FORTNITE

BUILDING AN OBSTACLE COURSE:

COLLISION DETECTION, TRIGGERS, AND EVENTS IN FORTNITE CREATIVE

STUDENT GUIDE

OBJECTIVE

Students will create an obstacle course using a creative array of devices that support collision detection and triggering of events.

BACKGROUND CONTEXT

Obstacle courses have been a great source of entertainment as well as a test of ability since the 1800s. Generally speaking, an obstacle course is a timed challenge where the player has to complete a course that has many obstacles along the way. The player needs to demonstrate speed, strength, agility, and sometimes, a cunning mind to outsmart certain challenges.

Have you ever made your own obstacle course at home, outside or maybe in school or camp? Have you ever seen any shows on television or videos online of people or animals trying to complete an obstacle course?

You are going to use Fortnite Creative to create an entertaining and challenging obstacle course. While creating your obstacle course, you will be leveraging the computer programming concepts of collision detection and event-driven design.

READY, SET, BUILD!

COLLISION DETECTION

While collision detection is happening in many of the things we do on computers, especially in video games, it is easy to take for granted. When collision detection is working properly, your games and programs operate as you would expect. If collision detection did not work, you would not be able to attack enemies, score goals or pick up coins. Games would be unplayable without collision detection.

When visualizing any action on a computer, it is typically in either two dimensions (2D) or three dimensions (3D). An example of 2D would be a traditional side-scrolling game, like the Dino Jump game in Google Chrome. An example of a 3D game would be Fortnite.

In 2D games, the detection of a collision is simply the moment when the pixels from two different objects overlap. This becomes a little challenging when the different elements have transparent pixels. Depending on the desired result, computer programmers need to build collision detection to work as expected.

There is an ever-growing world of applications for 3D collision detection. In a 3D environment, the shape of an object needs to be defined, and the collision has to be detected from all directions: left/right, top/bottom, and front/back. Games benefit from it, but there are many more real-world applications.

3D PRINTING– When designing a new object for a 3D printer, the act of combining shapes is a type of collision. The area where the two objects collide will help the printer decide if this is an inside or an outside area. With this information, the object can be printed efficiently.

SELF-DRIVING VEHICLES– There are many interesting challenges when designing a self-driving vehicle. Since vehicles are large and fast-moving, collision detection is important. Actually, we should take this one step further and call it collision-avoidance to keep everyone safe. Programmers would use advanced math to predict collisions and make decisions about how to best avoid them.

ROBOTIC AUTOMATION– When using robots to help perform tasks, they must be given additional senses to be more accurate, or to be safe around humans. Detecting possible collisions and safely avoiding them are important requirements for automated robotic systems.

EVENT-DRIVEN PROGRAMMING

Most computer programs you interface with today are using an event-driven concept. This means that the program will spend much of its time waiting for an action, or event. If you are reading a web page, the web browser is waiting for you. When you click a link, an event occurs. The program is designed to detect your mouse click, then determine what needs to happen next. If you click a website link, the click event handler will notice that you clicked a link, and will read the link to determine which website to load. Once it gets the website address, it will tell your browser to load, and it will resume waiting for your input.

In Fortnite Creative, you can simulate events by using communication channels in devices. For example, you could set an explosive to detonate when receiving a signal from Channel 5. You can place a trigger that will detect a collision with the player and send a signal on Channel 5. This means that if the player touches the trigger object, the explosive will detonate.

This last example shows how collision detection and event-driven concepts can be used together to create your incredible obstacle course.

When this project is complete, you will have a playable obstacle course—and you will have successfully implemented the important concepts of collision detection and events. If you have fun and are curious to learn more, there will be extension activities. You will also be able to dive deeper into these topics online by pursuing your own curiosity and inventiveness.

PROJECT OVERVIEW

To create your obstacle course, you will build a series of prototypes to prove that you understand how to create various challenges. The following instructions will take you through the process of building your catalog of challenges so you can assemble a completely unique obstacle course.

Let's start a new Grid Island and get busy!

GETTING STARTED

If you need a review on creating your island, refer to the Getting Started Guide.

STEP 1: PLACING YOUR INVENTORY ITEMS ON YOUR ISLAND

PLAYER SPAWN

First, place a **Player Spawn device** in the center of the map on the white square. Remember, the Player Spawn is in your **Traps** slot. To access this slot, you must be in Building mode. If you have any of the building elements equipped, you can use the mouse scroll wheel or key mapping to select the fifth Traps slot. When the Traps slot is selected, right-click your mouse to alternate between items that are equipped to this slot.

Select the Player Spawn in the Traps slot.

Place the Player Spawn on the white square in the center of the map.

Although configuration options are available for the Player Spawn device, no changes are necessary.

