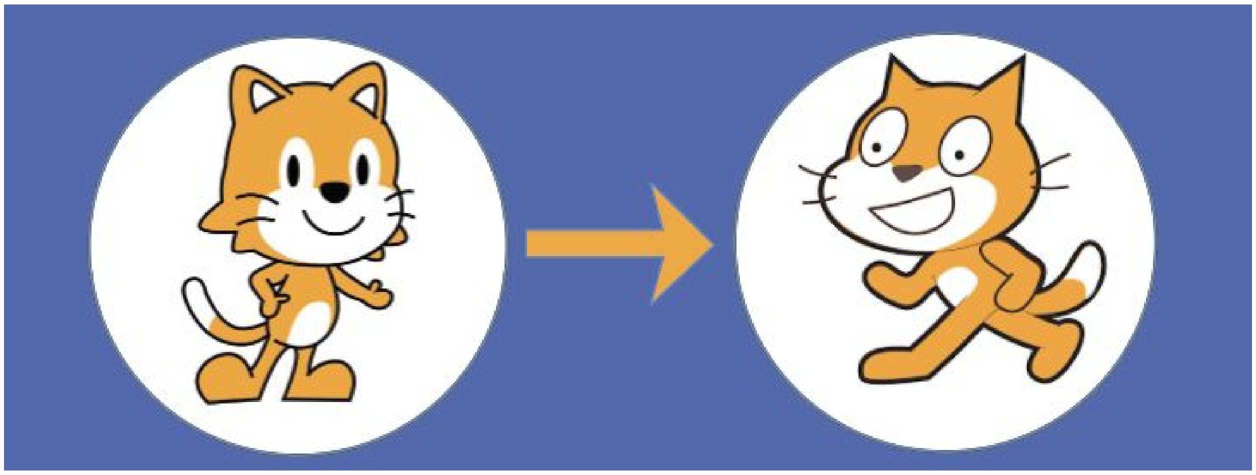


Transitioning from ScratchJr to Scratch

A curricular guide for helping teachers evaluate their child's expertise in ScratchJr, and utilize ScratchJr knowledge to begin learning Scratch

Claudia Mihm
DevTech Research Group
Eliot Pearson Department of Child Study and Human Development
Tufts University
http://sites.tufts.edu/devtech

ScratchJr to Scratch Transition Guide

This guide is for educators whose students have mastered ScratchJr, or are feeling restricted by it, and wish to move on to learning Scratch. While there is always more room for creativity in ScratchJr, there is also a potential for expanding both the creativity and the computational thinking learning in Scratch. There are a wide range of resources available for learning ScratchJr and Scratch, but this guide hopes to offer resources for the transition between the two platforms. The curriculum focuses on building on students' knowledge of ScratchJr in order to ease their transition into learning Scratch.

This guide has several parts. First, it features a chart of computational thinking abilities that ScratchJr beginners, ScratchJr experts, Scratch beginners who are coming from using ScratchJr, and Scratch experts who are coming from ScratchJr, each exhibit. It draws on Dr. Marina Bers' 7 Powerful Ideas of Computational Thinking to create the categories. This chart is provided as context for understanding the guide's approach to evaluating ScratchJr expertise. Second, it provides a lesson plan for transitioning students out of ScratchJr and evaluating their readiness for Scratch, and then easing their introduction to Scratch, relying on their ScratchJr knowledge.

Computational Thinking Skills

This chart contains examples of how the 7 Powerful Ideas of Computational Thinking appear in beginners and experts in ScratchJr and Scratch, specifically those who are learning Scratch after using ScratchJr. These 7 Powerful Ideas come from research by Dr. Marina Bers.

