Rollercoaster Theme Park (Minecraft Education)

Your task

Working in groups of no more than 3, film your in-game robot or “agent” as it completes the **eighth and final challenge** in the Code Builder Tutorial world found in Minecraft: EE. You will also be required to answer the questions seen below, upload and share the video, and download and share the code.

Phase 1

Filming

You may use any resources at your disposal to research and program your robots. You *do not* need to film the robot completing any other challenges in the Tutorial World. You only need to film your robot completing **the eighth and final challenge**. You will need to work on the other obstacles to help you prepare for the eight challenge.

It only needs to be as long as it takes to clearly show your in-game robot or agent completing the eighth and final challenge of the Code Builder Tutorial World. You are free and encouraged to provide any context or explaination you wish in the video. Imagine that you are teaching your teachers or your parents about how the robot works.

Submitting

Naming files and managing them properly is an important digital literacy skill. You should always be on top of being organized in your digital space. As you get older you will start to see just how many files will start to pile up! Always be ahead of the game and label files right away.

You can complete this phase at any time. As soon as it is done, and NEF gives approval, you can begin work on the second and main phase of the contest. To fully complete this phase groups must:

- Submit a short video showing the completion of the eighth and final challenge in the Code Builder Tutorial World.
- Submit answers to the questions seen on page 4.
    - **Name the file: “phase1Questions_schoolname_studentlastnames.docx”**
- Submit your code.
    - **You must name the file as follows: “phase1Code_schoolname_studentlastnames.mkcd”**
    - Follow the short “Exporting your Code” guide below for exporting your code.

World Setup

World setup for the first phase is very easy!

1. Launch Minecraft: EE
2. Select “Play”, then “Create New”
3. Select “Code Builder Tutorial”
4. Set the default game mode to “creative”
5. Click “Create” on the left side of the screen
6. **Once in the game world, connect Code Connection and load MakeCode**
    a. To see how to do this watch this video, or follow this guide.

Exporting your Code

In the MakeCode coding environment you will see a symbol of a green floppy disk at the bottom of the screen.

Click on it and then name your project as directed above. Save it somewhere safe!

You can also import code into MakeCode. From the MakeCode landing page (before you get into the coding environment) there is a button on the top right of the screen that looks like the image on the right.
Select it, then upload your .mkcd file into it. This is a good way to backup your work and share it with your group mates.

Pro Tips

- If you are working in a computer lab, you should be able to click on the “servers” tab from the “Play” menu and join your teammate’s server. You can connect multiple agents to a single server! This means you can work on the course at the same time! You will need to communicate what you learn to your teammates in order to finish the eighth challenge.
- You can create a new fresh code builder world if you need to and load your code into it.

- Experiment with the other built-in tutorials in MakeCode, a lot of them are good fun!
- Use the Hotkey “Alt+Tab” frequently to quickly flip your screen between MakeCode and Minecraft: EE.
- Click the “{ }” symbol seen at the top of the MakeCode environment to see what the code looks like in JavaScript, can you figure out what it says?

Questionnaire

In essay form, answer the following questions as specifically as possible. Use proper spelling and grammar! You may copy and paste the questions into your own document, save the file, then submit it. Follow the submission directions above!

Questions

1. To the best of your ability, describe the logic behind programming the robot to complete the eighth challenge.

2. What programming “tricks” did you discover to make your code more efficient? (Hint: Loops)

3. Describe what problems you overcame as a group. Did you need to do any extra research? Were there any obstacles that caused some issues in particular?

4. Describe who figured out what among your group.

Assessment Criteria

To complete Phase 1, and begin on Phase 2 groups must:

- Complete and submit the required items listed on page 2.
- The video must show the group’s in-game robot or “agent” complete the eighth and final challenge.
    - Students should provide narration.
- Complete and submit the questionnaire

In summary, this first phase is all about getting setup and becoming familiar with the software, the contest, and to figure out student grouping.
In phase one students will complete the in-game “Code Builder Tutorial”. Students will use MakeCode, a visual coding language for JavaScript to program an in-game robot to complete a series of tasks. This will familiarize students with the software, the coding environment, the contest, and they will have completed the famed “hour of code”.

Phase 2

In Phase 2 students will complete the main part of the contest. Their task is to build a theme-park which features mini-games, all the structures you would imagine at a theme park (hotels, ticket booths, food stands, etc), and a cart ride which brings one through the park. They are required to document the building process, utilize a coding element, answer a series of questions, gather footage, and send all of this to the NEF in a specific organized fashion.

