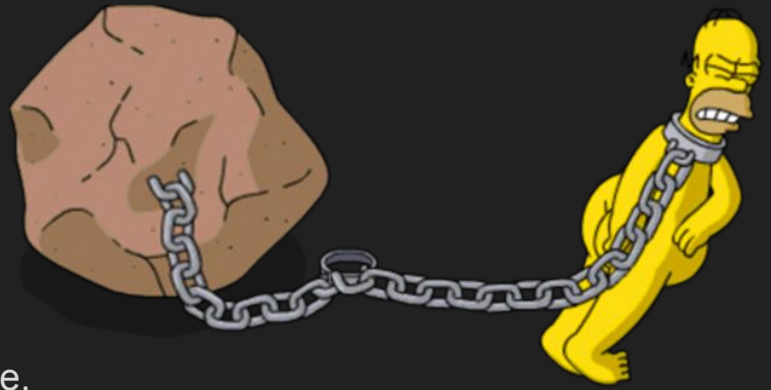
Unity Tools 1

Casual Introduction

Index

- Unity tools, casual introduction
- Unity tools state of the art
- Unity editor scripting 3D 1
- Unity editor scripting 3D 2
- Cinematics (Cinemachine + engine)
- Engine Animations 1 (Skeletal animations tools)
- Engine Animations 2 (Rigid animations tools)

Unity Tools, **why?**



TOOLS!

- Make everything in your engine more accessible.
- Save time and money.
- Improve our artists efficiency.
- Make our artists happy (ourselves too!)

Game development is **not difficult** but requires lot of **TIME**

Unity Tools, **why?**

- Unity provides basic built-in tools
- In many cases, we end up repeating many times the same step
- Example: Switching edit mode types (Hotkeys)
  - Manual mouse movement: 10s vs Hotkey 1s
- Henry Ford case: Biggest automation process ever done.
  - Unity is ford’s factory
  - We need custom tools and methodology in Unity to achieve similar results in our videogame production process.

Unity Tools

Case study

Unity Tools: Case Study 1

3D Digital Venue: Venue Tools

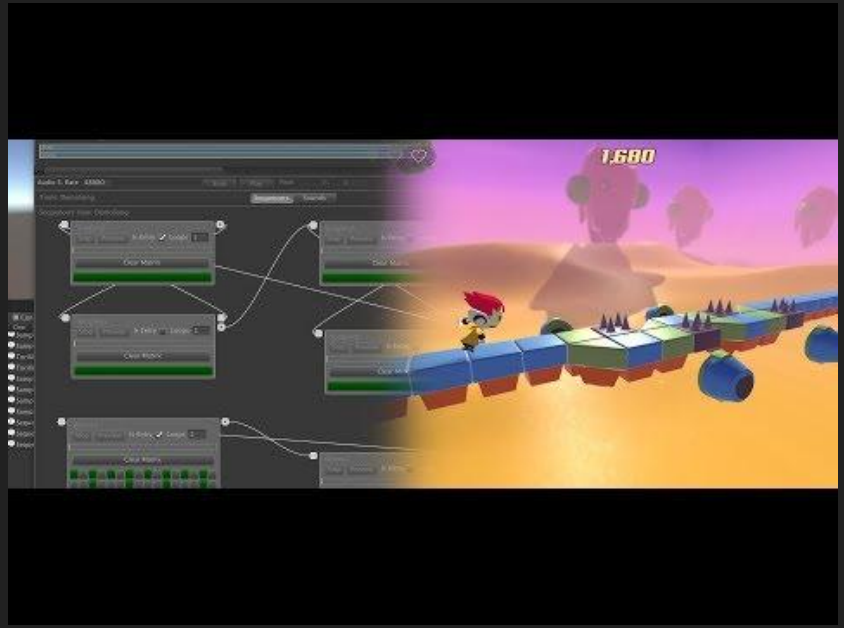
- Fast seat placement within 3d environment.
- Easy to configure the venue in unity thanks to tools and streaming assets (jsons)
- Selling point: We can make fast changes on venue configurations very fast. Very common in venues.
    - Topview generation tools:
      - Automatic polygon generation.
      - Automatic svg generation
    - Selection and edition tools
      - Edit seats materials

Venue Editor Tools v0.1a

Unity Tools: Case Study 2

Cocodrilo Dog: Tori

- Custom sound/level design tool.
- Adds control to pseudo-procedural tools
- Using tools as a selling point. Was a key point for investors/publishers to support this project.
- Indie studios also invest their resources in producing tools for the team.



Unreal Tools: Case Study 3

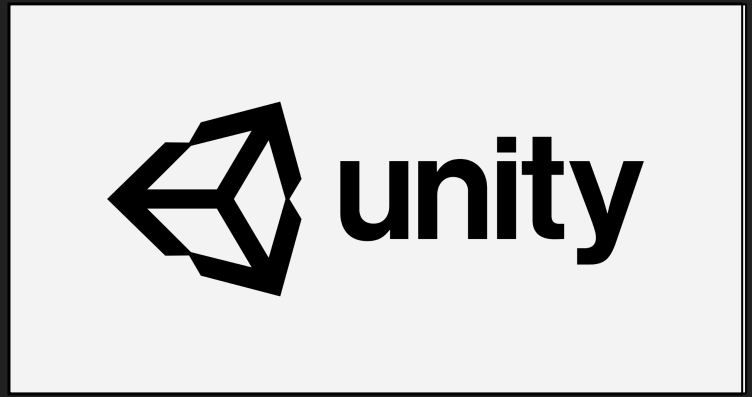
Anticto: Mutable

- Character customization system
- Started as a parallel project to the main project, steamroll.
- Helped them to continue with the development of their game thanks to the revenue that this tool gave to the company.
- Widely used by AAA games, such as Player Unknowns: Battlegrounds...

Many people make their livings thanks to tools programming with unity.

