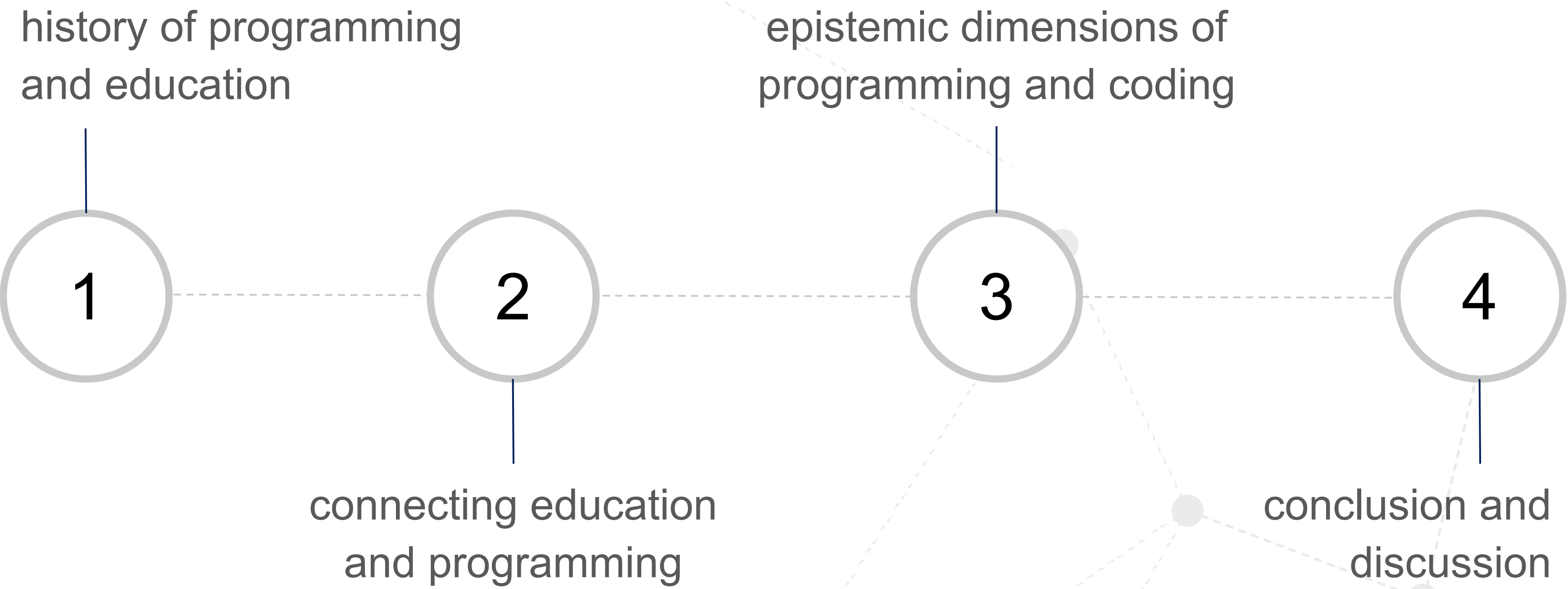
Epistemic Programming and Creative Coding

Programming as an Empowering Means for Self-Expression and Communication

Sven Hüsing, Carsten Schulte and Dan Verständig

Paderborn Colloquium on Data Science and Artificial Intelligence in School

Roadmap for today



1 history of programming and education

2 connecting education and programming

3 epistemic dimensions of programming and coding

4 conclusion and discussion

Historical view on Programming Education

1970s: From language courses to modeling/problem solving

📦📖 → 🤨💡

Merrienboer & Krammer (1987); Schulte (2013)

From Modeling to Shaping/Designing

1980s and 1990s:

PROBLEM-SOLVING THINKING = ALGORITHMIC THINKING

SOFTWARE-DEVELOPMENT = PROBLEM-SOLVING

Schulte (2001)

From Modeling to Shaping/Designing

1980s and 1990s:

PROBLEM-SOLVING THINKING = ALGORITHMIC THINKING

MODELING VS. CODING/HACKING

SOFTWARE-DEVELOPMENT = PROBLEM-SOLVING

Hubwieser (1999); Schubert (1991); Schulte (2001)

From Modeling to Shaping/Designing

1980s and 1990s:

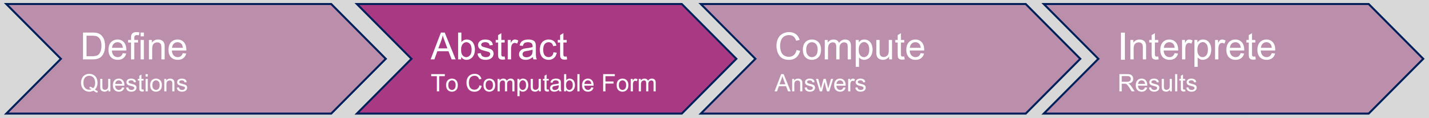
PROBLEM-SOLVING

=

"CONVERTING PROBLEMS FORMULATED IN EVERYDAY LANGUAGE INTO FORMAL RELATIONSHIPS"

Eberle (1996, p. 329, translated)

MODELING VS. CODING/HACKING



Define
Questions

Abstract
To Computable Form

Compute
Answers

Interprete
Results

Wolfram (2020)

Hubwieser (1999); Schubert (1991); Schulte (2001)

From Modeling to Shaping/Designing

1980s and 1990s:

TEACHING GOAL = TEACHING PROBLEM-SOLVING-SKILLS

TEACHING CONTENT = MODELING

CONNECTION WITH CS = SOFTWARE-ENGINEERING

Schulte (2001)

From Modeling to Shaping/Designing

1980s and 1990s:

➔Initial Aim:

Integrating societal impacts of CS through turning to modeling and to software engineering methods

Schulte (2001)

From Modeling to Shaping/Designing

1980s and 1990s:

→ Initial Aim:

Integrating societal impacts of CS through turning to modeling and to software engineering methods

"in the series of lessons studied, it is not possible, after algorithmization and programming, to relate these activities back to social issues" (Forneck, 1992, p.229, translated)

Schulte (2001)

From Modeling to Shaping/Designing

relation of CS and society: technological determinism

➔ “technology in general are the sole or prime antecedent causes of changes in society, and technology is seen as the fundamental condition underlying the pattern of social organization”

Chandler (1995); Schulte (2001)

From Modeling to Shaping/Designing

Revolution through Object Orientation

➔ Problem-Solving-Process consisting of several iterations

➔ networked thinking

https://www.mv-brackenheim.de/orchester/

https://www.istockphoto.com/de/vektor/cartoon-jazz-musiker-gruppe-vektor-illustration-gm1161177924-318084261

Bellin & Simone (1997); Schulte (2001)

From Modeling to Shaping/Designing

Revolution through Object Orientation

➔ Problem-Solving-Process consisting of several iterations

➔ networked thinking

➔ wider range of problems/more authentic problems

=> Student as lerner in an active role

Bellin & Simone (1997); Schulte (2001)

From Modeling to Shaping/Designing

Revolution through Object Orientation

Modeling != mapping of the real world situation

Modeling as designing complete systems

Shaping and Being Shaped

Schulte (2001); Schulte & Budde (2018)

From Modeling to Shaping/Designing

Conclusion: Object-Orientation:

- more open questions/problems regarding programming (in school)
  - not only algorithmically solvable problems
- cognitive and CS learning goals
- Interaction of Human and Computer/Programs

Schulte (2001)

3 Dimensions of Education

2 education as goal (individual learning)

1 education as output (governance)

3 education as process of transformative learning (philosophy of education)

