

JUSTICE-CENTERED EDUCATIONAL PROGRAMMING LANGUAGES

AMY J. KO

Professor

The Information School
University of Washington



ME AT 15, A
CLOSETED
TRANS GIRL,
HIDING FROM
MYSELF IN
DIGITAL
WORLDS







Credit: Associated Press

PROGRAMMING LANGUAGES (PL)

are built by and for the few, rather than for everyone.

STUDENTS WITH DISABILITIES

can't read/write code, access content, because we build PL tools that require pointing, sight

ENGLISH LANGUAGE LEARNERS

can't learn because they're forced to learn English before or while learning PL

MULTILINGUAL STUDENTS

STUDENTS WITHOUT DEVICES

can't enroll in CS classes that use PL that require these things to practice at home

STUDENTS WITHOUT INTERNET

have to leave their identities behind, trading them for capitalist ideals of efficiency, domination, and extraction woven into PL

EVERYONE BUT WHITE BOYS

have to wrangle PL, tools, and docs that demand particular kinds of communication and attention.

NEURODIVERGENT STUDENTS

WHAT WOULD
IT MEAN TO
DESIGN
EDUCATIONAL
PL FOR ALL?

THIS TALK

- ▶ I'll share:
 - ▶ My background and positionality
 - ▶ A brief review of conceptions of justice
 - ▶ Seven justice-centered requirements for educational PL, with bad, good, and aspirational examples (including my lab's work on Wordplay, a new PL).
- ▶ You'll leave with:
 - ▶ A **novel argument** about the relationship between PL design and justice.
 - ▶ Questions and possible answers about how to advance justice in PL design



WHO AM I TO SPEAK ON THIS? **Positionality**

- ▶ Background in CS + Psychology + Design
- ▶ Professionally privileged Professor
- ▶ Marginalized by race, gender, politics
- ▶ I design and build programming languages
- ▶ I study learning about computing
- ▶ I work with teachers, schools, community groups, and marginalized

**WHAT DOES “BETTER”
FOR “EVERYONE” MEAN?**

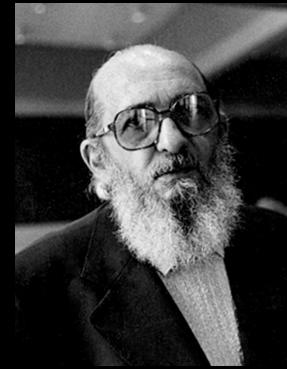
RAWLSIAN JUSTICE



- ▶ John Rawls' seminal *A Theory of Justice* (1971) defines justice through two principles:
 - ▶ Every person deserves a claim to the same set of equal basic liberties. (i.e., *there should be no “birthright” to greater freedom*).
 - ▶ Any social inequalities must satisfy two conditions:
 - ▶ They must stem solely from equality of opportunity (*not birthright*)
 - ▶ They must be to the greatest benefit of the least advantaged (*addressing inequities inherent to birth*).

EDUCATIONAL JUSTICE

- ▶ Paulo Freire (*Pedagogy of the Oppressed*)



- ▶ Rejected school as a context for “depositing” knowledge in minds
- ▶ Viewed education explicitly for fostering liberatory, collective, critical consciousness about learners’ “limiting situations”, through dialog, mutual understanding

- ▶ bell hooks (*Teaching to Transgress*)



- ▶ Freire’s ideas, in practice, are constrained by racial and patriarchal capitalism that Freire overlooked. These social and economic hierarchies limit what dialog students will engage
- ▶ hooks advocated for school to be a place to see these forces, connect them to students’ lived experiences, and organize around dismantling them

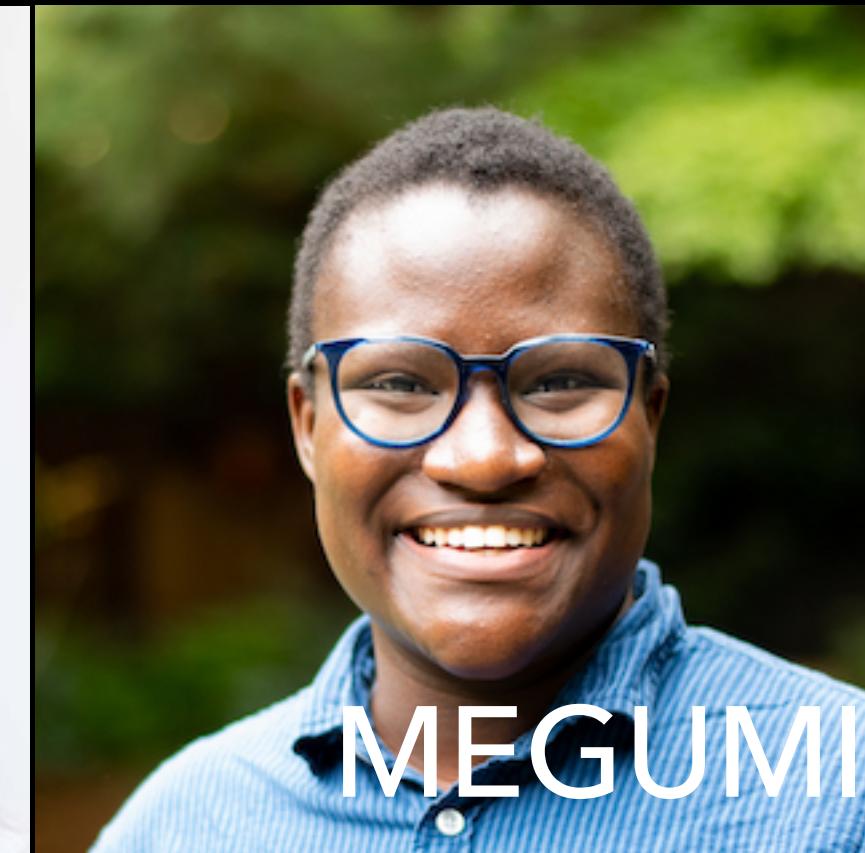
DESIGN JUSTICE



- ▶ Sasha Costanza-Chock (*Design Justice*) applies these many notions of justice to **design**, centering design choices at the margins, in communities:
 - ▶ Heal and empower communities
 - ▶ Center direct stakeholder voices
 - ▶ Prioritize community impact over design intent
 - ▶ View partnership as ongoing collaboration
 - ▶ Frame designers as facilitators not deciders
 - ▶ Value stakeholders' lived experiences
 - ▶ Share design knowledge with communities
 - ▶ Work toward community-led, sustainable outcomes
 - ▶ Reconnect communities rather than exploit them
 - ▶ Designers should understand a communities existing solutions before building new ones

WHAT DOES ANY OF THIS MEAN FOR EDUCATIONAL PL DESIGN?

- ▶ I worked with my colleague **R. Ben Shapiro** and my doctoral students **Jayne Everson** and **Megumi Kivuva** to translate these ideas of justice and our joint lived experience teaching computing into **design requirements** that we think best address injustices in current PL design for education.



7 JUSTICE-CENTERED REQUIREMENTS FOR EDUCATIONAL PROGRAMMING LANGUAGES

OUR APPROACH

- ▶ Costanza-Chock's community design principles were our starting point.
- ▶ From there, we examined the intersections between those principles, the spectrum of marginalization in education mapped by education justice researchers, and the design choices inherent to educational PL.
- ▶ This led to **7 design requirements** for educational PL. Meeting them means meeting the many principles of justice we just discussed.

ALTCODE — A TRAGICOMIC MNEMONIC

- ▶ **Accessible** – empower all abilities
- ▶ **Liberatory** – see computing for what it is, good and bad
- ▶ **Transparent** – comprehensible, inspectable computation
- ▶ **Cultural** – center learners' communities, values, languages
- ▶ **Obtainable** – free and feasible to access and use
- ▶ **Democratic** – shaped by youth and teachers
- ▶ **Enduring** – lasting and sustainable, as long as it is needed

CAVEATS

- ▶ There are 7, but that is not a magic number
- ▶ We don't claim this is the *only* "right" notion of justice – conceptions of justice evolve over time, and we don't represent all voices
- ▶ We *do* claim that if these requirements were met, there would be *many* more people globally who would be able to learn what programming languages are, how to use them, and possibly use them for problems in their community that no big tech company ever would.

REQUIREMENTS

EXAMPLES

- ▶ Throughout, I'll critique PL for their strengths and weakness
- ▶ I'll also include examples from **Wordplay**, our attempt at making one example of a justice-centered educational programming language. Not because Wordplay is perfect or best, but just because it tries new things others haven't.

Amy J. Ko, Carlos Aldana Lira, Isabel Amaya (2025). Wordplay: Accessible, Multilingual Interactive Typography . ACM SIGCHI Conference on Human Factors in Computing Systems (CHI) <https://doi.org/10.1145/3706598.3713196>

Wordplay is in **beta**, so it might not work as intended or be complete. You can still try it out and share ideas in [GitHub](#), see our [1.0 plans](#), and contribute to the [repository](#).



