PX/26

Universität Potsdam

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Literate Exploratory Programming for Asynchronous Collaboration

Christoph Thiede, Tom Beckmann, Marcel Taeumel, Robert Hirschfeld

PX/26 Workshop, 2026-03-16

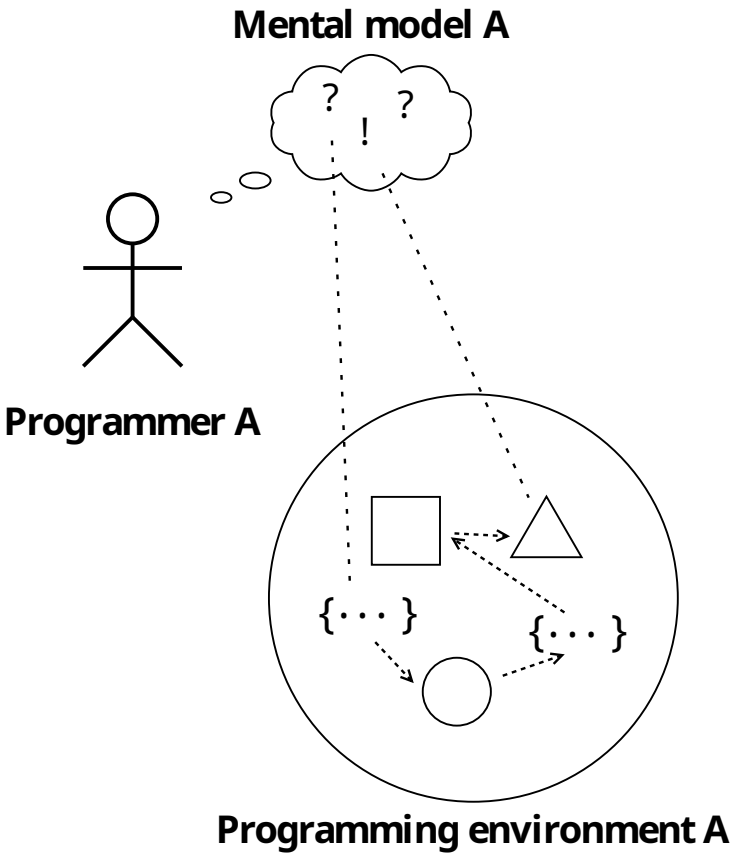
Software Architecture Group, Hasso Plattner Institute, Potsdam, Germany

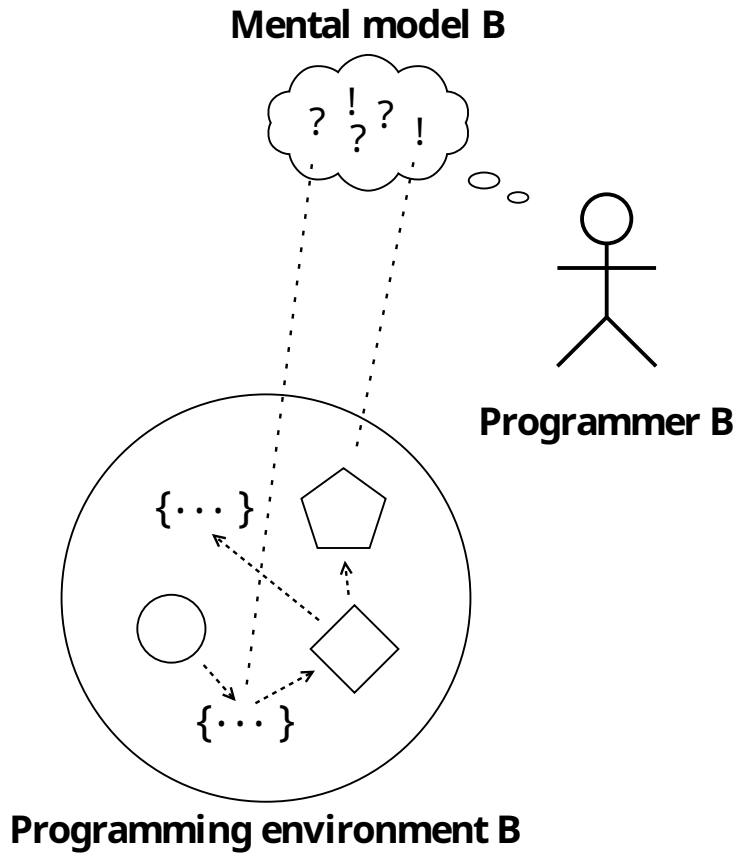
hpi.uni-potsdam.de/swa

Research Question

How can we improve
asynchronous collaboration
for exploratory programmers
without disrupting their flow?