Unity 2018 3.12f1

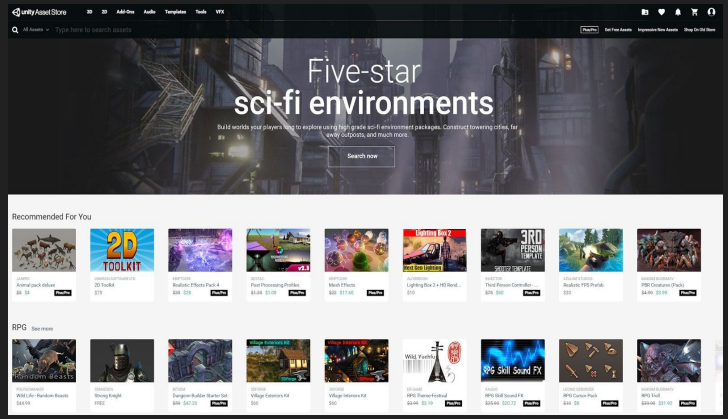
- Versions: Problems with packages compatibility.
- Lot of issues with newer versions
- In an stable video game development process, use LTS Versions!



- Target: 2018 3.12f1
- Source: https://unity3d.com/es/get-unity/download

Unity Tools: Asset store

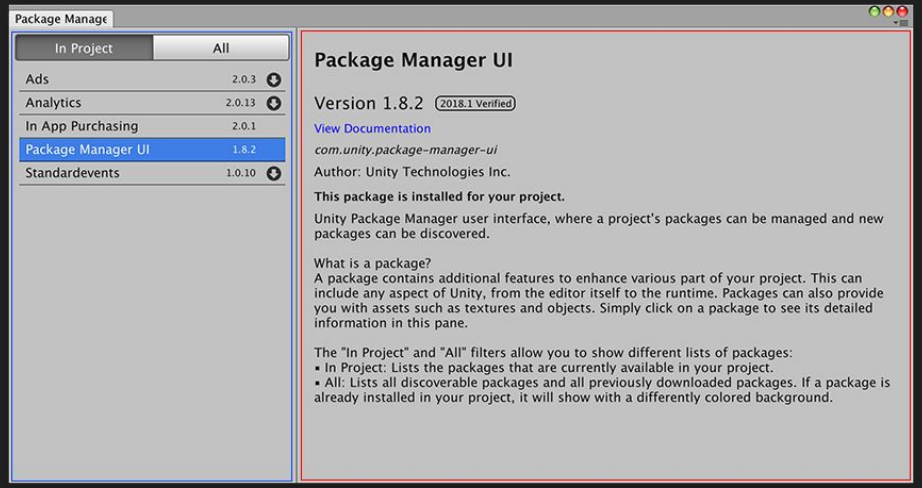
- Plenty of tools are already created
- Choose wisely, even the paid ones
- Save time and money!
  - Work time = money
  - Sometimes worth paying rather than coding your own tools!



- Everything in Unity is a package. Built in packages are already included

Unity Tools: Package Manager

- Official packages
  - Ads (mobile and f2p games)
  - Unity Analytics
  - Cinemachine
  - Oculus, ARKit and ARCore
  - Post processing Stack
  - ShaderGraph
  - Pro Builder
- Market packages
  - Behavior designer
  - GAIA
  - Console enhanced..



Unity Tools

Built-in/Custom Packages

Unity Tools: Asset store

3D Game Kit

- A 3D project designed for non-programmers.
- Artists and designers oriented.
- Collection of gameplay, tools, and systems
- Two level 3D game example
- Player controller
- Camera controller (cinemachine)
- Enemy AI, spline controller

Configure the controller to be playable with your new settings in a new scene.

Resource [GDC]: https://assetstore.unity.com/packages/essentials/tutorial-projects/3d-game-kit-115747

Unity Tools: Asset store

Kit characteristics

- The player is told how to control and move through the level. Also has hints using visual effects.
- Cinematics to guide the player
- Enemies are placed along splines
- Many visual debug elements for the gameplay
- Modular builded level
- Prefabs are widely used in the level

Resource [GDC]: https://assetstore.unity.com/packages/essentials/tutorial-projects/3d-game-kit-115747

Unity Tools: Project structure

- First step: Setup folder structure
  - Asset vs context structures
  - Big project: Context structure.
  - Small project: Asset structure.

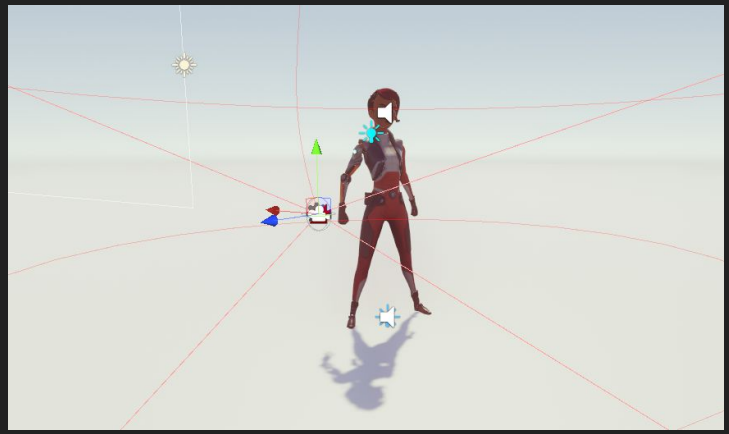
- Second step: Setup hierarchy structure
  - Logic group vs separators
  - Logic and decoration must be grouped separately.
  - Group gameplay functionality always!

Use prefabs everywhere you can!

Unity Tools: Player setup

- First step: Setup reference dependency
  - Setup the camera rig (use given prefab)
  - Adjust camera rig position relative to the player
  - Setup scene layers.

- Custom player: Hints
  - Keep every functionality separated.
    - E.G: Player input, player controller.
  - Build a FSM machine for player logic
  - Use scriptable objects to maintain states.
    - Each state as scripted object
  - If needed, built different controllers and different prefabs. (e.g my player can transform into an animal)

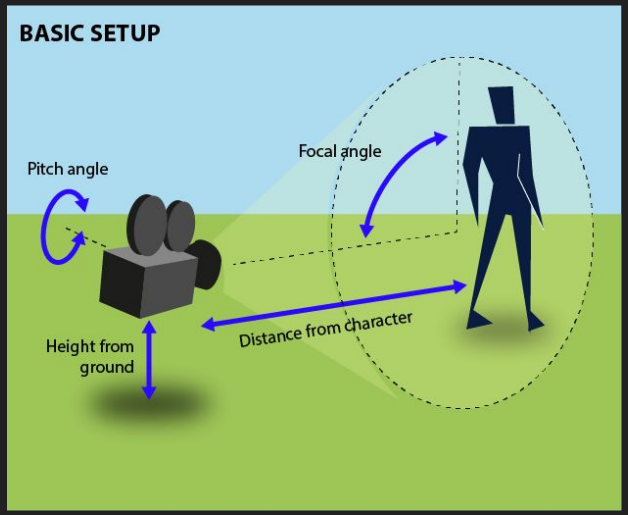


Unity Tools: Camera setup

- Camera is a key element in videogames
- Different settings, different emotions
  - Camera distance offset
  - Camera field of view
- Use free move zones to give some “freedom to the player”.
- It must be very clear which type of camera we want before designing our level.

