CANADA LEARNING CODE

Self-Driving Cars

By: Steve Blair + Kassandra Lenters
Duration: 120 minutes

LEVEL	SUBJECTS	PROVINCES / TERRITORIES	TOOL
Grades 4-6	Science and Technology, Social studies, Applied Design, Skills, and Technologies	Across Canada	Scratch

Overview

Learn about the future of transportation and build a game in Scratch that simulates how self-driving cars work.

Prep Work

- Watch the "Content 101: Scratch" video: http://bit.ly/content101-videos
- Review the **completed version** of the project: **http://bit.ly/self-driving-example**
- Print the Solution Sheet for the main activity: **bit.ly/self-driving-solution**
- This lesson requires computers and access to the internet - or the offline Scratch editor and starter projects downloaded onto computers ahead of time.

Lesson

Key Coding Concepts

- Algorithms
- Sequence
- Debugging
- Events
- Loops
- Conditional statements

Terminology

Radar
a radio device for detecting the position of things in the distance and the direction of moving objects

LiDAR

Created in partnership with

Introduction

Ask: How do cars work? (Who is able to drive cars? What do they need to know before they can start driving? etc..)

Ask: What if a car could drive by itself? Companies like General Motors have actually created something called “self-driving cars” and are testing and training them today. Have you heard of this?

We are learning about self-driving cars, and creating a game in Scratch to help us understand how they work.

Complete a KWL chart using anchor chart paper and markers.

- Ask: What do we *think* we **know** about self-driving cars? (Record answers under “K” for Know)
- Ask: What do we **want to know** about self-driving cars? (Record answers under “W” for Want to Know)
- Say: We are going to learn how self-driving cars work by watching a video and building a game in Scratch! Then we will come back to our chart and answer the “L” (Learned) section.

Part 1: Video

Watch this video about how self-driving cars work: **https://youtu.be/gEy91PGGLR0**

Similar to radar but uses lasers instead of radio waves.

Curricular Connections

Forces causing movement, Energy, Machines, Conservation of energy and resources, Sustainability and stewardship, Electricity and electrical devices, Interactions in the environment, People and environments, Pollution, Safety, Canadian issues and governance

References

General Motors https://www.gm.com/mol/selfdriving.html

“Autonomous car / self-driving car - How it works! (Animation)” video By Thomas Schwenke from Youtube https://youtu.be/gEy91PGGLR0

ScratchEd Rubric http://scratched.gse.harvard.edu/ct/files/Student_Assessment_Rubric.pdf

Discover more at the **General Motors website: gm.com/mol/selfdriving.html**

Part 2: Scratch

Code Along

- Open up a new Scratch project at scratch.mit.edu and click on “create” (top, left corner).
- Point out the main elements: Stage, Sprites, and Scripts. Demonstrate how to drag and connect blocks.
- Give learners a few minutes to click on blocks and explore.
- Go through 1-2 challenges with the group, where learners are tasked with trying to make something happen in Scratch. For example, “Try to make Scratch move” or “Try to make Scratch say something when the space key is pressed” (See the Code-Along Challenges doc (http://bit.ly/scratch-challenge-solutions-doc) for more examples and solutions)

Activity

We are building a game in Scratch to show how self-driving cars work.

Show the **example project** so learners know what they are working towards. Ask them what they see/hear - what is happening in this project?

Open the **starter project** (**http://bit.ly/self-driving-starter**) and review the Sprites and backgrounds. Have learners open the starter project on their screens and click "REMIX."

Use the **Solution Sheet** to guide learners through the following steps:

- Make the car move
- Add a radar
- Use radar to sense the road
- Choose a starting point
- Test and debug
- ... and any additional Add-Ons, if time

Closure

Close or put away computers and return to the KWL chart. Complete the “L” (Learned) section. Ask: What did we learn from the video we watched about how self-driving cars work? What did we learn from building our game in Scratch?

If there are still unanswered questions from the “W” (Want to know) section of your chart, you should be able to find most of the information from General Motors’ 2018 Self-Driving Safety Report.