Powerful Idea	ScratchJr Beginners can...	ScratchJr Experts can...		Scratch Beginners can...	Scratch Experts can...
Modularity	• Break up a story or action into smaller steps	• Break down more complex problems (i.e. with 3+ different actions) into small steps • Understand how to combine pieces of different programs to achieve a goal		• Understand how to break down problems using ScratchJr logic (same blocks/control flow)	• Use remix to reuse other's code • Use code from others or their own programs in new ways
Algorithms	• Understand what they want a character to do • Mostly use motion/looks blocks • Understand basics of *linear sequencing*	• Design non-linear algorithms • Use wider range of blocks to create program (such as messaging, orange control flow blocks, and changing the number parameter on movement blocks)		• Design algorithms and implement with *equivalent blocks*	• Use a variety of algorithms to make a wide range of programs
Control Structures	N/A	• Successfully use and understand the effect of a repeat loop • Understand how control is passed around through message blocks		• Understand simple loops and *equivalent trigger blocks* • Understand how to use messages in Scratch	• Understand and use all trigger and loop blocks • Use these blocks appropriately to serve a specific need
Hardware /Software	• Grasp distinction between iPad (hardware) and app (software)	N/A		• Understand that Scratch is on a website, which is accessed via a computer (hardware)	N/A
Debugging	• Articulate what a character should be doing, and when it is not doing that • Debug errors in visible output (i.e. motion)	• Debug more complex, non-visible bugs • Debug other's code		• Debug programs made with *equivalent blocks*	• Debug self and others' code
Representation	• Understand that different colors represent different types of actions	• Differentiate among categories of blocks		• Understand that characters move on a Cartesian plane • Understand equivalent categories in ScratchJr	• Understand all categories in Scratch • Utilize variables successfully
Design Process	• Understand basic design process • Walk through design process supervised	• Self-initialize the engineering design process		• Move through the design process while building with Scratch, with some supervision	• Can design, build, and iterate on a project independently in Scratch

Glossary:

- Linear sequencing: a sequence that executes one after the other, and does not loop back on itself or skip any steps (i.e. Kitten walks forward, then jumps, then says “hi!”)
- Non-linear sequencing: a sequence that does not execute one after the other, because it loops back on itself or contains conditionals (i.e. a repeat loop)
- Equivalent blocks: blocks that appear in both ScratchJr and Scratch (i.e. movement blocks)
- Algorithms: An algorithm is a series of ordered steps taken in a sequences to solve a problem or achieve an end goal.
- Modularity: Modularity breaking down tasks or procedures into simpler, manageable units that can be combined to create a more complex process.
- Control Structures: Control structure determine the order (or sequence) in which instructions are followed or executed within an algorithm or program.
- Representation: Representation is the notion that concepts can be represented using symbols (for instance, ScratchJr uses color to represent different types of instructions)
- Hardware/Software: Hardware and software are necessary for a computer system to operate, as software provides instructions to the hardware.
- Design process: a design process is an iterative process used to develop programs and tangible artifacts that involves several steps.

Transition Curriculum Overview

The chart above outlines the general computational thinking skills that a ScratchJr or Scratch user should have. This curriculum focuses on the two center columns (ScratchJr expert and Scratch beginner). Based on these skills, the transition curriculum has two parts. First, it has an expertise evaluation activity to ensure that students have the necessary skills to comfortably transition to Scratch. Second, it asks students to design a project in Scratch, expanding on a ScratchJr project, in order to support their transition to Scratch with their ScratchJr knowledge.

1. ScratchJr Expertise Evaluation: students will complete a design sheet and activity to evaluate ScratchJr expertise

    1. Creative ScratchJr activity

        1. Students will first complete their ScratchJr Project Design Sheet, where they will plan a story to tell, using ScratchJr. It will ask them to plan their characters and settings, as well as to think through some of their algorithm design.
        2. Once they have completed their sheet, the teacher will look over it with them to ensure they have a sense of the story they are building.
        3. Then, they will begin building their project in ScratchJr.
        4. During this time, the teacher should be observing their building process, if possible, keeping an eye out for behaviors outlined on the teacher evaluation sheet.

