Project Development & Geo Quest

Sebastian Grygorczuk - STEM Institute at CCNY

This Today's Agenda

- Go over how to brainstorm and conceptualize a game.
- Discuss scope and how to plan project development.
- Review **Geo Quest**, how it was built, and how it engages the player.
- Go over how to create an executable version of your games.

Project Development: Game Conceptualization

Game Idea

The first step in making a game is coming up with the idea. Not every idea will be great, so it's helpful to jot down as many as possible before refining them.

When brainstorming, consider the following:

- **Genre & Core Gameplay Loop**
  - Is it an RPG, FPS, strategy, or puzzle game? What are the key mechanics?
- **Unique Selling Points**
  - What makes your game different from others in the same genre?
- **Theme & Setting**
  - Will it be sci-fi, fantasy, historical, or modern? What kind of world will players explore?
- **Platform(s)**
  - Is the game for PC, consoles, mobile, or VR?

Examples of Genres Genres

- **Platformer** – Games where players navigate a character through levels with jumping, climbing, and other movements, often avoiding obstacles. (e.g., *Super Mario*, *Celeste*)
- **Puzzle** – Requires players to solve puzzles to progress. (e.g., *Tetris*, *Limbo*)
- **2D Fighting** – Involves one-on-one combat where players use various moves and combos to defeat their opponents. (e.g., *Street Fighter*, *Mortal Kombat*)
- **Shoot 'Em Up (Shmup)** – Involves shooting waves of enemies while dodging projectiles, often with a top-down or side-scrolling view. (e.g., *R-Type*, *Gradius*)
- **Visual Novel** – Focuses on narrative storytelling, often with multiple branching storylines and player choices. (e.g., *Doki Doki Literature Club!*, *Phoenix Wright: Ace Attorney*)
- **Rhythm** – Gameplay revolves around matching the rhythm of the music with timed inputs. (e.g., *Dance Dance Revolution*, *Crypt of the NecroDancer*)
- **Tower Defense** – Players strategically place defensive units to prevent enemies from reaching a specific point. (e.g., *Plants vs. Zombies*, *Kingdom Rush*)
- **Turn-Based RPG** – Players control a party of characters and engage in battles where actions are taken in turns, often featuring character progression and story-driven quests. (e.g., *Final Fantasy VI*, *Chrono Trigger*)

Gameplay Mechanics

Game mechanics are the rules, actions, and interactions that define how a game is played and experienced. They shape player interactions, establish challenges, and create the overall feel of the game.

- **Core Mechanics**
  - What are the fundamental actions? (Movement, combat, puzzles, crafting, exploration, etc.)
- **Game Progression**
  - How does the game unfold? (Levels, open-world, mission-based, or linear?)
- **Controls & UI**
  - How will players interact with the game? (Keyboard, controller, touchscreen?)
- **Multiplayer & Online Features**
  - Will it be single-player, co-op, PvP, or an MMO?
- **Difficulty & Balancing**
  - How will challenge levels be designed to ensure fairness and engagement?

Examples of Gameplay Mechanics

Each of these games is a platformer where the main actions involve running and jumping, but the core mechanics emphasize different aspects of platforming.

Mario: focuses on **precision jumping**, supplemented by power-ups and movement variations.

Sonic: emphasizes **speed and momentum**, enhanced by features like the Spin Dash, rings, and elemental shields.

Yoku's Island Express: relies on **pinball-inspired platforming**, indirectly moving Yoku across given stage.

Rayman: balances **platforming and combat**, augmented by gliding, unlockable abilities, and puzzle elements.

Game Goal

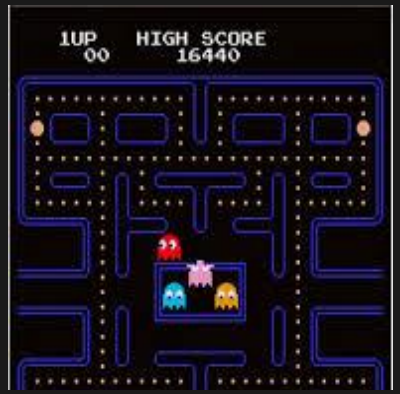
Games can share mechanics, such as platforming or combat, but their objectives shape how players engage with them.

- What is the primary goal of the game?
  - Storytelling, competition, exploration, or creativity?
- Does the game encourage replayability ?
  - High scores, achievements, or procedural content?
- How is it meant to be played?
  - The game is linear, open-ended, or dynamic based on player choices?
- How many player engage with it and how?
  - Single-player immersion, multiplayer competition, or cooperative play?

Examples of Game Goals

Mario: In most Mario games, the primary objective is to reach the end of each level while overcoming various obstacles, such as enemies and environmental hazards.

Under Night In-Birth: The goal is to defeat opponents in a series of fast-paced, anime-style 2D fighting matches, mastering combos and character abilities to win.



Pac-Man: The main objective is to achieve a high score by navigating mazes, collecting all the pellets, and avoiding capture by the ghosts.

Terraria: A sandbox adventure game where players can explore, gather resources, craft items, and defeat enemies and bosses. Since it's a sandbox, the goals are flexible, allowing players to pursue whatever objectives they set for themselves.

Setting, Narrative & Art

Every game has a setting, narrative, and art style that shape the player's experience. Consider the following:

- **What's the setting?**
  - Is it a futuristic city, a medieval kingdom, or a mysterious alien planet?
- **How much story is there?**
  - Is it a deeply narrative-driven experience or a simple backdrop for gameplay?
- **Is the game arcadey or story-driven?**
  - Will players chase high scores and fast-paced action, or will they follow a structured story?
- **What is the art direction?**
  - Will it be realistic, pixel art, cel-shaded, or low poly?

Examples of Setting and Narrative

Mario: Set in the vibrant Mushroom Kingdom, Mario is a plucky Italian plumber known for his iconic red hat and mustache, who embarks on adventures to rescue Princess Peach from the villainous Bowser.

Sonic: In a diverse world filled with lush landscapes and futuristic zones, Sonic the Hedgehog is a blue, anthropomorphic hedgehog renowned for his incredible speed and determination to thwart the evil Dr. Eggman's schemes.