With the Player Spawn device placed on the island, your player will appear at this location when entering your island instead of skydiving. This will save time, and your player will always start in the same location.

STEP 2: PROTOTYPES–TRAPS

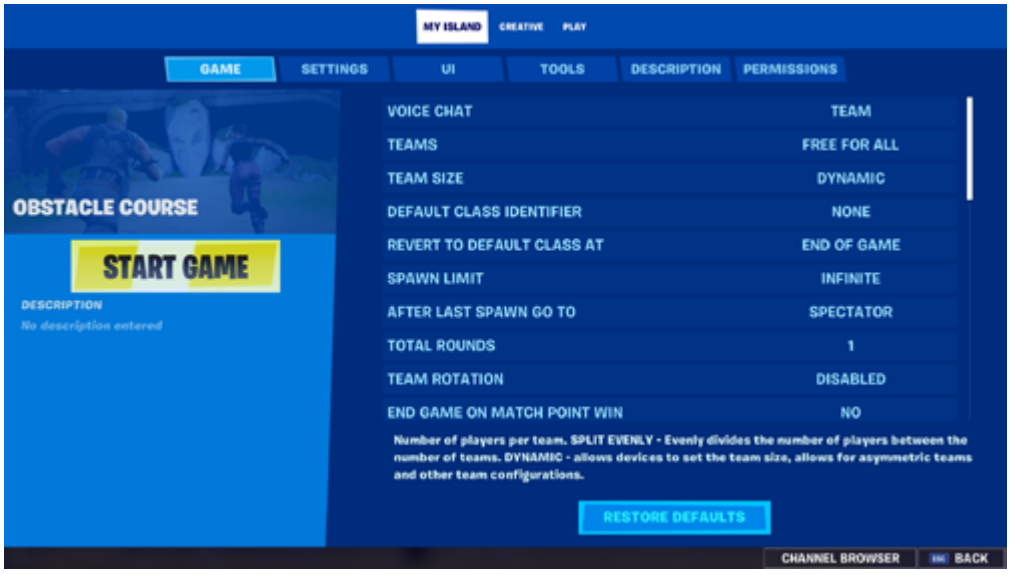
		
Damage Trap	**Poison Dart Trap**	**Full Damage Rail**

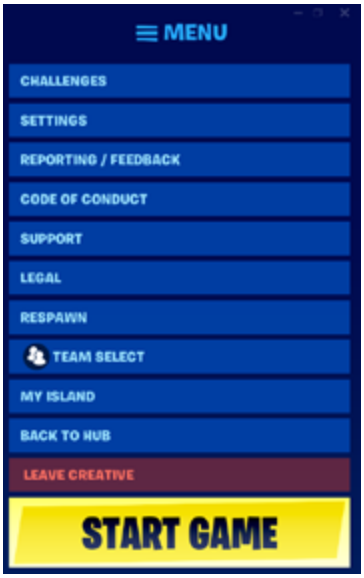
Build three small corridors near your Player Spawn, as shown, to test your traps.

	Place the **Damage Trap** in the center section of the first corridor. The Damage Trap can be placed on floors, walls or ceilings. It cannot be placed on ramps or cones, but can be hidden underneath these objects.
	Place the **Poison Dart Trap** in the center section of the next corridor. Poison Dart Traps are similar to the Damage Traps except the darts can travel and reach further than the Damage Trap.
	Place the **Damage Rail** in the third corridor. Notice that the inventory has both full-size and half-size Damage Rails. These obstacles can only be placed on the floor and around the edges when used as a building material.

Can you outsmart these traps? Start the game to test them. (Did you remember your Player Spawn plate?)

To Start Game...	
Access the **Start Game** button from the **Inventory -> My Island** menu	Press the Esc key.





TESTING YOUR TRAPS

Now that you have placed these traps, complete the process by understanding how they work. This way, you will know how and when to use them in your final obstacle course.

	**Damage Trap** How much damage does the player get if hit by the trap? How can the player avoid the trap? How can you make this trap more challenging?
	**Poison Dart Trap** How much damage does the player get from this trap? How is this trap different from the Damage Trap?
	**Damage Rail (full and half sizes)** How much damage does the player get from this trap? What happens to the player when they hit the rail? How does the player avoid this trap?

BUILDING VS. PROP

By default, any building materials will snap together, fitting each piece inside a single grid space. This is useful for building consistently with clean layouts. More importantly, these pieces are structural, meaning they can support other builds.

The **Phone Tool** provides an option to convert building supplies to props when copying and pasting. The benefit of converting things to props is that you can resize, rotate and position them more freely. The disadvantage is that these objects lose their structural quality when converted to props.

