
Gameplay

Overview

- We will focus on gameplay mechanics
  - Not covering narrative, aesthetics, etc.
- Lots of advice from *The Art of Game Design* by Jesse Schell
  - Professor at Entertainment Technology Center at CMU
- We will also be introducing the final project today

Questions

Overarching question: What makes games fun?

Other important questions:

- What is a game?
- What is a puzzle?
- What does fun mean?
- How do I balance my game?
- Will feature *X* improve my game?

What is a game?

- A partial list of important game qualities
  - Has goals
  - Has conflict
  - Has rules
  - Can be won and lost
  - Is interactive
  - Has challenge
  - Can create internal value
  - Engages players
  - Is a closed, formal system
- Not every game has all of these qualities

A Simpler Definition

"A game is a problem-solving activity, approached with a playful attitude." - Jesse Schell

- Why is this important?
  - Consider what problems the player is solving
  - Think about how to add interesting new problems

What is a puzzle?

"A puzzle is a game with a dominant strategy" - Schell

- Puzzles are key to many games
- Goal must be easily understood
- Sense of progress important
- It's not fun being stuck
- How to prevent the player from getting stuck
  - Provide hints
  - Dynamically adjust difficulty
    - e.g. add more time for a timed challenge after the player fails
  - Provide a way of skipping puzzles

What does fun mean?

- Many categories of fun
  - Fun of learning
  - Fun of mastery
  - Fun of exploration
  - Fun of overcoming challenges (called Fiero)
  - Social fun
- There are also different categories of players
  - Bartle's taxonomy has 4 categories based off surveys of MUD (multi-user dungeon) players

Bartle's Taxonomy of Player Types

- Achievers
  - Want to achieve goals of the game, overcome challenges
- Explorers
  - Want to know breadth of game, pleasure of discovery
- Socializers
  - Seek relationships with other players, pleasure of fellowship
- Killers
  - Mainly enjoy competing and defeating others

Beyond Bartle's Taxonomy

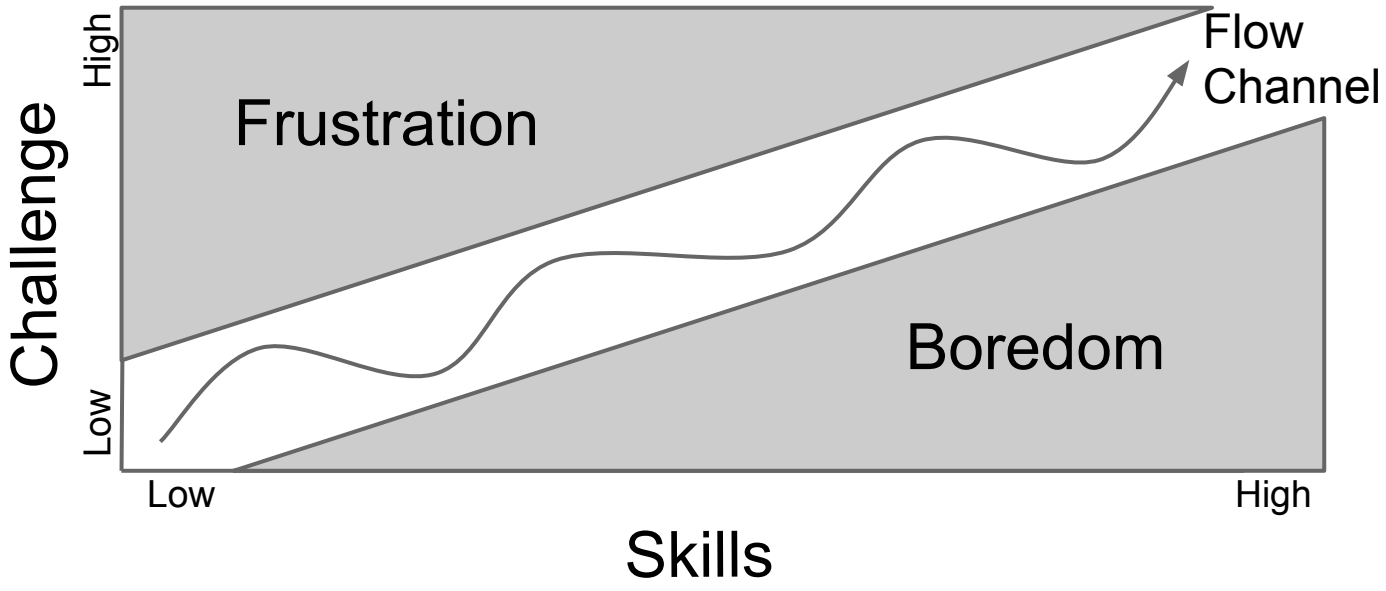
- Bartle proposed expanded model in 2003
  - Friend
  - Griefer
  - Hacker
  - Networker
  - Opportunist
  - Planner
  - Politician
  - Scientist
- Others argue for a "component" model
  - Measure each type/component independently
  - http://www.nickyee.com/daedalus/motivations.pdf