Wordplay

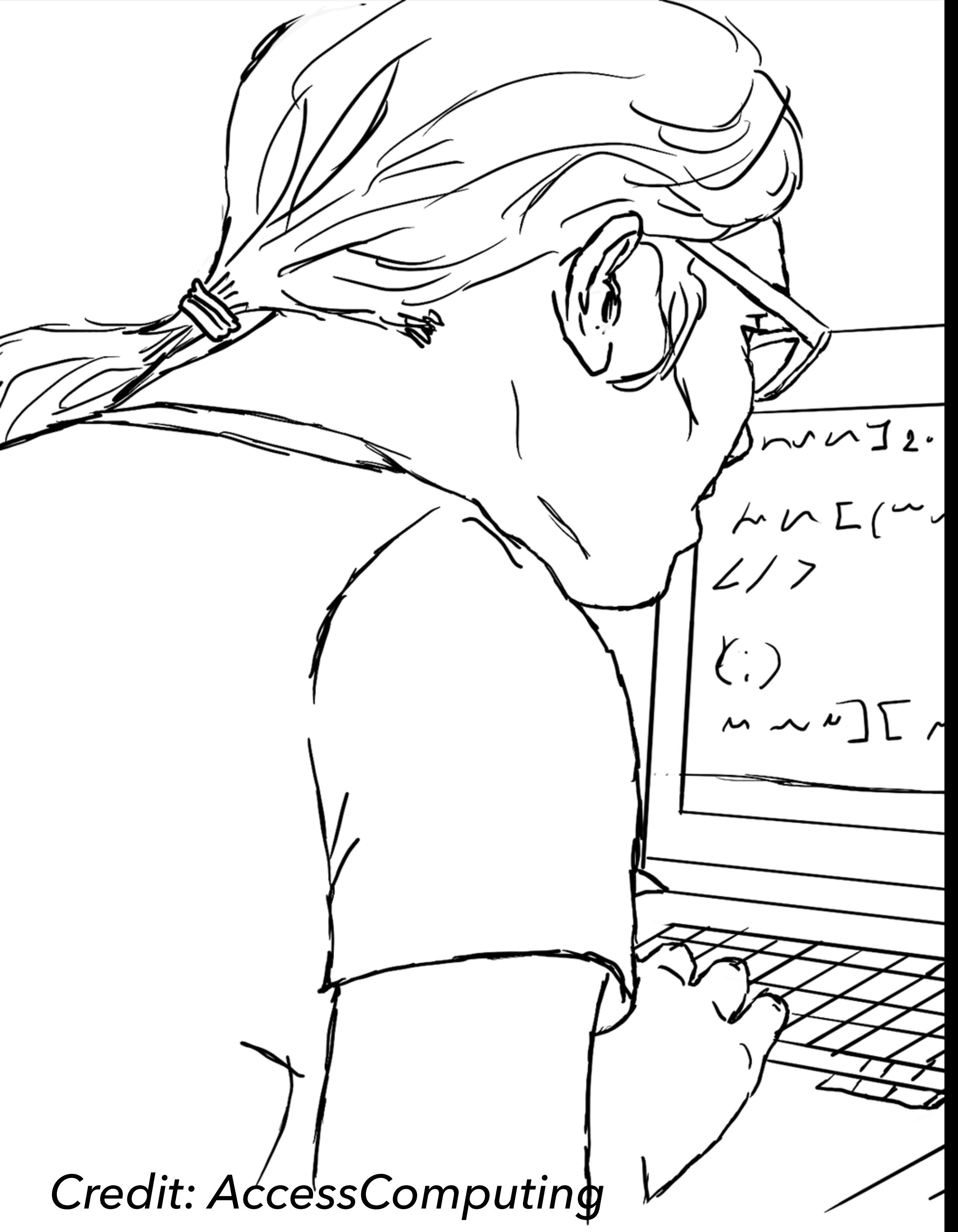


Create interactive stories with words, symbols, emojis, and code.

Wordplay is programming language that enables you to:

- Playfully animate words and emojis 😊
- Use time ⏳, sound 🎵, websites 🔗, and physics 🌎
- Share 🤝 with friends, groups, or anyone
- Code in any world language 🌐
- Edit with mice 🠰, touch 🤚, and keyboards 🖱️
- Debug forwards ➡️ and backwards ➡️
- View with screens 🖥️ and screen readers 🎧

Free forever from the [University of Washington](#).



Credit: AccessComputing

SUPPORT ALL ABILITIES

ACCESSIBLE

THE REQUIREMENT

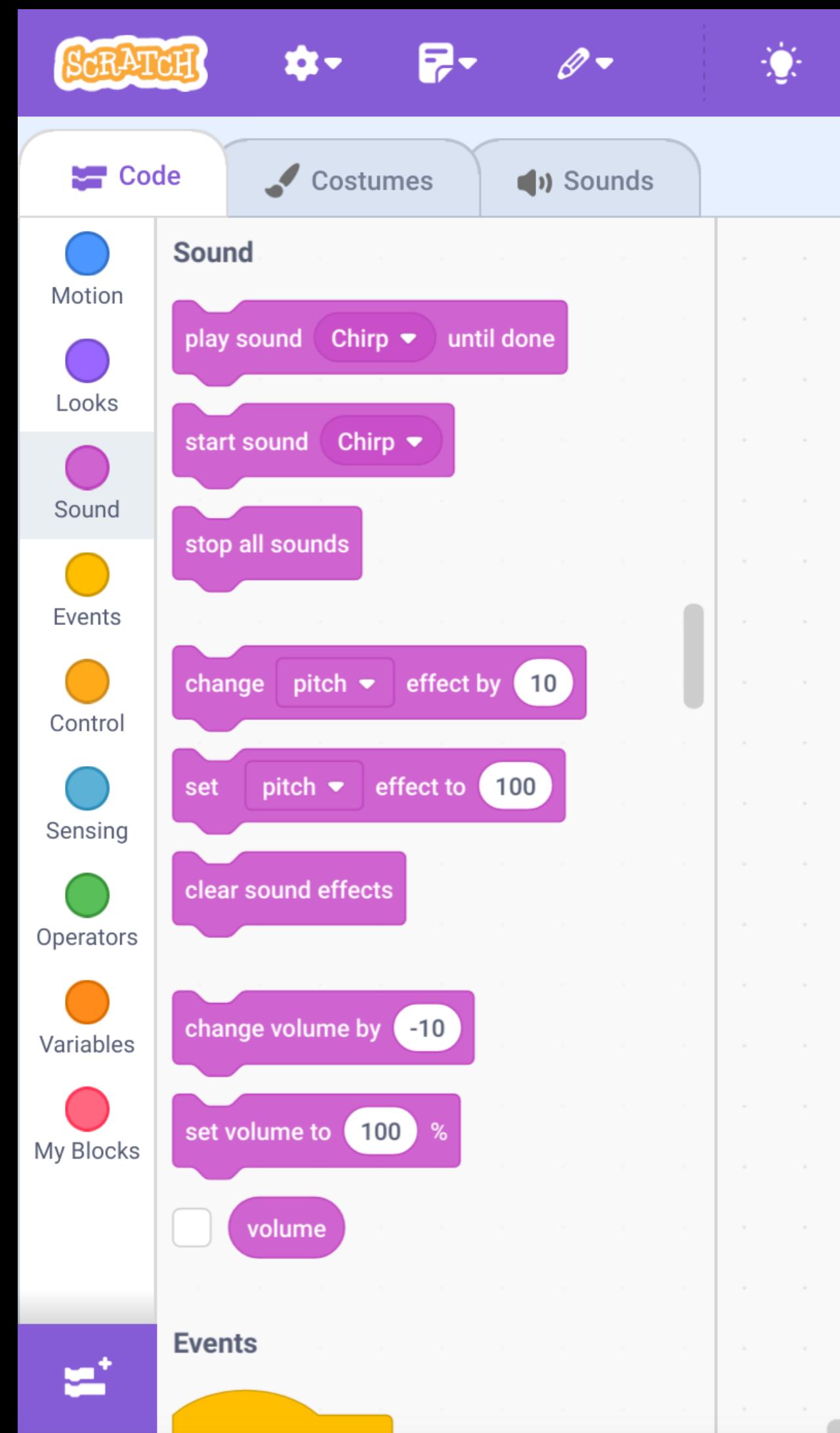
- ▶ *Learners and teachers must be able to use the full functionality of an educational programming language with **whatever input** they can provide and **whatever output** they can perceive and comprehend*
- ▶ In practice, this means:
 - ▶ Not just mice and keyboards, but speech, Braille keyboards and displays, switches, gaze
 - ▶ Not just perceptual and motorphysical, but also diversity in **reading** abilities, **learning**, **attention**, **sensory processing**, and more.

WHY?

- ▶ **Disability justice:** all people deserve the right to participate in our computational worlds, independent of what abilities they were born with, lost, gained.
- ▶ The world we have is designed for sighted, hearing, healthy, people.
- ▶ The world *should* be designed in a way that eliminates this assumption, working for everyone.