The centerpiece of their theme park's will contain no less than three separate mini-games, or in the theme of the build they could be seen as "rides" or "attractions" for the theme park. Mini-games in Minecraft are very popular, using the in-game tools and features of Minecraft: EE and MakeCode there are many creative opportunities to gamify interaction. The final test for students will be to have their teacher's play through the mini-games. Students are to serve as guides, and teach their teachers about the world of Minecraft. This interaction will need to be filmed and sent to the NEF. A cell phone is all that is needed for filming.

Some ideas for mini-games include:

- Using movement to complete some task (climbing puzzles)
- Using the bow to shoot at things a la amusement park events (Shoot moving targets, shoot from moving targets etc)
- Crafting things, possibly timed races
- Puzzle solving using MakeCode
- Redstone machine creation or puzzles
- Making use of the in-game Chemistry tools
- Building races

Your task

Using Minecraft: Education Edition and MakeCode your task is to work in groups of 3 to build a theme park complete with "rides" and "attractions". Like any theme park you must build a number of structures to support the park. This includes:

- Hotels
- Ticket Booths
- Roads / Paths
- Lighting Systems
- Restaurants
- A Cart system that shows off the park
- Panoramas or other "displays" (think Disney)
- Statues
- Aid stations
- Be creative!

All lights should be powered using RedStone or solar power.

A theme park wouldn't be a theme park without some rides and attractions! Amongst all of requirements listed above you are **required to build a minimum of three mini-games.** The mini-games purpose will be to teach a new Minecraft player how to play. The games can be anything from climbing puzzles, to games that involve the use of MakeCode to complete. Check out this link for more ideas.

Here are some other ideas:

- Using movement to complete some task (climbing puzzles)
- Using the bow to shoot at things like real life amusement park attractions (Shoot moving targets, shoot from moving targets etc)
- Crafting things, possibly timed races
- Puzzle solving using MakeCode
- Redstone machine creation
- Making use of the in-game Chemistry tools

As part of this phase of the contest you must use the in-game robot you worked with in the first phase to help you build the park. They are great for building big walls, roads, laying tracks, etc. You will be required to send your MakeCode file just like the first phase. You will also be required to film and document the building process.

Documenting the Process

As you work with your team to build, keep track of who does what, and take pictures using the in-game camera tool. Search for the item "camera" in-game. Periodically during development the NEF will contact all of the participants and require a short video showing how the build has been going. Always be prepared to show your work and be able to explain what you are building!

Filming

Along with periodic "check-ins" using video. All groups are required to record their teacher's going through their mini-games. Groups should see it as an opportunity to teach their teachers about the world of Minecraft. Using a cell phone camera is fine. Students should provide some narration and describe their world and mini-games.

Submitting

Naming files and managing them properly is an important digital literacy skill. You should always be on top of being organized in your digital space. As you get older you will start to see just how many files will start to pile up! Always be ahead of the game and label files right away.

To complete Phase 2 and be eligible for cash prizes you must:

- Submit a short video showing students leading teachers through their theme park and mini-games.
- Submit your Minecraft World file
    - **Name the file: “phase2World_schoolname_studentlastnames.mcworld”**
        - The world should contain the elements listed in the “Task” above.
- Submit answers to the questions seen on page 4.
    - **Name the file: “phase2Questions_schoolname_studentlastnames.docx”**
- Submit your code.
    - **You must name the file as follows: “phase2Code_schoolname_studentlastnames.mkcd”**
- A zip file containing screenshots taken during the building process.
    - **Name the file: “phase2ScreenShots_schoolname_studentlastnames.zip”**
- Submit the video, answers to the questions, the world files, and your code

World Setup

1. Launch Minecraft: EE
2. Select “Play”, then “Create New”
3. Select “Blocks of Grass” (You may use a “New World” if you wish)
4. Set the default game mode to “creative”
5. Click “Create” on the left side of the screen
6. **Once in the game world, connect Code Connection and load MakeCode**
    a. Video | Document

Exporting your Code

In the MakeCode coding environment you will see a symbol of a green floppy disk at the bottom of the screen.

Click on it and then name your project as directed above. Save it somewhere safe!

You can also import code into MakeCode. From the MakeCode landing page (before you get into the coding environment) there is a button on the top right of the screen that looks like this:
Select it, then upload your .mkcd file into it. This is a good way to backup your work and share it with your group mates.

Exporting your World

1. From the “Worlds” menu, select the pencil button next to the name of your world.
2. Scroll to the bottom of the next page and select “Export Word”
3. Name it as described above and save it somewhere safe.

