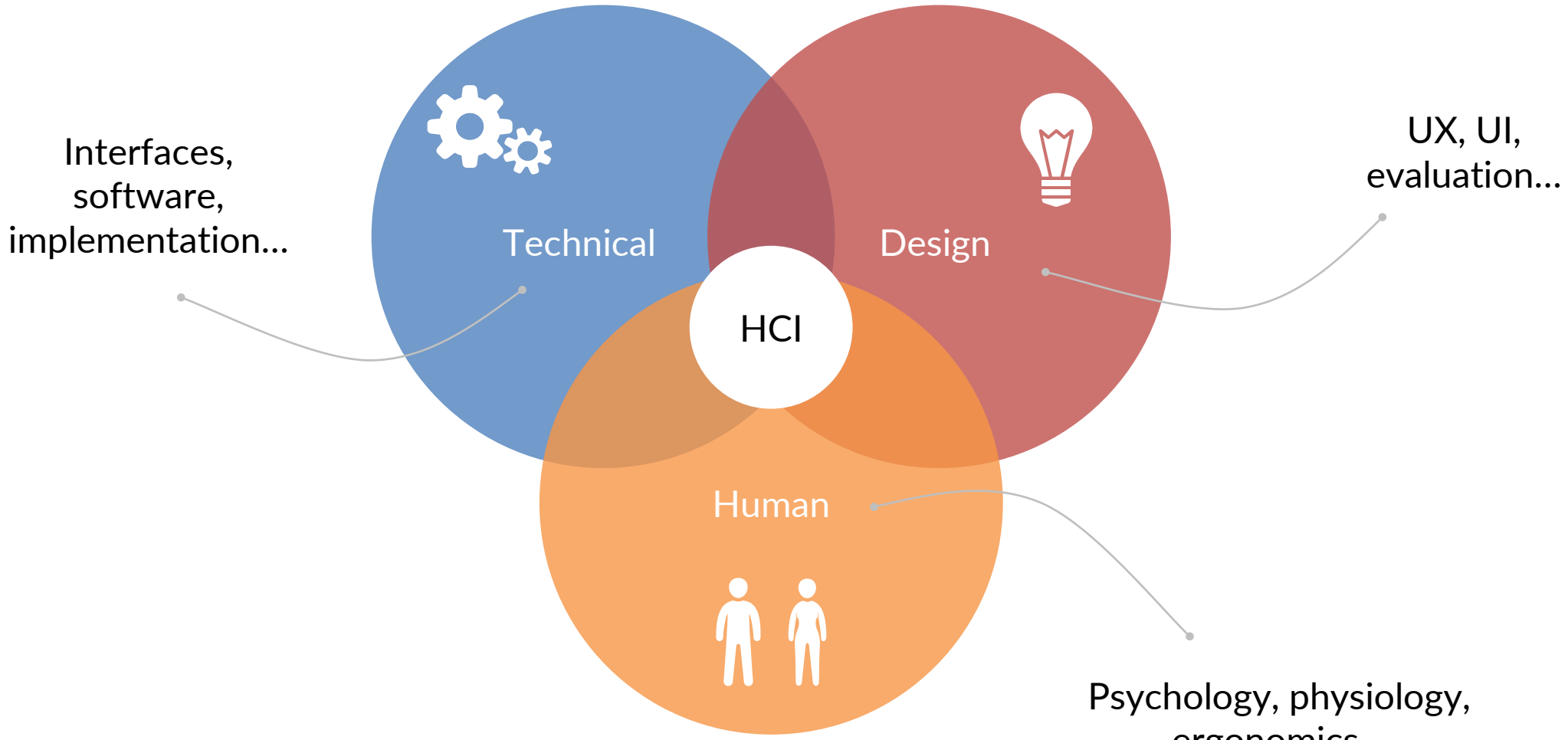
HUMAN-SYSTEMS INTERACTIONS

guillaume.bouyer@ensiie.fr
www.ensiie.fr/~bouyer/

V2.1e

Human-Computer Interactions



Objectives

Basic knowledge on how to design, program and evaluate HCIs for interactive projects

Practice

- Development (Unity)
- Search for information/inspiration
- Taking into account the needs/context of the user and the ideas/constraints of the designer

Concepts applicable for games, VR, AR and PFE

bouyer@ensiie.fr

http://www.ensiie.fr/~bouyer/JIN

Office 111 @ ENSIIE

Methodology and planning (cf. project document)

 Course on interfaces and associated features

 Development Gamepad / Unity (individual, graded)

 Analysis/report on existing 2D platform games (paired, graded)

 Search for information on platform game development (everyone)

 Development of the platform controller

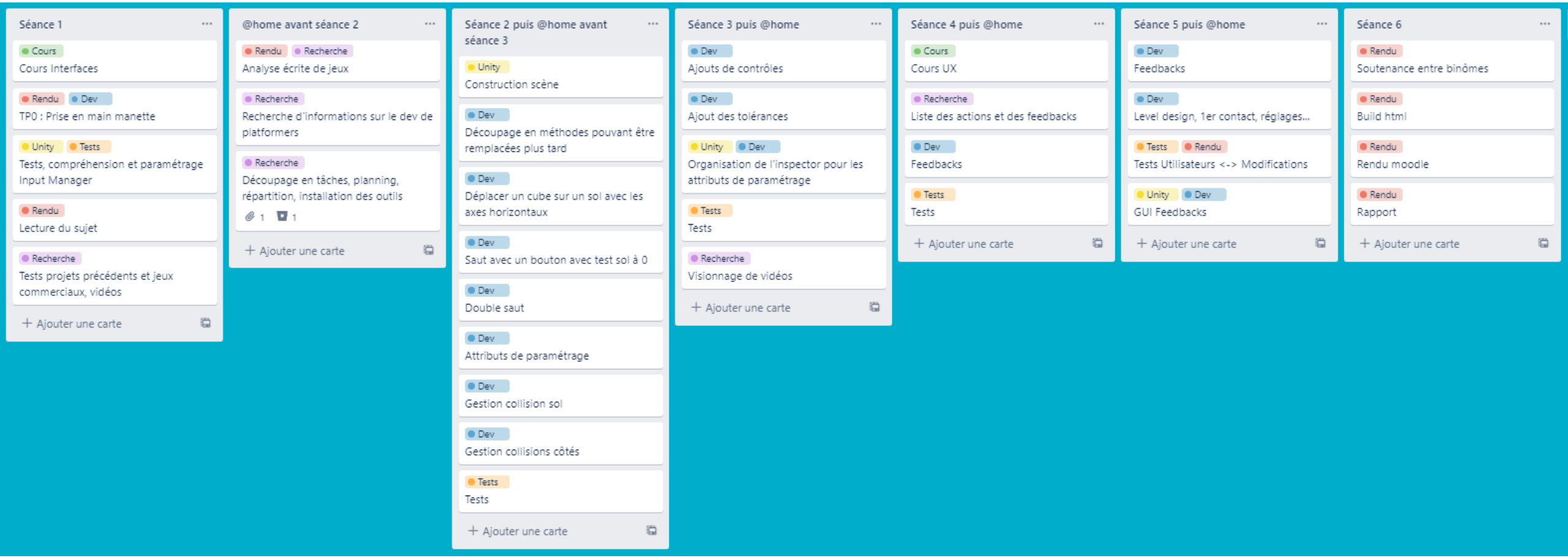
 Search for information (and videos) on "game feel" and feedback

 Course on UX and evaluations

 Development of feedback

 Testing and project finalization

Methodology and planning

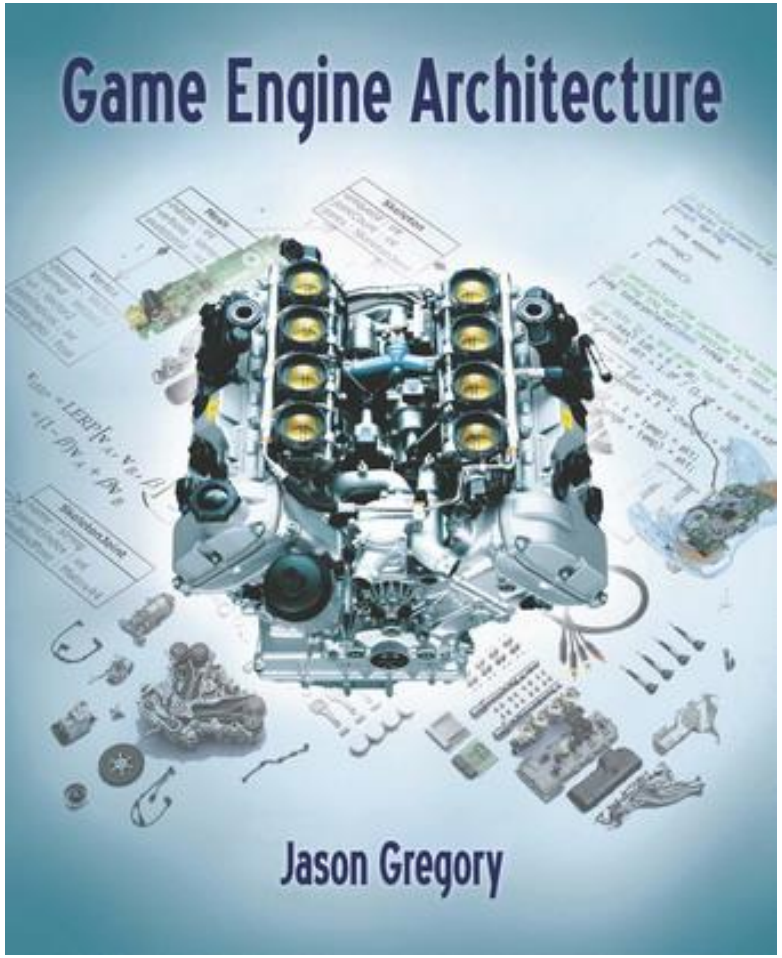


PART 1

HUMAN-GAMES INTERFACING

Reference

Game Engine Architecture, Jason Gregory, A K Peters/CRC Press, 2009 (http://www.gameenginebook.com/)



Types of devices

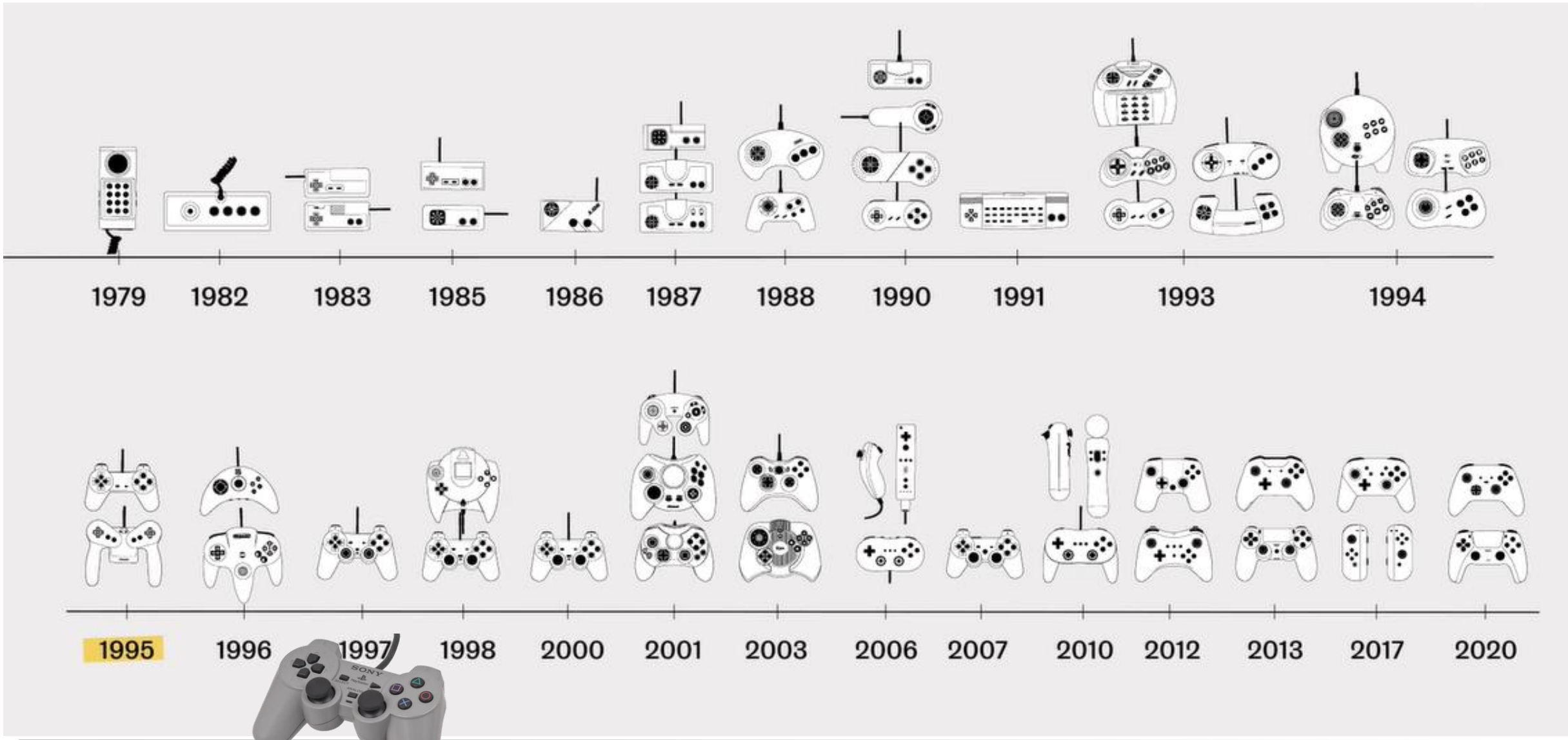
- Keyboard & mouse
- Joypad controller
- Gesture/3D controller
  - Wiimote, Kinect, PS Move, touch surface
- Hybrid controller
  - Wii U, PS Vita, 3DS, VR controller
- Built-in controllers
  - Arcade machines
- Specialized input devices and adapters
  - Music devices
  - Steering wheels
  - Dance pad