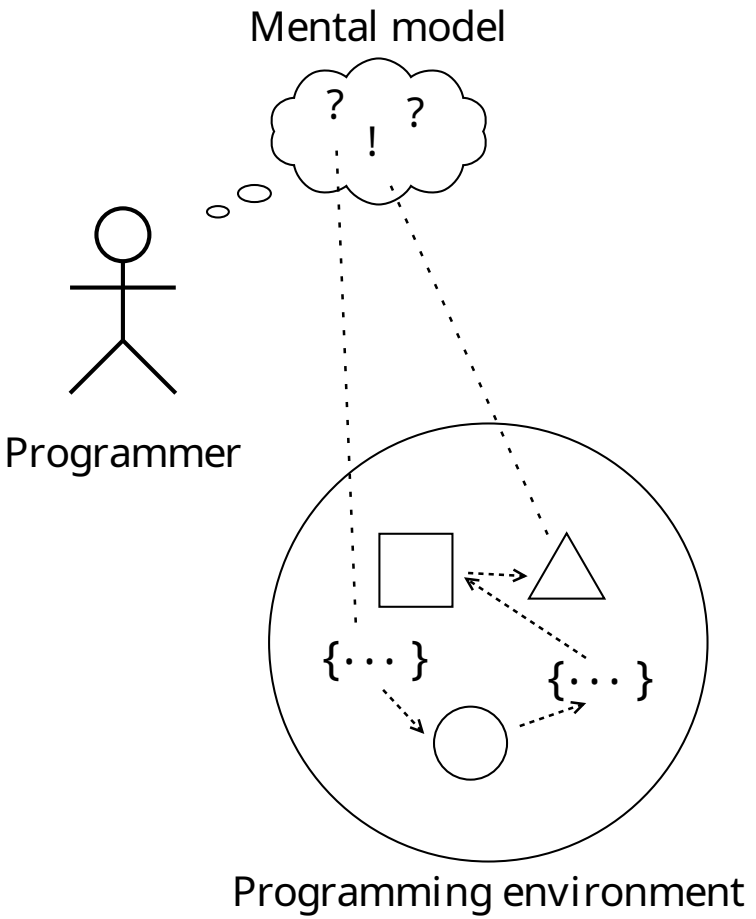
Motivation


Mental model A
? ? !
Programmer A
{ · · · }
{ · · · }
Programming environment A


Mental model B
? ! ? ? !
Programmer B
{ · · · }
{ · · · }
Programming environment B

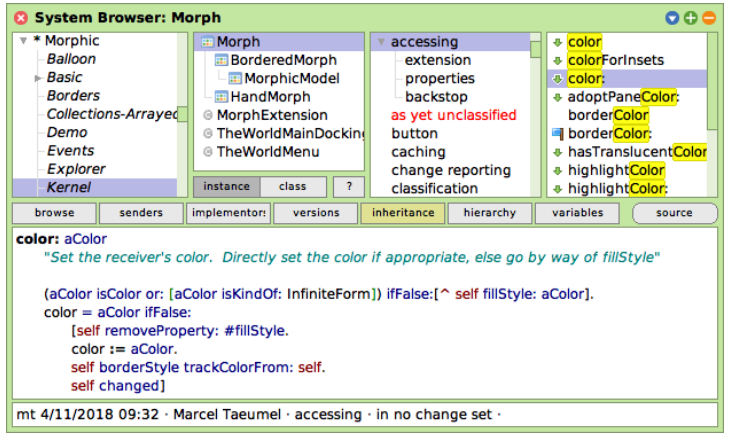
Background: Exploratory Programming

- Programmers gradually develop understanding or solution for a problem [Kery2017, Sandberg1988]
- Conversation between programmer and system through programming environment [Taeumel2022]
- Scientific metaphor: ask questions, conduct experiments, interpret results
- Interact with and navigate between artifacts (methods, runtime objects, debugger stack frames, ...)

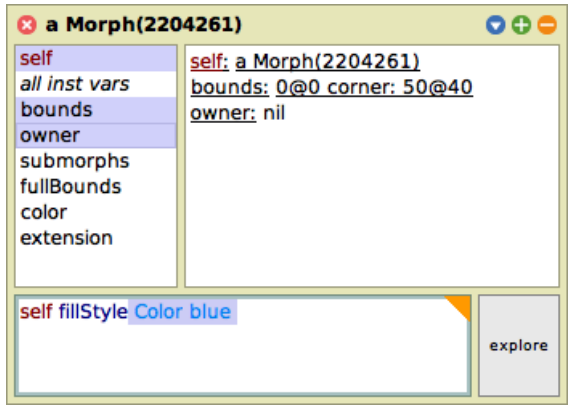


Example: Exploratory Programming in Squeak/Smalltalk

Fine-grained tool windows provide access to runtime artifacts, can be flexibly juxtaposed on the desktop/canvas, and allow for unrestrained navigation.

