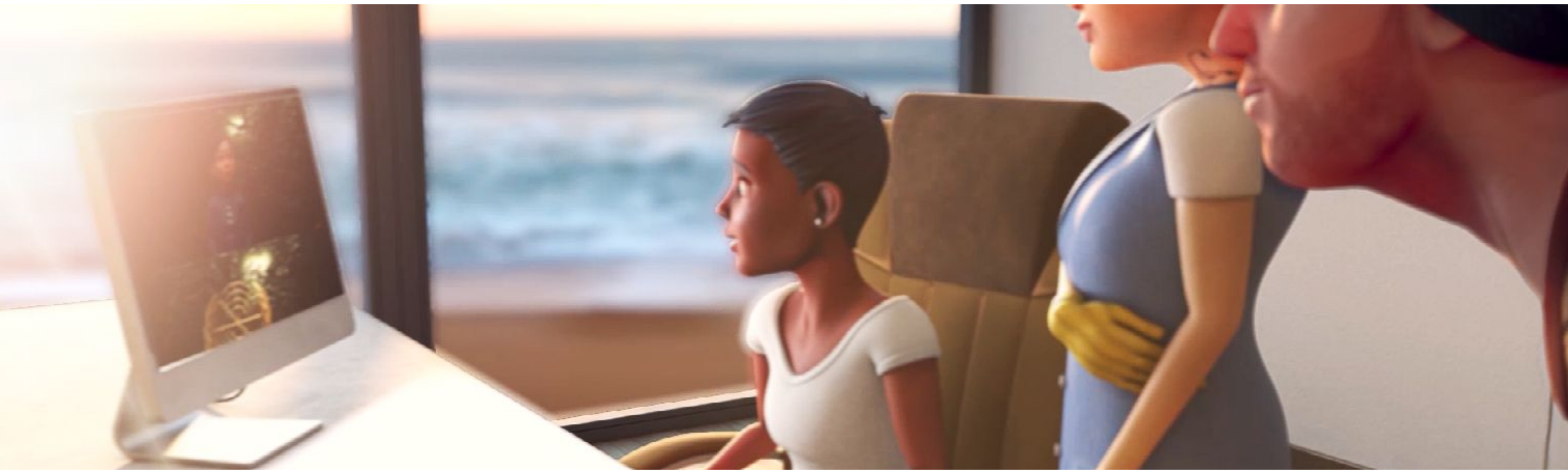
gamelearn

HOW TO CREATE A VIDEO GAME FOR TRAINING



Introduction

In this manual, you'll learn how to design a videogame for training purposes. We'll offer you a step-by-step guide on how to do so, taking advantage of Gamelearn's more than 15 years of experience as pioneers and world leaders in corporate training through the use of video games.

Discover Gamelearn's Editor, the #1 tool on the market for creating incredible video games for training without having to write a single line of code.

Editor

The three pillars of training with video games

A video game for training purposes must include the following three components:

1. **Content**: The learning portion of the course--the reason why students are taking the course.

2. **Practical elements**: Exercises, simulations, and dynamic portions of the course that allow students to put what they're learning into practice alongside personalized feedback.

3. **Story**: The narrative that ties together the course theory and practical elements of the game. A good story can help ensure high engagement and completion rates. Any type of adventure, thriller, or quest type of story can help pull the course together.

Step 1: Decide on what you'd like to teach

To design a high-quality video game for training, it's of utmost importance that your content be top-notch. The video game you end up creating is nothing more than the vessel by which your content will be presented to students. At the end of the day, if your content isn't good, your videogame never will be.

Attractive, interesting content is:

- practical
- useful
- applicable to one's professional and personal life

You must first decide on the content of your videogame.

1. **Define the learning objective:** Clearly establish what you expect students to have learned by the time they've finished playing.

2. **Create a course outline:** Write out the basic foundation of the course. Don't worry about the details at this point. Just think about the overarching concepts and large blocks of information you'd like to include in your course.

Your video game should last no longer than 3 hours so as not to overload students with too much information in a single game. The game being too long could drastically reduce students' retention of information and could have a negative impact on your completion rates.

Remember, three, 3-hour courses are always better than just one, 9-hour course. If you need to teach a lot of content, consider creating a "series" instead of a "movie". Divide the story into chapters. Spread out the learning experience.

Step 2: Edit down your course

Once you've made a general outline of your course, it's best to keep in mind that the average person has a very short attention span. This is true even for courses administered in video game format.

