2024-2025

Achievement Guide

18th Annual Oregon Game Project Challenge

www.ogpc.info

OREGON GAME PROJECT CHALLENGE

Overview

Achievements are broken down into multiple levels. To qualify for a higher level, a team must fully complete all requirements from the lower levels. Even if it would be possible to complete some part of a higher level without completing everything from the lower level, a team will only qualify for the lowest fully completed level. Teams will only qualify for completed levels – i.e. all requirements for the level have been met. For example, a team that has not fully completed the standard level cannot qualify for either encouraged nor extra credit levels.

The achievements are broken into the following levels:

- **Standard**: New teams should try to target the Standard level of every achievement. These are achievements we feel should be attainable by students of any skill level.
- **Encouraged**: The Encouraged level is slightly more advanced than the Standard level. These are achievements that most teams should work towards, even if you can't quite get all of them.
- **Extra Credit**: The Extra Credit level is the most advanced. Teams targeting this level will have to go above and beyond to consistently reach the Extra Credit level.

Achievements labeled with a book symbol (📖) have a documentation requirement. The relevant criteria will also have this symbol and indicate that they contain documents that need to be present within the Game Portfolio document linked on TMS and presented on competition day.

Achievements At a Glance

PROGRAMMING

Maintenance of Code

Code should be easy to follow, saved in a central location with backups, and have development over time. No spaghetti code!

Standard

- Code is clean: labeled properly and named descriptively
- Save multiple copies of the project, or project is saved to the cloud/online

Encouraged

- Code is navigable and logically separated into functional sections
- Git/version control history shows at least a 5 commits/patches over time

Extra Credit

- General best practices[1] were followed and the code would allow for future development
- Git/version control history shows 10 or more commits, ideally smaller or even daily commits

Explanation

This achievement focuses on your use of source control, or a method of backing up code, and cleanliness of code. These are both fundamental to maintaining a good codebase and allow others to easily step in and orient themselves with new code.

Source control is a critical part of keeping track of your work, how it looks over time, and knowing who made each change. Almost every professional development group uses it to keep track of their code. Solutions like Dropbox, Google Drive, and OneDrive are an acceptable version control starting point and ensure that your game will carry on if your laptop were to break! For advanced teams, Git and Mercurial are professional grade source control solutions that are still approachable for students and have many free services to host your code. There is a lot of documentation on these options so feel free to explore and give them a try!

[1] Code should be easy to read with simple, descriptive names and easy to follow logic. This also includes comments, tabbing, semicolons, and functions that are as simple as possible (avoid putting all logic into single functions that do everything!). Visual coding languages (such as Scratch) should be grouped into logical sections and be labeled.

CS Fundamentals 📖

How did you leverage Computer Science fundamentals such as conditionals, loops, and data structures to implement a mechanic?

Standard

- 📖 Document at least 2 samples of the game's loops and conditionals
- 📖 Document the game's usage of least 1 standard data structure (such as a list, map, dictionary, stack, or queue)
- Team is prepared to describe how the game uses loops, conditionals, and data structures and how they work

Encouraged

- 📖 Document the game's usage of least 2 different standard data structures (such as a list, map, dictionary, stack, or queue)
- 📖 Document high level descriptions of all major mechanics' implementations, preferably as pseudocode or a block diagram
- Team is prepared to discuss their implementations of the game's mechanics

Extra Credit

- 📖 Document how and where your game implements and uses an algorithm
- Player state is stored in a non-global data structure
- Codebase uses classes or similar data/functionality grouping structures where appropriate

Explanation

Loops and conditionals are a basic part of keeping a project's code a reasonable length and easy to read. As a codebase grows, the use of consistent tools makes it easier to find bugs quickly without needing to treat each section differently. In addition to consistent code fundamentals, diagraming can also make it easier to understand how the code is intended to work and find issues in advance.

Documenting how your game is implemented is key for a growing codebase. This documentation can be used to help bring another person onto your team without needing to verbally describe everything, and keeps the existing team members on the same page for how each section is intended to work.

For higher tiers, make sure to use loops, conditionals, and data structures frequently and consistently. Diagram more difficult sections of code and avoid global variables!

Bug Smashing 📖

Show how problems were fixed, avoided, and purged!

Standard

- 📖 Document an example of a logic bug (not typo or compiler error) that your team encountered and describe how it was fixed or worked around
- Game demonstration does not crash, freeze, or otherwise completely break

Encouraged

- Talk about a proactive strategy you used to avoid bugs
- Gameplay demonstration is free of major/obvious bugs
- 📖 Document your testing process to search for bugs

Extra Credit

- Show your team's system to keep track of bugs you've noticed
- Game executed flawlessly during demonstration
- 📖 Document how your team leveraged a standard testing framework[2]

Explanation

A bug describes a mistake in the logic of your code (more than a typo). If you forget a semicolon and your code stops compiling, this is known as a "syntax error". A logic bug is specifically when your code runs, but does not behave as you intended. These bugs often require the programmer to make changes to their code: adding an if statement to handle a specific case, verifying your math is correct, or correcting overstepping the end of an array.