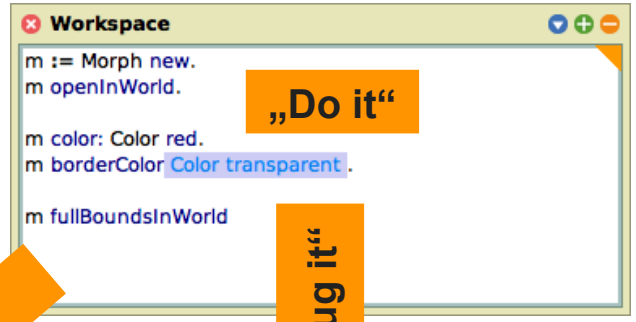


System Browser
Navigate and edit code in a structured interface



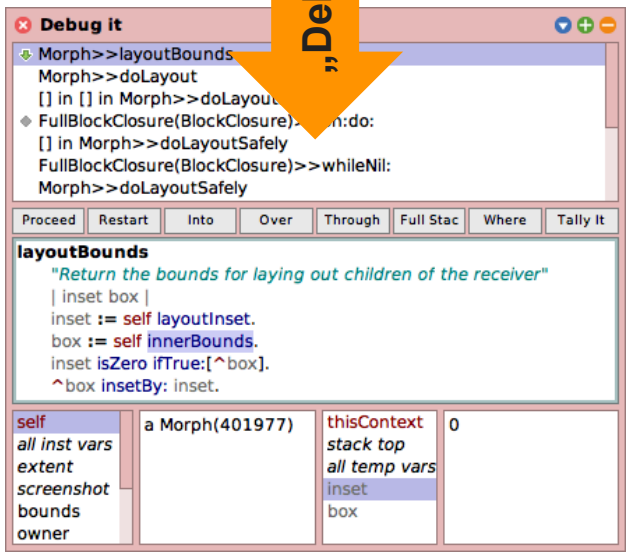
Inspector
Explore and change properties of runtime objects

„Inspect it“



„Debug it“

Workspace
Run experiments, inspect objects, start debuggers

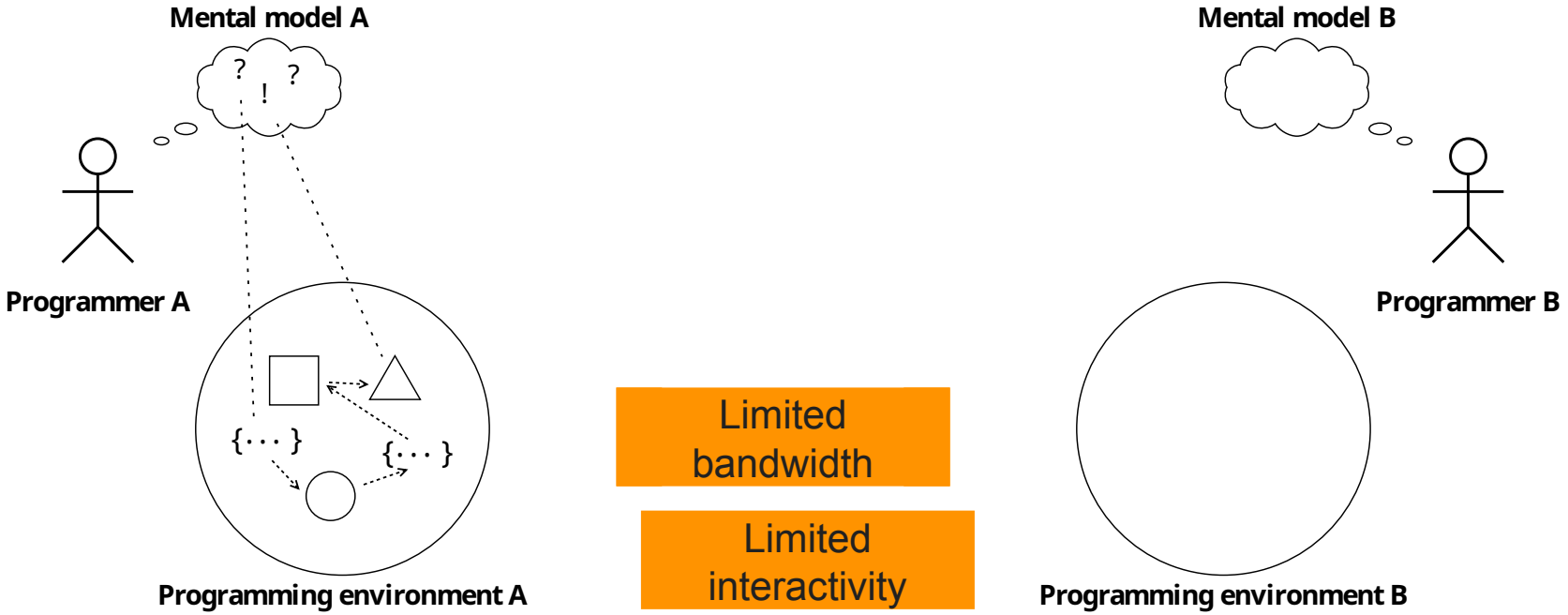


Debugger
Execute multiple code snippets step-wise, side-by-side

[https://squeak.org]

Problem

How can programmers share exploratory programming sessions with each other?