Various form factors and layouts

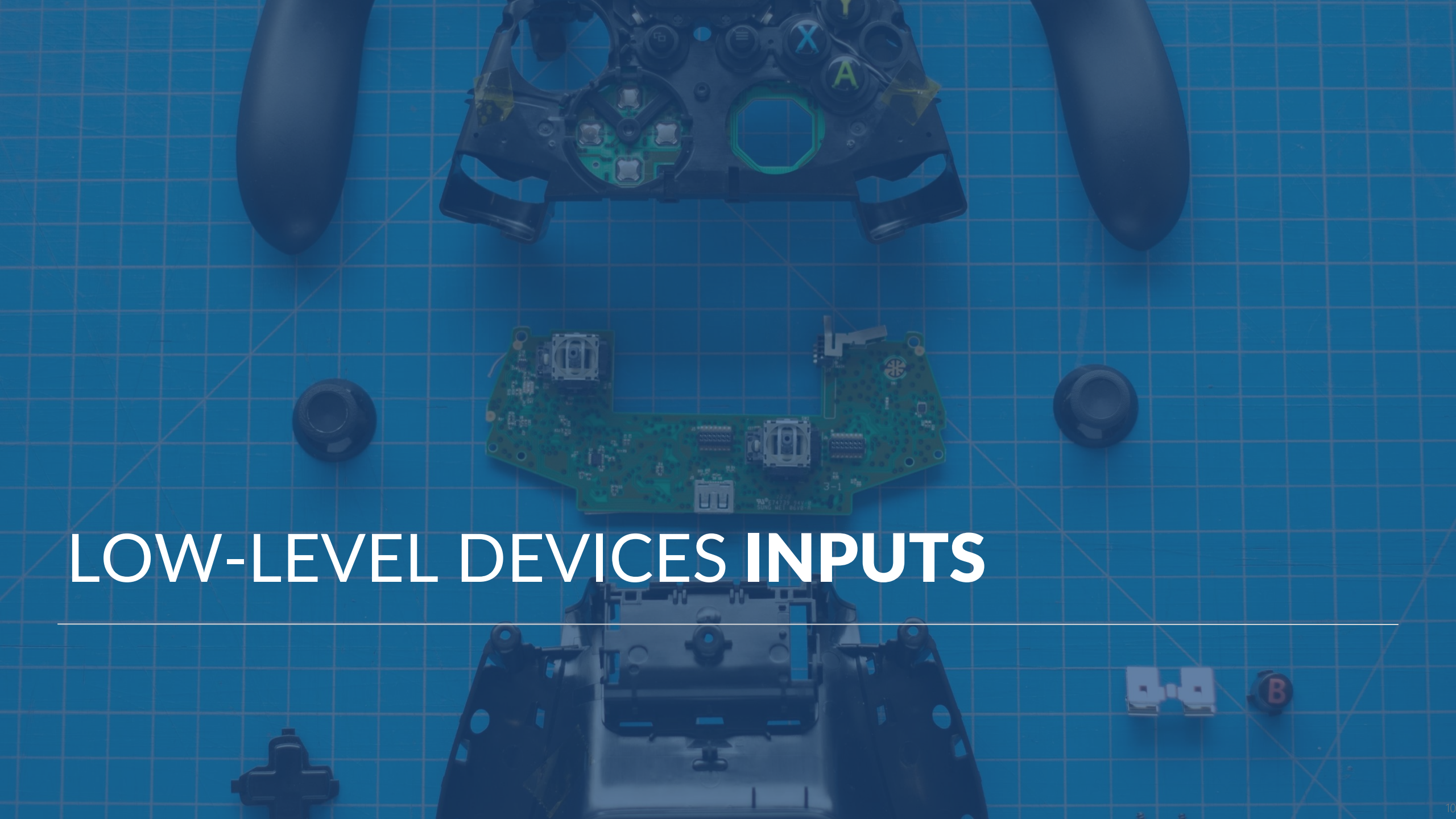
Common input types

Some kinds of outputs

Joypad Evolution



Source: Laurent Bolli, Joypads ! Le design des manettes



LOW-LEVEL DEVICES **INPUTS**

Autopsy of a Gamepad

Source: Evan Amos, Les consoles de jeux vidéo

Digital Buttons

2 states:

pressed = down

not pressed = up

(cf. closed or open electrical flow)

1 button usually represented by a bit

0 = up

1 = down

→ states of all of the buttons on a device packed into a single unsigned integer value

Digital Buttons: Microsoft ’s XInput API

Struct contains a variable wButtons that holds the state of all buttons

```
typedef struct _XINPUT_GAMEPAD {
    // 16-bit unsigned integer
    WORD wButtons;
    // 8-bit unsigned integer
    BYTE bLeftTrigger;
    BYTE bRightTrigger;
    // 16-bit signed integer
    SHORT sThumbLX;
    SHORT sThumbLY;
    SHORT sThumbRX;
    SHORT sThumbRY;
} XINPUT_GAMEPAD;
```

Bit mask defines which physical button corresponds to each bit in the word

```
#define XINPUT_GAMEPAD_DPAD_UP      0x0001 // bit 0
#define XINPUT_GAMEPAD_DPAD_DOWN    0x0002 // bit 1
#define XINPUT_GAMEPAD_DPAD_LEFT    0x0004 // bit 2
#define XINPUT_GAMEPAD_DPAD_RIGHT   0x0008 // bit 3
#define XINPUT_GAMEPAD_START        0x0010 // bit 4
#define XINPUT_GAMEPAD_BACK         0x0020 // bit 5
#define XINPUT_GAMEPAD_LEFT_THUMB   0x0040 // bit 6
#define XINPUT_GAMEPAD_RIGHT_THUMB  0x0080 // bit 7
#define XINPUT_GAMEPAD_LEFT_SHOULDER  0x0100 // bit 8
#define XINPUT_GAMEPAD_RIGHT_SHOULDER  0x0200 // bit 9
#define XINPUT_GAMEPAD_A            0x1000 // bit 12
#define XINPUT_GAMEPAD_B            0x2000 // bit 13
#define XINPUT_GAMEPAD_X            0x4000 // bit 14
#define XINPUT_GAMEPAD_Y            0x8000 // bit 15
```

An individual button’s state can be read by masking the wButtons word with the appropriate bit mask

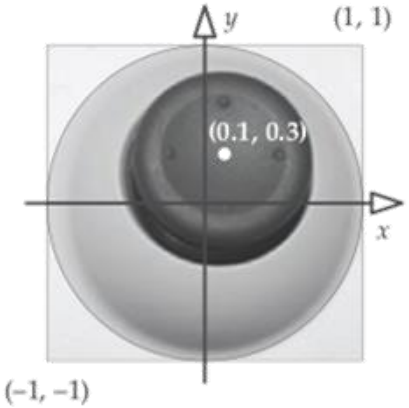
```
bool IsButtonADown(const XINPUT_GAMEPAD& pad){
    // Mask off all bits but bit 12 (the A button).
    return ((pad.wButtons & XINPUT_GAMEPAD_A) != 0);
}
```

Analog Axes and Buttons

Axes

Range of values

Used to represent the degree to which a trigger is pressed, or the 2D position of a joystick (two analog inputs x and y)



Buttons

Ex : Metal Gear Solid 2

Signals usually too noisy to be usable

Input signal usually digitized using integer or float

Analog Axes and Buttons: Microsoft ’s XInput API

16-bit signed integers for left and right thumb sticks

[-32768, 32767]

8-bit unsigned integers for left and right shoulder triggers

[0 , 255]

```
typedef struct _XINPUT_GAMEPAD {
    // 16-bit unsigned integer
    WORD wButtons;
    // 8-bit unsigned integer
    BYTE bLeftTrigger;
    BYTE bRightTrigger;
    // 16-bit signed integer
    SHORT sThumbLX;
    SHORT sThumbLY;
    SHORT sThumbRX;
    SHORT sThumbRY;
} XINPUT_GAMEPAD;
```

Relative Axes

- The position of an analog button, trigger, joystick, or thumb stick is absolute
  - Clear zero value
- For relative devices
  - No clear location at which the input value should be zero
  - Zero input value = the position of the device has not changed
  - Non-zero input values = delta value from last time
  - *Ex: mice, mouse wheels, track balls…*

Spatial Inputs

6D Pose

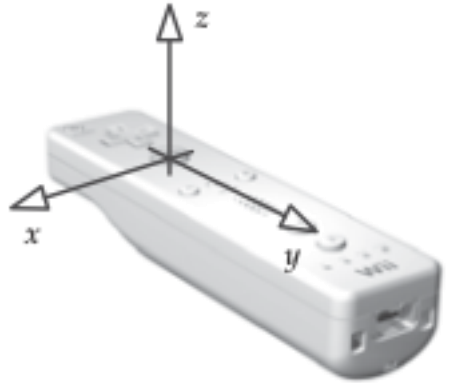
- VR Controllers and headsets
- Inside-out or Outside-in

3D Position

- IR Camera (Wiimote)
  - Location, size and distance of 2 fixed IR LEDs
- Camera (EyeToy)