As you can see in the image, Full Damage Rails snap to the edges of the grid. However, with **Building to Prop** set to **ON**, the Full Damage Rails can be positioned more freely and the still function. (Note: Not all building materials can be converted to props.)

By understanding collision detection, you can take advantage of the weaknesses of certain traps. In the case of the Damage Trap, there are two ways to bypass the trap.

Collision detection for triggering the trap does not reach across the full width of the grid. If you carefully run along the opposite wall, you can avoid triggering the trap.

If the game designer wants to prevent this, they may put traps on both walls—and the floor—and the ceiling. Ugh!

If you cannot bypass the trap, you can trick it, because the trigger collision detection and the damage collision detection are not equal.

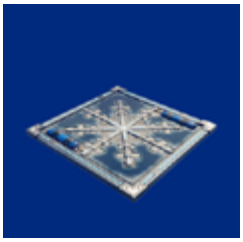
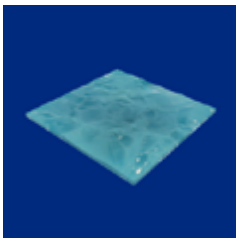
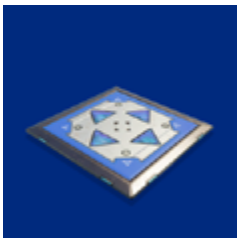
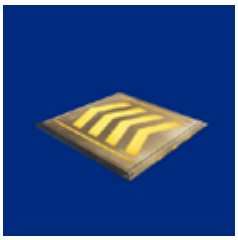
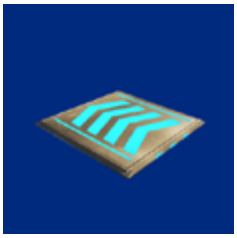
This means you can slowly approach the trap and hit the trigger detection without being in range to take damage from the spikes.

After the spikes trigger and miss, you can quickly run by before the trap resets.

Let's explore some other types of obstacles!

STEP 3: PROTOTYPES–EFFECT TILES

Next you will explore some items that affect the player movement. While these are not directly able to damage the player, they affect the movement in some way that could make it difficult for the player to complete a challenge on the first try.

				
Chiller	**Ice Block**	**Bouncer**	**Speed Boost**	**Movement Modulator**

You are continuing to build your arsenal of obstacle course prototypes, and can implement each of these to observe their impact on the player. You are also learning the language of Fortnite Creative. With normal programming languages, you learn the vocabulary, or syntax of the language to execute commands. Since Fortnite Creative is a 3D experience for the player, you are not using a typical programming language, but are using in-game building and devices to achieve a similar result.

Testing the Effects of these Devices

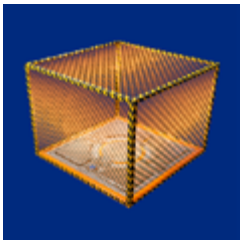
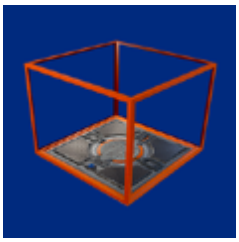
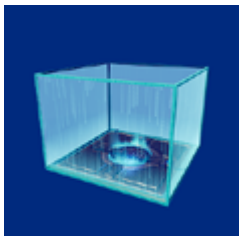
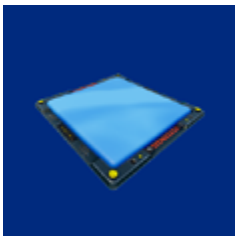
	**Chiller** What happens when the player walks on the chiller? How long does the effect last?
	**Ice Block** What happens if the player tries to stop on the ice block? Does this effect seem to work for time or some distance?
	**Bouncer** What happens when the player steps on the bouncer? What effect is applied to the player when they hit the bouncer? (Hint: Does the player take damage?)
	**Speed Boost** What happens when the player steps on the Speed Boost? What if the player hits this device from different directions?

	**Movement Modulator** Did you notice that this device can be customized? What is the primary way this device affects the player? What attributes can be customized? Did you see that this device can be invisible during the game? (**Customize -> Visible in Game**)

When you are finished with these prototypes, you will have five more obstacles that can be incorporated into the mayhem of your obstacle course.

STEP 4: PROTOTYPES-VOLUMES

Your next adventure will be testing devices that have a configurable collision area. You can customize the width, height and length of these devices.