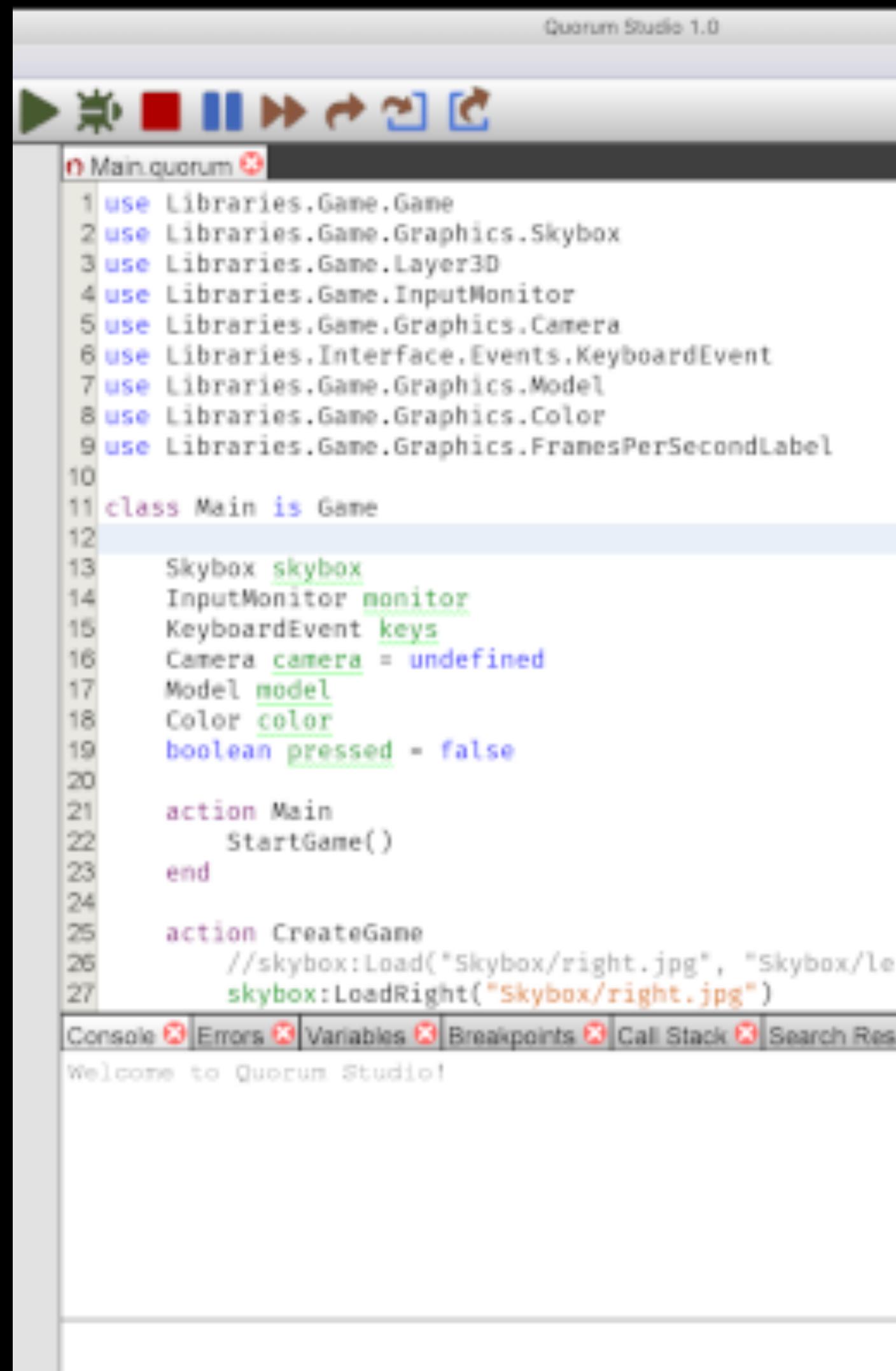
BAD: SCRATCH

- ▶ Scratch requires the use of a **pointer** (mouse or touch).
- ▶ This excludes anyone who cannot use a pointer. It's success at popularizing the structured code editors of the 1980's, and the drag and drop paradigm of Alice of the 2000's, has meant a proliferation of "block-based languages" that blind learners cannot use, that learners with motor tremors cannot use, that quadriplegic learners cannot use.
- ▶ Advocacy to the Scratch team has led to little change in Scratch's accessibility, despite multiple opportunities during rewrites and redesigns over the past 20 years.



BETTER: QUORUM

- ▶ Quorum's language is designed to be highly screen readable for learners who are blind or dyslexic, and rely on screen readers.
- ▶ It also offers screen readable output of 2D and 3D graphics.
- ▶ It has been widely adopted in schools for the blind as it is the only screen readable language, IDE, and platform that works and isn't designed for professional developers.



The screenshot shows the Quorum Studio 1.0 interface. The top bar displays the title "Quorum Studio 1.0". Below the title is a toolbar with various icons for file operations like Open, Save, and Print. The main window contains a code editor with the following Quorum script:

```
1 use Libraries.Game.Game
2 use Libraries.Game.Graphics.Skybox
3 use Libraries.Game.Layer3D
4 use Libraries.Game.InputMonitor
5 use Libraries.Game.Graphics.Camera
6 use Libraries.Interface.Events.KeyboardEvent
7 use Libraries.Game.Graphics.Model
8 use Libraries.Game.Graphics.Color
9 use Libraries.Game.Graphics.FramesPerSecondLabel
10
11 class Main is Game
12
13     Skybox skybox
14     InputMonitor monitor
15     KeyboardEvent keys
16     Camera camera = undefined
17     Model model
18     Color color
19     boolean pressed = false
20
21     action Main
22         StartGame()
23     end
24
25     action CreateGame
26         //skybox:Load("Skybox/right.jpg", "Skybox/left.jpg")
27         skybox:LoadRight("Skybox/right.jpg")
```

At the bottom of the code editor, there are tabs for "Console", "Errors", "Variables", "Breakpoints", "Call Stack", and "Search Results", all of which have red X marks indicating they are not currently active. A welcome message "Welcome to Quorum Studio!" is displayed at the bottom of the editor area.

WORDPLAY: ALL ABILITIES

- ▶ A multi-modal, WCAG compliant editor that supports text editing, block editing, menu editing
 - ▶ Future work on speech-based editing.
 - ▶ WCAG-compliant program output that comes for free.
 - ▶ API's that require multilingual descriptions of visual content (e.g., font faces).

RainingLetters

```
..... f(letter • Letter)
..... letter.y > 0m ? (letter.y • letter.y - 0.5m).angle: letter.angle + let-
→ → → → (letter.y • size + 2m).visible: ⚡ (0 • amps) > 50
....)

`Convert the letters into phrases`
```

🌐 (

```
.... [
..... ☀ ("cloud" • size: 3m • place: ⚡ (-8m • 12m))
..... ☀ ("cloud" • size: 5m • place: ⚡ (8m • 11m))
..... ☀ ("diagonal squares" • place: ⚡ (0m • 0m))
.... ] .+
.... letters.translate(
..... f(letter • Letter) .
..... .... ☀ (
..... .... letter.letter •
..... .... size: 1m
..... .... place: ⚡ (letter.x • letter.y)
..... .... rotation: letter.angle • 1°
..... .... rest: ⚡ (
..... .... opacity: letter.visible ? 1 • 0 •
..... .... color: ☀ (100% • 0 • 0 °)
..... .... )
..... )
.... )
..... place: ⚡ (0m • -2 • size • -5m) • background: ☀ (63% • 93 • 248 °)
)
```

✖ before evaluation close)

OPEN QUESTIONS

- ▶ Few PL are designed *with* learners with disabilities around the things they might want to make
- ▶ What would a gaze, sound and movement-based IDE for making purely gaze, sound, and movement-based apps be like?
- ▶ How can code editors seamlessly integrate speech and audio feedback?
- ▶ How can program output of all kinds be made accessible?
- ▶ How can PL be designed to make it easier to make software itself more accessible?



Credit: NY Times

FOSTER CRITICAL CONSCIOUSNESS

LIBERATORY

THE REQUIREMENT

- ▶ *Educational PL must empower learners with new conceptions of the natural, social, and artificial worlds, enabling them to imagine futures of computing that dismantle racial, patriarchal capitalism, and colonialism.*
- ▶ In practice, this means:
 - ▶ Centering the reality that computing is both amazing and powerful, but also kills, harms, marginalizes, and disempowers.
 - ▶ Making space in PL design, tools, tutorials and communities for the inherently political nature of computing.

WHY?

- ▶ **Critical consciousness** (Freire, hooks). To have a just world, everyone must understand how and why it is unjust in relation to their lived experiences, so we can fix it together.
- ▶ That includes the **computational** world, and PL are key media that shape the our computational worlds.

BAD: CODE COMBAT

- ▶ A for profit platform that centers war, violence, "*the feeling of wizardly power at their fingertips by using typed code*", and learners as factory workers producing more than "*1 billion lines of code*"
- ▶ Erases the reality that code is literally a tool of war, used to more efficiently kill people at scale, to silence resistance to dictators, etc.



CodeCombat hides the limitations of computation behind stories of profit, domination, and xenophobia.

BETTER: GIDGET

- ▶ It's not the most political of PL, but it does frame robots and computers as ***fallible, ignorant, but reliable tools***
- ▶ This framing is used throughout the game to show learners that machine intelligence is limited and largely stems from human intelligence, demystifying code as "magic".



Gidget conveys its fallibility.

WORDPLAY: LIBERATORY

- ▶ Language constructs are anthropomorphized with personalities and relationships with each other than center the limited and narrow views with which they conceive the world.
- ▶ Learners are positioned as the only ones of overcoming these limitations, by understanding the nuances of human experience fully.

