

# 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.

## **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!

## **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.

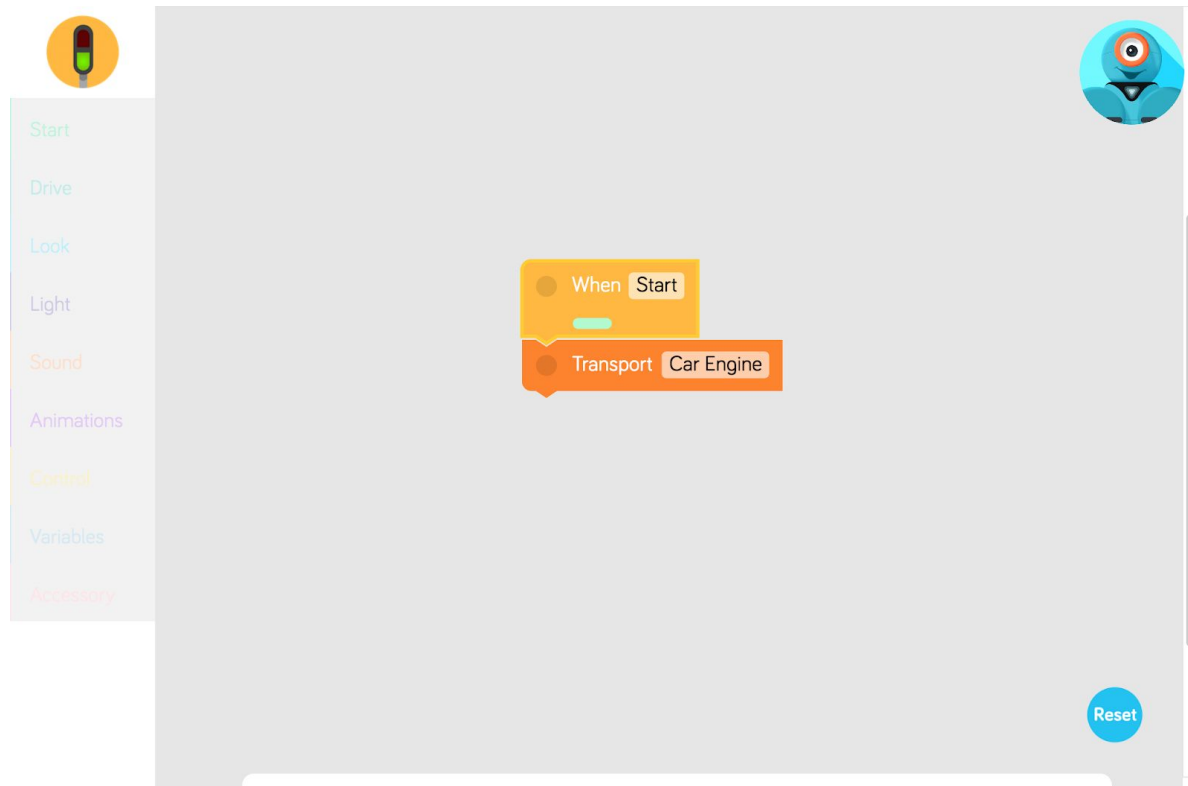
**4. Fireman Dash** \*You must have Dash's Launcher Accessory for this activity. Upfront time should be spent modeling how to use the launcher in the "Create New" section of the Blockly App. Students should practice using the launcher's directionality and power percentage before attempting this activity.

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.

# Solutions

## Challenge 1

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



The image shows a Scratch script editor interface. 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' event block (yellow) followed by a 'Transport Car Engine' block (orange). In the bottom right corner of the workspace is a blue 'Reset' button. Below the workspace is a white instruction box that reads: 'Press the **START** button in the bottom left corner to get Dash's engine going!'. At the bottom left is a green 'START ►' button. To its right is a brown left arrow button. To the right of the instruction box is a blue right arrow button. Below these buttons is a progress bar consisting of 10 segments, with the first segment highlighted in blue and the others in green.

## Challenge 2

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

The interface features a central workspace with a light gray background. On the left is a vertical palette with categories: Start (green), Drive (blue), Look (light blue), Light (orange), Sound (red), Animations (purple), Control (yellow), Variables (light blue), and Accessory (pink). Each category has a corresponding icon. The workspace contains two blocks: an orange 'When Start' block and a green 'Forward 50 fast' block. In the top right corner is a blue robot icon (Dash). In the bottom right corner is a blue 'Reset' button. At the bottom of the screen is a control bar with a green 'START' button with a right arrow, a blue left arrow button, a progress bar with 10 segments (the first is blue, the rest are green), and a blue right arrow button.