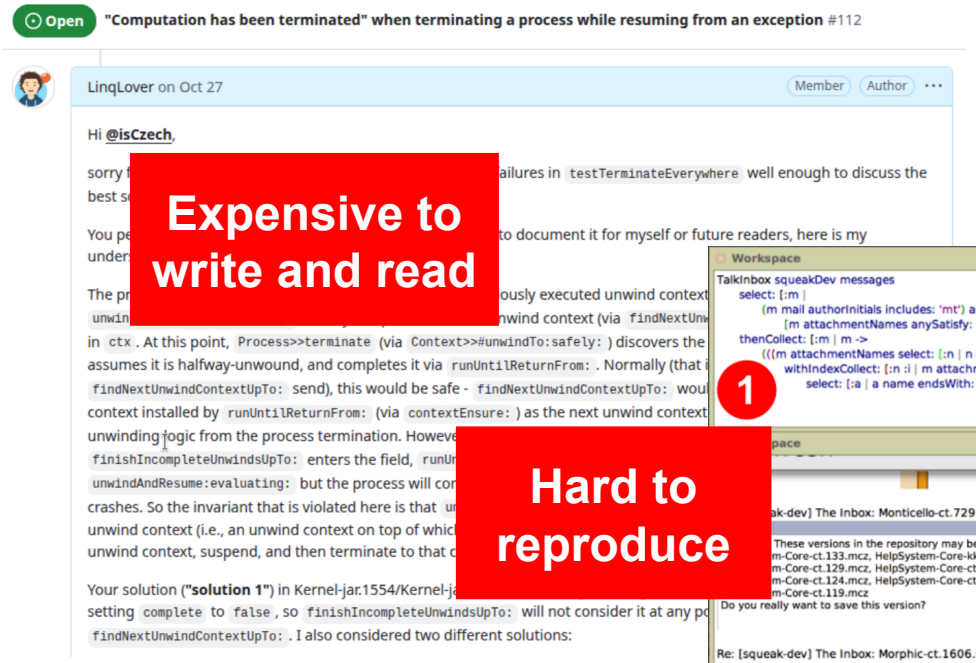
Goal: Share
- Ideas and approaches
- Reproducible experiments
- Observations
- Learnings, conclusions, next steps

Communication requirements:
- Distilled
- Self-explanatory
- Clear reading order

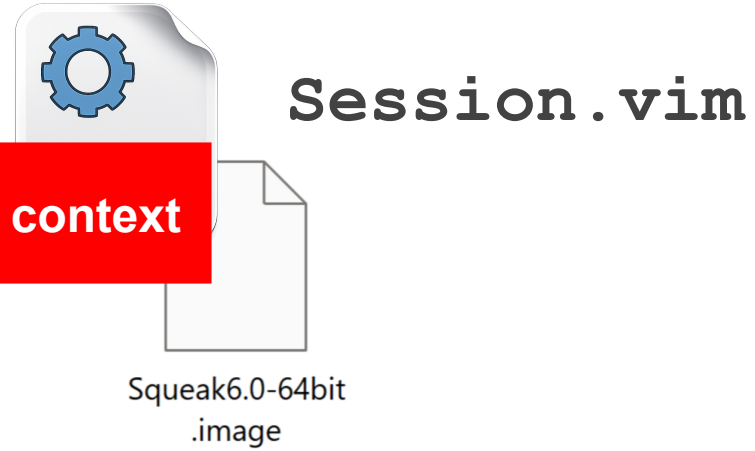
Problem

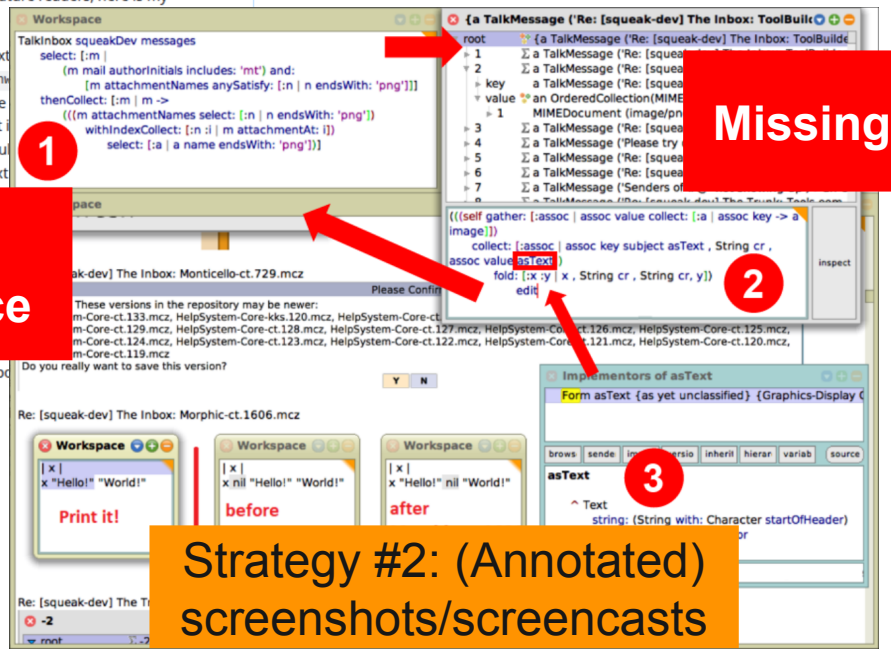
How can programmers share exploratory programming sessions with each other?

Strategy #1: Detailed prose



Strategy #3: Environment snapshots





Literate Programming to the Rescue?

- Literate programming: interweave program(ming) and explanation [Knuth1984]
- Modern forms: computational notebooks [Kery2018], active essays [Yamamiya2009], ...

