

Lecture Week 3

# Design Elements

the  
game**design**initiative  
at cornell university

# Actions

---

✂ **Verbs** that describe **what the player can do**

✂ Walk

✂ Run

✂ Jump

✂ Shoot

✂ Does not need to be attached to an avatar

✂ Build

✂ Swap

✂ Rotate

# Actions

---

✂ **Verbs** that describe what the player can **do**

✂ Walk (left or right) (walk, but faster!)

✂ Run (up; jump/run for left or right)

✂ Jump (left or right)

✂ Shoot

✂ Does not need to be attached to an avatar

✂ Build

✂ Swap

✂ Rotate

# Actions

---

✂ **Verbs** that describe what the player can **do**

✂ Walk (left or right) (walk, but faster!)

✂ Run (up; jump/run for left or right)

✂ Jump (left or right)

✂ Shoot

Action  
Platformer

✂ Does not need to be attached to an avatar

✂ Build

✂ Swap

✂ Rotate

# Actions

---

✂ **Verbs** that describe what the player can **do**

✂ Walk (left or right) (walk, but faster!)

✂ Run (up; jump/run for left or right)

✂ Jump (left or right)

✂ Shoot

Action  
Platformer

✂ Does not need to be attached to an avatar

✂ Build

✂ Swap

✂ Rotate

# Designing Actions

---

- ✂ Starts with brainstorming the verbs
  - ✂ Define the types of verbs
  - ✂ Define the scope of the verbs
- ✂ **Design Goals**
  - ✂ Enough verbs to avoid being too simple
  - ✂ But not so much to be confusing (verb bloat)
  - ✂ Do the verbs *directly* achieve the goal?
- ✂ Each verb maps to a single **input**

# Primary Actions



⌘ How do verbs, goals relate?

⌘ Imagine there is no challenges

⌘ What verbs *must* you have?

⌘ **Example:** Platformers

⌘ **Goal:** reach exit location

⌘ Only need movement verbs

⌘ Killing enemies is *optional*

⌘ Other actions are *secondary*

⌘ **Design Goal:** Primary focus

⌘ Secondary verbs lead to bloat

⌘ Add features with interactions

# Interactions

- ⌘ Not a *direct* action of player
  - ⌘ Outcome of the **game state**
  - ⌘ Can happen w/o controller
- ⌘ **Example:** collisions
  - ⌘ Accidental or player forced
  - ⌘ May be bad (*take damage*)
  - ⌘ May be good (*gain power-up*)
- ⌘ **Other Examples:**
  - ⌘ Resource acquisition





# Game Mechanics

---

## ✂ Game mechanic

- ✂ Relationship between verbs and interactions
- ✂ Often call this relationship the “rules”
- ✂ **Gameplay** is manifestation of these rules

## ✂ Example: Joust

- ✂ **Verbs**: Flap; go left or right
- ✂ **Interaction**: Collision with opponent
- ✂ **Rule**: If hit opponent, lower player dies

# Verb Minimalism



- ✂ Keep verbs at a minimum
  - ✂ Mechanics are all interactions
  - ✂ Common in mobile, tablet
  - ✂ Due to lack of input modes
- ✂ **Example:** Sneak Beat Bandit
  - ✂ Has only one verb: *move*
  - ✂ Rhythm game; move to beat
  - ✂ All movement on rails
  - ✂ If obstacle in way, turn

# Avoid Verb Proxies

---

✂ **Proxy**: verb that activates another verb

✂ “Use an item” (what does the item do?)

✂ “Shoot” (what does the weapon do?)

✂ Make your verbs **outcome oriented**

✂ Fire standard bullet (like shoot, but says what it shoots)

✂ Fire freezing beam (what it does and how it is applied)

✂ Important questions to ask

✂ Does it help me reach a goal?

✂ Does it overcome a challenge?



# Combining Actions

---

- ⌘ Verbs can combine in interesting ways
  - ⌘ **Run** and **jump** in a platformer
  - ⌘ **Strafing fire** in a shooter
- ⌘ Typically result of the interactions
  - ⌘ Each verb interacts with environment in different way



# Combining Actions

---

## Running Jump

---

- ✂ Can move while in midair
  - ✂ Just horizontal movement
  - ✂ Not realistic; it is a game
  - ✂ Many platformer challenges assume this type of control
- ✂ Different than a *long jump*
  - ✂ Less height than reg. jump
  - ✂ No control once in the air

## Strafing Fire

---

- ✂ Based on “real life” property
  - ✂ Bullets travel in straight line
  - ✂ Movement changes origin
  - ✂ Walking side-side makes a spray (used in covering fire)
- ✂ But some features are gamy
  - ✂ Bullets slower than life
  - ✂ Character faster than life
  - ✂ Creates interesting effects

# Challenges

---

## ⌘ Obstacles

- ⌘ Prevent progress towards goal
- ⌘ Have to be “overcome”

## ⌘ Opponents

- ⌘ Players with their own goals
- ⌘ May or may not need to be overcome

## ⌘ Dilemmas

- ⌘ Can only perform one of several actions
- ⌘ “Correct” choice not immediately clear

# Challenges: Limitations

---

✂ You **cannot** always perform an action

✂ Shooting may require **bullets**

Resource

✂ Cannot (always) jump in mid air

✂ **Limitation**: requirement to perform action

✂ Boolean test (like an if-then)

✂ Checked at time of user input

✂ Only **one** limitation per verb

✂ If more than one, split into more verbs

# Challenges: Resources

---

⌘ Game State: **numeric** and **symbolic** values that represent the game world at a specific moment

⌘ *Numeric* quantities are often called **resources**

⌘ **Examples (player)**: bullets, health points

⌘ **Examples (external)**: monster spawns

⌘ *Symbolic* values are “yes-or-no” quantities

⌘ Used for “lock-and-key” challenges

⌘ Typically create shallower gameplay



# Putting It All Together

---

✂ Start with your **vision**

✂ I want to \_\_\_\_\_

✂ This creates setting and player goals

✂ Create a (partial) list of the following:

✂ **Objectives**

✂ **Actions**

✂ **Interactions**

✂ **Challenges**



Sketch **player modes** to show them in action

Questions?