Pro Tips

- Experiment with the other built-in tutorials in MakeCode, a lot of them are good fun!
- Experiment, do research, seek inspiration whenever and wherever possible.
- Use the Hotkey “Alt+Tab” frequently to quickly flip your screen between MakeCode and Minecraft: EE.
- Click the “{ }” symbol seen at the top of the MakeCode environment to see what the code looks like in JavaScript, can you figure out what it says?

Questionnaire

In essay format, answer the following questions. Use proper spelling and grammar! You should also exemplify by discussing specific parts of your build. You can insert screenshots into the document if you wish! You may copy and paste the questions into your own document, save the file, then submit it. Follow the submission directions above.

Questions

1. Describe your theme park. What is the theme? Why did you build things the way you did?

2. Describe your mini-games. Who built what? How did you come up with the idea for the games? What are the rules and how do you win?

3. Describe what challenges you had, and how you overcame them. Use examples.

4. How did you put your agent to work? Be descriptive.

Assessment Criteria

%	0-7%	8-13%	14-19%	20-25%
Technically Sound?	No files submitted, no files run properly, nothing is labelled. No video were submitted.	Files are not labelled, many of the files do not open or run properly. The videos do not work.	The necessary files are submitted, but not labelled. The files run without a problem. The videos are not high quality.	All of the necessary files are submitted, labelled properly, and run fluidly. Videos are of high quality and easy to consume.
Creativity and Originality of build	Blocks chosen do not relate to the intended design. The design is a copy & paste from another design. Mini-games were	Blocks chosen relate in some ways to the intended design. The design copies many elements from another design. Mini-game	Blocks chosen relate well to the intended design. Some Redstone is used. Mini-games are playable.	Blocks chosen relate very well to the intended design. Redstone and switches power lights and open doors. Mini-games are unique and playable.

	not created.	requirements were incomplete		
Design and code process	Any code is “sloppy” and all over the place. Does not use loops. Little to none of the requirements were met	Any code is somewhat organized and uses some loops. More than one of the requirements was not met	Any code is organized and uses loops and functions well. One of the required items was not completed.	Any code is very organized and makes use of Loops very well. Each required item is constructed and lights are powered.
Demonstrates ability to work with a team	Students did not work well together to create or document the build. No evidence of roles or contribution in the questionnaire replies	Students attempted to work well together to create and document the build. Each student’s role and contribution is not very clear in the questionnaire replies.	Students worked well together to create and document the build. Each student’s role and contribution is made clear in the questionnaire replies.	Students worked very well together to create and document the build. Each student’s role and contribution is made very clear in the questionnaire replies.

- Is the build **Technically Sound?**

If something is technically sound essentially all the moving parts work as intended. In this case, the files open and work fluidly from machine to machine. The Agent assists as part of the build without a hitch. Any redstone circuits are complete circuits completing specific functions. Any moving parts in the mini-games flow smoothly. And the videos are of high quality. Moving files around and having the code execute on different machines is a normal everyday task to any sort of engineer or programmer which requires significant digital literacy skills as well as organizational skills. Students and staff should ensure files are labelled in an efficient way.

- Does the build **demonstrate creativity and originality**?

The object that is built makes effective use of the large number of options Minecraft provides. Blocks are chosen for a reason, they are not just randomly chosen. This demonstrates critical thinking and problem solving skills to create an aesthetic that is recognizable by all. Students will need to experiment with different types of blocks, as well as work iteratively to achieve the desired effect. **Game design** requires a deep understanding of human interaction and learning. Minecraft offers many opportunities to create an experience. A mini-game can serve as an

educational experience, and the creation of one is an exercise in critical thinking, iterative design, and creativity.

- Is there **evidence of systematic design and efficient use of code**?

Using the Agent to create something will require extensive planning, iterative design, and problem solving skills in both the MakeCode coding environment as well as in the Minecraft environment. Being able to function between several platforms is a critical skill for programmers and engineers in the field. Coders should strive to be as efficient as possible and use as few lines of code as they can to execute tasks. Less code means smaller file size, which makes a computer function more efficiently. Also, the more organized the code of a program or website is, the easier it is for a team to work together on a project. Further, all of the required design elements are present.

- **Demonstrates ability to work with a team.**

Communication skills are one of the most important skill sets an engineer or programmer needs. Often times these sort of professionals work in teams to think critically and problem solve. Competitors will need to set roles, and goals, for each member of the team. The written response to the worksheet questions should include descriptions of what each team member contributed.