- Strengths:
  - Linear reading order
  - Reproducible
- Weaknesses for exploration:
  - Linearity restricts multitasking [Wang2019]
  - Too many details/rejected attempts -> chaotic/overwhelming [Rule2018]

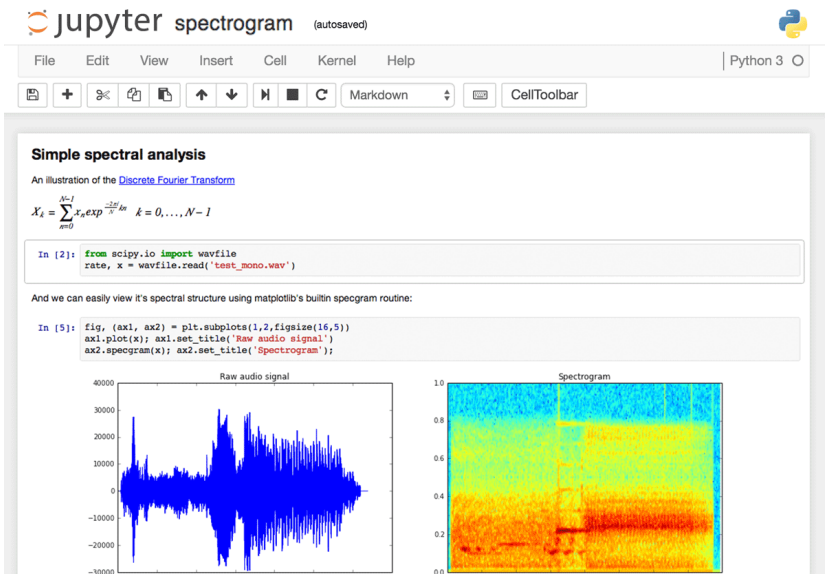
In the following, we will use a fixed, commutative ring $\mathbb{A}$. We are specifically interested in commutative algebras over that ring, so let us introduce a short name for those and let us use '$\mathbb{A}$' to refer to the $\mathbb{A}$-algebra $\mathbb{A}$.

```
module SpecExamples (𝔸asRing : CommRing {ℓ}) where
  𝔸-Alg = CommAlgebra 𝔸asRing
  𝔸 = CommAlgebraExamples.initial 𝔸asRing
```

The synthetic spectrum of an $\mathbb{A}$-algebra R, Spec R, is supposed to be a space such that the ring of functions on Spec R is R. Moreover, Spec R, should be so small and rigid, that the only homomorphisms $R \to \mathbb{A}$ are evaluations at points. The latter can be turned around to give a definition:

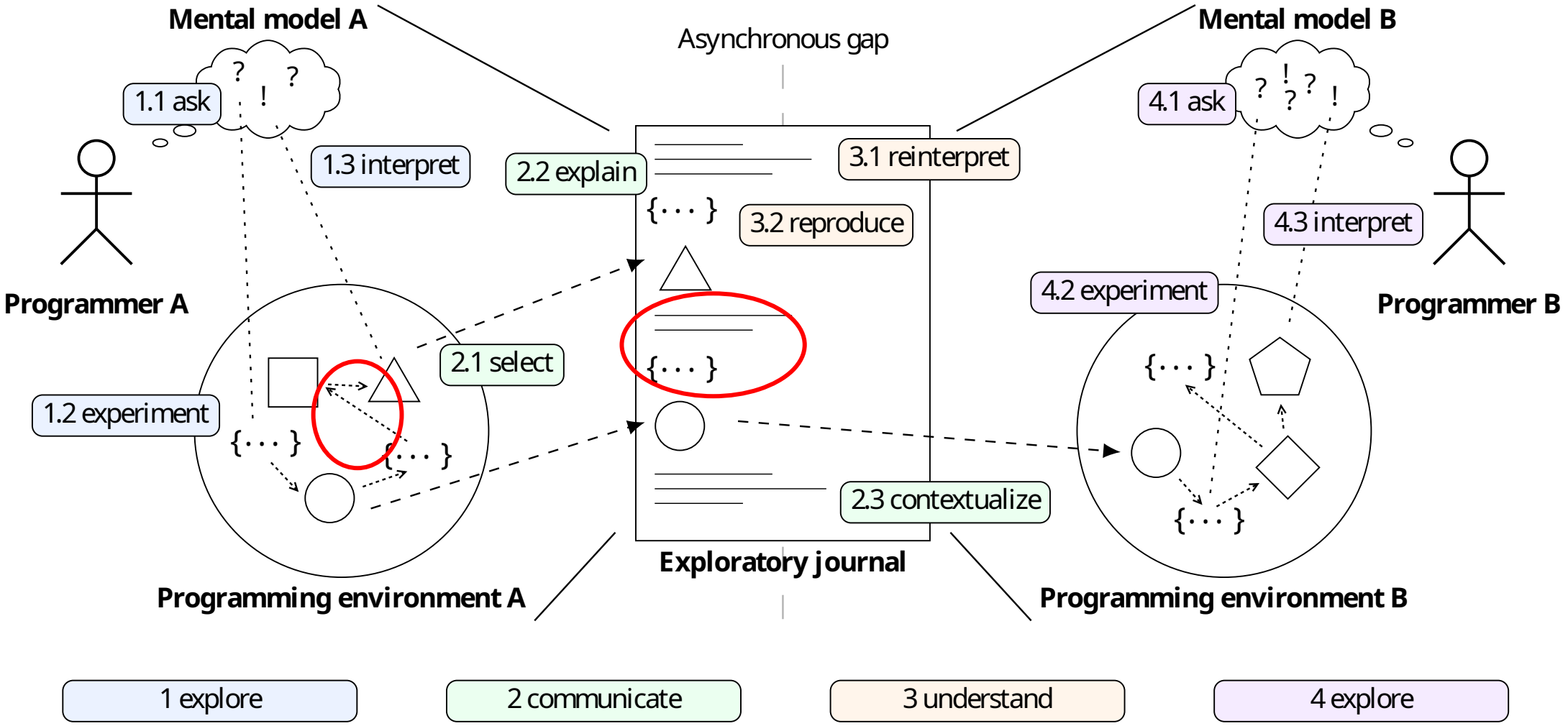
```
Hom : 𝔸-Alg → 𝔸-Alg → Type ℓ
Hom R S = CAlgHom R S

Spec : 𝔸-Alg → Type ℓ
Spec R = Hom R 𝔸
```