You need to remove all non-essential content. Take another look at your course outline and remove as much as possible.

Once you've finished, look over it again and continue editing. You should remove at least 20% of the content you originally accounted for.

Remember that beyond just going through the course's content, students also need time to learn how to play the game, run through simulations, and navigate the story. All of these other elements can take up to two-thirds of the playing time. That means that for a 3-hour game, you would only have an hour, at most, to transmit your content.

A good rule of thumb is to never include more content than you could cover in a one-hour lecture.

Step 3: Organize your content

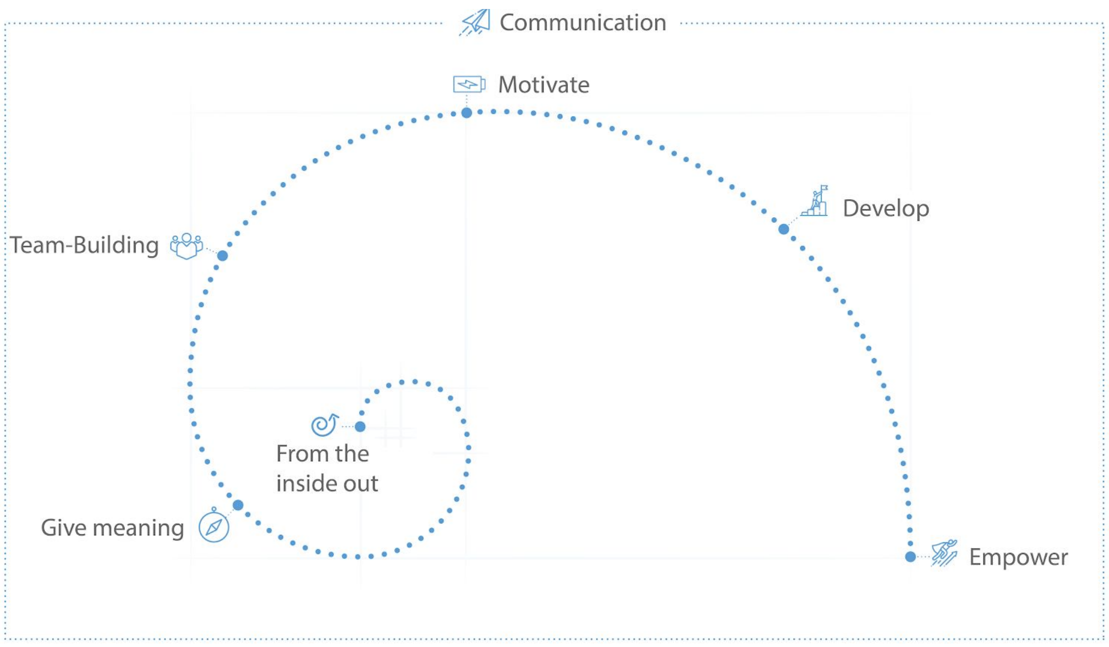
Once you’ve eliminated as much non-essential content as possible, it’s then time to establish the order you’d like to present it in. You must define the path students are meant to take to discover the content. You need to structure your content.

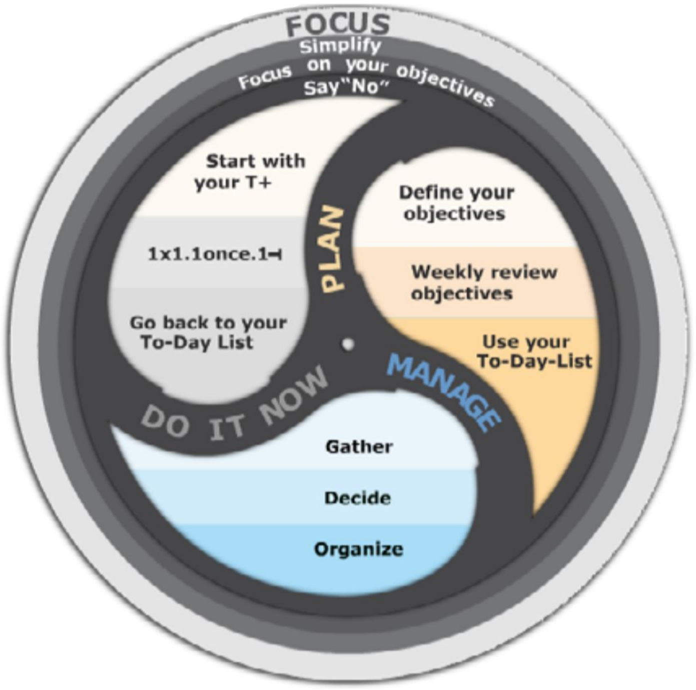
Divide your content into "lessons." Ideally, you would have around ten lessons. At this point, they don’t need to be written out, but you should define what they are going to cover. The lessons can be presented as text, video, audio, etc. This is how students will learn the course theory throughout the game.