Assessment

Learning Outcomes (Scratch)
I can give the computer instructions to tell it what to do
I can use conditional statements to control what happens in my project
I can use loops to make things happen more than once
I can use events to control when things happen
I can keep testing and debugging to make my project better

Use the following ScratchEd rubric to assess the learner’s ability to communicate their design process (see “Experimenting and Iterating” and “Testing and Debugging”):
http://scratched.gse.harvard.edu/ct/files/Student_Assessment_Rubric.pdf

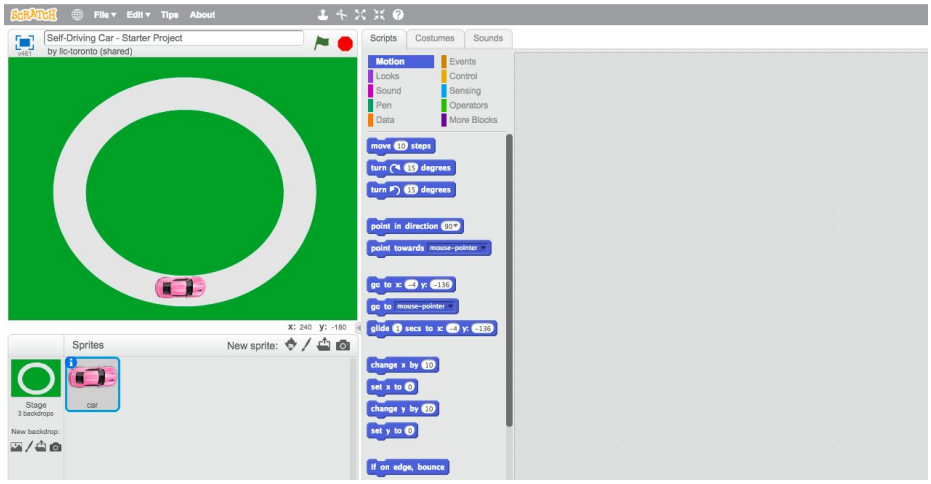
Extensions

- Have learners write about what a future with self-driving cars could look like (e.g. How would it affect the environment?). Refer to General Motor’s “imagine” statements in their 2018 Self-Driving Safety Report (above).
- Brainstorm possible scenarios that a self-driving car may face, then have learners add the scenarios and solutions to their project in Scratch (e.g. an animal crossing the road).
- Machine learning is an important part of training self-driving cars to know what to do in different scenarios. Spend time learning more about Machine Learning through Google’s Teachable Machine (https://teachablemachine.withgoogle.com/)
- Have learners build a self-driving car prototype (using lego, play-doh, pipe cleaners, etc.). Have them label the sensors or any other important features, and share a Design Statement - including which design decisions they made, and why.

Self-Driving Cars

STEP 1: Open the Starter Project

1. Sign in to Scratch
2. Open the starter project: **bit.ly/self-driving-starter**
3. Remix the project & change the project name



STEP 2: Make the car move

1. Make the car move (change the number of steps to a smaller number as we test)
2. Make this happen forever
3. Make this start when the green flag is clicked

STEP 3: Add a radar

Q: How do self-driving cars know what is around them? A: Radar (also Lidar or Camera)

Let's add a radar:

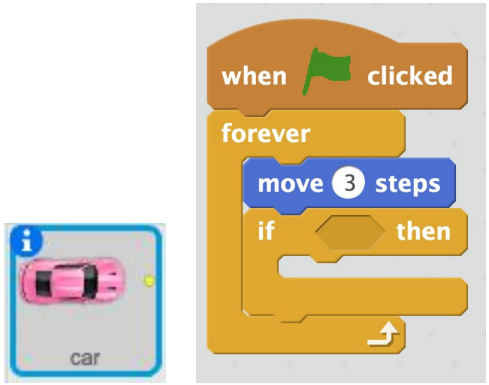
1. Select the "Costumes" tab

2. Choose a car
3. Select the paintbrush tool
4. Choose a colour (Choose the same colour as a class - e.g. Yellow)
5. Increase the brush size for a larger dot (drag the toggle to the right until you are happy with the size)
6. Click in front of the car to draw the radar