A well-built game should perform well, even when being watched! Glitching through walls, breaking the physics engine, failing to pick up an item on keypress, or causing the game to stop responding or even crash are all to be avoided!

Comprehensive testing strategies help prevent bugs from reaching the player and ruining the game's experience. Testing helps to locate bugs early in development and ensure everything is working as intended. For higher tiers, we're looking for clearly defined systems for finding, tracking, and smashing bugs. These systems will (hopefully) help your team produce a cleaner product that will run smoother when demoing for the judges.

[2] Standard testing frameworks are tools that are commonly used to write and execute unit tests for software projects. These include tools such as Jasmine (many languages), Crispy (GameMaker), Pytest (Python), or Jest (JavaScript) depending on what language you use. Some engines provide their own frameworks, such as Unity Test Framework.

Dynamic Aspects

Games make use of time and other modular elements to enhance gameplay.

Standard

- Game features more than one version of at least one asset to represent different states and dynamically swaps those assets at least once (for example, player can pick up and wear a yellow shirt)
- Animate something with a state machine (for example, swap 2 walking sprites)

Encouraged

- Game features a time related system which meaningfully affects gameplay (for example, day/night, weather cycles, etc.)
- Game swaps/combines/modifies assets or modifies mechanics based on time system (for example, day/night cycles, seasons, in one level; music speeding up as the countdown nears zero; etc.)

Extra Credit

- Choose at least one of these three:
  - Game features a full-featured time system integrated into major mechanics of the game
  - Game generates content procedurally
  - Game has dynamic or procedural level generation

Explanation

Introducing an in-game day/night cycle, weather pattern, or other similar time-based event system is one way to vary gameplay, introduce some interesting programming challenges, and encourage advanced planning. Many popular games make use of weather, time of day, sandstorms, snow, fog, and other such atmospheric cycles. Think of games such as Stardew Valley with its weather and day/night cycles, Pokémon games such as Sword and Shield with berries respawning after 24 hours, or Pokémon Go with the various Eevee evolutions. While it would be tempting to simply swap assets for darker ones on a given map, we're looking for a real-time change (since this is a programming achievement!). Think about how a level or map could change, what events could be introduced, or how you can change assets at runtime through code.

For the highest tier, there are a few options to explore procedural content or level generation. Minecraft is a great example of procedural content generation in many ways. For level generation, a unique game world is generated for each playthrough. For procedural content: clouds, plants, music, and even textures can be generated to be unique for each save. Alternatively, implement a time system that affects multiple major mechanics of your game.

Object Permanence

Menus allow players to interact with additional game content and access saved data.

Standard

- Game starts up with a menu that allows players to:
  - Start a new game
  - View the game's credits.

Encouraged

- Game has a start menu with additional options to:
  - Use saved data (view a leaderboard, resume a saved game, or see other saved statistics)
  - Change options/settings (at least 2 functional, such as volume, difficulty, or keybindings)

Extra Credit

- Game has at least 1 persistent record such as a save system, leaderboard, settings, or other game related data that can be accessed between runs
- Game has a pause menu with resume, and at least 4 functional game settings (volume, difficulty, keybindings, etc)

Explanation

Menus are often the first way a player will experience your game. While easy to take for granted, it is important to ensure the menus operate smoothly and behave as a first-time player would expect.

For additional options/settings, controls like volume, brightness, difficulty, graphics, or key bindings are all good starting points. A pause menu with the option to resume the game, change volume, and quit the game would suffice but we encourage extra settings so players have the opportunity to make sure the game is set up how they want to play. The credits should include information about both your team and your game, and give you an opportunity to give thanks for others who may have helped during development.

For higher tiers, save something locally to your machine! While not every game will have a saving system, it should be possible to at least save your user's settings so they do not have to configure them for every run.

Game Design

Design Doc 📖

Write a design document. What are your milestones? What is being communicated to the player? What is the general direction and mood of the game?