3D Orientation

- 3 accelerometers along each axis
- Based on constant downward gravity



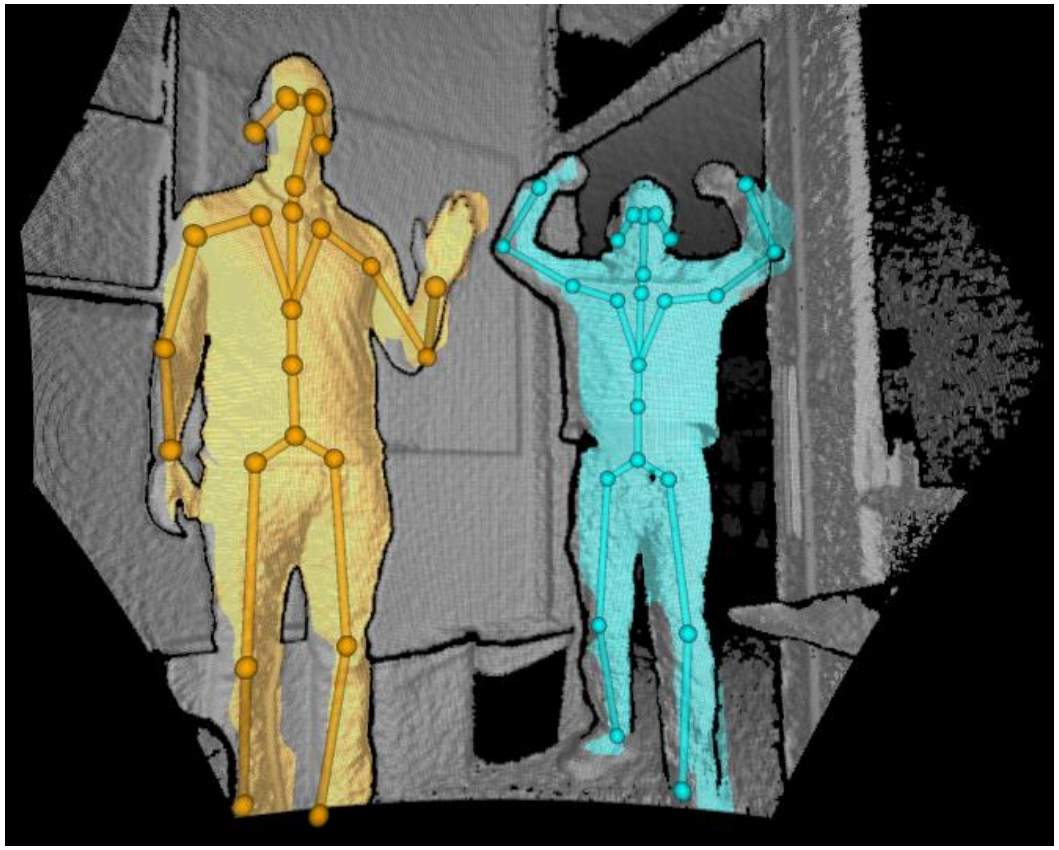
Spatial Inputs

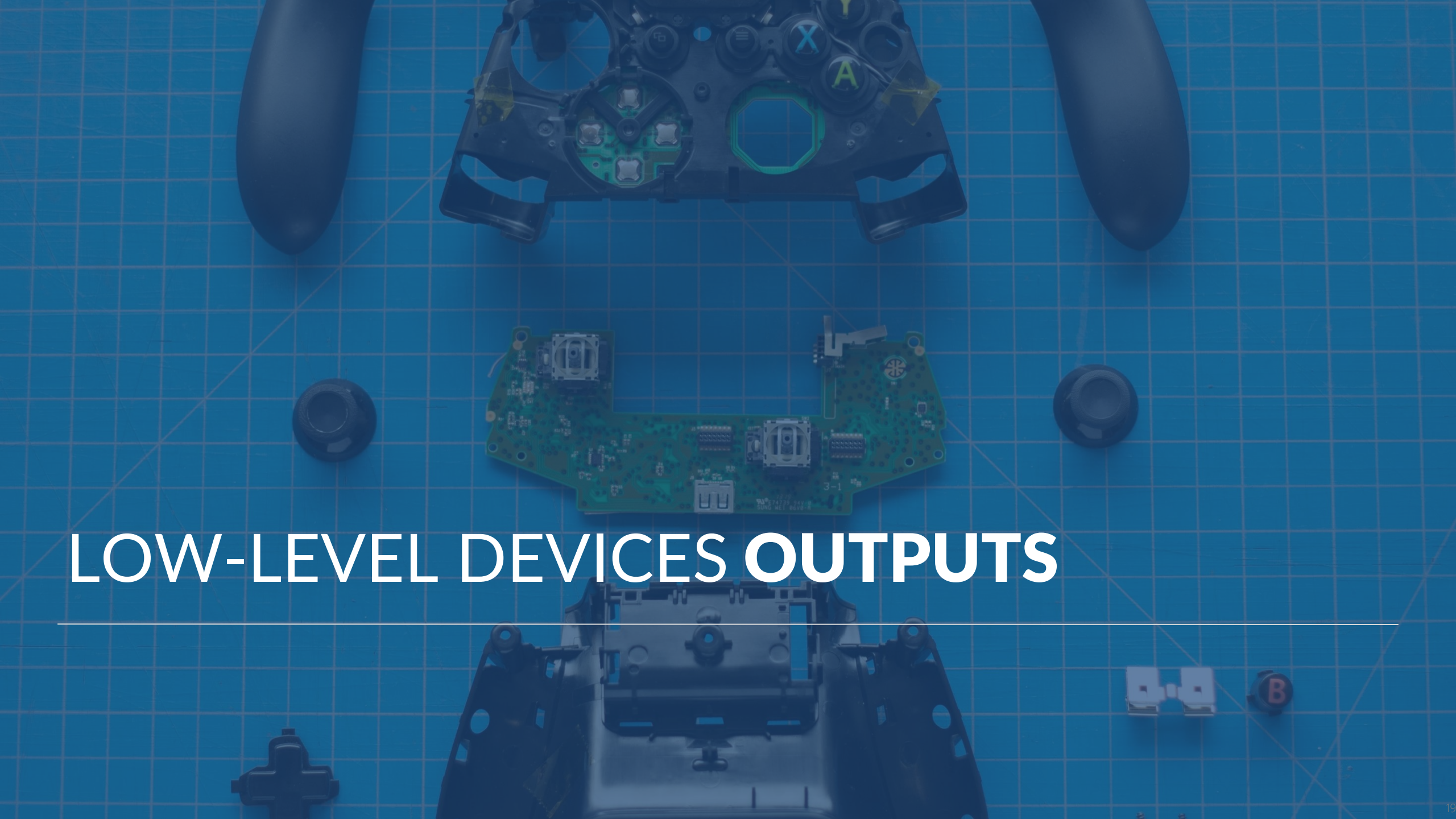
Touch

- Single or multiple contact points on screen or surface
- Movements

Body tracking

- Computer vision-based techniques
- Hand poses, body movements, facial expressions…





LOW-LEVEL DEVICES **OUTPUTS**

Rumble & Force-Feedback

Vibrations

One or more motors rotating a slightly unbalanced weight at various speeds

Game can control:

- motors on/off
- speeds -> different tactile effects

Force

Actuator(s) driven by a motor

Resist the motion of the player

Game can control:

- motors on/off
- strength and direction of the forces

Ex. arcade driving games: steering wheel resists the player's attempt to turn it, simulating difficult driving conditions or tight turns

Other Outputs

- Audio
  - Small speaker
  - Embedded USB audio I/O device
- Memory card slot on the pad (Dreamcast)
- LEDs
- Specific outputs for specific controller (music instruments…)

HIGH-LEVEL **INTERFACE SYSTEMS**

Game Engine Interface Systems

Inputs processing

- From raw data to smooth, pleasing, intuitive behaviors in-game
- Input management features

Level of abstraction

- Decouple raw inputs & logical game actions
- *Ex. button-mapping table*

Dead Zone

Analog axis => input values between predefined range I_{min} and I_{max}

Ideal = control not touched => steady and clear "undisturbed" value I_0

$I_0 = 0$

$I_0 = (I_{min} + I_{max})/2$ or $I_0 = I_{min}$

In practice:

- Noisy voltage produced by device
- No input fluctuates around I_0

Definition of a dead zone

- Any input values within the dead zone clamped to I_0
  - $[I_0 - \delta, I_0 + \delta]$ for a joystick
  - $[I_0, I_0 + \delta]$ for a trigger
- Wide enough to deal with the most noisy inputs generated by a normal device
- Small enough not to interfere with the responsiveness expected by the player

Analog Signal Filtering

A noise signal is usually of a high-frequency, relative to the signal produced by the player

Examples

- Discrete 1st order low-pass filter
  - Combine the current unfiltered input value with last frame's filtered input
  - $f(t) = (1 - a)\, f(t - \Delta t) + a.u(t)$
  - $with\ a = \frac{\Delta t}{RC + \Delta t}\ \ and\ RC\ constant$
- Average on n frames
  - Store the input data in a n-element circular buffer

Detecting Button Up and Down

Bit-wise operators to compare buttons' state bits between frames

- Previous XOR Current => 1 for changed buttons
- Result AND Current => button-down event
- Result AND NOT Current => button-up event

Detecting Chords

Group of buttons pressed at the same time

- Watch the states of the buttons
- Perform the requested operation when all of them are down

Problem 1: Ambiguities if chord includes a button assigned to an action

- Perform both actions ?
  - Game design such that a chord does all the actions of the individual buttons + additional action
- When detecting the individual button-presses, check that the other chord buttons are not down before action

Detecting Chords

Problem 2: Humans rarely press buttons in the same frame

- Delay between individual button-down event detection and action
  - If chord detected during this time it prevails
- Begin the single-button move immediately and preempt it by the chord
- Detect the chord when buttons pressed, but trigger effect when released

“Gesture”

Sequence of actions performed within maximum time-frame

Implemented with a history of actions

- 1st detected component
  - Add to buffer, with a time stamp
- Each subsequent detected component
  - Time difference -> if within the time-frame, add to buffer
- If entire sequence completed within the time-frame
  - Generate event
- If any non-valid intervening inputs detected, or if any component outside valid time window
  - Reset buffer

Robustness

- Managing multiple devices for multiple players
  - One-to-one mapping between controller index and player index
  - Assigning controllers to players when start
- Detecting low-battery conditions
  - Handled by game or OS
  - Unobtrusive message and/or sound effect
- Lost connection
  - *Ex: controller being unplugged or running out of batteries*
  - Usually pause gameplay, display a message, and wait for the controller to be reconnected
  - Multiplayer games
    - Suspend or temporarily remove the avatar corresponding to a removed controller
    - Allow the other players to continue playing the game
    - The removed/suspended avatar might reactivate when the controller is reconnected

Cross-Platform

- Platform-specific versions of all the code that requires device I/O, conditional compilation directives…
- Hardware abstraction layers
  - Translate between raw control ids on the current target hardware and abstract control ids
  - Use abstract controls according to their function in the game
  - Introduce higher-level functions that detect abstract gestures, with custom detection code on each platform

Input Re-Mapping

Many games allow to choose the controls

Ex: sense of the vertical axis for camera, predefined button mappings, full control of individual keys

Mapping

Table to maps each control index to a logical function

Only permit logic re-mappings, depending on the input data

Normalize all the inputs and group into classes

Context-Sensitive Controls

A single physical control can have different functions depending on context

- *“Use” button -> open, pick up…*
- *Modal control -> navigate and control camera, steer a vehicle…*

Problem: how to decide the state given the context

- *Ex: equidistance between 2 items*

State machine, priority system…

Lots of trial-and-error

Control Ownership

- Controls might be “owned” by different parts of the game
  - *Ex: Player control, camera control, menu system (pausing…)*
  - “Logical devices” each assigned to a system (camera, player…)
- Disabling Inputs
  - *Ex.: Disable all player controls during cinematic, disable free camera rotation when walking through a doorway*
  - Use a bit mask to disable individual controls on the input device
    - When needed, neutral or zero value returned instead of the actual value read

Interfacing in practice

Interactions are the basis of the player mechanics: correct and smooth handling is an important part of any good game

Deal with

- Variations between different input devices
- Filtering
- Command mappings
- Achieving the right “feel”
- Limitations from manufacturers (technical requirements checklists TRCs)

=> Devote significant time and engineering to a careful and complete implementation of the interface system

Unity

Input

http://docs.unity3d.com/ScriptReference/Input.html

Input Manager

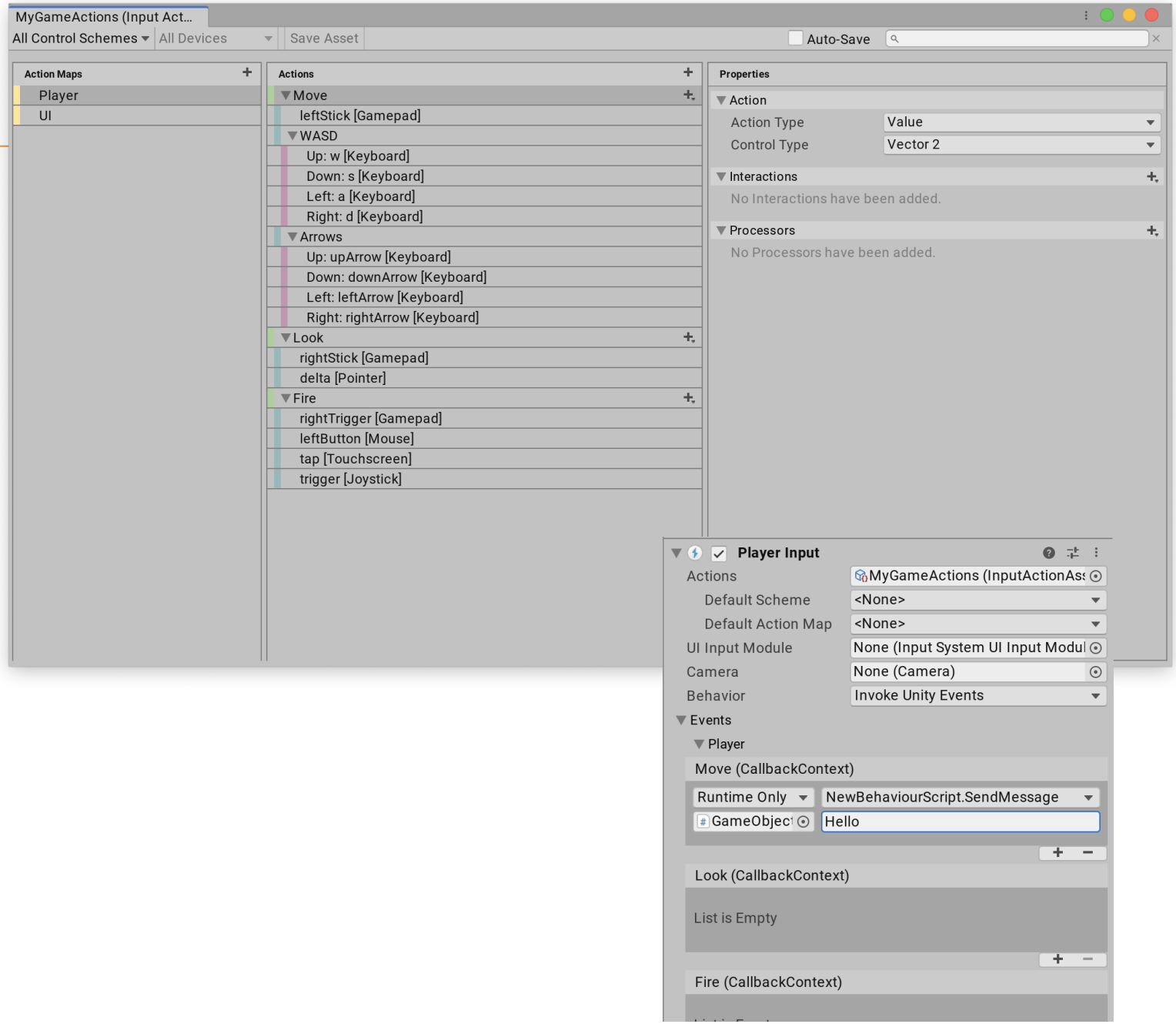
Custom axis and buttons, dead zone, gravity, sensitivity, key binding…