- Example:

Third person shooter camera
VS
Third person platformer camera

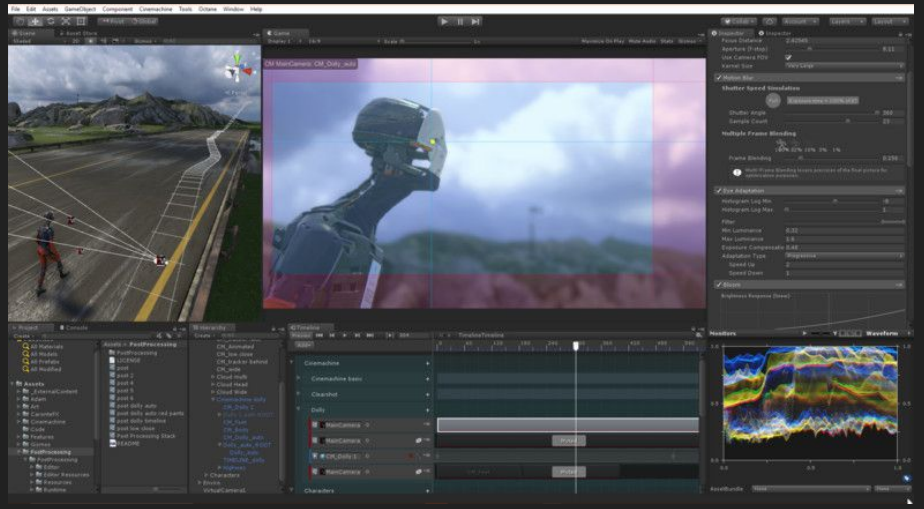


Resource:
https://www.gamasutra.com/blogs/YoannPignole/20150928/249412/Third_person_camera_design_with_free_move_zone.php

Study different videogame cameras and design your camera according to your target.

Unity package: Cinemachine

- Used to build gameplay and cinematic cameras
- Components:
  - Camera brain
  - Camera Controller: Rigs
    - TopRig
    - MiddleRig
    - BottomRig
    - Noise
  - Colliders: Used to limit camera movement

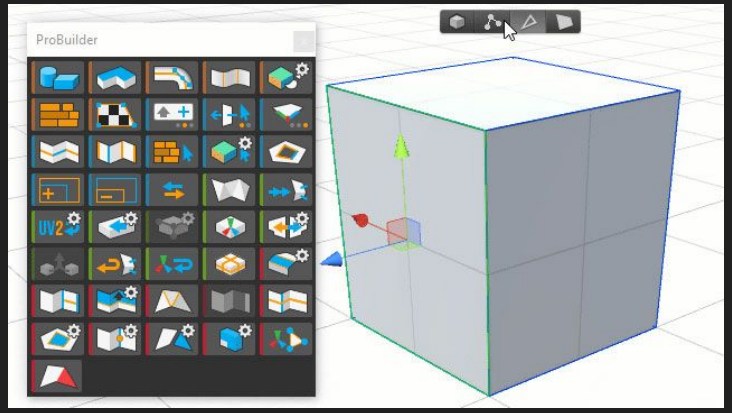


Asset Store: Cinemachine

- Free look camera
  - Used as normal thirdperson controller
  - Follow/ look-at a given transform (the player)
- Track Dolly camera
  - Camera to follow a path (e.g Silent Hill)
  - Different cameras and shots can be blended together
- Clear shot camera
  - Get a clear and useful shot of your focus point
- State driven cameras
  - Allow us to swap between different cames given player states (animations)
  - Very useful when working with player states.

Asset store: Pro Builder

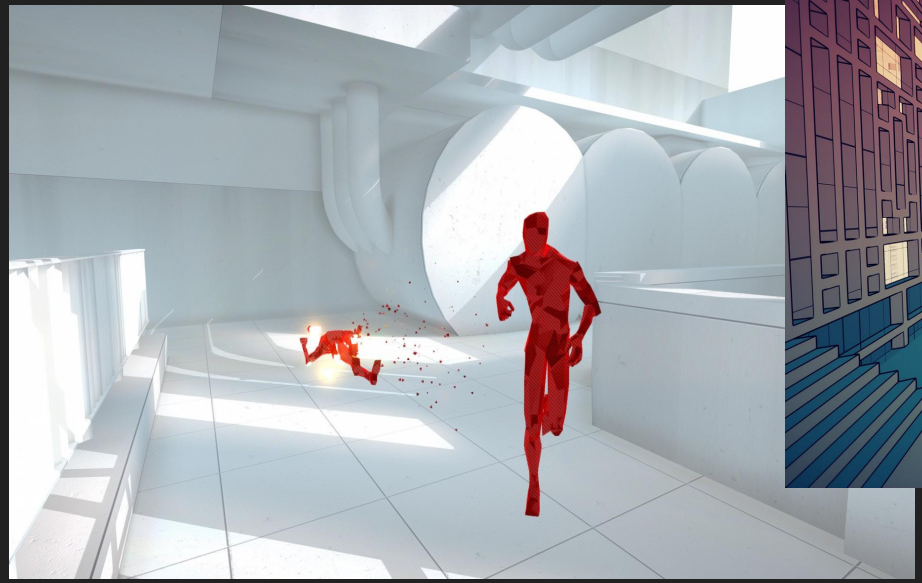
- Very useful for prototyping
- Easy to implement level design
- 3DSMax like modeling edition
- You can build, edit, and texture geometry
  - Create/Edit meshes
  - Spline/Bezier geometry
  - Edit materials and shaders
  - Scripting API
- Extended with ProGrids: Snapping.

