



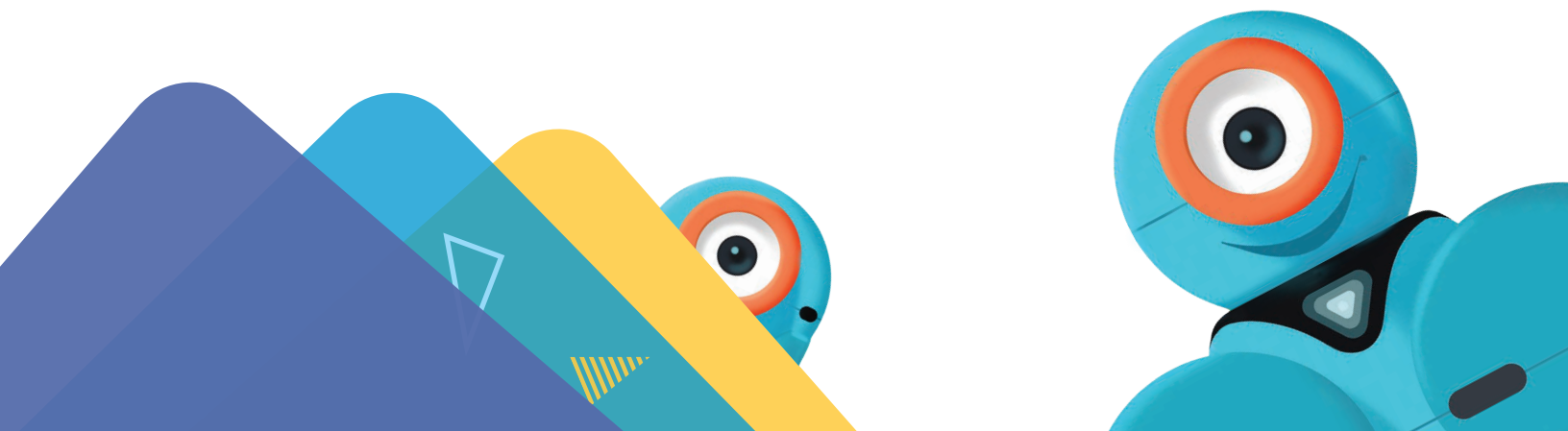


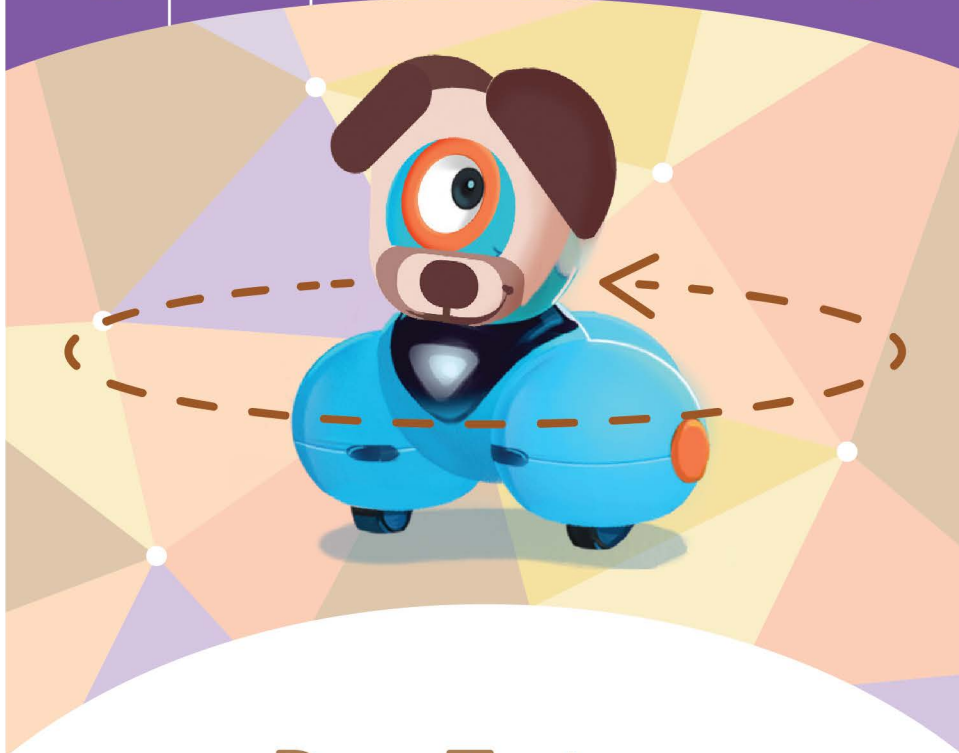
Dash the Puppy (E 3.4 - E 3.6): Teacher Packet

In this packet, we've included resources that will help you and your students as they independently complete the **Dash the Puppy Challenge Card** set:

-  **Challenge Cards:** a set of 3 **Challenge Cards** for students to practice coding concepts
-  **Solution Guides:** hints, suggestions, discussion questions, and cross curricular extension activities for each card
-  **Worksheets and Resources:** implementation strategies, planning/reflection worksheets for students, and an evaluation worksheet
-  **Lesson Plan (optional):** a whole-class instruction lesson focusing on the coding concept(s) that students will practice while completing the Challenge Cards

Looking for More? Visit: www.education.makewonder.com





Dog Trainer

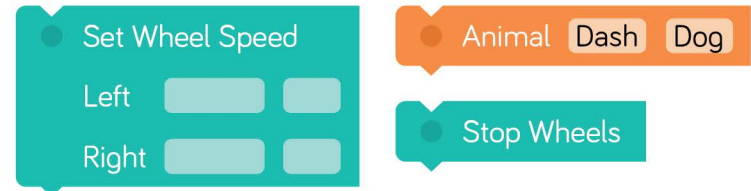
Dash is acting like a puppy
and you are the trainer.
Train Dash to turn in a circle!