Anyone reading, listening to, or watching the lessons should be able to easily understand what you're trying to teach.

Recommendation: Build a model that allows you to graphically represent the base structure of your content. This model will be very useful for students to better understand what you’re trying to convey.

For example:





Keep the following points in mind:

- **Gradually incorporate your content into the game.** First, teach a concept. Then, have students practice it. Once you've done so, you can then introduce a second concept. Students should practice the first and second concepts, after which the third concept is introduced. Repeat this method for every lesson.

- **Use repetition.** This has been proven to be an extremely effective way to ensure students retain what they have learned.

By the end of this step, you should have created a detailed structure of all your lessons as well as how you'd like them to be organized within the game.

Step 4: Define how students can put the content into practice

What makes game-based learning so special is how it enables students to practice and live out the skills they've learned, all while receiving customized feedback.

Your game must impart experiential learning. This step is when you should decide exactly how you'll make that happen.

The Gamelearn Editor has different possibilities. There are several building blocks that have been developed for this purpose. How you use them will depend on your specific needs and your creativity.

For example, if you're teaching your product's sales pitch, it would make sense for you to first explain the pitch through a lesson. After you've thoroughly explained it, you could then put your students in similar situations to ones they'd face in real life. In this example, you'd probably want to think about the typical conversations that take place between your sales reps and customers. You could then use Editor's "Dialogue" building block, which allows you to create such a conversation, leaving room for students to respond to questions. You could even offer feedback and score students however you like.

Alternatively, if you wanted to have students learn how to differentiate between two parts of an engine, you could use the block entitled "Mini-game: Clickable Image" where you could present different parts of the engine and ask students to select the correct ones.

If students are meant to read and understand a manual containing your corporate values, you could use the "Mini-game: Fill in the Blanks" building block to confirm the correct understanding of the values before allowing them to advance in the adventure. You don't necessarily need to limit yourself to information included in the game itself. You can ask questions inviting students to look for the answers on the intranet, internet, in manuals, books, etc. There's no need to unnecessarily limit yourself in this regard.

For each lesson, define an action or activity allowing students to practice what they learned.

You have two possibilities:

1. Explain the content first and then ask students to practice by correctly applying the theory previously explained.

2. Let students run through simulations before presenting the theory. Correct their mistakes and allow them to repeat the simulation once they've seen why their original answers were marked incorrect.

The practical elements of the game must be skillfully designed. They need to offer adequate feedback, explaining what students have done well and exactly what needs to be improved on.

Learn how to correctly manage the feedback options available to you in each of the Editor's building blocks.

Step 5: Define the story draft

Only once you've firmly established what you want to teach and how you are going to teach it, should you begin thinking about the story that will pull your content together.

At this point, everything depends on your own creativity. That being said, we do have some tips for you:

1. **Try to make the underlying story relate to what you are trying to teach**.

   In a negotiation course you could have players become merchants, entrepreneurs, agents for actors or athletes, politicians, police negotiators, etc. Players could advance through the game facing different negotiations to achieve a final objective: end up with a highly successful company, sign a lucrative contract for their client, become president, close a large international agreement, make a peace treaty, etc.

   In a leadership course you could have the player become the leader of a sports team, a nation, a professional team, an NGO, etc. and the end goal could be to overcome a great challenge, win a competition, complete a project, help solve an important problem, etc.

   In a customer service course you could have the player assume the role of any professional: a doctor, official, waiter, tele-operator, etc. You could have players face different situations in which they would have to solve problems with customers to reach any number of objectives: end up on top of the leaderboard, become the employee of the month, earn a promotion, become president of the company, etc.

2. **Get inspiration from your favorite books and movies.** Don't be afraid to create a story that resembles something you've seen or hear before. In the end, captivating stories often tend to follow the same patterns: a hero who saves the planet, a detective who solves a case, a researcher who finds a cure, an adventurer who finds a treasure, a traveler who discovers a hidden place, a survivor who overcomes a great challenge, etc.