5. If a teacher is unable to observe during the building process, there is still an opportunity to evaluate their expertise by asking students questions and and examining their planning sheet and final project. The teacher should encourage the student to iterate on their project, adding to their story as long as they would like.

b. **Resources:**

i. ScratchJr Project Design Sheet

ii. Teacher Evaluation Sheet

2. Transition to Scratch: student will expand from their ScratchJr project in Scratch

1. Expansion Activity:

1. Students will be asked to fill out the Scratch Project Design Sheet, in which they are asked to expand on the story they just told, but this time in Scratch. They are prompted to either continue telling their story, adding new characters and settings, create a game out of their story, or re-tell their story. They will also be given a blocks guide, which helps them understand how to achieve the same behavior in Scratch as they would in ScratchJr. It also highlights a few new things that they are able to do in Scratch, such as move their character with arrow keys, and have characters change costumes. For this part, teachers are encouraged to refer to the blocks guide, and rely on their student's ScratchJr knowledge in order to help them understand the Scratch world.

2. Students will then build their projects in Scratch, according to the plan they designed, and utilizing the blocks guide and other Scratch resources.

b. **Resources:**

i. Scratch Project Design Sheet

ii. ScratchJr to Scratch Blocks Guide

Transition Curriculum, Part 1: ScratchJr Expertise Evaluation

Required Resources:

- ScratchJr Project Design Sheet
- Tablet (to access ScratchJr)
- Writing Utensil

1. Complete ScratchJr Project Design Sheet: ask students to complete the project design sheet, planning their project for ScratchJr

ScratchJr Project Design Sheet

Write or draw the story that you want to tell in ScratchJr: Who are your characters? What are they doing? Where are they?

In your story, pick an action that you think might be hard to program. What blocks do you think you will need to program that action?

What characters in ScratchJr will you use? Which backgrounds?

2. Teacher Observation Sheet Part 1: After Completing Design Sheet

Sit down with your student, and have their design sheet in front of you. Some questions ask you to observe what they have written, and some questions are for you to ask your student directly.

If there is more than one question in a category, check yes if they fulfill at least one of the questions.

Powerful Idea	Computational Thinking Ability	Questions to Ask While Evaluating	Does the student demonstrate this skill?
Modularity	● Break down more complex problems (i.e. with 3+ different actions) into small steps ● Understand how to combine pieces of different programs to achieve a goal	Do they break their story up into more than one step on their planning sheet?	❏ Yes ❏ No
Algorithms	● Design non-linear algorithms ● Use wider range of blocks to create program (such as messaging, orange control flow blocks, and changing the number parameter on movement blocks)	For the question *"In your story, pick an action that you think might be hard to program. What blocks do you think you will need to program that action?"* If they wrote something: Do they point out an action and some blocks they could use to complete it? If they didn't write something: Are they able to tell you what a specific character is going to do, and what blocks they will need to accomplish that action?	❏ Yes ❏ No
Hardware/ Software	● Grasp distinction between iPad (hardware) and app (software)	N/A	❏ Yes ❏ No
Control Structures	● Successfully use and understand the effect of a repeat loop ● Understand how control is passed around through message blocks	Are they planning on using messaging blocks or repeat loops in their program? Can they explain how these would work?	❏ Yes ❏ No
Debugging	● Debug more complex, non-visible bugs ● Debug other's code	N/A	❏ Yes ❏ No
Representation	● Differentiate among categories of blocks	Can they explain the difference between the various categories? Can they correctly identify explain the purpose of a block when asked?	❏ Yes ❏ No

3. Build ScratchJr Project: Now, have your student begin building the ScratchJr project that they designed. If you are able, observe their process while they are working, and complete the questions under the "To Observe" labels in Part 2 of the teacher evaluation sheet. Whether or not you are able to observe while they are working, use the "To Ask" questions in Part 2 after they are finished building their project.

4. Teacher Observation Sheet, Part 2: During and After Programming a Story:

If you are able to, observe your student while they are programming. While you are observing them, watch for the behaviors asked about under the “To observe” heading in each category. If you are unable to observe them while they are building, sit down with them and ask them the questions under the “To ask” heading in each category. As with the above questions, check “yes” if they fulfill at least one of the questions in each category.