			
Barrier	**Damage Volume**	**Mutator Zone**	**Sequencer**

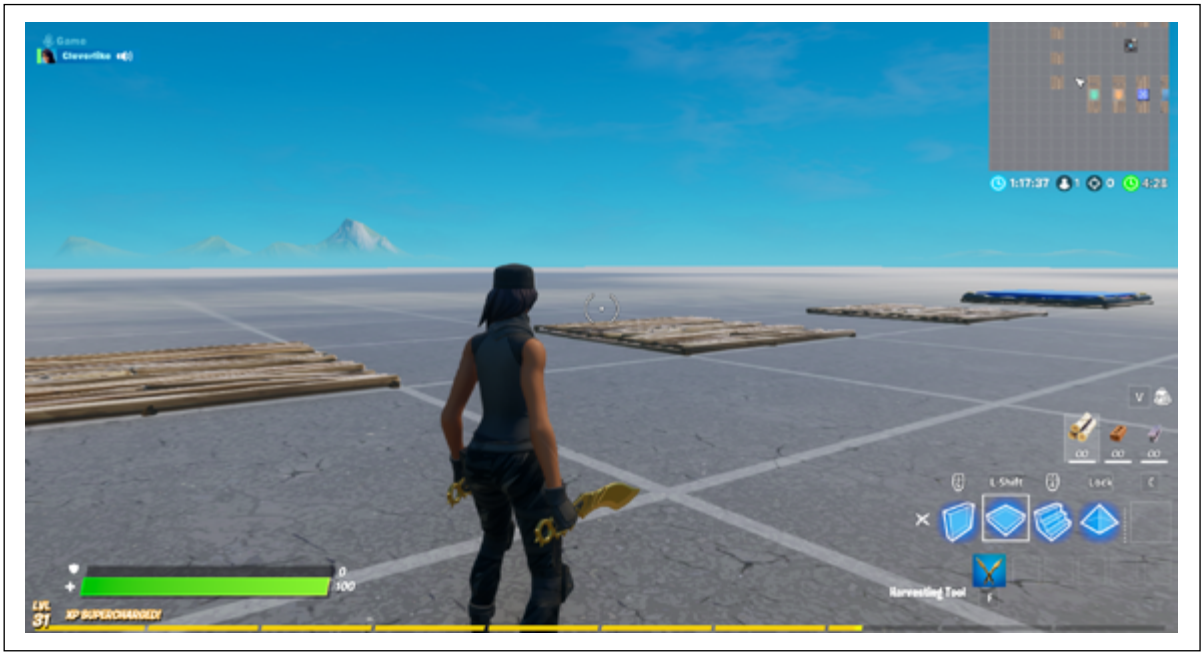
Since these devices have a configurable size, they can be used in both gameplay and in helping to manage the flow of the game.

Set up each of these four volumes so you can experiment with them and understand how they can be useful in your obstacle course.

Each of these devices has several customization options. These devices bring the concepts of collision detection and event-driven programming together into a powerful device.

When placing these devices, you will have to place a floor first, then place the device on the floor. It will not work on props, walls, ramps, cones or edited floors.

If your Sequencer volume faces the wrong way in your setup, don't worry—this can be configured with the **Zone Direction** setting when customizing the device.



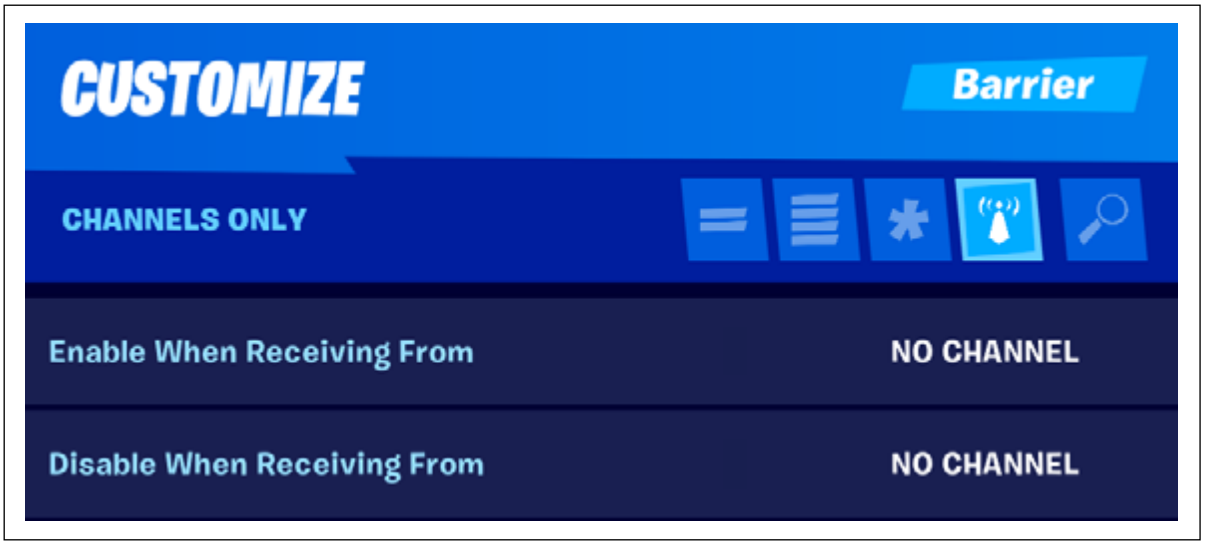
When placing these devices, you will have to place a floor first, then place the device on the floor. It will not work on props, walls, ramps, cones or edited floors.