Difficulty

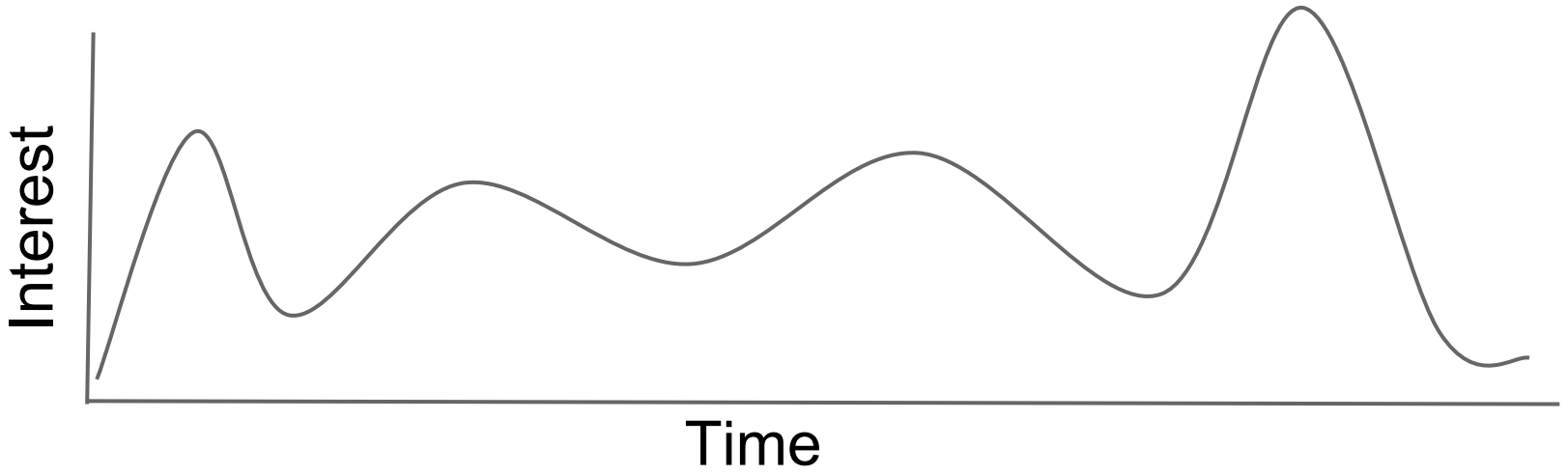
- What's appropriate difficulty for the creator is way too difficult for the average player
- Usually want difficulty levels and/or adaptive difficulty
  - Widens your audience
  - Adds longevity for dedicated players
- Playtest, playtest, playtest!
  - It will be obvious what playtesters find too difficult

Flow

"A feeling of complete and energized focus in an activity, with a high level of enjoyment and fulfillment" - Schell



Interest Curves



- Used in all kinds of storytelling
- Starts off with a *hook*
- Includes a *climax* close to the end

Case Study: Left 4 Dead

- FPS based on a group of 4 survivors in a zombie apocalypse
- Extremely simple goal: get from a starting position to a safe house
- What makes it fun and replayable?
- AI director adjusts pacing based on how you play
  - Procedurally populates world with enemies