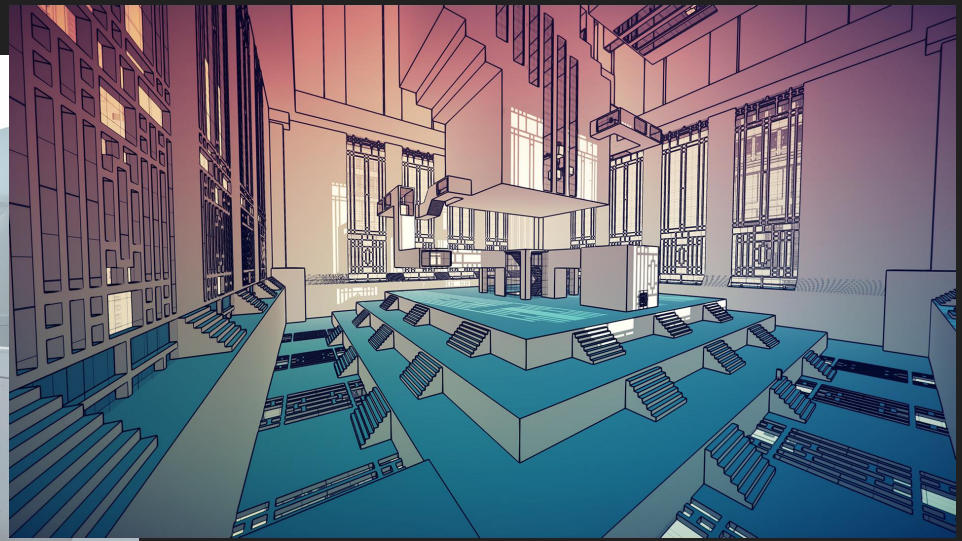


The tool can be extended, very useful to work with selections

Asset store: Pro Builder

- Examples





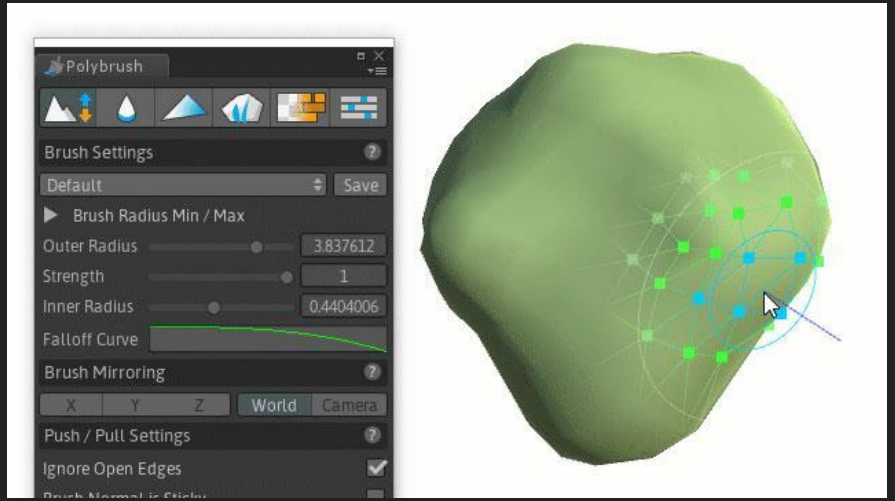
Asset store: Pro Builder

- Brown accent: Shape and mapping options (uv, texture…)
- Red accent: geometry actions (bevel, flip, inset, detach…)
- Blue accent: selection tool (grow, invert, loop…)
- Green accent: object actions (mirror, merge, flip, subdivide)

- Build a level with pro builder tools (e.g racetrack, jump level..)
  - Use only the tools explained at class
  - There are entire game levels made only with pro builder (But not recommended)
  - Manifold garden, Superhot VR, Gladiabots…
  - Specially lowpoly games, super handy!

Asset store: Poly Brush

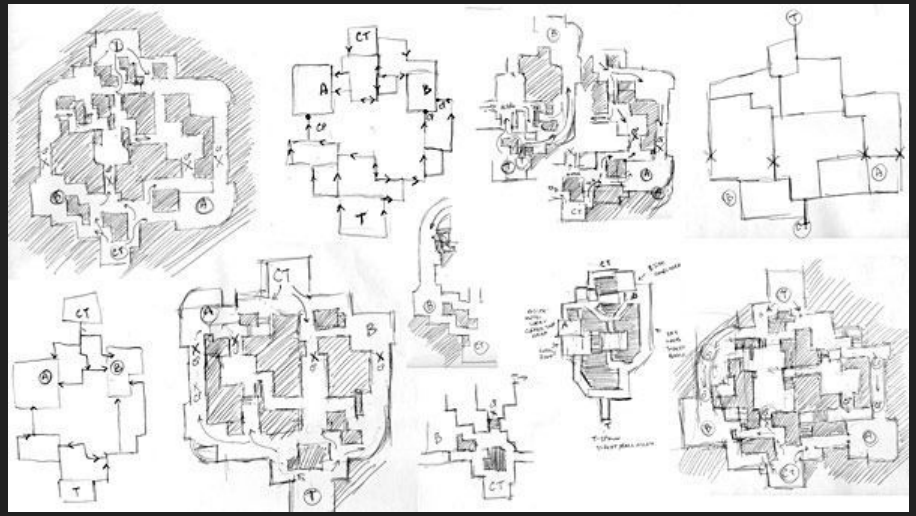
- Brush terrain editor
  - Very easy to work together with pro builder.
  - Can be used to spawn prefabs too
  - Used to easily Paint textures into complex meses.



Asset store: Pro Builder

Don’t jump directly into pro builder

Mockup first!



Resource [GDC]: https://www.youtube.com/watch?v=iNEe3KhMvXM

“THE WHITEBOARD TEST”

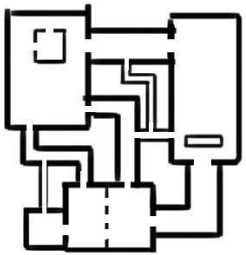
Level designers aspiring to work in the game industry are often asked to perform a “whiteboard test” – to sketch out a first person multiplayer level and explain it.