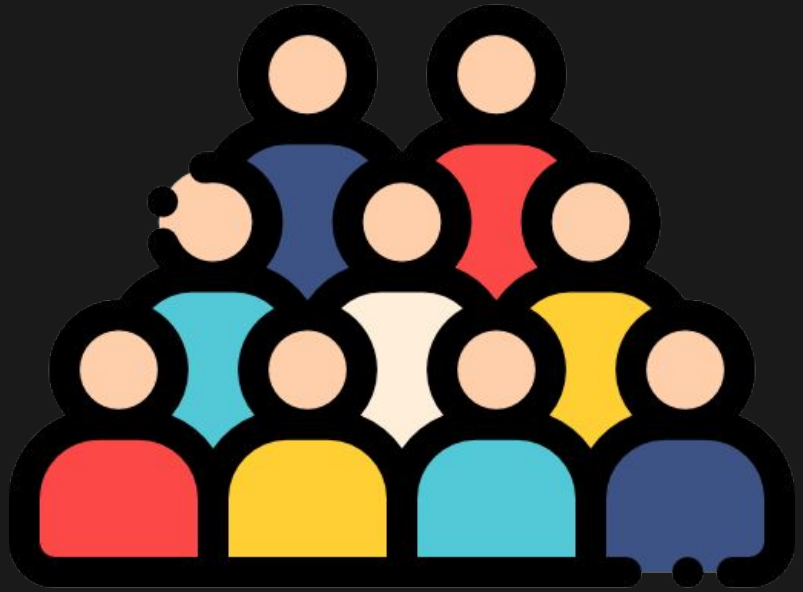
Yoku's Island Express: On the vibrant, tropical island of Mokumana, Yoku, a dung beetle turned postal worker, uses his unique pinball-inspired mechanics to deliver mail.

Rayman: Taking place in the magical and colorful Glade of Dreams, Rayman is a limbless hero with the ability to throw his fists and perform acrobatics.

Focus and Audience

Games range from cinematic, story-driven adventures to purely mechanics-focused experiences. Your game will fall somewhere in between, so it's essential to decide what kind of experience you want to create.

- **Who is the target audience?**
  - Are you making it for casual players, hardcore gamers, or a niche community?
- **What experience do you want to provide?**
  - Is it an emotional journey, a challenging skill test, or a relaxing escape?
- **Does your game appeal to its intended audience?**
  - If the experience is unclear or unappealing, it may struggle to find players.



Even if the game is just for you, defining your audience helps shape the design choices and overall direction

Examples of Focus and Audience

Celeste

Focus: **Gameplay-focused**; emphasizes precise platforming mechanics and challenging levels while weaving in a heartfelt narrative about overcoming personal struggles.

Target Audience: Primarily aimed at players who enjoy challenging platformers, as well as those looking for a touching story about mental health and perseverance.

Phoenix Wright: Ace Attorney

Focus: **Narrative-focused**; centers on storytelling, character interactions, and solving cases through dialogue and evidence presentation.

Target Audience: Targeted at fans of visual novels and puzzle-solving games, particularly those who enjoy courtroom drama and engaging narratives.

Terraria

Focus: **Gameplay-focused**; focuses on exploration, crafting, and combat in a sandbox environment, allowing players to shape their own experiences.

Target Audience: Appeals to a wide range of players, particularly those who enjoy sandbox games, exploration, and creativity, from children to adults.

Sea of Stars

Focus: **Narrative-focused**; combines classic turn-based RPG mechanics with a rich, story-driven experience featuring character development and engaging world-building.

Target Audience: Aimed at fans of retro-style RPGs and those who appreciate deep storytelling, nostalgia, and vibrant pixel art.

Project Development: Scope

Scope

Strong **organizational foundations** are essential for any project, whether you're developing a game or constructing a building.

Scope defines the size and scale of your project, varying based on time, resources, and team size.

A large team over several years can build something as complex as *Fortnite*, while a weekend project will be much smaller.

Proper scoping ensures that your game remains manageable, avoiding delays or incomplete development.

Key factors include setting a **clear timeline**, **defining project goals**, **budgeting**, and choosing the **target platform** to ensure a successful outcome.

Project Objectives and Goals

Before starting any project, it's crucial to define its **purpose and objectives**. Asking key questions helps establish a clear direction:

- **What is the purpose of the project?**
  - Understand why you're creating it and what impact it will have.
- **What problem does it solve?**
  - Identify the challenge or need your project addresses.
- **What are the key success criteria?**
  - Determine measurable goals to define when the project is successful.

By answering these questions, you create a **strong foundation** for planning and execution.

Stakeholders & End Users

To ensure a successful project, you need to define who it's for, who will create it, where resources will come from, and confirm that everyone is aligned on the final product.

- **Who are the stakeholders (clients, team members, investors, users)?**
  - Players, developers, and potential investors.
- **What are their needs and expectations?**
  - Players want a fun experience, developers need clear goals, and investors look for a viable product.
- **How will feedback be gathered?**
  - Playtesting, team meetings, surveys, and prototyping.
- **How will alignment be ensured?**
  - Regular check-ins, iterative development, and refining based on feedback.

Scope and Deliverable

Once you have a clear idea of your project and its stakeholders, you must assess its complexity and scope.

- **What are the core features and functionalities?**
  - Precise platforming mechanics, geometric art style, and multiple challenging levels.
- **What is included and what is explicitly not included?**
  - Included: basic movement, physics-based obstacles, and level progression. Not included: online multiplayer or complex AI.
- **What are the expected outputs?**
  - A functional 2D platformer with at least three levels, a polished user experience, and clear documentation for future improvements.

Timeline & Milestones

When planning the project's timeline, expect delays and track progress through milestones rather than total time.

- **What is the overall timeline?**
  - The project should be completed within three months, allowing time for development, testing, and adjustments.
- **What are the key milestones or phases?**
  - Concept & planning (Week 1-2), prototype development (Week 3-6), level design & mechanics refinement (Week 7-9), playtesting & bug fixing (Week 10-11), and final polish & release (Week 12).
- **Are there any dependencies that could affect deadlines?**
  - Availability of team members, debugging challenges, and potential scope adjustments could impact the timeline.

Budget & Resources

The project's budget is nonexistent, so all development must rely on free resources and existing skills.

- **What are the financial constraints?**
  - No budget, avoid purchases and use free assets, open-source tools, and personal expertise.
- **What tools, software, and hardware are needed?**
  - Unity (free version), Visual Studio Code, Blender for basic models, and personal computers for development.
- **What human resources (team, contractors) are required?**
  - A small team of developers, artists, and testers working voluntarily or collaboratively.

Technical and Operational Feasibility

It's important to assess whether you have the skills and resources necessary to complete the project successfully. While ambition is valuable, setting realistic goals within your current capabilities will help maintain motivation and prevent uncertainty.

- **Are there any technical constraints or risks?**
  - Limited experience with advanced mechanics or multiplayer functionality may pose challenges. Performance optimization for different platforms could also be an issue.
- **Does the team have the necessary expertise?**
  - The team should be comfortable with Unity, C# scripting, and basic game design principles.
  - If certain skills (e.g., advanced animation or networking) are lacking, alternative solutions or learning resources should be considered.
- **What infrastructure is required (cloud, on-prem, third-party services)?**
  - No external infrastructure is needed; development will be local, using Unity's built-in tools. If necessary, free version control services like GitHub can be used for collaboration.

Risk Assessment & Contingency

No matter how well you plan, challenges will arise. It's crucial to anticipate potential risks and develop strategies to address them before they disrupt progress.

