

Driving School - Dash

Summary

Description

In this first puzzle, your students will learn how to program Dash over a series of 12-driving lesson challenges.

Learning Procedure

By the end of this puzzle, students will understand that a program is an **algorithm** composed of a **series of procedures**. Using drag and drop blocks, students will learn how to program Dash to move **forward** and **backward**, **turn left** and **right**, **look left**, **look right**, **look forward**. The challenges require students to **add**, **connect** and **delete blocks from a program**. Students will be introduced to **pre-programmed sounds**, i.e., car engine sound and say Hi, colored **lights**, as well as, editing **eye pattern** blocks to simulate an expression, i.e., a smile. By the end of this puzzle, your students should feel confident to take Dash out for a spin around the Wonder Workshop neighborhood or, at the very least, the hallways of your school.

Concepts Covered

- **Start** - students will recognize the **Start** button in the right-hand corner of the screen as the way to start a program.
- **When Start** - students will learn how to use a **When Start** block to begin a program.
- **Sound** - students will learn how to use pre-programmed **Sounds**, i.e., **car engine** and **Say Hi**, in a program.
- **Drive**
 - students will learn how to use command blocks in drive that move Dash's two wheels **forward** and **backward**.
 - students will learn how to program Dash to move **forward** and **backward**.
 - students will learn how to program **left** and **right turns**.
- **Look**
 - students will recognize the different ways they can program Dash's head to move i.e., **up**, **down**, **forward**, **left** and **right**.
 - students will learn how to edit Dash's head movement.
- **Light**
 - students will learn to edit the **All Lights** block to control Dash's ears and chest color.
 - students will learn to edit the **Eye Pattern** block, the 12 LED eye lights.

In App

Vocabulary:

Start: the beginning of or to begin the program that was created

When Start: executes a command when the Start block begins

Drive Controls: the ability to maneuver Dash around using a directional pad

Look Controls: the ability to have Dash look around using a directional pad

Light Controls: the ability to manage the lights on Dash and Dot

Reflection Questions:

1. Which block is used to begin a program?
2. What types of pre-programmed sounds are available in the **Sound menu**?
3. Identify the different directions in which Dash can be programmed to move.
4. Identify the 5 different directions that you can program Dash's head to move.
5. What is the difference between **All Lights** and **Eye Pattern** blocks? Hint: Which lights does each control? What color are lights in each? What else is different about these two light sources?

Extension Activities

1. Bus Driver Dash

Have students create an imaginary layout of the Wonder Workshop neighborhood square on an oversized sheet of bulletin board paper. Direct students to include streets that run horizontally and vertically with a fire station, gas station, restaurant and any other fun places they can imagine. Then have students program Dash to travel from one destination to another. Have students measure this distance in centimeters and then time Dash's trip. Based on this information, have students create word problems which involve Bus Driver Dash's neighborhood, Example: The distance from the restaurant to the movie theater is 180 cm. It took Bus Driver Dash 2 minutes to get there. Measure the distance from the fire station back to Dash's home. Estimate how long it will take for Dash to make this trip.

*Determine whether you want to keep the speed constant or have the students adjust Dash's speed in the accelerometer located in drive blocks. **To increase the difficulty level for 5th grade require students to convert centimeters to meters.

Standards: 1.MD.A.2; 2.MD.A.1; 4.MD.A.2; 5.MD.A.1

2. Dash's Driver's Manual

Now that students have received a license to program Dash, have them teach other students. Instruct students to create Dash's Driver's Manual, which includes all of the programming blocks they learned in the Driving School Puzzle. Require students to include funny hand-drawn illustrations or free online images. This manual may be created by hand using markers and paper or digitally using a presentation program like Keynote (for Mac) or Google Slides. Teach students how to insert sound effects for a multimedia Driver's Manual. The end of the presentation should include a driver's test!

