

Announcements

- The surveys are important, please fill them out. Doing so is worth points.
- Assignment 2 due today ... all assignments are due before class.

Functions – The Punchline

Lawrence Snyder
University of Washington, Seattle

It's BIG!

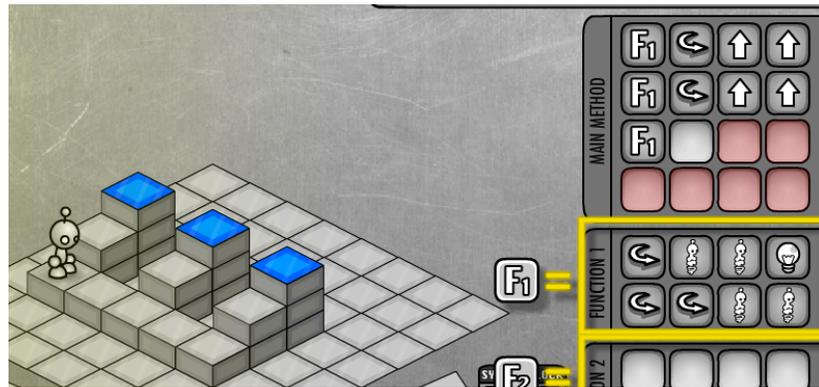
- Functions may seem “obvious” but they are a HUGE idea ...
- They allow us to solve problems by first creating some useful instructions, and then using them to simplify our work
- Let's review how they work ...

Just Do It!

The Function Becomes A Concept

- Because $F_1()$ “processes a riser,” we think of the programming task as

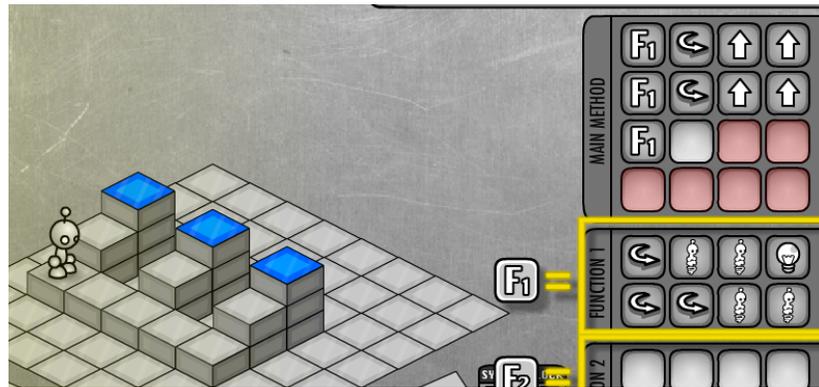
Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$



The Function Becomes A Concept

- Because $F_1()$ “processes a riser,” we think of the programming task as

Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$

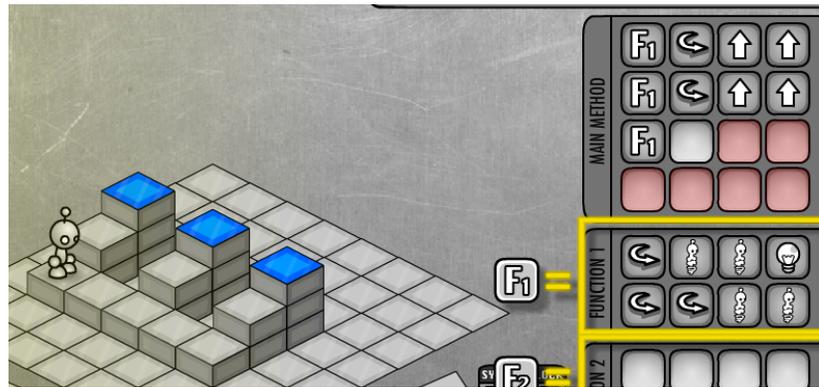


- With $F_1()$, we simplify the programming to just 5 conceptual steps rather than 21

The Function Becomes A Concept

- Because $F_1()$ “processes a riser,” we think of the programming task as

Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$
Move to next riser	
Process a riser()	$F_1()$



- With $F_1()$, we simplify the programming to just 5 conceptual steps rather than 21
- But, WAIT! What is “Move to next riser”?
 - It’s a concept ... make it a function!
 - `Move_to_next_riser()`

A Five Instruction Program

Is this beautiful, or what?

