Polished Game Development

Diana Algma
I will talk about

- How to develop a polished game
- How to polish your game
- How to optimize your game
What is polish?

- Unpolished game:
  - Something is incomplete
  - Has imperfections
  - Clunky animations
  - Sound didn’t fade off at the end of the level
  - Some small things are a bit off

- Polished game:
  - Only complaint you have: “I didn’t like it”
Angry Birds vs Crush the Castle

- **Angry Birds** (Video)
  - Cute graphics
  - Annoying music
  - Childish fonts
  - Basic UI
  - Ripoff of Crush the Castle
  - 100M-500M installs on Android

- **Crush the Castle** (Video)
  - UI gives overview of the castle under attack
  - Too serious
  - Too difficult
  - Negative actions
  - 100k-500k installs on Android
Angry Birds vs Crush the Castle

- **Angry Birds** is more polished and more successful sequel to **Crush the Castle**

- Improvements that **Crush the Castle** could have done:
  - Add characters that fire slingshots
  - Backstory
  - Variety in graphics
  - Negate difficulty
  - Positive reinforcement instead of negative impact
Polished game development

- MVP
- One at a time
- Consistency
- First impressions
When creating gameplay

“Perfection (in design) is achieved not when there is nothing more to add, but rather when there is nothing more to take away.” - Antoine de Saint-Exupery

The player can do WHAT? -> HOW?
  - Make the player’s fingers happy
  - Show the world before and after
  - Designing objects: “show, don’t tell” (+ interactions)
Game balancing

- Two elements: level 1 and others
- Lead the eye in the direction you want the player to go to
- Allow player to safely observe hazards the first time
- Balance is about how ready the player is for the next challenge
- Winning is no fun unless it’s challenging
- Extra design elements for advanced players = longevity and enjoyment for everyone
- The best way to teach someone is when they don’t know they’re being taught
Hierarchy of the best ways to learn:

- Being told
- Reading ourselves
- Audio and visual presentations
- Demonstration
- Discussion
- Practice by doing
- Teaching others

You can't get bad reviews if game journos can't get past the tutorial.
User experience

- Icons - Always user test!
- Make the route to where the user wants to go as short and fast as possible
- Design for common use case first
- Fingers again!
- Button placement
  - Hotspots
  - OK-cancel
  - Consistency
User experience

- Make something animate on every screen
- Transition between menus
- Choose the right font
- <= 4 different styles of text on any screen
- Red & green -> color blind mode
Visuals - strong compositions

- First thing the player sees
- Color palettes (big, bright, colorful = high weight)
- Leave some room between and around elements
- Depth by layers or lighting and colors
Visuals - appealing assets: silhouettes

- Silhouettes
  - Should work
  - Shows the characteristics
  - Give (negative) space
Visuals - appealing assets: colors

- Colors
  - A little color makes a big difference
  - Strong, saturated = cheerful, positive, lively
  - Pastel = light, soft
Order stuff by level of stand-out-ness

1. Gameplay information like attack animations
2. Threats - enemies
3. Player characters
4. Environment
Collision data

- 2D - compute collision maps offline
- 3D - use low-polygon mesh for collisions
UI

- Best UI - never noticed, naturally intuitive
- RRR - remove-relocate-represent
- Key equivalents and hover functionality for desktop
- Add background -> make it move -> make it move slower
- Add sky -> add sun/moon -> make those move over time and give moon phases
- Show values increasing
- Subtle animations
Clean HUD & polished end

- Show the most important stuff
- Hide stuff that you can

- For end/fail screen
  - Add fade-out/zoom-out
  - Animate words “Game over”
  - Show game animating in background
Audio

- IMPORTANT!

- Categories:
  - Music
  - In-game effects
  - Voice acting
  - UI sounds
Music

- Theme!
- Fitting within a stereotype makes it easier to sound good, but more difficult to sound great
What makes a sound?