I think I'm supposed to make decisions? Like this?

```
number: · -100  
number · < · 0 · ? · 'negative' · 'positive'
```

"negative"

But have you ever thought about how we decide?

Doesn't it seem like decisions should be more nuanced than just yes or no? Is deciding between `T` and `⊥` all there is?

Aren't you worried that if these are the only kind of decisions we can make, we'll be missing some important context about the world?

Documentation for the conditional expression, in which it expresses uncertainty about the expressibility of binary decision making.

OPEN QUESTIONS

- ▶ Can programming language syntax and semantics be sociopolitical? How?
- ▶ What are the opportunities and limits of PL themselves promoting learners' critical consciousness about the good and bad of computing in society?
- ▶ How might liberatory PL be resisted by schools, governments, and parents who do not want youth to know about computing's dark side? Are there ways that PL can be subversively political?



Stolen art from ChatGPT

MAKE CODE COMPREHENSIBLE

TRANSPARENT

THE REQUIREMENT

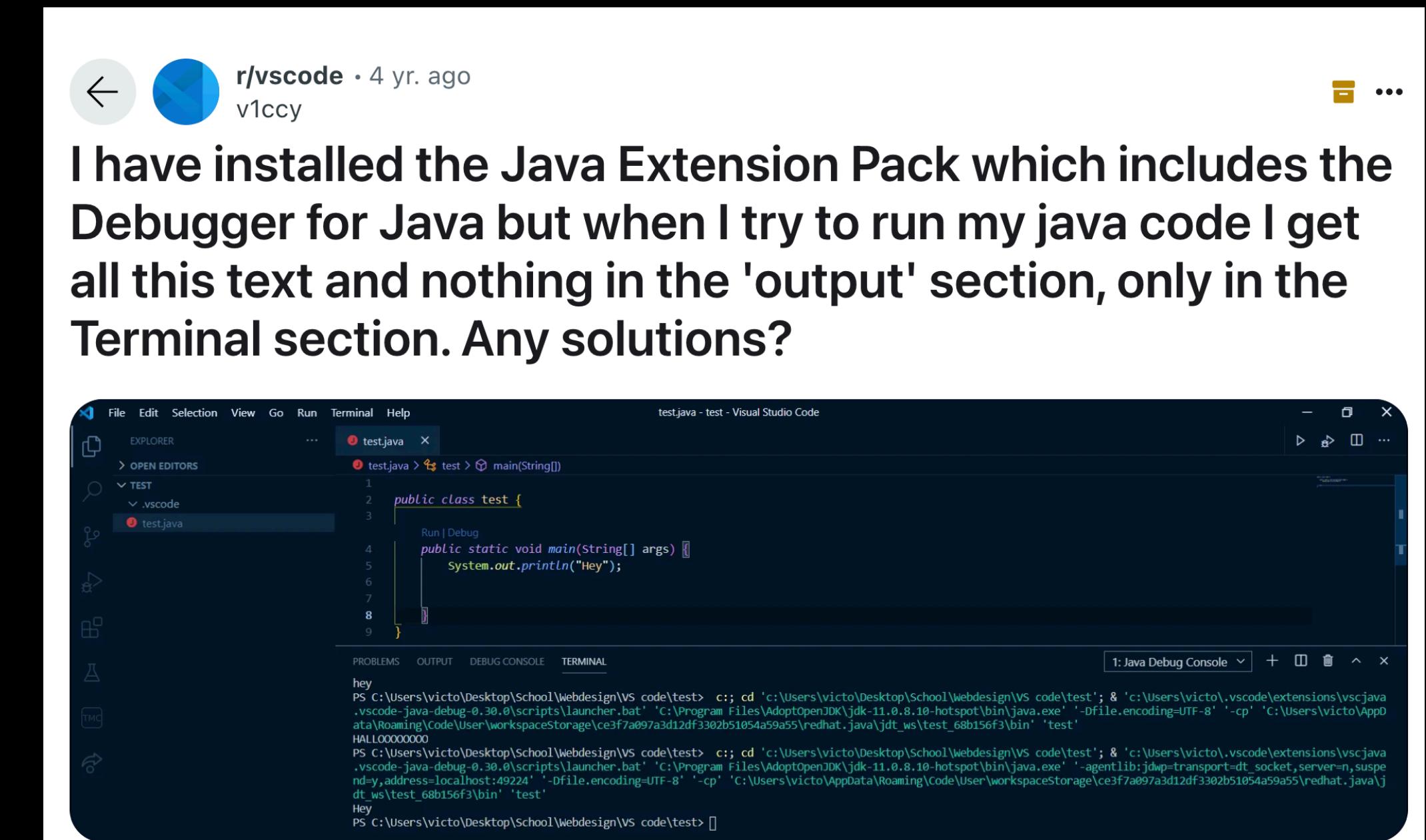
- ▶ *To foster youth agency via program comprehension, program execution must be navigable in both directions and at multiple levels of granularity.*
- ▶ This requirement is essential to **agency**: learners must feel they understand and have control over program behavior, rather than it controlling them.
- ▶ In practice, this means:
 - ▶ Flexible, accessible control over the speed and direction of a program's execution
 - ▶ Explanations of program execution that enable youth to understand what programs do, how they do them, demystifying them

WHY?

- ▶ One cannot critique, control, or reimagine something if one does not know what it is or how it works. Bourdieu described understanding of our institutions and social worlds as central to liberation from “symbolic domination”.
- ▶ The incomprehensibility of code is our field’s symbolic domination; it has for too long enriched and empowered a small, elite group – you and I – at everyone else’s expense.
- ▶ Centering comprehensibility, and transparency of software behavior more broadly, is central to **agency**.

BAD: NEARLY ALL PROFESSIONAL PROGRAMMING LANGUAGES

- ▶ Everything except for print statement requires complex configuration, poor control over execution, no reversibility.
- ▶ This poor support for transparency of execution means learners who try to comprehend programs in these languages struggle far more to understand what code is doing.



File Explorer
OPEN EDITORS
TEST
.vscode
test.java

Run | Debug
public static void main(String[] args) {
 System.out.println("Hey");
}

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

hey
PS C:\Users\victo\Desktop\School\Webdesign\VS Code\test> c:; cd 'c:\Users\victo\Desktop\School\Webdesign\VS Code\test'; & 'c:\Users\victo\.vscode\extensions\vscodejava.vscod...
vscod...-java-debug-0.30.0\scripts\launcher.bat' 'C:\Program Files\AdoptOpenJDK\jdk-11.0.8.10-hotspot\bin\java.exe' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\victo\AppData\Roaming\Code\User\workspaceStorage\ce3f7ae973d12df3302b51054a59a55\redhat.java\jdt_ws\test_68b156f3\bin' 'test'
HALLOOOOOOO
PS C:\Users\victo\Desktop\School\Webdesign\VS Code\test> c:; cd 'c:\Users\victo\Desktop\School\Webdesign\VS Code\test'; & 'c:\Users\victo\.vscode\extensions\vscodejava.vscod...
vscod...-java-debug-0.30.0\scripts\launcher.bat' 'C:\Program Files\AdoptOpenJDK\jdk-11.0.8.10-hotspot\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspe...
ndy,address=localhost:49224' '-Dfile.encoding=UTF-8' '-cp' 'C:\Users\victo\AppData\Roaming\Code\User\workspaceStorage\ce3f7ae973d12df3302b51054a59a55\redhat.java\jdt_ws\test_68b156f3\bin' 'test'
Hey
PS C:\Users\victo\Desktop\School\Webdesign\VS Code\test>

Archived post. New comments cannot be posted and votes cannot be cast.