Start

Drive

Look

Light

Sound

Animations

Control

Variables

Accessory

When Start

Forward 50 fast

Reset

START ▶

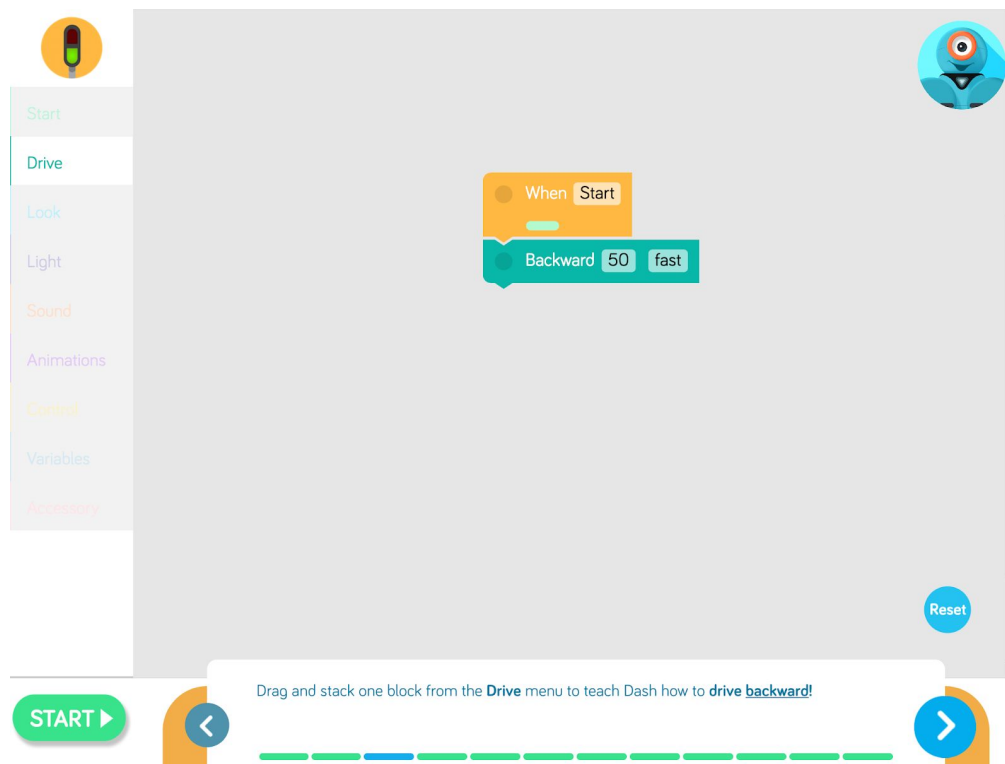
<

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

>

## Challenge 3

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



The image shows a block editor 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 currently selected. The main workspace is a light gray area where blocks are stacked. It contains two blocks: an orange 'When Start' block and a teal 'Backward 50 fast' block. A 'Reset' button is located in the bottom right corner of the workspace. At the bottom of the interface is a control bar with a green 'START' button, a blue left arrow, a series of colored dashes (green, blue, green, green, green, green, green, green, green), a blue right arrow, and an orange right arrow.