Left 4 Dead: AI Director

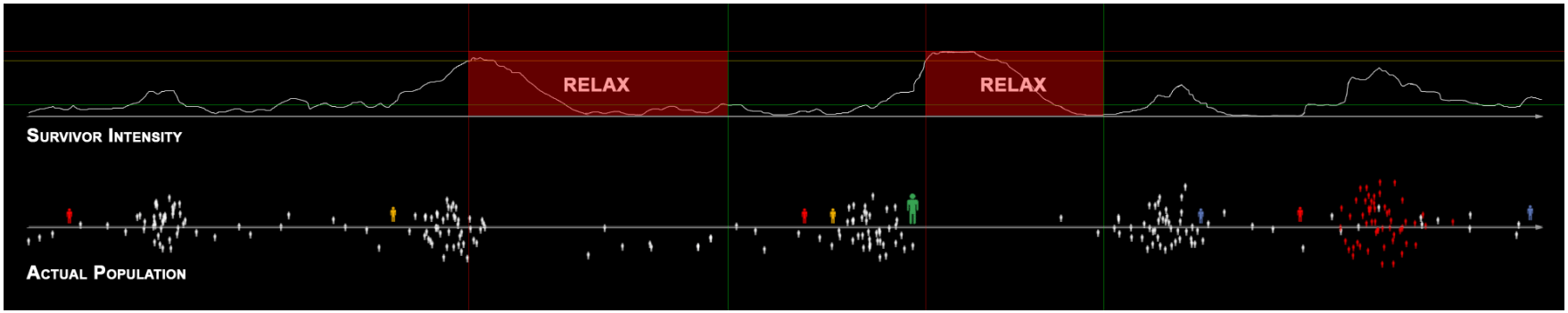
- Promote "structured unpredictability"
  - Uniform randomness isn't exciting
  - Exact positioning isn't replayable
- Want to move toward a target intensity
  - Model "Emotional Intensity" of each player
  - Take max of intensity of each player
  - If too high, temporarily remove threats
  - If too low, create an interesting population

Left 4 Dead: "Emotional Intensity"

- Model survivor intensity as a number
- Increase intensity when player is damaged, pushed off a ledge, etc.
- Decay intensity over time when player is not engaged

Left 4 Dead: Population Modulation

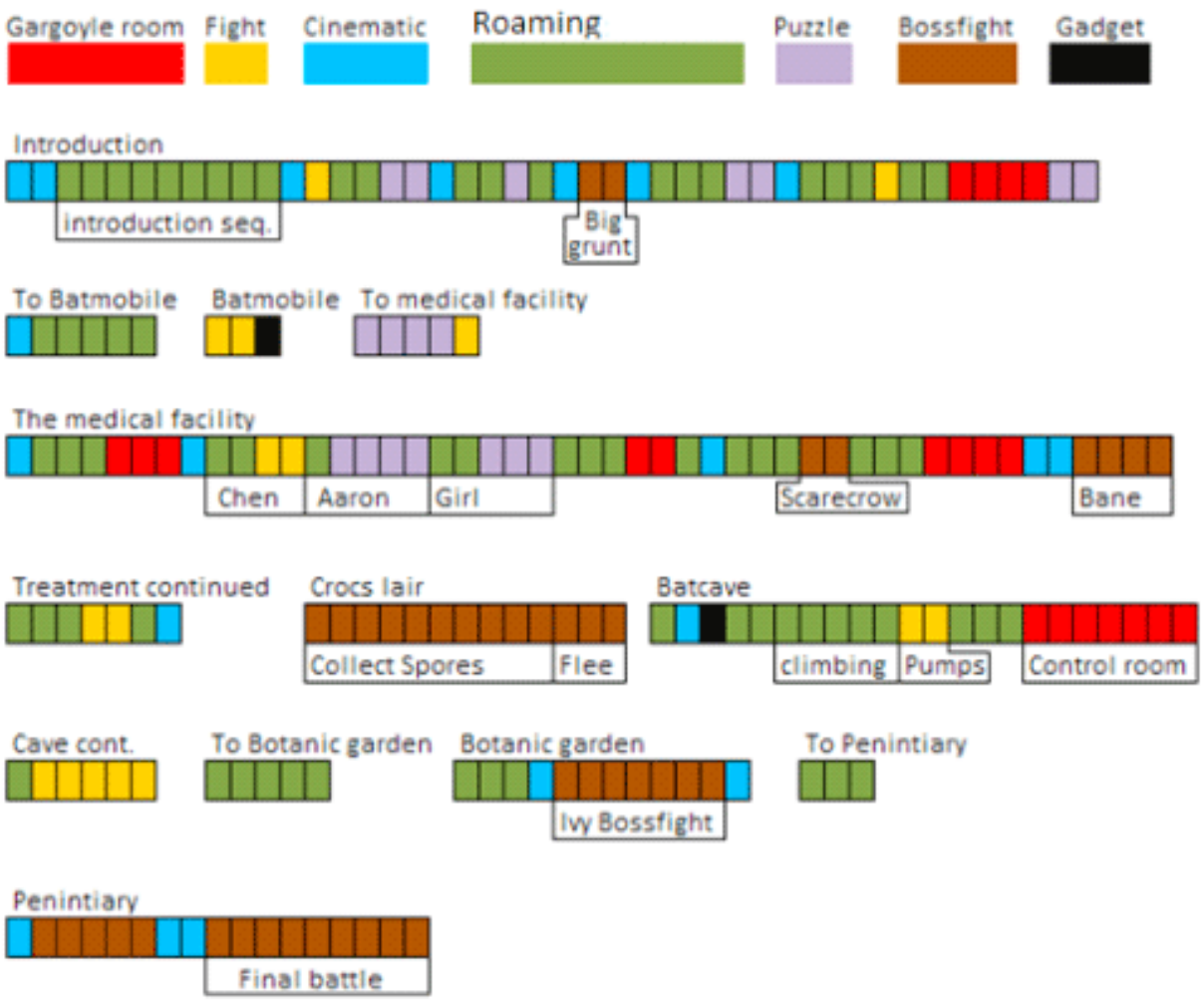
- Use emotional intensity to modulate population
  - Build up, sustain, fade, and relax
  - Sound familiar? All about Interest curves!