Standards: W.1.2; W.2.2; W.3.2; W.3.4.2; W.5.2

3. Parking Puzzler

Create a paper grid with enough parking spaces for each Dash in your classroom to park. Label the columns A, B, C, D, etc. and the rows 1, 2, 3, 4, etc. Create as many squares as the number of Dash robots your students will be using. Using the Wonder Workshop neighborhood students created in the first activity (Bus Driver Dash), have students design a program which begins at a destination in the Wonder Workshop neighborhood and ends in an assigned parking space. Dash must follow the rules of the road. That means looking both ways when crossing at an intersection. Dash must not crash into another parked car while parking. To increase the level of difficulty require Dash back into a parking space.

Standards: CCSS MP1; CCSS MP4

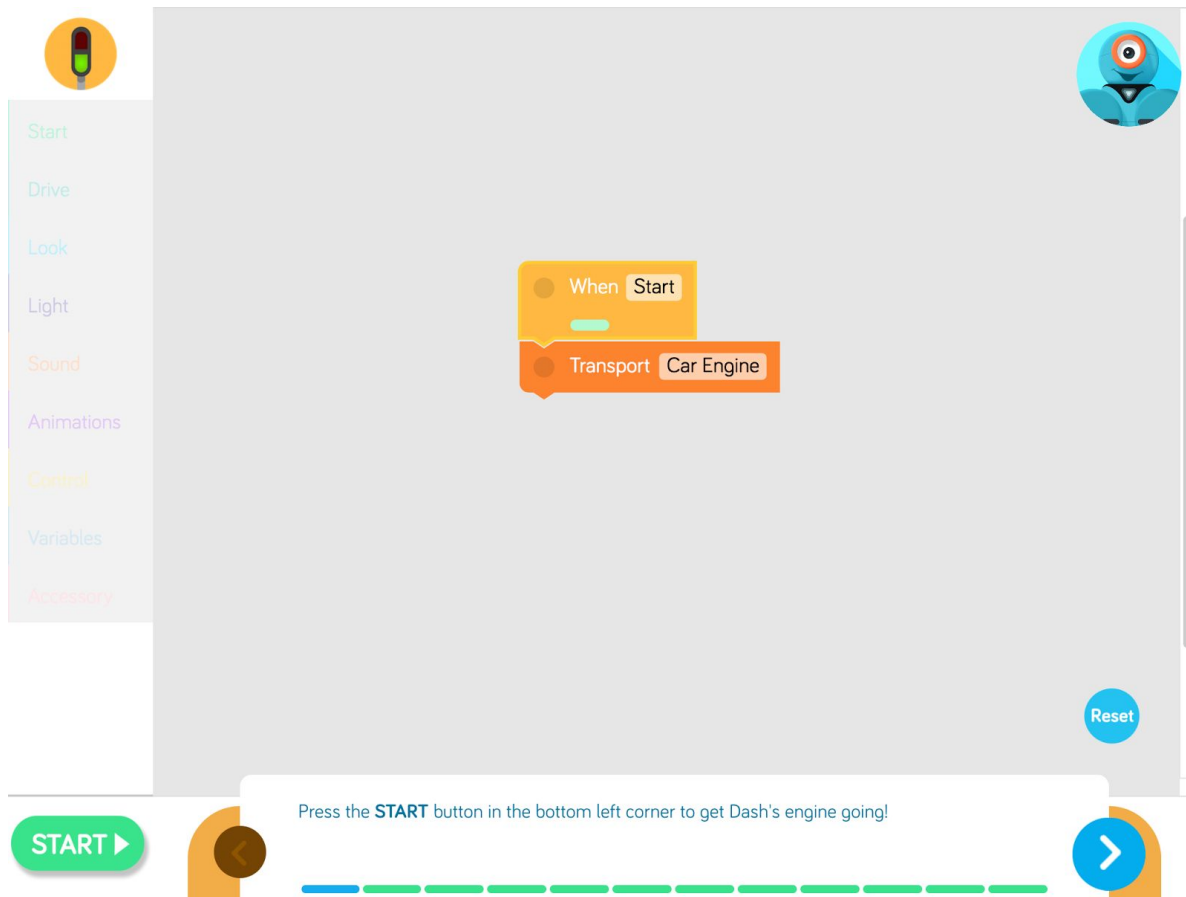
4. Fireman Dash *You must have Dash's Launcher Accessory for this activity. Upfront time should be spent modeling how to use the launcher in the "Create New" section of the Blockly App. Students should practice using the launcher's directionality and power percentage before attempting this activity.