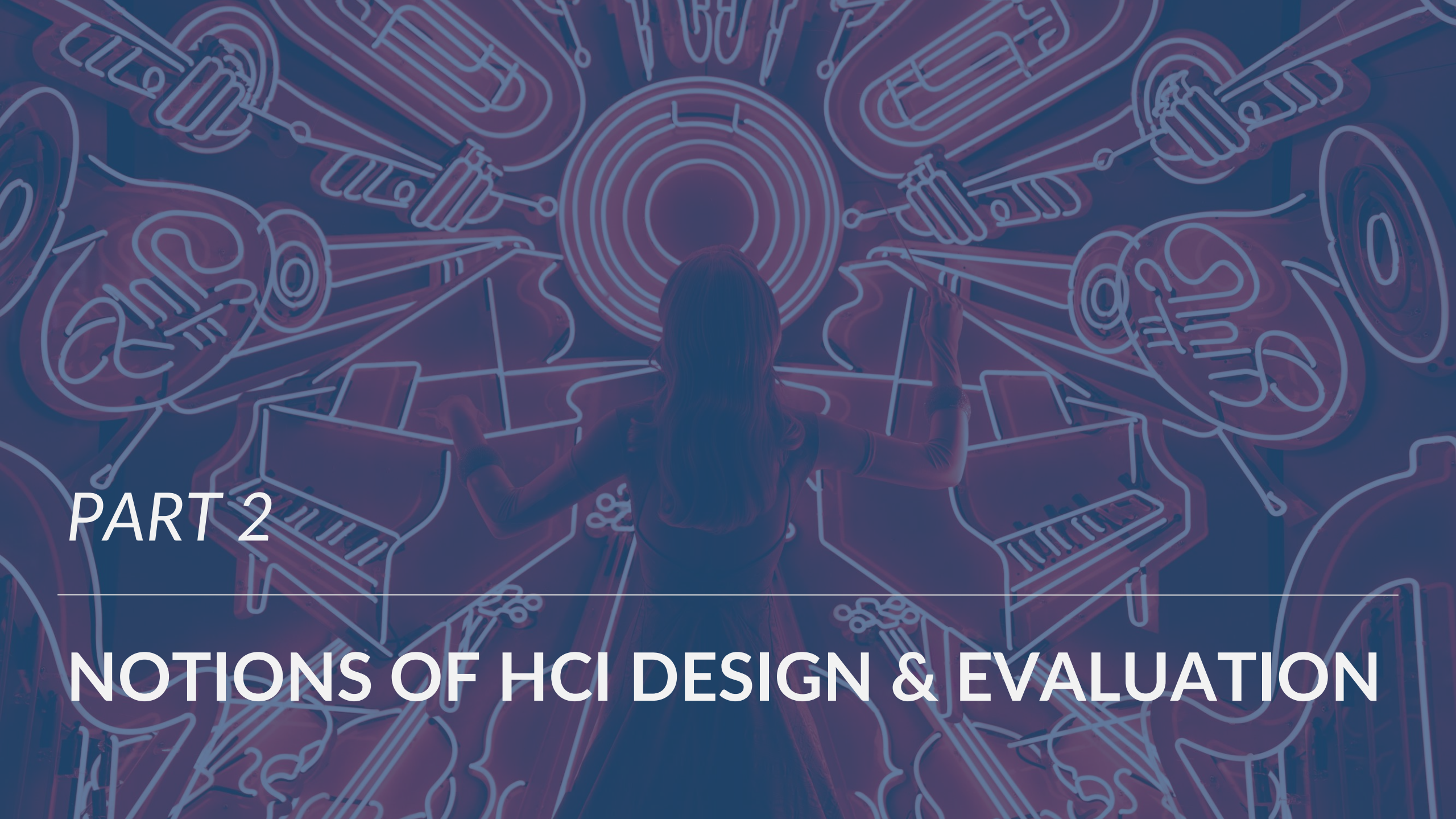
Time

Unity

New Input System

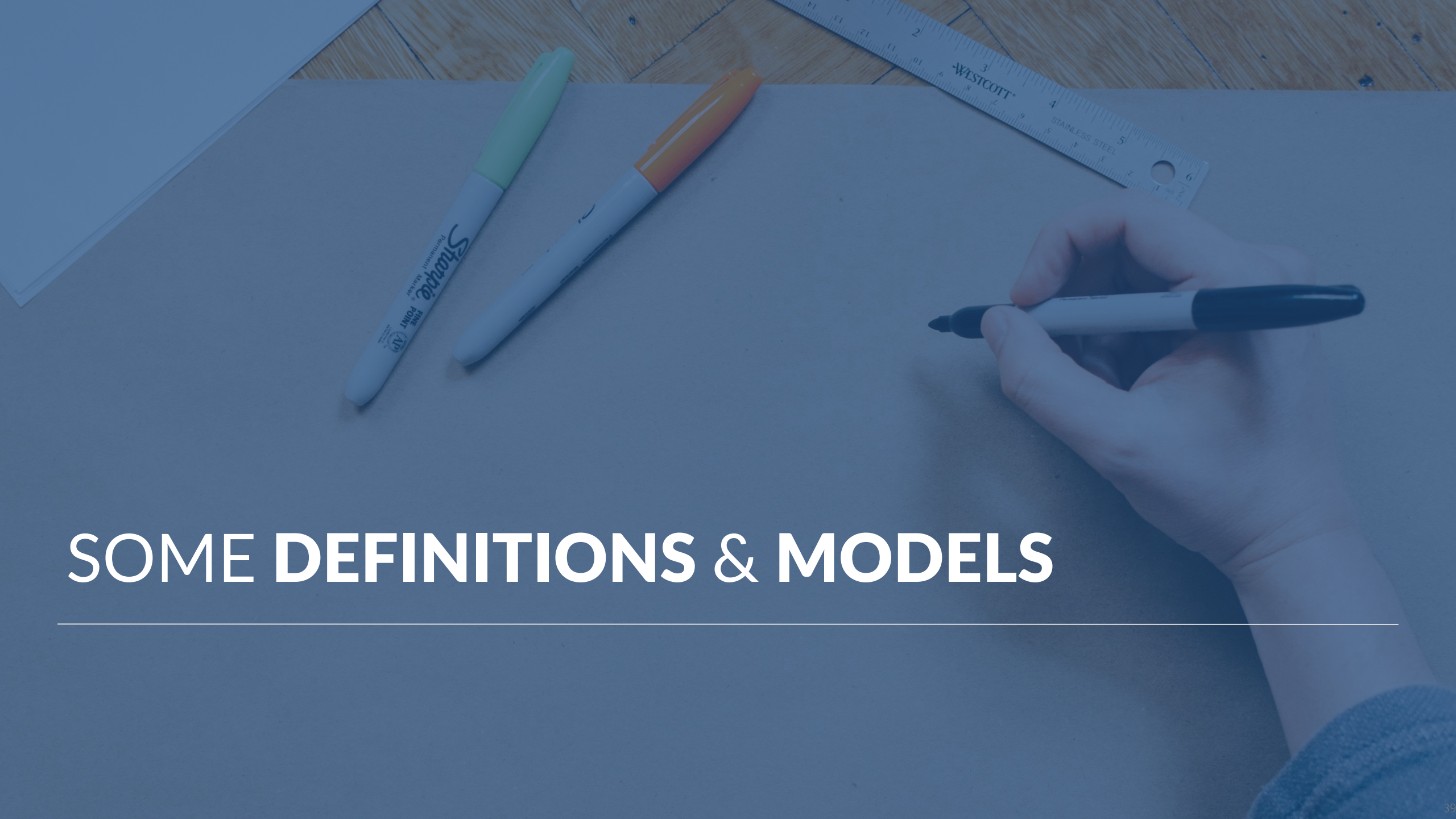
https://docs.unity3d.com/Packages/com.unity.inputsystem@1.4/manual/QuickStartGuide.html





PART 2

NOTIONS OF HCI DESIGN & EVALUATION



SOME **DEFINITIONS** & **MODELS**

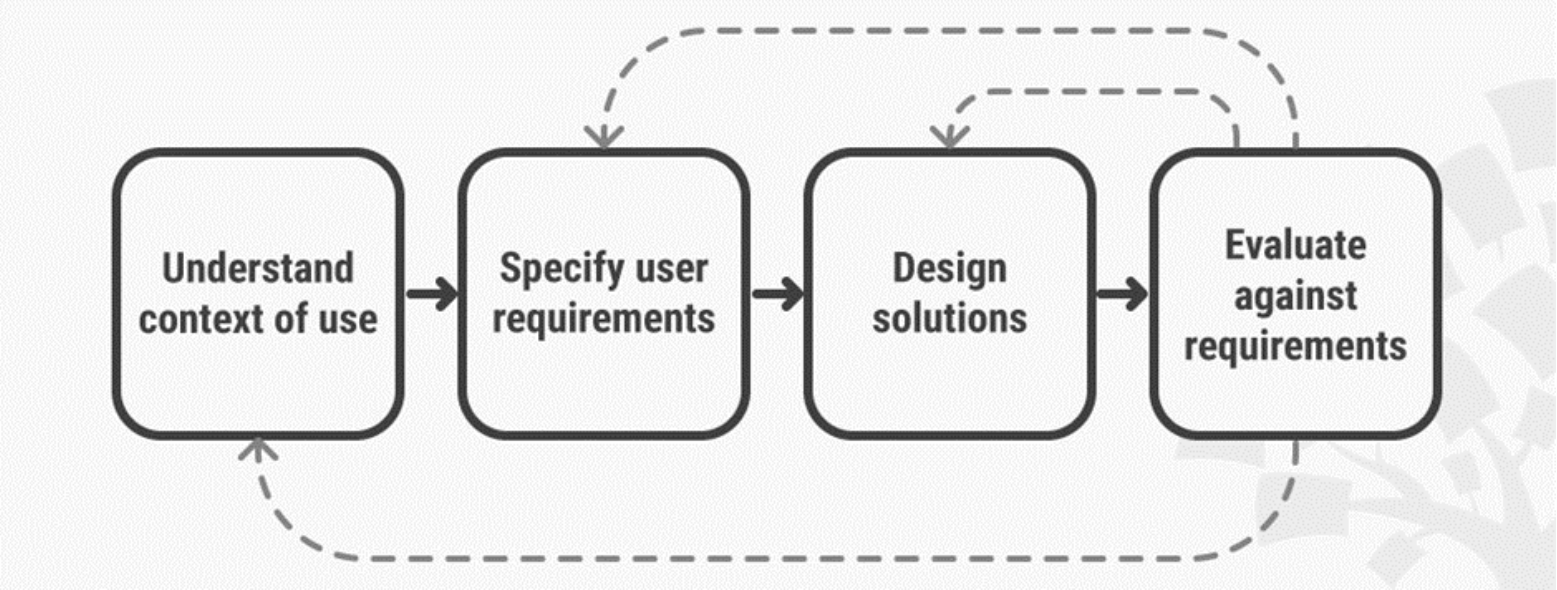
User-Centered Design

Norman & Draper, 80's

Iterative design process in which designers focus early on the **users, their needs,** their tasks and their environment in each phase of the design process

Active participation of users

Iteration of solutions, until the needs and requirements expressed by users are fulfilled



Source: interaction-design.org

User Experience (UX)

"User experience encompasses all aspects of the end-user's interaction with the company, its services, and its products."

Don Norman and Jakob Nielsen

Introduced by Norman, Miller & Henderson (95)

- Beyond "user interface" and "usability"
- Covers all aspects of a person's experience with a system, including industrial design, graphic elements, interface, physical interaction and instructions for use

Popularized by Merhloz (98) et Garrett (02), esp. for web design

User Experience (UX)

Requirements

1. Meet the exact **needs** of the client
2. Create products that are a **pleasure** to own/use
   (through simplicity and elegance)

=> True user experience is more than just giving customers what they say they want, or providing checklist features

=> Seamless merging of services from multiple disciplines: engineering, marketing, graphical, industrial and interface design.