BARRIER

The **Barrier** is designed to prevent a player from accessing certain areas.
Here are some of the customization options available.

Barrier Style	Change the look of the barrier. Change from invisible to another design. Try the nebula and starfield settings.
Base Visible During Game	Do you want the player to see the device in game?
Enabled During Phase	Should the barrier be active at the start of the game or should it be off? (If off, it would need to be triggered by an event to be activated.)
Zone Shape	As you make the size bigger, should it be a hollow box or solid? A hollow box can be used to contain the player in a single large barrier.

Events supported by the Barrier device.



The Barrier can be enabled and disabled during the game by using communication channels. Various methods for transmitting a signal on a specific channel are outlined in the next step.

Testing the Barrier

Customize the Barrier that has already been placed in your island.

- Set **Base Visible During Game** to Yes.
- Select a Barrier Style other than **invisible**.
- Make sure **Enabled During Phase** is set to **All** or **Gameplay Only.**
- You can make slight modifications to the **Barrier Width, Barrier Depth**, and **Barrier Height**.

After choosing your customization settings, start the game so you can see your Barrier in the game. In this image, the Barrier Style was set to **Brick** and the size was changed to 3 x 3 x 1.

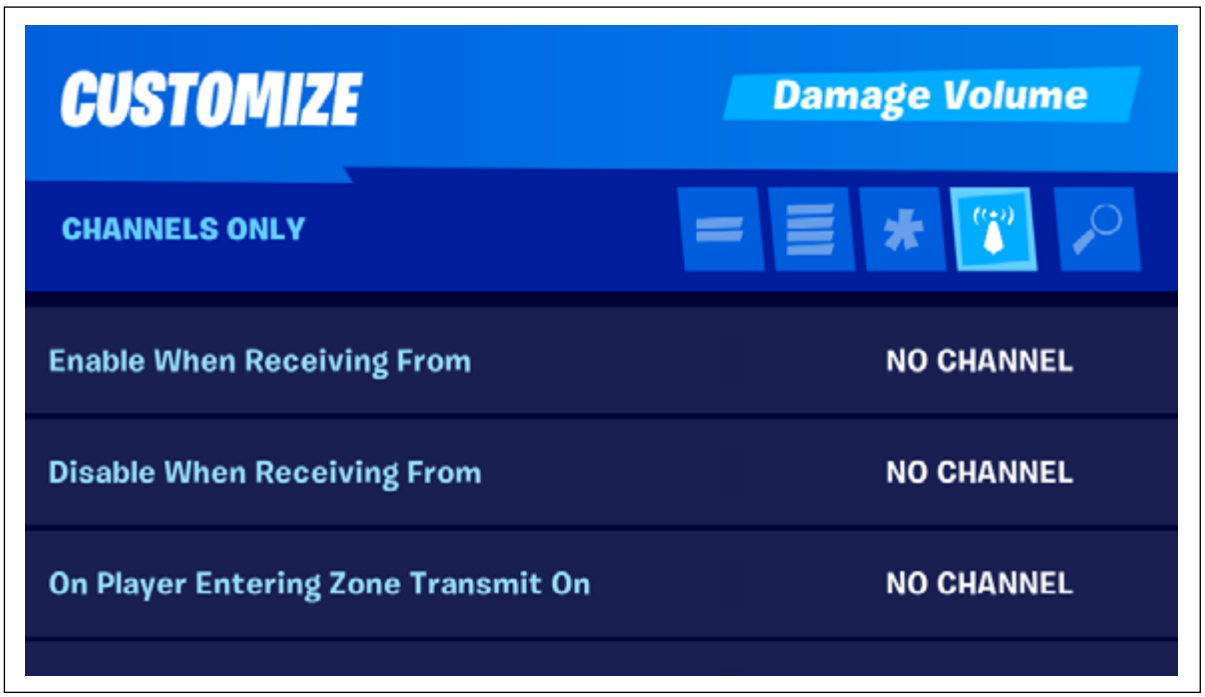
DAMAGE VOLUME

Damage Volume allows you to customize the amount of damage delivered to the player when they are within this area. The player can be eliminated instantly or they can receive a certain amount of damage at a timed interval.

