

Level Designing In Games

Sujoy Bit, Souvik Chakraborty

Swami Vivekananda University

sujoybit3d@gmail.com, souvikc@svu.ac.in

Abstract

Level Design in gaming world refers to the building the platform where the game is to be played. It not only includes the environment but also the character, assets, colors, effects, in a nutshell it involves everything. It is the levels which we play in a game and with progression of the story and as well as the character development; multiple levels are created. Thus it is needless to say that the game levels are the most fundamental area of a game.

Key Words

Levels, Game play, Game Design Document, Neuceno, Save Points, Check Points

Introduction

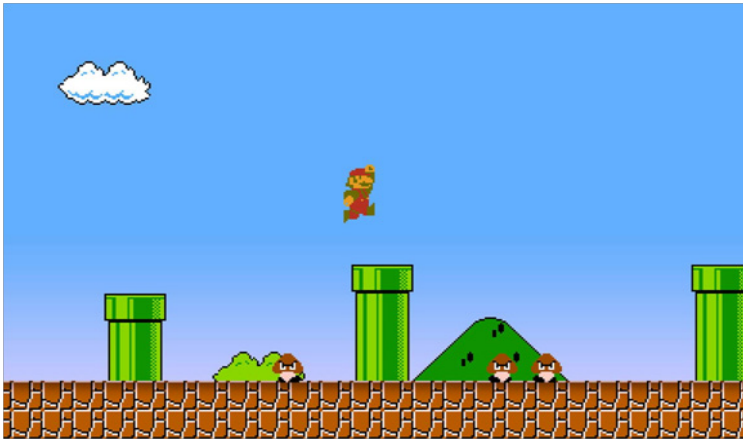
Level design is the phase of the game development process that deals with creating the stages, maps, and missions of the game. The overall purpose of level design is to create interactive situations or events within the game environment to challenge the player and keep them engaged. It is sometimes used as a synonym for environment design. While at many smaller game studios, the same designer is often responsible for both, these are two distinctive terms: Environment design only focuses on the background and scenery composition.

Level design brings together all elements of the game to shape the player experience: game mechanics, game play, obstacles, story, and so on.

The evolution of level design

Game levels as a concept are a remnant of the early games in which the player had to move from one side of a level to the next while avoiding or squashing threats. The levels would serve as a tool to mark progress and gradually increase difficulty. They were also a necessity due to the technological limitations of the time.

In the *early 1980s*, level design for 2D games like Super Mario Bros. or Sonic mainly involved the placement of various obstacles, power-ups, and enemies on a simple map.



Images from Super Mario

With the advent of 3D games in the *1990s*, level designers working on games like Doom and Quake began creating mazes-like maps for the player to explore, assigning enemies and items to the appropriate positions.

In *recent years*, game level design has evolved into a much more complex process, requiring skills from architecture, art, programming, scripting, psychology, and graphic design.



Image from Elder Scrolls Skyrim

Today, the job of a level designer involves a wide range of tasks, including creating huge, detailed environments, placing items and enemies, scripting their behavior in response to events in the game, as well as analyzing players' behavior and manipulating it in order to create an ideal experience.

Elements In Level Design

3Cs Generally, a level designer will adjust and improve the camera, character, and control elements of a game while developing a test "gym" to support the game designer's efforts. This procedure can help a level designer comprehend how the three Cs are being set up in the game, which will ultimately aid to set the scene for the important metrics. When you are teaching and demonstrating these moves to the player, you should look to provide feedback to the game creators in order to enhance the experience.

The camera

The camera gives the player the perspective they will use to play the game and aids in driving your metrics.

Here are a few instances of how a level designer might take the camera into account during production.

To prevent clipping when the player passes through doors, use the camera's height.

Using the camera and field of vision (FOV) to show the player's point of view and how the game's scenes will be framed.

Analyzing the benefits and drawbacks of various viewpoints: In a first-person perspective, the player is only looking one direction, which restricts their ability to observe opponents sneaking up from behind. In contrast, in an isometric game, the player usually sees all around them.

Personality

Because it refers to the character's abilities, weaponry, and other distinctive characteristics, the character element is significant to a level designer. In the end, this is your arsenal for determining how the player will interact and explore your levels.

Characters and environments are given a sense of scale via the camera position and field of view. This is even more crucial in virtual reality since you are immersed in the surroundings. Use the tutorial to learn more about virtual reality. ©. 2023. Unity. Technologies 12.0f.112. | unity.com VR

Beginner: The Escape Room

Throughout the character's evolution, keep track of any changes to their powers and other characteristics, as these may affect how the level is played.