If you want to get more ideas and improve your storytelling skills, complete this brief and interesting online storytelling course developed by Pixar (creators of Toy Story and Finding Nemo). Link.

Step 6: Write out the Lessons

After you've created a basic script, defined the lessons, put them in order, and thought of how they will be put into practice, you can start writing the final texts for your lessons.

Avoid being too theoretical, being too technical in your explanations, or providing long lists of definitions. Provide specific ideas and tools that students can use in real life situations. Students greatly appreciate it when you summarize information that gets straight to the point. They want a guidebook of solutions and tools--not an encyclopedia. Don't waste anyone's time. Make every sentence count.

There's always a better way of saying things: a clearer, more concise, simpler way. Many times, you'll find that something that, at first, seemed important is actually non-essential and should, thus, be removed. Write as if each letter cost you money. Set a goal for yourself to make sure your final text ends up being just half the length of your original one. Review your text as many times as necessary to summarize, cut, and fine tune it. It can't be stressed enough how important this is.

Any lesson should be able to be read (written lesson), listened to (audio), or viewed (video) in a maximum of 3 minutes. Less is more.

Before writing your text, consider the format you are going to present it in. Remember that Editor allows you to show your lessons in different formats: plain text, pdf, presentations, audio, video, etc. Depending on the format you choose, you'll have to modify the way you word the lessons. The tone and style of a written text is not the same as one that is meant to be read or acted out.

Depending on the type of content and the way your game is set up, you'll have to select the lesson format that best suits your needs. Don't assume that your lessons have to be in text format. You can include an audio or video recording of an expert in the field, you can make an animation video, an animated presentation, a video to which you apply comic style filters (if you don't have the resources to make an animated video), etc.

We recommend that you try to fit your lessons into the story. Have a character in the game teach the students, mention parts of the story in your lessons, and make the tone and language of the lessons match those of the game.

For more ideas on how to write your lessons, click on the following link.

If ten lessons aren’t enough, you can add additional documents through the use of "reference documents." Editor has a specific functionality to allow for this; however, we recommend that you limit the use of this feature.

Step 7: Define the game mechanics

1. Define the objective of the game:

   Some examples of game objectives are the following: Players must follow clues to find a treasure, players have to lead an investigation to find the culprit of a crime, players must manage a project to save the planet, players must grow their company by negotiating with different people, etc.

2. Define the "core loop":

   Every game has a basic mechanic that is repeated over and over again.

   Examples used in Gamelearn games:

   “Merchants” (negotiation)

   Players run through the following core loop:

   - negotiate with characters from the game
   - receive feedback on the negotiation
   - receive lessons from the mentor with tips for improvement
   - repeat negotiations to apply lessons learned from the mentor
   - grow business with the money made during the negotiation

   This loop is repeated 6 times, with 6 different negotiation cases, corresponding to the 6 levels of the game.

   “Triskelion” (time management)

   Players run through the following core loop:

   - manage email and receive feedback and points
   - manage calendar, activities, and tasks & receive feedback with points
   - search for clues to find out which new city to visit

- meet new characters who impart new lessons on time management
- pass tests on the lessons

This loop is repeated 20 times, corresponding to the 20 levels of the game.

Echo (coaching)

Players run through the following core loop:

- listen to lessons from a mentor
- pass tests on learning
- conduct coaching sessions with their coachee to apply what they have learned and receive feedback and points.

This loop is repeated 3 times, with 3 different conversations with your coachee, corresponding to the 3 levels of the game.

3. Define the scoring system:

Games are based on point systems. In essence, players accumulate points as they go through a game. The player with the most points ends up on top of the leaderboard and is, therefore, declared the winner.

Depending on the story you've created and the subject you're teaching, you'll decide the number of points assigned to each action and the reasoning behind them.

Throughout the game, the player will make decisions, answer questions, solve puzzles, etc. Each time this is done, you can award different types and amounts of points.

With Editor, you can divide your point system into three types: gaming metrics, skill indicators, and grades.

NOTE: You are not required to use points in your game. But it is highly recommended to increase engagement and improve completion rates.