https://schneide.blog/2021/01/25/literate-formal-math/



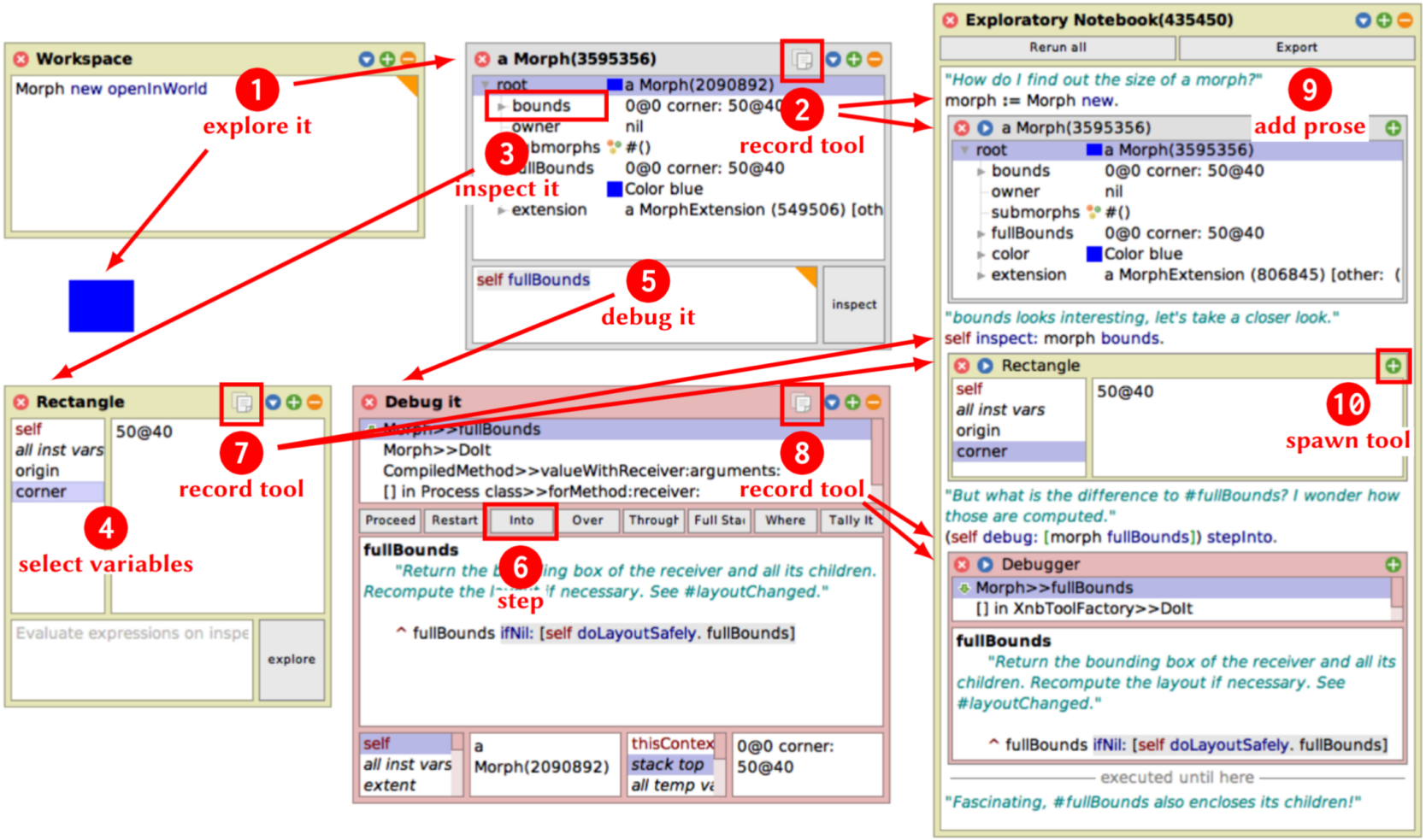
https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts/

Approach: Literate Exploratory Programming



Demo

- https://zenodo.org/records/18889420



Discussion

Preliminary experience report

- Better trade-off between unbound exploration and communication
  - Preserve multitasking and spatial flexibility of traditional ExP environment
  - Automatically track reproducible experiments
- High experience of immediacy due to tool reuse and arrangement in journals [Ungar1997]
- Also useful as „brain dumps“ for self-organization

Limitations

- Requires fine-grained tools for single artifacts
- Cannot automatically select relevant side effects for observations
- Explanation of ideas remains intrinsic overhead