Start

Drive

Look

Light

Sound

Animations

Control

Variables

Accessories

When Start

Backward 50 fast

Reset

START ▶

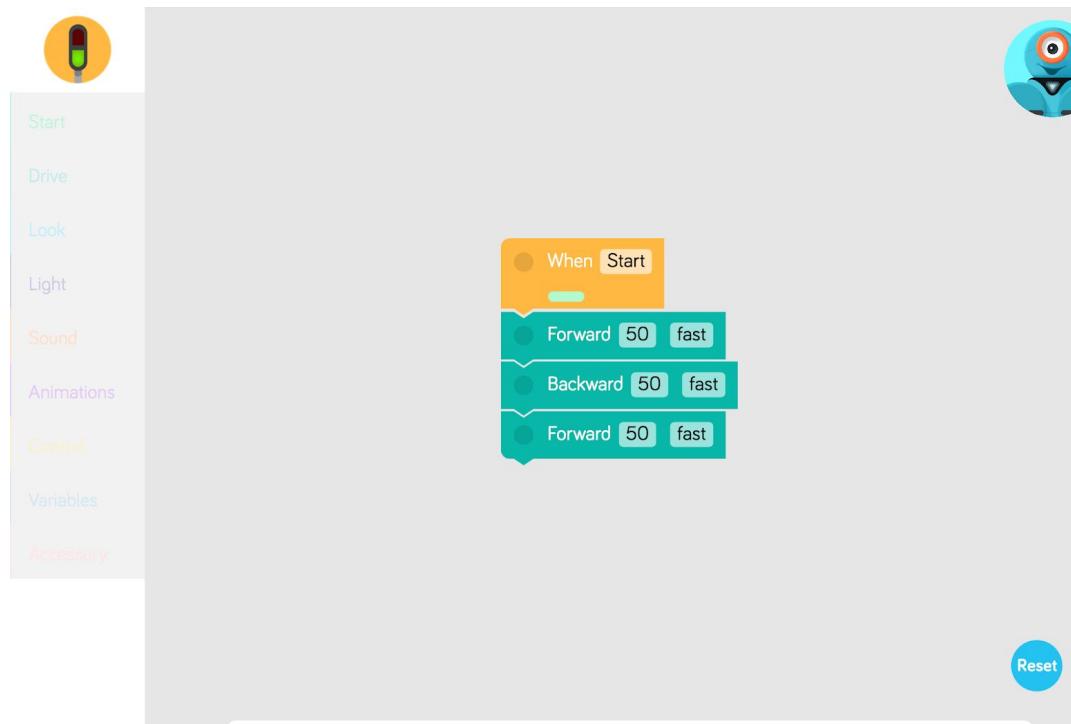
◀

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

▶

## Challenge 4

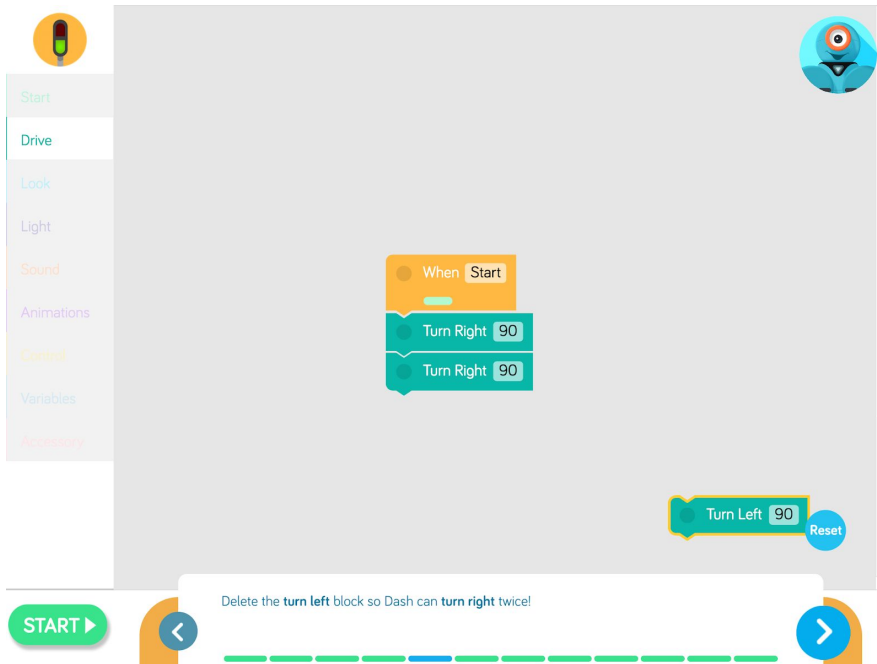
Unscramble the blocks to teach Dash how to **drive forward, backward** and then **forward** again.



