Class 8:

Discoverability
Mon

Wed

Fri

9/17
Playtesting and Bias

9/12
Discoverability

9/14
Learning Progressions

9/19
Alpha Testing 1

9/21
Alpha Testing 2

Wed 9/19 @ 10:10am: Alpha Prototype
Review:

Nobody reads and nobody listens
Review:

- Tutorials have questionable effectiveness

### Graphs:

- **Foldit**
  - No Tutorial: 4 levels completed
  - Tutorial: 7 levels completed

- **Refraction**
  - No Tutorial: 15 levels completed
  - Tutorial: 15 levels completed

- **Hello Worlds**
  - No Tutorial: 10 levels completed
  - Tutorial: 10 levels completed
Review: design for discoverability

Braid (2008)
Review: progress through mechanics

I haven't talked to anyone lately, but at least I can solve my own problems.

The Company of Myself (2009)
Review: World 1-1 discoverability
What was specifically mentioned?

- empty space
- looks threatening;
- low penalty
- hard to miss
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
5-10 minutes: pair activity

• Play *The Company of Myself*
  http://www.kongregate.com/games/2DArray/the-company-of-myself

• Describe the tutorial structure:
  • What mechanics are explicitly taught?
    • How are they taught?
  • What mechanics are not taught?
    • How are they *discoverable*?
  • Make a list:

<table>
<thead>
<tr>
<th>Mechanic</th>
<th>How taught?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Action</strong>: move left and right</td>
<td>tutorial message visible immediately</td>
</tr>
</tbody>
</table>
**Action**: move left and right

My attention is stolen by a green square on the other end of the room. I want to be its friend more than anything that I've ever wanted. I decide to use the Arrow Keys to approach it.

*The Company of Myself* (2009)
**Action:** level win confirmation

*Up close, I can see that the green square is actually a door. I think that we can be friends anyway. I decide to push the Spacebar, to move to the next room.*

*The Company of Myself* (2009)
Action: jump

Spotting a couple of platforms ahead, I decide to use the Up Arrow Key to jump over them.

The Company of Myself (2009)
Interaction: death

I remember that the R key will let me reset the current level to try again.

The Company of Myself (2009)
Action: restart

I remember that the R key will let me reset the current level to try again.

The Company of Myself (2009)
Action: clone

I begin to wonder what would happen if I pressed the Spacebar before I completed a level.

The Company of Myself (2009)
Interaction: stand on clone

I begin to wonder what would happen if I pressed the Spacebar before I completed a level.

The Company of Myself (2009)
Observation: *actions* are taught

*I begin to wonder what would happen if I pressed the Spacebar before I completed a level.*

*The Company of Myself (2009)*
Observation: *interactions are not*

*I begin to wonder what would happen if I pressed the Spacebar before I completed a level.*

*The Company of Myself* (2009)
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
How to make discoverable?

- *Impossible to pass* without experiencing it
- *Isolation* from other game mechanics
- Player is relatively *safe*
How to make discoverable?

- *Impossible to pass* without experiencing it
- *Isolation* from other game mechanics
- Player is relatively *safe*
Impossible to pass without learning

My attention is stolen by a green square on the other end of the room. I want to be its friend more than anything that I've ever wanted. I decide to use the Arrow Keys to approach it.

The Company of Myself (2009)
Tradeoff: player freedom

Plants vs. Zombies (2009)
Player is free, but cannot proceed

Braid (2008)
Player is free, but cannot proceed

Braid (2008)
Player is free, but cannot proceed

I remember that the R key will let me reset the current level to try again.
Player is free, but cannot proceed

I start thinking about how awful it would be to fall. Luckily, I know that I can push the R Key at any time to restart the level.
How to make discoverable?

- *Impossible to pass* without experiencing it
- *Isolation* from other game mechanics
- Player is relatively *safe*
How to make discoverable?

- *Impossible to pass* without experiencing it
- *Isolation* from other game mechanics
- Player is relatively *safe*
Isolation: linguistic example

- A bottle of *tesguino* is on the table.
- Everybody likes *tesguino*.
- *Tesguino* makes you drunk.
- We make *tesguino* out of corn.

*tasquino*: (n) corn beer made by the Tarahumara Indians of Mexico
Isolation from other mechanics

My attention is stolen by a green square on the other end of the room. I want to be its friend more than anything that I've ever wanted. I decide to use the Arrow Keys to approach it.

The Company of Myself (2009)
Isolation from other mechanics

The Company of Myself (2009)
Isolation from other mechanics

Spotting a couple of platforms ahead, I decide to use the Up Arrow Key to jump over them.