Here are some examples of how a level designer can consider the character.

Consider the following.

The character's talents and how to gradually teach and demonstrate them to the player.

The character's strength at any given point in the game and whether or not the difficulties at different levels are appropriate for their degree of strength and power.

— Character customization choices for the player and their style of play, such as hacking, run-and-gun, stealth and soon.

— The player’s ability to double jump. How far and how high can they jump, and are you making good use of their abilities in your designs?

Affordance- The characteristic of an object that indicates its potential applications or clarifies how it can or ought to be utilized.

Artificial Intelligence- Artificial intelligence, or NPCs, are in-game entities whose functionality depends on computer programming rather than human interaction.

Blockout- A method of making extremely basic environment designs with rudimentary geometric shapes and game art (mockup-like).

Color Coding- Using distinct colors to mark things in a design mockup as a way of identification or classification.

Preproduction- Some of the most important subjects and activities in the preproduction phase of game creation are covered in this area, including design philosophies, metrics, paper designs, and mapping out game mechanics. When carefully considered, these procedures can assist you in refining your game concept and effectively putting it into production.

Coherent Design- Developing a comprehensive grasp of the game world you’re creating—its theme, style, historical period, and so forth—leads to coherent level design. Research for these components, such as collaborating with colleagues to develop the game’s theme and narrative, as well as its main metrics and game play mechanics, will go a long way toward giving players an immersive, intricately detailed world. Coherence in writing is achieved by making sure that every sentence and paragraph flows logically and smoothly into the next. Similar to this, coherent design in game production seeks to be methodical and consistent in its delivery, with each component of the game’s design logically connected to the others to provide a comprehensive experience. Acid Nerve’s “Death’s Door” has excellent interwoven level design. The Unity Creator Spotlight interview with the team is available [here](#).

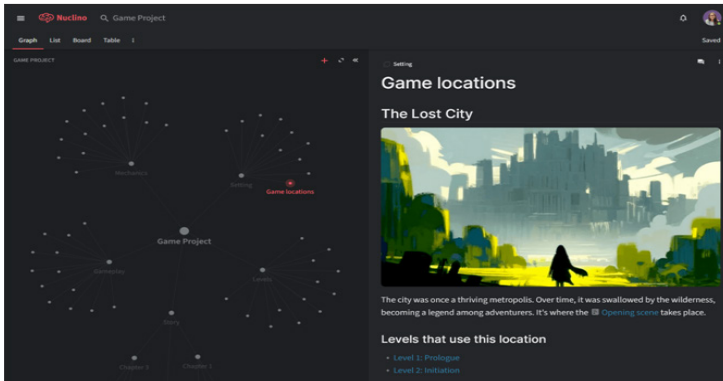


Image from Death's Door

Create a game design document

Creating a Game Design Document (GDD) is the first step of any new game design project. It starts as a single-page game concept and evolves together with your project, keeping all ideas, assets, and design decisions organized in one place.

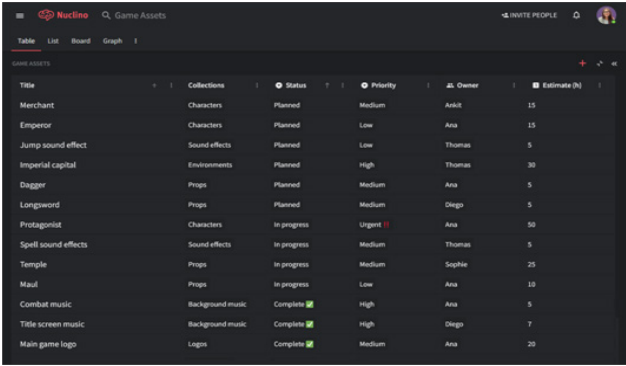
Here's an example of a game design document created in Nuclino:



Game design document example (Artwork credit: Stephane Wootha Richard)

[Nuclino](#) is a unified workspace for collaborative game design documentation, [worldbuilding](#), and [project planning](#).

Nuclino allows you to create long-form documents and organize them in a variety of visual ways. The nested list view is handy for organizing and collaborating on your [game design documentation](#) in real time. The [Kanban board](#) view is great for prioritizing features and [planning sprints](#). The table view can be used to easily sort and filter long lists of game design assets. The graph allows you to visualize the links between different topics, levels, characters, and game assets like in a [wiki](#).