Case Study:
Batman: Arkham Asylum

- Third-person adventure, player is Batman
- Variety of gameplay types
  - Puzzles
  - Roaming
  - Combat
  - Gargoyle room (sneaking without being noticed)
  - Cinematic (no player interaction)
  - Boss fight
  - Gadget (player given new gagdet to play with)
- How to combine all these elements?
  - Pacing through variety

Batman: Arkham Asylum



Batman: Arkham Asylum

- Interleaves contrasting gameplay elements
  - Rarely more than 5 minutes of successive combat
  - What does the interest curve look like?
- Steady progress
  - New gadgets awarded roughly every 60 minutes of gameplay
  - Allows short bursts of exploration, entirely open world often too overwhelming
    - Anticipation of unlocking new areas
  - Clear objectives and clear rewards

Balance

- All games, especially multiplayer ones, have some notion of balance
- Unbalanced game becomes old quickly
  - Good balance provides longevity
- What is balance?
  - Preventing a single dominant strategy
  - Strive for *emergent complexity*

Balance: The Problems

- Balancer's Paradox
  - Can't balance weapon's power until we know player's health
  - Can't balance player's health until we know weapon damage
- Fairness != balance
  - Rock-Paper-Scissors is fair
  - But it's a boring game, all roles are the same!
  - We want balance with asymmetric gameplay

Balance: Suggestions

- Balance in passes
  - Don't backtrack in the middle of a pass!
- Starting with paper design helps
  - List out as many specifics as possible as a baseline
- Know what cannot change
  - List invariants, tweak everything else
- Make large changes, not small ones
  - e.g. Use factors of 2 to adjust scale, not "I'll add 0.5"

Case Study: Halo 3

- Sniper rifle was overpowered in Halo 2
- How to fix this?
- Make the balance instantly noticeable
  - Not total ammo
- Balance what the player can see
  - Not damage per shot
- Don't add a new weakness
  - Instead balance out strengths

Halo 3: What to change?