1. Create a **function** to teach Dash to drive in a **circle**. Give the function a name (such as **FCircle**).



2. Put a **Set Wheel Speed** block, a **sound** block, and a **Stop Wheels** block **inside** the function.



3. Under the **When Start** block, **Call** the **Circle Function**.



4. Add some **lights** and **sounds** to give Dash praise for doing a good job!
5. Then **Call** the **Circle Function** again so that Dash gets more practice.



Dog Trainer

Time: 10-15 minutes

Hints

- To name a function, tap on the name, erase the word "Function," and add your own title. The "F" in Function does not delete, so all your function names will begin with an "F."
- Set each wheel at a different speed in order to get Dash to spin in a circle.
- To choose which function to call, tap on the **Call** block and select the function you want to use from the menu.

E | 3.4
Functions

1. Create a **function** to teach Dash to drive in a **circle**. Give the function a name (such as **FCircle**).

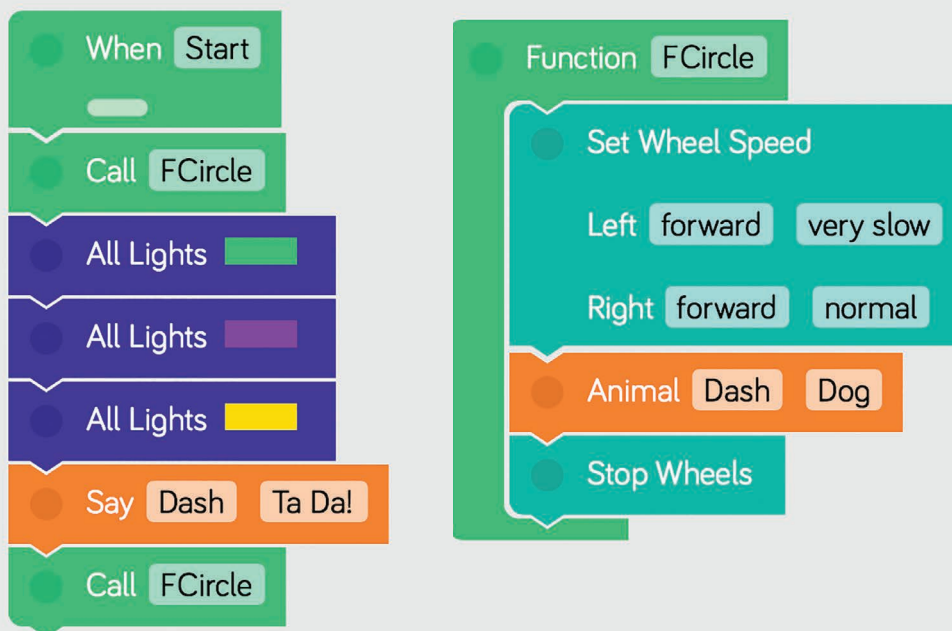
2. Put a **Set Wheel Speed** block, a **sound** block, and a **Stop Wheels** block **inside** the **function**.

3. Under the **When Start** block, **Call** the **Circle Function**.

4. Add some **lights** and **sounds** to give Dash praise for doing a good job!

5. Then **Call** the **Circle Function** again so that Dash gets more practice.

Suggested Solution:



Discussion Questions

1. A **function** is a coding shortcut. Instead of writing the entire code sequence each time you want to use it, you can create a function. Whenever you're ready to use the coding sequence, just use the **Call** block. When is it helpful to use a function instead of a **Repeat** or **When** block?
2. What other tricks would you like Dash to perform? What kind of **functions** would you need to make for each trick? What blocks would you use?

Cross-Curricular Connections

MATH

- Have students add functions that make Dash turn 5 full circles. Then have students try making Dash turn 10 full circles. (CCSS.MATH.4.MD.C.5.A)
- Make Dash's trick more complicated by using the **Set Wheel Speed** and **Stop Wheel** blocks and having Dash turn a specific number of degrees. (CCSS.MATH.4.MD.C.5.A)

ELA

- Have students research techniques used to train two different animals. Have them write a composition comparing and contrasting the techniques. Then have them write a function to demonstrate Dash completing a trick after successful training techniques had been used. Finally, have students write a different function showing how Dash would complete a trick if the training techniques used were unsuccessful. (CCSS.ELA.W.4.2)

NOTES:

E

3.5

Functions



Tricks Galore!

As a trainer, you are responsible for teaching a variety of tricks. Teach Dash two different tricks.

wonder
workshop

Copyright © 2017 Wonder Workshop, Inc. All rights reserved.

E

3.5

Functions



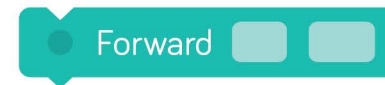
1. Dash needs to learn more tricks! Create **2** new **functions**.