- **What are the biggest risks (technical, financial, market)?**
  - Technical risks include difficulties implementing complex mechanics or encountering performance issues.
  - Financial risks are minimal due to the no-budget approach, but lack of proper tools or resources could slow development.
  - Market risks involve low player interest or competition from similar games.
- **What mitigation strategies can be put in place?**
  - Breaking the project into smaller, manageable milestones will help track progress and identify issues early.
  - Using existing free assets and tools can prevent resource limitations. Regular testing ensures technical stability, reducing major setbacks.
- **How will unexpected changes be managed?**
  - Flexibility in design will allow for adjustments if certain features prove too difficult.
  - If development falls behind schedule, prioritizing core mechanics over secondary features will keep the project on track.

Compliance & Legal Consideration

Legal concerns may not be a major issue for this project, but it's still important to ensure everything is properly handled.

- **Are there industry regulations to follow?**
  - Since this is a small-scale game, there are no strict industry regulations to worry about.
  - However, if the project expands, understanding platform-specific guidelines (e.g., Steam, App Store) will be necessary.
- **Are there data privacy concerns (e.g., GDPR, CCPA)?**
  - As long as the game does not collect user data, privacy regulations should not be a concern.
  - If online features are added in the future, compliance with data protection laws will become important.
- **Do contracts, NDAs, or licenses need to be reviewed?**
  - **The project should only use open-license art, music, and assets to avoid copyright issues.**
  - **No stolen or unauthorized content should be included.**
  - **This include AI Generated products.**

Communication & Collaboration Plan

Effective communication is key to a successful team project. Many projects fail not due to technical issues, but because of poor collaboration and unclear expectations.

- **What tools will be used for communication (Slack, email, meetings)?**
  - The team should use a mix of real-time and asynchronous communication tools.
  - Discord or Slack can be used for quick discussions, while email or Trello can help track progress and tasks.
- **How frequently will progress updates be shared?**
  - Weekly check-ins should be scheduled to ensure everyone is aligned, with additional updates as needed for major milestones or blockers.
- **Who is responsible for decision-making?**
  - While decisions should be made collaboratively, a project lead should be designated to make final calls when necessary to keep progress on track.

Post-Project Consideration

Once the project is completed, it's important to evaluate its success and determine if further support or updates are necessary.

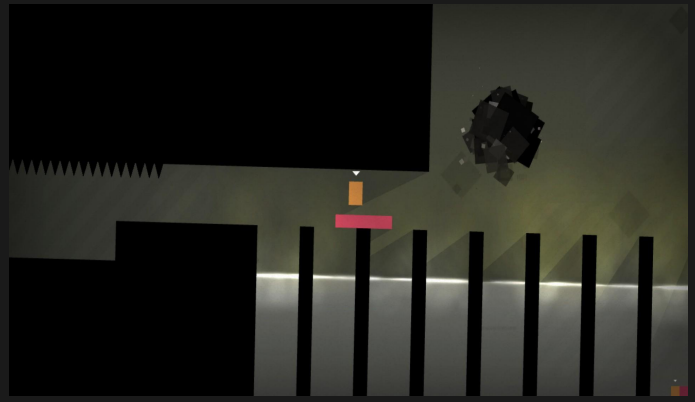
- **What does the handover process look like?**
  - Ensure all documentation, assets, and code are properly organized for future reference.
  - If handing off to another team, provide clear instructions and necessary training.
- **Is ongoing maintenance/support needed?**
  - Determine if the project will require updates, bug fixes, or additional content post-launch. If so, establish a plan for continued support.
- **How will success be measured after completion?**
  - Define key metrics such as player engagement, user feedback, or performance benchmarks to assess the project's impact and effectiveness.

Project Development: Geo Quest

Geo Quest Idea

Geo Quest Concept Summary

- **Visual Style:** Minimalist yet clean 2D platformer, inspired by *Thomas Was Alone*.
- **Gameplay Focus:** Precise platforming challenges, similar to *Fortnite* parkour levels.
- **Platform & Controls:** Designed for PC, using keyboard-only controls.
- **Level Structure:** A series of challenge levels, progressing sequentially upon completion.
- **Narrative:** No story, purely skill-based gameplay focused on player mastery.



Geo Quest Mechanics

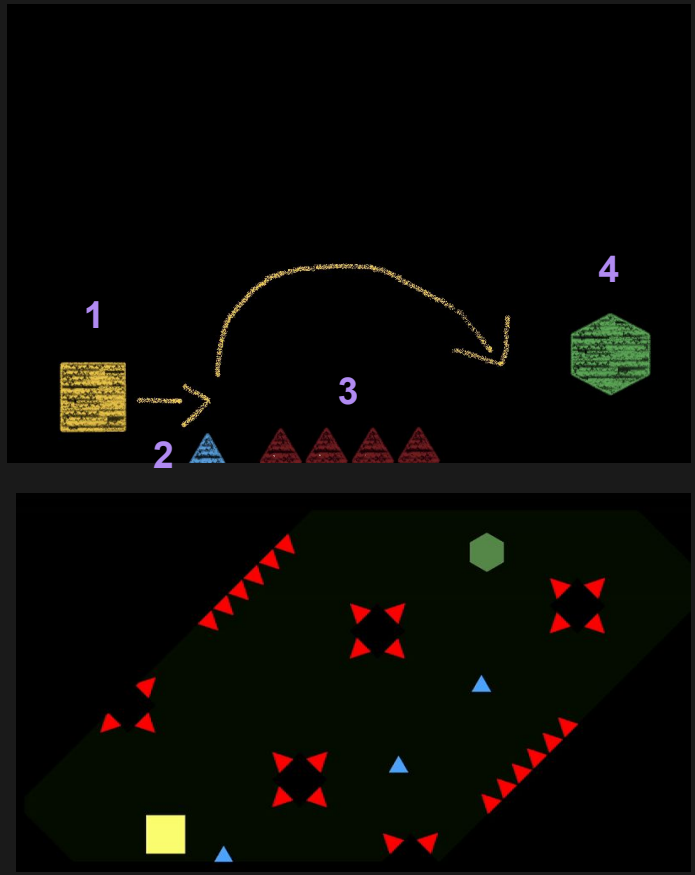
Geo Quest Mechanics Summary

- **Player Movement:** Move left and right using A/D or Left/Right Arrow keys.
- **Bounce Pads:** Launch the player into the air to avoid spiky obstacles.
- **Spiky Obstacles:** Upon touching these the player will reload into the same level they're currently in.
- **Level Progression:** Touching a hexagon at the end of a level loads the next stage.
- **Wall Interaction:** High friction allows the player to stick to walls for maneuvering.

Thumbprint

A **thumbprint** is a rough sketch of the game in action, helping to visualize core mechanics and level layout. Key elements to label and describe:

1. **Player:** The main controllable character that moves left and right and interacts with platforms.
2. **Bounce Pad:** Propels the player upward, helping them navigate obstacles.
3. **Spiky Obstacle:** Hazard that players must avoid to reach the goal.
4. **Goal:** Marks the end of the level; touching it advances to the next stage.