Powerful Idea	Computational Thinking Ability	Questions to Ask While Evaluating	Does the student demonstrate this skill?
Modularity	● Break down more complex problems (i.e. with 3+ different actions) into small steps ● Understand how to combine pieces of different programs to achieve goal	**To observe:** While they are building, do they use pieces from other programs they have built, or code from other characters in their story? **To ask:** Do they use pieces of code or lessons they learned from previous projects?	❏ Yes ❏ No
Algorithms	● Design non-linear algorithms ● Use wider range of blocks to create program (such a messaging, orange control flow blocks, and changing the number parameter on movement blocks)	**To observe:** Do they use message blocks, repeat loops, or wait blocks in their program? **To ask:** Are they able to tell you what a character is going to do, and what blocks they will need to accomplish that action?	❏ Yes ❏ No
Control Structures	● Successfully use and understand the effect of a repeat loop ● Understand how control is passed around through message blocks	**To observe:** Do they use message and repeat loops? **To ask:** Can they explain what message and repeat blocks do in their program? Are they able to explain what would happen if you took the block out?	❏ Yes ❏ No
Hardware /Software	● Grasp distinction between iPad (hardware) and app (software)	**To observe:** Are they able to open the app on the tablet?	❏ Yes ❏ No

		To ask: Do they know that ScratchJr is an app on the tablet?	
Debugging	• Debug more complex, non-visible bugs • Debug other's code	**To observe:** While building, do they solve errors without assistance? **To ask:** Are they able to explain how they solved a bug after they fixed it?	❏ Yes ❏ No
Representation	• Differentiate among categories of blocks	**To observe:** Are they able to find blocks they are looking for by selecting the proper category? **To ask:** Can they correctly identify/explain the purpose of a block in their program when asked?	❏ Yes ❏ No
Design Process	• Self-initialize the engineering design process	**To observe:** Do they follow their design sheet plan? **To ask:** Do they change the goal of their project at all throughout the process, and if so, are they able to explain why?	❏ Yes ❏ No

5. Teacher Observation Sheet, Part 3: Final Evaluation

When you have completed both part 1 and part 2, use the chart below to evaluate whether your student is a ScratchJr expert.

Powerful Idea/Important Skill	**Checked yes in either Part 1 *or* Part 2?**
Modularity	❏ Yes ❏ No
Algorithms	❏ Yes ❏ No
Control Structures	❏ Yes ❏ No
Hardware /Software	❏ Yes ❏ No
Debugging	❏ Yes

	❑ No
Representation	❑ Yes ❑ No
Design Process	❑ Yes ❑ No
Do they have the motor ability to type/operate a mouse?	❑ Yes ❑ No
Do they have the reading ability to read all the blocks in Scratch?	❑ Yes ❑ No

If you checked "yes" to all of the above categories, then the student displays proficient ScratchJr knowledge, and is ready to move on to Scratch.
Note: If your student does not satisfy all of the categories, there are many resources available for teaching ScratchJr at http://scratchjr.org and at http://sites.tufts.edu/devtech.

You have completed Part 1 of the transition curriculum. If your student satisfies the criteria for ScratchJr expertise, then they are ready to move on to Part 2 of the curriculum, Learning Scratch.

Transition Curriculum, Part 2: Learning Scratch

Required Resources:

- Scratch Project Design Sheet
- ScratchJr to Scratch Blocks Guide
- Computer (to access Scratch)
- Writing Utensil

1. Scratch Project Design Sheet: ask students to complete the project design sheet, planning their project for Scratch, and beginning to learn about the Scratch interface.

Scratch Project Design Sheet

You are going to expand on the story you just built in ScratchJr, but using Scratch instead. Scratch is similar to ScratchJr, but you can do more with it! Before you start planning, go to www.scratch.mit.edu . When you get there, click "create" in the top left corner. Then, on the right side of your screen, click "Getting Started with Scratch," and go through all the steps it tells you to do. Once you have finished, come back to this sheet and plan your project.