2. Use **sound** blocks to make a function that teaches Dash to **speak**.



3. Use **sound**, **light**, and **drive** blocks to make a function that teaches Dash to **protect** you with loud noises, flashing lights, and brave moves.



4. To train Dash to do the tricks, **call** each **function** at least **3 times**. Practice makes perfect!

Tricks Galore

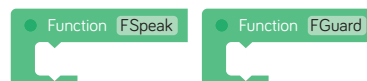
Time: 20-25 minutes

Hints

- Sometimes Dash likes to be funny and do the unexpected. When you program the **Speak Function**, have Dash speak words in addition to barking.
- To make Dash's lights flash, add a **Repeat** block inside the **Function** block.



1. Dash needs to learn more tricks! Create **2** new **functions**.



2. Use **sound** blocks to make a function that teaches Dash to **speak**.

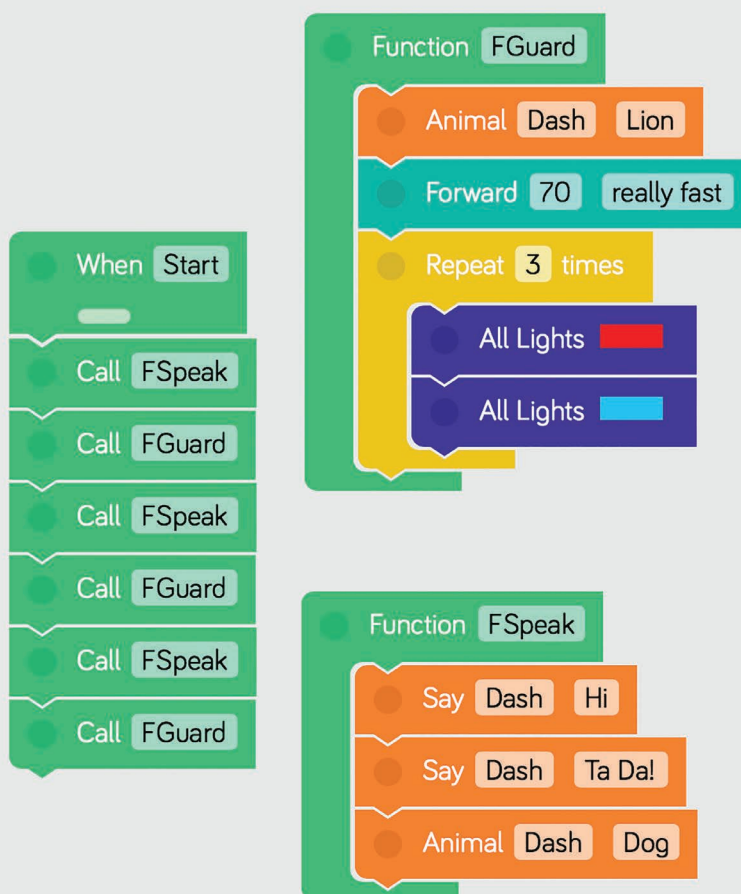


3. Use **sound**, **light**, and **drive** blocks to make a function that teaches Dash to **protect** you with loud noises, flashing lights, and brave moves.



4. To train Dash to do the tricks, **call** each **function** at least **3** times. Practice makes perfect!

Suggested Solution:



Discussion Questions

1. What would this program look like if you did not use functions?
2. How could you teach Dash a third or fourth trick? Would this be difficult or easy to do?

Cross-Curricular Connections

MATH

- Have students calculate the number of centimeters Dash travels during this challenge. Then have them change the number of centimeters Dash drives in the **Function Guard** block and solve the equation again. (CCSS.MATH.4.NBT.A.1)

ELA

- Have students record sentences using sound blocks and include them in a function to train Dash about the differences between to, too, and two. (E.g., He went to the pet store. I went there too. We got two treats for Dash.) Have them write another function to help Dash learn about the differences between there, their, and they're. (CCSS.ELA.L.4.1.G)

NOTES:



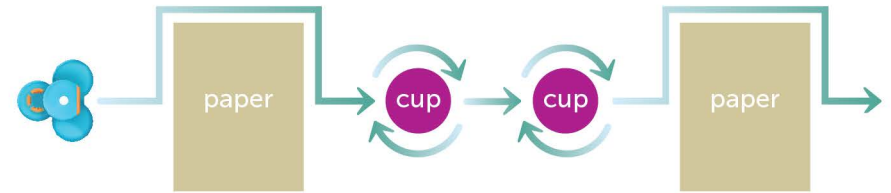
Obstacle Course!

Many animal trainers challenge their pets by having them go through obstacle courses. Now it's Dash's turn!



Materials: 2 sheets of paper,
2 cups, tape, ruler

1. Use **cups** and **paper** to set up **4 obstacles**. Place the obstacles **30 cm apart** and set Dash **in front of** them. Use **tape** to mark each obstacle's location and Dash's starting spot.



2. Program Dash to go through the obstacle course using **2 functions**—one for each obstacle type.

Hint: You will need to **call** each function **multiple times**.



Add more obstacles to the course or change the order of the obstacles.

Obstacle Course!