Below are some examples, and what they might say about you. (To “pass”, you might have to be able to sketch at Advanced or Expert level)

A lot of this “progression” is tied to advances in CPU and GPU power. **Also, I’m not saying the right-most is the best level, but it’d be the most “well-crafted.”**

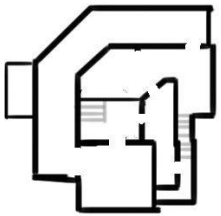
BEGINNER
over-complicated structureless mess, has probably never tried to build this in a level editor tool, no clear flow or differentiation... no attempt to break lines of sight, no really discernable patterns

scrawled on the back of your history textbook during algebra class



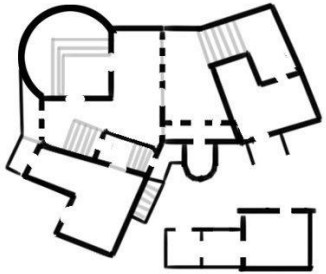
INTERMEDIATE
understands design theory but applies it very literally... very boxy floorplan that unimaginatively breaks line of sight, results in “room-hallway-room” syndrome... maybe you’re an avid modder or a student with a lot of talent and potential, but you still need a lot more practice

huge Half-Life 1 Deathmatch levels your first Natural Selection map



ADVANCED
simpler, more memorable floorplan, good use of patterns, good mix of narrow vs. open differentiation, and nice mix of height with stairs... but matchy-matchy 45 degree bends feel a lot like a “video game level” (though many designers and games would never really care about that)

graybox in a vacuum, probably sci-fi, Quake 3 levels, “pure gameplay”



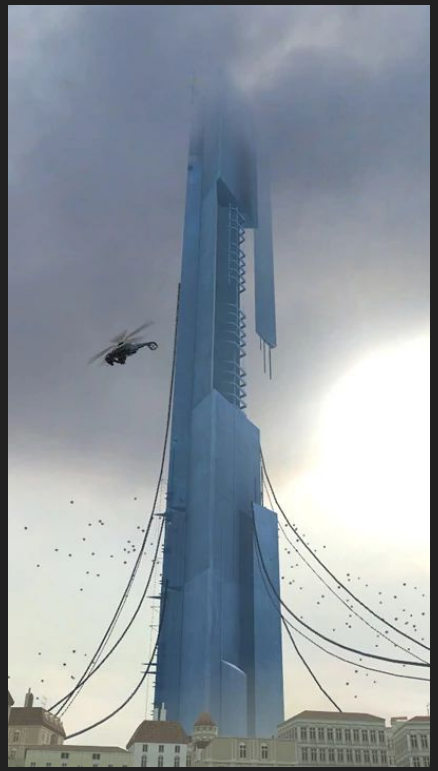
EXPERT
meaningfully breaks from grid, clear research of real-life buildings, imbues spatial differentiation with cultural differentiation, “tells a story”, careful use of symmetry... still relatively simple and memorable floorplan, still breaks line of sight... still a “video game level” but it has non-abstract internal logic to it

an Uncharted 4 level, recent CS:GO maps, usually anything overtly concerned with a narrative aesthetic of photorealism and an art budget to go for it

“The Whiteboard Test” by Robert Yang (@radiatoryang) www.debacle.us

Asset store: Pro Builder tips

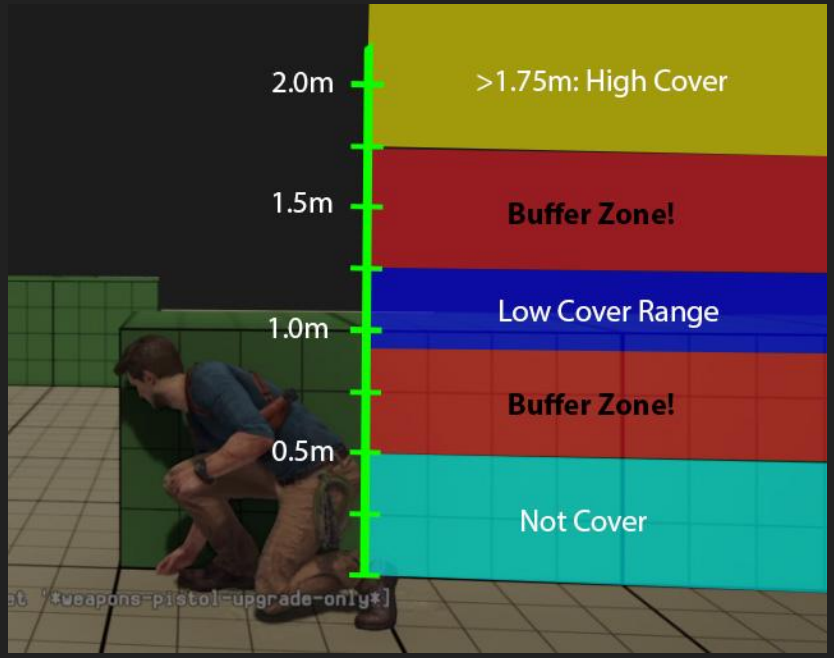
- Set metrics (relative to player size)
- Use defined tile sets
- Guiding lines
- Use the s curve method.
- Keep it wide
- Use unique points in your level as reference
- Color codes and symbols.
- Teach mechanics.



Resource: http://designreboot.blogspot.com/2009/03/level-design-primer-s-curve-variations.html

Level design tips

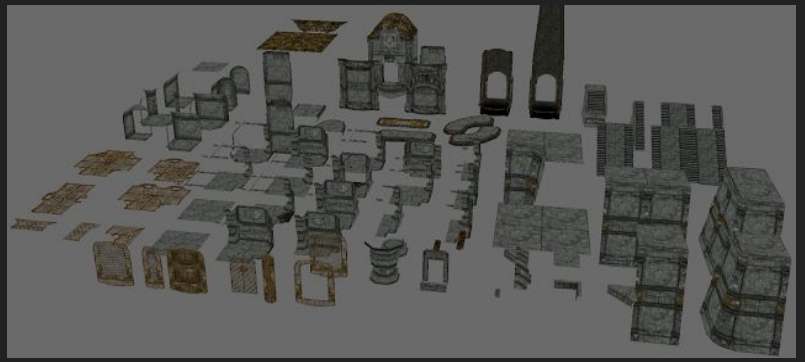
- Metrics: player size related.
  - Buffer element sizes related to player volume.
  - Mechanics MUST be taken in count in order to design your metrics.
  - Preserve primitive shapes.
  - Use this parameters to create your level tile sets.



Resource: https://80.lv/articles/defining-environment-language-for-video-games/

Level design tips: Tile sets

- Tile sets are very useful when working with modular level design.
- The artists can start building its assets with only knowing the metrics.
- Use some utilities to help you to place them.