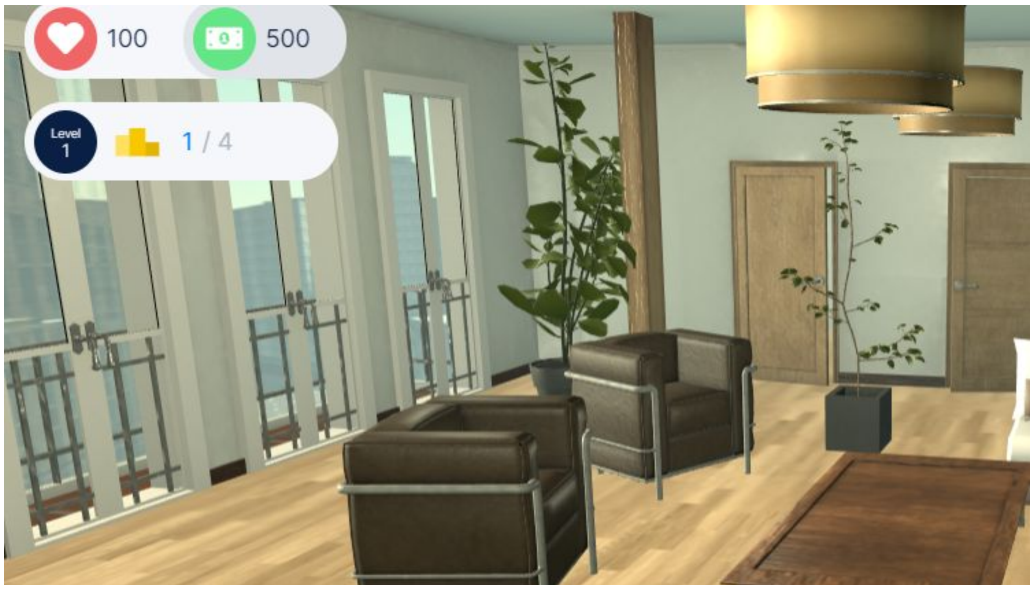
You can assign points corresponding to all the building blocks available in Editor. All building blocks give you the opportunity to add or subtract points (of the three types).

1. **Gaming Metrics:** This refers to the scoring feature related to gameplay-- the adventure and competition aspect of the game. This scoring indicator is generally used to determine the winner of the game.

   For example, in a sales game, you could use the total amount sold as a gaming metric. The player's objective would be to sell as much as possible.

   In a game on leadership, the player could accumulate leadership points. In an adventure story, the player could have to work to add to their life or strength indicators. In a game on stress management, the player could have to minimize stress points or increase energy points, etc.

   These indicators are always present in the upper left-hand corner of the player's screen..



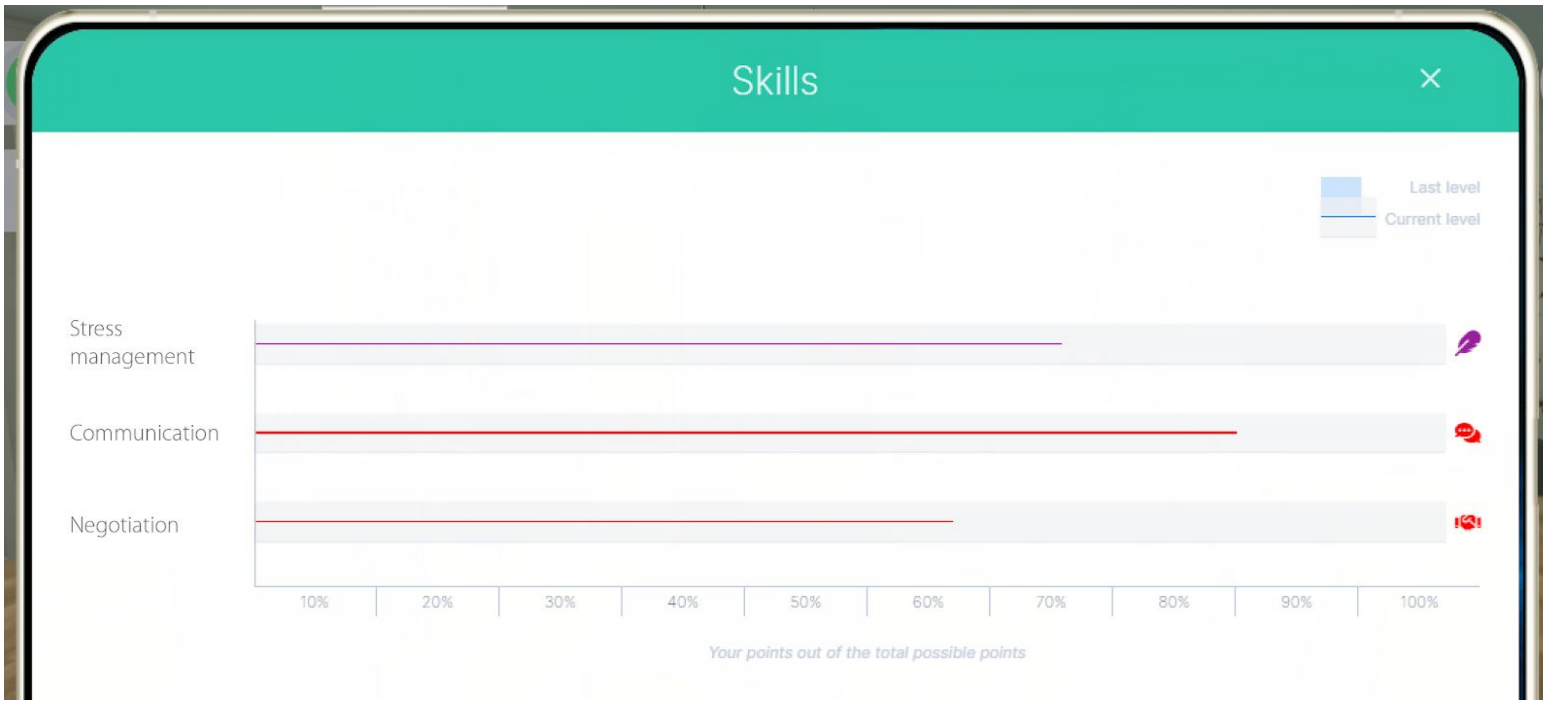
You can define up to 4 different gaming metrics in a single game.