In Scratch, you can continue telling your story, adding new characters and places, turn your story into a game, or just retell your story in Scratch. What are you going to do (circle one)?

Add to my story **Make a game out of my story** **Retell my story**

In Scratch, characters are called Sprites. They are very similar to the characters in ScratchJr, but remember that when you see the word "sprite," it really means character.
Take a look at the sprites and backgrounds in Scratch -- there's some new ones that aren't in ScratchJr. Are you going to add any characters or locations? Write or draw your next chapter, game, or retelling that you are going to do in Scratch.

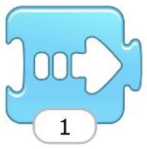
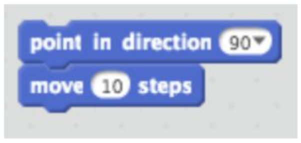
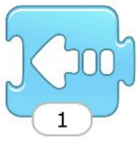
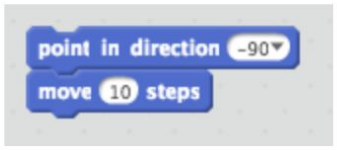
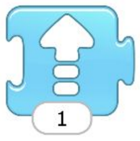
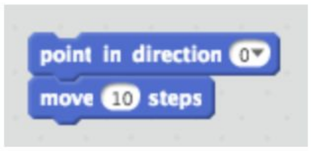
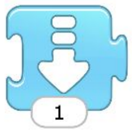
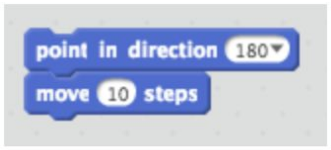
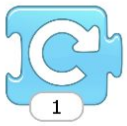
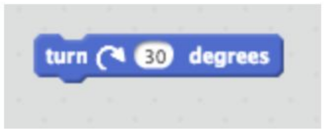
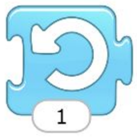
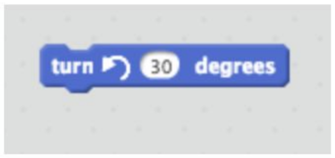
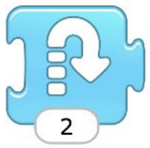
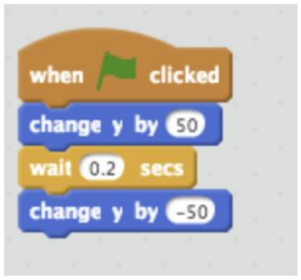
Is there anything your characters need to do that there isn’t a block for? What blocks could you combine to do this?

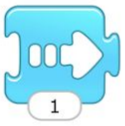
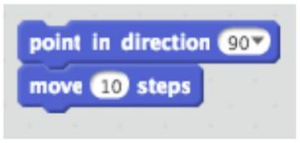
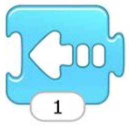
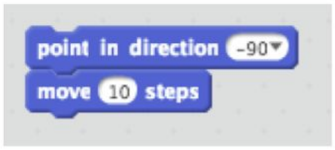
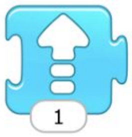
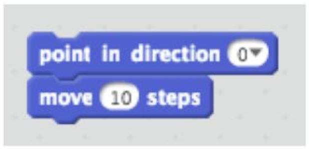
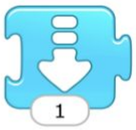
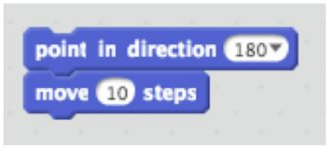
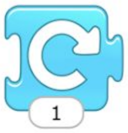
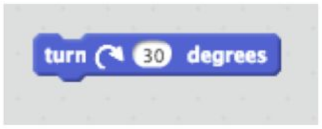
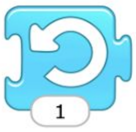
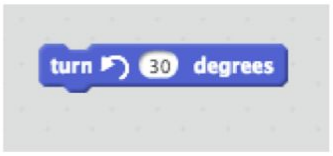
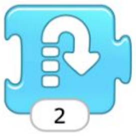
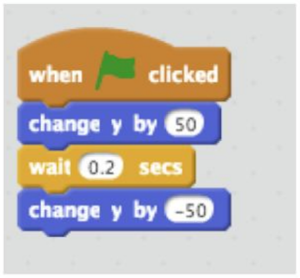
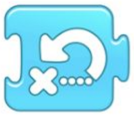
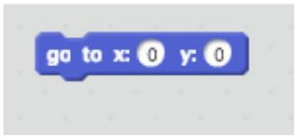
Look at the blocks guide that your teacher has, and mark any blocks that you think you might use. At the bottom of the guide are some ideas for new things you can do in Scratch. Will you use any of those?