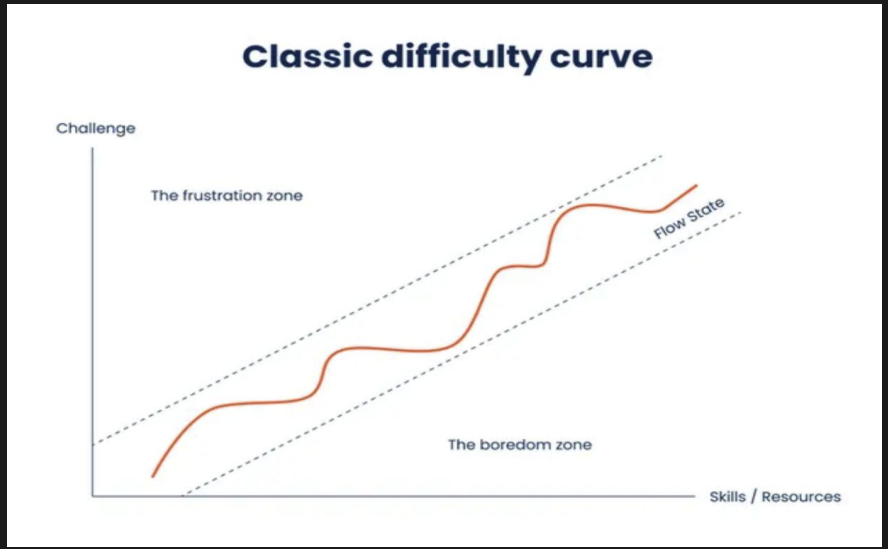


Player Engagement

Games keep players engaged by constantly introducing new challenges, mechanics, or story elements. If a game stops teaching the player something new, it risks becoming repetitive or boring.

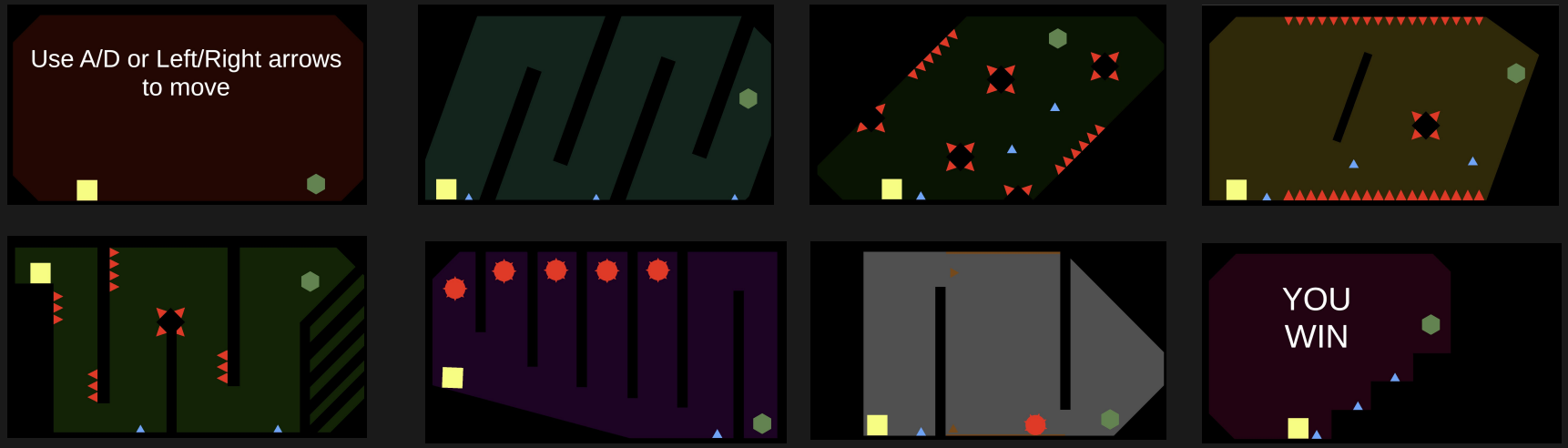
- **Flow State:** A balance between difficulty and engagement, keeping players challenged but not frustrated.
- **Progression:** Introducing new mechanics, enemies, or obstacles to maintain interest.

Striking the right balance ensures that players stay motivated and immersed in the experience.



Source

Geo Quest



In **Geo Quest**, each of the eight levels introduces a new challenge or tests the player's mastery of existing mechanics.

Each level ensures that players progressively build their skills while maintaining engagement and challenge.

Geo Quest - Level 1 & 2

Level 1: Basic Movement

This is a simple level with no obstacles, designed to introduce the player to the core movement mechanics. The player learns how to move left and right using **A/D or Left/Right Arrow keys**.

Level 2: Bounce Pads & Obstacles

This level introduces the first challenge: walls blocking the direct path to the goal.

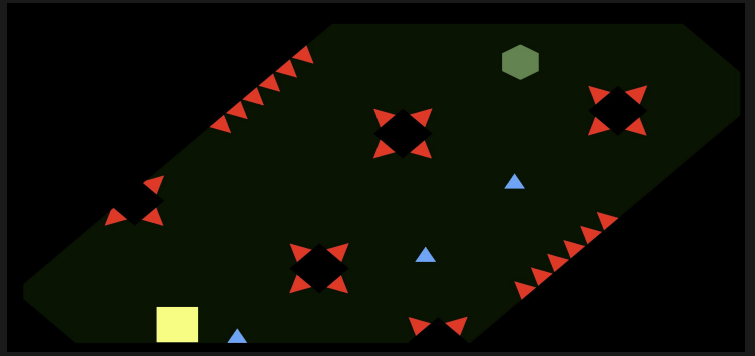
To overcome it, the player must use a **bounce pad**, which launches them into the air. This teaches the player how to interact with bounce pads and adjust their movement mid-air.

Geo Quest - Level 3 & 4

Level 3: Spikes & Precision Jumping

This level introduces **spikes** as a hazard, requiring the player to navigate carefully.

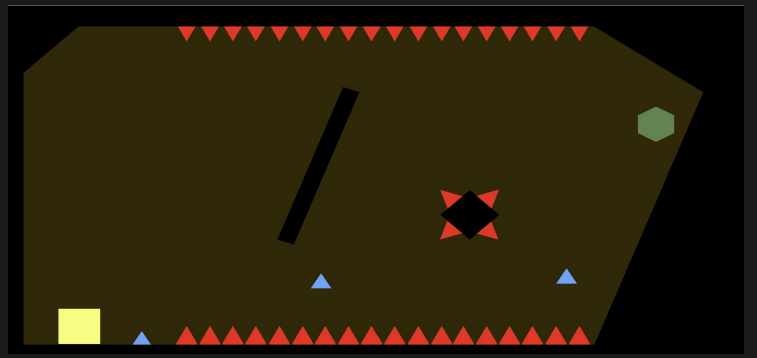
The challenge tests their ability to control jumps with precision, reinforcing what they learned in previous levels while adding a risk factor for mistakes.



