DESIGNING USABLE PROGRAMMING LANGUAGES

ELIZA WELLS, EMBEDDED ETHICS
4/12/22
Take a moment to make sure you’re sitting near 2 or 3 people to talk to. Share your **name** and **favorite TV show** with each other.
WHO AM I?

Eliza Wells, elizaw@mit.edu
PhD candidate in Philosophy at MIT
Research interests: moral and social philosophy
Graduate Fellow for Embedded EthiCS @ Harvard
PLAN FOR TODAY

1. Introduce the philosophical framework of egalitarianism as a way of thinking about justice
2. Discuss the importance of programming language usability from an egalitarian perspective
3. Practice thinking like an egalitarian using case studies
WHY ARE YOU INTERESTED IN COMPUTER SCIENCE?
WHAT DO YOU THINK YOU WILL GAIN FROM ENGAGING WITH IT?
BEING A COMPUTER SCIENTIST PROVIDES ACCESS TO IMPORTANT GOODS.

- **Opportunities**
  - Jobs, connections, ways to contribute

- **Resources**
  - Money, knowledge, power

- **Abilities**
  - Skills, ways of thinking
DOES IT MATTER WHO HAS ACCESS TO THOSE GOODS?
DOES IT MATTER WHO HAS ACCESS TO THOSE GOODS?

Philosophers argue yes!

Questions about how goods should be distributed are questions of justice: what do people deserve?
Egalitarianism is one answer to the question of justice. Because all persons have equal moral standing, it is unjust for them to be treated unequally based on morally irrelevant traits.
EGALITARIANISM

Egalitarianism is one answer to the question of justice. Because all persons have equal moral standing, it is unjust for them to be treated unequally based on morally irrelevant traits.

Traits that a) are not related to the basis of the treatment b) the person has limited control over.
EGALITARIANISM

Compare:

- Rejecting a candidate for a software development job because she is a slow runner
- Rejecting that candidate because she is a slow coder
- Rejecting her because she is a woman
WHEN ARE THE GOODS OF COMPUTER SCIENCE DISTRIBUTED UNEQUALLY?

One case we will consider today:

programming language usability.
Imagine a programming language that was so difficult to learn that no one could use it effectively.

- What problems would that raise?
- Would this language be worth developing further?
DO COMPUTER SCIENTISTS DESIGNING PROGRAMMING LANGUAGES HAVE A RESPONSIBILITY TO MAKE THEM USABLE?
EXAMPLE: RUST

Rust has many technical features that programmers value: it’s high-performing (12x faster than Python), efficient, memory-safe, thread-safe, etc.
EXAMPLE: RUST

How long did it take you to get productive in Rust?

- Less than a week: 7.1%
- Less than a month: 33.8%
- Less than a year: 30.3%
- More than a year: 6.4%
- I don't feel productive: 22.3%

Rust community survey, 2018
**EXAMPLE: RUST**

How long have you used Rust and still feel unproductive?

- More than 3 years: 14.2%
- 1 to 2 years: 5.7%
- 2 to 3 years: 1.6%
- 6 to 12 months: 12.1%
- 3 to 6 months: 16.0%
- Less than a week: 5.8%
- Less than a month: 19.8%
- 1 to 3 months: 24.9%

Rust community survey, 2018
WHAT MAKES A PROGRAMMING LANGUAGE USABLE?

1. **External resources**
   a. Educational resources (e.g. code camps)
   b. IDEs

2. **Internal resources**
   a. Libraries, packages
   b. Documentation
WHAT MAKES A PROGRAMMING LANGUAGE USABLE?

1. **External resources**
2. **Internal resources**
3. **Features of the language**
   a. Intuitive design
   b. Simplicity
   c. Natural language use
   d. Continuity with other programming languages
WHAT MAKES A PROGRAMMING LANGUAGE USABLE?

Rust’s implementation of memory safety is unintuitive and hard to master.
THERE CAN BE TRADEOFFS BETWEEN USABILITY AND OTHER TECHNICAL FEATURES.

- Some ways of making a language more usable can make it less effective.
- Sometimes, it’s not obvious how to make a language more usable in the first place.

Usability is **one feature among many** that computer scientists should care about.
USABLE FOR WHOM?
Imagine a programming language that was only difficult to learn and use productively for certain members of the population.

- What problems would that raise?
ONE POSSIBLE VIEW

Computer scientists are responsible only for building programming languages that are as technically excellent (safe, efficient, fast, etc.) as possible.

If only some people are able to make effective use of those tools, that’s okay.
Is this language not usable for some people because of morally irrelevant factors?

If so, the lack of usability (even if it’s unintentional) constitutes an injustice.
External resources

Educational resources and IDEs are not optimized for magnification or screenreaders.

Features of the language

Languages that use spaces instead of tabs or snake_case instead of camelCase require more cognitive load.
THINK LIKE AN EGALITARIAN

1. What goods are at stake?
2. What factors limit access to those goods?
3. Are those factors morally relevant?
4. What can we do about it?
WHAT GOODS ARE AT STAKE?

Being able to use a programming language impacts access to...

- Opportunities
- Resources
- Abilities
WHAT FACTORS LIMIT ACCESS TO THOSE GOODS?

ARE THOSE FACTORS MORALLY RELEVANT?
NON-NATIVE ENGLISH SPEAKERS

External resources
- Instructional material in other languages is often poor or outdated

Internal resources
- Documentation is often in complex, idiomatic English with culturally specific examples

Features of the language
- Many languages make heavy use of English vocabulary and syntax

---

```python
con = sqlite3.connect(":memory:")
con.execute("CREATE TABLE person ... <omitted for space>")
with con:
    con.execute("INSERT INTO person ... ", ("Joe",))
# con.rollback() is called after the with block finishes
# with exception, exception still raised and must be caught
try:
    with con:
        con.execute("INSERT INTO person ... ", ("Joe",))
except sqlite3.IntegrityError:
    print("couldn't add Joe twice")
```

Figure 1. English is ubiquitous in source code, as shown in this example adapted from the official Python language docs for SQL database management [55]. English appears in comments, variable names, `sqlite3` standard library API identifiers, and in both Python and SQL keywords.

From "Non-Native English Speakers Learning Computer Programming: Barriers, Desires, and Design Opportunities," by Philip Guo
WHAT CAN WE DO ABOUT IT?
DO COMPUTER SCIENTISTS DESIGNING PROGRAMMING LANGUAGES HAVE A RESPONSIBILITY TO MAKE THEM USABLE FOR EVERYONE?
COMPUTER SCIENTISTS SHOULD STRIVE TO ENSURE WHERE THEY CAN THAT MORALLY IRRELEVANT FACTORS DON’T IMPACT A PROGRAMMING LANGUAGE’S USABILITY.
THANK YOU!

Evaluation: https://tinyurl.com/CS152S22

Contact: elizaw@mit.edu