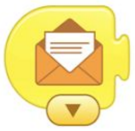
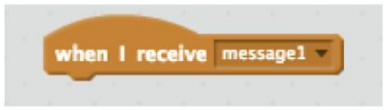
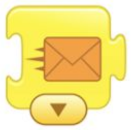
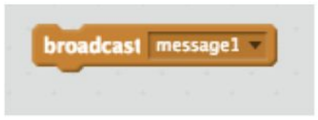
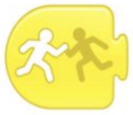
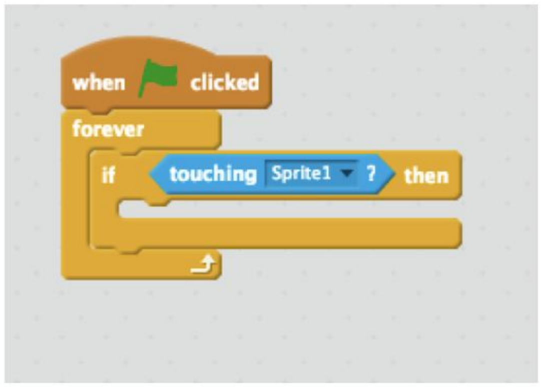
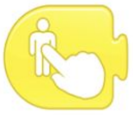
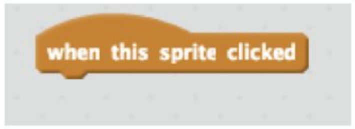
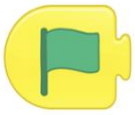
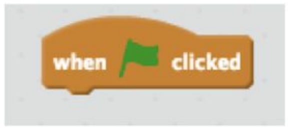
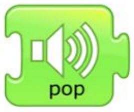
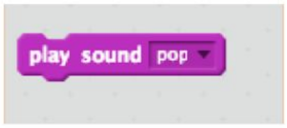
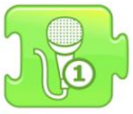
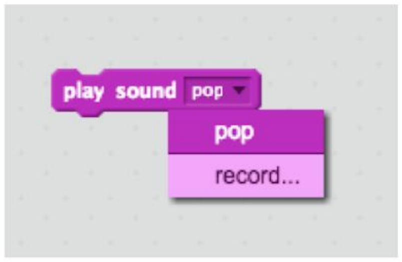
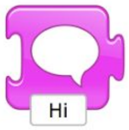
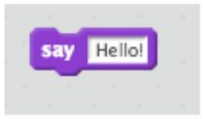
2. Build Scratch Project: Now, have your student begin building the Scratch project that they designed. The blocks guide can be very helpful if they do not know how to do something in Scratch. There are many more resources for teaching Scratch on their website, but this is a good start to help build your student's comfort in Scratch.