Time: 50-60 minutes

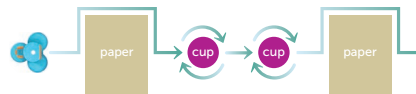
Hints

- Have Dash move slowly and in small increments to get around each circle.
- You might need to add **Drive** and **Turn** blocks between functions to get Dash to the proper starting point for each obstacle.
- Since the functions are already written, it's easy to add more of the same obstacles to your course in any order you choose.



Materials: 2 sheets of paper,
2 cups, tape, ruler

1. Use **cups** and **paper** to set up **4 obstacles**. Place the obstacles **30 cm apart** and set Dash **in front of** them. Use **tape** to mark each obstacle's location and Dash's starting spot.

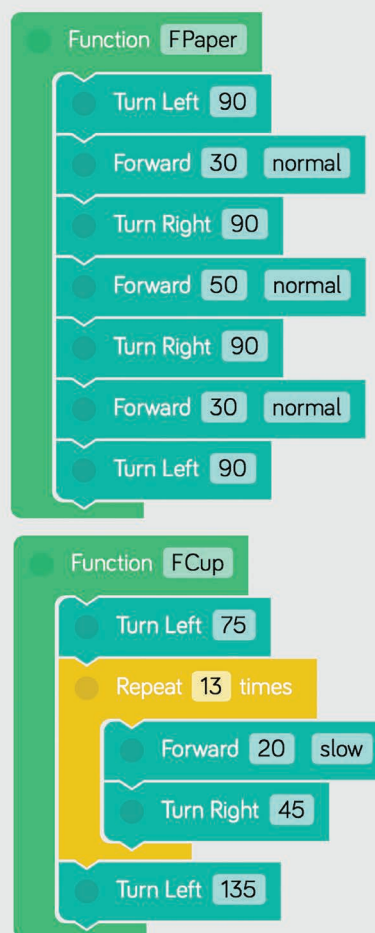
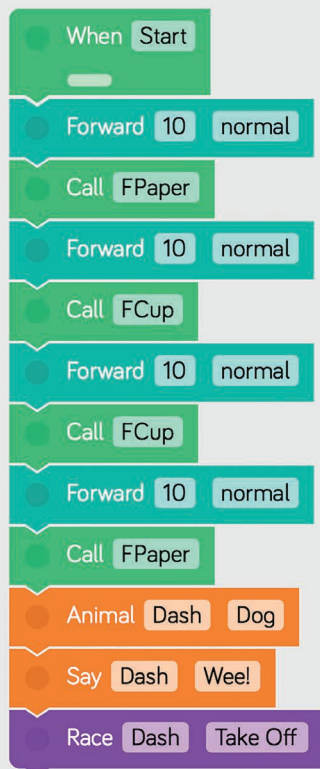


2. Program Dash to go through the obstacle course using **2 functions**—one for each obstacle type.

Hint: You will need to **call** each function **multiple times**.

★ Add more obstacles to the course or change the order of the obstacles.

Suggested Solution:



Discussion Questions

1. How would you need to change your program if you wanted to change the order of the obstacles in your course?
2. What other obstacles could you add? What functions would you need to program to go around each of the new obstacles?

Cross-Curricular Connections

MATH

- Have students write an equation showing the total number of centimeters Dash traveled in this challenge. Students must use both multiplication and addition.
(CCSS.MATH.4.NBT.B.4, CCSS.MATH.4.NBT.B.5)
- Have students drive Dash through the obstacle course 2 times. Have students modify the equation they created for the first math extension to calculate the new total number of centimeters Dash traveled. (CCSS.MATH.4.NBT.B.4, CCSS.MATH.4.NBT.B.5)

ELA

- Have students work in pairs. Have pairs take turns creating obstacle courses for their partners using classroom objects (e.g., chairs, desks, backpacks, etc.). Make sure their partners face away from them or cover their eyes so they can't see the obstacle course until later in the activity. Then have students write a paragraph that tells their partner how to get through the obstacle course. Have the partners read the paragraph before walking through the obstacle course with their eyes closed. After the activity, have students evaluate how well they wrote the directions in their paragraphs and how they could improve them. (CCSS.ELA.L.4.3)

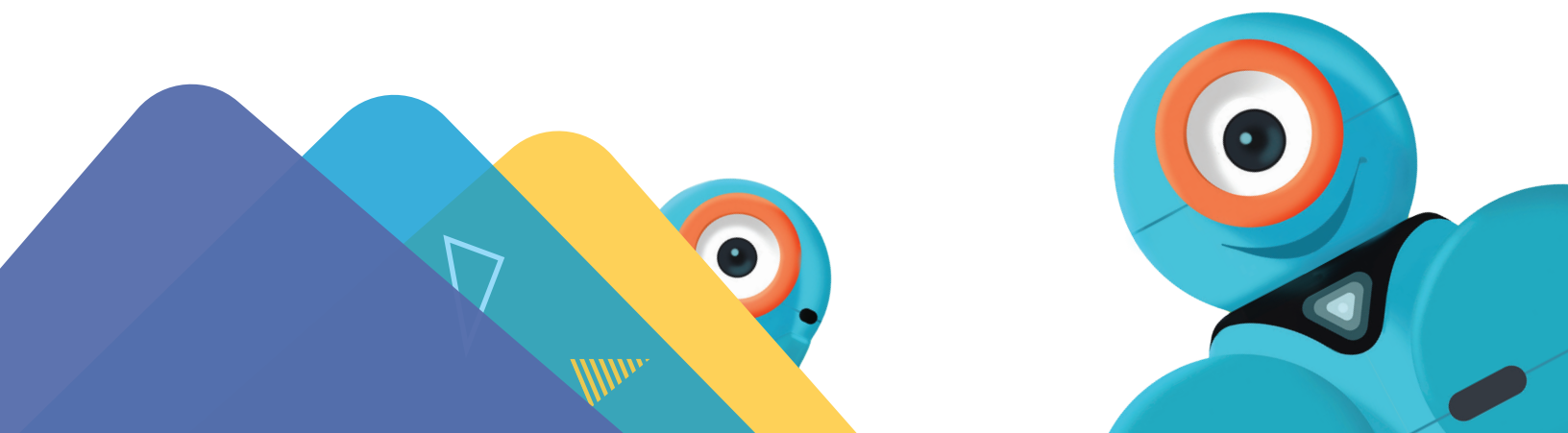
NOTES:

Worksheets & Resources

In this section, you will find the following worksheets/resources:

- Challenge Card Tips & Tricks
- Planning Worksheets
- Reflection Worksheets
- Troubleshooting Strategies
- Problem Solving & Debugging Strategies
- Evaluation Rubric
- Glossary

Looking for More? Visit: www.education.makewonder.com



Challenge Card Tips & Tricks



Determine Team Roles

Swap roles with your teammates for each challenge. Team roles include lead programmer, robot wrangler, and documentarian.



Plan Your Path

Draw out the path you want Dash to follow. Then plan out the blocks you'll need. You can also get up and walk the path that you think Dash should take.



Mark Your Spots

Use tape to mark Dash's starting spot and the location of any obstacles/objects.



Go Back to Start

Always put Dash back at the starting spot before playing a program again.



Use the When Start Block

Place your blocks under the **When Start** block. The **When Start** block should always be on your screen.



Think in Centimeters

Dash moves in centimeters. A centimeter is about the width of your finger.



Check Off the Steps

Use a dry erase marker to check off each step as you complete it. Make sure you erase the marks after you're done.



Help Your Robots Hear You

If the classroom is noisy, use the **Hear Clap** cue instead of the **Hear Voice** cue. Ask the teacher if you may try out your program with Dash and/or Dot outside or in the hallway.



Set a Time Limit

Give yourself or your team a set amount of time in which to complete the challenge

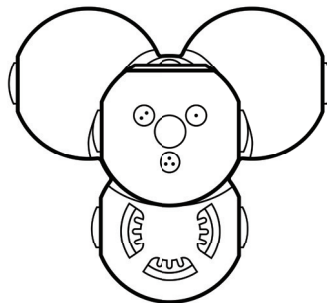
Dash Planning Worksheet

Name(s): _____ Date: _____

Coding Level: _____ Card #: _____

What do you want Dash to do?

Draw out the steps of the challenge or write a few sentences describing your goal.



General Planning Worksheet

Name(s): _____ Date: _____

Coding Level: _____ Card #: _____

1. What do you want Dash or Dot to do?

Draw out the steps of the challenge or write a few sentences describing your goal.



2. What will you do to achieve your solution?

What will each team member do? What steps will you need to take? What blocks will you use?



Reflection Worksheet

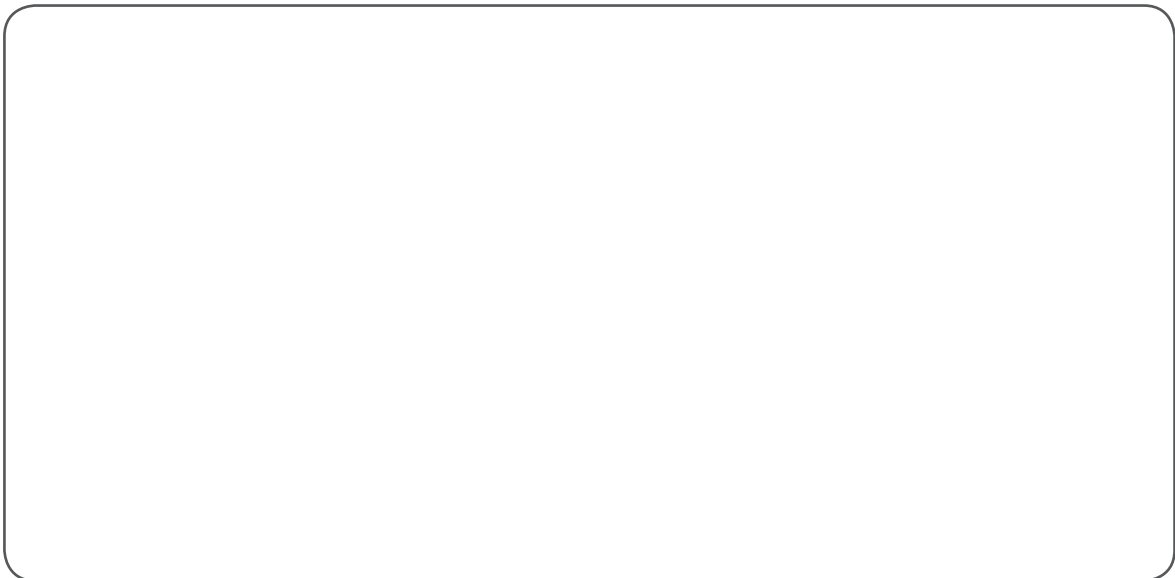
Name(s): _____ Date: _____

Coding Level: _____ Card #: _____

1. What did Dash and/or Dot do when you ran your program?



2. Did you make any mistakes? If so, how did you fix them?



Advanced Reflection Worksheet

Write a reflection entry in your Wonder Journal. Try to answer these questions as part of your reflection:

Results

- What did Dash and Dot do when you ran your program?
- Did you make any mistakes? If so, how did you fix them?