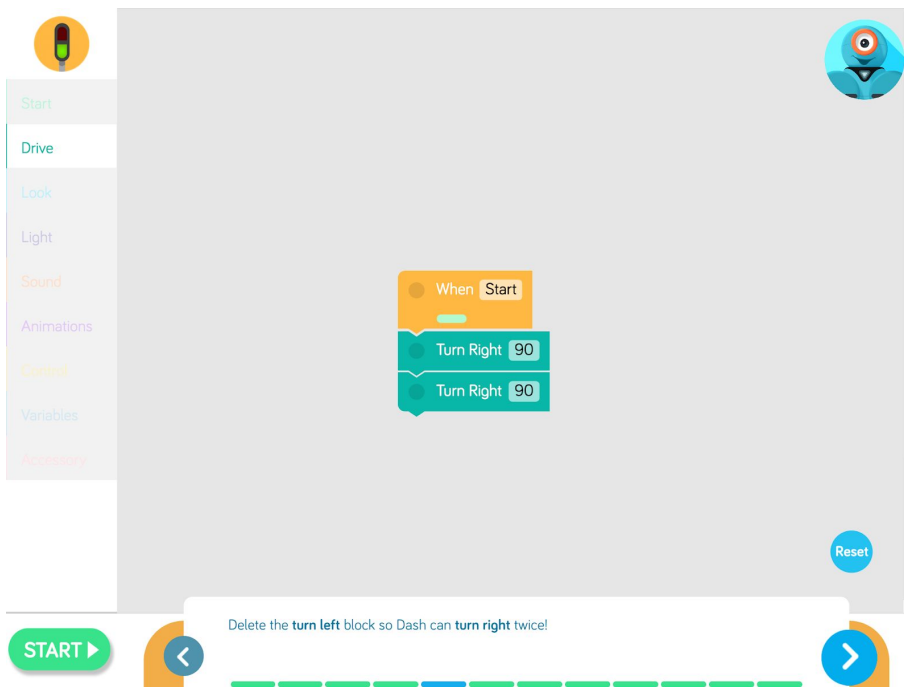
The image shows a block editor interface for teaching Dash the robot. On the left is a vertical sidebar with category icons and labels: Start (green), Drive (light blue), Look (cyan), Light (light green), Sound (orange), Animations (purple), Control (yellow), Variables (light blue), and Accessory (pink). The main workspace is a light gray area. In the top-left corner is a traffic light icon, and in the top-right is the Dash robot head icon. In the center of the workspace, three blocks are stacked vertically: an orange 'When Start' block, followed by a teal 'Forward 50 fast' block, then a teal 'Backward 50 fast' block, and finally another teal 'Forward 50 fast' block. In the bottom-right corner of the workspace is a blue circular 'Reset' button.

## Challenge 5

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



The interface shows a Scratch-like environment. On the left is a category menu with 'Start' selected. The main stage is a grey rectangle with a traffic light icon at the top left and a blue robot icon at the top right. In the center of the stage, there is a script area containing three blocks: an orange 'When Start' block, followed by two teal 'Turn Right 90' blocks. In the bottom right corner of the stage, there is a teal 'Turn Left 90' block with a blue 'Reset' button next to it. At the bottom of the interface, there is a green 'START' button, a blue left arrow button, a text instruction 'Delete the turn left block so Dash can turn right twice!', a blue right arrow button, and a blue 'Reset' button.



The interface is identical to the one above, but the 'Turn Left 90' block and its associated 'Reset' button have been removed from the bottom right of the stage. The 'Reset' button is now located in the bottom right corner of the stage area. The rest of the interface, including the menu, stage elements, and bottom controls, remains the same.