Level 4: Sliding & Wall Cling

This level teaches the player about **wall clinging and sliding**.

A diagonal wall is placed to visually indicate that the player should slide down and launch themselves toward the next bounce pad. This reinforces movement mechanics and introduces a new way to navigate obstacles.

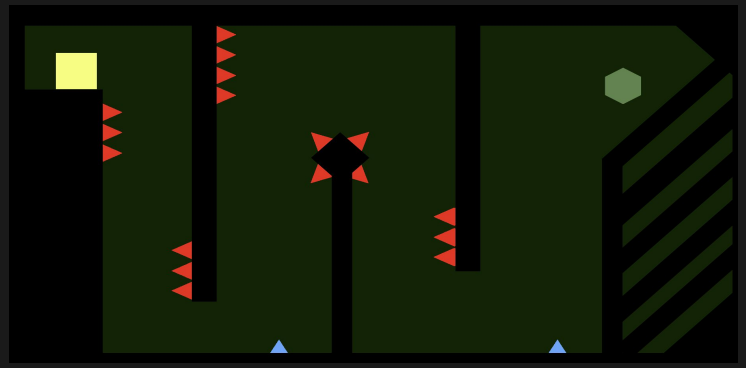


Geo Quest - Level 5 & 6

Level 5: Skill Test

This level combines **wall sliding and bounce pads** in a more complex way, requiring the player to use both mechanics together.

While no new elements are introduced, the challenge increases as the player must navigate obstacles using their previously learned skills.



Level 6: Physics Enemy Ball

This level introduces a new mechanic, a falling enemy ball that the player must avoid.

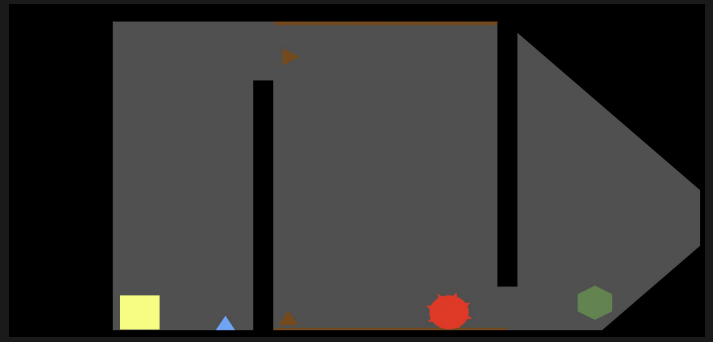
This adds an element of **timing and reaction**, requiring the player to anticipate the ball's movement while navigating the level.

Geo Quest - Level 7 & 8

Level 7: Ball-Specific Physics

This level introduces **special surfaces and bounce pads** that affect only the enemy ball, not the player.

This mechanic allows for **controlled movement of the ball,** creating obstacles the player must anticipate and avoid while progressing through the level.



Level 8: The Finale

The final level offers a **satisfying climb** to the goal, free from complex obstacles.

This serves as a **reward for the player's progress**, reinforcing their mastery of the mechanics and providing a **sense of accomplishment** as they complete the game.

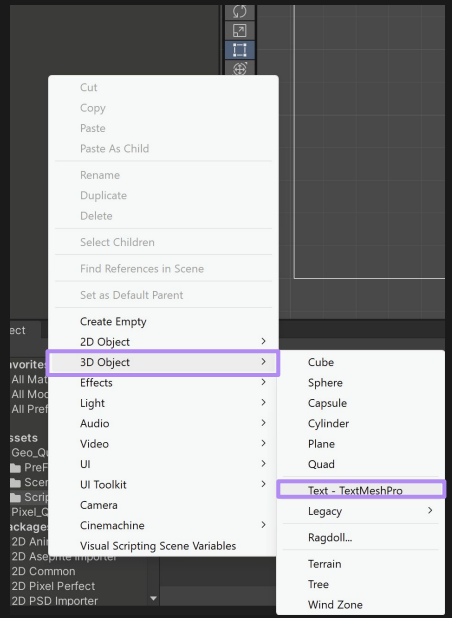
Geo Quest: Unity Tricks

Text - TextMeshPro

In Unity, you can create **text** using **TextMeshPro**, a powerful and flexible text-rendering system.

To create it go to **GameObject > 3D Object > Text - TextMeshPro** to generate a text object in the world.

The **TextMeshPro - Text** component lets you **edit the text, font, size, color, and style** directly in the Inspector. Text works similarly to text-editing programs like **Google Docs or Microsoft Word**, allowing bold, italic, alignment, and spacing adjustments.



Layers

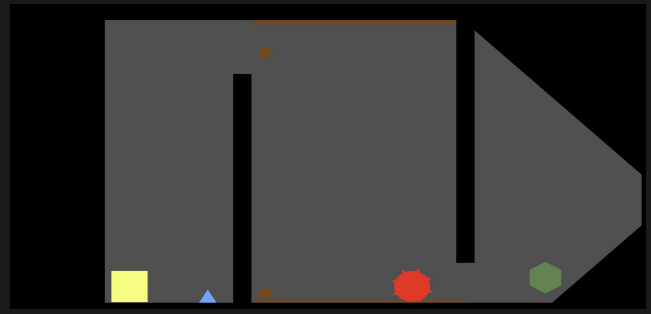
In this demo, we introduce an interaction system where the **brown object affects only the ball and not the player**.

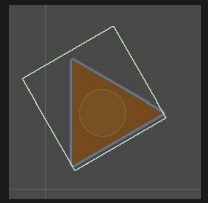
Unity allows you to define which objects interact with each other using **layers**.

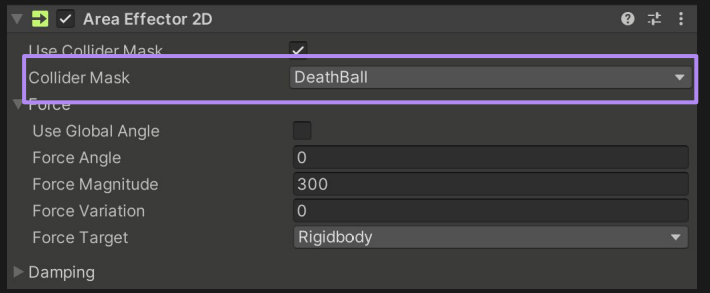
Every **Effector** has a **Collider Mask** input where you can **select specific layers** it will interact with.