2. Skill Indicators: This refers to what you are trying to teach with your game.

   For example, in a course on stress management that goes through an adventure-filled, treasure hunt story, the gaming metric would correspond to the value of the treasure found and the skill indicator would be regarding stress management.

   Throughout the game, each time the player finds treasure he makes "money" (gaming metric) and, as he answers questions, makes decisions, or solves problems related to stress management, the player gets "stress management" points (skill indicator).

   During the game, players are able to visualize the points gained in each skill for evaluation purposes.



You can define up to 8 skills indicators for each game.

3. **Grades:** This refers to the numerical scoring method you can use to evaluate students. While gaming metrics and skill indicators are visible throughout the game, grades are only available to players at the end of the course. You can keep it for yourself, show it only to the player, or share it with all players in the final ranking.

   Editor allows you to set the range of grades seen in your game. For example you could set anything below 50 points as an “F”, from 50-60 as a “D”, from 70 to 80 as a “C”, from 80 to 90 as a “B”, and from 90-100 as an “A”. You can come up with your own breakdown of points and the terminology for each tier.

Step 8: Design the game navigation

In this step, you should define how you want players to move throughout the scenes of the game.

You will have to decide if you want the players to move freely around the scenes or if you want to limit their movement (and move them yourself), if you want them to use the map, or if you would rather they use doors, elevators, or other elements in the scenes to move around. You can also choose to use a combination of both options.

If you're going to opt for using the map, define the scenes you would want to appear in it and when you would want them to become accessible or inaccessible to the player.

At this point, it's important that you have a full understanding of how the building blocks "activate" and "deactivate" work as well as the concept of "visible/invisible" and "active/inactive scenes" and "map configurations.

Step 9: Write your game's outline

In a shareable, easy-to-navigate document, write out the steps players will have to follow to go through the game from beginning to end.

To create your outline, remember the basic elements of Editor:

1. **Levels:** Players must feel like they are moving and evolving through the game. Your story may require you to recreate different stages, days, phases, etc. Levels can be used to achieve this.
2. **Scenes:** In each level, you can create different scenes where events can take place. Different characters and objects can be found in scenes. A level can have more than one scene.
3. **Flows:** This is the chain of actions and events that take place when a player clicks on different characters and objects in the scene.

Your game's outline is the simplified structure of your "movie script." You already know what you want to teach, you know how you want students to practice, you have decided the plot of your story, and you have defined the mechanics of the game. Now you must create a summary, step by step, of what will happen in the game.

In order, you should build the skeleton following these steps:

1. Structure the levels of the game.
2. Determine the scenes you will include in each level.
3. Specify the flows of actions that will happen in each scene.

Be careful in how you mix together the theoretical content, practical exercises, and the story. All elements must be well-balanced and distributed throughout the game.

Here's an example of how a project management course could be set up*[1]:

[1] *Simplified overview.

1. Story

A meteorite is heading for Earth. Its impact would mean the end of humanity. Players must lead/manage a project to build a spaceship that can destroy the meteorite. Players have three virtual months to complete this project.