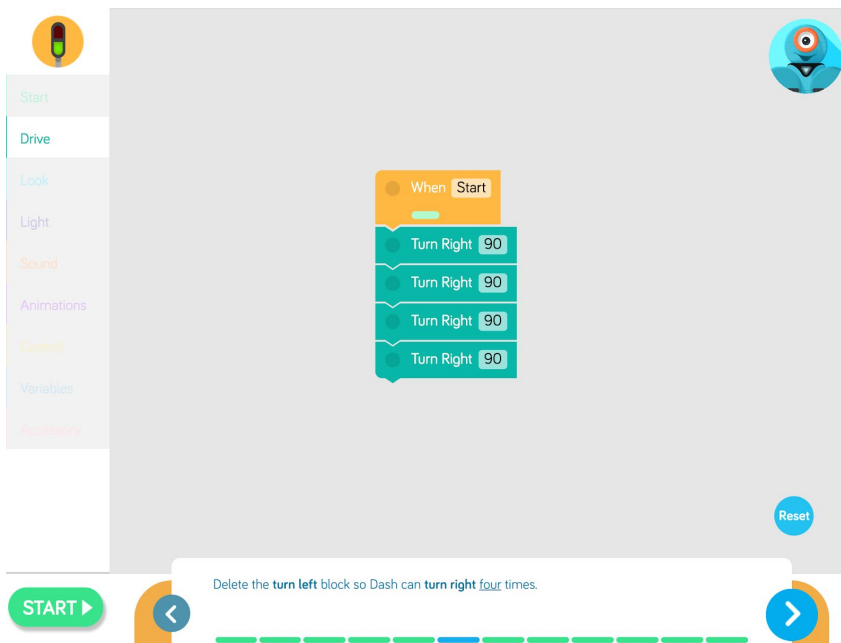


## Challenge 6

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



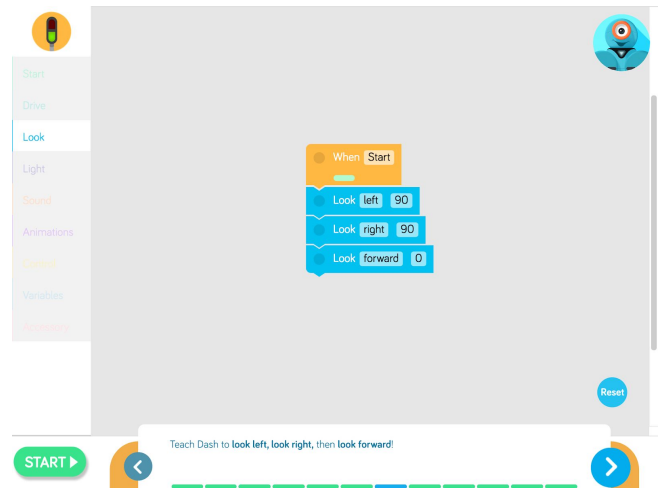
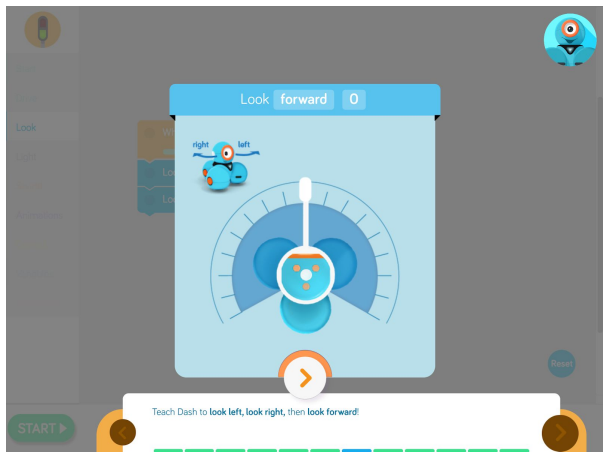
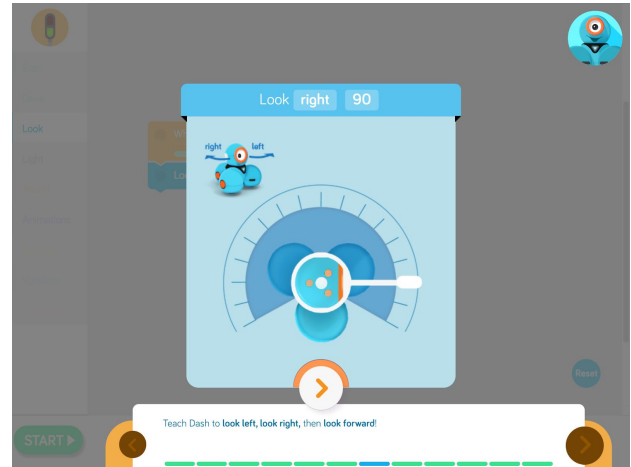
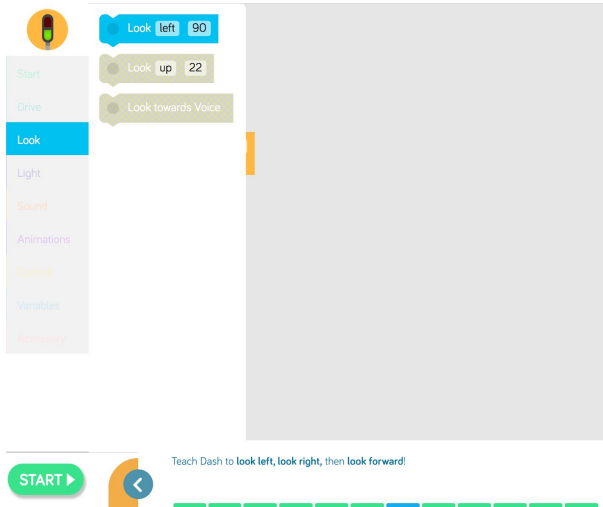
The image shows the Scratch script editor interface. On the left is a sidebar with categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessories. The main workspace contains a script starting with a 'When Start' block, followed by two 'Turn Right 90' blocks, then another two 'Turn Right 90' blocks, and finally a 'Turn Left 90' block with a 'Reset' button. A challenge instruction box at the bottom reads: 'Delete the turn left block so Dash can turn right four times.' Below the instruction is a progress bar with 10 segments, where the 4th segment is blue and the others are green. Navigation buttons for 'START', previous, and next are also present.