Standard

- 📖 Team created a game design doc that describes the game at a high level
- 📖 Design doc lists milestones (that can hopefully be completed by the main event!)
- 📖 Design doc contains basic info such as win condition, genre, and target audience

Encouraged

- Design doc has been updated at least once to show changes in the development process
- 📖 Design doc has multiple milestones set
- 📖 Design doc communicates key game mechanics, the main character, and the setting

Extra Credit

- 📖 Design doc has been maintained through development with multiple updates and a changelog
- 📖 Design doc has UI mockups
- 📖 Design doc has all core game mechanics, all important characters, and info on all levels or scenes

Explanation

Design documents are an important tool in software development to help keep the team focused on a clear and unified vision. If one of your team members doesn't understand what the team is trying to make, it could lead to arguments or other slow-downs as you try to get everyone aligned. Additionally, design documents help drive everyone towards milestones which allows your team to demonstrate mechanics even before your game is completed, which can be critical to receive feedback early in the development cycle!

As you work on your game, you'll probably find that things will change: requirements will be replaced or removed, your story will develop, and you'll probably miss some deadlines. For higher tiers of this achievement, show how your document changed, and covers most (if not all) of your game. The design document needs to be maintained, and complete! It should be included in your team's Game Portfolio.

Mechanics 📖

Game tasks the player with completing various goals however the specific processes and environments in which the player accomplishes those goals can vary greatly.

Standard

- Player is given more than one mechanic for interacting with the game (for example, teleportation, rewinding time, picking up items, jumping, sprinting, etc.)
- Game has at least one, well-crafted and well-designed environment (for example, level, biome, map, etc.)

Encouraged

- Game has notably different mechanics implemented into the game (jump and double jump are not notably different)
- 📖 Game design doc describes multiple settings that could showcase mechanics. Examples:
    - Ice level that changes how the player moves
    - Dark dungeon that requires torches or light sources
    - Space station with low gravity

Extra Credit

- Game has at least one novel or unique mechanic (other than move, jump, pick up item, sprint, etc.)
- Player can have a unique experience between playthroughs

Explanation

The main objective for this achievement is that your game has more than one setting for players to explore implemented into the game, and more than one way to explore each setting. This variation helps keep your players engaged through the whole experience of your game. Even the most interesting story can feel boring if only a single mechanic is used for hours on end.

Your varied settings could be independent levels, separate areas in one level, or the same area in a level but different textures and/or models. Mario Odyssey demonstrates this concept well by implementing desert levels (quicksand), ice levels (sliding with movement and needing cold weather clothes), and water levels (geysers). There are great examples of the environment impacting the mechanics. Various mechanics could be gliding, persuading, crafting, eating, staying on rhythm, or other ways your player can interact with the world!

For higher tiers, focus on finding a unique mechanic that your game can highlight. Finding a fun way for a player to interact with your game is key to holding interest!

Target Audience 📖

Who is the target audience/demographic? What concessions had to be made for them? Does the game adapt to the player?

Standard

- 📖 Document at least one target audience (kids like me, adults, my friends, etc.)
- 📖 Document at least one feature or design consideration to better fit the target audience

Encouraged

- 📖 Document at least one specific and well-defined target audience (for example, middle-aged grand strategy gamers, teenage casual gamers, elementary age Roblox fanatics, etc.)
- Team held design reviews to ensure game continued to be fun/engaging for the target audience
- Game has been playtested by the target audience(s)

Extra Credit

- 📖 Document playtesting feedback by members of the target audience(s) who are not team members
- Game mechanics, controls, color scheme, etc. all fit the needs of the target audience(s)
- Game has settings (difficulty, color blind mode, etc.) intended to better adapt to different audiences

Explanation

Every single game has a target audience: the people by which the game was designed to be played. A few examples would be “elementary school students learning to type,” “commuters with long public transit rides,” or “11 to 18-year-olds who socialize through games." To make your game appeal to the audience you have chosen, you will have to make some decisions about what is appropriate for those players. For example, a commuter likely wants games that can be quickly picked up and put down without long play sessions or loading screens. In contrast, people trying to socialize in-game may desire a more immersive experience. Keeping your target audience in mind, helps avoid adding features or requirements to your game that your players dislike. While there is no right or wrong target audience choice, you need to make a convincing argument for your audience and related choices.

For higher tiers, make sure as much of your game as possible is built around your target audience. Simply saying you have a target audience doesn’t mean much if there are no design considerations that go into the game's implementation.

Evolution of Design 📖

How was the game initially conceptualized? What changed when real people playtested? What was axed due to time or other constraints?

Standard

- 📖 Document meeting notes indicating changes in mechanics, assets, or other aspects of gameplay
- Game has been playtested (play from the beginning to the end)

Encouraged

- 📖 Document images showing early game builds with features, assets, characters, etc., that were dropped due to scope or time issues (take screenshots every now and then!)
- 📖 Team conducts a playtesting session and documents feedback and observations

Extra Credit

- At least one playtesting session was recorded (Screen capture is perfect)
- 📖 Game has been playtested by non-teammates on at least two separate occasions with team documenting feedback after each session

Explanation

Playtesting is an important part of any game creation process. It allows your team to verify that the game you designed is fun, engaging, and communicates what you were trying to communicate. Additionally, it helps identify where players may be confused or lost, even if it was clear to you as the creator.