UX vs User Interface

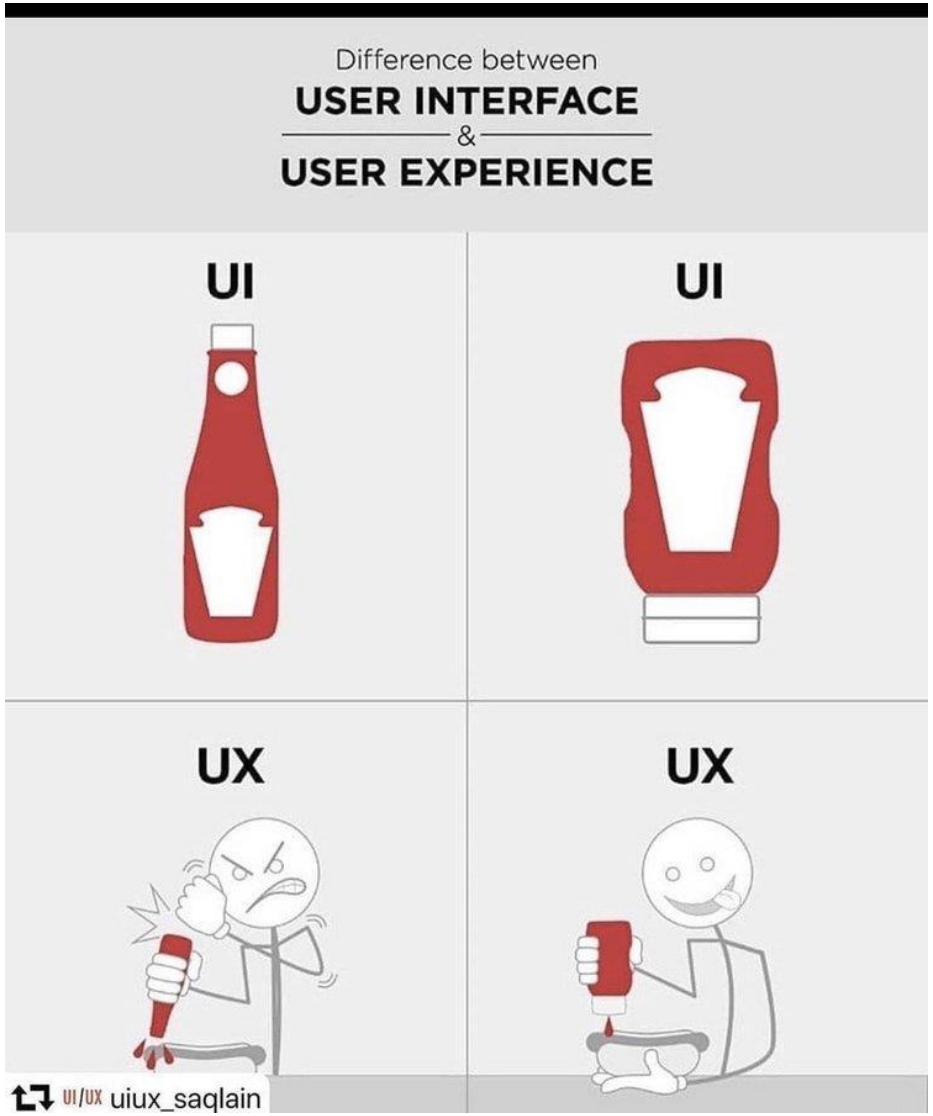
UI = anything a user may interact with to use a product or service

Screens, touchscreens, keyboards, sounds, lights…

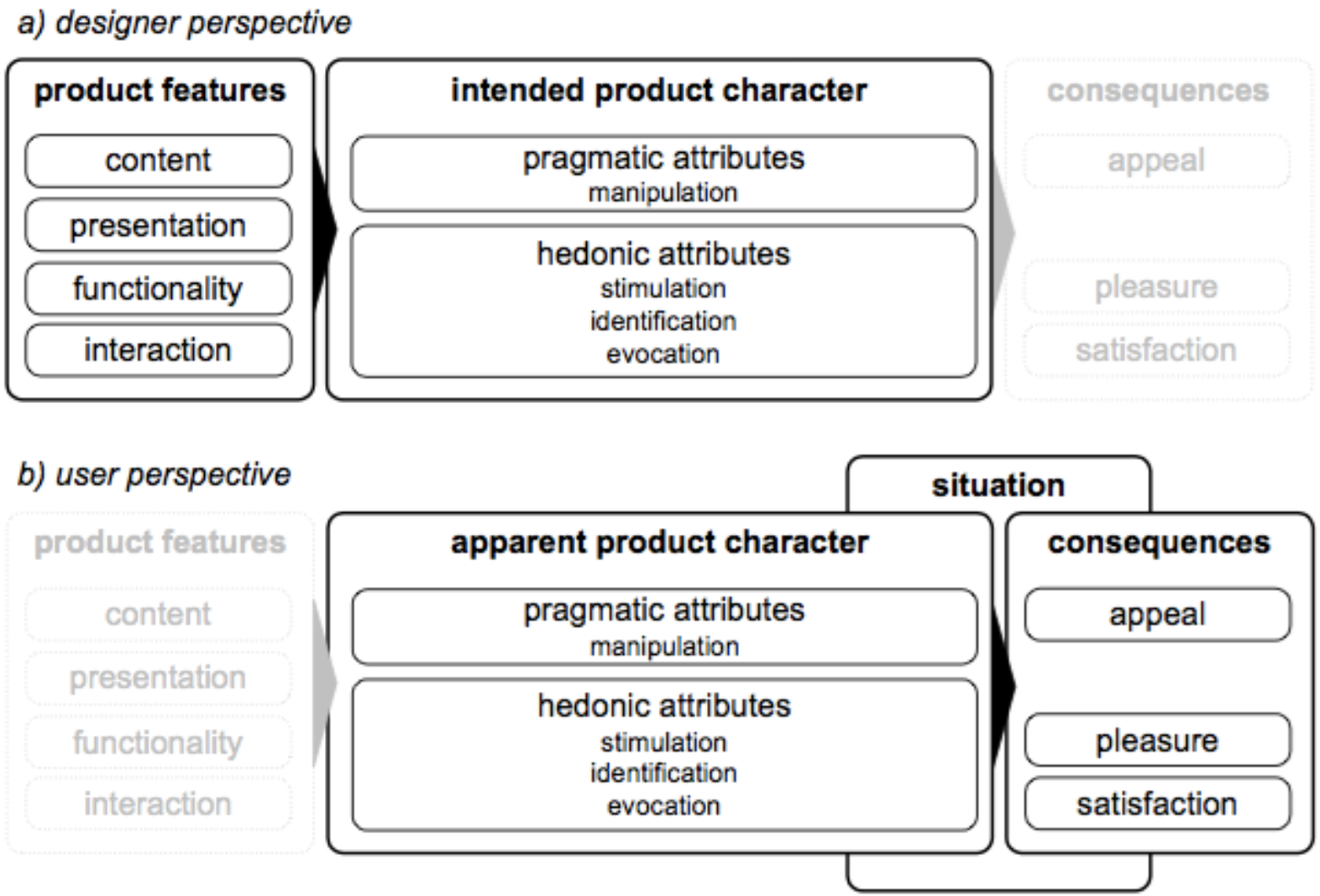
UX focuses on the user's journey through the product to solve a problem

UI focuses on how a product's surfaces look and function, a series of snapshots in time

UX vs User Interface



UX: Designer vs. User Perspectives



Source: Marc Hassenzhal, The thing and I: Understanding the relationship between user and product (2003)

UX Honeycomb (Web)

Useful

Fill a need. Overwise no real purpose for the product itself.

Usable

Easy to use, familiar to understand, short learning curve.

Desirable

Aesthetics attractive and easy to translate.

Findable

Information easy to navigate. Able to quickly find a solution to a problem.

Accessible

Users with disabilities can have the same user experience as others.

Credible

The company and its products or services need to be trustworthy.

Valuable

Value to the business which creates it and to the user who buys or uses it,

Source: Peter Morville

Usability

How easy & pleasant the features are to use

Learnability

How easy to accomplish basic tasks the first time?

Efficiency

Once users have learned, how quickly can they perform tasks?

Memorability

When users return after a period of non-use, how easily can they restore their skills?

Errors

How many errors, how severe, how easily can they recover?

Satisfaction

How pleasant?

Discoverability

Need to determine what the system does, how it works, what actions are possible, the current state of the device…

Must provide information to build a conceptual model of the environment

Linked to psychological concepts such as

- Feedback
- Affordances
- Constraints
- Sense of control

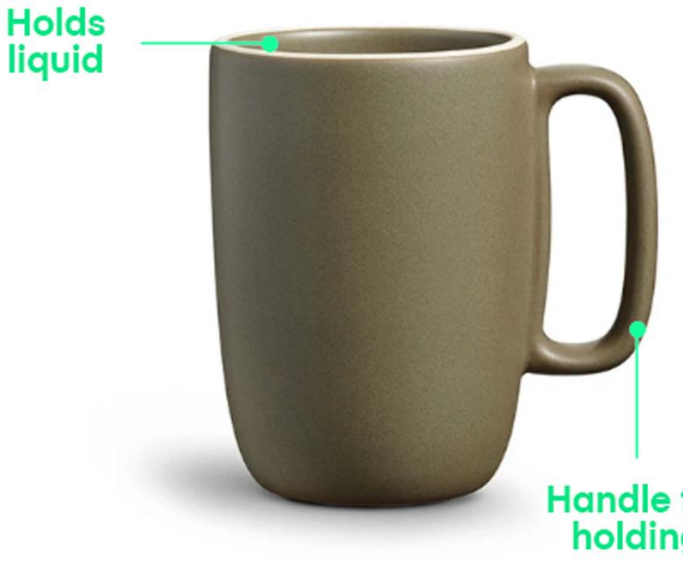
Affordances (Gibson 1986)

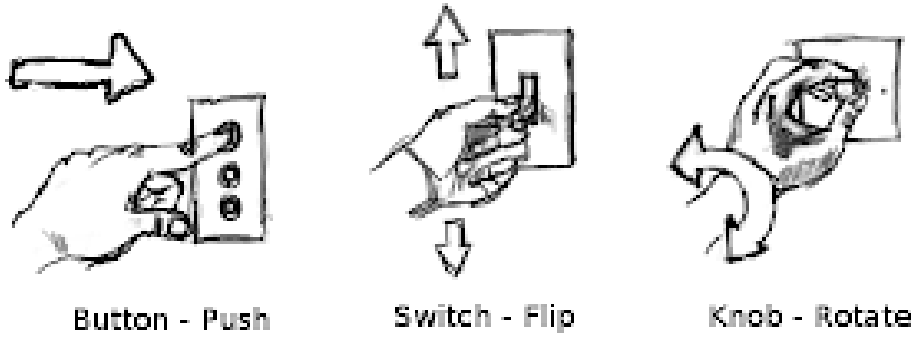
"Affords" = "is for"

A property of an object or environment that communicates a way to use it

"Function follows form"

Ex.: door, drawer handle, button, hyperlink…





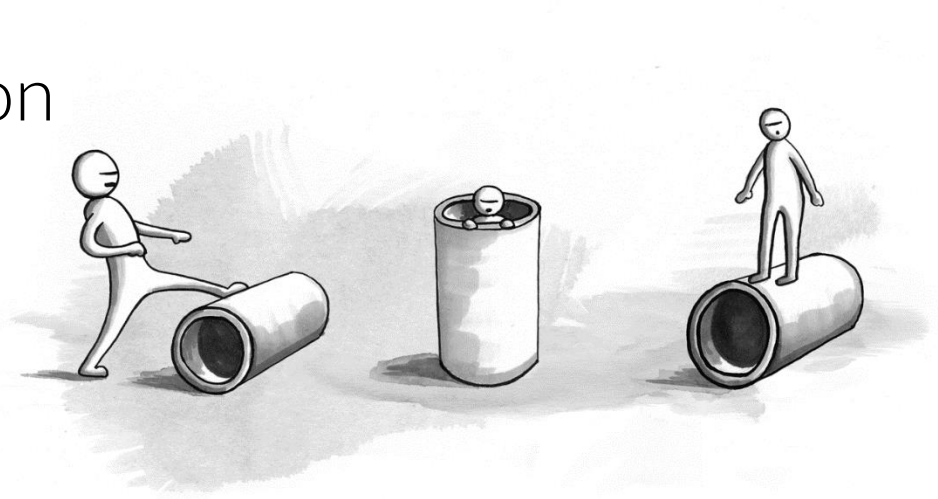
Submit Submit Submit

Affordances (Norman 2013)

The relationship between
- an object and its properties
- a person and its capabilities

The possible interactions between people and the environment

Anti-affordance = the prevention of interaction



Ex :
- *Chair affords support and sitting and lifting*
- *Glass affords transparency but prevents passage*

Signifiers (Norman 2013)

The signaling components of affordances

Perceivable indicator to communicate the purpose, structure, and operation of the device to the people who use it

Deliberate or unintentional

Ex: location of the action

User Interface integration

Non-Diegetic

2D elements, viewfinder, notifications…

Diegetic

3D elements integrated into the space and consistent with the experience

Enhance immersion

Spatial

3D elements integrated into space, but not realistic

Less immersive but more comprehensible

Meta

Effects: shaking, blurring, color changes, etc.