2 5 Share

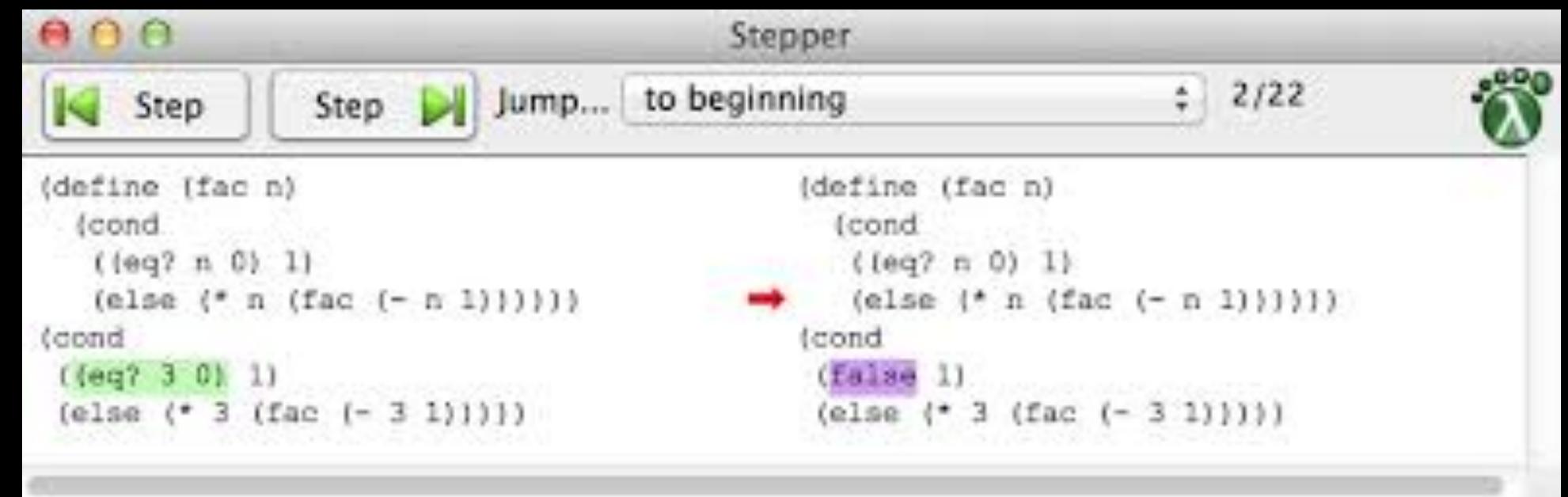
Sort by: Best

Search Comments

kennethmf · 4y ago ·

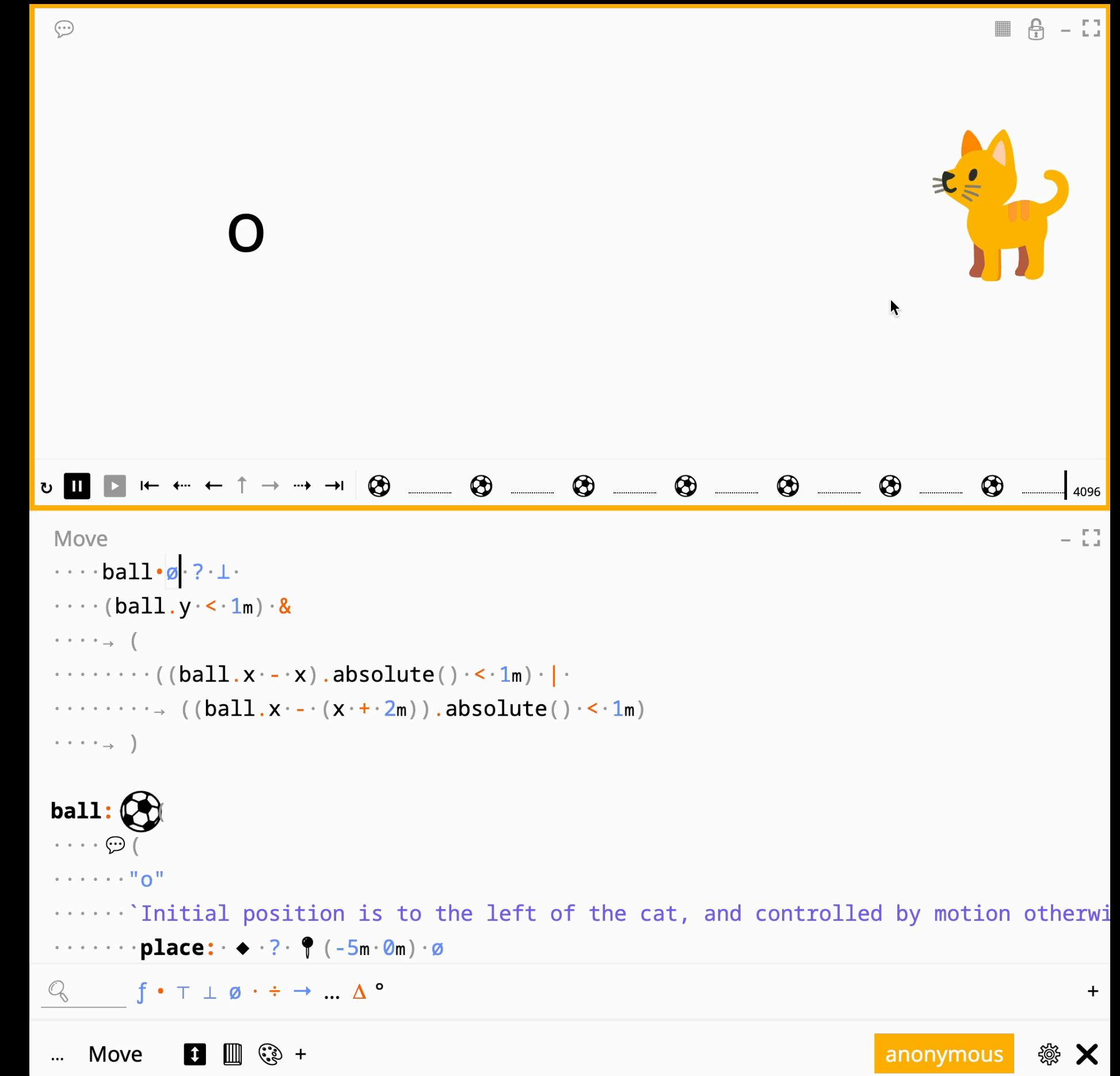
BETTER: RACKET + DR. RACKET

- ▶ Racket offers a nice reversible stepper, allowing learners to go forward and backward through an expression's evaluation, using a "rewriting" metaphor
- ▶ In addition to being reversible, this is more granular than line-by-line stepping, giving precise visibility into program behavior.



WORDPLAY: REVERSIBLE, GRANULAR

- ▶ In Wordplay, programs can be run forward and backwards, infinitely and instantaneously
- ▶ Program evaluation can be stepped at an extremely fine granularity, giving localized, accessible, explanations of every step in all supported languages.



OPEN QUESTIONS

- ▶ How can *all* EPL support highly flexible, reversible, granular inspectability of program evaluation?
- ▶ How might LLMs be used to explain program execution for different literacy levels, in different natural languages, across different cultures?
- ▶ How can technical transparency support liberatory, critical learning about what programs do and why?



EMBRACE ALL LANGUAGES,
CULTURES, AND VALUES

CULTURAL

Credit: Clay Banks

THE REQUIREMENT

- ▶ *EPL must be culturally responsive and sustaining in how they are designed, explained, and framed, enabling identity-inclusive pedagogy.*
- ▶ In practice, this means:
 - ▶ Supporting multilingual learners, using language flexibly, not just English
 - ▶ Drawing upon many cultures to describe and explain concepts in programming, not just Western, white settler cultures
 - ▶ Questioning the Western cultural ideas embedded in CS, including binary truth values, discrete math, and rigid categories

WHY?

- ▶ **Decolonization.** Our social worlds are shaped by a history that has centered the culture and language of colonizers, and steadily erased all other culture.
- ▶ Humanity deserves to shape the cultural worlds they live in, including restoring those from the past and creating new ones.
- ▶ Computer science has not resisted colonization, it has embraced it and amplified it. It has even become a discipline that *itself* colonizes, redefining and displacing the ideas of other disciplines with its own, at the expense of nuance.

BAD: PYTHON

- ▶ Syntax is English only, no translations, only a few non-English locales of documentation
- ▶ Python 2 had very weak Unicode support, privileging Latin characters only
- ▶ Libraries are full of English metaphors (“pickle”, “nanny”, “abc”)
- ▶ “Zen of Python” simplicity mantras are in tension with diversity:
 - ▶ *“There should be one – and preferably only one – obvious way to do it.”* – obvious to whom?
 - ▶ *“Special cases aren’t special enough to break the rules.”* – whose rules and why not?



r/madeinpython • 2 yr. ago

Tungara2007

...

Python with Spanish Syntax

This video is a short demo of a project I'm working on. I hope that this project can help people who are starting to program and explore if they are interested in the Python language, without the need to worry about understanding the English commands.

<https://youtu.be/445I5zPk9Vw>



factorpolar • 2y ago ·

As a native Spanish speaker, I must say that this is cool. Trying to help students who still can't understand English very well is a great and noble goal.

However, I must say that learning programming like this might be counterproductive for students. I **really don't** want to discourage you, just let me explain from my very particular point of view.