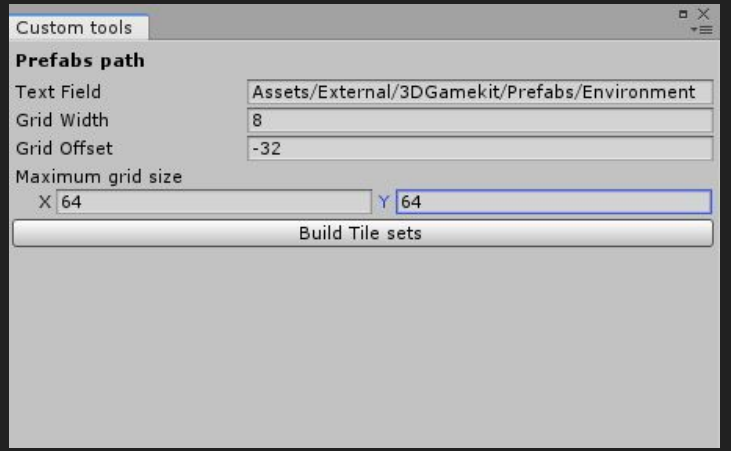


We want to have all props visible to have metric proportions

http://www.gamasutra.com/blogs/JoelBurgess/20130501/191514/Skyrims_Modular_Approach_to_Level_Design.php

Exercise: Tile Set Loader

- Build an editor window
- Define number of prefabs per row
- Define prefab grid size
- Define offset per prefab