STEP 4: Use radar to sense the road

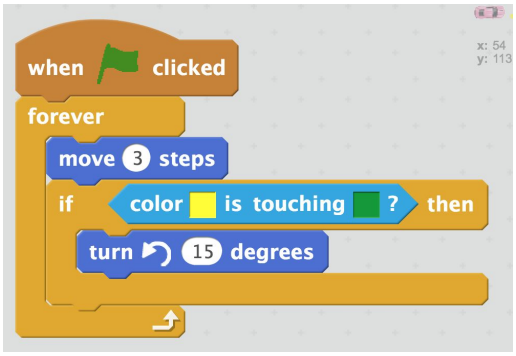
Q: How can we tell the car to stay off the grass? (A: Tell it to turn)

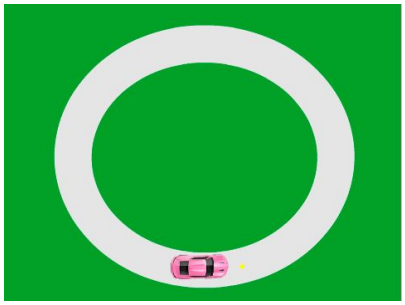
1. Add an If condition (for checking if the radar is touching the grass)



Q: How does the car know what is road and what is grass? We need to teach it what the grass looks like.

2. Use colour sensing to tell the car what the grass looks like: Sense if the radar (yellow) is touching the grass (green) → Click on the colour square then select the colour in your project to set the colours
3. Make the car turns if this is true
4. Test it out!





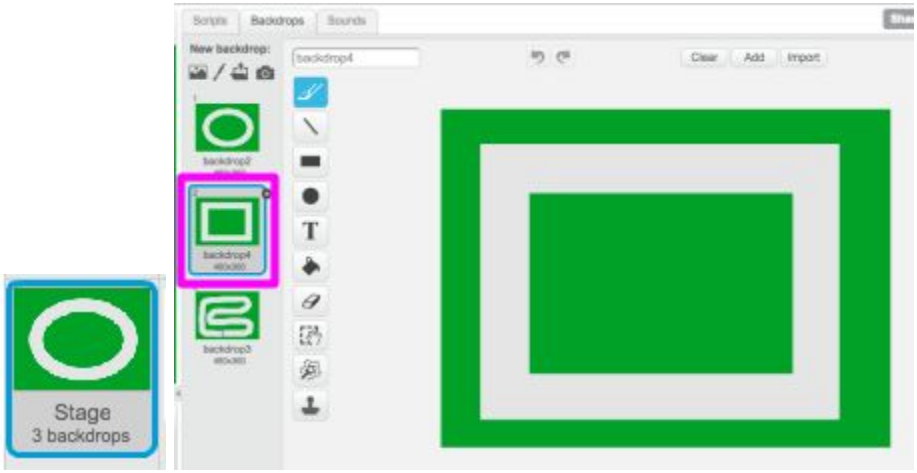
STEP 5: Choose a starting point

1. Click and drag the car to where you want it to start
2. Grab the "Go to X, Y" block with the updated position
3. Make the car start at this position when the green flag is clicked
4. Make the car start facing straight (90 degrees)

STEP 6: Test and debug

Q: What is a "bug" in coding? What does 'debugging' mean? (A: A bug is a problem, or something that doesn't work in our program. Debugging is finding the bug and fixing it)

1. Have learners test out and 'debug' their programs
2. Once they have a working example for the first track (backdrop), test it out on the next track
3. Find track 2 by selecting the "Stage" then "backdrop2"
4. Click on the car, then the "Scripts" tab to get back to your code
5. ***Challenge for older learners:*** *Test it out with backdrop 3 (*This is where most issues will arise!)*



BUG 1) The car won't turn:

Q: Are we checking to see if the correct colours are touching?
Try clicking on the colour square in the sensing block, then selecting the exact colour match of the sensing dot(s) in front of the car