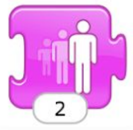
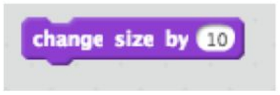
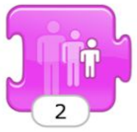
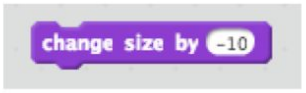
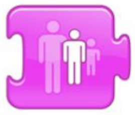
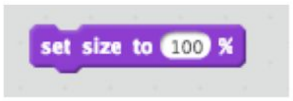
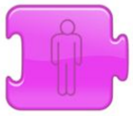
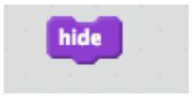
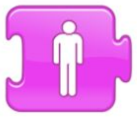
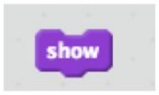
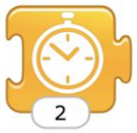
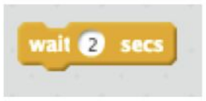
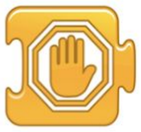
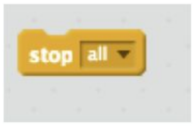
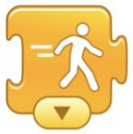
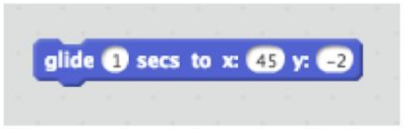
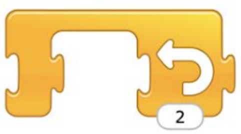
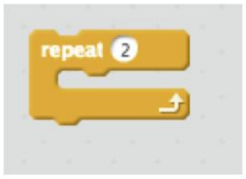
ScratchJr to Scratch Blocks Guide

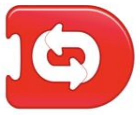
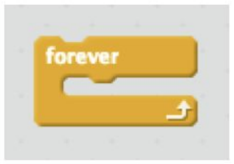
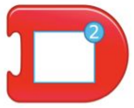
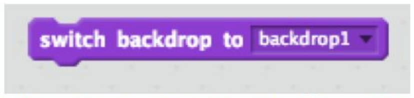
This guide is a reference tool to help ScratchJr experts understand how to accomplish similar goals using Scratch blocks. The first section matches each block in ScratchJr to its most similar companions in Scratch. Students are encouraged to use this section while building in Scratch to help them use their ScratchJr knowledge. The second section includes a few suggestions for new actions that students were not able to do in ScratchJr, but the new blocks in Scratch allow them to do.

ScratchJr Block	Scratch Block(s)
	 point in direction 90▾ move 10 steps
	 point in direction -90▾ move 10 steps
	 point in direction 0▾ move 10 steps
	 point in direction 180▾ move 10 steps
	 turn ↻ 30 degrees
	 turn ↺ 30 degrees
	 when ⚑ clicked change y by 50 wait 0.2 secs change y by -50

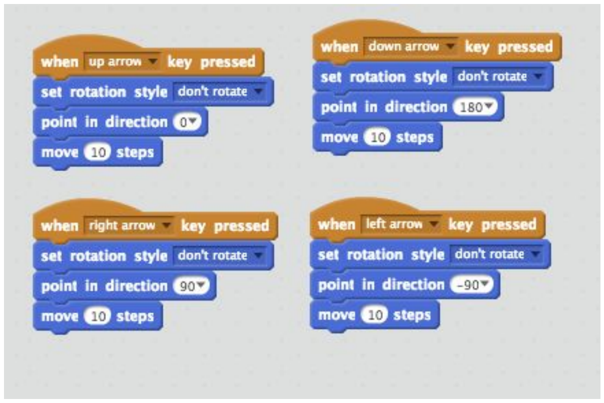
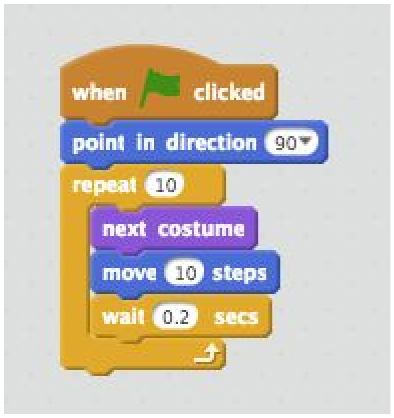
ScratchJr Block	Scratch Block(s)
	 point in direction 90▾ move 10 steps
	 point in direction -90▾ move 10 steps
	 point in direction 0▾ move 10 steps
	 point in direction 180▾ move 10 steps
	 turn ↻ 30 degrees
	 turn ↺ 30 degrees
	 when ⚑ clicked change y by 50 wait 0.2 secs change y by -50
	 go to x: 0 y: 0