Source: Siddarth Kengadaran

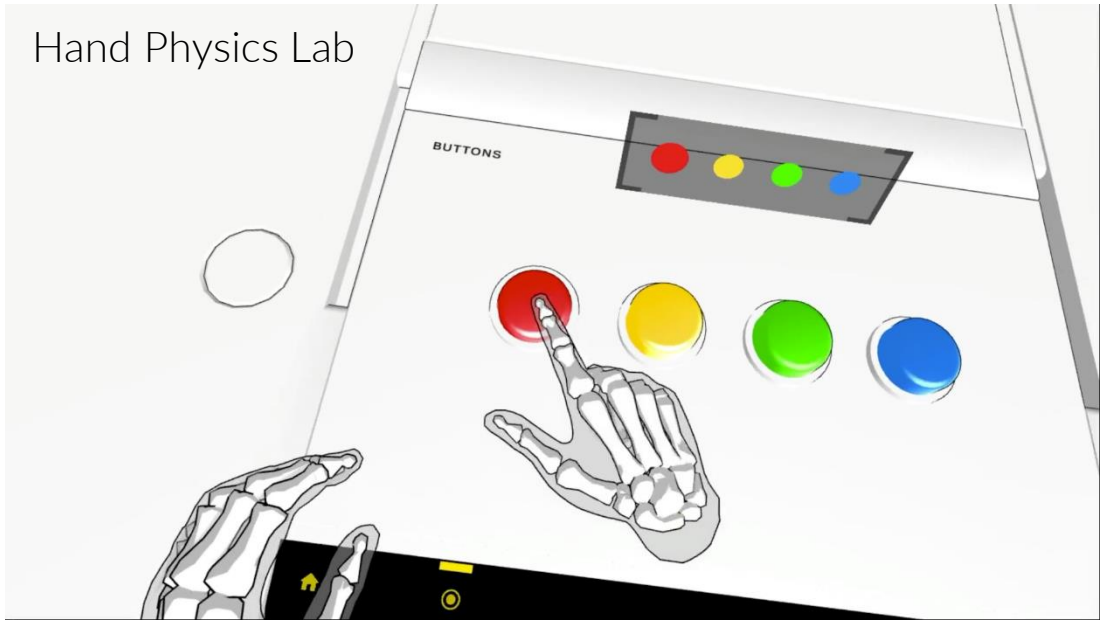
Affordances & Signifiers in Games

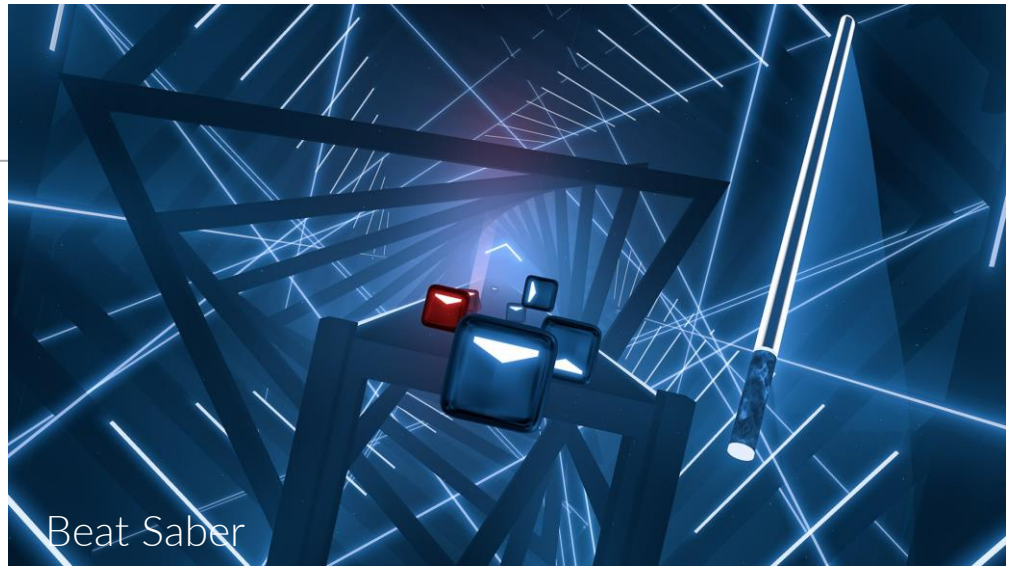
In game design, used to tell players how to interact with game, environment, items, NPCs, combat…

Uncharted 4 climbing and rope swing areas

Braid

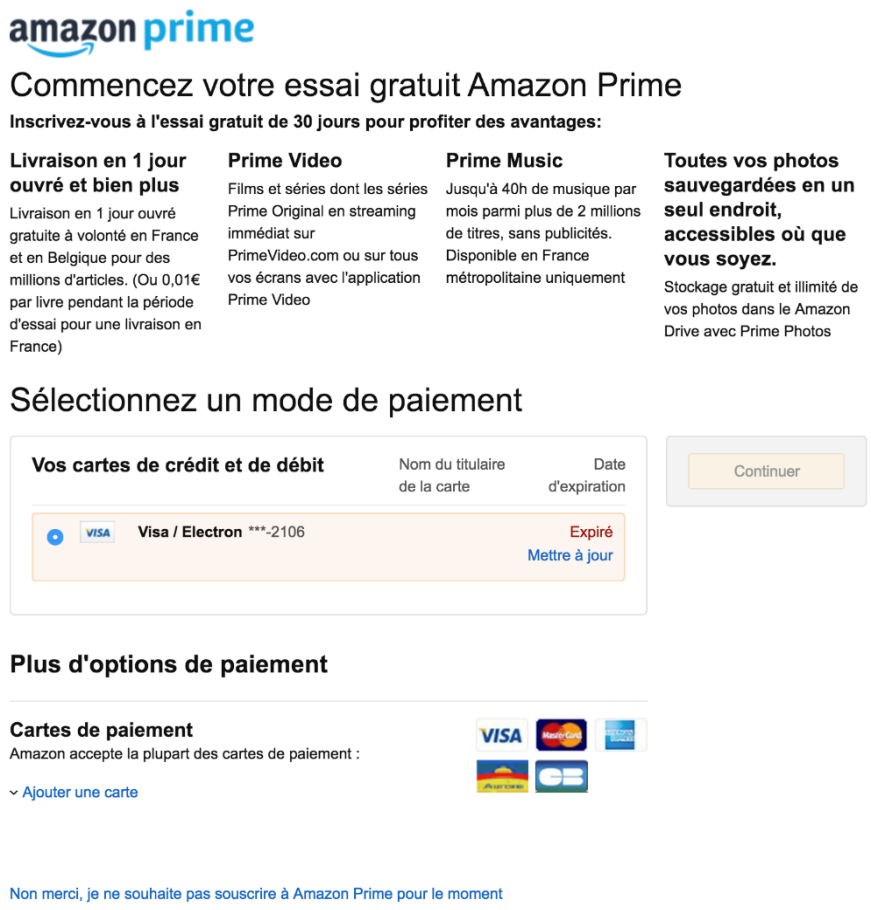
Affordances & Signifiers in VR

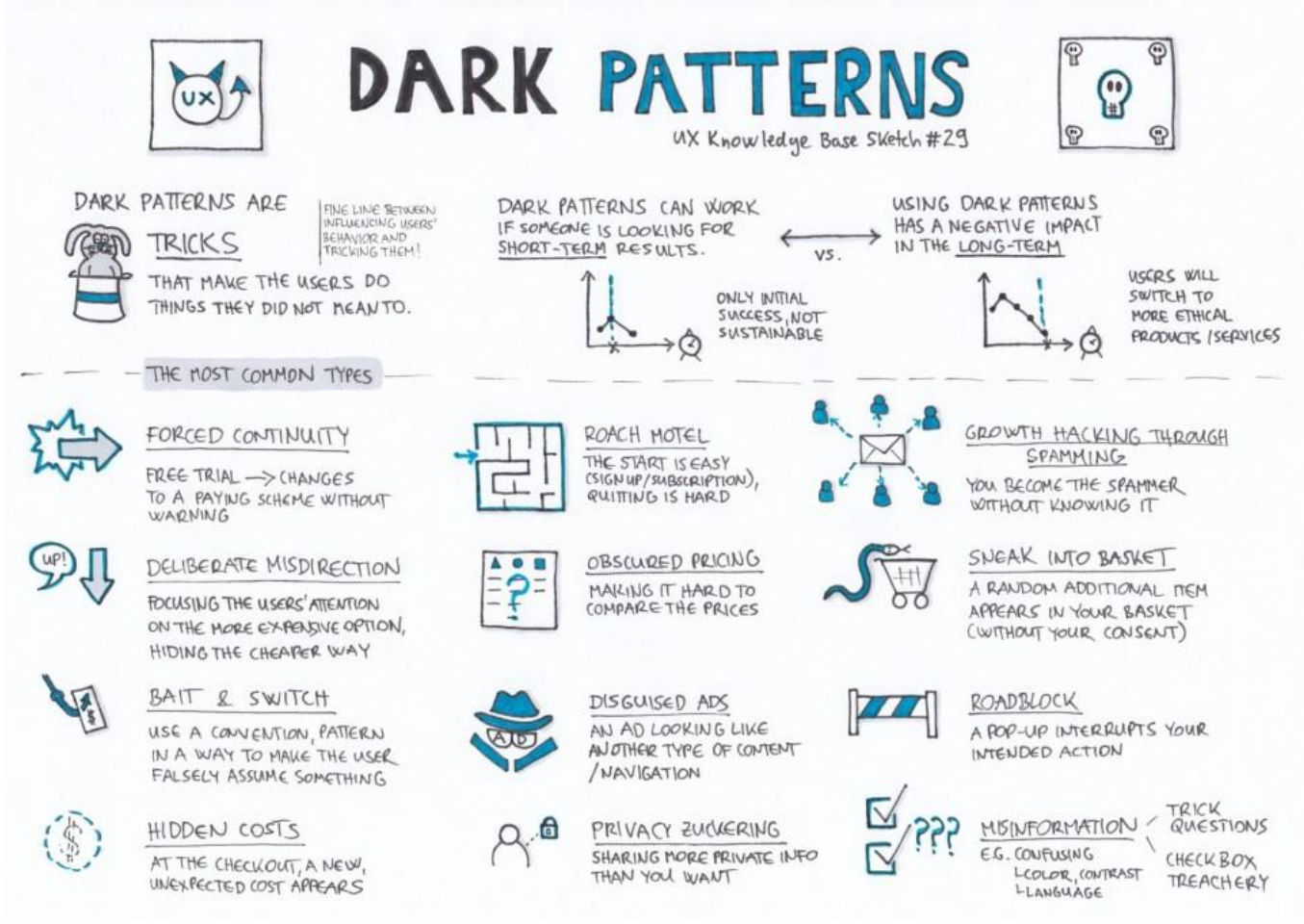




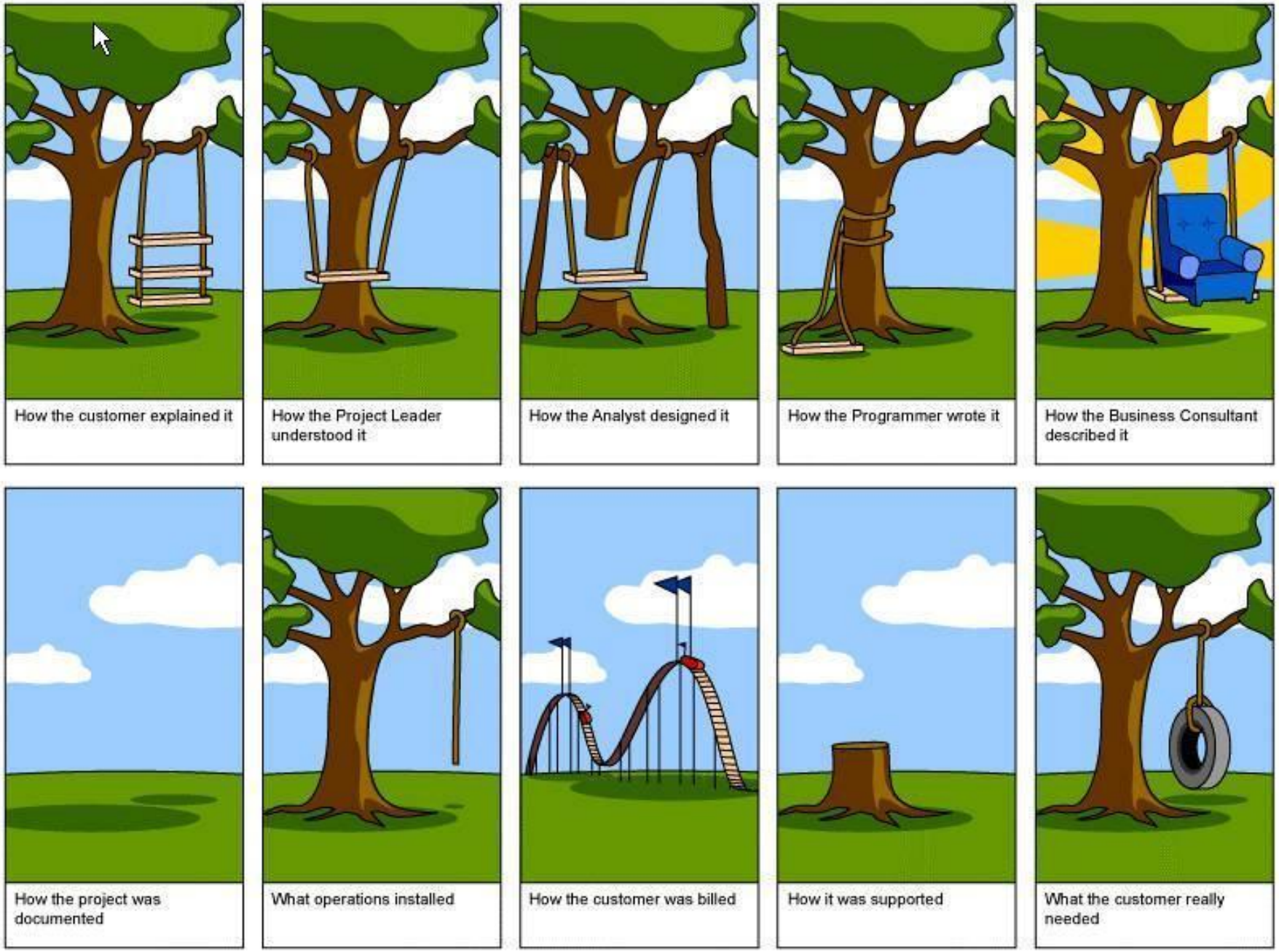
Dark patterns

Elements of product design created to make users do things they might not want to do—actions that benefit the business, not users