The screenshot shows the Nuclino interface with a table view of game assets. The table has columns for Title, Collections, Status, Priority, As Owner, and Estimate (h). The data is as follows:

Title	Collections	Status	Priority	As Owner	Estimate (h)
Merchant	Characters	Planned	Medium	Akai	15
Emperor	Characters	Planned	Low	Ana	15
Jump sound effect	Sound effects	Planned	Low	Thomas	5
Imperial capital	Environments	Planned	High	Thomas	30
Dagger	Props	Planned	Medium	Ana	5
Longsword	Props	Planned	Medium	Diego	5
Protagonist	Characters	In progress	Urgent	Ana	50
Spell sound effects	Sound effects	In progress	Medium	Thomas	5
Temple	Props	In progress	Medium	Sophie	25
Maul	Props	In progress	Low	Ana	10
Combat music	Background music	Complete	High	Ana	5
Title screen music	Background music	Complete	High	Diego	7
Main game logo	Logos	Complete	Medium	Ana	20

Image from interface from Nuclino

Visual collaboration is also seamlessly built into Nuclino. You can add an infinite collaborative [canvas](#) anywhere and use it to create diagrams and whiteboards directly within your design document, without switching tools.

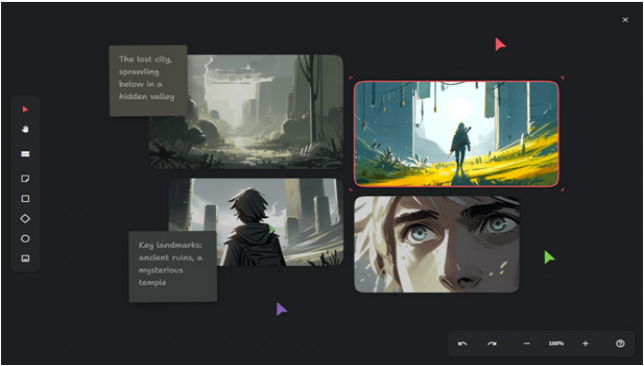


Image from interface from Nuclino

You can use it to visualize your game's core gameplay loop, capture different mechanics and interaction flows, brainstorm ideas using sticky notes, organize concept art, and more.

If you don't have a GDD by the time you begin working on the level design, the first step would be to create one. Document everything that's important about your level: goals, objects, events, a summary of the play through, puzzles, secrets, and so on.

The documentation step is important since it allows you to scope your level properly and to reduce the risk of potential problems arising during development.

White-boxing: Keeping your level at bay White-boxing, also known as grey-boxing, is the process of making and arranging basic 3D objects to determine which layout best fits the level of design and style you're going for. By keeping the white box simple, you can simply tweak the level without having to change the lighting, graphics, or other features, which speeds up the iteration process. Some level designers would rather leap straight into white-boxing rather than initially making paper ideas since they enjoy working in 3D and become accustomed to the game engine operations. You must decide which strategy is most effective for you. This guide's Part III offers detailed instructions on how to use it. To efficiently construct shapes and terrain, use Unity's ProBuilder. It offers pre-made modeling tools and common elements for game design. As an alternative, primitive 3D objects can be placed directly in the scene view.

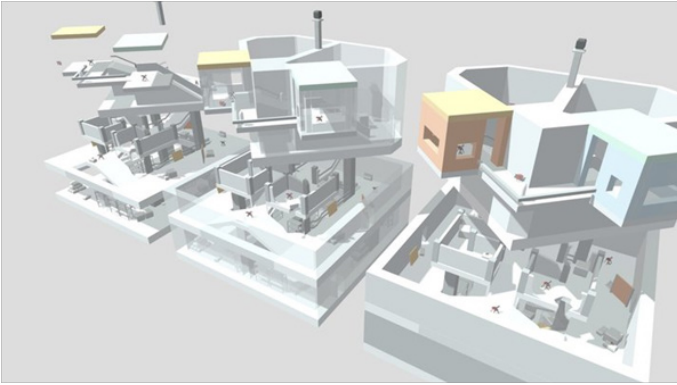


Image of 3d blocking in Level Designing

Critical Path- The critical path is the longest path that can be taken to finish a game. Consider it the route you would need to follow to test all of the content you want to ship as effectively as possible.

The well-known 2D game *Hollow Knight*, created by Team Cherry, was created using Unity. The path required to finish the 112% of the game, or the totality of the map, would be the game's crucial path.

Golden Path- The best gameplay components, awards, and/or secrets are available on the golden route.