Framework to describe goals of subject related education

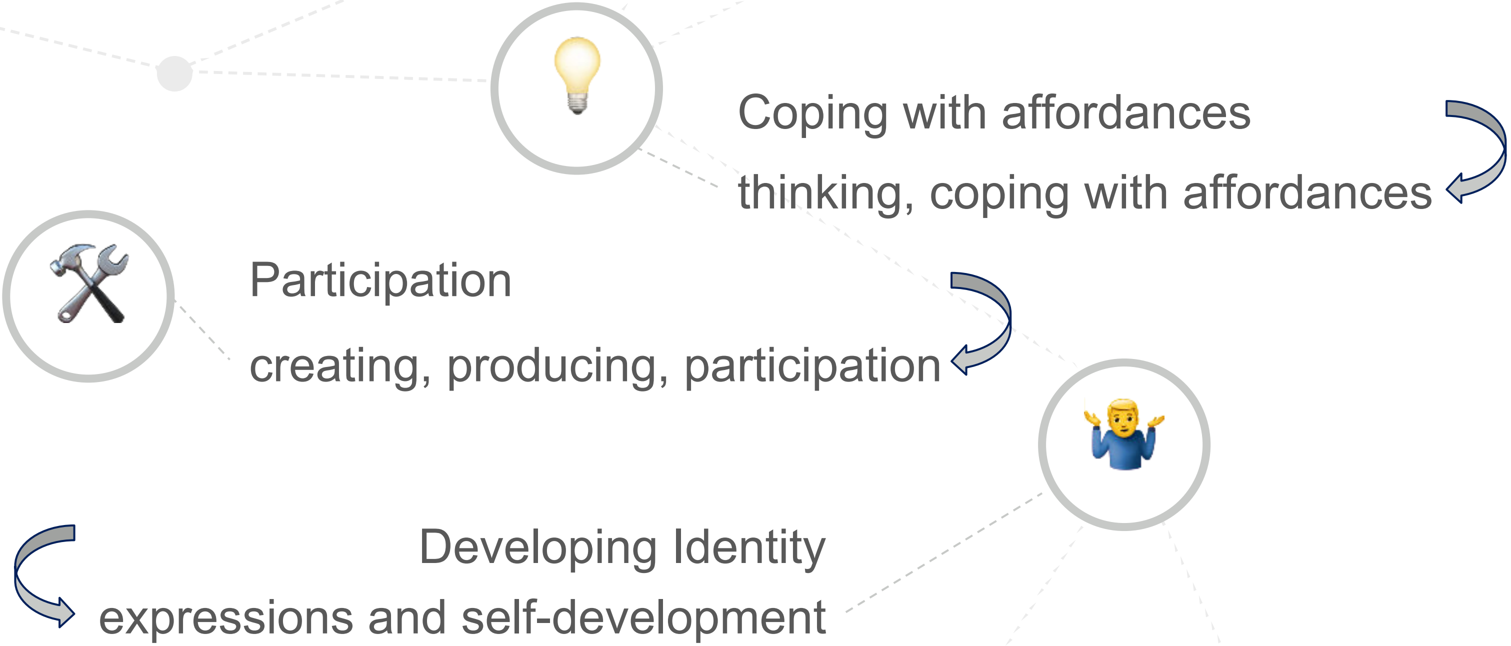
Coping with affordances

Participation

Developing Identity

GFD (2009); Schulte (2013)

Framework to describe goals of subject related education



Coping with affordances

thinking, coping with affordances

Participation

creating, producing, participation

Developing Identity

expressions and self-development

GFD (2009); Schulte (2013)

Framework to describe goals of subject related education

Coping with affordances

thinking, coping with affordances
trial and error/experiments

Participation

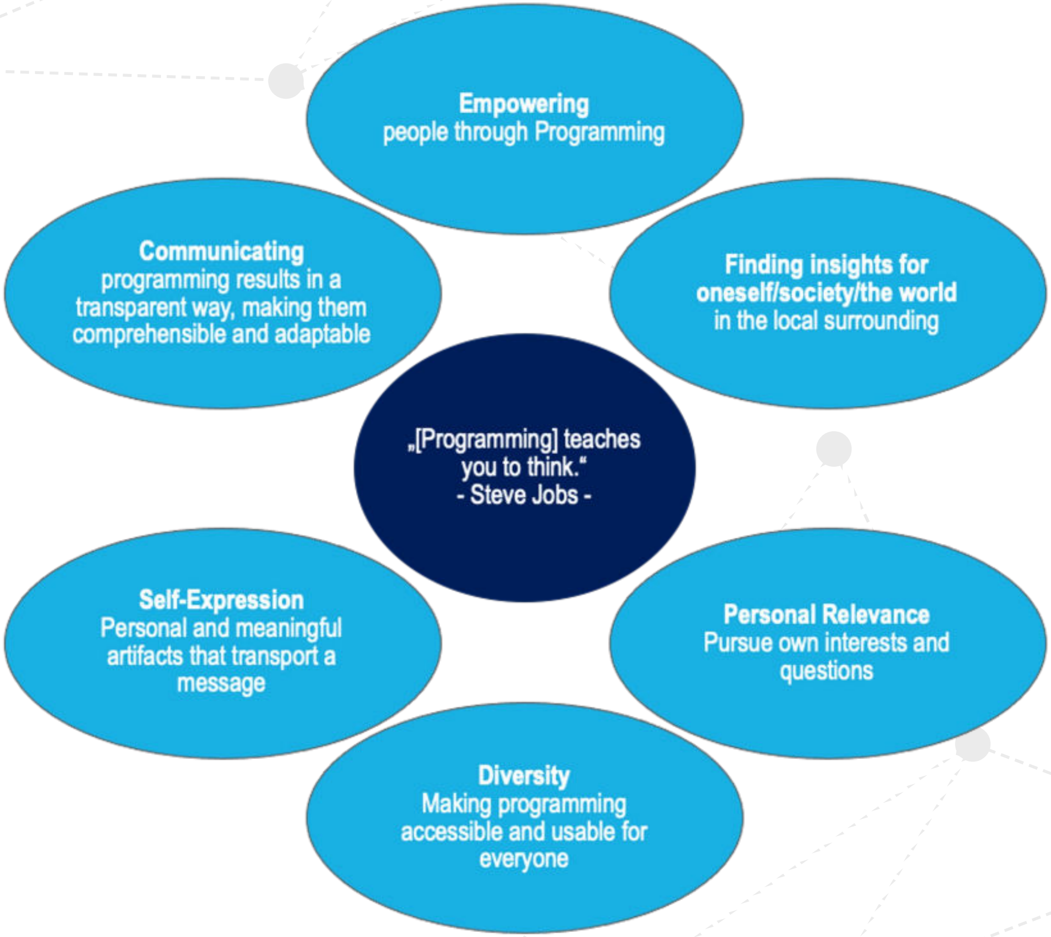
creating, producing, participation
projects

Developing Identity

expressions and self-development
remixing

Schulte (2013)

Vision of Epistemic Programming



Empowering
people through Programming

Communicating
programming results in a transparent way, making them comprehensible and adaptable

Finding insights for oneself/society/the world
in the local surrounding

„[Programming] teaches you to think.“
- Steve Jobs -

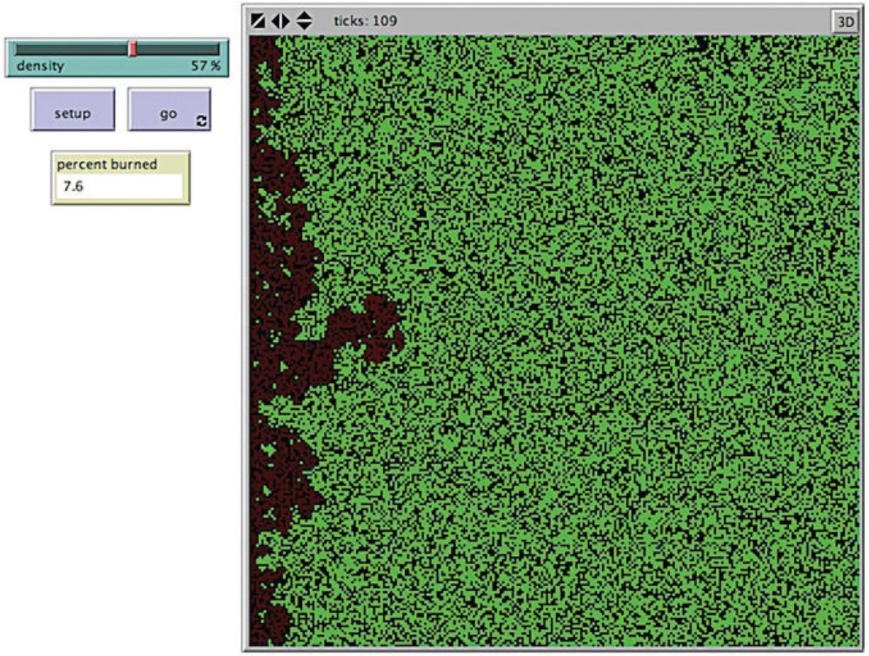
Self-Expression
Personal and meaningful artifacts that transport a message