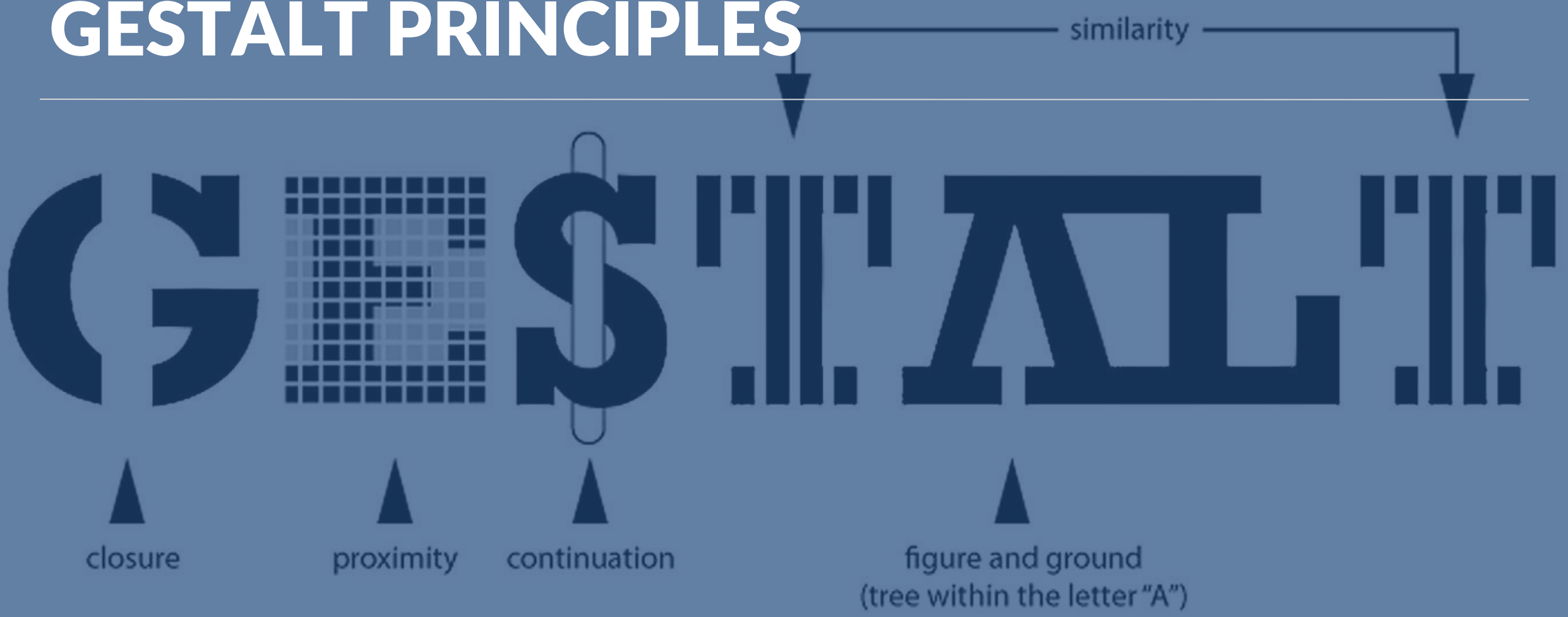




Source : https://uxknowledgebase.com/dark-patterns-3b41ed7a690e


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

GESTALT PRINCIPLES



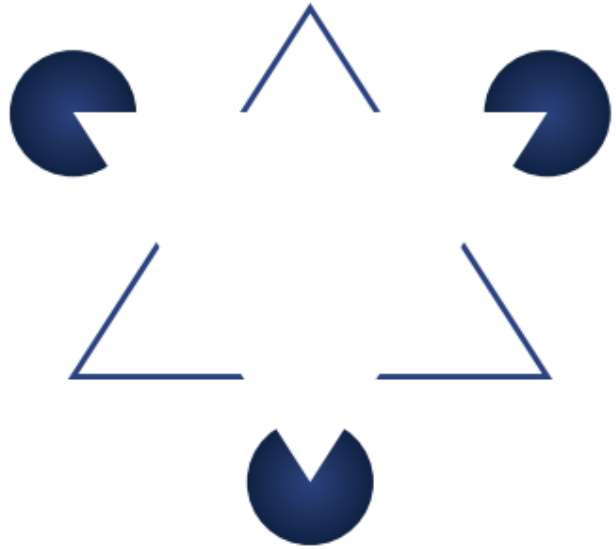
Gestalt Principles

- Perception theory (20's)
- Detail how our mind assembles and interprets visual elements
- Human mind focuses more on a whole than on its parts

Gestalt Principles for UI/HUD

Figure/ground principle

Multistability: Is this a duck or a rabbit?



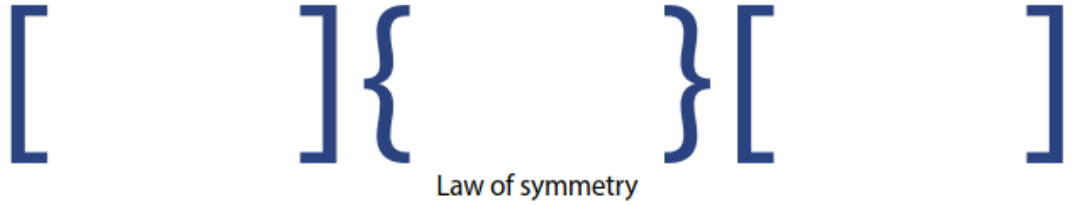
Law of closure

Continuity

Source: Hodent 2018

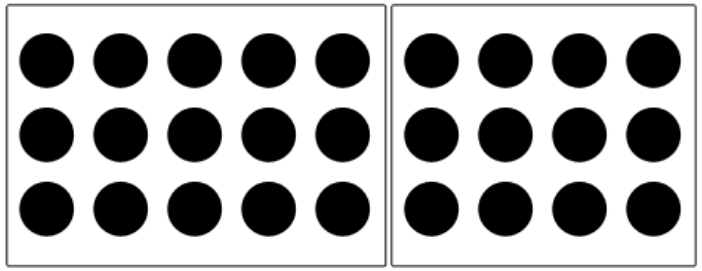
Gestalt Principles for UI/HUD

Law of similarity

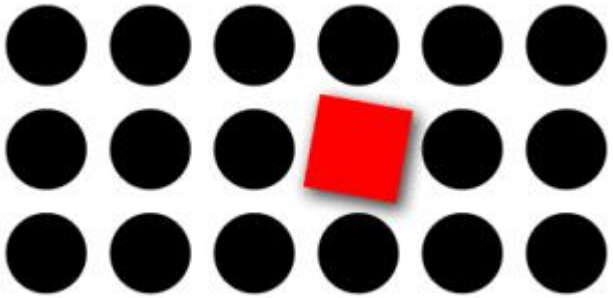


Law of symmetry

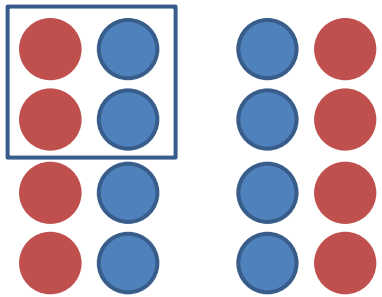
Law of proximity



Common region



Focal Point



Source: Hodent 2018

Examples

Examples

Figure 3.5

Far Cry 4 (Ubisoft), skills menu. (Courtesy of Ubisoft Entertainment, © 2014. All Rights Reserved.)

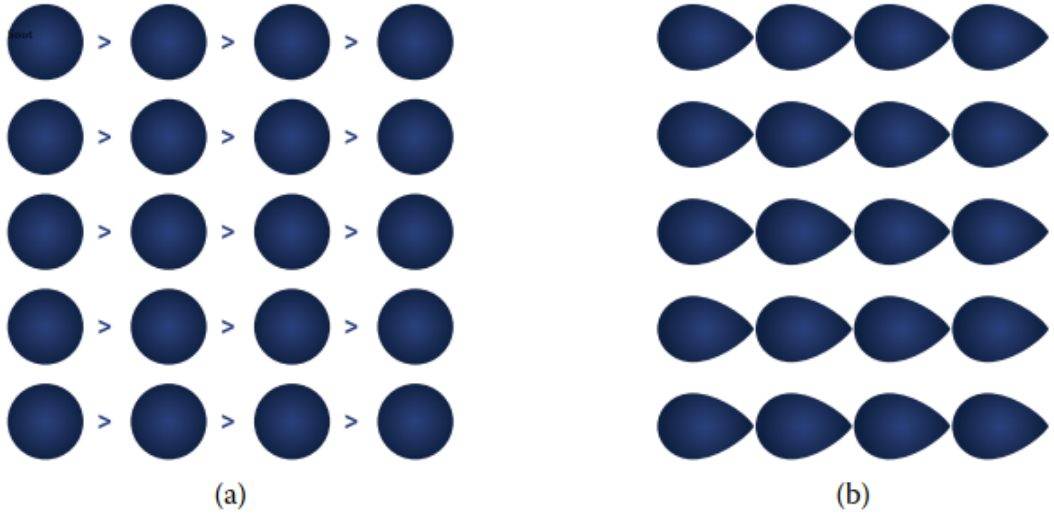


Figure 3.6

(a) *Far Cry 4* skill tree pattern (b) Applying Gestalt principles to make the skill tree pattern more easily readable.

Source: Hodent 2018

FEEDBACK

Photo by Elia Pellegrini on Unsplash

Feedback

Communicate to the user the results of their actions through explicit information (Norman 2013)

Allows users to know that their actions have been taken into account, or to know their consequences

Ex: button pressed, object moved…

Lets people knows that the system is doing something

Ex: loading icon

Cf 1st Nielsen heuristic (visibility of system status)

Feedback

- Every action is associated with an expectation, which feedback will confirm or not ("knowledge of result")
  - Reassures, even if the result is negative
  - Essential for learning
  - Lack of feedback creates a feeling of lack of control
- Can be positive/negative
  - Reinforce the user in his action vs. indicate an error or an impossibility

Feedback

- Immediate
- Informative
  - Bad feedback can be worse than no feedback, causing distraction, anxiety or irritation
- Planned and prioritized
  - Important information must be attention-grabbing; other information must not be intrusive
  - Too much feedback can be worse than too little

Feedback in Games

Jan Willem Nijman - Vlambeer - "The art of screenshake" https://youtu.be/AJdEqssNZ-U

Why Does Celeste Feel So Good to Play? | Game Maker's Toolkit https://youtu.be/yorTG9at90g

Juice it or lose it - a talk by Martin Jonasson & Petri Purho https://youtu.be/Fy0aCDmgnxg

Secrets of Game Feel and Juice | Game Maker's Toolkit https://youtu.be/216_5nu4aVQ

SOME **EVALUATION METHODS**

User Experience Research

Practice of analysing a user's interaction with the product to find insights and identify weaknesses

Qualitative methods

- Focus on observation
- Suitable to know the users, get 1st impression, difficulties, thoughts…
- *Interview*, *diary studies*, *usability testing*…

Quantitative methods

- Gather numerical data
- Suitable for measuring success or discovering deficiencies, answer questions like "how many," "how much," "how often,"… time, errors, satisfaction, usage…
- Need more users for good stats
- *Clickstream analytics*, *A/B testing*, *survey*…

Usability testing

Iterative method of testing few functionalities of a digital product by observing **real users** as they attempt to complete tasks on it

Goals

- Get user reactions and feedback
- Check if the user can perform the tasks proposed
- See if product meet user's expectations
- Check if the design is matching business decision to real world use

Usability testing in practice

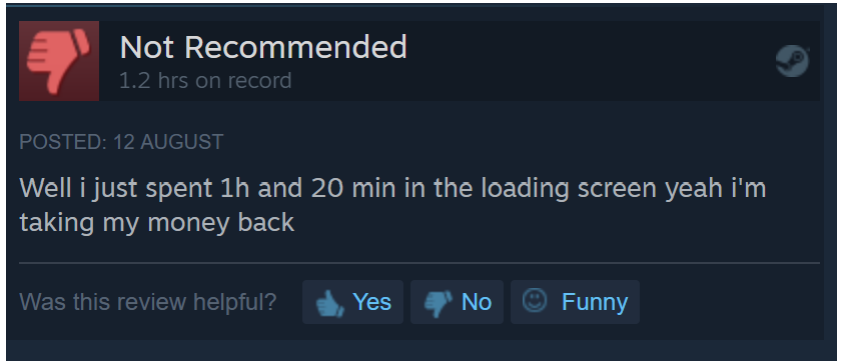
- Starting questions
  - Why do we test?
  - What do we want to know?
  - How will we use the results?
- Participants
  - Representative of target audience
  - Rule of 5
- Prepare the setup, task and speech
  - Product can be incomplete
  - Keep test environment as realistic as possible
  - Be clear on the scope of the test