Connections

- What did you like the most about this challenge? Why?
- What was the most difficult part of the challenge? What did you learn from it?

Next Steps

- If you had more time, how would you change or add to your code?
- What are you planning to do next? Will you try another Challenge Card or start a new coding project?

Troubleshooting

If your program is not running correctly . . .

- Check if Dash and/or Dot are turned on.
- Make sure Dash and/or Dot are connected to the app.
- Make sure your blocks are connected to the **When Start** block.
- Try restarting the app.

If Dash and/or Dot are disconnecting . . .

- Turn off the robots and turn them on again. Then reconnect the robots to the app.
- Press play and then press stop to make the robots reset.
- Try charging the robots.

Three, then me!

- Ask or get help from three of your classmates. If you still need help, then ask the teacher.

Problem Solving & Debugging

Break down the challenge

- What do you need for the challenge? Which robots? Which materials and/or accessories?
- What are Dash and/or Dot supposed to do?
- Have you solved similar challenges to this one?
- Focus on one step at a time.

Plan your solution

- Draw a picture or make a list of what you want Dash or Dot to do.
- What blocks will you need to complete the challenge?
- Are there any hints on the card that can help?
- Use tape to mark Dash's starting point.
- Use tape to mark each obstacle's location.

Test Your Code

- Does your code complete the challenge?
- If not, play your code again. Watch as the program goes through each block. Do you notice any mistakes?
- Do you need to change, delete, or add more blocks?
- Are your blocks telling Dash to do something when you actually want Dot to do something?

Improve your work

- Ask another student or group to check your program.
- Is there an easier way to complete the challenge? Can you use fewer blocks?
- How can you improve your program? Could you add more lights, sounds, or other customizations?

Evaluation Rubric

	Programming	Reflection & Documentation	Collaboration & Communication	Creativity
1 Novice	Completed part of the activity and needed assistance throughout the process.	Use a journal, worksheets, and/or multimedia tools (such as video and images) to document some of the activity results.	Participated little or not at all in classroom discussions. Demonstrated little to no cooperation with group members during the activity.	Demonstrated limited creativity in developing ways to complete the activity.
2 Developing	Used the targeted coding concept(s) to complete the activity with some assistance.	Incorporated some target vocabulary and some thoughtful reflection on the coding process while documenting activity results using journal entries and multimedia tools.	Occasionally participated in classroom discussions and cooperated somewhat with group members.	Developed a few different ways to complete the activity, but the solution was not particularly creative.
3 Proficient	Used the targeted coding concept(s) to complete the activity without assistance.	Incorporated target vocabulary and reflection on the coding process. Clearly documented activity results using journal entries and multimedia tools.	Actively participated in classroom discussions. Answered questions and cooperated with group members during the activity.	Applied the iterative process to develop creative and unexpected solutions for the activity.
4 Exemplary	Used the targeted coding concept(s) to complete the activity without assistance. Enhanced the solution with more efficient (e.g., fewer blocks) and/or advanced features (e.g., lights, sounds) in the code.	Incorporated advanced target vocabulary and in-depth reflection on the coding process. Thoroughly and clearly documented and presented activity results.	Actively participated in classroom discussions and cooperated with group members. Gave constructive feedback to others and effectively incorporated feedback from others.	Went above and beyond to develop, revise, and execute imaginative solutions for the activity.



dash & dot[™] challenge cards

Coding Lessons

The following lesson plan is a part of
our **Learn to Code Curriculum**.

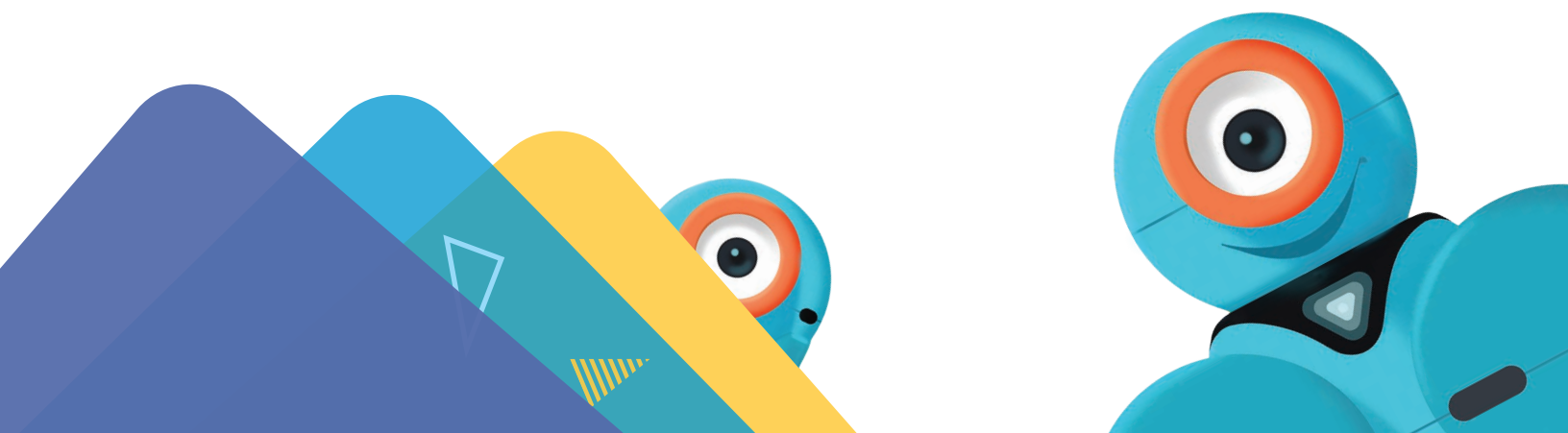
To check out more free coding lessons, please visit:

www.education.makewonder.com/curriculum/learn_to_code

Cross Curricular Lessons

Looking to go beyond the **Hour of Code**? Your students can **code to learn** via our project-based cross curricular lessons!

www.education.makewonder.com/curriculum/code_to_learn



Level E - Lesson 5

Functions: Part 2

Lesson Information

Overview/Description

Students will use **functions** and combine them with **event handlers** and **loops**.

They will use *Blockly* preset programs and **Challenge Cards** to practice their new skills.

Coding Level: E

Prior Experience:

Before this lesson, it is recommended that students complete:

- Level A–D: All Lessons
- Level E: Lessons 1–4
- [Optional] Level A–D **Challenge Cards**: All
- [Optional] Level E **Challenge Cards**: 1.1–3.3

Learning Objectives:

Students will:

- Define a **function**.
- Design a program that combines **event handlers**, **loops**, and **functions**.
- Use **functions** to revise code and complete coding challenges in an iterative process.

Target Grade Range: 3–5

Suggested Group Size: 2–3 students per robot

Time Required: 45–60 minutes

Materials:

- 1 Dot robot per group
- 1 tablet per group
- projector or interactive display with mirroring capability
- pencils
- **Challenge Cards**: E 3.4, E 3.5, E 3.6

- [Optional] 1 Dash robot per group
- [Optional] access to **Twitter** and **Instagram**

Resources/Downloads:

- **Troubleshooting** handout
- **Problem Solving & Debugging** handout
- **Wonder Journal: Dot Planning** and/or **General Planning** worksheets
- **Wonder Journal: Reflection** worksheets
- ***Blockly*** Puzzle Checklist
- **Evaluation Rubric**
- [Optional] **Challenge Card Checklist**
- [Optional] **Challenge Card Tips & Tricks** handout

Preparation:

- Fully charge the tablets and robots.
- Install the **Wonder Workshop Blockly** app on each tablet.

Review

FUNCTIONS REVIEW

1. Briefly review how students used **functions** during the previous lesson by asking:
 - “What is the purpose of **functions**?” (Sample response: “**Functions** help make programs more efficient by allowing us to repeat bits of code throughout the program.”)
 - “How are **functions** different from **loops**?” (Sample Response: “**Loops** repeat bits of code consecutively, while **functions** repeat bits of code in many different places.”)

CODE.ORG CONNECTIONS [Optional]

1. Review *Code.org*'s lesson focusing on **functions** and **loops**:
<https://studio.code.org/s/courseee-draft/stage/7/puzzle/1>.
2. Point out how students needed to use both **functions** and **loops** to complete the lesson. Ask, “What was difficult about this task?”
 - Sample response: “It was difficult to follow the different parts of the code. I wasn’t sure what would happen next.”

Direct Instruction

INTRODUCTION

1. Say, “We have done a couple of activities using **functions**. Now we are going to combine **functions** with some other familiar coding concepts like **loops** and **event handlers**.”
2. Review a real-life example where you could use all of these concepts together:

- When you go to the store, you use a **loop** to walk through the store: you walk up an aisle, turn right or left, and then repeat. Once you are in the aisle, you use an **event handler**: you keep walking until an **event** happens, such as seeing something that's on your shopping list. When the **event** happens, you **call** a "pick up" **function**: you reach out your hand, pick up the object, pull your hand back, and put the object in the cart.

QUICK CHECK

- Ask, "What is a **loop**?" (Possible Response: "A **loop** is a piece of code that repeats.")
- Ask, "How does an **event handler** work?" (Possible response: "An **event handler** waits for an event to occur before running the program under it.")

Guided Practice

Activity: Tickle, Tackle

1. Project your tablet screen, open the *Blockly* app, and go to the menu at the top left of the screen.
2. Go to the **Create New Project** menu and select the **Tickle Tackle** preset program. Then tap "create."



3. Say, "Let's see how *Blockly* uses both **functions** and **event handlers** in the **Tickle, Tackle** preset program."

- Ask the students to look at the program and predict what they think will happen when it is run.

- Run the program to see if the students' predictions were correct.
4. Ask, "How many **event handlers** can you find in this program?"
 5. Have student volunteers describe what happens when the program starts.
 - Possible response: "Dot says, 'How do you do?'"
 6. Ask, "How would you make this program work for Dash instead of Dot?"
 - Possible response: "Change each **event handler** to Dash **events**. Change Dot's **sound** and **animation** blocks to Dash's sounds and animations."
 7. Say, "What if we wanted Dot to giggle twice when Dot's Top Button is pressed? What should we change?" (Possible response: "Add another **Call** block.")
 - Have a student volunteer add another **Call** block under the selected **event handler** and then press play to see how the program changed.