A good playtesting session should show you areas that need improvement, things you've done well, and any additions you may need and haven't considered. It's a great way to get some feedback on your game before you have a panel of judges looking over your work and giving you a score!

For higher tiers, make sure to show how your game evolved. Playtesting sessions often provide useful feedback to shape your game and recording a session can highlight where the player had issues. Playtesting is also more useful if the person doing the playtesting is not on your team!

Performance Review

How does the game communicate with the player? How is the player taught new mechanics?

Standard

- Game has at least 1 indicator for the player's state (health bar, experience, resources, score, etc.)
- Game provides periodic feedback to the player (getting points, leveling up, etc.)

Encouraged

- Game has at least 2 indicators for the player's state
- Game provides instruction on how to use new mechanics as they're introduced

Extra Credit

- Game provides specific and continuous feedback to the player in more than 2 ways beyond numbers or bars.
    - Examples include, but are not limited to: camera shake, sound effects, environmental changes, NPC reactions, etc.
- Game includes audio or visual feedback cues that indicate a player's performance when interacting with the game's mechanics[3]

Explanation

It is important for your game to effectively communicate information to your players so they understand how well they're doing. Health, money, and carry weight are all important pieces of information to the player but aren't usually visible in the game world. While numbers or displays may be sufficient to show these values to a user, more complex or subtle cues can provide a more immersive experience.

Similarly, your game needs to communicate effectively to players about what they can do and how well they're doing it. Your game should teach players about mechanics and how to use them as they're introduced. This usually means that you start the player off with a few simple features/mechanics, allow the player to get used to them, and then add more until the player has full control of every mechanic.

[3] For higher tiers, feedback provided to your player should be delivered in a manner that fits the theme of your game. For example, a common technique in farming games is to change the color of the soil when a plant has been watered recently. In addition, the feedback should relate to how well the player is using the mechanics and provides indication or reward for excellent execution. For example, a perfectly timed double jump could create a special particle effect, maybe the player goes slightly farther, or even makes a "hya!" sound!

Art and Assets

Artistic Expression 📖

Make your game visually unique and give it your own flair!

Standard

- Create 5 or more custom visual assets
- 📖 Create a game logo and icon (not part of the 5 or more count)
- 📖 Cite all sources for assets not created by the team (those from creative commons, libraries, your game engine, AI tools)

Encouraged

- Create more than 10 custom visual assets
- Create title art or a splash screen (not part of the more than 10 count)
- 📖 Document the tools and processes used for creating visual assets

Extra Credit

- 📖 Have a reviewer, ideally someone outside your team, assist with an Art Critique[4] and document their feedback as well as any changes your team made to address feedback

Explanation

Graphical assets include everything from character sprites to plant models, particle effects to skybox/backgrounds, and even UI elements such as buttons. These assets do not necessarily need to be implemented into the game to count, but they should be intended to be included. Only assets created by the team will count for this achievement. Any public domain, creative commons, or outside the team assets used must be cited! Editing an asset made by someone outside your team does not count toward this achievement. Remember, you cannot pay for assets (nor commission).

In addition to the assets in the game, creating a logo and title screen art are great ways to get people excited for your game. Though often simple images, the creative process to create a logo and icon are often challenging and require multiple passes before everyone is happy!

[4] An Art Critique is a collaborative process which reviews artistic works to explore how they can be improved. It is critical that this process is done with care, as one's artwork is often a deeply personal expression. This process must be approached with the intention of focusing on how the artwork can be tailored to fit the overall project's theme and to ensure a consistent stylistic approach.

Hey DJ 📖

Create a unique auditory experience to provide depth and immersion for your game.

Standard

- Record at least 5 voice lines or sound effects made by the team (footsteps, light switch, running water, crunching leaves, etc.)
- 📖 Cite all sources for assets not created by the team (creative commons libraries or your game engine)

Encouraged

- Create one original music track (at least 30 seconds) to be included into the game as background music, title music, etc
- 📖 Document the tools and processes used for creating sounds effects and music

Extra Credit

- Create 3 or more music tracks (~30 seconds each)
- 📖 Have a reviewer, ideally someone outside your team, provide feedback for at least one of your music tracks and document their feedback, as well as what changes (if any) your team made to address feedback. This feedback should be focused on how the track could be tailored to fit your specific game.

Explanation

The music and sounds of your game help draw a player in and create a layer of immersion that isn’t possible via any other method. Hearing your character walk across leaves, that satisfying thunk after pulling the correct lever, dings and bells for finding important objects, or playing emotional music all help enhance the scene and pull your players in.