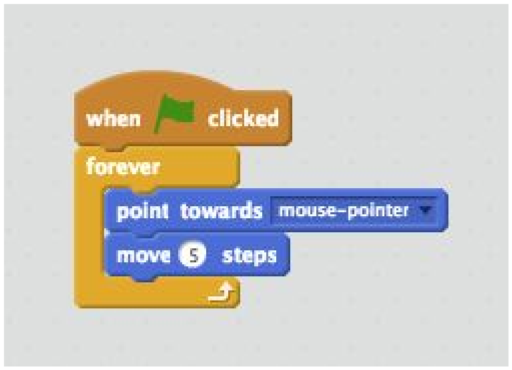
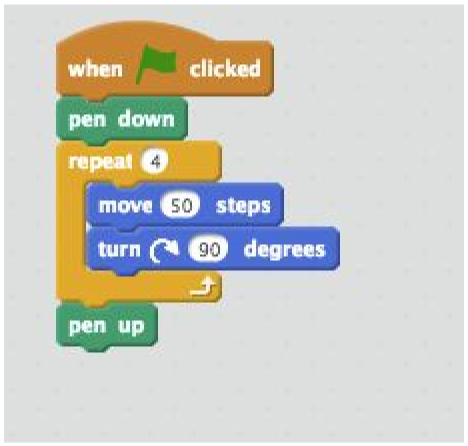
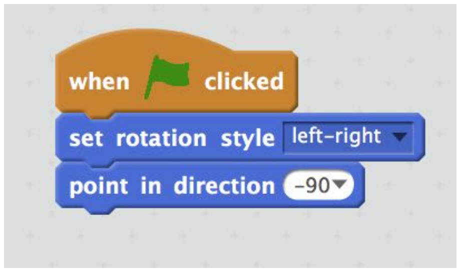
	 when I receive message1
	 broadcast message1
	 when ⚑ clicked forever if touching Sprite1 ? then
	 when this sprite clicked
	 when ⚑ clicked
	 play sound pop
	 play sound pop pop record...
	 say Hello!

	 change size by 10
	 change size by -10
	 set size to 100 %
	 hide
	 show
	 wait 2 secs
	 stop all
	 glide 1 secs to x: 45 y: -2 The x and y values are the location to glide to. Increase the number of seconds to move slower, decrease to move faster.
	 repeat 2

	 forever
	 switch backdrop to backdrop1 There are not pages in Scratch, but you can change the backdrop at any point
	There is not a parallel block, you do not need to use end blocks in Scratch

New Features to Try in Scratch

Feature	Blocks
Move sprites using arrow keys	 when up arrow key pressed / set rotation style don't rotate / point in direction 0 / move 10 steps; when down arrow key pressed / set rotation style don't rotate / point in direction 180 / move 10 steps; when right arrow key pressed / set rotation style don't rotate / point in direction 90 / move 10 steps; when left arrow key pressed / set rotation style don't rotate / point in direction -90 / move 10 steps
Make a sprite look like they are walking to the right by changing costumes	 when clicked / point in direction 90 / repeat 10 / next costume / move 10 steps / wait 0.2 secs For characters with “walking” costumes, look at the “Walking” theme in the sprites menu

Have a sprite follow the mouse around the screen	 `when [flag] clicked` / `forever` / `point towards mouse-pointer` / `move 5 steps` Change the number in the move block to make the character move faster or slower!
Have your sprite draw a square	 `when [flag] clicked` / `pen down` / `repeat 4` / `move 50 steps` / `turn ↻ 90 degrees` / `pen up`
Point your character to the left	 `when [flag] clicked` / `set rotation style left-right` / `point in direction -90`