- So, what should be changed?
- Increase time to zoom
  - Doesn't fix close range use
- Increase length of reload
  - Feels like a weakness
- Reduce number of shots in clip
  - Pressure to constantly reload
- Increase time between shots
  - Instantly noticeable, but doesn't weaken original role

Halo 3: Final Decision

- Time between shots increased from 0.5 seconds to 0.7 seconds
- *Lots* of testing before the change was finalized

Game Spaces

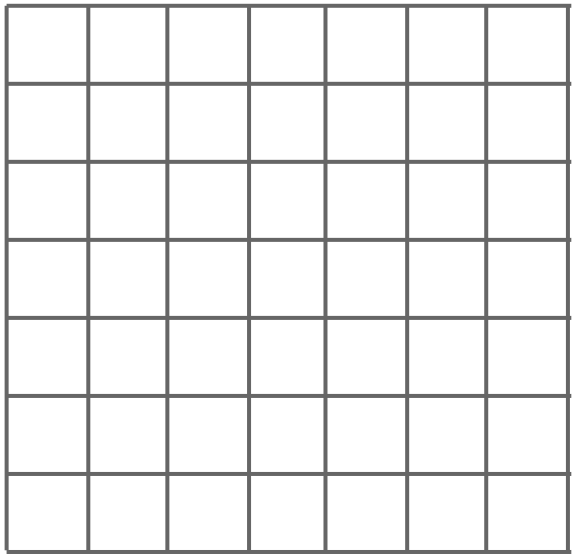
- More than just the geometry of a world
- Graph that connects different game areas
- Consider different types of game spaces
  - Linear
  - Grid
  - Web
  - Hub
  - Divided Space

Game Spaces: Linear

- Player moves forward and backward
- Examples
  - *Super Mario Brothers*
  - *Crash Bandicoot*
  - *Guitar Hero*

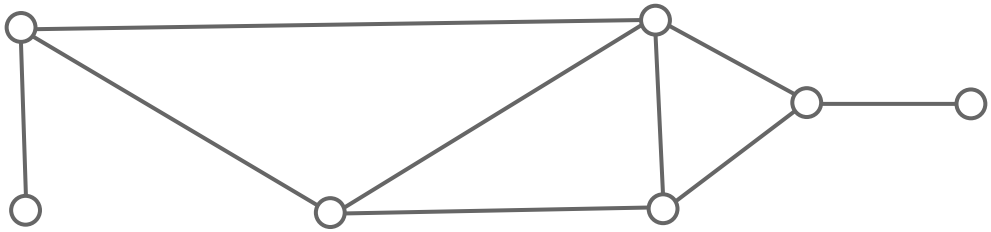
Game Spaces: Grid

- Easy to understand
  - For both players and computers
- Cells don't need to be square
  - *Civilization V* uses a hexagonal grid
- Examples
  - *Fire Emblem*
  - *Legend of Zelda (NES)*
  - *Civilization*



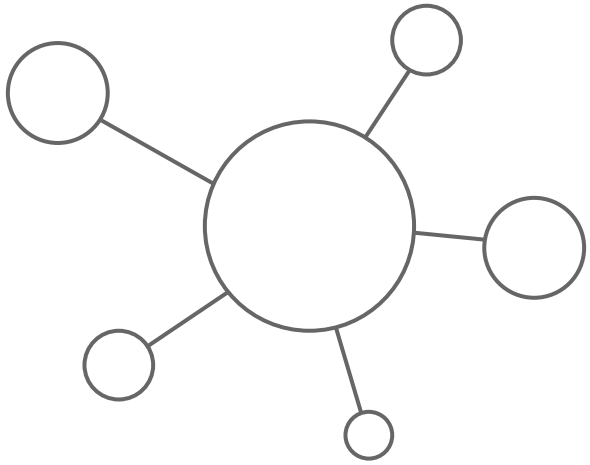
Game Spaces: Web

- Interest points connected by paths
  - Common technique: several branches ending at same location
- Examples
  - *Zork*
  - *Defense of the Ancients (DotA)*
  - *Onslaught Mode in Unreal Tournament 2004*