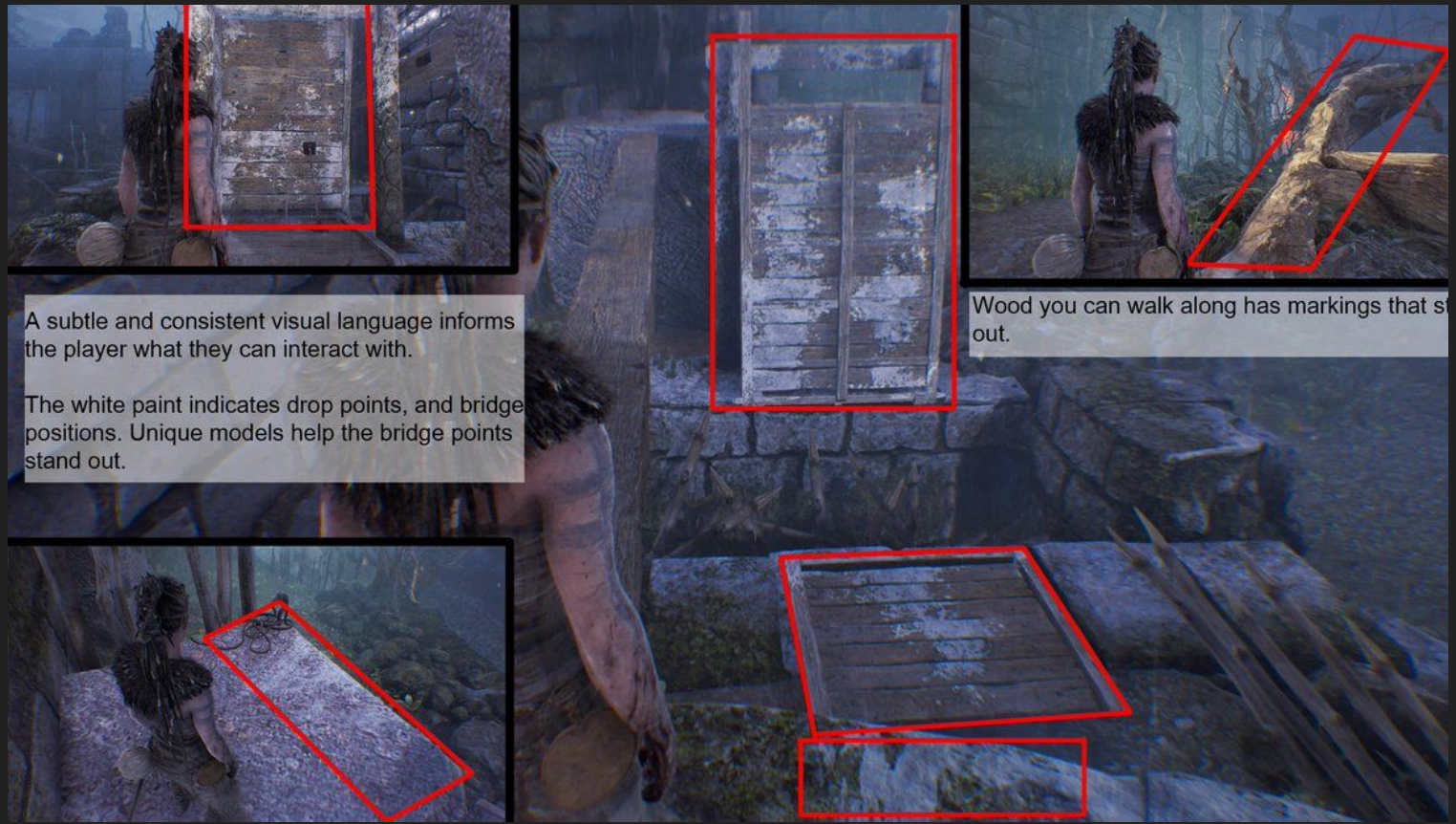


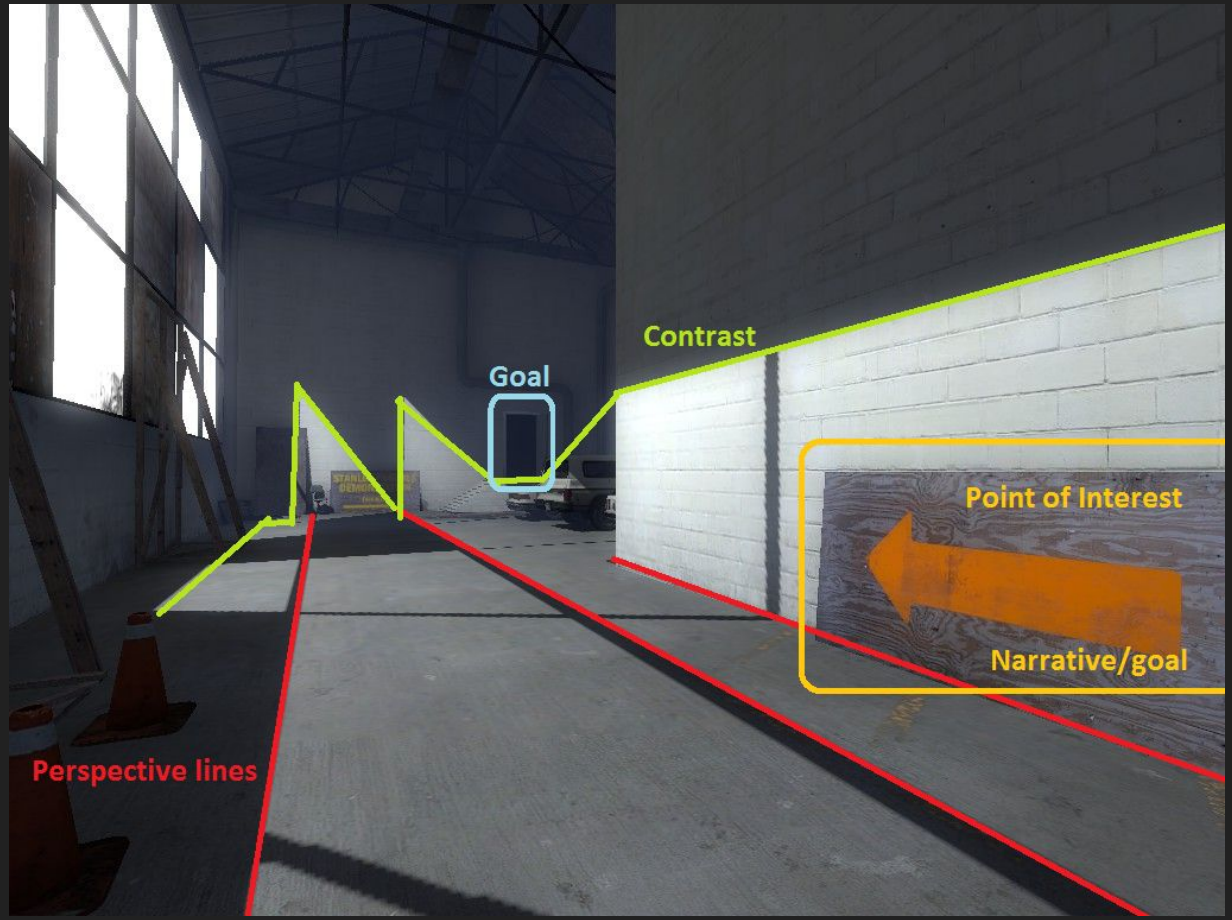
Code provided: EditorPrefabLoader.cs

Level design tips: Guiding lines

- Helps to guide the player
- They must be used in combination with the other techniques mentioned
- Attract the attention and use it to implement some of the gameplay.

e.g: Place enemies in lit zones through the guidance path.


A subtle and consistent visual language informs the player what they can interact with.
The white paint indicates drop points, and bridge positions. Unique models help the bridge points stand out.
Wood you can walk along has markings that s
out.


Contrast
Goal
Point of Interest
Narrative/goal
Perspective lines

Level design tips

- Player guidance
  - By using elements that resemble symbols. E.g hidden arrows within the environment.
  - By using color coding: Different colors can transmit the player different emotions, fashion, hierarchy and progression.
  - Follow KISS principle. The design should be stupid enough so that it can be automatically identified by the user.

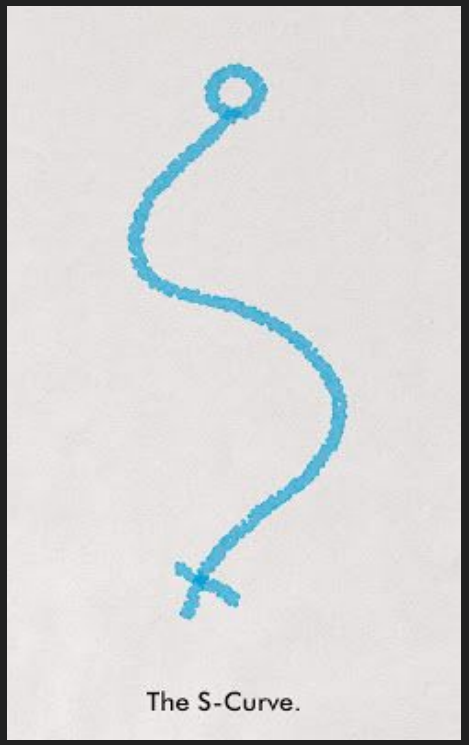
Resource:
http://www.gamasutra.com/blogs/HermanTulleken/20150729/249761/Color_in_games_An_indepth_look_at_one_of_game_designs_most_useful_tools.php