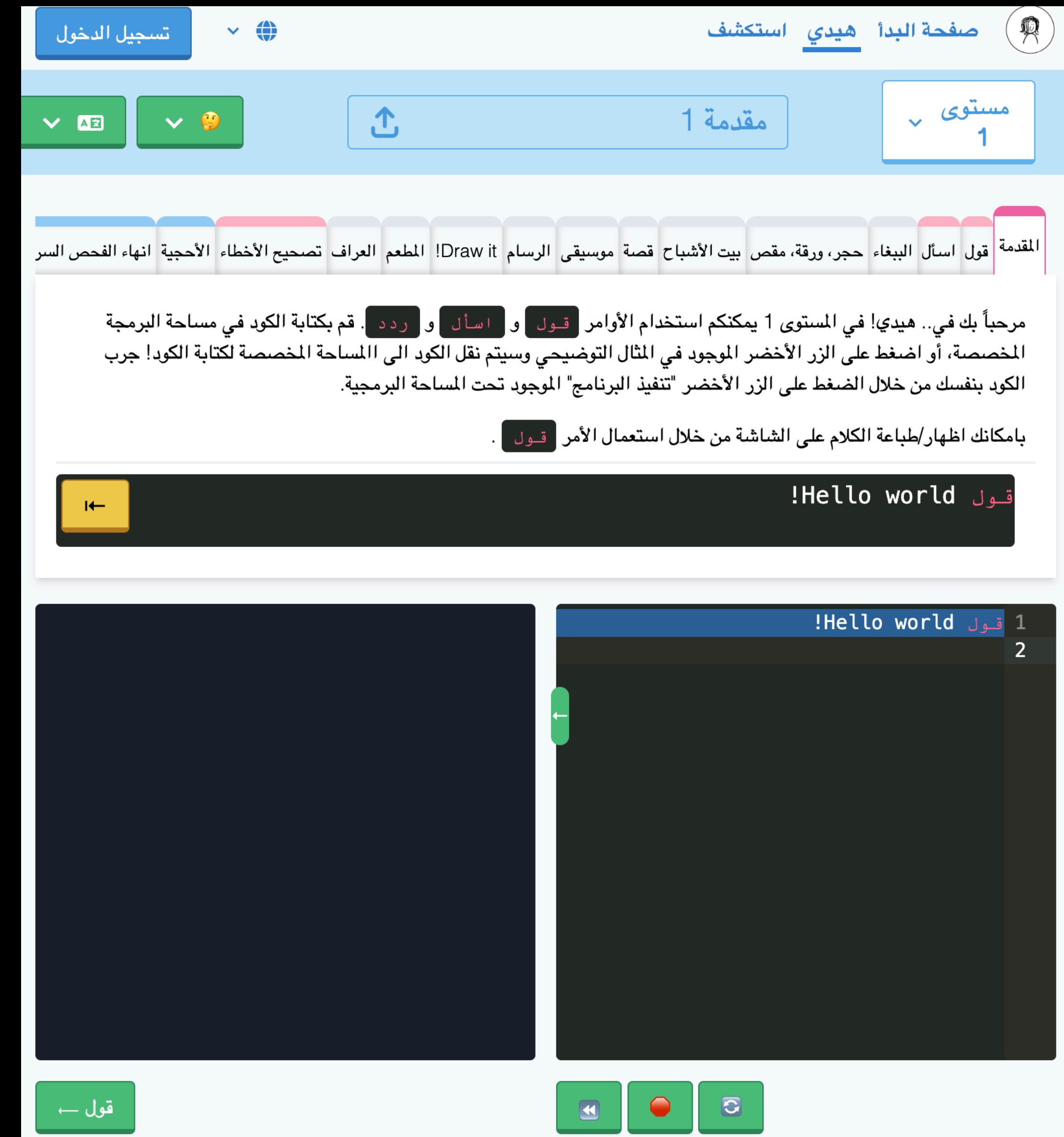
As you know, most programming languages use English keywords. If students want to use those programming languages, they will need to learn those keywords and their meaning, sooner or later.

Would your project make it easier for non-English speakers to learn programming? Probably, so I would encourage you to continue working on it. However, in my personal opinion, I think learning programming in the usual way helped me to learn English words, instead of the other way around.

In other words, *the colonizers won, stop trying to decolonize Python, its not realistic.*

BETTER: HEDY

- ▶ 47 different language supported, even localizing the language syntax to mirror different language grammars.
- ▶ Doesn't support mixing languages to support our multilingual world, but gives a glimpse of what a truly global language and platform might look like.



WORDPLAY: MULTILINGUAL CODE AND OUTPUT

- ▶ All names, text, and documentation in programs can have any number of language-tagged aliases
- ▶ This allows programs *and output* to be “skinned” and automatically translated into any combination of natural languages

WhatWord
press space to begin



The screenshot shows a terminal-like interface for the "WhatWord" application. At the top, there's a small icon of a computer monitor and keyboard, followed by the text "words". Below that is a command-line interface with the following text:
`a list of guesses and a secret word`
`Game(guesses • [""] · secret • "") · (`
` · · → guessesRemaining: · (secret. « () · · 2) · - · guesses. « ()`
` · · → status: ·`
` · · → secret · = · "" · ? · "start"`
` · · → secret · → · ["] . all(f(letter · "") · guesses.has(letter)) · ? · "won"`
` · · → · → · guessesRemaining · ≤ · 0 · ? · "lost"`
` · · → · → · → · "playing"`
`)`

At the bottom, there's a status bar with the text "start: · Game([] · "")" and several icons: a checkmark, the application name "WhatWord", a refresh symbol, a document icon, a gear icon, the word "words", a search bar with a magnifying glass icon, a plus sign, and user information "anonymous" with a gear icon.

OPEN QUESTIONS

- ▶ How can EPL support multilingual learners, while also supporting their very rational economic motivation to be English fluent?
- ▶ How can data structures and algorithms be described with a multiplicity of cultural metaphors, rather than just English, Western ones?
- ▶ How might youth be empowered to create their own EPL, with their own ideas about how computation should work?



REQUIRE NO COST

OBtainable

Credit: Rawpixel | Deposit Photos

THE REQUIREMENT

- ▶ *Learners must be able to access an EPL and its tools and resources independent of their financial means.*
- ▶ In practice, this means:
 - ▶ EPL must be free
 - ▶ EPLs must not require paid access to the internet
 - ▶ EPLs must not require purchasing personal devices
 - ▶ EPL must assume old hardware, constrained and slow internet access.

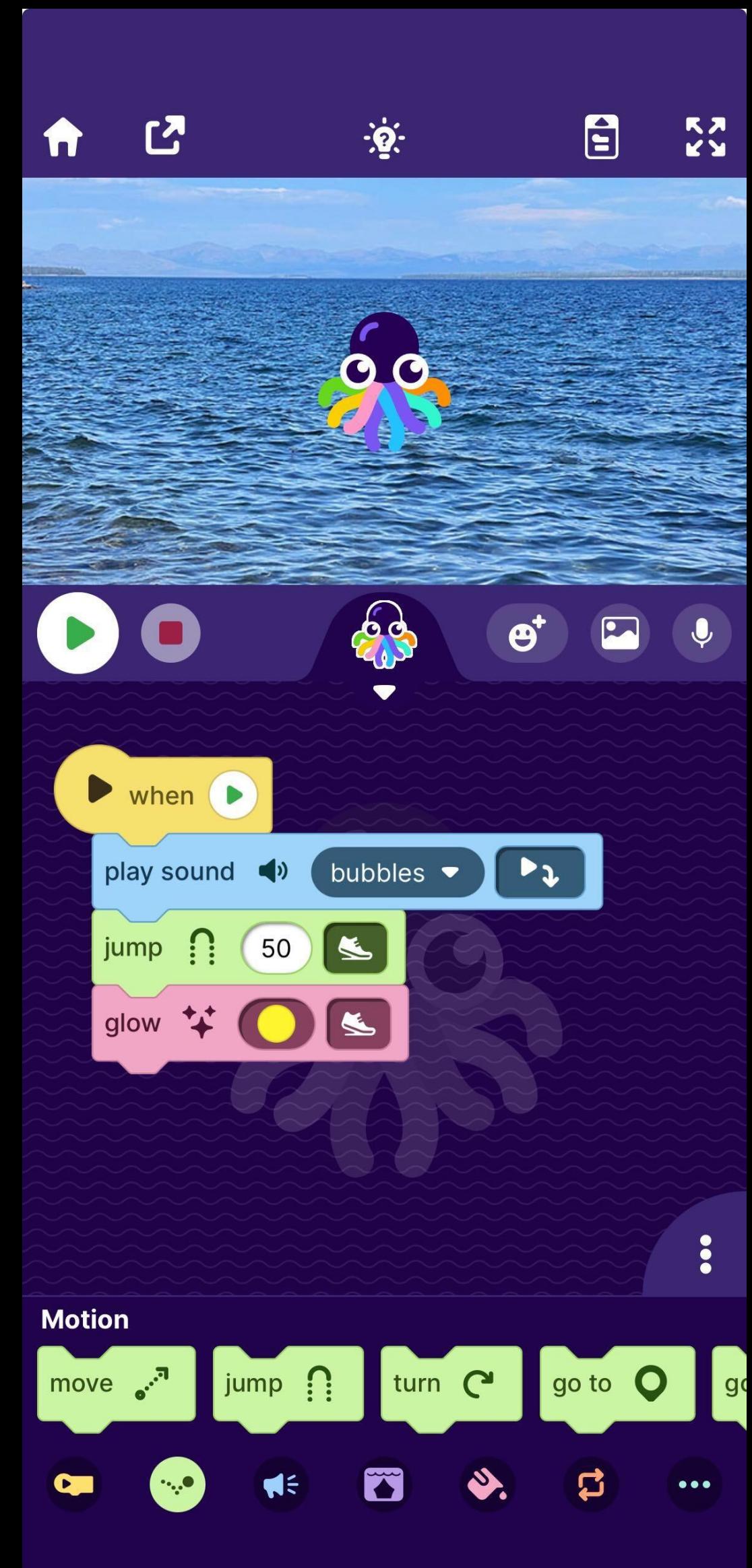
WHY?

- ▶ **Economic justice.** People's ability to participate in the world should not be shaped by the economic conditions in which they are born, or the opportunities shaped by the systems of oppression that surround them.
- ▶ Computer science has broadly ignored this right, instead designing for those that can access modern devices and the internet, and leaving everyone else behind, in pursuit of profit.