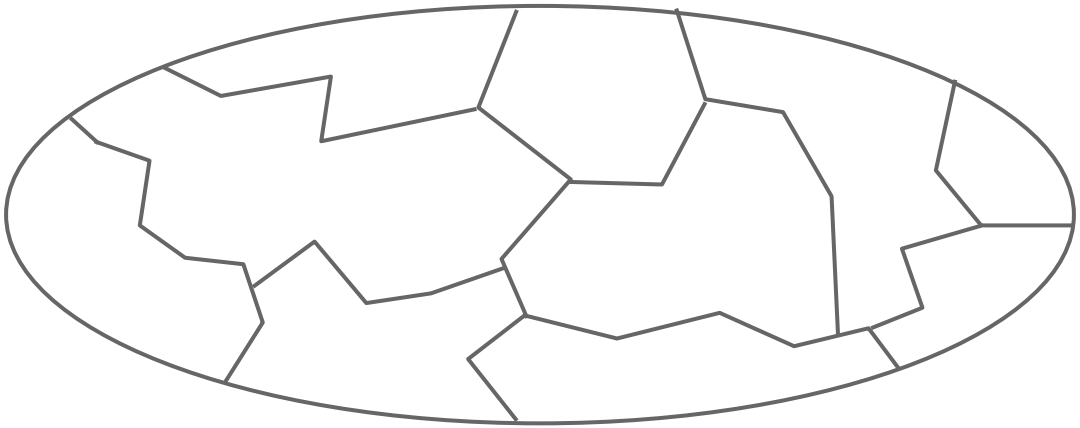
Game Spaces: Hub

- Central hub connects to areas of interest
- Examples
  - *Banjo Kazooie*
  - *Diddy Kong Racing*
  - *Super Mario Galaxy*



Game Spaces: Divided Space

- Irregularly divide world into sections
  - Most like a real map
- Examples
  - *Zelda: Ocarina of Time*
  - *Darksiders*
  - *Metroid Prime*



Conclusions

- No formula for a fun game
- Try to maintain *balance* and allow players to stay in the *flow channel*
- Make prototypes
  - Paper or electronic
  - No other way to get a sense of a mechanic's feel
- Playtest
  - Trust people's gut feelings, not their suggestions

Platformer Week 4

Platformer: Week 4

- Make your platformer into a game
- Minimum requirements
  - An achievable win condition
  - Some form of enemies / obstacles
    - At least one enemy must use your pathfinding
  - In-game UI
- Playtesting delayed until Friday, April 6
  - Remember to submit a bash script if your game requires changing environment variables

Game UI

- Standards vary across genres
  - Follow conventions of the genre
    - e.g. In an RPG, show common spells in a toolbar the bottom of the screen
- Avoid excess UI
- Main objectives
  - Usability
  - Beauty

Game UI: Guild Wars

- Conveys tons of information
  - But can get cluttered very quickly...

Game UI: Dead Space

- Embed UI in game world
  - Player's health visible on backpack

Game UI: Journey

- As minimalistic as possible
  - Doesn't need UI at all...

Game UI: Orthographic Projection

- Map OpenGL coordinates to pixel coordinates
- Use orthographic projection

```
GLint view[4];
glGetIntegerv(GL_VIEWPORT, view);
glMatrixMode(GL_PROJECTION); glPushMatrix();
glLoadIdentity();
glOrtho(view[0], view[2], view[1], view[3], -1, 1);
glMatrixMode(GL_MODELVIEW); glPushMatrix();
glLoadIdentity();
... // drawing code
glMatrixMode(GL_PROJECTION); glPopMatrix();
glMatrixMode(GL_MODELVIEW); glPopMatrix();
```

Game UI: Health Bars

- Want health bars above enemies
- Project points from world to screen space
  - Use gluProject
- To project *p* into screen space

```
int view[4]; double model[16], proj[16];
glGetDoublev(GL_MODELVIEW_MATRIX, model);
glGetDoublev(GL_PROJECTION_MATRIX, proj);
glGetIntegerv(GL_VIEWPORT, view);
double sx, sy, sz; // screen-space coordinates
gluProject(p.x, p.y, p.z, model, proj, view, &sx, &sy,
&sz);
... // drawing code using sx, sy, and sz
```

Game UI: Health Bars

- What if enemy is behind the player?
- *gluProject* will return a z value greater than 1
  - Don't draw health bar in this case

Final Project

Final Project: Overview

- Open-ended
  - Develop both an engine and a game
- Work in teams of 2-5
- Bag of topics
  - Split into major and minor topics
  - Each group member must implement one major topic
- Proposal

Final Project: Bag of Topics

- Example major topics
  - Triggers / scripting system
  - Networking
  - AI planner system
  - Level editor
  - Advanced graphics
  - Portals

Final Project: Bag of Topics

- Example minor topics
  - Particle system
  - Basic audio system
  - Terrain generation
  - Gravity volumes
  - Integrating existing library like ODE