In this activity, students will work in small groups to program Dash to get to a fire located on a 9 x 9 grid. You must determine a coordinate for Dash to begin, e.g., square A8. Then Dash must drive to the fire, another designated coordinate on the same grid, e.g., square E4. Place a target cup on this square. Dash must "put out the fire" by launching balls into the target cup. The balls will be used as water to put out the fire. The team that can program Dash to get to the fire and launch the most balls/water into the target cup will be declared the winners.

Standards: CCSS MP1; CCSS MP4

Challenge 1

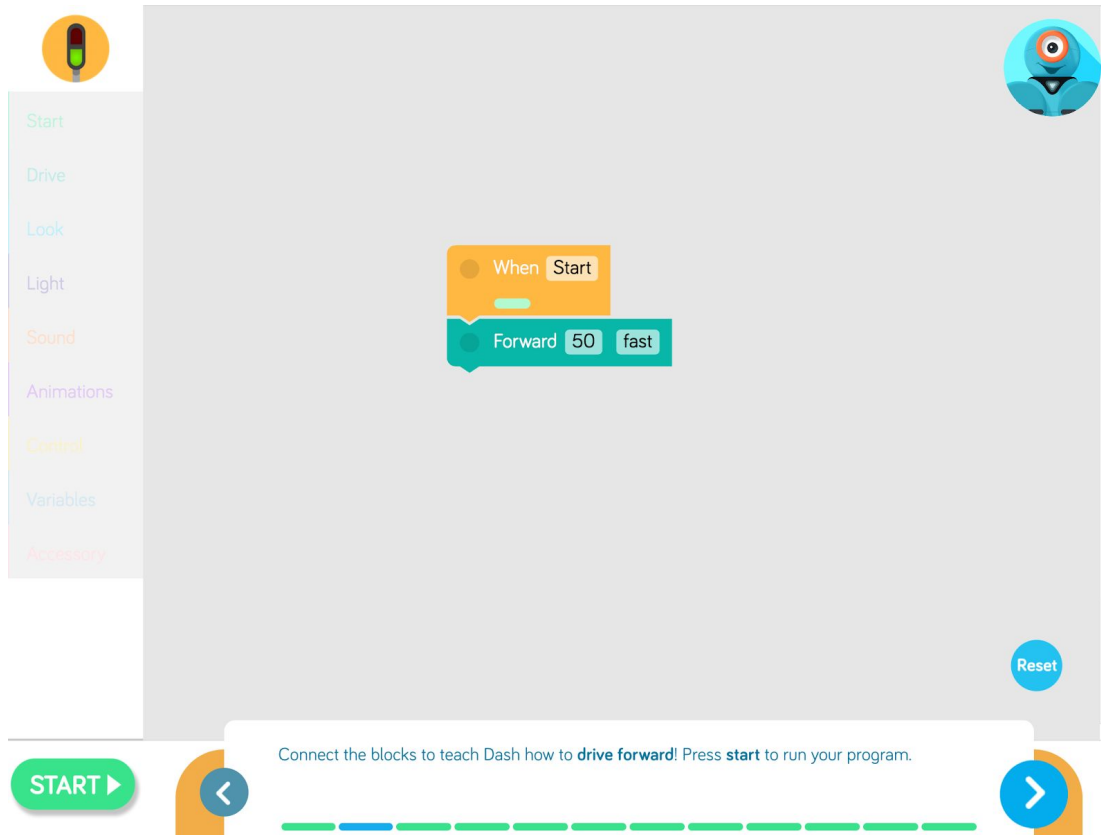
Press the **START** button in the bottom left corner to get Dash's engine going!



The image shows a programming interface for a car simulation. On the left is a vertical menu with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessory. The main workspace contains a script with two blocks: a yellow 'When Start' block and an orange 'Transport Car Engine' block. A 'Reset' button is in the bottom right of the workspace. At the bottom of the screen, there is a 'START' button on the left, a progress bar with 10 segments (the first is blue, the rest are green), and a right arrow button on the right. A text box above the progress bar says: 'Press the **START** button in the bottom left corner to get Dash's engine going!'.