Damage Type	Elimination or Damage Over Time
Damage	If Damage Over Time, how much damage?
Damage Tick Rate	If Damage Over Time, how often is damage applied?
Shield Damage	Does the damage subtract from shields first or immediately from the player's health?
Affects Players	Make sure this is Yes.
Affect Vehicles/Affects Unmanned Vehicles	Check these if using vehicles.
Enabled at Game Start	If No, it will have to be turned on in game using communication channels. If Yes, it can be disabled in-game with communication channels.

EVENTS SUPPORTED BY THE DAMAGE VOLUME

In addition to enabling and disabling this device by receiving a signal on a channel, the device can also transmit a signal when a player collision is detected. It is capable of sending a signal when the player enters the zone, and again when the player leaves the zone.



Testing the Damage Volume

Test the Damage Over Time function of the Damage Volume.

- Set Zone Visible During Game to Yes.
- Set Damage Type to Damage Over Time.
- Set the Damage value to 5.
- Set the Damage Tick Rate to 2 seconds.

When you start the game, you should see the Damage Volume, with the orange warning stripe moving from top to bottom. When you enter the zone, you will notice a smoke effect on the player, and 5 health removed from the player every 2 seconds.

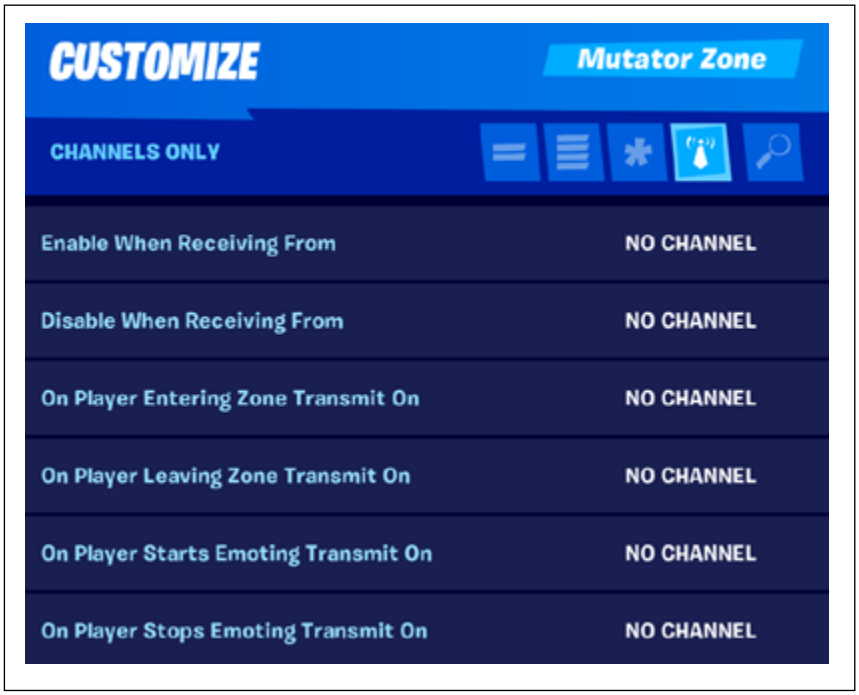
MUTATOR ZONE

The Mutator Zone is intended to control the behavior of a player when they are inside the volume. It can be used to disable weapons and disable building. The Mutator Zone can also detect if the player is emoting, so you can use it to trigger events when the player emotes.

Allow Weapon Fire	Default is No.
Allow Building	Can the players build when in this area?
Pick Up Life Span	This setting can be used to limit the time an item sits on the ground. If set to Destroy Immediately, it would prevent users from sharing items.

EVENTS SUPPORTED BY THE MUTATOR ZONE

This device supports enable/disable an enter/leave, just like the Damage Volume, but it also adds support for emotes. It can detect when the player is within the zone and when players start and stop emoting. A signal can be sent on a channel when emoting starts and on a different channel when emoting stops.

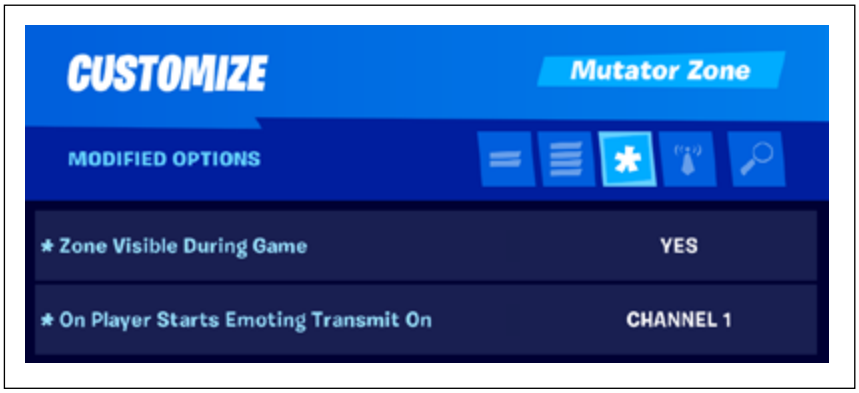


On Player Entering Zone Transmit On	Send a signal when player enters zone
On Player Leaving Zone Transmit On	Send a signal when player leaves zone
On Player Starts Emoting Transmit On	Send a signal when player emotes in zone
On Player Stops Emoting Transmit On	Send a signal when player stops emoting in zone