OBTAINABLE

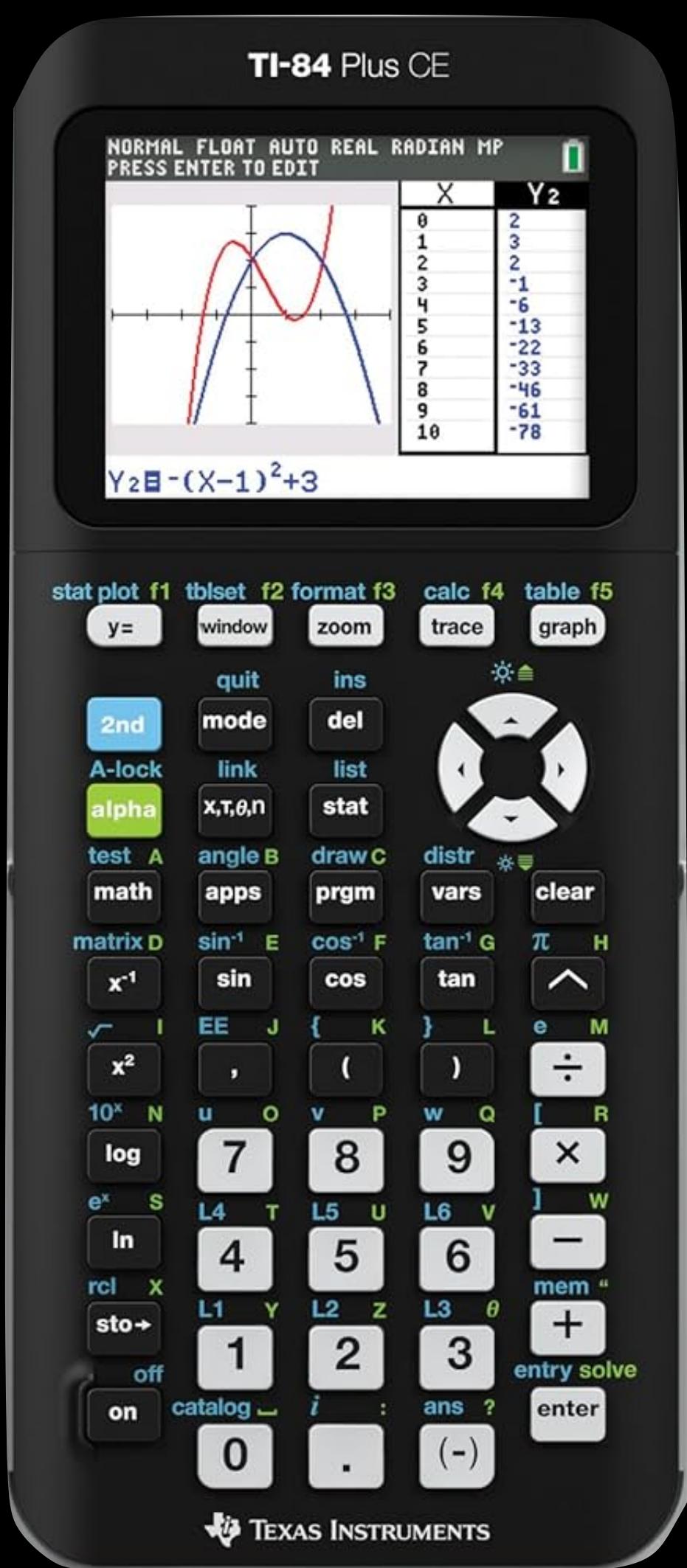
BAD: OCTOSTUDIO

- ▶ It is free and only requires internet access to download, which is just.
- ▶ But it requires access to an Android 8 or iOS 15 compatible device, the ability to install applications on it, and time to use the device.
- ▶ The only youth who might have this access are those either with their own devices, or in schools with enough resources to maintain 1:1 device access.



BETTER: TI GRAPHING CALCULATORS

- ▶ Low cost, and most schools already own them for math education, and have existing subsidies.
- ▶ Portable, battery powered, requires no internet access, and has a simple PL with access to a variety of sensors (speakers, LEDs).
- ▶ Problematic in how TI has a near monopoly over this market, accruing massive profit margins, limiting innovation.



WORDPLAY: ANY BROWSER, ANY DEVICE

- ▶ Wordplay is free, on the web, and does not require an active internet connection
- ▶ Its footprint is tiny, as text, emojis, and programs require only minimal device storage
- ▶ It's fully functional on smartphones, tablets, laptops, desktops, ancient school and library computers



OPEN QUESTIONS

- ▶ How can EPLs be financed to sustain an ecosystem of hardware and software without exploiting youth and schools for profit?
- ▶ How can we reconcile a need for a multiplicity of platforms to meet a diversity of learner needs with the limited capacity to sustain platforms?
- ▶ If we embrace EPLs that aren't obtainable, how can we sustainably subsidize access to EPLs to make them obtainable when school funding continues to decay?

Credit: Amy J. Ko



CENTER POWER AT THE MARGINS

DEMOCRATIC

THE REQUIREMENT

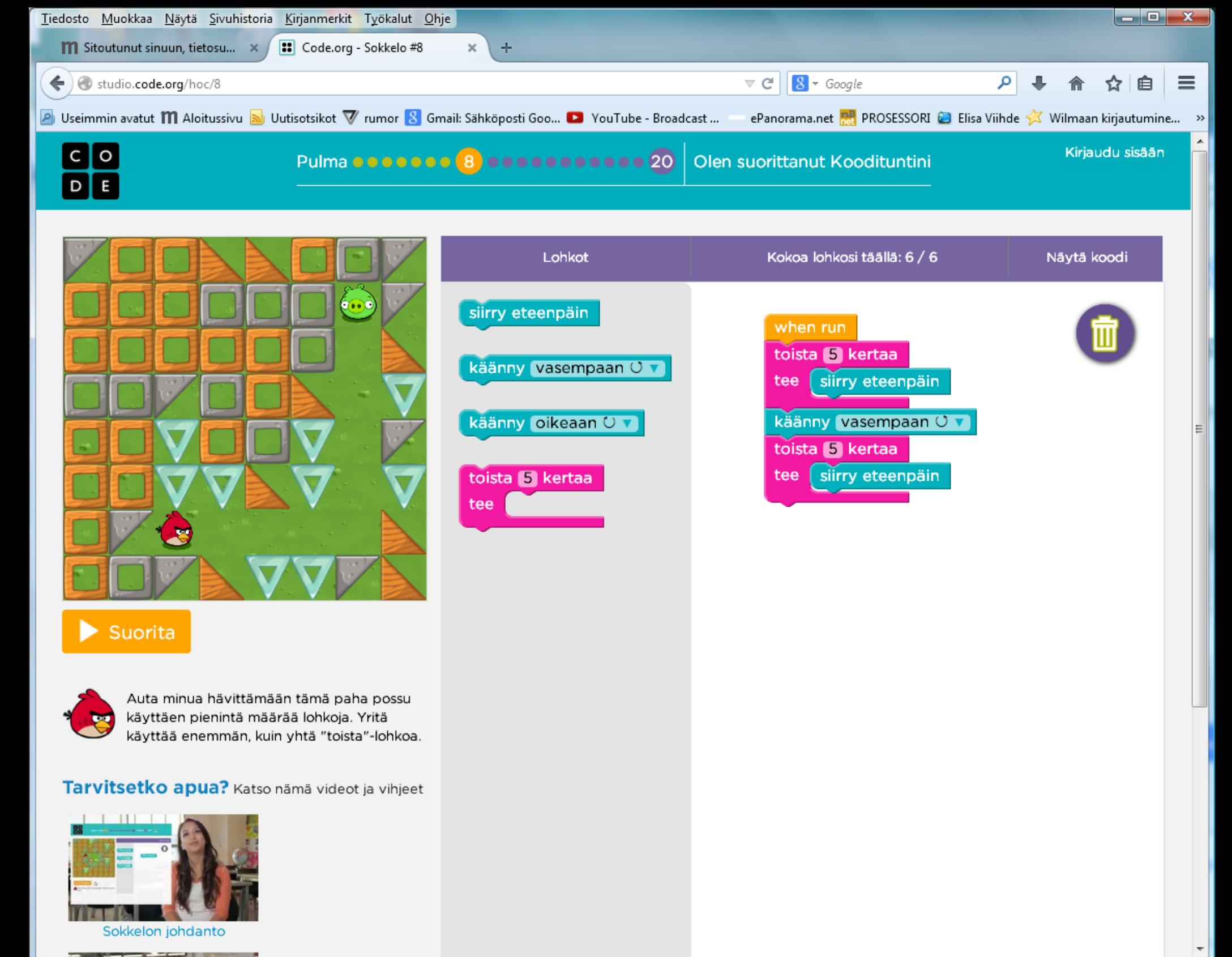
- ▶ *EPLs must be governed by and accountable to learners and their communities of support, especially those marginalized in computing and society more broadly.*
- ▶ In practice, this means:
 - ▶ EPL must be open source
 - ▶ EPL designers must give up the power to design to teachers and students
 - ▶ They must have community processes to engage, gain power, and influence design
 - ▶ Design processes must be organized to center community needs, not other goals, like research, profit, or innovation

WHY?

- ▶ The power to shape programmable media should be one that **everyone** has, as the media is used to shape what rights and opportunities everyone has.
- ▶ In other words, programming language creators have no right to control the language unilaterally without the voices of those who are impacted by them, directly, or indirectly.