Only assets created by the team will count for this achievement, however, any public domain or creative commons assets used must be cited!

For higher tiers, create multiple music tracks that are intended for use within the game. These can be title screen tracks, background music for a level, victory anthems, or any other musical applications that enhance your game.

Cohesive Look and Feel

Everything fits together nicely, and nothing stands out.

Standard

- Have at least one color palette (consistent set of colors used everywhere)
- Have at least one soundscape (consistent set of sounds used everywhere)

Encouraged

- Assets are all of a consistent scale, resolution, and style; no assets are unintentionally jarring
- Sounds and music are a consistent volume, quality, and style; no sounds feel out of place or break player immersion
- Most player actions have sounds and animations associated with them

Extra Credit

- Audio and visual assets have depth that fit in with the game’s setting and world
- Elements in the game’s world have visual cues to indicate interactivity or facilitate immersion
- All player actions have sounds and animations

Explanation

No one wants to play a game where half the assets are high resolution and interesting to look at while the other half are blurry and hard to identify. Similarly, nobody enjoys playing a game if the ambient sounds are quiet, but flipping a switch is so loud that you throw your headphones. While there is certainly a time and place for jarring visual or auditory experiences, it must be intentional!

In addition, color matters too. One (or more) color palettes can improve your player's ability to identify items in game and help scenes to feel cohesive. Choose a type of color palette (Monochrome, Analogous, Complementary for example) and be ready to explain your choice. Displaying your palette along with your concept art is a good idea and can help build a cohesive brand.

For higher tiers, make sure player actions have sounds associated with them and make sure all assets fit into your world and soundscape.

High Fidelity

Game uses particle/atmospheric effects and animated graphics to enhance the game.

Standard

- Create one particle or atmospheric effect (rain, fog, smoke, sparks, etc.)
- Choose one:
  - Create at least one animation sprite sheet (4 or more frames)
  - Create at least one rigged and animated model

Encouraged

- Create 2+ particle or atmospheric effects
- Choose one:
  - Create 2 or more animation sprite sheets (4 or more frames each)
  - Create 2 or or more rigged and animated models

Extra Credit

- All particle or atmospheric effects used in the game are created by the team to enhance the look and feel of the game
- All animation sprite sheets or rigged and animated models are created by the team

Explanation

Have you ever sent something up in a satisfying puff of smoke, or had sparkles float around after you've cast a spell? These kinds of effects provide feedback for the player and make the game feel more immersive.

Animated graphics help make movements and actions feel more realistic. Animated graphics refers to objects and/or characters with multiple frames. Just moving a sprite is not sufficient. It could be a tree swaying back and forth in the background, a walking animation, a bird flapping across the sky, something growing or hopping, or twirling. They can be simple, as long as they have more than one frame.

It should be noted that any kind of gore effects or animations not only won’t count toward this achievement but could disqualify your entry! Remember to keep your games E10+ qualifying.

For higher tiers, make more effects and animations though some can still be added from open-source libraries. For the highest tier, all effects and animations must be created by the team.

Setting the Scene 📖

Players are given multiple, unique experiences in intentionally crafted environments

Standard

- 📖 Document the artistic direction for at least 1 Setting in your game.
  - This should include the intended color scheme, thematic elements (wintery, spooky, vibrant, etc), and how music and art can be used together for a consistent experience.

Encouraged

- 📖 Document the artistic direction for 2 or more Settings in your game.
- 📖 Document how each Setting has multiple unique artistic aspects that are distinct from other Settings
  - Examples include having different color schemes, themes, musical aspects

Extra Credit

- Audio and visual assets and design are consistently tied together for all settings
  - These assets match the unique style of the current Setting (for example, spooky room plays spooky music, springtime on the farm has upbeat music and lush plant life)

Explanation

The setting of a game is very important (especially when the theme could be interpreted as a setting). The player should be able to easily understand and feel immersed in whatever setting your game creates. Whether it takes place underground, in space, or on a boat, make sure the setting contributes to the player's experience. As these settings are defined, they'll also take on a larger role in describing the artistic approach of your game. If your setting is a school, for example, it would be expected to have the school colors be consistent in the scenes. An underwater setting can go beyond simply being blue tinted, and can instead explore the colors of various fish and plants!

Incorporating different environments into the game can be challenging depending on your genre, but this does present unique ways for your team to express themselves! Even card games can have unique table designs and card art! Exploring these directions can lead to interesting gameplay effects. Maybe you end up with an icy cave setting for your card game, which inspires your team to add gameplay mechanics around cards being periodically frozen if they aren't moved! This can be a great avenue for inspiration and exploration.

THEME AND STORY

Thematically Inclined

OGPC provides a theme, you make it your own!