TESTING THE MUTATOR ZONE

It is time to introduce the idea of events using the device channels. Turn off the Damage Volume from the previous example if the player emotes in the Mutator zone.

Set the **Damage Volume** to **Disabled** when receiving a signal on **Channel 1**. Walk up to the Damage Volume and select **Customize**. Change **Disable When Receiving From** to **Channel 1**.



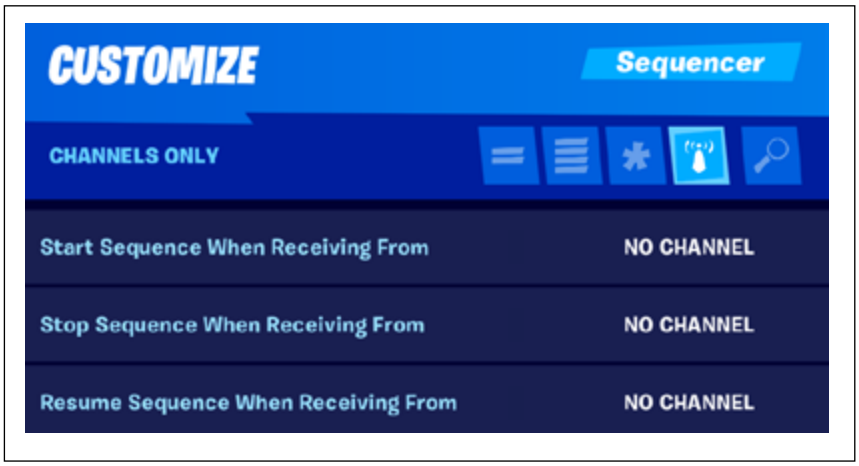
Use the  icon to show all options available for the device.
When you have made these changes, select Start Game and enter the Mutator Zone.

While in the Mutator Zone, you should be able to see that the Damage Volume is active.
If it has been configured correctly, the Damage Volume will disappear when the player starts to emote.

SEQUENCER

The **Sequencer** is a useful device that can interact with other devices in addition to the player. When triggered, the Sequencer will send a pulse from one side of the zone to the opposite side.

Looping	How many times does this repeat when activated?
Tempo (bpm)	How fast does the pulse move through the zone?
Zone Direction	Which side of the base should the zone come from?
Activation Type	When activated, does it start a new pulse or toggle the pulse?
Pulse Direction	How should the pulse move through the zone?
Damage Level	How much damage should the player get if hit by the pulse?