Personal Relevance
Pursue own interests and questions

Diversity
Making programming accessible and usable for everyone

Sven Hüsing, Carsten Schulte and Dan Verständig

Epistemic Programming - Products



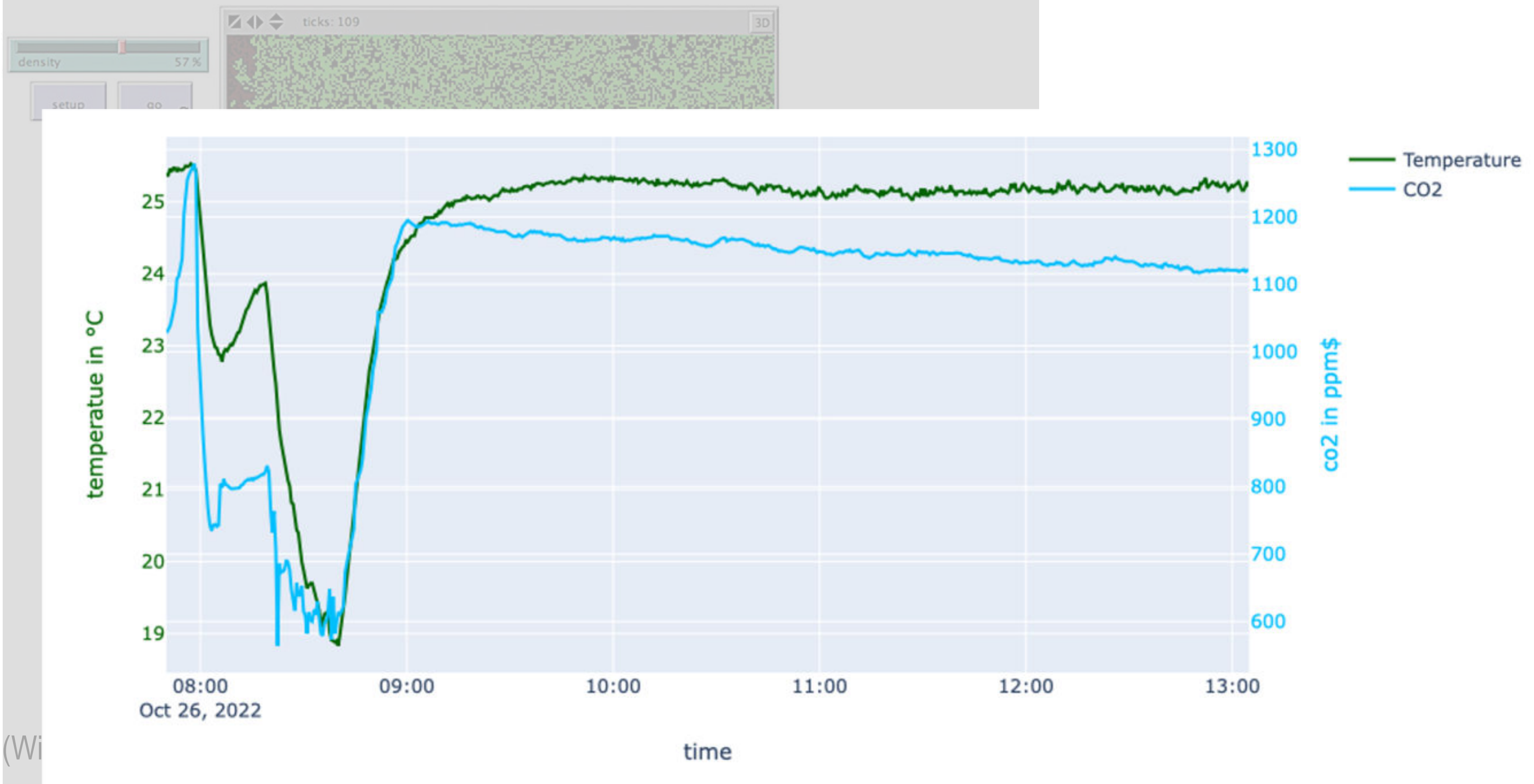
density 57 %

ticks: 109

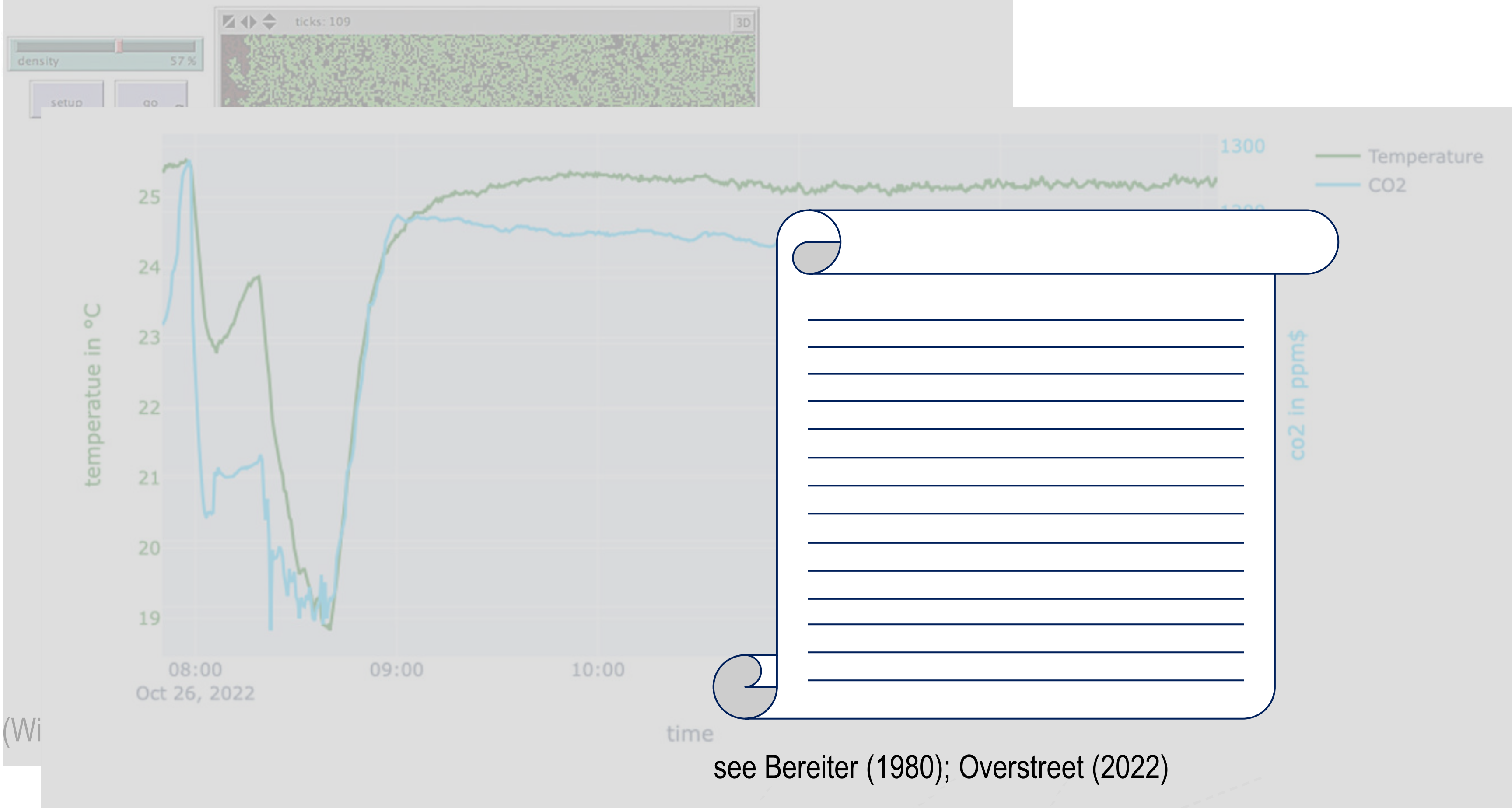
percent burned 7.6

(Wilensky et al. , 2014)
(also see Seoane et al. (2022))

Epistemic Programming - Products



Epistemic Programming - Products



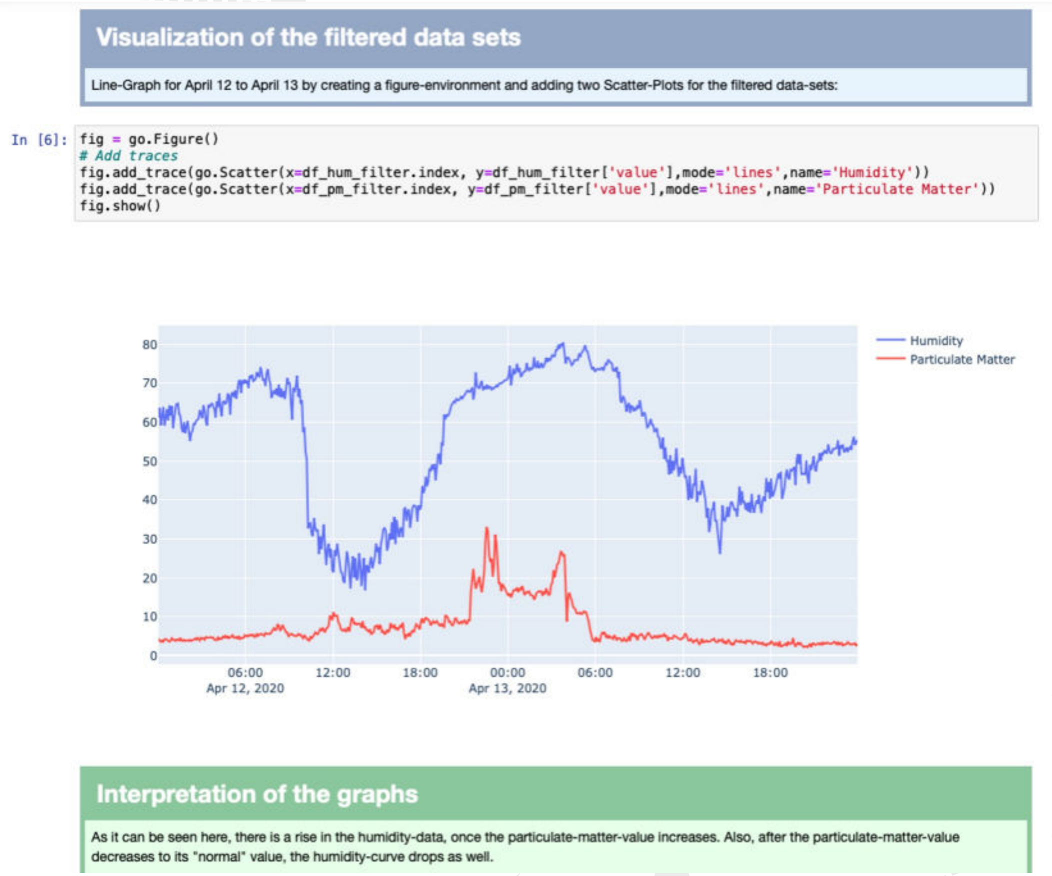
(Wi

see Bereiter (1980); Overstreet (2022)