Standard

- The season’s theme is present in the game.

Encouraged

- The season’s theme is clearly connected to the story or mechanics.

Extra Credit

- The season’s theme is consistently tied to multiple aspects of the game – music, mechanics, setting, visuals, etc.

Explanation

The theme ought to be evident in every aspect of your game from sound effects to core mechanics. Game objects, dialog, sprites/models, music, and level design all need to be consistent with one another and relate to the theme. A cohesive theme ties your game together and makes it more interesting and immersive.

This is not to say that your game can’t have aspects that vary from the provided theme, just be sure that everything is consistent within your game's universe. You can have lots of fun here, we encourage students to find creative interpretations of the theme, just be sure that everything is integrated and consistent. The game should be truly integrated with the theme across setting, story, music, visuals, and mechanics.

For higher tiers, tie the theme into most (or hopefully all) of your game. The theme should not be an interchangeable "coat of paint" on top of your game. Make sure it is well integrated into game design and mechanics.

Intentional Design📖

Plan out the theme, progression, player choices, and potential endings.

Standard

- 📖 Create a rough storyboard for at least 1 scene in the game

Encouraged

- 📖 Create a storyboard for the player's full path through the game (high level overview for the entire game)

Extra Credit

- 📖 Create detailed[5] storyboards for all important moments in the game: opening, climax, key turning points, etc. which include:
    - Major characters involved
    - Mechanics that the player is expected to interact with
    - Description of the scene's importance to the overall story

Explanation

A storyboard is a very important part of your design process and story development. A storyboard helps you plan how a scene looks, who moves where, and what parts of the story are progressed. Your storyboards should include descriptions that describe what is happening, and why the scene is important. A copy of the storyboards should be present in your Game Portfolio.

For higher tiers, include more about your story. The more important moments you can include the better! For the highest tier, make sure that all of your important moments have detailed storyboards. Planning out the game's major moments ahead of time is incredibly useful for ensuring the game remains consistent for the player and that each scene has a clear purpose for being included.

[5]Detailed storyboards should include a brief description of each major scene. These descriptions should include the main characters involved, the scene's significance to the overall story, and how the player is expected to interact within the scene. The intention is to ensure that each scene fits in with the overall vision of the game's story or experience.

Campfire Stories 📖

Every good story needs a cast and a good ending.

Standard

- 📖 Create an outline that shows the major events of the story
- 📖 The story has multiple characters

Encouraged

- 📖 The game has a fleshed-out story with multiple sections
- 📖 At least 1 character is developed

Extra Credit

- 📖 Story has a clear beginning, middle, climax, and end.
- 📖 Story has at least 1 secondary or supporting character who is developed equally with the main character

Explanation

Everyone loves a game with a captivating story and a satisfying ending. Well-developed characters and fleshed out stories will draw your players in and make them want to come back to your game again and again. Planning ahead and tackling your story early in your game's development will help the team stay on track and help ensure all team members are on the same page.

You should have an outline for your story which acts as an overview of your story including major plot points and other key decisions. There's no strict length or word count, as long as you've covered everything important and can walk a judge through your story.

Good stories also generally have multiple characters. Sometimes it's an old friend or enemy, maybe a love interest, or a close family member, but there should be at least one other character in the story. The who, what, when, where, and how of your story are completely up to you as long as your story starts somewhere, has some progression, and ends somewhere else, developing characters through your plot points.

For higher tiers, make sure your story builds towards a climactic event and finishes by concluding your major plots. Include a supporting character which your story spends some time on. They can't just be a quick blip on the radar, they need to recur and have a decent place in the story. To be clear, supporting characters are not an enemy type. They are characters in your story, they might be an enemy, they might be a companion, they might just be a narrative voice like in Bastion, but they need to have character development.

Scratching the Surface 📖

Games are much more immersive when there is a deep story and fleshed-out world to explore. Everything for this achievement must be expressed via gameplay as dialog, narration, carvings, graffiti, audio logs, books, etc.

Standard

- 📖 Write a background for at least 1 location or character and explain how it is presented to the player
- 📖 Write at least 1 bit of backstory for the world or environment of the game and explain how it is presented to the player

Encouraged

- 📖 Write a background for multiple locations or characters and explain how they are presented to the player
- 📖 Write multiple bits of backstory for the world or environment and explain how they are presented to the player

Extra Credit

- 📖 Create a reference guide (printed or uploaded to TMS) for important elements of the story including:
    - An illustrated glossary of all characters and locations
    - Characters and locations that were not implemented in your game
    - Ability, power, and item descriptions

Explanation

Players love games with rich lore and an expansive universe to explore. Hints of the universe continuing around the events of your story help to build immersion and keep players coming back to discover more.