QUICK CHECK

- How many **functions** can you use one program? (Possible response: "You can use an unlimited number of functions.")
- If you use more than one **function**, how does your program know which one to "call?" (Sample response: "You assign each function a different name so that when you call the function, the program knows which one you want.")

Independent Practice

Have students work on the following activities in small groups (ideally 2–3 students per robot).

Encourage students to share tablet and robot time. Have them establish and rotate through roles such as:

- **Lead Programmer:** Holds the tablet and manipulates the code.
- **Robot Wrangler:** Retrieves and resets the robot after every program attempt.
- **Documentarian:** Records group results, thoughts, and progress. Illustrates group designs and ideas.

When students work together while coding, they're able to help each other identify mistakes and develop creative solutions!

Tickle Tackle Extension

1. Have students use the **Wonder Journal: Dot Planning** and/or **General Planning** worksheets to design ways to add to or alter the **Tickle Tackle** preset program. They can:
 - Change the **function** or add more **functions**.
 - Change or add **lights**, **movements**, and **sounds** in each **function**.
 - Add a **loop**.
 - Change the **cues**.

2. After they finish revising the program, have students:
 - Complete a **Wonder Journal: Reflection** worksheet.
 - Take a screenshot of their *Blockly* code.
 - Take a video of Dot while the code is running.

Challenge Cards [Optional]

1. You can purchase our *Learn to Code* Challenge Card sets and Curriculum Guide here: <https://store.makewonder.com/#/education>.
2. Have students complete the following Challenge Cards:
 - **E 3.4: Dog Trainer**
 - **E 3.5: Tricks Galore**
 - **E 3.6: Obstacle Course**
3. For each challenge, encourage students to:
 - Use the **Wonder Journal: Dash Planning** and/or **General Planning** worksheets to discuss how they can complete the challenge.
 - Review the **Troubleshooting** and **Problem Solving & Debugging** worksheets if they run into any problems with their code.
4. After they finish each challenge, have students:
 - Complete a **Wonder Journal: Reflection** worksheet.
 - Take a screenshot of their *Blockly* code.
 - Take a video of Dash while the code is running.

Wrap Up

Student Presentations

1. Have student groups take turns sharing one of their programs with the class. Encourage them to:
 - Explain their design thinking. (E.g., "We wanted to add another **function** to the program because we wanted to include our own sounds," "We changed actions in the **function** blocks here to make Dot act dizzy.")
 - Share any obstacles and difficulties they overcame during the activity. (E.g., "At first, we weren't sure about the best place to add a **loop**, but then we ran the program a few times with the loop in a variety of locations to find the best place to put it.")
2. Encourage students to ask each other how they accomplished different objectives and give each other feedback on their programs. Possible questions/feedback includes:
 - "How did you (add another **function**)?"
 - "I like where you (added the **loop**)."
 - "What if you (changed the **cues**)?"

Follow-Up Questions/Discussion

- Why are **functions** important? (Sample response: “**Functions** help repeat non-consecutive sequences of code”.)
- How was your program affected by loops? (“**Loops** allowed me repeat bits of code over and over.”)

Assessment

- Use our **Evaluation Rubric** to review students’ work and presentations.
- [Optional] Share your students’ work with the world using @wonderworkshop and #dashanddot!

Standards

CSTA

- Create programs that include sequences, events, loops, and conditionals.
- Modify, mix and incorporate portions of an existing program into one’s own work to develop something new or add more advanced features.
- Take on varying roles with teacher guidance, when collaborating with peers during the design, implementation, and review of program development.

ISTE

- 4d: Exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems.
- 5a: Formulate problem definitions suited for technology-assisted methods such as data analysis, abstract models and algorithmic thinking in exploring and finding solutions.
- 6b: Create original works or responsibly repurpose or remix digital resources into new creations.
- 7c: Contribute constructively to project teams, assuming various roles and responsibilities to work effectively toward a common goal.

NGSS

- 3-5-ETS1-2: At whatever stage, communicating with peers about proposed solutions is an important part of the design process, and shared ideas can lead to improved designs.
- 3-5-ETS1-3: Tests are often designed to identify failure points or difficulties, which suggest the elements of the design that need to be improved.

Common Core

- CCSS.ELA-LITERACY.W.2.6: With guidance and support from adults, use a variety of digital tools to produce and publish writing, including in collaboration with peers.
- CCSS.ELA-LITERACY.W.2.8: Recall information from experiences or gather information from provided sources to answer a question.
- CCSS.ELA-LITERACY.SL.3.1: Engage effectively in a range of collaborative discussions (one-on-one, in groups, and teacher-led) with diverse partners on grade 3 topics and texts, building on others' ideas and expressing their own clearly.
- CCSS.ELA-LITERACY.SL.3.3: Ask and answer questions about information from a speaker, offering appropriate elaboration and detail.
- CCSS.ELA-LITERACY.SL.3.4: Report on a topic or text, tell a story, or recount an experience with appropriate facts and relevant, descriptive details, speaking clearly at an understandable pace.