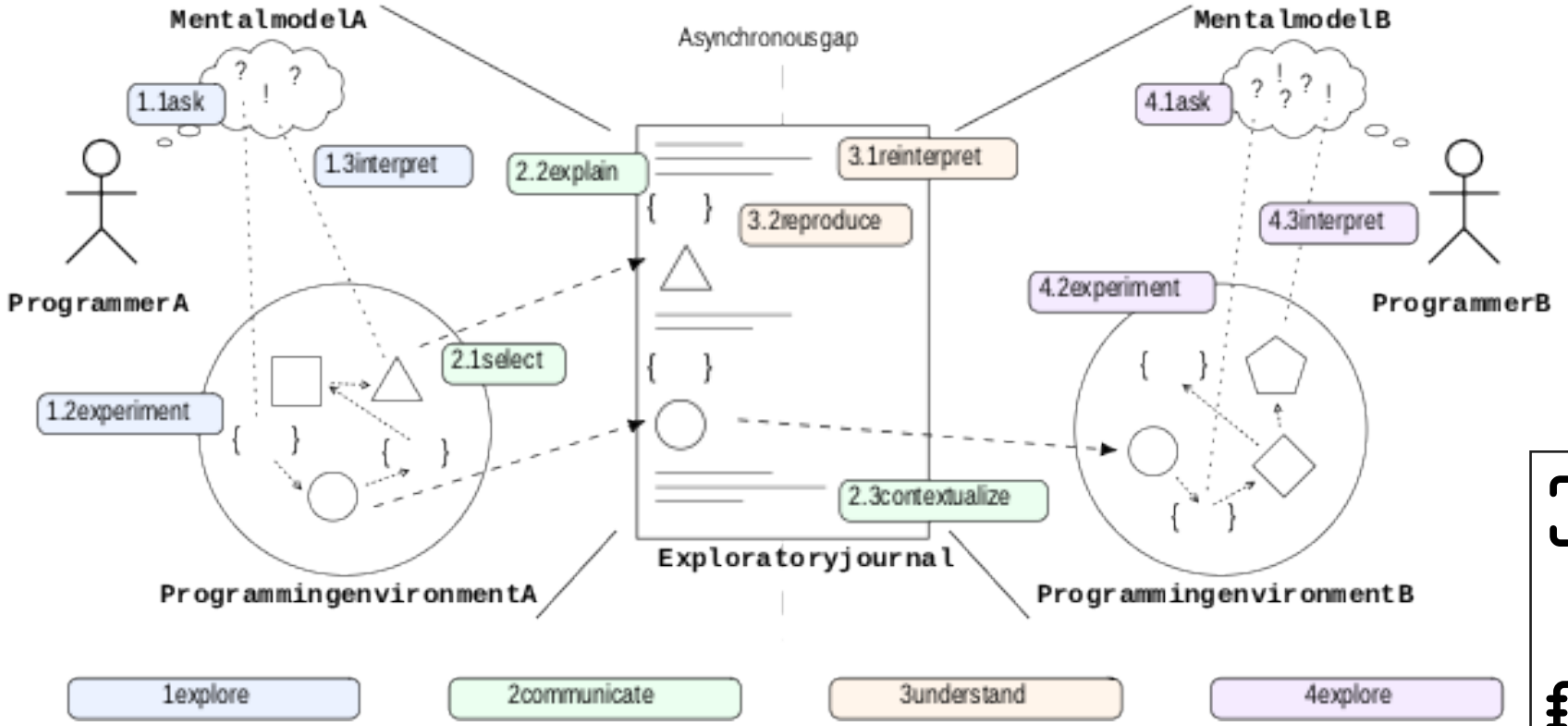
Read the paper for the details :)

Future Work

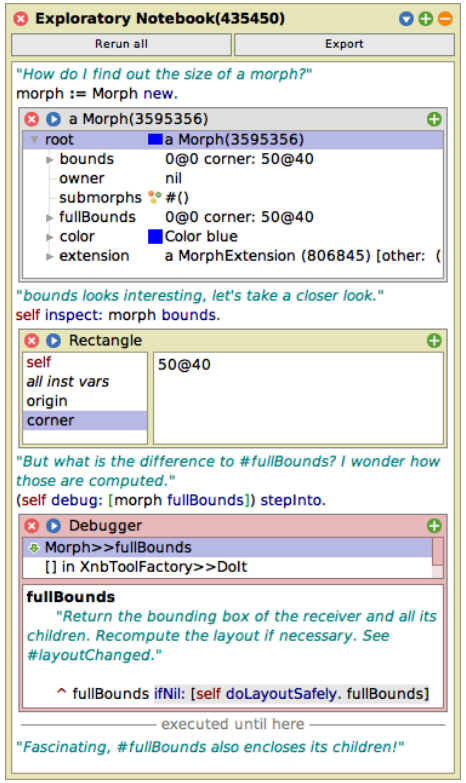
- How could we enrich the structure of exploratory journals? [Lincke2008, Beckmann2025]
- Could exploratory journals be used in documentation/training settings?
- Could we transform LXP into a means for real-time collaborative exploratory programming?

Conclusion

How can we improve asynchronous collaboration for exploratory programmers without disrupting their flow?



Try out our web demo!