Challenge 2

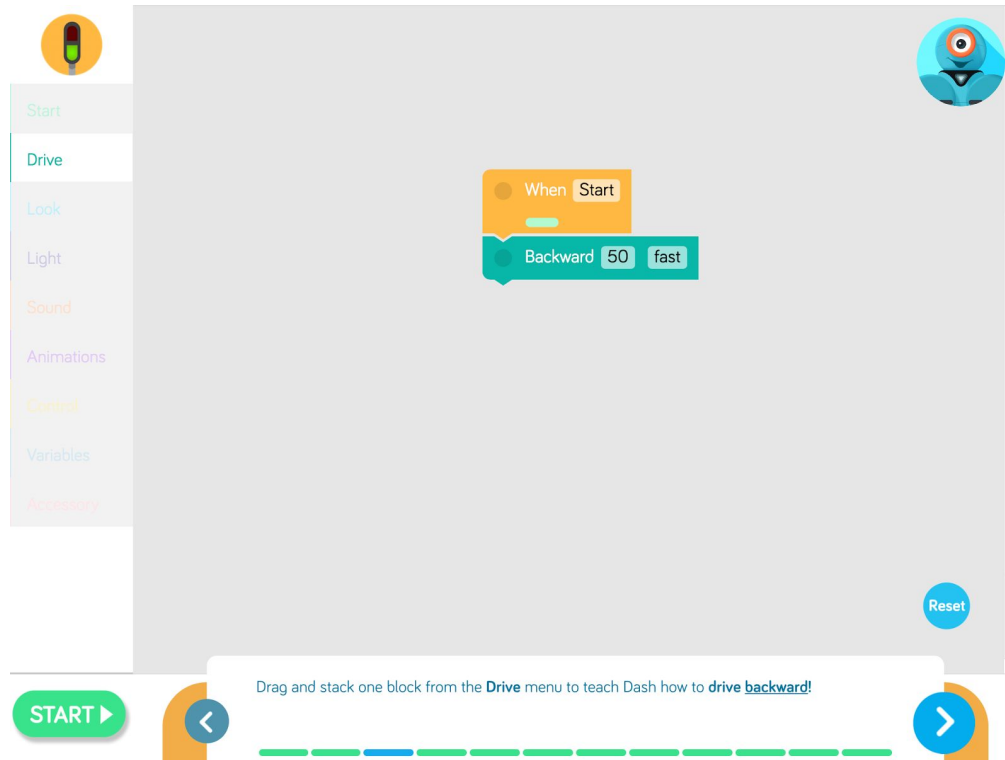
Connect the blocks to teach Dash how to **drive forward!** Press **start** to run your program.



The image shows a programming interface for the Dash robot. On the left is a vertical menu with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessory. The main workspace contains two blocks: an orange 'When Start' block and a teal 'Forward 50 fast' block. A 'Reset' button is in the bottom right of the workspace. At the bottom of the screen, there is a 'START' button with a right arrow, a left arrow button, a horizontal dashed line, and a right arrow button.