2. Mechanics

2.1 Purpose of the game:

Players must successfully lead a project to save the planet by destroying a deadly meteorite.

2.2 Core loop:

- The mentor gives a lesson on project management.
- The player applies what they learned in the previous lessons to face different project management situations. Decisions will be made, feedback given, and points added or subtracted based on the player's actions.
- Different storytelling elements are applied to tell the story in an exciting way that keeps players engaged throughout the whole game.

2.3 Scoring system:

- Gaming Metrics:
  - "Space points"

- Skill Indicators:
  - "Leadership"
  - "Projects management"

- Grades:
  - Not applicable

3. Skeleton

Level 1: A large meteorite is heading straight towards earth. The President assembles a special team to build a spaceship that can destroy it.

- Scene 1: The protagonist's office: The Minister of Defense informs the player that a national emergency requires them to go to the operations center immediately.
    - Flow 1: **Minister** (character included in the scene) > a "**Dialogue**" is launched when clicking on the Minister > a **New Objective** is revealed to the player ("Go to the center of operations").

      Minister > Dialogue > New Objective

    - Flow 2: Project Management **Manual** (object included in the scene) > (Upon clicking on the manual...) **Dialogue** ("This information will be useful to manage the project") > **Test** (to confirm the student has understood the manual)

      Manual > Dialogue > Test

    - Flow 3: **Hammer** (object included in the scene) > (Upon clicking on the hammer, the protagonist performs the action...) **Object Collection** (this adds the object to the protagonist's inventory of objects to be used later in a mini-game).

      Hammer > Object Collection

    - Flow 3: **Door** (clickable area defined in the scene) > **Change Scene** (upon clicking on the door, the protagonist is taken to Scene 2: Center of Operations)

      Door > Change Scene

- Scene 2: Center of Operations: The President informs the player of the meteorite and instructs the player to lead a project to stop it.

    - Flow 1: **President** (character you've added to the scene) > "**Dialogue**" (when clicking on the President, a conversation is launched where the president instructs the player to stop the meteorite) > **Complete Objective** (complete the objective established in the previous scene - "Go to the center of operations" -) > **New Objective** (establish a new objective for the player - "Stop the meteorite") > **Change Scene** (after completing this flow, the protagonist is sent to the mentor's house).

      President > Dialogue > Complete Objective > New Objective > Change Scene

- Scene 3: Mentor's House: The protagonist speaks with the mentor to get some guidance. From this point on, the mentor will help the protagonist manage the project.

    - Flow 1: **Mentor** (character you added to the scene) > a "**Dialogue**" is launched when clicking on the mentor (the protagonist asks the mentor for help) > **Text** (the first lesson on project management) > **Test** (to test understanding of the lesson) > **Dialogue** (the mentor wishes the protagonist luck) > **Change Scene** (the protagonist is sent to the Center of Operations to begin working).

      Mentor > Dialogue > Text > Test > Dialogue > Change Scene

- Scene 4: Center of Operations: First Team Meeting

  - Flow 1: **Automatic Trigger** (upon entering the scene, this flow automatically begins) > **Dialogue** (first conversation between team members) > **Comic Strip** (you would have to include a visual representation of the launching of this project using both images and texts).

    Automatic Trigger > Dialogue > Comic Strip

  - Flow 2: **Computer** (an object in the scene) > Upon clicking on the computer, the mini-game, "**Mini-game: Writing**" begins (the protagonist must write an email asking to prepare a workspace to construct the spaceship).

    Computer > Mini-game: Writing

  - Flow 3: **Door** (clickable area you defined in the scene) > **Change Scene** (upon clicking on the door, the protagonist is taken to the workspace) > **Loop** (defined so that this flow repeats every time the player clicks on the door).

    Door > Change Scene > Loop

- Scene 5: Workspace: The player organizes the construction of a spaceship with builders and technicians.

  - Flow 1
  - Flow 2
  - ...
  - Flow n

Level 2: The team works against the clock to build the ship.

- Scene 1: Coffeeshop: The mentor teaches the protagonist a lesson.
    - Flow 1
    - Flow 2
    - …
    - Flow n
- Scene 2: Center of Operations: The player will have to make decisions, answer questions, and resolve conflicts to carry out the project.
    - Flow 1
    - Flow 2
    - …
    - Flow n
- Scene 3: Construction Site: The player talks with a technician about an issue that could prevent the spaceship from taking off. The player will have to solve a puzzle to get the spaceship back on track.
    - Flow 1
    - Flow 2
    - …
    - Flow n
- Scene 4: Protagonist's House: The news reports progress on the project.
    - Flow 1
    - Flow 2
    - …
    - Flow n

Level 3: The team travels in the spaceship to the meteorite and destroys it.