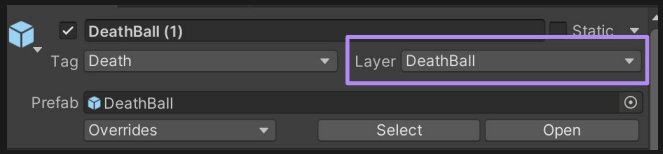
A new layer called **"DeathBall"** was created, ensuring that only objects assigned to this layer are affected.

The **DeathBall** GameObject was assigned to the **"DeathBall"** layer, while the player was left unaffected.





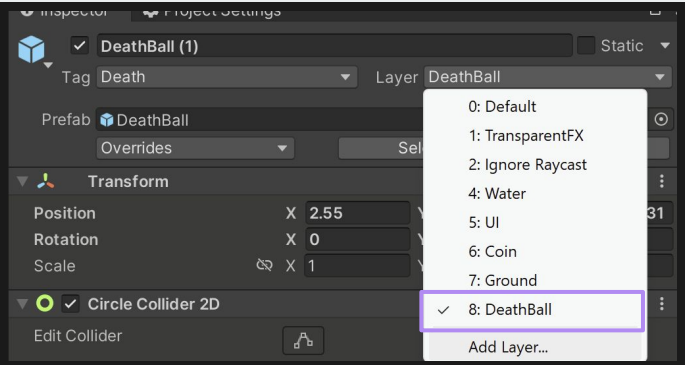


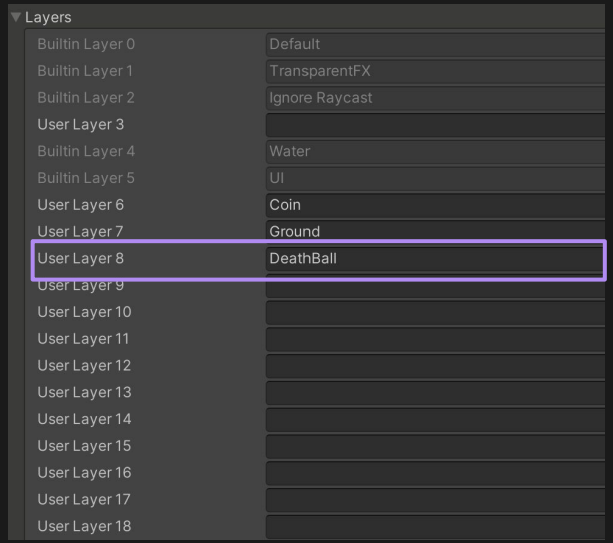


Creating Layers

To **create a new layer** in Unity, follow these steps:

1. Click on the **Layers** dropdown in the **Inspector**.
2. **Select "Add Layer..."**, This will open the **Tags & Layers** menu.
3. Unity provides a predefined number of layers. The **grayed-out layers** are reserved by Unity and cannot be modified.
4. **Enter a Name** in an available slot, type the **name of your new layer** (e.g., "DeathBall").
5. back to your **GameObject**, open the **Layer dropdown**, and select the **newly created layer** from the list.

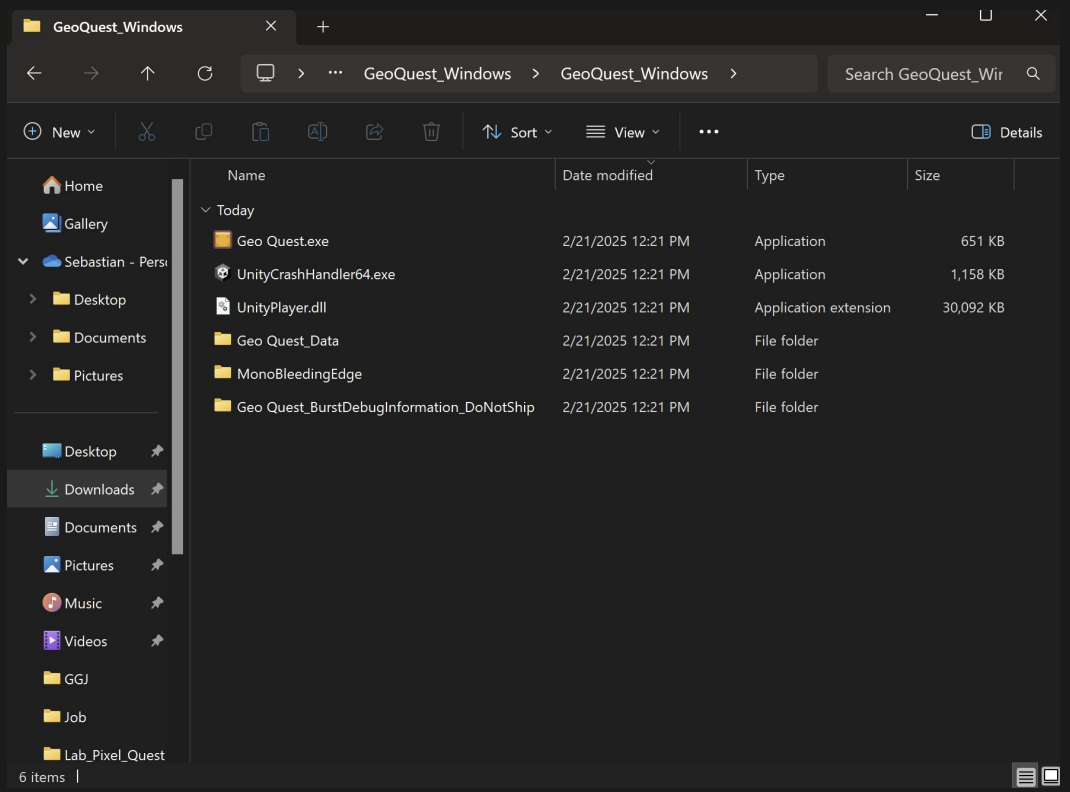




Geo Quest: Making a Build

Creating .exe

Once you've finished designing our game, you need to create an **executable file** so others can play it.



Project Settings

Before building your game, it's important to **check the Project Settings** to ensure everything looks correct in the final executable.