Final Project: Initial Proposal

- Must be completed by everyone individually
  - Even if you already have a group
  - Everyone must submit their own idea
    - This is an exercise in game design
- One page with two sections
  - Engine feature
  - Game idea
- Due next Friday, right before break
- More details in handout

Final Project: Proposal Presentations

- Some day before Friday April 6th
  - What day works best? Monday?
- 2-3 minute pitch of initial proposal
  - Opportunity to get feedback and find group members
  - If you already have a group, only one member needs to present

Final Project: Final Proposal

- More detailed writeup
  - Done once per group
- Three sections
  - Engine features
  - Engine integration
  - Game idea
- Due the Friday after break
- More details in handout

Final Project: More Details

- Weekly checkpoints
  - Final project handout will be released next week with more information
- Playtesting in class on April 27
- Playtesting notes from public playtesting due May 4
  - Playtest at least 10 people per group member
  - Take notes for each playtester
- Final showcase on/around May 11
  - Open to the public
  - Exact date not yet finalized

Final Project: Achievability

- Consider your strengths and weaknesses, as well as time constraints
  - Probably will not be visually stunning
- Most time will be spent on technology, not content
  - Use procedural generation where possible
  - No MMOs or large-scale RPGs!
- Adjust project scope based on team size

C++ Tip of the Week

- C-style casting
  - Can do several things at once
  - Hard to distinguish intent from mistakes

```
(T)ptr
T(ptr)
```

- C++-style casting
  - Different options help to explicitly state intent
  - Some additional behavior that C-style casts lack

```
static_cast<T>(ptr)
const_cast<T>(ptr)
reinterpret_cast<T>(ptr)
dynamic_cast<T>(ptr)
```

C++ Tip of the Week

- static_cast
  - Converts between compatible types (both up and down a class hierarchy)
  - Can cast pointers to and from void *
  - Can't cast away const

```
struct Base {};
struct Derived : Base {};
struct Other {};

Base *base = new Derived;
Derived *derived = static_cast<Derived *>(base); // ok
Other *other = static_cast<Other *>(base); // error
```

C++ Tip of the Week

- const_cast
  - Can cast away const
  - Modification is undefined if the original variable (the target of the pointer) was declared const

```
int a1 = 1;
const int *a2 = &a1;
int *a3 = (int *)a2; // ok but could be a mistake
int *a4 = static_cast<int *>(a2); // error
int *a5 = const_cast<int *>(a2); // ok and explicit

const int b1 = 2;
int *b2 = const_cast<int *>(&b1);
*b2 = 3; // undefined behavior
```

C++ Tip of the Week

- reinterpret_cast
  - Directly reinterpret the bits of one type as another
  - Mostly used for manipulating internal representations

```
int i = 0x7f800000; // bits for positive infinity
float *ptr = reinterpret_cast<float *>(&i);
float &ref = reinterpret_cast<float &>(i);
long bits = reinterpret_cast<long>(ptr);
```

C++ Tip of the Week

- dynamic_cast
  - Does any pointer-to-pointer cast between two types that have v-tables
  - Checks validity of cast at *runtime*, returns NULL on failure for pointers
  - Uses run-time type information (RTTI)

```
struct A { virtual ~A() {} };
struct B : A {};
struct C : A {};

A *a = new B;
B *b = dynamic_cast<B *>(a); // b != NULL
C *c = dynamic_cast<C *>(a); // c == NULL
```

C++ Tip of the Week

- C-style casts
  - Defined as the first of the following that succeeds:
    - const_cast
    - static_cast
    - static_cast then const_cast
    - reinterpret_cast
    - reinterpret_cast then const_cast

References

Booth, Michael. *The AI Systems of Left 4 Dead.* http://www.valvesoftware.com/publications/2009/ai_systems_of_l4d_mike_booth.pdf

Coulianos, Filip. *Pacing and Gameplay Analysis in Theory and Practice*. http://www.gamasutra.com/view/feature/6447/pacing_and_gameplay_analysis_in_.php?page=3

Griesemer, Jaime. *Design in Detail*. http://downloads.bungie.net/presentations/Griesemer_Jaime_DesignInDetail_Sniper.pdf

Schell, Jesse. *The Art of Game Design*. Morgan Kaufman Publishers: 2008.