Sven Hüsing, Carsten Schulte and Dan Verständig

Computational Essays - Exploring the Example

- Example:



Visualization of the filtered data sets

Line-Graph for April 12 to April 13 by creating a figure-environment and adding two Scatter-Plots for the filtered data-sets:

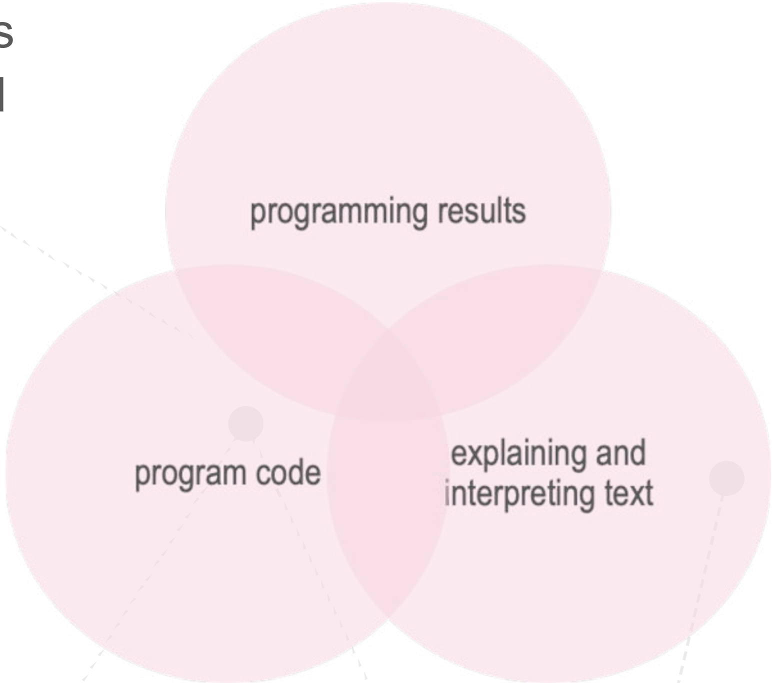
```
In [6]: fig = go.Figure()
        # Add traces
        fig.add_trace(go.Scatter(x=df_hum_filter.index, y=df_hum_filter['value'],mode='lines',name='Humidity'))
        fig.add_trace(go.Scatter(x=df_pm_filter.index, y=df_pm_filter['value'],mode='lines',name='Particulate Matter'))
        fig.show()
```

Interpretation of the graphs

As it can be seen here, there is a rise in the humidity-data, once the particulate-matter-value increases. Also, after the particulate-matter-value decreases to its "normal" value, the humidity-curve drops as well.

Computational Essays

- Making programming results as well as emerging insights comprehensible and reproducible for the reader
- from reader to tinkerer to programmer

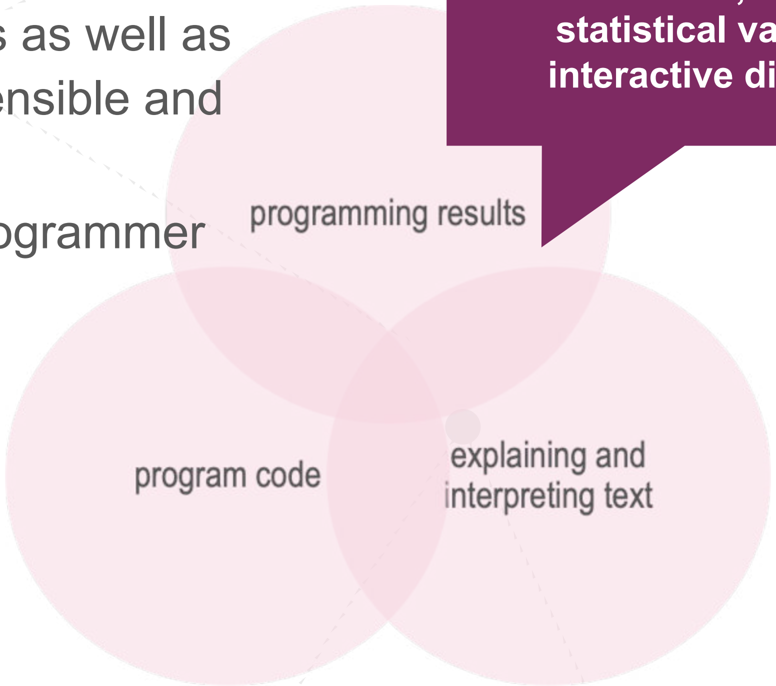


Sven Hüsing, Carsten Schulte and Dan Verständig

PADERBORN UNIVERSITY ProDaBi Project Data Science and Big Data at School UNIVERSITÄT BIELEFELD Deutsche Telekom Stiftung

Computational Essays

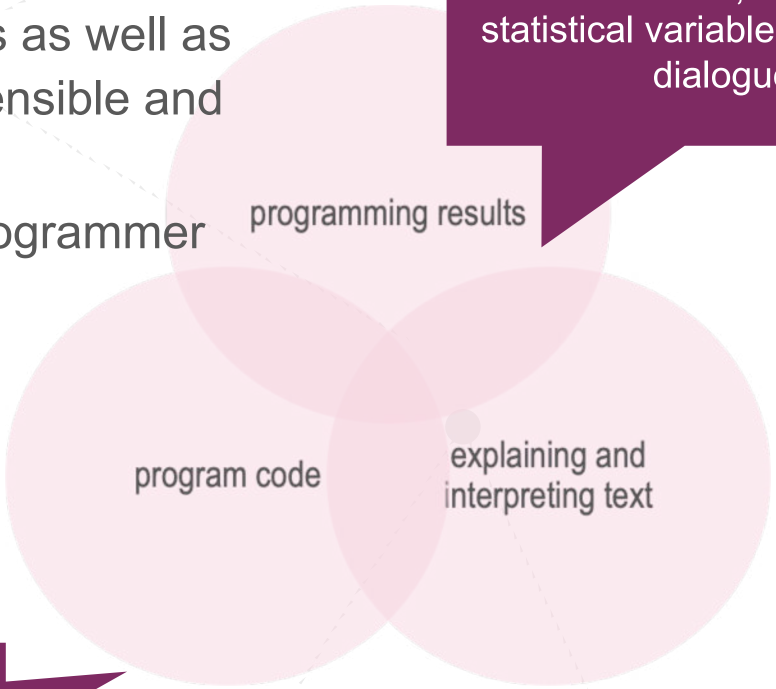
- Making programming results as well as emerging insights comprehensible and reproducible for the reader
- from reader to tinkerer to programmer



visualizations, simulations, statistical variables, interactive dialogues

Computational Essays

- Making programming results as well as emerging insights comprehensible and reproducible for the reader
- from reader to tinkerer to programmer

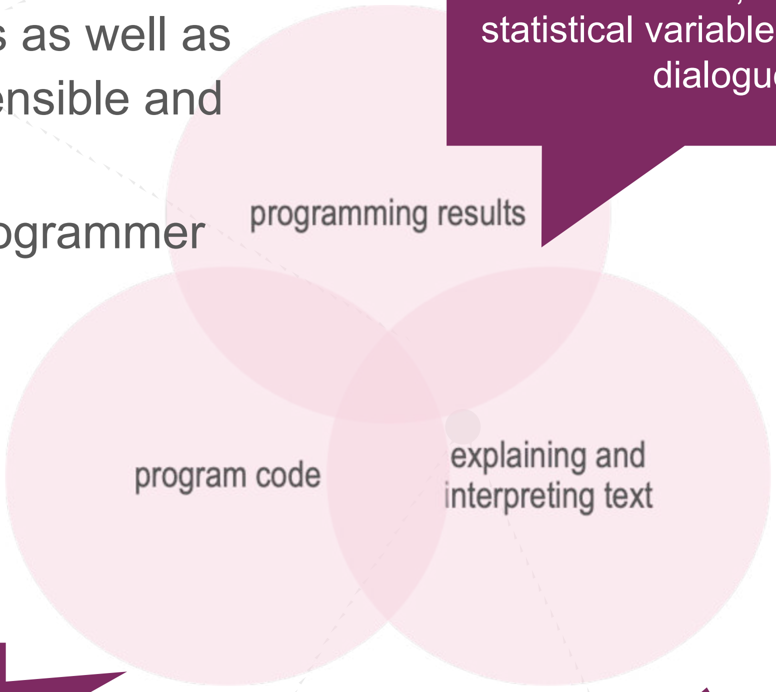


visualizations, simulations, statistical variables, interactive dialogues

adaptable and executable

Computational Essays

- Making programming results as well as emerging insights comprehensible and reproducible for the reader
- from reader to tinkerer to programmer