Challenge 3

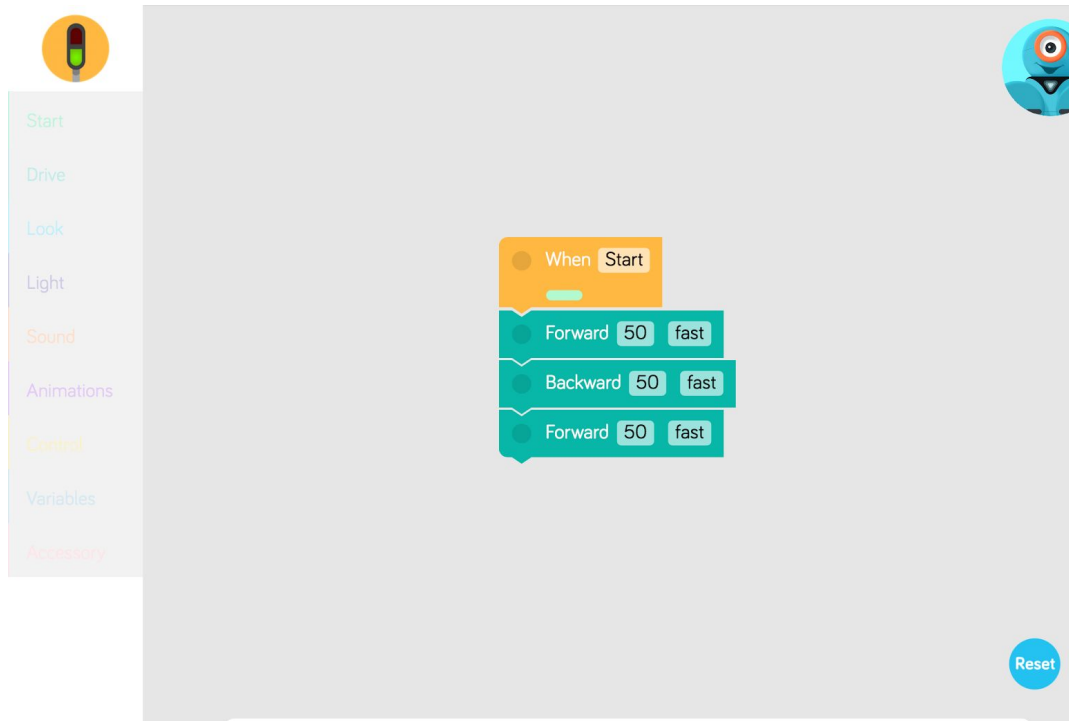
Drag and stack one block from the **Drive** menu to teach Dash how to drive **backward**!



The image shows a programming interface for the Dash robot. On the left is a vertical menu with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessories. The 'Drive' category is selected. The main workspace contains a script starting with a 'When Start' block followed by a 'Backward 50 fast' block. A 'Reset' button is in the bottom right of the workspace. At the bottom of the screen, there is a 'START ►' button, a left arrow button, a progress bar with 10 segments (the second is blue), and a right arrow button. A text box above the progress bar contains the instruction: 'Drag and stack one block from the **Drive** menu to teach Dash how to drive backward!'.

Challenge 4

Unscramble the blocks to teach Dash how to **drive forward, backward** and then **forward** again.



The image shows a programming interface for the Dash robot. On the left is a vertical menu with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessory. The main workspace contains a script starting with a 'When Start' block, followed by three 'Drive' blocks: 'Forward 50 fast', 'Backward 50 fast', and 'Forward 50 fast'. A 'Reset' button is located in the bottom right corner.

```
When Start
  Forward 50 fast
  Backward 50 fast
  Forward 50 fast
```

Challenge 5

Delete the **turn left** block so that Dash can **turn right** twice!

The interface shows a menu on the left with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessories. The 'Drive' category is selected. The main workspace contains a script starting with a 'When Start' block, followed by two 'Turn Right 90' blocks. A 'Turn Left 90' block is positioned at the bottom right of the workspace, with a 'Reset' button next to it. At the bottom, there is a 'START' button, a progress bar with a blue segment, and a navigation bar with left and right arrows. A text box at the bottom reads: 'Delete the **turn left** block so Dash can **turn right** twice!'.

The interface is identical to the one above, but the 'Turn Left 90' block is no longer present in the workspace. The 'Reset' button is now located at the bottom right of the workspace area. The progress bar and navigation bar at the bottom remain the same, with the text box at the bottom reading: 'Delete the **turn left** block so Dash can **turn right** twice!'.

Challenge 6

Delete the **turn left block** so Dash can **turn right four** times.

The Scratch script editor shows a sequence of blocks: a yellow 'When Start' block, followed by three teal 'Turn Right 90' blocks, and finally a teal 'Turn Left 90' block with a blue 'Reset' button. The 'Drive' category is selected in the left sidebar. Below the editor is a 'START' button and a progress bar with a blue segment.

START ▶

◀ _____ ▶

Delete the **turn left block** so Dash can **turn right four** times.