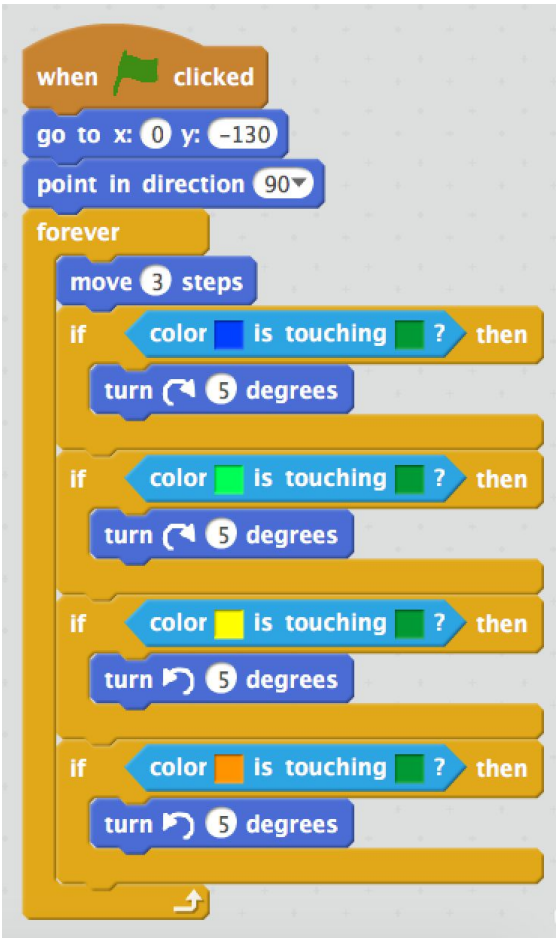
BUG 2) The car is driving along the edge of the road OR car gets stuck:

Q: How far ahead should the car be looking?
Try erasing the dot and drawing it further in front of the car

BUG 3) The car starts turning around, driving the wrong way on the road, moving in circles

Q: How many radars did the car have in our video?
Try adding more radars around the car - **Note: Use a new colour for each radar**
Still an issue? Try moving the dots further away or closer to the car. Keep testing!

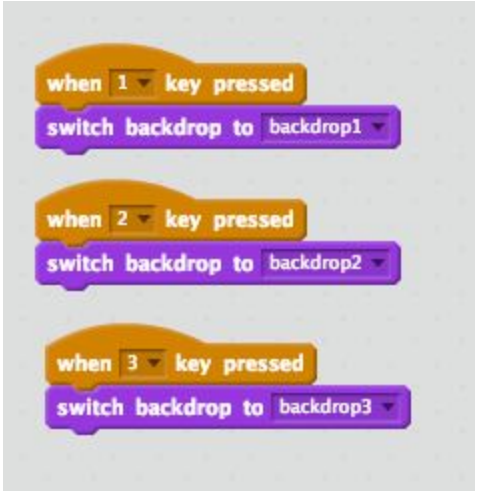
Here's an example solution with 4 radars (note that the original yellow dot that we placed directly in front of the car has been moved to be off-centre to help sense only one side of the grass):



ADD-ON: Quickly switch backdrops

1. Make the backdrop change when a key is pressed
2. Do the same for all of your backdrops, with a different key for each

Example:



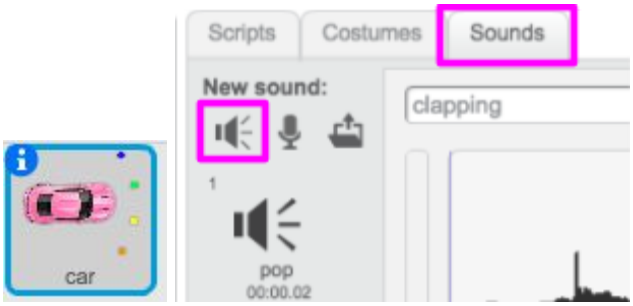
ADD-ON: Add a finish line

*This would simulate Lidar (laser sensors)

1. Draw a finish line on the backdrop using a new colour
2. Make the car stop when the front radar sensor touches the finish line (try using colour sensing again)
3. Add these instructions AFTER the if/else conditional