- Consider every object and every action it has or could have
- Every interaction should make a different noise
- For example a gun:
  - Gunshot
  - Reload
  - Safety catches on and off
  - Cocking
  - Bullet ricochet (different surfaces)
  - Bullet casings dropping (different surfaces)
  - Jamming
The feeling

- Make sure pregame has music
- Know what the player is feeling and vary the music accordingly
- Theme with different variations for adapting to player
- Pause between tracks
- Changing between tracks: fade-out/fade-in or some kind of mixing
Where do I get the sounds???:S

- Use old stuff
- License existing composition
- Use work under one of the creative commons licenses
  - CC0 - do what you want with it
  - CC BY - do what you want but give attribution to the composer(s)
- In-game effects - recordings
Some links for audio

- **Music:**
  - beatsuite.com
  - incompetech.com/music
  - purple-planet.com

- **In-game effects:**
  - freesound.org
  - freesfx.co.uk
  - audiojungle.net
  - soundsnap.com
Voice acting

- Very easy to do but hard to do well
- Reasons of failure:
  - Bad script
  - Bad acting
- Bad voice acting < no voice acting
UI sounds

- Don’t make them prominent or memorable
- Might become annoying otherwise
- No variation is better here
Sound + animation

- Rhythm
- Trigger sounds when transitioning from one animation frame to another
- Transitional animations - Responsiveness vs animation cohesion
Ways to polish

- Think about your platform, for example phones:
  - Pausing when a call comes in and resuming after
  - Using vibrations
- Preloaders (or dynamically loading)
- Custom cursors
- Add cool lighting and particle effects
- Detail
- **NO BUGS!**
Easter eggs

- Secret messages
- Special resources
- References in plain sight
- Hidden rooms and levels
- Mini games
- Extra bonuses
- Foreshadowing
Optimization
Locate bottlenecks

- GPU - fillrate, lower resolution
- CPU - batches, check how many
- GPU & CPU - vertices, <100k on mobile
- Other - script? physics? - Unity profiler
CPU optimization

- 1000 triangles & 1 mesh vs 1000 triangles & 1000 meshes
  - GPU doesn’t care, CPU does
- Combine close objects
- Use fewer materials - texture atlas
- Use fewer things that cause objects to be rendered multiple times
  - Reflections, shadows, per-pixel lights
GPU optimization

- Don’t use any more triangles than necessary
- Try to keep the number of UV mapping seams and hard edges (doubled-up vertices) as low as possible
- Use compressed textures
- Generate mipmaps for textures used in a 3D scene
- Avoid using transcendental math functions and writing your own operations that are built-in
- Consider less precise floating point values on mobile
Lighting performance

- Use lightmapping
  - Runs faster
  - Looks better
- Pixel light vs vertex light
- Avoid illuminating a single object with multiple lights
- Avoid combining meshes that are far enough to be affected by different sets of pixel lights
- Use mobile or unlit built-in shaders
LOD and per-layer cull distances

- Cull small objects more aggressively
- **Level of Detail** system
- Manually set per-layer culling distances on camera
- Small objects into separate layer and use `camera.layerCullDistances`
- Use skyboxes to “fake” distant geometry
Occlusion culling

- Occlusion Culling - Unity Manual
Occlusion culling

- Set your objects to
  - Occluder Static
  - Occludee Static
- Window -> Occlusion Culling
- Bake!
Realtime shadows

- Shadow maps
- Shadow distance
- Perspective aliasing
  - Cascades
References

- “Polished Game Development: From First Steps to Final Release” by Steven Goodwin, 2016
- [3 Simple Steps to Improve your Game’s Graphics](#)
- [The Challenges of Developing for PC and Mobile. Part 2: Graphics](#)
- [5 Simple Techniques To Add Polish To Your Game](#)
- [5 Important Ways to Add Polish to Your Game](#)
References

- Using a Texture Atlas to Optimize Your Game
- https://unity3d.college/2017/07/06/pixel-vertex-lights-unity3d/
- https://docs.unity3d.com/Manual/LevelOfDetail.html
- https://docs.unity3d.com/Manual/LightPerformance.html
- https://docs.unity3d.com/Manual/SL-DataTypesAndPrecision.html
- https://docs.unity3d.com/Manual/OcclusionCulling.html
- https://docs.unity3d.com/Manual/DirLightShadows.html
- OPTIMIZING GRAPHICS PERFORMANCE WITH HELP OF REAL MOBILE DEVICES
References

- Angry Birds (PC Gameplay - 1080p)
- Crush The Castle: Gameplay Part One
- 2D Map Collision Detection
- Collision Meshes
- Helix Snake’s top 50 favorite Skate 3 clips