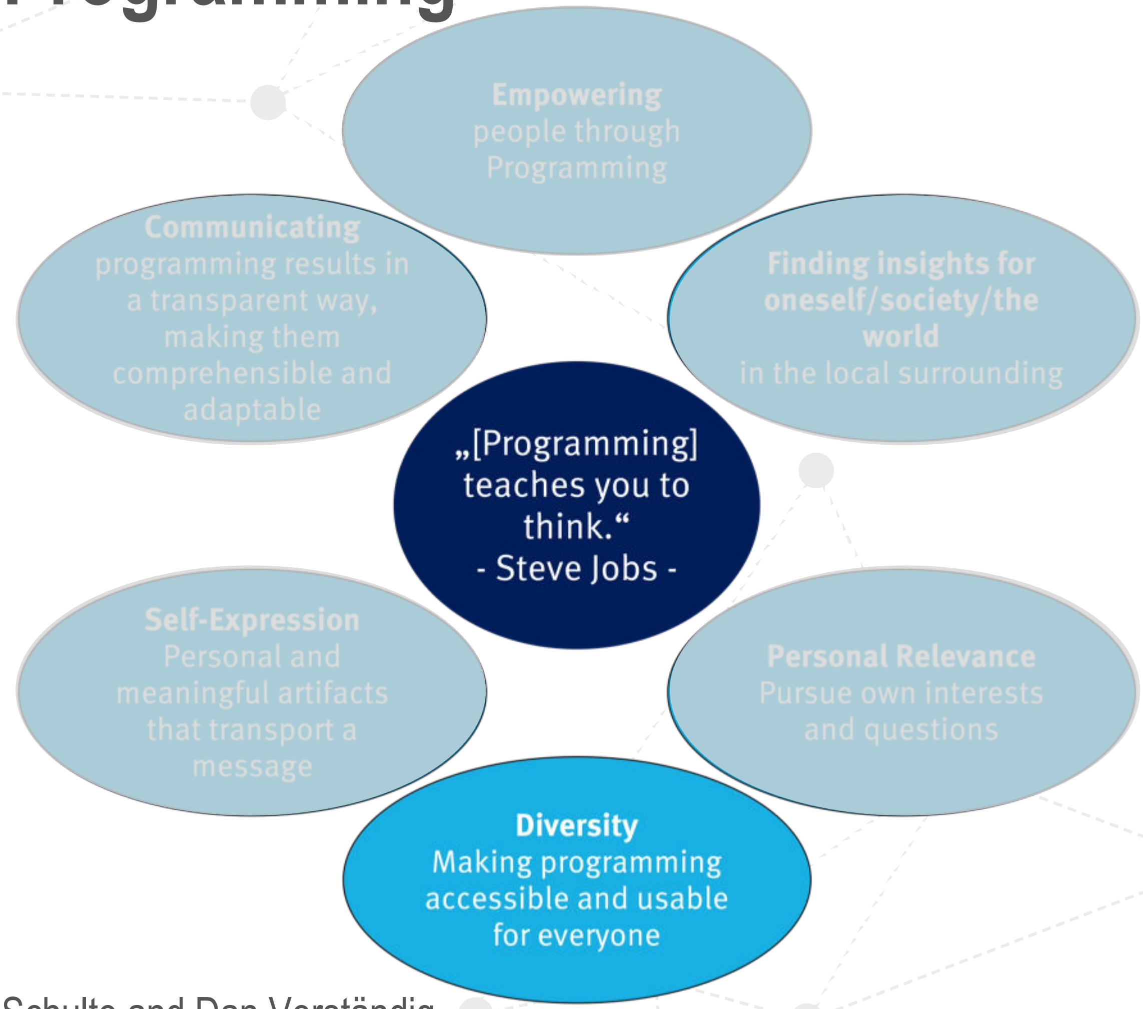


visualizations, simulations, statistical variables, interactive dialogues

adaptable and executable

documentation of the code, description of the process, interpretation of results

Epistemic Programming



„[Programming] teaches you to think.“
- Steve Jobs -

Empowering
people through Programming

Finding insights for oneself/society/the world
in the local surrounding

Personal Relevance
Pursue own interests and questions

Diversity
Making programming accessible and usable for everyone

Self-Expression
Personal and meaningful artifacts that transport a message

Communicating
programming results in a transparent way, making them comprehensible and adaptable

Epistemic Programming

How to empower young people - and especially programming novices - to use programming as a means for gaining personally relevant insights?

Framework to describe goals of subject related education

Coping with affordances

thinking, coping with affordances

trial and error/experiments

Participation

creating, producing, participation

projects

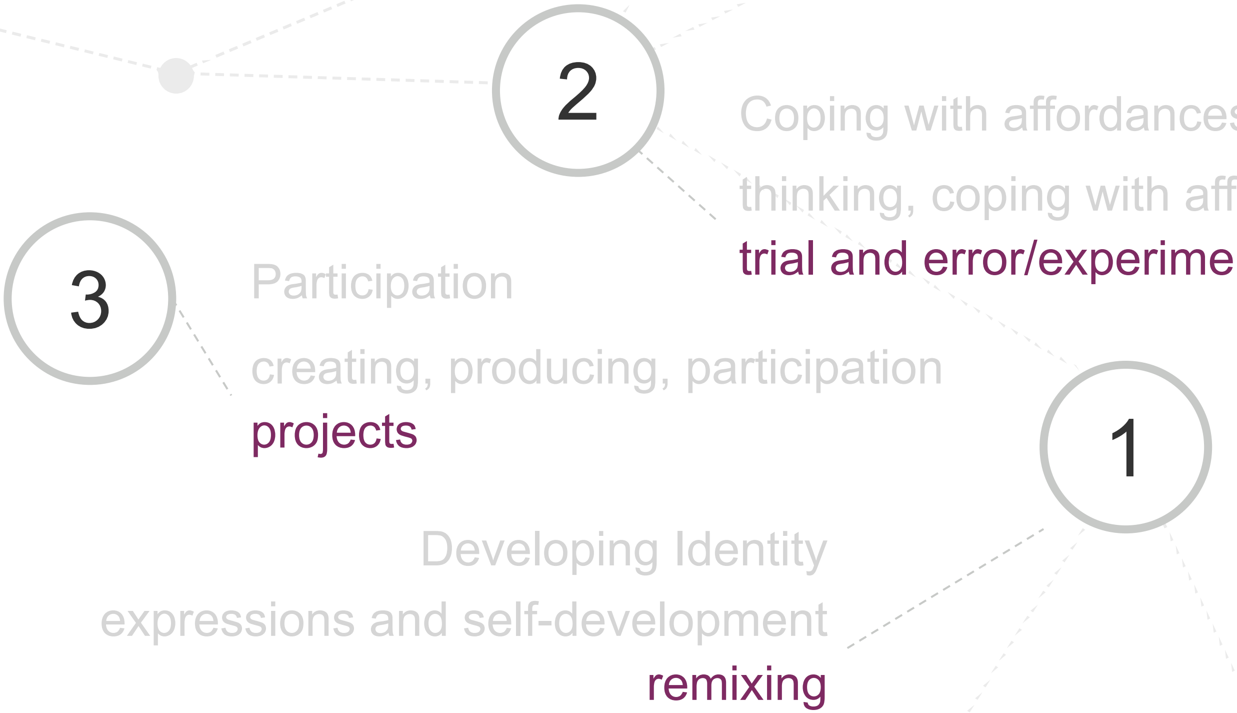
Developing Identity

expressions and self-development

remixing

Schulte (2013)

Framework to describe goals of subject related education



Coping with affordances

thinking, coping with affordances
trial and error/experiments

Participation

creating, producing, participation
projects

Developing Identity
expressions and self-development
remixing

Schulte (2013)

Framework to describe goals of subject related education

1 remixing

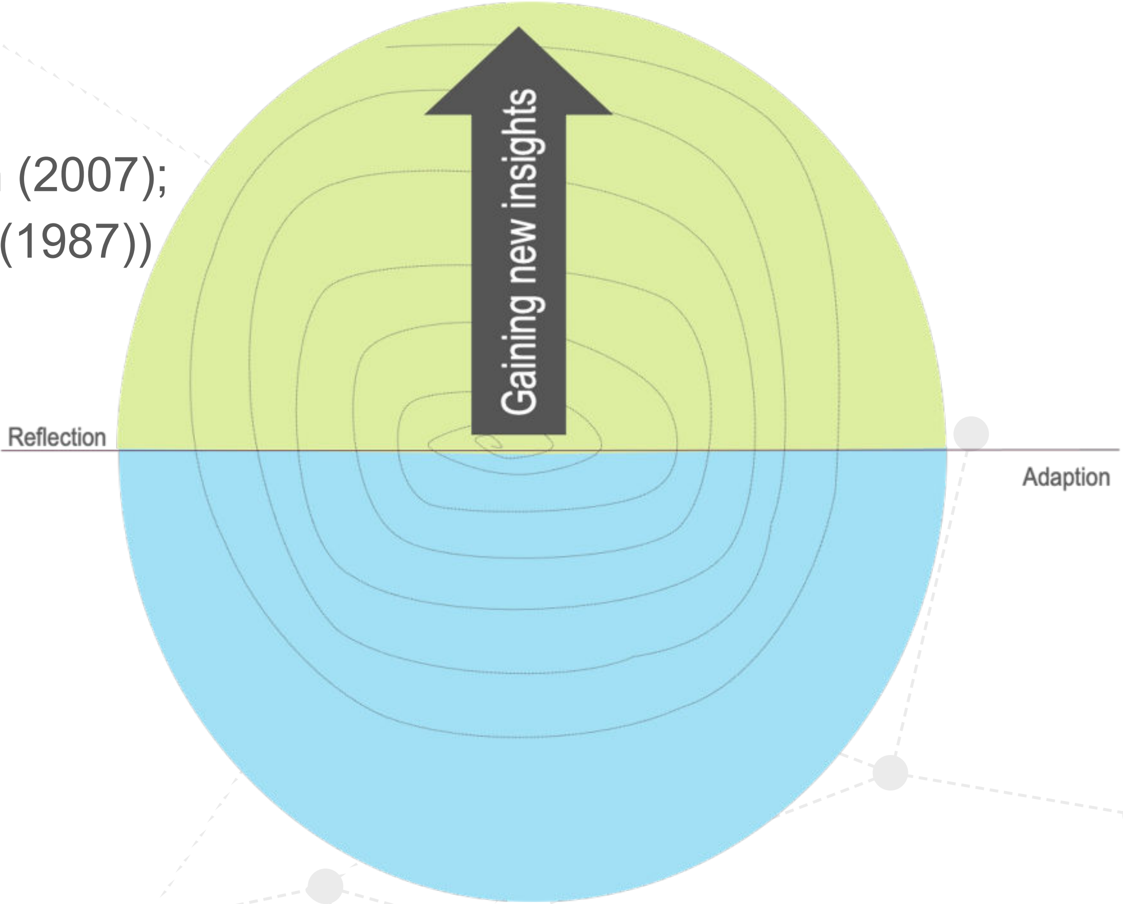
2 trial and error/experiments

3 projects

Schulte (2013)