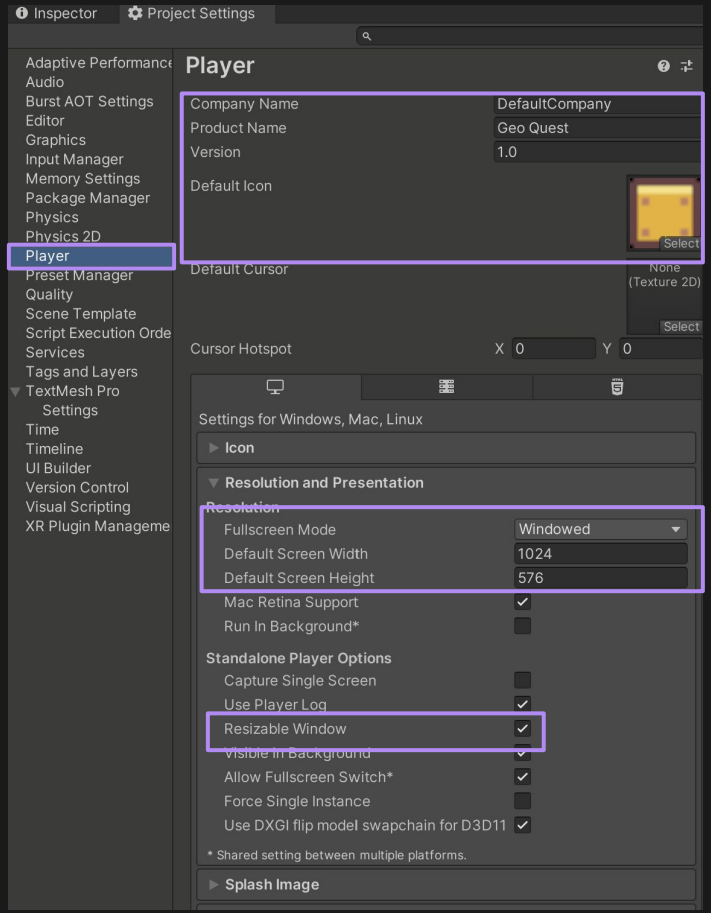
Go to Edit > Project Settings > Player, This opens the **Player Settings** panel.

Set the **Company Name, Game Name, Version, and Icons** that will appear in the game folder.

Since we're making a **Windows** game, focus on this section.

Adjust Window Settings:

1. **Fullscreen Mode:** Set to **Windowed** (prevents forced fullscreen).
2. **Resolution:** Set **Width: 1024, Height: 576** (a good window size without taking the full screen).
3. **Enable "Resize Window"**, Allows players to adjust the window size.



Build Settings

Once you've configured your **Project Settings**, it's time to set up your **Build Settings** to prepare your game for export.

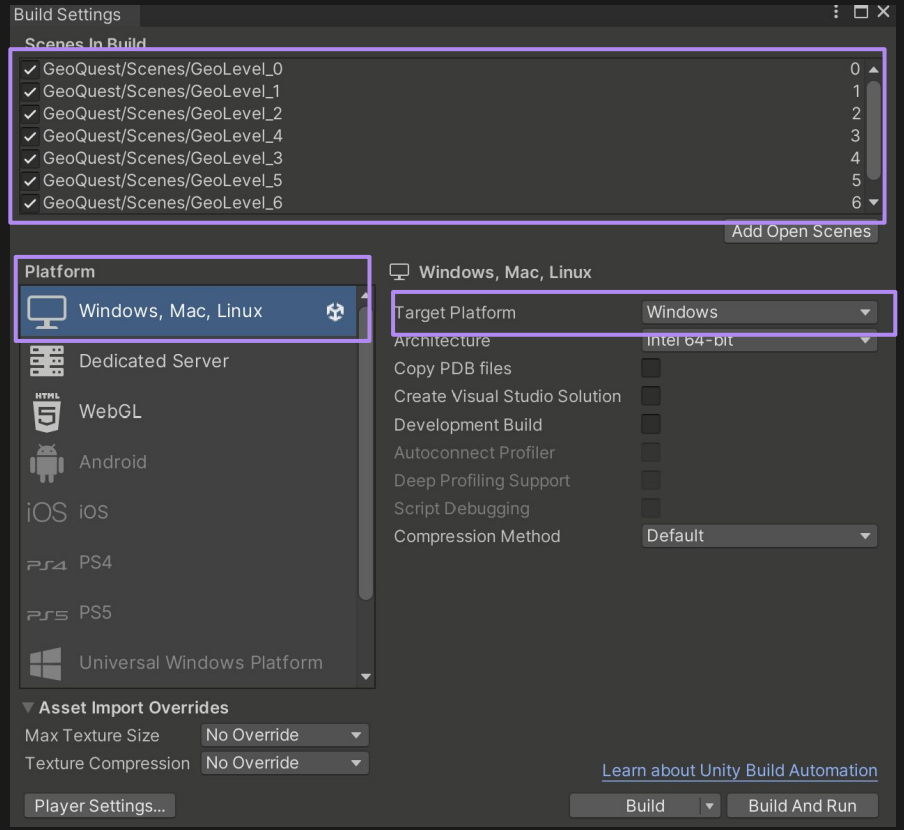
Go to **File > Build Settings** to access the build menu.

Add Scenes to the Build:

- In the **Scenes In Build** section, make sure **all the scenes** you want in the final game are listed.
- Drag and arrange the **starting scene** to the **top** of the list to ensure the game begins in the correct place.

Select the Platform:

- Choose **Windows, Mac, and Linux Standalone** (since Unity groups them together).
- Under **Target Platform**, make sure **Windows** is selected.



Build

Once all settings are configured, follow these steps to complete your build:

Click "Build"

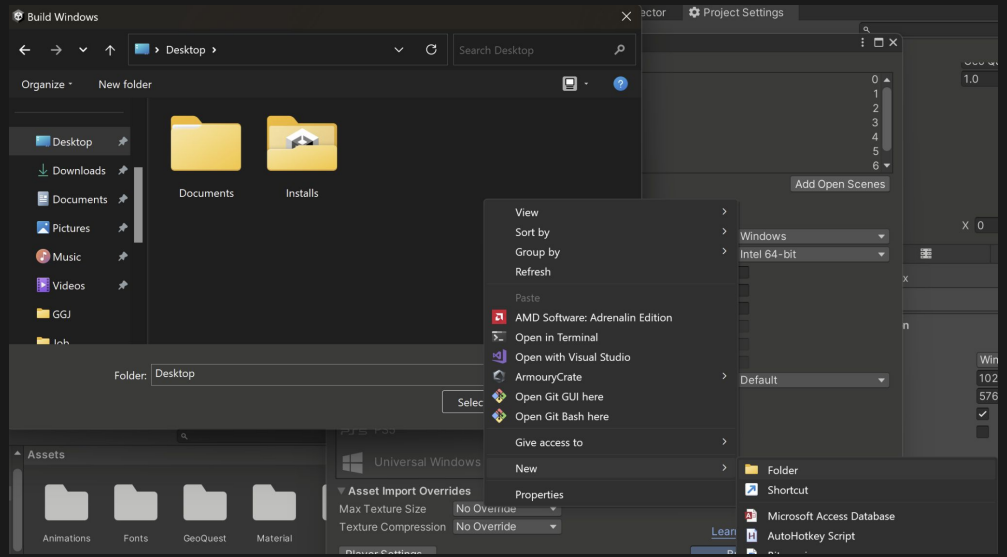
- Open **Build Settings** and click the **Build** button.
- A **File Explorer** window will appear, prompting you to choose a save location.

Create a New Folder

- Navigate to your **Desktop**.
- Create a **new folder** and name it after your game (e.g., **Geo Quest**).
- Select this folder and confirm.