Some games, like *TUNIC* (by TUNIC Team), include many endings, with the path needed to obtain the best ending being referred to as the "golden one." *TUNIC* is an isometric action adventure game that has numerous endings.

Since the golden road is meant to be the most appealing and enjoyable route for the player, be sure to test it with coworkers to find areas of confusion and annoyance as well as those that contribute to the content's shine.

The golden route usually includes most, if not all, of the game's systems of play so that the user can enjoy everything it has to offer. It does not, however, contain every element of the game.

For instance, in a game having numerous endings, the crucial path would involve playing through each ending, whereas the golden road would just be the most captivating, fascinating, or enjoyable way to reach the single ending.

Side quests, secrets, and shortcuts are examples of secondary and tertiary paths.

How many paths are required? You should ideally provide pathways for every play type in your game. For instance, Eidos Montreal's game *Deus Ex: Human Revolution* supports a variety of gameplay modes, including hacking, stealth, and run-and-gun. Determine the preferred play styles so that designers can create levels based on them. Furthermore, provide several approaches to a problem that depend on each style or combination.

It can be annoying to force a player to go back through a region they have already explored with little to no reward. The goal of a better player experience is to make sure that after they finish a side quest or shortcut, they can return to the action fast.

Additionally, one approach to reward and encourage the player to explore throughout the game is to add material, even if it's minor, to the end of trails and open-ended spaces.

Focus on Authenticity and Immersion

Even if your game is set in a fictional world with supernatural elements, take care to make it seem real. Immersion and suspension of disbelief are critical for the player to stay engaged and appreciate other aspects of the level design, such as gameplay and story.

The three-point rule for instructing players in mechanics

This gives your player time to become familiar with a new system or mechanic by having them repeat it several times throughout a play session. Generally speaking, the player should perform a new action or series of actions at least three times before becoming comfortable with them. Space out new mechanisms and systems to prevent gamers from being overloaded. Let's examine an illustration of the rule of three for implementing additional mechanisms in a conventional platformer:

When a player first sees a single adversary, they discover that they may defeat it by leaping on it.

The player then faces a number of adversaries after that initial encounter, and in order to destroy them, the player must jump on each of them once. The action of jumping on the enemy's head to defeat them is further cemented by this.

In the third segment, the player comes across a collection of adversaries that are separated by distance. Although the player can hop on each enemy's head to defeat them, more skill is needed this time.

The player should be aware that they can defeat both single and numerous foes by jumping on them once after these three instances.

Save Points and Checkpoints

Some games may have static save points, meaning the player must manually locate them in order to save their game. Static spawn points allow the designer flexibility over what is saved, when, and where. These are helpful in circumstances where saving the game in any particular location is too complicated for the system.

In some games, players can save almost anytime during gameplay or with certain restricted options. This usually calls for additional work and robust game code because it can be difficult to reload a save anywhere after finishing any given action.

A combination of both sorts of save points are used in many games. For example, a game may let you save anywhere, but all enemies are reset, and you spawn in a static area chosen by the game's creator rather than where you saved the game. This simplified method can be effective in complicated games, enabling players to save at any moment while maintaining a basic enough save system to reduce the likelihood of errors and glitches.

Checkpoints

Checkpoints are positioned across a level, are momentarily stored in memory, and are only available while the player advances through a mission or level.

This strategy breaks a lengthy mission down into smaller parts and makes it easier for a player to proceed by allowing a creator to place temporary saves throughout the task. Additionally, checkpoints are helpful when a sequence has multiple tasks if the quest is too complex to reload at any desired moment.

Steer clear of soft-locking

Make sure you test every checkpoint in the game to prevent the player from being soft-locked. When the player can no longer advance or rewind through the game, a soft lock occurs, frequently requiring them to load a prior save or begin the playing from scratch.

Conclusion

Thus level designing is a crucial part of the game and it is safe to say it is the heart of the game. It is the longest process both in the pre production and post production. With the above mentioned procedure we can have a level for a game, it must be noted that all the level must go through this process, thus making it the longest process.

Reference

- Deaths Door (game) for image
- Introduction to Unity- Unity Hub
- <https://jpcatholic.edu/NCUpdf/courses/DIGM203-UE4BeginnersQuickStartGuideV2.pdf>
- <https://cdn2.unrealengine.com/training-course-catalog-fall-2021-110-v1-40555f602e78.pdf>
- Mastering Game Design with Unity 2021-BNB Publication