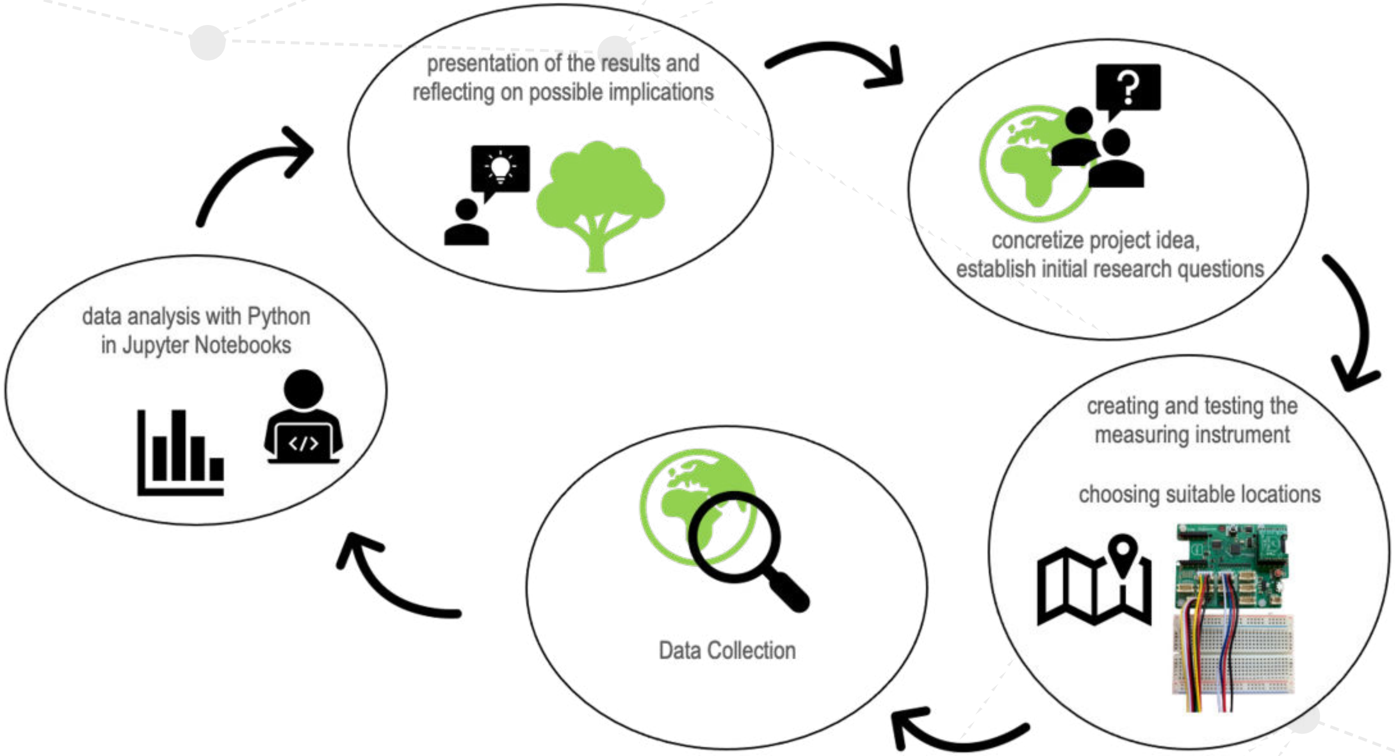
Scaffolding Epistemic Programming

- Using libraries and existing tools and methods
- Worked Examples
  (Atkinson et al. (2000);
  Caspersen and Bennedsen (2007);
  Merrienboer and Krammer (1987))
  - Trial & Error;
    Tinkering



Sven Hüsing, Carsten Schulte and Dan Verständig

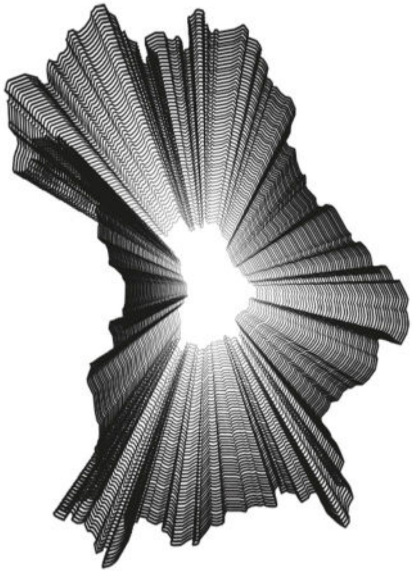
Exemplary Epistemic Programming Project


presentation of the results and reflecting on possible implications
concretize project idea, establish initial research questions
creating and testing the measuring instrument
choosing suitable locations
Data Collection
data analysis with Python in Jupyter Notebooks
?
</>

Sven Hüsing, Carsten Schulte and Dan Verständig

creative coding and self-expression

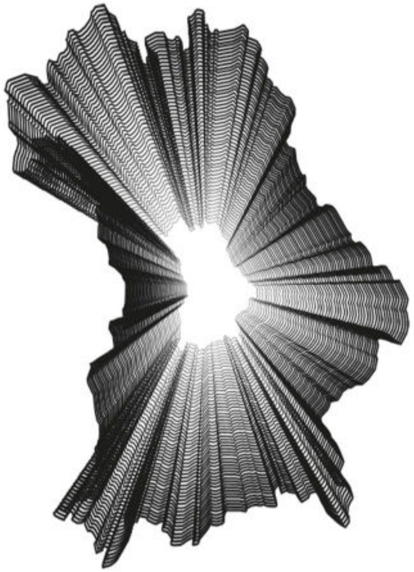
- creative and aesthetical self-expression
- individual problem solving
- sensing the performativity of code



Sven Hüsing, Carsten Schulte and Dan Verständig

creative coding and self-expression

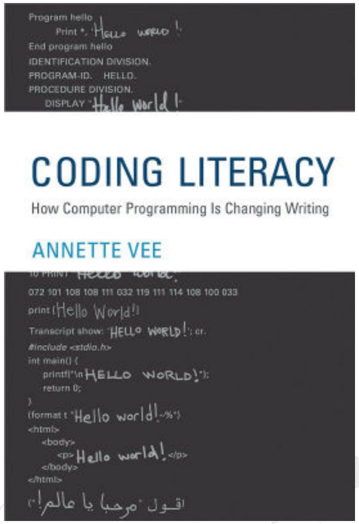
- creative and aesthetical self-expression
- individual problem solving
- sensing the performativity of code



```
10 PRINT CHR$(205.5+RND(1)); : GOTO 10
```

NICK MONTFORT, PATSY BAUDOIN,
JOHN BELL, IAN BOGOST, JEREMY DOUGLASS,
MARK C. MARINO, MICHAEL MATEAS,
CASEY REAS, MARK SAMPLE, NOAH VAWTER

Montfort et al. (2014)



```
Program hello
    Print *, 'Hello world!'
End program hello
IDENTIFICATION DIVISION.
PROGRAM-ID.   HELLO.
PROCEDURE DIVISION.
    DISPLAY 'Hello world!'
```

CODING LITERACY

How Computer Programming Is Changing Writing

ANNETTE VEE

```
10 PRINT "HELLO WORLD"
072 101 108 108 111 032 119 111 114 108 100 033
print ('Hello World!')
Transcript show: 'HELLO WORLD!'; cr.
#include <stdio.h>
int main() {
    printf("\nHELLO WORLD!");
    return 0;
}
(format t "Hello world!~%")
<html>
    <body>
        <p>Hello world!</p>
    </body>
</html>
(اقـول "مرحبا يا عالم!")
```

Vee (2017)