Usability testing in practice

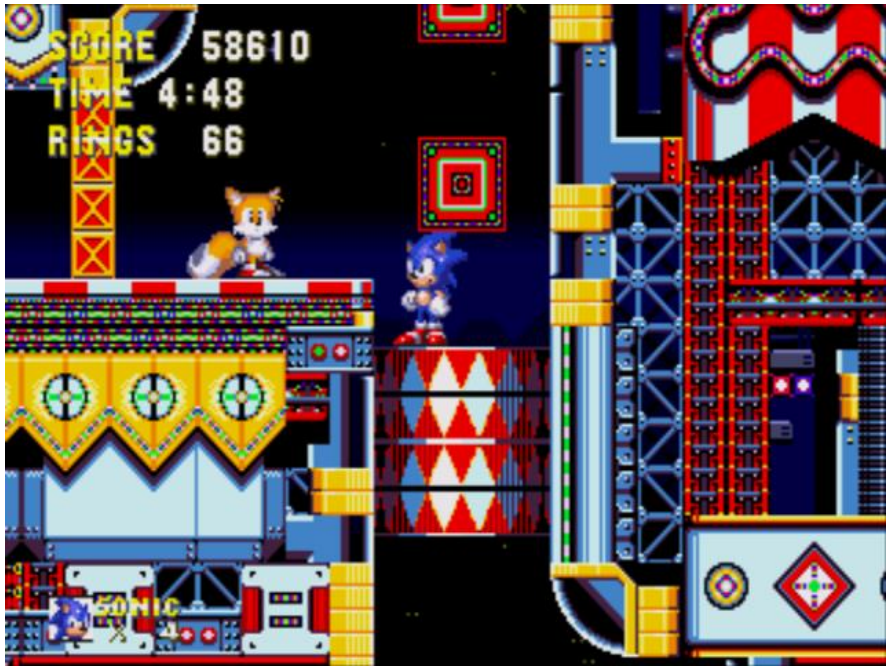
- Ask users to externalize thoughts and feelings (think aloud)
- Do not coach/influence the users
  - But can give specific directions or let free
- Takes notes & record/log the session
  - Focus on what users do
  - Quantitative information: time on tasks, success and failure rates, effort (#clicks, perception of progress)
  - Qualitative information: stress responses, subjective satisfaction, perceived effort or difficulty
- Do not jump into any conclusions during the session
- Do not take it personally
- Summarize, react and repeat

Design choices: Examples

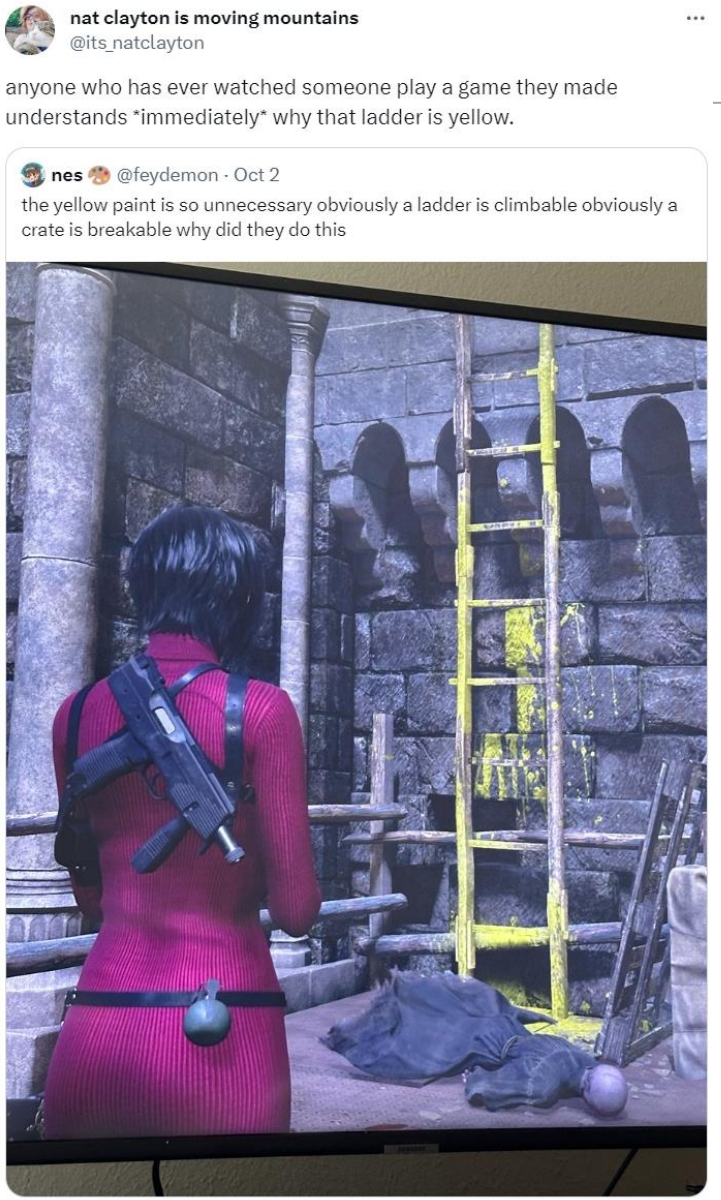
No Man's Sky : start



Design choices: Examples



Sonic 3: moving an obstacle



Resident Evil 4 : yellow ladder debate

5 seconds test

- Measure what information users take away and what impression they get within the first 5s after viewing a design
- Commonly used to check whether web pages effectively communicate their intended message
  - *Ex: What is the purpose of the page? What are the main elements you recall? Who do you think the intended audience is? Did the design/brand appear trustworthy?*
- Not suited to measure comprehension of complex information
  - A page that requires lots of reading (prefer a design survey)
  - Predicting user behavior (prefer a click test or navigation test)
  - Asking complex questions (prefer a design survey)

AttrakDiff

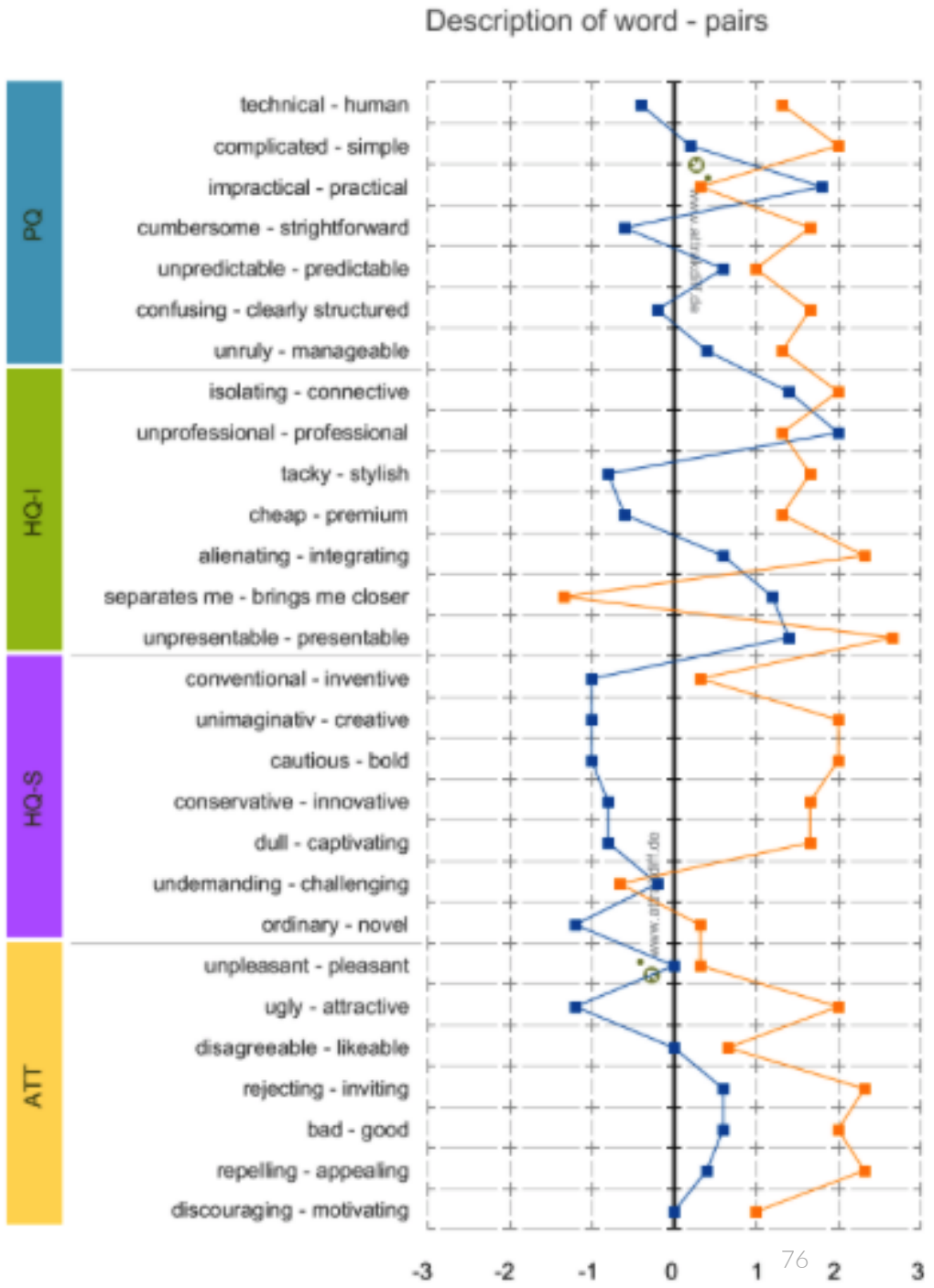
(Hassenzahl, Burmester & Koller, 2003)

Questionnaire

- Pragmatic quality
- Hedonic quality (identity - stimulation)
- Attractiveness

Helps to understand how users personally rate the usability and design of the product

- Single Evaluation
- A/B Comparison
- Before-After Comparison



Heuristic evaluation

Small set of evaluators examine the interface and judge its compliance with recognized usability principles (the "heuristics")

Advantages

- Low cost, quick and easy to apply
- Can obtain feedback early in the design process
- Heuristics can be used both as a design and evaluation support

Heuristic evaluation

10 Usability Heuristics for User Interface Design (Jakob Nielsen 1994)

1: Visibility of system status
2: Match between system and the real world
3: User control and freedom
4: Consistency and standards
5: Error prevention
6: Recognition rather than recall
7: Flexibility and efficiency of use
8: Aesthetic and minimalist design
9: Help users recognize, diagnose, and recover from errors
10: Help and documentation

NB : Applicable to video games

Heuristic evaluation

10 Heuristics for an optimal User Experience (Colombo & Pasch, 2012)

Derived from the flow theory (Csíkszentmihályi, 1975)

1. Clear Goals
2. Appropriate Feedback
3. Focused Concentration
4. Ergonomical Transparency
5. Technology Appropriation
6. Challenges/Skills Balance
7. Potential control
8. Follow the Rhythm
9. Know Thy User's Motivations
10. Conservative Innovation

Heuristic evaluation

Others

Ergonomic criteria for the evaluation of human-computer interfaces (Bastien & Scapin 92)

Ergonomic criteria for Human-Virtual Environments Interactions (Bach & Scapin 2005)

Playability heuristics for mobile games

…

Heuristic evaluation

Drawbacks

Requires knowledge and experience to apply the heuristics effectively

- Judgment often based on expertise rather than heuristics

Trained usability experts are sometimes hard to find and expensive

May identify more minor issues and fewer major issues, or even false issues

Non-exhaustiveness of the dimensions covered by the heuristics

The heuristics are often vague, no precise recommendations or evaluation grid/criteria

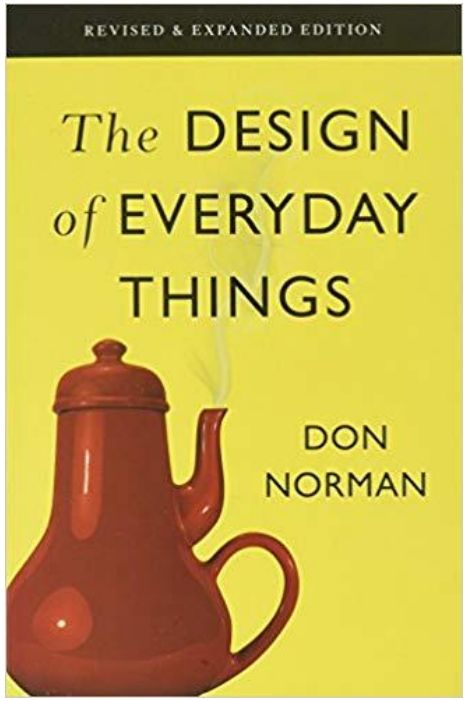
=> Limited validity and reliability, recommended to use it in combination with other user-centered methods

Expert review

Less formal evaluation

Experts base their report not only on heuristics, but rather on their knowledge of user tasks, HCI guidelines and standards, and personal experience

Further readings



https://uxdesign.cc/

https://www.smashingmagazine.com/usability-and-user-experience/

https://www.nngroup.com/articles/

https://medium.com/topic/ux

https://uxplanet.org

http://www.allaboutux.org/

https://flupa.eu/