BAD: CODE.ORG STUDIO

- ▶ Open source with contributors guidelines, with advisory boards to shape product priorities
- ▶ Unfortunately, design authority is centralized in code.org's design and engineering staff, not in the youth or teachers that they serve



BETTER: PROCESSING.ORG

- ▶ Open source, with ample community contributions and pull requests
- ▶ The foundation runs public events that solicit advocacy
- ▶ Funds fellowships for teachers to explore and shape the platform
- ▶ Partners with advocacy organizations at the margins of computing
- ▶ Directly engages communities and community leaders to shape priorities



WORDPLAY: STUDENT- AND TEACHER-LED

- ▶ We run a quarterly design studio with middle, high, and college students and teachers to contribute design, development, localization, community organizing, and governance, to the open source project
- ▶ We've hosted a youth and teacher advisory council to inform critical design and governance choices, guiding the project priorities



OPEN QUESTIONS

- ▶ How can we sustain the creation and support of communities, especially with low resource schools and families?
- ▶ How can we manage conflict in communities with different needs, and who should hold power to resolve these conflicts?
- ▶ How can EPL remain *redesignable* in response to evolving needs in a community, when they are often built in such immutable ways?



BUILT TO LAST

ENDURING

Credit: Unknown

THE REQUIREMENT

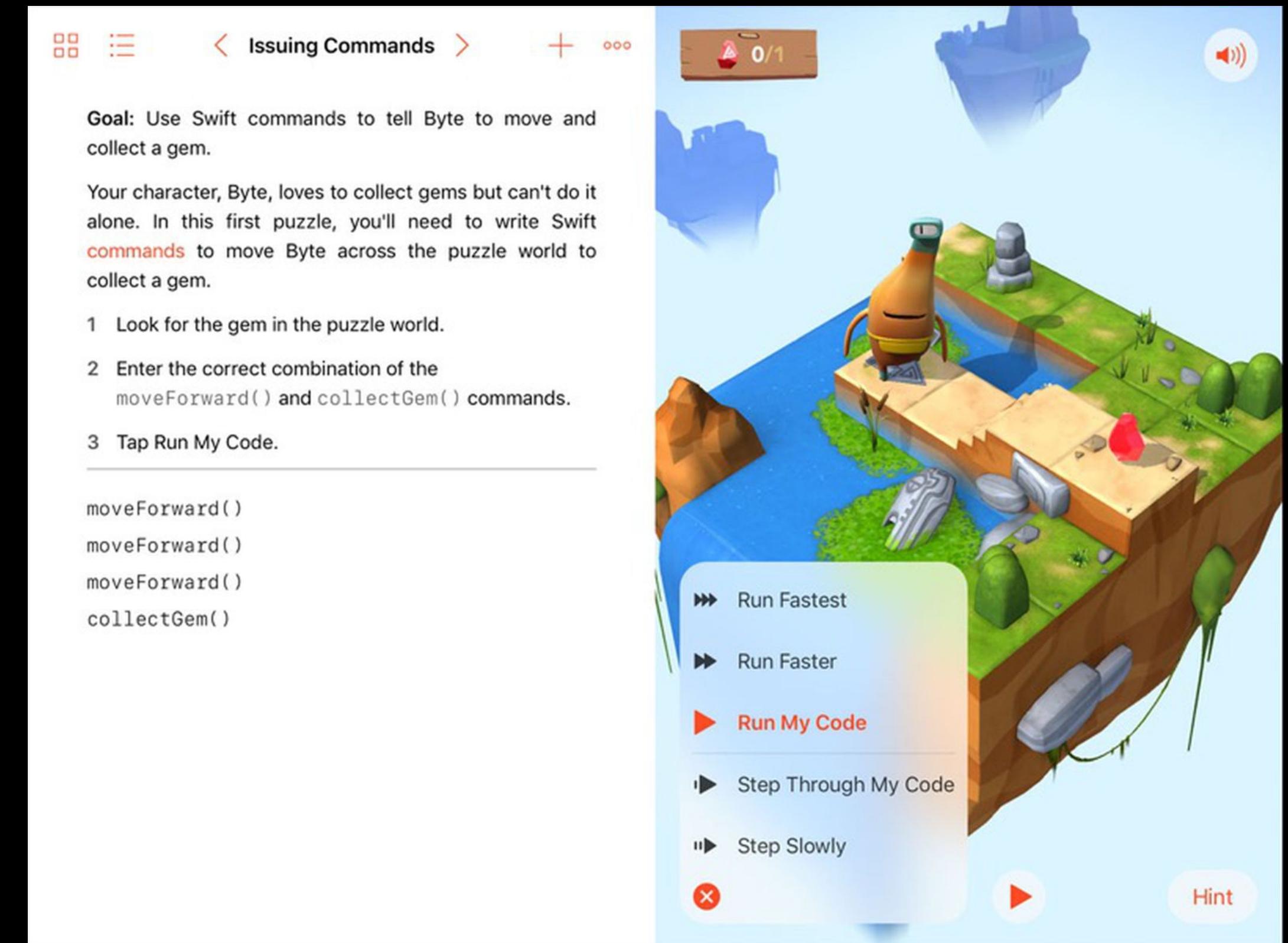
- ▶ *EPL must be sustainable for as long as a community needs them to be, respecting a community's capacity for change and planet's capacity for computation.*
- ▶ In practice, this means:
 - ▶ EPL must be sustainable, maintainable, and resilient
 - ▶ EPL must also be discardable when they no longer serve justice

WHY?

- ▶ Educational programming languages, in service of public education, or **public infrastructure**.
- ▶ Infrastructure should be sustainable and built to last, but also amenable to replacement when it no longer serves the public good.
- ▶ Current EPL governance is far from sustainable or replaceable: most are built with very little support, and problematic languages that become popular are hard to replace.

BAD: SWIFT PLAYGROUNDS

- ▶ Solid platform and curriculum, billions in funding to sustain it
- ▶ No statement of how long it will be supported, limiting adoptability by teachers and districts long term
- ▶ No way to stop or mitigate Apple's capitalist efforts to weave it into classrooms, even when such efforts might do harm



BETTER: SCRATCH

- ▶ Large base of funding, now centralized in the Scratch Foundation
- ▶ More than 20 years of support, including multiple re-implementations.
- ▶ Limited openness means that community's capacity to maintain the platform may be limited if the foundation were to stop supporting the project.



Credit: Scratch Foundation

WORDPLAY: BUILT TO LAST

- ▶ The platform is fully open source, with extensive onboarding documentation for contributions
- ▶ The platform is fully web standards compliant, with minimal cloud-dependencies for persistence and auth
- ▶ The platform relies on text, no images, minimizing energy and storage use
- ▶ But it has a single point of failure: *me.*



Wordplay

A justice-centered programming language for the world's languages.

27 followers

United States of America

README.md

Hello! We are **Wordplay**, a community of researchers, educators, and other contributors who are working towards a vision of justice-centered programming languages.

By justice, we mean programming languages that:

- Are accessible, usable, and expressive for everyone, regardless of their language fluency and way of reasoning, communication style, or background.

OPEN QUESTIONS

- ▶ What are justice-centered models for sustaining EPL technically, socially, and politically, to promote resilience?
- ▶ How can governance be organized to give teachers and youth power to retire EPL that are doing more harm than good?

WHAT'S NEXT?

THE KEY POINT

- ▶ Educational PL play an instrumental role in structuring what kinds of computing education are possible, who education serves, what kinds of digital worlds are possible, and whether those worlds are just.
- ▶ Being justice-centered means redistributing the power to design EPL to learners' and their communities, to more intentionally center and support their needs, values, cultures, and abilities
- ▶ ALTCODE requirements are one possible way to operationalize justice for EPL design and they raise many technical, social, and political grand challenges for future work.

THIS IS (VERY) HARD

- ▶ The challenges are **technical, social, and political**:
 - ▶ Transparency requires a performance hit
 - ▶ Multiculturalism requires political judgements about language, ideas, culture
 - ▶ Democracy requires power sharing, conflict resolution, compromise
 - ▶ Accessibility can create complexity
 - ▶ Endurance requires \$, time
- ▶ And all of this in world that increasingly bans, litigates, and defunds diversity, equity, and justice efforts, doubling down on racial and patriarchal capitalism.

JUSTICE-CENTERED EDUCATIONAL PL ARE HARDLY ENOUGH

- ▶ We still need:
 - ▶ Properly funded public schools
 - ▶ A diverse, well-supported CS teaching workforce
 - ▶ Accessible classrooms
 - ▶ Universal access to devices and the internet
 - ▶ Teaching methods that are culturally responsive, sustaining
 - ▶ Teachers to make the most of all of the above

SOME OF US WILL BUILD, SOME OF US WILL
ORGANIZE, AND SOME OF US WILL TEACH.

I HOPE SOME OF YOU WILL JOIN US,
CREATING A COMPUTATIONAL WORLD THAT
WORKS FOR EVERYONE, ONE PL AT A TIME

Accessible
Liberatory
Transparent
Cultural
Obtainable
Democratic
Enduring

JUSTICE-CENTERED EDUCATIONAL PROGRAMMING LANGUAGES

DISCUSS

Learn more at amyko.phd and wordplay.dev