- Scene 1: Coffeeshop: The mentor teaches the protagonist a lesson.
  - Flow 1
  - Flow 2
  - ...
  - Flow n
- Scene 2: Spaceship: The player will have to make decisions, answer questions, and resolve conflicts in order to get the ship to reach the meteorite on time.
  - Flow 1
  - Flow 2
  - ...
  - Flow n
- Scene 3: Meteorite: The player will have to solve a puzzle to destroy the meteorite.
  - Flow 1
  - Flow 2
  - ...
  - Flow n
- Scene 4: Mentor's house: The news reports on the success of the project. End of Game. Student Survey.
  - Flow 1
  - Flow 2
  - ...
  - Flow n

Step 10: Write the script

When you have finished writing the outline, it's time to write the script. All definitive texts to be used in the game are to be written at this time.

If in the skeleton you wrote “Dialogue with the Minister”, you would then need to write out exactly what that dialogue would include. Where you wrote “Final Comic”, you would now need to write out the complete text for that comic. If you planned on including an “End of Game Message”, you would need to write out that text at this point, etc.

Step 11: Build the skeleton and decide what additional resources you'll need

10.1 Skeleton

The "skeleton" is the general structure of the game you build in Editor. It consists of all the levels and flows you have previously defined, not including the text. You're just looking to recreate the skeleton you've already created.

The idea is to have a dry run through of the game to make sure everything is working correctly and fits together before adding the text. Doing so will save you time and make it easier to make adjustments later.

For example:

- Create the "Dialogue" block but don't fill in the actual text. Name it something like "first conversation with the team."
- Create the "Text" block but don't fill in the actual lesson. Name it something like "Lesson 1: Introduction to Project Management."
- Create the "Image" block to show the project management model, but only upload a provisional image with a white background that says "Model".

This way we can make sure the moving parts work and make sense together as a skeleton version of the game before filling in the details.

First build the skeleton of the game. Then, later, add the body (texts, images, videos, sounds, etc.)

10.2 Additional materials list

As you're building the skeleton, make note of all the additional materials you'll need to fill in later. For example, you'll see that you may need background images for some mini-games, video or audio, perhaps you'll have to create another graphic, image, or drawing, etc.

Compile all the materials you know you'll need in a list.

Put everything from this list in a folder entitled "Game Materials."

10.3 Material creation

Create all the materials from your list (images, audio files, videos, etc.)

Include all the materials in the same folder where you stored the list.

Having all your materials together in the same folder will save you a great deal of time and effort when it's time to "upload" everything to Editor.

Step 12: Revise the game navigation

Make sure players can comfortably and intuitively access all the scenes.

Carefully check that the scenes' activation and deactivation settings are programmed correctly.

Make sure you have limited the access to the scenes you meant to appear later in the story (through the use of doors and maps). Be sure the door loops work correctly and that all necessary scenes are accessible when they need to be.

Step 13: Fill in the skeleton

At this point, you have to complete the skeleton with the rest of the content, materials, texts, etc.

Block by block, go through all the sections and upload the texts, images, videos, audio files, etc.

If you're well organized, you'll be able to copy and paste right from your script to the Editor. Remember, you have your "Game Materials" folder with all the resources you wanted to put in.

Step 14: Test

Once your game is ready, it's important to carry out various tests with people who were not involved in developing the game.

It doesn't matter how many times you have checked and double-checked the game, there will ALWAYS be mistakes that slip through the cracks. You need other people with fresh eyes (beta testers) to play the whole game and catch mistakes.

When they play through, get their feedback, and ask them to report all the errors they find.

With their feedback, you should be able to find the parts of the game that aren't clear or cause confusion, incorrect flows, places where people might get stuck, etc.

You need to understand and correct all the reported mistakes.

You'll have to carry out at least two tests: the first should be a group of 3-4 people to catch the obvious mistakes. The second should be a group of 10-30 people to find the mistakes that are harder to find. This way, you won't be wasting the bigger group's time and you'll get the most out of their effort and feedback.

NEVER release the game to your clients without doing these tests. Every small error that should have been caught in beta testing is amplified when real players see them.