The Scratch script editor shows a sequence of blocks: a yellow 'When Start' block, followed by four teal 'Turn Right 90' blocks. The 'Drive' category is selected in the left sidebar. Below the editor is a 'START' button and a progress bar with a blue segment.

START ▶

◀ _____ ▶

Delete the **turn left block** so Dash can **turn right four** times.

Challenge 7

Teach Dash to **look left**, **look right**, then **look forward**.

A screenshot of the Scratch 'Look' menu. The 'Look' category is selected. The 'Look left 90' block is highlighted. Other visible blocks include 'Look up 22' and 'Look towards Voice'. A 'START' button is at the bottom left, and a progress bar is at the bottom.

A screenshot of the Scratch 'Look right 90' block. A magnifying glass is over the block, and a small robot icon is shown with 'right' and 'left' labels. A 'START' button is at the bottom left, and a progress bar is at the bottom.

A screenshot of the Scratch 'Look forward 0' block. A magnifying glass is over the block, and a small robot icon is shown with 'right' and 'left' labels. A 'START' button is at the bottom left, and a progress bar is at the bottom.

A screenshot of the Scratch script area. A 'When Start' block is followed by three 'Look' blocks: 'Look left 90', 'Look right 90', and 'Look forward 0'. A 'START' button is at the bottom left, and a progress bar is at the bottom.

Challenge 8

Unscramble the blocks to teach Dash to **drive forward**, then **look left**, and then **turn left**. Finally make Dash **look forward**.

START ▶

◀

Unscramble the blocks to teach Dash to **drive forward**, then **look left**, and then **turn left**. Finally make him **look forward**.

▶

Challenge 9

Unscramble the blocks to make Dash turn **all lights red**, then turn **all lights green**.

START ▶

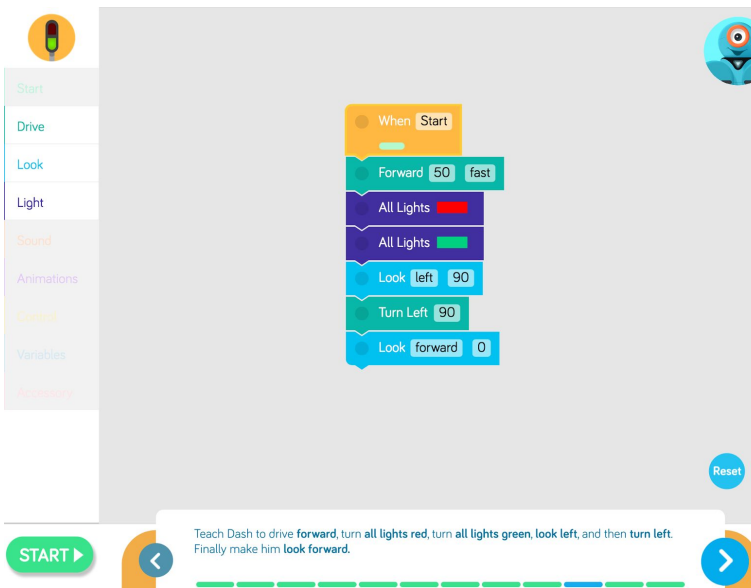
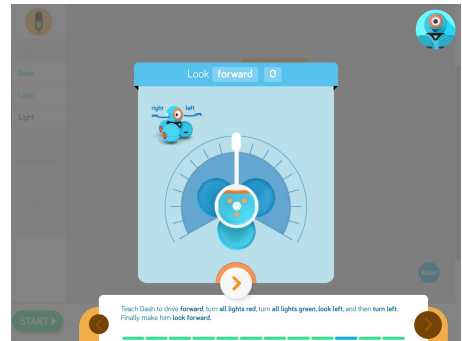
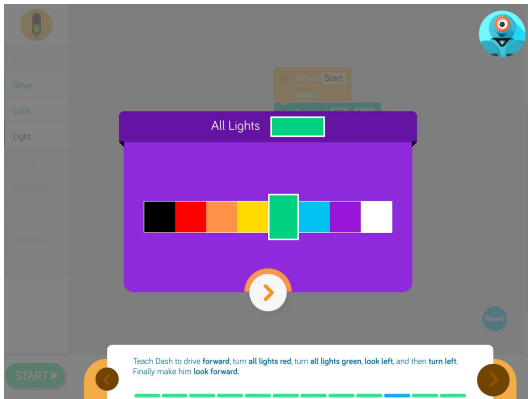
◀

Unscramble the blocks to make Dash turn **all lights red**, then turn **all lights green**.