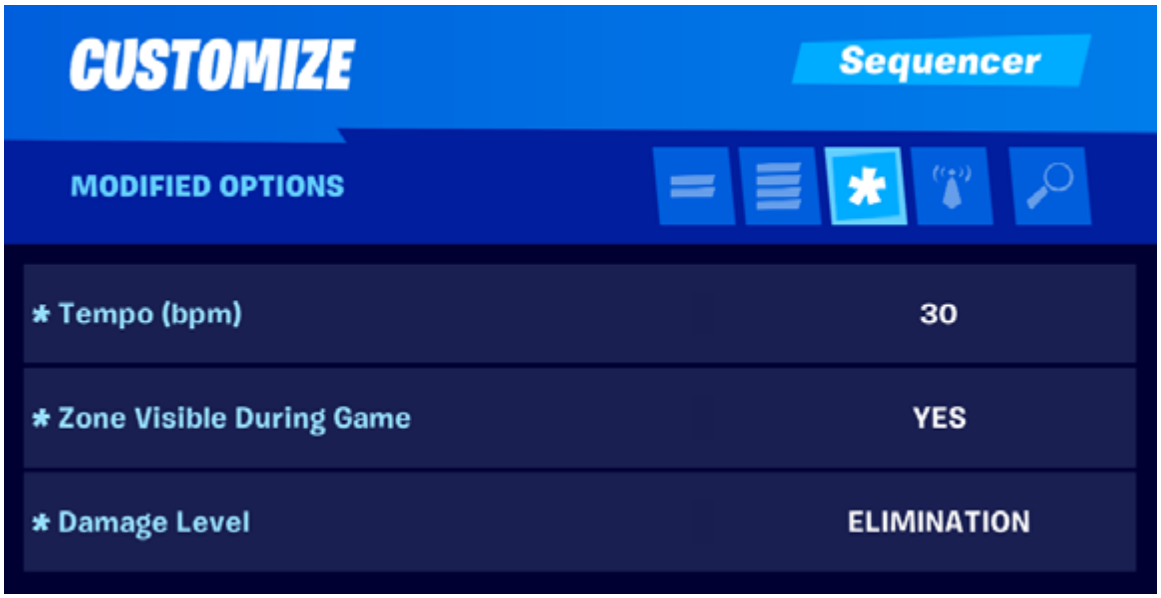


TEST THE SEQUENCER

The Sequencer can be used to activate **Triggers** which you will cover in the next step. Another nice feature is that you can create a pulse that will eliminate or damage the player if there is a collision.

Simulate a slow-moving pulse that will start behind the player. This ensures that the player keeps moving. If the player stops moving or moves too slowly, they will be eliminated and have to start over.

Customize the Sequencer to lower the speed and set the damage level.

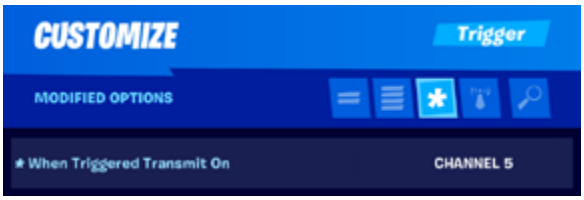
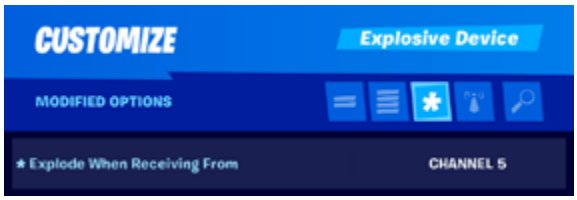


Start the game step on the Sequencer pad, and run around the barrier ahead of the pulse. After you get ahead of the pulse, move into the zone and see if you get eliminated when the pulse reaches your player.

STEP 5: PROTOTYPES–TRIGGERS

You should be starting to see how some devices can detect a collision AND send/receive signals to other devices. Let's quickly review **Triggers**, so you can initiate a signal when the player touches the trigger. You would use this instead of **Zones** to get a smaller and more precise trigger area.

For this final prototype, get a Trigger and an Explosive Device from your creative inventory.

	
Trigger	Explosive Device
	

Set your Trigger in the center of a tile and set the Explosive Device about two tiles away. (Remember to use Grid Snap in your Phone tool Options for easy alignment when placing devices.)

Test your Trigger. If done correctly, the Explosive Device should be activated when your player collides with the trigger.

NOTE: Triggers are Props, so they can be resized, rotated and positioned freely on your island. This gives you creative control on how different events can be triggered.

STEP 6: BUILD YOUR OBSTACLE COURSE

Hopefully, your mind is exploding with ideas for creating your Obstacle Course in Fortnite Creative. It is time to put your understanding of collision detection and event-driven programming to work.

The Challenge

Build an obstacle course with at least five different types of obstacles.

- Create an obstacle that is based solely on building materials. You can use the wood, brick or metal materials, or even grab some elements from the Gallery in your Creative Inventory.
- Create an obstacle that uses traps in your construction.
- Create an obstacle that uses one or more of the different Volumes.
- Create an obstacle that changes in the game by being triggered by a device that transmits a signal upon activation.
- Invent your own type of obstacle.

Test your course to make sure it works and is balanced so it can be completed without being too easy.

Ask someone to play your obstacle course, and ask for feedback. Constructive feedback can help you improve your game.

When giving feedback to your peers, make sure to highlight the positive aspects of the obstacle course and give specific feedback and ideas on how it can be improved. Feedback and sharing of ideas is a valuable skill to practice as a creator.

EXTENSION ACTIVITIES

If you want to take this exercise further, here are some challenges for you to consider.

1. Implement a Sequencer to make sure the player maintains a steady pace. If they go too slowly, the pulse from the Sequence should catch up with them and give them some level of damage.
2. There are more devices that were not covered in this document. Can you find new devices to use as obstacles and triggers?
3. Create a way to detect the end of the game and display the player Time as their score.
4. Add sound effects to your course with the Speaker and various triggers.
5. Research collision detection and understand the challenges of detecting collisions in 2D and 3D games.
6. Research event-driven pProgramming and learn what types of programming languages are popular with these types of programs.

FORTNITE

BUILDING AN OBSTACLE COURSE:

COLLISION DETECTION, TRIGGERS, AND EVENTS IN FORTNITE CREATIVE

STUDENT GUIDE