The image shows a game interface with a robot on a grid of blocks. The robot is positioned on a 5x5 grid of blocks, with a stack of 3 blue blocks on the left and a stack of 2 blue blocks on the right. The robot is facing right. The interface includes a top toolbar with icons for movement, rotation, and function calls (F1, F2). A yellow box highlights the function editor, which contains a grid of function calls (F1, F2) and a 'RUN' button. The function editor is divided into 'MAIN METHOD', 'FUNCTION 1', and 'FUNCTION 2'. 'FUNCTION 1' contains a sequence of icons: a left arrow, a lightbulb, a vertical line with a dot, and a right arrow. 'FUNCTION 2' contains a left arrow, an up arrow, and another up arrow. The interface also shows a 'CMDS' counter at 15, a 'CALLS' counter at 0, and a 'CLEAR COMMANDS' button. A yellow box highlights the text 'Program Is Only Function Calls' and 'Process_R' next to an F1 icon, and 'Move_2_N_R' next to an F2 icon. A yellow box also highlights the text 'Is this beautiful, or what?'.

Program Is Only Function Calls

Process_R

Move_2_N_R

MAIN METHOD

FUNCTION 1

FUNCTION 2

RUN

CMDS 15 CALLS 0 CLEAR COMMANDS

← MENU WALKTHROUGH ?

Abstraction ...

- Formulating blocks of computation as a “concept” is **functional abstraction** [A better definition in a moment]
- What we did just now is important ...
 - We spotted a coherent (to us) part of the task
 - We solved it using a sequence of instructions
 - We put the solution into a function “package”, gave it a name, “process a riser,” and thus created a new thing, a concept, something we can talk about & use
 - Then we used it to solve something more complicated ... and then we did it again!

Abstracting

- Collecting operations together and giving them a name is *functional abstraction*
 - The operations perform a coherent activity or action – they become a *concept* in our thinking
 - The operations accomplish a goal that is useful – and typically – is needed over and over again
 - *Functions* implement functional abstraction: 3 parts
 - A name
 - A definition (instruction seq), frequently called a “body”
 - Parameters –stuff inside the parentheses, covered later

People Abstract All The Time

- Functional abstractions in which you are the agent, but someone taught you:
 - Parallel parking
 - Backstroke in swimming
- Functional abstractions you recognized and in which you are the agent
 - Doing a load of laundry
 - Making your favorite {sandwich, pizza, cookies, ...}
- Others?

