired pop-up windows, re-entering form data, or even not being clear on how to return to the screen they were just viewing.
- If the interface doesn't feel safe, the player is less likely to explore, and may even quit playing.

Instant Gratification

- Players want to see immediate results from the actions they take.
- If you can reliably predict the first thing a new player is likely to do, you should design the UI to make that first thing stunningly easy.
- Don’t hide away introductory functionality behind anything that needs to be read or waited for, such as registrations, long sets of instructions, long scripted tutorials, long loading screens, etc.

Deferred Choices

- This follows from Instant Gratification – allow the player to defer choices that are not required. They may not know the answer at all, they may want to think about it and come back to it later, or they may be in a hurry.
- A few specifics:
  - Don’t accost the player with too many up-front choices before they understand the implication of those choices; example of “bad” is character generation in many of the Elder Scrolls games.
  - On forms, clearly mark required fields and minimize the number that are required.
  - Consider pushing some of the choices into secondary information to show a short list of the most common.
  - Use good defaults wherever possible, but even these have a cost.
  - Make it possible to return to deferred choices later by saving incomplete data or a statement telling a player where to find this again.
  - Defer registration until after experiencing the game.

Satisficing

- Term coined by social scientist Herbert Simon that describes the willingness to accept "good enough" instead of "best" if learning all the alternatives might cost time or effort.
- If an interface presents an obvious option or two that can be used immediately, the player is likely to try it, and with good design choices, that will often be the right choice.
- A few specifics:
  - Use "calls to action" in the interface to guide exploration
  - Make labels short, plainly worded, and fast to read – avoid game-specific jargon
  - Use the layout of the interface to communicate meaning, especially color, form, and icons
  - Make navigation simple and provide escape hatches
  - Keep the interface simple as possible to make it easier to identify the correct thing. Don't have multiple buttons that do closely related things.
- Once one method is learned, a player is unlikely to learn a new method.

Changes in Midstream

- Players will often change what they're doing in midstream, either through emergency (preemptive information), social interaction, remembering a previous goal, or formulating a new goal based on something in the interface.
- Provide opportunities for players to change what they're doing. Don't lock them into a choice poor environment unless there is a good reason to do so (these do exist).
- Also, make the task reentrant if possible; if the player leaves a task half finished, remember the parts they'd previously completed when they return to this task.

Habituation

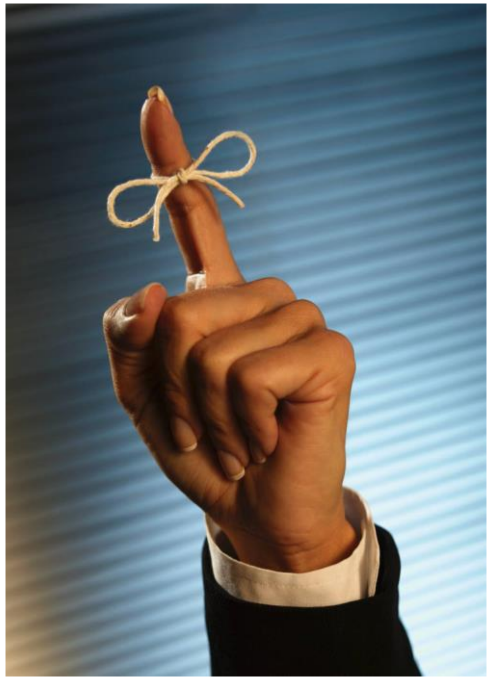
- Habituation describes the process of certain frequent physical actions to become reflexive, such as Ctrl-S for save functionality. These actions have become sub-conscious schemas, accessed nearly without thinking.
- Cross-Game Habits
  - This happens most frequently with genre conventions.
  - Be careful about fighting the player’s habits developed in other games.
- In-Game Habits
  - Be consistent in how behaviors work from UI screen to UI screen, or it will be a source of continual errors.
- Habituation describes why people routinely say OK to destructive confirmation dialog boxes – they’ve become habituated to closing dialog boxes because they think they know what they mean or because it is usually safe to do so.

Spatial Memory

- When people interact with an interface, they often find things they've used before by remembering where they found/placed it previously, not what they were named.
- This accounts for why people look in certain locations for standard interface elements like close buttons, ok buttons, etc.
- This is also why it is a generally bad idea to dynamically reorder menu lists to insert new items or optimize items. The tops and bottoms of lists and menus are also special locations, cognitively speaking. These are noticed and remembered more than items in the middle of a list.
- Try to keep all functions in the same location from screen to screen whenever possible.

Prospective Memory

- Prospective memory occurs when you plan to do something in the future and arrange some way of reminding yourself to do it; e.g. putting a book by the door to remind yourself to bring it to a friend.
- Consider ways to leave some artifacts around the UI that identify unfinished tasks.
- Consider ways for the player to add reminders of things to come back to later. “Continue Game” function is just one example.



Streamlined Repetition

- Repetitive tasks quickly become non-rewarding for players and they begin to avoid these tasks.
- Look for patterns of behavior that can be repetitive and consider optimizing these repetitive (i.e. boring) experiences.
- Examples include, “get all” command, group select and action, auto-loot features, switch gear sets, etc.
- Consider whether macros or simple scripting would help.

See You Next Class