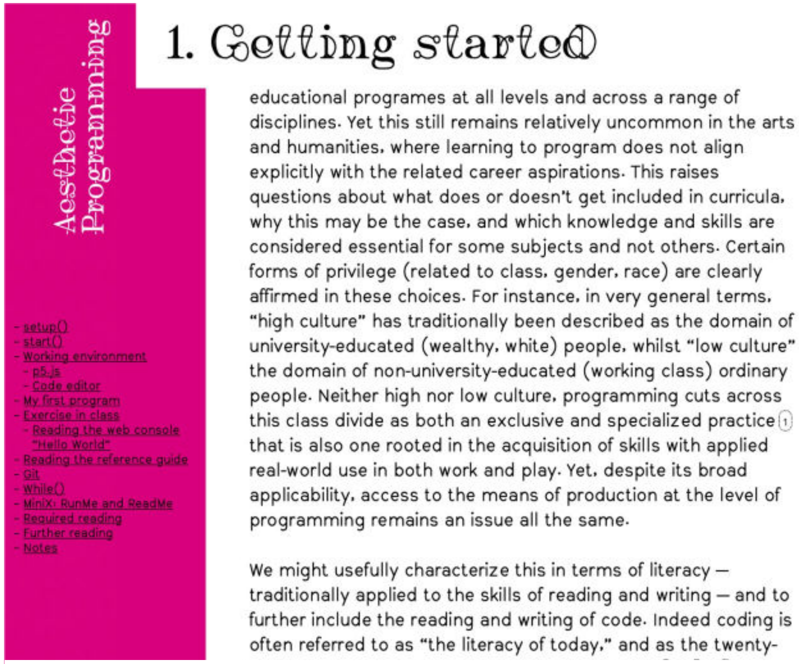
aesthetic programming

A Handbook of Software Studies

Winnie Soon
Geoff Cox

Soon & Cox (2020)

creative coding and self-expression

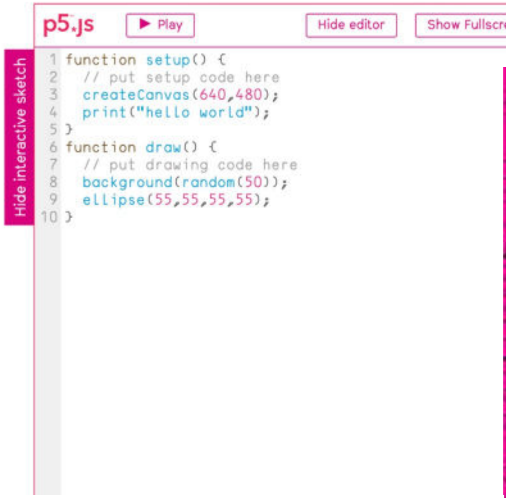


Aesthetic Programming

1. Getting started

educational programes at all levels and across a range of disciplines. Yet this still remains relatively uncommon in the arts and humanities, where learning to program does not align explicitly with the related career aspirations. This raises questions about what does or doesn't get included in curricula, why this may be the case, and which knowledge and skills are considered essential for some subjects and not others. Certain forms of privilege (related to class, gender, race) are clearly affirmed in these choices. For instance, in very general terms, "high culture" has traditionally been described as the domain of university-educated (wealthy, white) people, whilst "low culture" the domain of non-university-educated (working class) ordinary people. Neither high nor low culture, programming cuts across this class divide as both an exclusive and specialized practice[1] that is also one rooted in the acquisition of skills with applied real-world use in both work and play. Yet, despite its broad applicability, access to the means of production at the level of programming remains an issue all the same.

We might usefully characterize this in terms of literacy — traditionally applied to the skills of reading and writing — and to further include the reading and writing of code. Indeed coding is often referred to as "the literacy of today," and as the twenty-

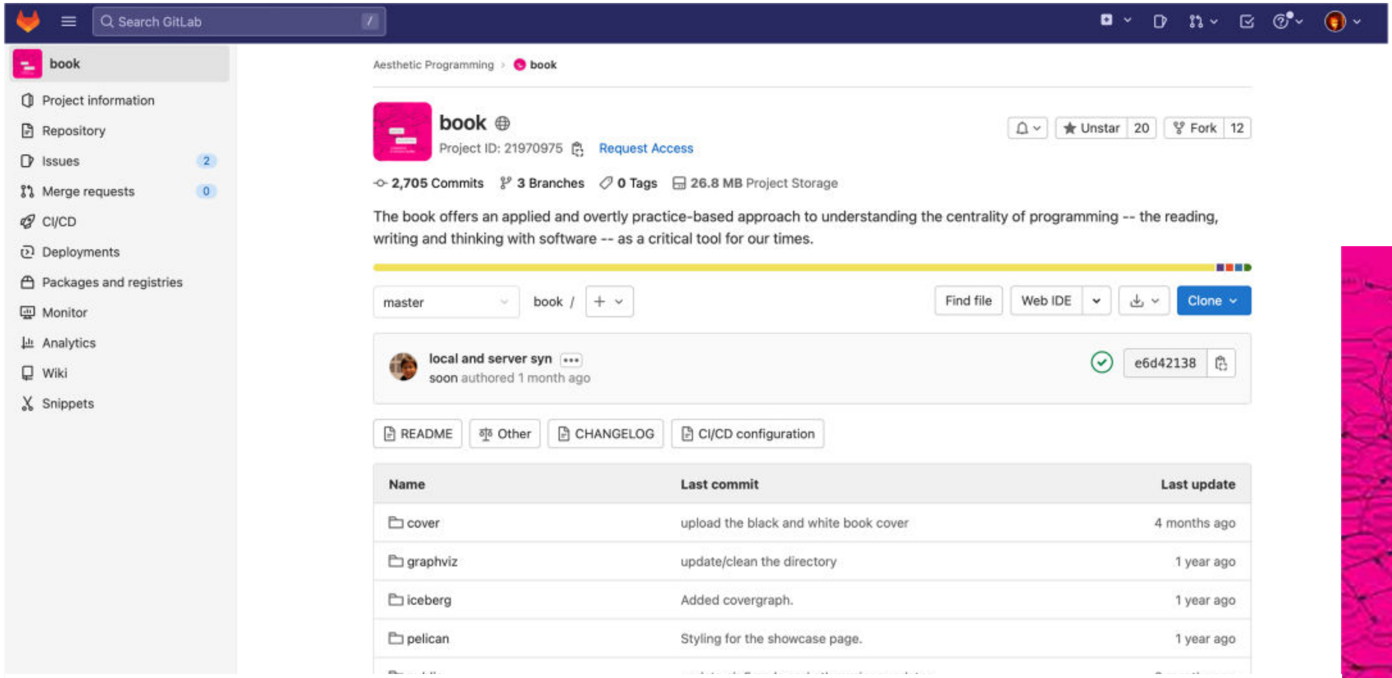


```
function setup() {
  // put setup code here
  createCanvas(640,480);
  print("hello world");
}
function draw() {
  // put drawing code here
  background(random(50));
  ellipse(55,55,55,55);
}
```

aesthetic-programming.net

Soon & Cox (2020)

creative coding and self-expression



Search GitLab

book

- Project information
- Repository
- Issues 2
- Merge requests 0
- CI/CD
- Deployments
- Packages and registries
- Monitor
- Analytics
- Wiki
- Snippets

Aesthetic Programming › book

book

Project ID: 21970975 Request Access

Unstar 20 Fork 12

2,705 Commits 3 Branches 0 Tags 26.8 MB Project Storage

The book offers an applied and overtly practice-based approach to understanding the centrality of programming -- the reading, writing and thinking with software -- as a critical tool for our times.