This image shows the same Scratch script editor after the 'Turn Left 90' block has been removed. The script now consists of a 'When Start' block followed by four 'Turn Right 90' blocks. A 'Reset' button is located at the bottom right of the workspace. The challenge instruction box at the bottom remains the same: 'Delete the turn left block so Dash can turn right four times.' The progress bar shows the 4th segment as blue, indicating the task is complete. The 'START' and navigation buttons are also visible.

## Challenge 7

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



## Challenge 8

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

The block editor shows the following sequence of blocks:

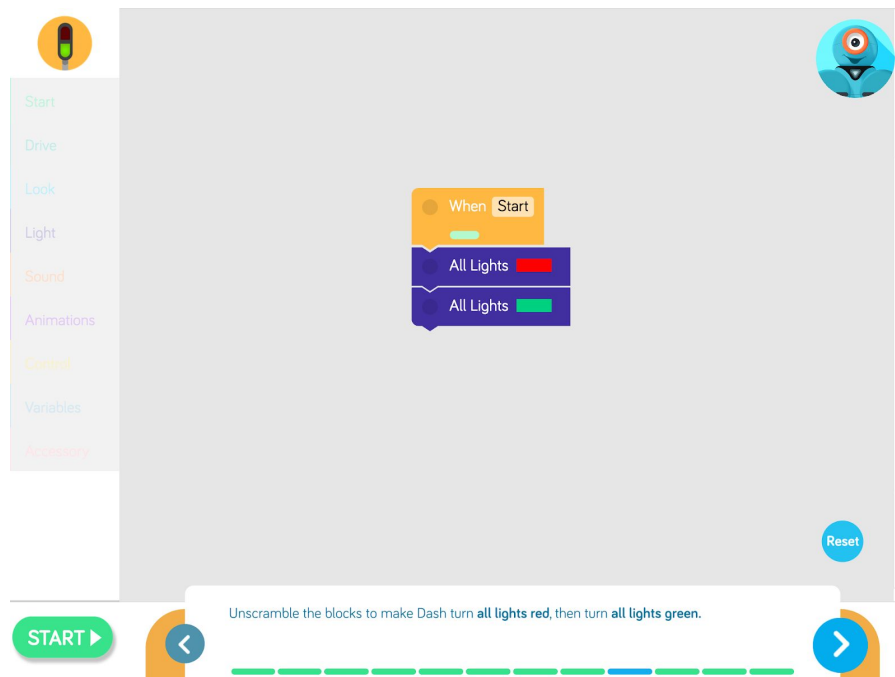
- When Start
- Forward 50 fast
- Look left 90
- Turn Left 90
- Look forward 0

The bottom control bar contains a **START** button, a left arrow, a progress bar with 10 segments (the 5th segment is blue), a right arrow, and a **Reset** button.