Think about the characters in your game and write some detailed backstories for them. Consider where and when your game takes place and write some history. Think about how you want to portray these histories to the player, whether books, notes, or graffiti, and make sure the player is capable of piecing together at least a little bit of your stories!

For higher tiers, create a reference guide to the story, background, and characters present in the game. This reference guide should not be your design doc, this should be a fun supplement to your game that a player would be interested in exploring. The goal is to provide additional context and flavor for your game, as well as expand on details that didn't quite make it into the playable game itself.

Research & Development 📖

Research the season’s theme and document how that exploration impacted game development.

Standard

- 📖 Conduct research on the season’s theme and write a paragraph (at least 150 words) describing what you found, including at least 1 source.
    - Focus on describing specific details from the research that can be explored further and how they could apply to your team's project.

Encouraged

- 📖 Document how your team's research can tie into and influence your game in specific ways. Explore how what you learned can guide the music and art direction, as well as game mechanics.
- Team clearly communicates how they explored the theme and how their understanding of it developed through their research and development

Extra Credit

- 📖 Document additional research your team took as a result of the theme exploration research. Explore which topics were particularly interesting, and how could learning more about them enhance your game?
- Team is prepared to provide a clear presentation on their understanding of the theme and how it applies to their project
- Show how the team's understanding of the theme changed as the game's development progressed

Explanation

In most creative fields, it is rare to encounter a scenario where the first design that comes to mind is truly the best. Painters practice, books are edited, and even video games change over time as they explore their theme and test out how they can best express their creative vision. A critical step in designing a successful project is to research the themes that are present in order to understand how they can best be expressed. For example, a farming game that has made no attempt to learn about plants, soil, and weather has the potential to miss out on entire systems that add depth and creativity. Each one of those topics can again be expanded to learn about an entire new domain! While not every project needs to be grounded in reality, this additional exploration during development opens doors that can lead to fun innovations. Be prepared to share how your research helped elevate your project.

MANAGEMENT

Team Players 📖

The team interacted in a respectful manner, communicated well, and overcame challenges together.

Standard

- Team members met (in person or virtually via Zoom, Hangouts, Discord, etc.) at least once a month for a group work session
- Team describes a challenge they faced while working together as a team

Encouraged

- 📖 Team members met (in person or virtually via Zoom, Hangouts, Discord, etc.) at least once a month and kept a detailed record of what was worked on at each meeting

Extra Credit

- 📖 Team members met (in person or virtually via Zoom, Hangouts, Discord, etc.) at least once a week and kept a detailed record of what was worked on at each meeting

Explanation

Communication as a team is vital to creating a positive team environment as well as a polished game. Having trouble figuring out what to work on next? Send a message or email out to the group. Not sure how this bug is happening, or why Gimp isn't working? Send out a message to everyone. Found a cute cat picture you want to share with everyone? Make sure your whole team gets to see it! There are many free tools you can use such as Slack, Remind, or even just email. Pick a tool and use it through the development cycle to stay connected with your team.

For higher tiers, make sure you communicate as often as possible. Using review sessions and weekly check-ins can help keep the team organized and on schedule and answer questions that might be blocking progress.

Public Relations

Be professional! Be ready to pitch your game and make it memorable.

Standard

- Team members all wear coordinated clothing (printed, matching colors, or matching styles / themes)
- Appoint a team spokesperson who is ready to introduce your team and present your “elevator pitch”
- Create something for your table that clearly identifies your team (sign, banner, flag with your team name, etc.)

Encouraged

- Create a development blog/vlog/social media page that contains 5 or more posts focused on at least 2 different aspects of the development process (coding, art, etc.)
- Make your table appealing to visitors with at least 1 item of interest that will make someone want to stop by your table (a candy bowl, “swag” like stickers or pins, a small model of a game character)

Extra Credit

- Discuss how your game was advertised to the public before OGPC (playtest party, public blog, public release, posters in the school?)
    - Tell us how you engaged beyond the OGPC community about your game

Explanation

You know what looks cool? A whole team in matching outfits! While some teams do make T-shirts for their team, you do not have to buy new clothes for your team to look coordinated. You can all wear jeans and black t-shirts, or dress pants/skirts with white shirts. You could even dress up like characters in your game. You decide on your team’s look and make sure everyone matches.

You don’t have to spend much money to really bring the team spirit. You can paint or cut out a banner, you can design a brochure on the computer, or even have fun with handmade pins! Big shows like PAX and E3 are full of handouts to get people to remember the games and visit booths. Screenshots, descriptions, team information, game logo and your team logo are all good things to add to your items.

For higher tiers, describe how you marketed your game to the public before OGPC. From advertising within your school, communities, or other circles, getting public attention is a great way to get early feedback on how your game will be received and presents opportunities to make it even better!