master book / + Find file Web IDE Clone

local and server syn

soon authored 1 month ago

e6d42138

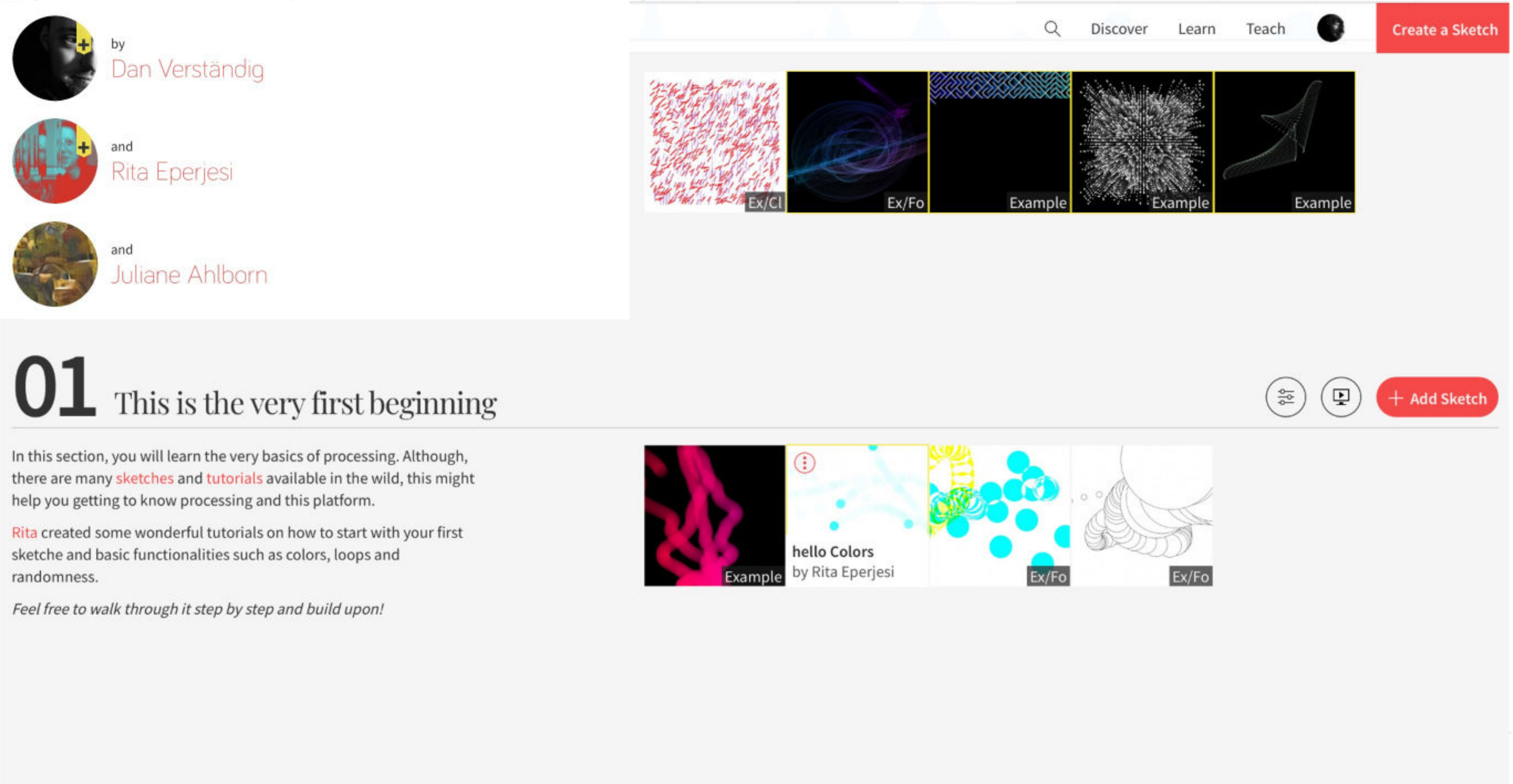
README Other CHANGELOG CI/CD configuration

Name	Last commit	Last update
cover	upload the black and white book cover	4 months ago
graphviz	update/clean the directory	1 year ago
iceberg	Added covergraph.	1 year ago
pelican	Styling for the showcase page.	1 year ago

gitlab.com/aesthetic-programming/book

Soon & Cox (2020)

creative coding and self-expression



by
Dan Verständig

and
Rita Eperjesi

and
Juliane Ahlborn

01 This is the very first beginning

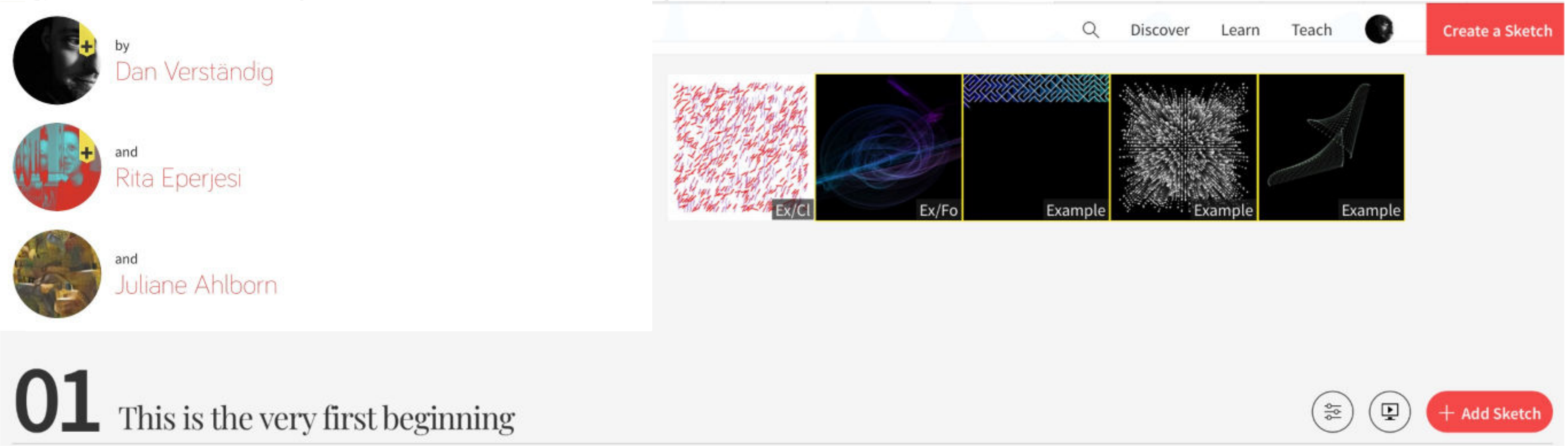
In this section, you will learn the very basics of processing. Although, there are many sketches and tutorials available in the wild, this might help you getting to know processing and this platform.

Rita created some wonderful tutorials on how to start with your first sketche and basic functionalities such as colors, loops and randomness.

Feel free to walk through it step by step and build upon!

hello Colors
by Rita Eperjesi

creative coding and self-expression



by Dan Verständig

and Rita Eperjesi

and Juliane Ahlborn

01 This is the very first beginning

02 Errors

03 Nature

04 Control

05 Power

Lessons Learned

- Programming is not only a means for Software-Engineers to create complex software products.
- Programming can also empower people - and especially programming novices - to learn something about their environments regarding topics of interest.
- People can express themselves, own ideas or visions through programming.
- This can happen…
  - in a spiral-like process of tinkering and reflecting.
  - …based on remixing/combining/adapting code.
  - …within student-driven (data-)projects.

References

- Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from Examples: Instructional Principles from the Worked Examples Research. Review of Educational Research, 70(2), 181–214.
- Bellin, D.; Simone, S.S..: The CRC card book. Addison Wesley, 1997.
- Bereiter, C. (1980). Development in writing. In L. W. Gregg & E. R. Steinberg (Eds.), Cognitive processes in writing (pp. 73–93). Erlbaum.
- Caspersen, M. E., & Bennedsen, J. (2007). Instructional Design of a Programming Course: A Learning Theoretic Approach. Proceedings of the Third International Workshop on Computing Education Research, 111–122.
- Chandler, D. (1995, September 18). Technological or Media Determinism. http://visual-memory.co.uk/daniel//Documents/tecdet/tecdet.html
- Eberle, F. (1996). Didaktik der Informatik bzw. einer informations- und kommunikationstechnologischen Bildung auf der Sekundarstufe II: Ziele und Inhalte, Bezug zu andern Fächern sowie unterrichtspraktische Handlungsempfehlungen (1. Aufl). Verl. für Berufsbildung Sauerländer.
- Forneck, H.-J. (1992). Bildung im informationstechnischen Zeitalter: Untersuchung der fachdidaktischen Entwicklung der informationstechnischen Bildung (1. Aufl). Sauerländer.
- GFD (2009). "Mindeststandards am Ende der Pflichtschulzeit" - Positionspapier der GFD. https://www.fachdidaktik.org/cms/download.php?cat=Veröffentlichungen&file=Mindeststandards_Ende_Pflichtschulzeit.pdf [acc. 2022-11-02]
- Hubwieser, P. (1999). Modellierung in der Schulinformatik. Log in, 19(1), 24–29.
- Merrienboer, J. J. G., & Krammer, H. P. M. (1987). Instructional strategies and tactics for the design of introductory computer programming courses in high school. Instructional Science, 16(3), 251–285.
- Montfort, N., Baudoin, P., Bell, J., Bogost, I., Douglass, J., Marino, M. C., Mateas, M., Reas, C., Sample, M., & Vawter, N. (2014). 10 PRINT CHR$(205.5+RND(1)); The MIT Press.
- Overstreet, M. (2022). Writing as extended mind: Recentering cognition, rethinking tool use. Computers and Composition, 63, 102700.
- Seoane, M. E., Greca, I. M., & Arriassecq, I. (2022). Epistemological aspects of computational simulations and their approach through educational simulations in high school. SIMULATION, 98(2), 87–102.
- Schubert, S. (1991). Fachdidaktische Fragen der Schulinformatik und (un)mögliche Antworten. In P. Gorny (Ed.), Informatik und Schule 1991 (Vol. 292, pp. 27–33). Springer Berlin Heidelberg.
- Schulte, C. (2001). Vom Modellieren zum Gestalten–Objektorientierung als Impuls für einen neuen Informatikunterricht. Informatica Didactica, 3.
- Schulte, C. (2013). Reflections on the Role of Programming in Primary and Secondary Computing Education. Proceedings of the 8th Workshop in Primary and Secondary Computing Education, 17–24.
- Schulte, C., & Budde, L. (2018). A Framework for Computing Education: Hybrid Interaction System: The need for a bigger picture in computing education. 18th Koli Calling International Conference on Computing Education Research (Koli Calling '18), 18, 10.
- Soon, W., & Cox, G. (2021). Aesthetic programming: A handbook of software studies. Open Humanities Press.
- Vee, A. (2017). Coding literacy: How computer programming is changing writing. The MIT Press.
- Wolfram, C. (2020). The math(s) fix: An education blueprint for the AI age. Wolfram Media, Inc.
- Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering computational literacy in science classrooms. Communications of the ACM, 57(8), 24–28.