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

## Challenge 9

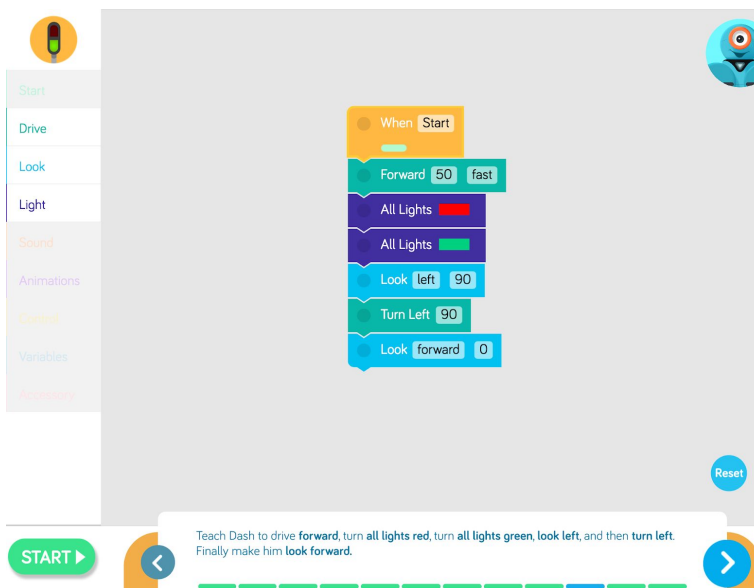
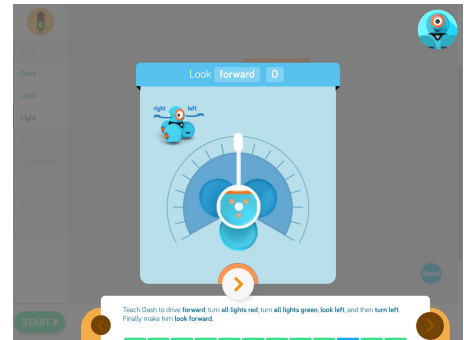
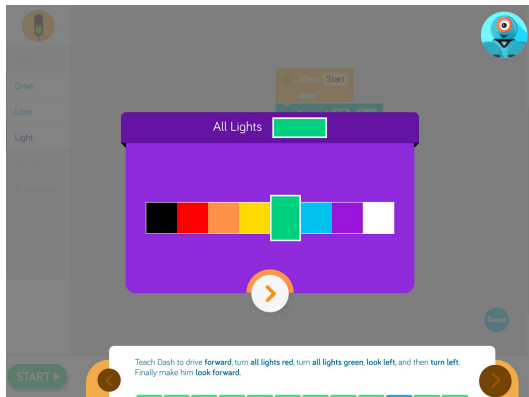
Unscramble the blocks to make Dash turn **all lights red**, then turn **all lights green**.



The interface shows a block editor with a left sidebar containing categories: Start, Drive, Look, Light, Sound, Animations, Control, Variables, and Accessories. The main workspace contains a script starting with a 'When Start' block, followed by two 'All Lights' blocks with red and green indicators respectively. A 'Reset' button is in the bottom right. At the bottom, a 'START' button is on the left, and a progress bar with 12 segments (11 green, 1 blue) is in the center, flanked by left and right navigation arrows. A text box above the progress bar reads: 'Unscramble the blocks to make Dash turn **all lights red**, then turn **all lights green**.'

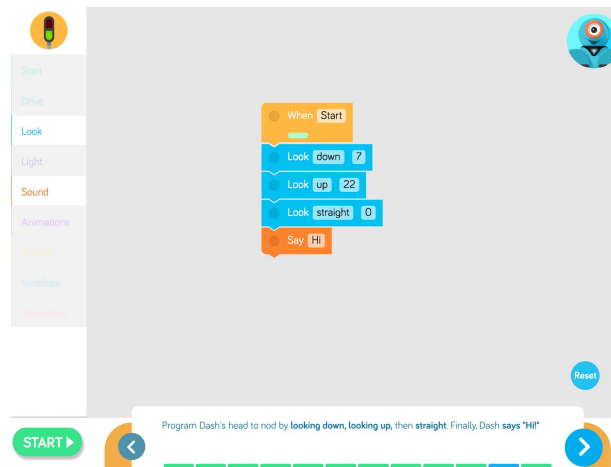
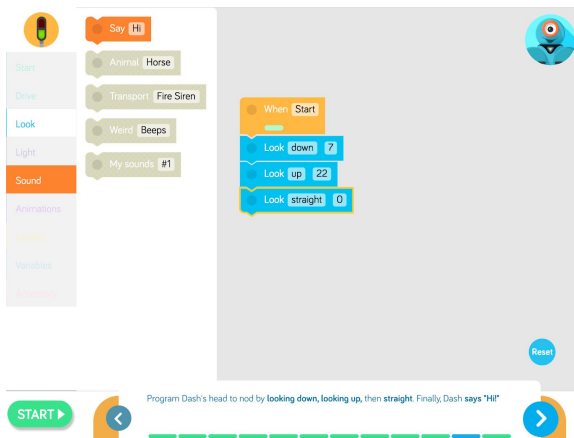
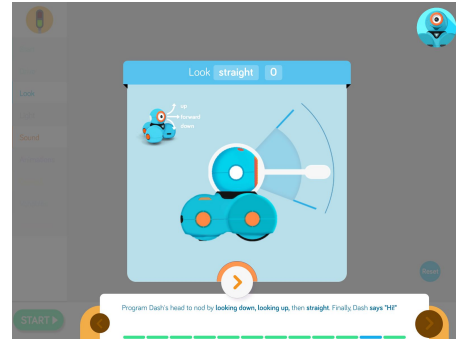
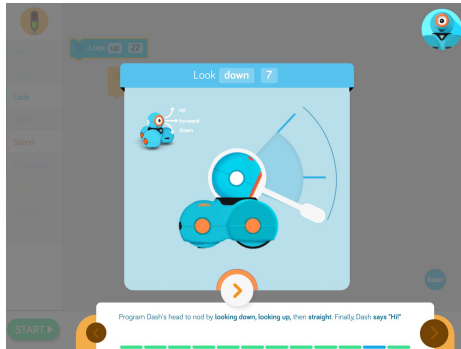
## Challenge 10