▶

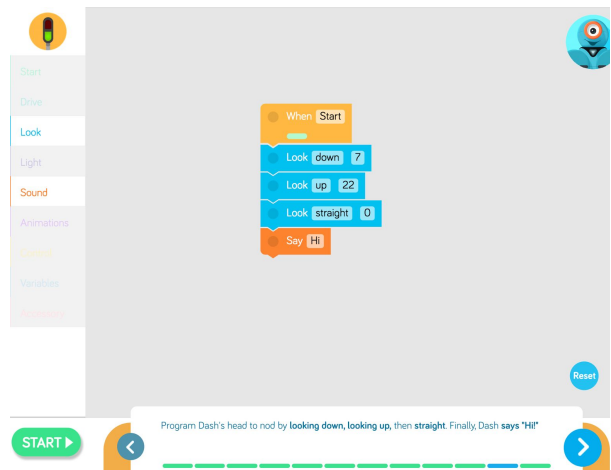
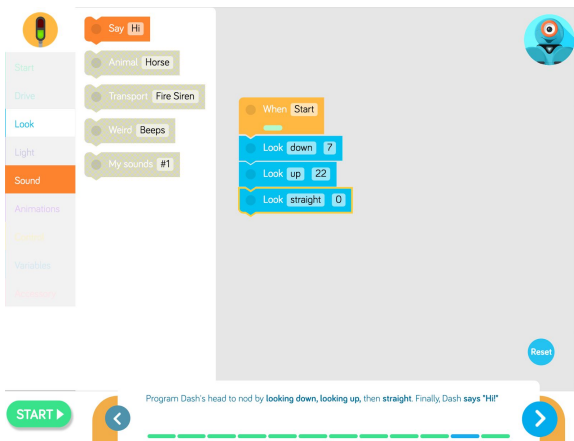
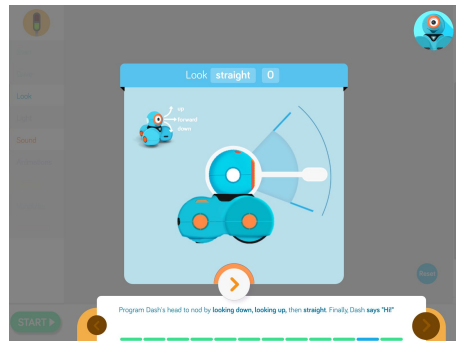
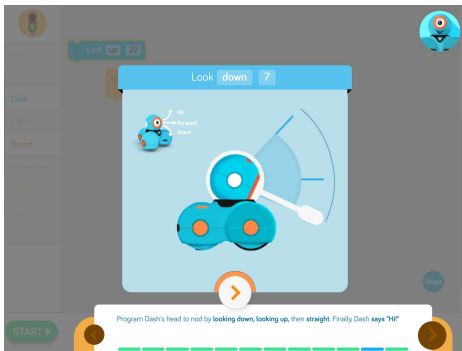
Challenge 10

Teach Dash to drive **forward**, turn **all lights red**, turn **all lights green**, **look left** and then **turn left**. Finally make Dot **look forward**.