The Company of Myself (2009)
Isolation from other mechanics

My attention is stolen by a green square on the other end of the room. I want to be its friend more than anything that I've ever wanted. I decide to use the Arrow Keys to approach it.

The Company of Myself (2009)
How to make discoverable?

- Impossible to pass without experiencing it
- Isolation from other game mechanics
- Player is relatively safe
How to make discoverable?

- *Impossible to pass* without experiencing it
- *Isolation* from other game mechanics
- Player is relatively *safe*
Safety

My attention is stolen by a green square on the other end of the room. I want to be its friend more than anything that I've ever wanted. I decide to use the Arrow Keys to approach it.

The Company of Myself (2009)
Safety

*Doom* (1994)
Safety

Safety

Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Nielsen’s heuristics for UI design

1. Make system status visible
2. Match the real world
3. Provide control and freedom
4. Be consistent
5. Prevent errors when possible

Nielsen’s heuristics for UI design

6. Facilitate recognition rather than recall
7. Be flexible and efficient
8. Use minimalist design
9. Help users recognize and recover from errors
10. Provide help and documentation

1. Make system status visible

The system should always **keep users informed** about what is going on, through **appropriate feedback** within reasonable time.
1. Make system status visible

*Note:* This image is from the video game *Minecraft* (2011).
1. Make system status visible

2. Match the real world

The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
2. Match the real world

*Braid* (2008)
3. Provide control and freedom

Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
3. Provide control and freedom

The Company of Myself (2009)
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
Evolution of Final Fantasy UI

*Final Fantasy VII* (1997)
Evolution of Final Fantasy UI

Final Fantasy VIII (1999)
Evolution of Final Fantasy UI

Final Fantasy XII (2006)
Evolution of Final Fantasy UI

**Final Fantasy XIII (2009)**
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
4. Be consistent

Users should not have to wonder whether different words, situations, or actions mean the same thing. **Follow platform conventions.**
5. Prevent errors

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
5. Prevent errors
5. Prevent errors
6. Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
6. Use recognition rather than recall
6. Use recognition rather than recall

Plants vs. Zombies (2009)
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
- Be consistent
- Prevent errors when possible
- Facilitate recognition rather than recall

*King’s Quest VI* (1992)
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
- Be consistent
- Prevent errors when possible
- Facilitate recognition rather than recall

*King’s Quest II* (1985)
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
- Be consistent
- Prevent errors when possible
- Facilitate recognition rather than recall

*Braid* (2008)
7. Be flexible and efficient

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. **Allow users to tailor frequent actions.**
8. Use minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
8. Use minimalist design
9. Help users recover from errors

Error messages should be expressed in plain language, precisely **indicate the problem**, and constructively **suggest a solution**.
9. Help users recover from errors

*Braid* (2008)
10. Provide help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.
10. Provide help and documentation

NEW TECHNIQUES!

- Holding a shell
  ![Symbol showing holding a shell](image)
  (Holding the B Button)

- Running with a shell
  ![Symbol showing running with a shell](image)
  (Holding the B Button)

- Kicking the shell
  ![Symbol showing kicking a shell](image)
  (Releasing the B Button)

- Breaking a block
  ![Symbol showing breaking a block](image)

When Mario has a tail

- Accelerating
  ![Symbol showing accelerating](image)
  Power Meter going up

- More acceleration
  ![Symbol showing more acceleration](image)
  Meter full, (P) starting to flash

- Take off
  ![Symbol showing take off](image)
  Press the A Button repeatedly

- Mario can only fly for a short time.
  ![Symbol showing Mario can only fly](image)
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
- Be consistent
- Prevent errors when possible
- Facilitate recognition rather than recall
- Be flexible and efficient
- **Use minimalist design**
- Help users recognize and recover from errors
- Provide help and documentation
Examples

- Make system status visible
- Match the real world
- Provide control and freedom
- Be consistent
- Prevent errors when possible

- Facilitate recognition rather than recall
- Be flexible and efficient
- Use minimalist design
- Help users recognize and recover from errors
- Provide help and documentation
Nielsen’s heuristics for UI design

1. Make system status visible
2. Match the real world
3. Provide control and freedom
4. Be consistent
5. Prevent errors when possible
6. Facilitate recognition rather than recall
7. Be flexible and efficient
8. Use minimalist design
9. Help users recognize and recover from errors
10. Provide help and documentation
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Outline

1. Case study
2. Principles of discoverability
3. Heuristics of user interface design
4. Group activity
Group Activity

• Pick two *actions* in your game
  • Brainstorm tutorials for this
  • Sketch them (including UI)

• Pick two *interactions* in your game
  • Brainstorm *discoverable situations*
  • Sketch them (including UI)