Wait for Unity to Build the Game

- The process may take a few minutes.
- Once finished, the folder will contain **all necessary game files**, including the **.exe file** (for Windows).



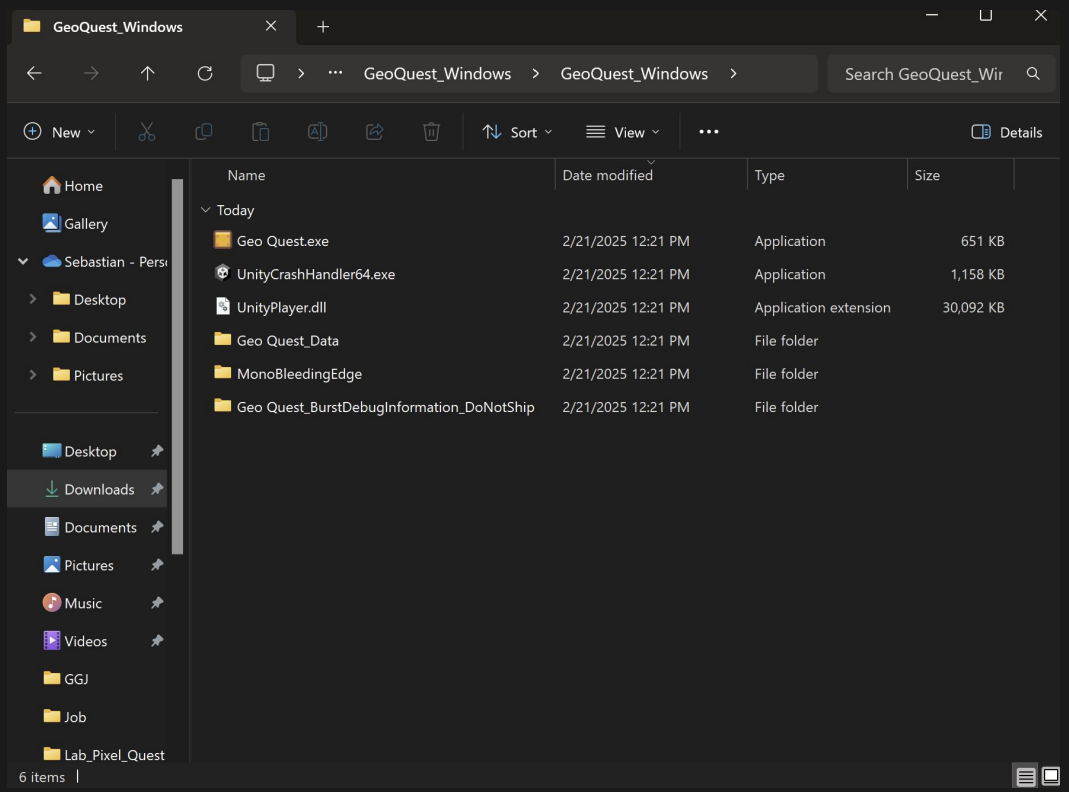
Executable

Once everything is built, you will see the name of your game with a .exe extension.

Double-click on it to open the game and test it to ensure everything runs smoothly.

Play through your game to check for any bugs or issues, and let others try it to gather feedback.

If you want to share it, consider uploading the game folder to a platform like Itch.io or Google Drive for easy access.

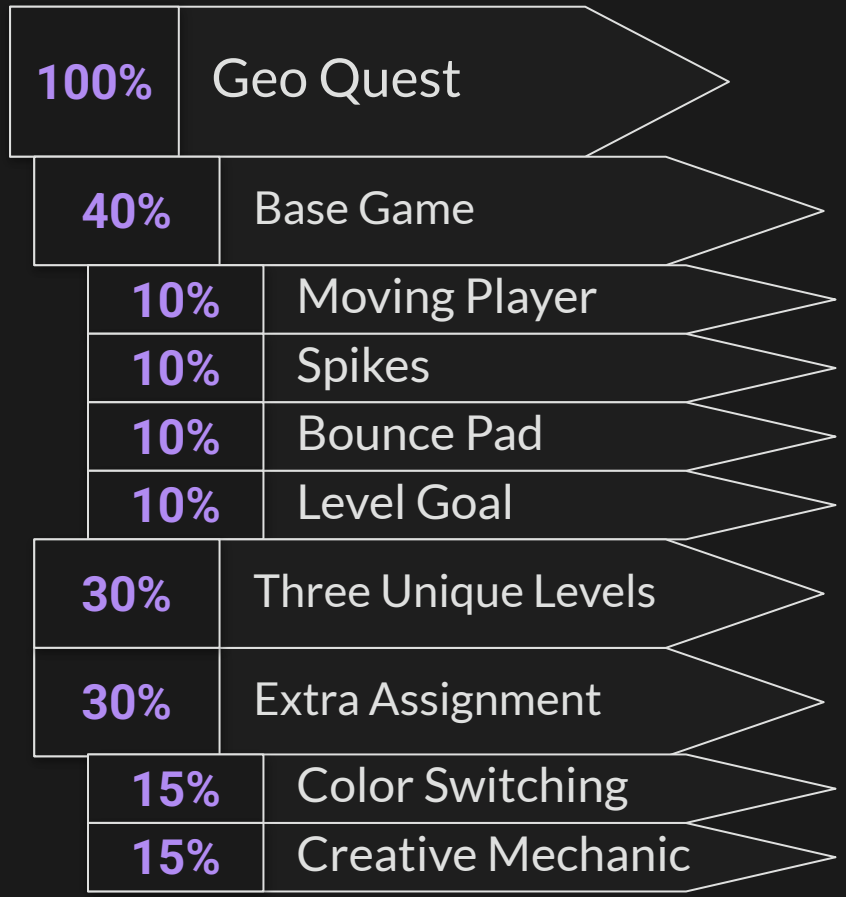


Grading & Submission

Grade

Geo Quest is worth **7.5%** of your overall grade, with the grading breakdown as follows:

- **40%** for implementing core mechanics covered in class, including player movement, dying to spikes, bouncing off bounce pads, and transitioning between levels upon touching the goal.
- **30%** for designing a minimum of three unique levels that challenge the player's skills.
- **30%** for adding new mechanics:
  - **15%** for implementing a mechanic that allows the player to switch colors when pressing 1, 2, or 3.
  - **15%** for a creative addition of your choice to enhance the gameplay.



Submission

Submission Guidelines

You will create an **.exe file** of your game, then **zip compress** the folder and upload it to the provided link.

https://forms.gle/FJqahKQxz49LyK6t5

How to Zip Compress Your Game:

1. **Right-click** on the folder containing your built game.
2. Select **"Compress to Zip file"** or **"Send to > Compressed (zipped) folder."**
3. Ensure the compressed folder has a **zipper icon**, indicating it has been properly compressed.

Alternative Submission:

If you are unable to upload the file, email it to **sgrygorczuk@gmail.com** and **include your name** in the email.