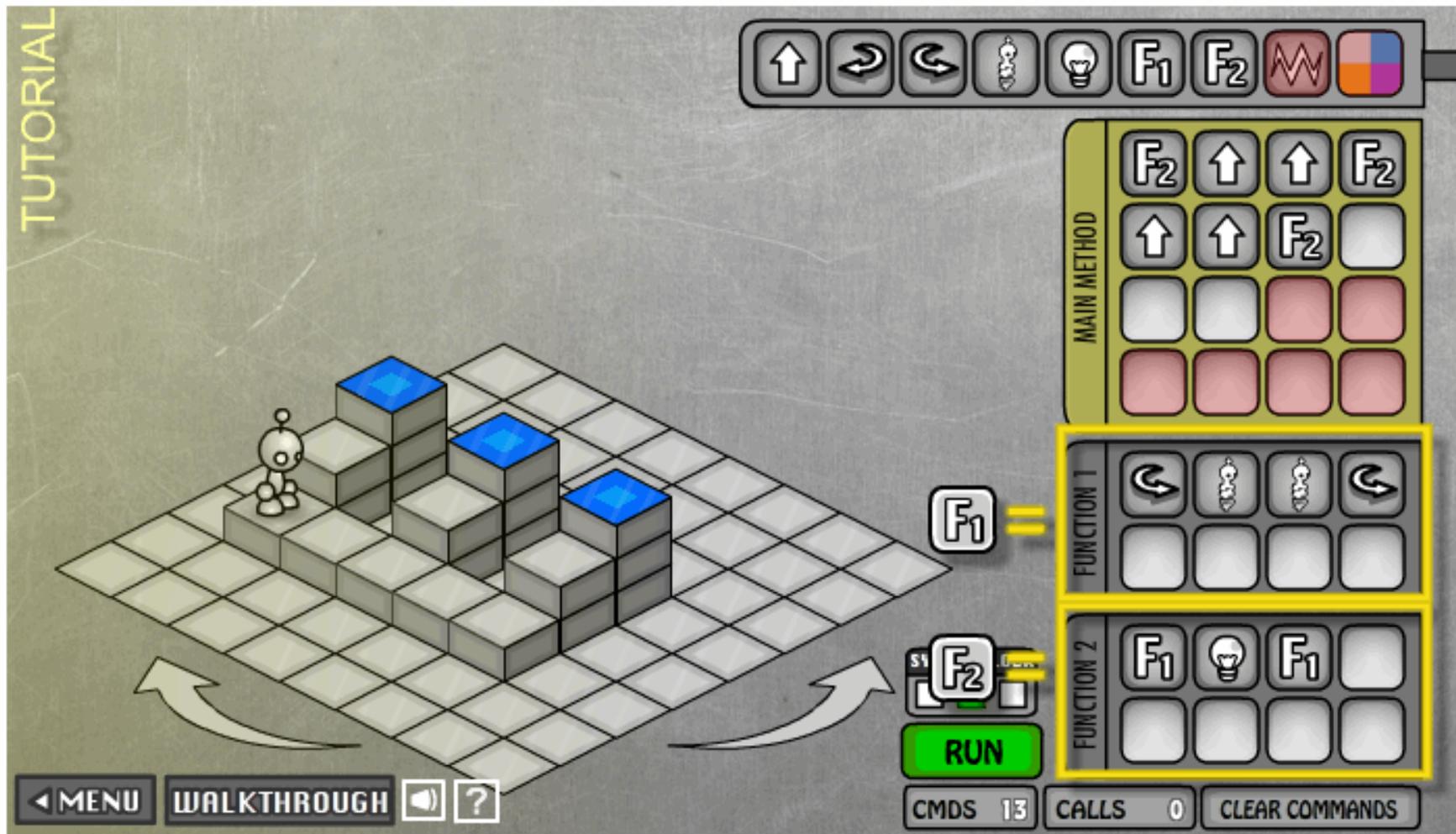
No “Correct” Way To Abstract

- We have abstracted “process a riser” and “move to the next riser” as components of a solution
- As concepts, they are packaged into functions
- Maybe you thought of this in a different way
- That is, there can be other “coherent” parts of a solution

Just Do It!