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



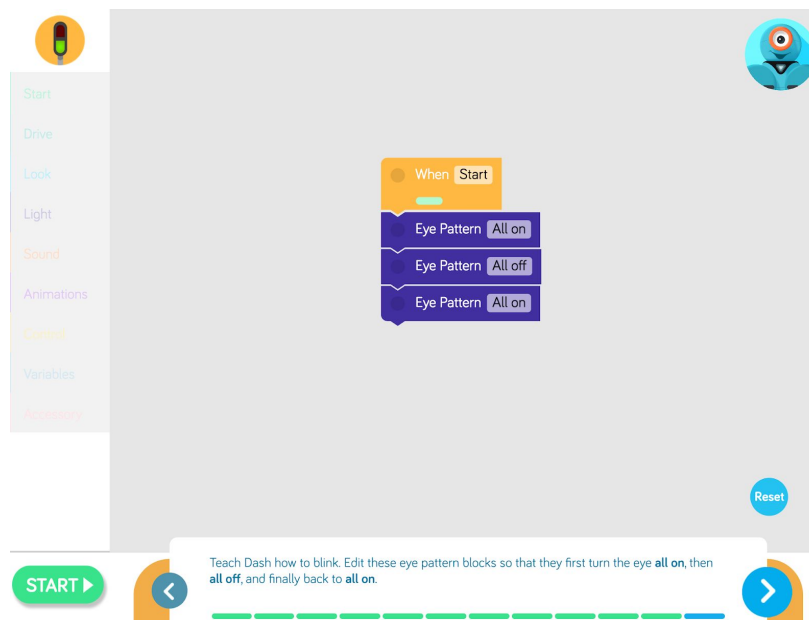
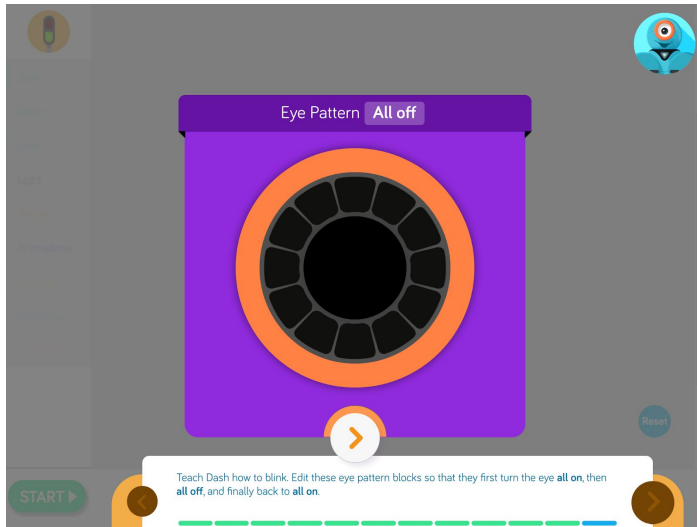
## Challenge 11

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 on**.



# 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**

W.1.2; W.2.2; W.3.2; W.3.4.2; W.5.2 *\*Relates to Extension Activity: Dash's Driver's Manual*

RI.1.6; RI.1.10; RI.2.5; RI.2.10; RI.3.4; RI.3.10

## **CSTA K-12 Computer Science Standards**

- 1A-DA-05 Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.
- 1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
- 1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.
- 1A-AP-11 Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- 1B-AP-08 Compare and refine multiple algorithms for the same task and determine which is the most appropriate.
- 1B-AP-10 Create programs that include sequences, events, loops, and conditionals.
- 1B-AP-11 Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.
- 1B-AP-12 Modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

## **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.