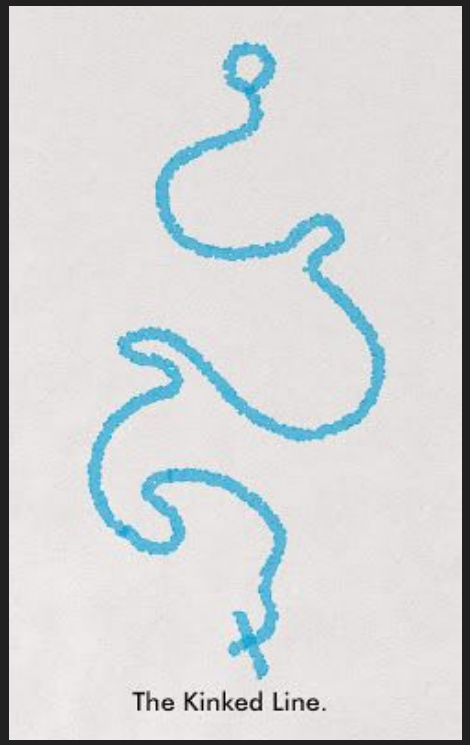

P


P

Level design tips

Denial and reward





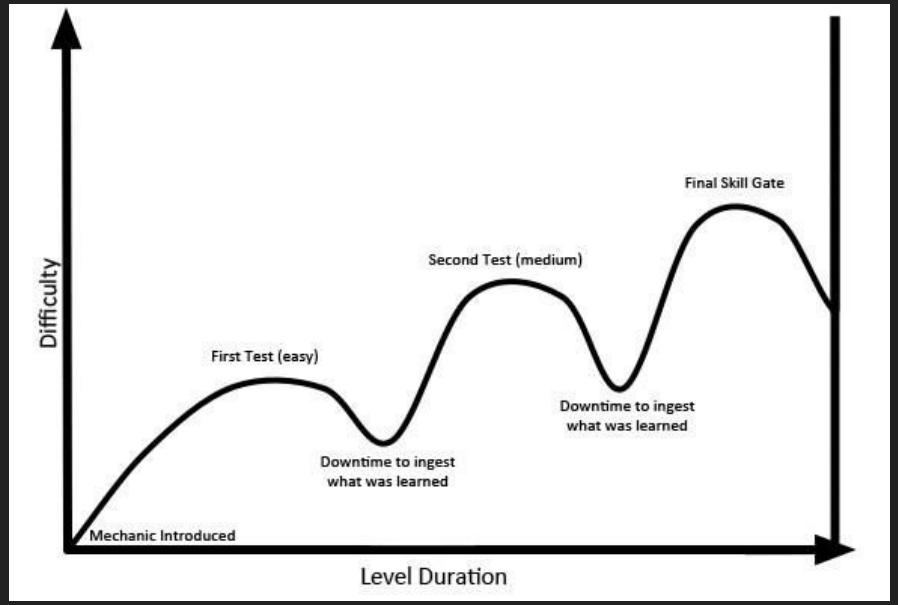
Level design tips

- Use landmarks
  - Must be visible from every part of the map
  - Used by the player to know its objective/goal (player guidance)
  - Possibility to have more than one Secondary landmarks must be less visible than primary ones.

Level design tips: Learn to teach mechanics

- Pace and teach new mechanics to the player
- Key principle that usually level designers often fail to achieve.
- Should be incremental

E.g: Portal intro labs.



Level design tips

Level design order

- Paper design (build laws)
- Whiteboxing: checking that metrics and gameplay Works
- Gameplays: Add the final details and the rest of the necessary gameplay, including AI. The game must be playable from the beginning till the end.
- Visuals: Graphics + vfx + sound + cinematics…

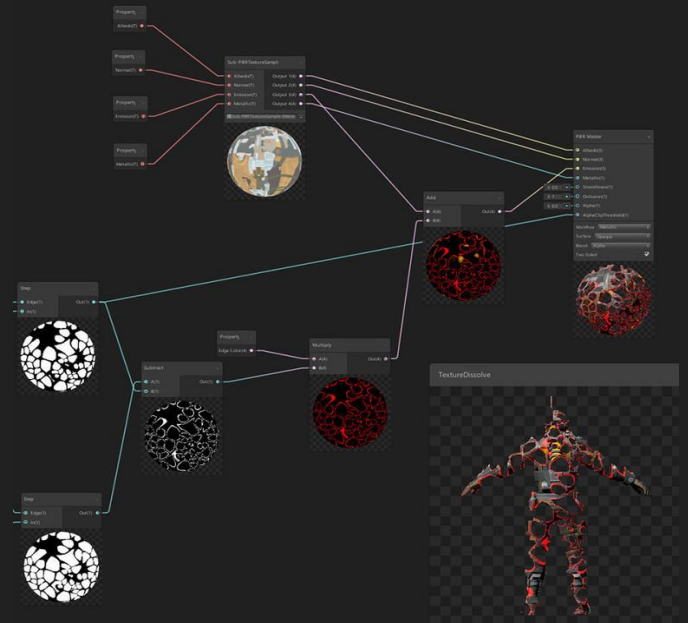
Asset store: Pro Builder

- Bonus: You can create your custom assets with pro builder
  - Create a custom shape
  - Edit Uvs
  - Texturize
  - Export
  - Not recommended

  - Exercise: Create a barrel and texture it.
  - Very useful for level desginers.
  - The artist can continue the work from the designer by exporting the blocks.
  - Can also create simplified colliders too.

Asset Store: Shadergraph

- Procedurally alter your surface appearance
- Warp and animate UVs
- Modify the look of your objects using familiar image adjustment operations
- Change your object's surface based on information about it
- Tweak a shader's visuals using the Material Inspector.
- Share node networks between multiple graphs and users by creating subgraphs



Resource: https://github.com/TwoTailsGames/Unity-Built-in-Shaders

Asset Store: Shadergraph

- Setup process:
  - Needs the Light Weight Render Pipeline
  - Supports also High Definition Render Pipeline

- Update your project:
  - Setup Lightweight pipeline at graphics assets
  - Update your project materials to the new setup.
  - Build a tool to update your custom shaders (bonus)
  - Create a pbr shader graph.

Asset Store: Shadergraph

- Create a fade fresnel effect
  - Make it fluctuate over time
  - Allow to change color and speed in inspector
- Create a geometry modification effect
  - Make it translate over time
  - Make it rotate over time
- Create a simple dissolve effect
  - Use a noise texture
  - Dissolve it with a shader parameter

Exercise:

- With the tools provided in class do: (in pairs)
  - Design a new level for your game with the level design concepts seen at class.
  - Draw the mockup of the level
  - Build the prototype of the level using pro builder tools
  - Add the player prefab to the scene as seen in class
  - Add a goal to the level

- Bonus:
  - Create a shader with shadergraph to the pickup items.
  - Add a cool postprocessing settings to the camera

Resources

- https://anticto.com/mutable/
- https://assetstore.unity.com/packages/essentials/tutorial-projects/3d-game-kit-115747
- http://devmag.org.za/2012/07/12/50-tips-for-working-with-unity-best-practices/
- https://www.gamasutra.com/blogs/YoannPignole/20150928/249412/Third_person_camera_design_with_free_move_zone.php
- https://www.youtube.com/watch?v=iNEe3KhMvXM
- http://designreboot.blogspot.com/2009/03/level-design-primer-s-curve-variations.html
- Resource: https://80.lv/articles/defining-environment-language-for-video-games/
- https://www.gamasutra.com/blogs/TomPugh/20181022/329044/Level_Design_Tips_and_Tricks.php
- http://www.gamasutra.com/blogs/HermanTulleken/20150729/249761/Color_in_games_An_indepth_look_at_one_of_game_designs_most_useful_tools.php