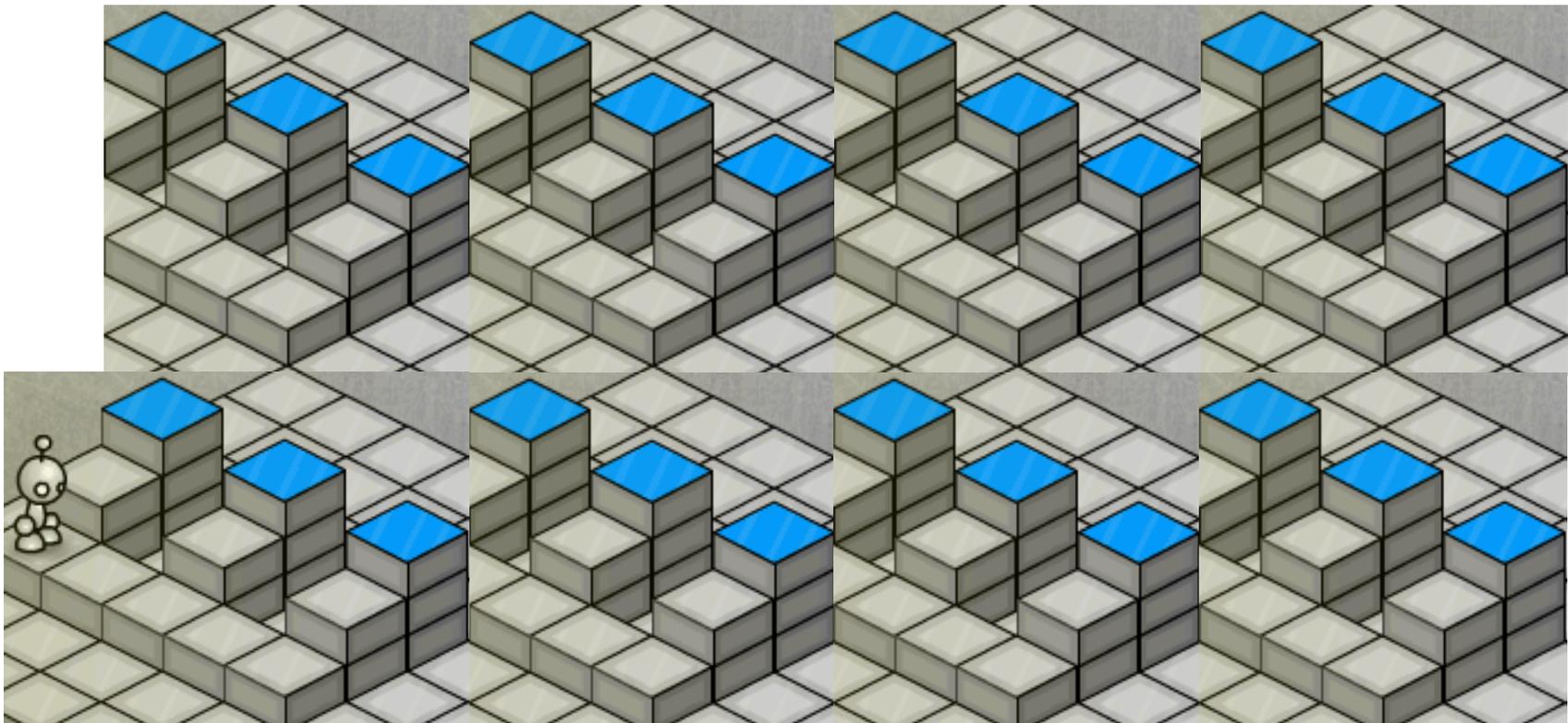
The Function Is Just The Packaging

- Another way to use a function for the risers



Keep On Using Abstraction ...

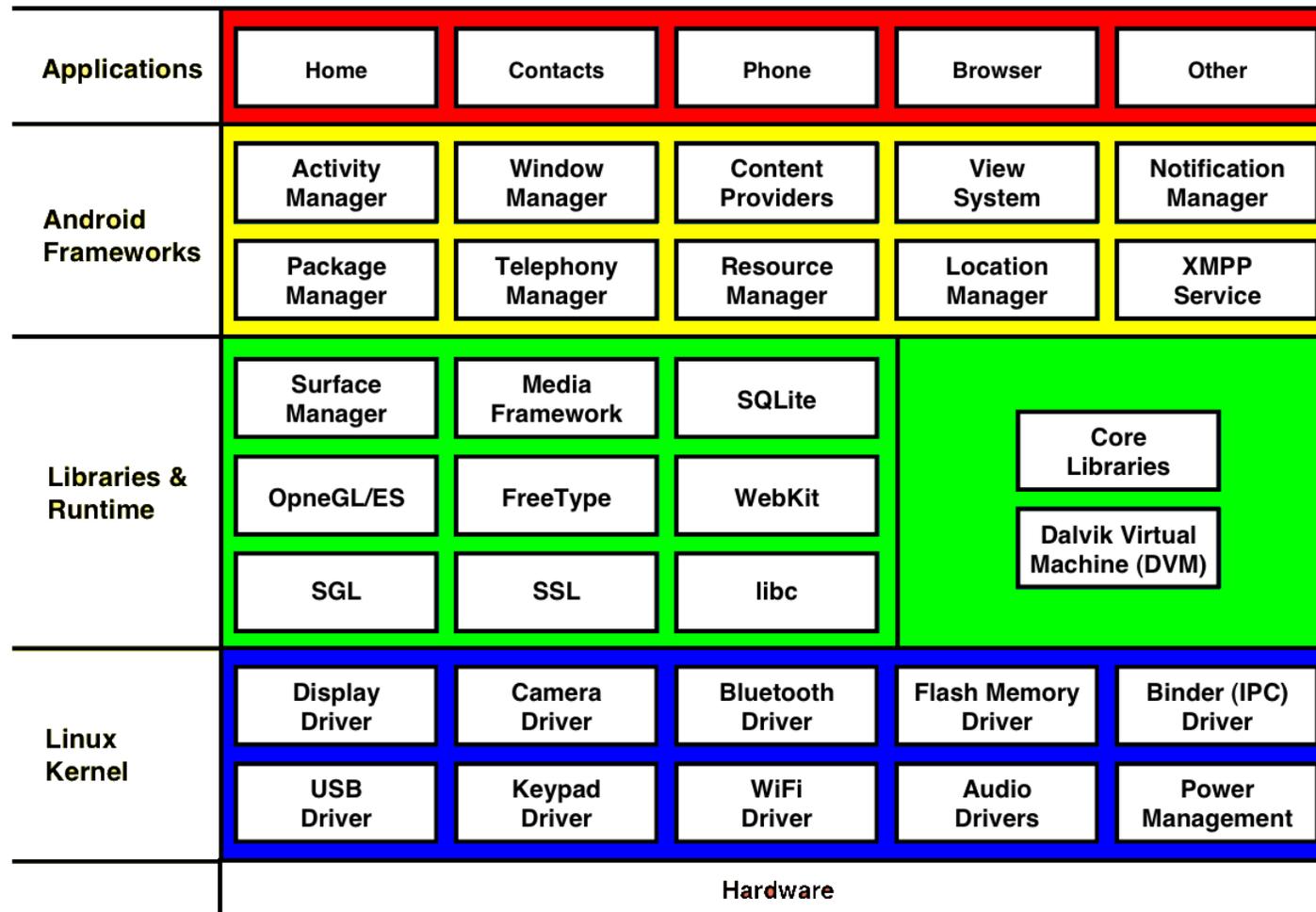
- If M.C. Escher handed us a problem ... what would we do?