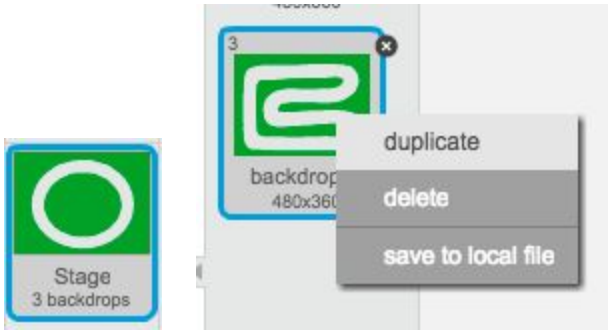
ADD-ON: Add sounds

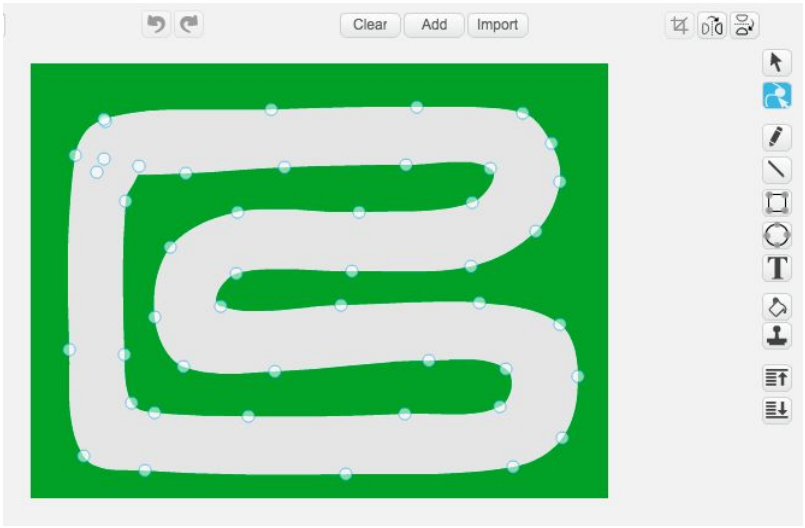
1. Select a sound under the “Sounds” tab - Or record your own!
2. Under “Scripts” use the “play sound until done” block to make the car play a sound (if you use ‘play sound’ it might cut the sound off - “until done” helps prevent this)



ADD-ON: Draw more tracks

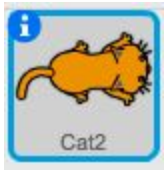
1. Select the stage > Backdrops
2. Duplicate a backdrop
3. Use the reshape tool: Click on the track for the edges to appear, then click and drag the little dots to reshape the track.

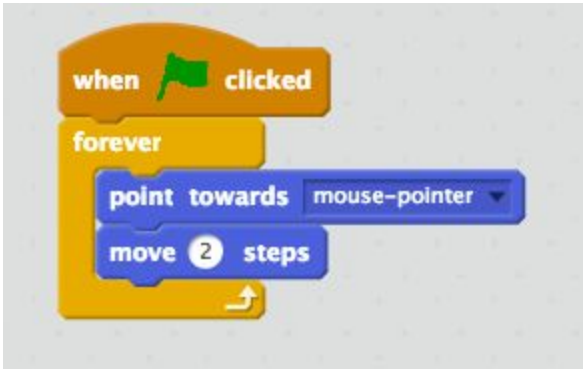




ADD-ON: Moving obstacles

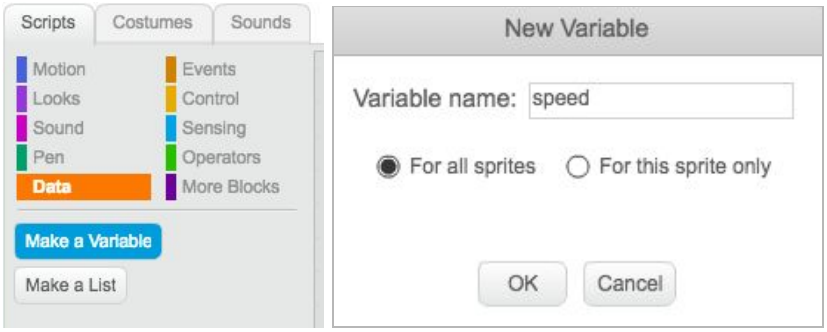
1. Add a new sprite (from the library)
2. Make it move (try using loops, or making it follow your mouse pointer)
3. Make this happen when the green flag is clicked
4. Make the car wait until the sprite is not touching the sensors before driving - or make something happen when they do touch





ADD-ON: Increase the speed

1. Make a variable called “speed” ← This will remember the speed value (number)



2. Replace the value for move with “speed” - so we can change how fast it moves
3. Make the speed increase when the up arrow key is pressed
4. Make the speed decrease when the down arrow key is pressed
5. Reset the speed after the green flag is clicked

Q: How fast can you go before noticing more “bugs”?