Tracking mechanism

- Record dependencies between artifacts and experiments
- Extract relevant artifacts

Exploratory journal

- Prose
- Executable experiments
- Embedded tools for artifacts

Want to discuss further thoughts?
✉ christoph.thiede@hpi.de :-)

Reading

- Christoph Thiede, Tom Beckmann, Marcel Taeumel, and Robert Hirschfeld. 2026. Literate Exploratory Programming for Asynchronous Collaboration. In *Companion Proceedings of the 10th International Conference on the Art, Science, and Engineering of Programming (‹Programming› Companion '26), March 16–20, 2026, Munich, Germany*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3801119.3801129 (to appear)
- https://github.com/hpi-swa-lab/exploratory-notebooks

Literature

[Beckmann2025] Tom Beckmann, Joana Bergsiek, Eva Krebs, Toni Mattis, Stefan Ramson, Martin C. Rinard, and Robert Hirschfeld. 2025. Probing the Design Space: Parallel Versions for Exploratory Programming. *The Art, Science, and Engineering of Programming* 10, 1 (feb 2025), 33 pages. doi:10.22152/programming-journal.org/2025/10/5

[Kery2017] Mary Beth Kery and Brad A. Myers. 2017. Exploring Exploratory Programming. In *Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, Raleigh, NC, USA, 25–29. doi:10.1109/VLHCC.2017.8103446

[Kery2018] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. 2018. The Story in the Notebook: Exploratory Data Science using a Literate Programming Tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal, QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–11. doi:10.1145/3173574.3173748

[Knuth1984] Donald E. Knuth. 1984. Literate Programming. *Comput. J.* 27, 2 (jan 1984), 97–111. doi:10.1093/comjnl/27.2.97

[Lincke2008] Jens Lincke, Robert Hirschfeld, Michael Rüger, and Maic Masuch. 2008. SophieScript - Active Content in Multimedia Documents. In *Sixth International Conference on Creating, Connecting and Collaborating through Computing (CS '08)*. IEEE Computer Society, USA, 21–28. doi:10.1109/C5.2008.12

[Rein2025] Patrick Rein, Stefan Ramson, Tom Beckmann, and Robert Hirschfeld. 2025. An Information Foraging Interpretation of Liveness. In *Symposium on Visual Languages and Human-Centric Computing* (Raleigh, NC, USA) *(VL/HCC)*. IEEE Computer Society, Los Alamitos, CA, USA, 128–138. doi:10.1109/VL-HCC65237.2025.00022

[Rule2018] Adam Rule, Aurélien Tabard, and James D. Hollan. 2018. Exploration and Explanation in Computational Notebooks. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. ACM, New York, NY, USA, 1–12. doi:10.1145/3173574.3173606

[Sandberg1988] David W. Sandberg. 1988. Smalltalk and Exploratory Programming. *SIGPLAN Notices* 23, 10 (1988), 85–92. doi:10.1145/51607.51614

[Taeumel2022] Marcel Taeumel, Jens Lincke, Patrick Rein, and Robert Hirschfeld. 2022. A Pattern Language of an Exploratory Programming Workspace. In *Design Thinking Research: Achieving Real Innovation*, Christoph Meinel and Larry Leifer (Eds.). Springer, Cham, 111–145. doi:10.1007/978-3-031-09297-8_7

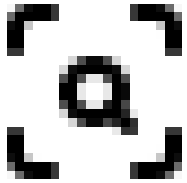
[Ungar1997] David Ungar, Henry Lieberman, and Christopher Fry. 1997. Debugging and the Experience of Immediacy. *Commun. ACM* 40, 4 (apr 1997), 38–43. doi:10.1145/248448.248457

[Wang2019] April Yi Wang, Anant Mittal, Christopher Brooks, and Steve Oney. 2019. How Data Scientists Use Computational Notebooks for Real-Time Collaboration. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW, Article 39 (nov 2019), 30 pages. doi:10.1145/3359141

[Yamamiya2009] Takashi Yamamiya, Alessandro Warth, and Ted Kaehler. 2009. Active Essays on the Web. In *Seventh International Conference on Creating, Connecting and Collaborating through Computing (CS '09)*. IEEE Computer Society, Los Alamitos, CA, USA, 3–10. doi:10.1109/C5.2009.10

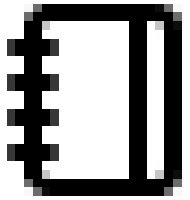
Implementation

- Integrate literate exploratory programming into traditional exploratory programming system
- Two components:



Tracking mechanism

- Record dependencies between artifacts and experiments
- Extract relevant artifacts

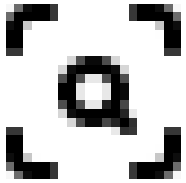


Exploratory journal

- Prose
- Executable experiments
- Embedded tools for artifacts

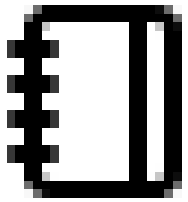
Read the paper for the details :)

Implementation for Squeak/Smalltalk



Tracking mechanism

- Associate each tool with linked list of navigation steps/experiments (messages or code snippets)
- Patch standard tools/editors to track interactions and code execution



Exploratory journal

- Variation of Squeak‘s default workspace tool
- Experiments: Smalltalk code snippets executed against small tooling DSL
- Prose: Comments
- Artifacts: Inline compacted tools to represent print-it results of experiments
- Import from/export as HTML flavor

Read the paper for the details :)