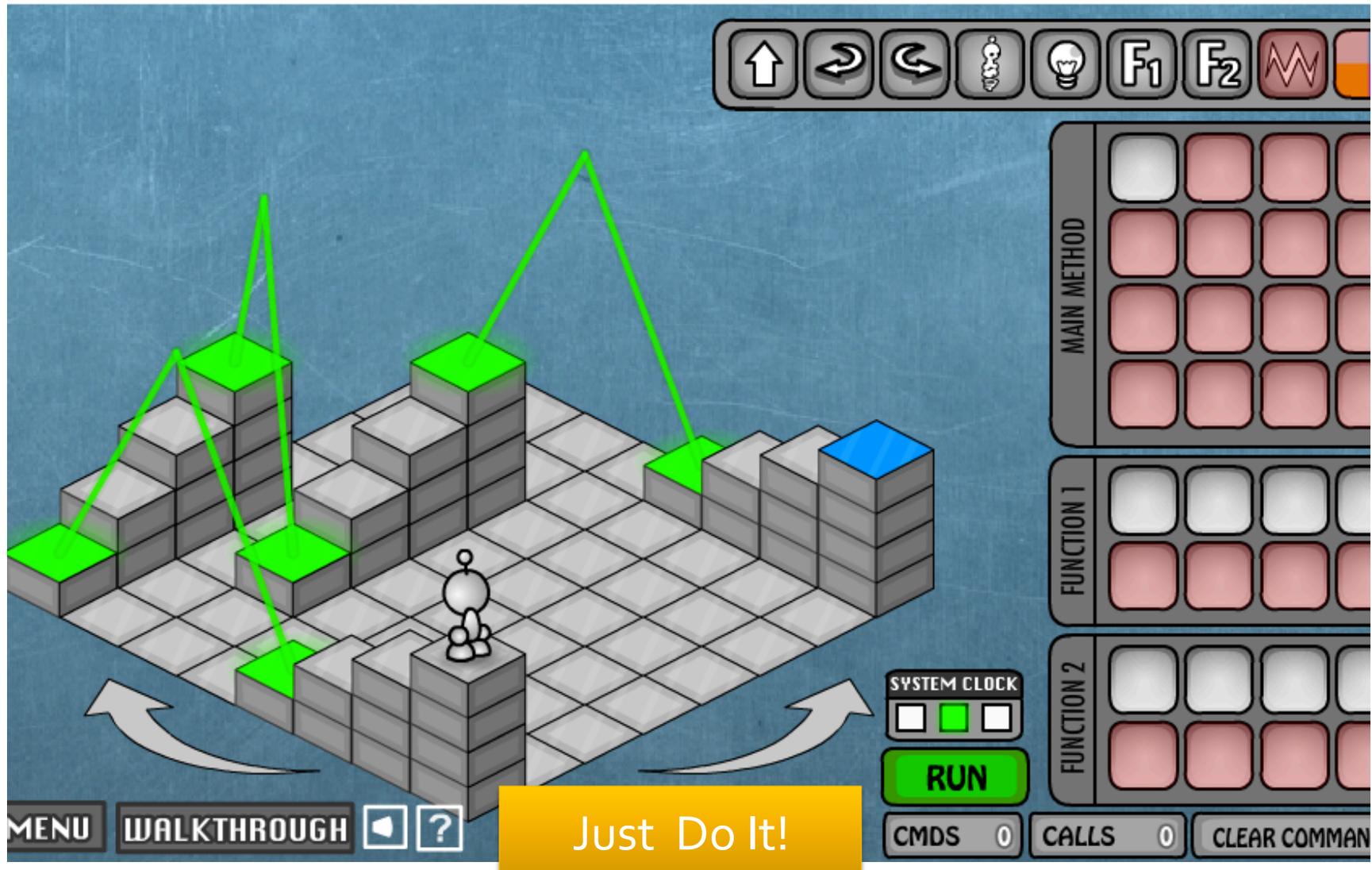
It only simplifies our **thinking**; the bot still does all the work

How Useful Is This Idea

- Say "Hi" to Android's Software Stack



And, By Request ...



Functions – They're Big

- Functions – the packages of computation – will be used everyday in this course
- Functional Abstraction – the process of spotting a concept, “packaging” it as a function (at least in your own mind) and using it to solve some tougher problem – will be used everyday for the rest of your life!