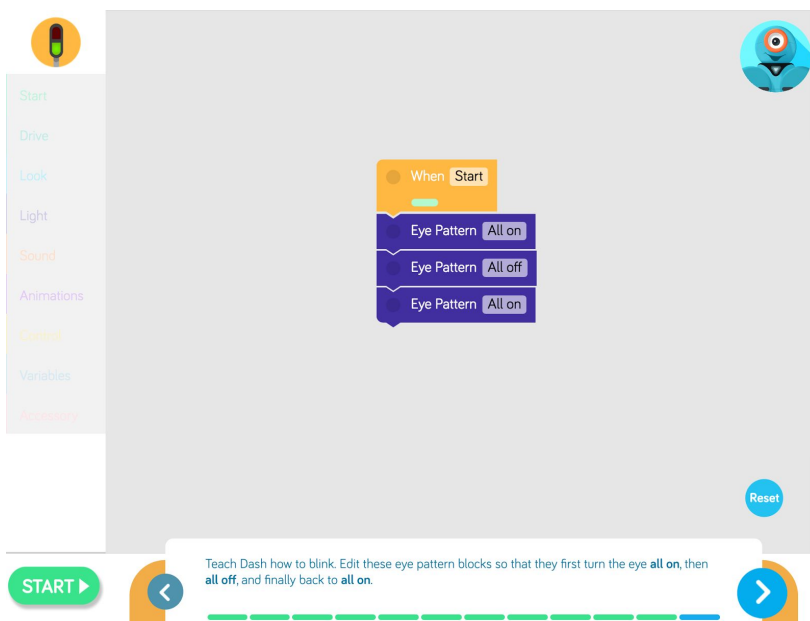
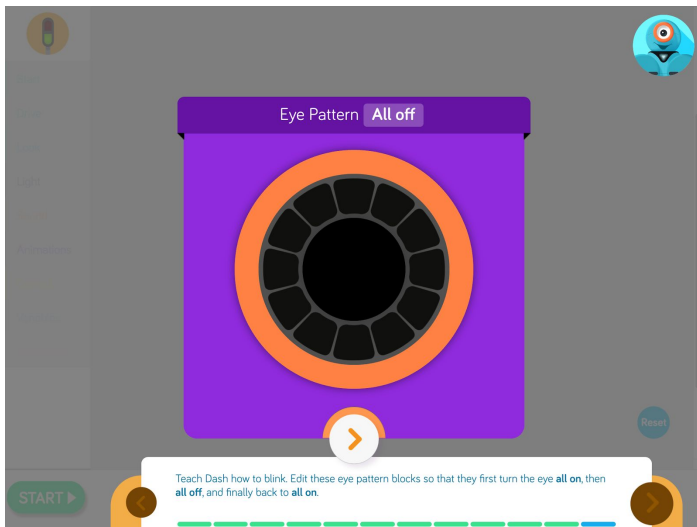
Challenge 11

Program Dash's head to nod by **looking down**, **looking up**, then **straight**. Finally, Dash **says "Hi!"**



Challenge 12

Teach Dash how to blink. Edit these eye pattern blocks so that they first turn the eye **all on**, then **all off**, and finally back to **all one**.



Educational Standards

CC Mathematical Practices:

1, 2, 4, 5, 6, 7, 8

CC Math Standards *Relates to Extension Activity: *Bus Driver Dash*

1.MD.A.2; 2.MD.A.1; 4.MD.A.2; 5.MD.A.1

CC Language Arts Standards *Relates to Extension Activity: *Dash's Driver's Manual*

W.1.2; W.2.2; W.3.2; W.3.4.2; W.5.2

CSTA K-12 Computer Science Standards

- CT.L1:3-03. Understand how to arrange information into useful order
- CPP.L1:3-04 - Construct a set of statements to be acted out to accomplish a simple task.
- CT.L1:6-01. Understand and use the basic steps in algorithmic problem-solving.
- CT.L1:6-02. Develop a simple understanding of an algorithm
- CPP.L1.3-04. Construct a set of statements to be acted out to accomplish a simple task.
- CPP.L1:6-05. Construct a program as a set of step-by-step instructions to be acted out.
- CT.L2-03. Define an algorithm as a sequence of instructions that can be processed by a computer.
- CT.L2-06. Describe and analyze a sequence of instructions being followed.
- CT.L2-07 - Represent data in a variety of ways: text, sounds, pictures, numbers.
- CT.L2-14. Examine connections between elements of mathematics and computer science including binary numbers, logic, sets, and functions.

Next Generation Science Standards NGSS

- 3-5-ETS1-2. Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.