Project Management

Use tools to help manage work: issue tracking, milestones, task management and more.

Standard

- Team uses some form of task management tool (Trello, Jira, Asana, Excel, GitHub Projects) and is prepared to present a dashboard (or equivalent) during judging

Encouraged

- Task management uses some form of grouping (tasks and subtasks)
- Tasks are assigned owners and state is tracked as work is completed
- Include bug management with tasks

Extra Credit

- Task management is broken into sprints or other milestones
- Add weights/priority to tasks

Explanation

Project management isn’t easy, but it's a crucial step in keeping a game on schedule and having it ready to compete with at the Main Event. Tools such as Trello or Jira (feel free to use whatever tool works best for your team) can make this process easier/faster and having everything in an electronic format makes it easier for everyone to be checking in even if they're working on tasks from home.

For higher tiers, group your tasks (the more tiers the better), track completion of individual tasks and sprints, use weights on your tasks, and track your bugs. This might seem like a lot at first, but it is how real-world projects are managed! The grouping of tasks, weighting of tasks, and tracking of tasks makes sure that work gets done and that nothing is being forgotten about. This process also helps prioritize which tasks to take on next to make sure your team is on track. Generally, there are fewer milestones than there are tasks: think of "complete all character animations" or "complete rough draft of story" as milestones and "make a walking animation" or "record a dialog line" as smaller tasks. Assign time estimates for all of your smaller tasks, add them up to see how long you expect your milestones to take.

Checking the Boxes

Complete TMS. All of it. I dare you.

Standard

- Fill out at least these fields on TMS: "About this Entry", Game name, Team member list
- Include a basic description of the game
- Leave no remaining placeholder images (team members, team logo, game poster)

Encouraged

- Fill out all game entry page details on TMS
- Include a playable link or download link to your game
- All team members are linked to game entry
- Include more than one sentence for the About section and the Instructions section

Extra Credit

- Game entry on TMS has 10 or more images (screenshots showing off different aspects of the game, mock “box art”, concept art, images of the team at work, etc.) and all images include descriptions
- Include a full game description (200 or more words, think Steam store page), instructions, etc.
- All team members have appropriate roles set in TMS and have profile images, and a team group image is included

Explanation

The Team Management System at https://tms.ogpc.info/ allows you to publish your game to the world on the OGPC site. It provides a platform for you to describe your game, your team, and how your game was created. Best of all, it allows others to play your game and check out other games.

No game on Steam or Epic would sell well without all the details filled out. TMS has sections for team logo, game logo, team group photo, and game screenshots. Make sure you add logos/screenshots/images wherever there’s space! Not only is it worth the achievement, our awards presentation gets generated directly from TMS, so you don’t want to be the team with “NO PICTURE” next to its name!

For higher tiers, you’ve got space for pictures on TMS, but you can also upload your game or link to it so people can play it. It’s so much fun to be able to play everyone else’s game, so make sure you give teams the chance to play yours.

Hype Train

The Game Trailer is well put together, and acts as an effective marketing piece to highlight the game's mechanics, theme, and story. The trailer should be designed to entice a viewer into wanting to play or know more about your project.

Standard

- Trailer video is submitted to TMS
- Trailer is scripted, well-practiced, and within the time limits (see Competition Manual)

Encouraged

- Trailer video has well-balanced, normalized audio (nothing is ear-blastingly loud or unintelligibly quiet)
- Trailer includes in-game footage
- Trailer is edited (no mistakes, blunders, or awkward gaps)

Extra Credit

- Create at least one additional trailer, promotional video, or behind the scenes video (at least 1 minute long)
- Videos have reasonable special effects (titles, fades, transitions, etc.)

Explanation

Do you drool over videos for the newest games on Steam and YouTube? It’s time to make your own! Capture some gameplay, add some titles, maybe even create a custom scene. Watch a few game trailers to see what you like about them, and then make something similar for your game. Some movie maker software even makes it easy to combine clips with music in different styles. Make sure to use music and voice clips that you have permission to use (same rules as your game assets) and don’t forget to cite them at the end. Even better, use your own content!

For higher tiers, make sure that everything fits together well, use clips of gameplay from your videos with your voices, and even use a storyboard for the videos just like for your game’s story. If you feel really ambitious, make an additional video for your game in a different style than the required videos. Trailers, promotional videos, and behind the scenes videos generate hype and help draw people into your game. Everyone loves learning a little more about their favorite games!

Changes and Amendments

The Oregon Game Project Challenge reserves the right to change, amend, modify, suspend, continue, or terminate any or all rules and regulations of the main event, including achievements, either in an individual case or in general, at any time without notice.

Change Log:

December 2024 – First release updated for 2024-2025 challenge year