**Jonathan Aldrich**, Professor  
School of Computer Science  
Carnegie Mellon University

**Interests**: designing languages to improve quality and productivity at scale

**Questions for the community:**
- How can we gather evidence about how programmers work at scale and in professional settings? How should this impact language design?

**Ideas for the community:**
- Language theory, SE theory, and diverse forms of empirical evidence should complement one another in language design
- Software Engineering in particular has a critical and underappreciated role to play
Research Interests:

“Bridging the gap between Programming Languages and Human Computer Interaction”

• Evaluation and Usability of Programming Languages and Tools, Human Aspects of Software Engineering and Programming, Software Visualization, API Usability

Questions for Community:

• How to improve **usability** of PL and tools to make programmers more efficient?
• How to **evaluate** PL and tools more effectively for novices, end-users, and experts?
• How to get programming language designers to **adopt** human centred design methods?

Ideas for Community:

• Understand what methods to **employ and adopt** to evaluate PL and tools
• Create **community standards** for evaluating PL and tools
Ameer Armaly

• University of Notre Dame
• aarmaly@nd.edu
• http://cse.nd.edu/~aarmaly
• Interests: accessibility, program comprehension, low-level programming

• Questions:
  • How do we think about addressing the problems faced by programmers with disabilities?
  • How can we improve the auditory reading experience for programming languages?
  • How can we make alternative representations of programs more accessible?
Johannes Bechberger (uqddy@student.kit.edu)

- Master student at KIT
- ...with focus on
  - software engineering
  - compiler construction
  - language design
  - parallel computing
- Question to the community

*How to develop strict type systems that are usable and expressive?*
Research Interests:
• Compiler error messages: roles in learning, and enhancement of
• Novice programmer behaviour: how can we make learning more effective
• (Almost) anything to do with CS1 (or 0)

Questions:
• How can we make tools (existing or otherwise) most compatible with learning?

Ideas:
• I would like to discuss studying the readability of compiler error messages, and other improvements to novice tools/environments
Andrew Begel, Senior Researcher
Microsoft Research (http://www.andrewbegel.com)

**Research:** I study software engineers at Microsoft to find out what makes them tick. Then I employ user-centered design to build tools to help them improve themselves.

**Recent work:**
- Learning about and helping autistic software developers
- Exploring developer cognition and affect with eye tracking, fMRI, and biometrics
- Investigating the organizational dynamics of software teams
- Improving the process of tech transfer from research to product groups

**Question:**
- What can we learn about the problems in programming languages and tool design by observing extremely short-duration cognitive errors?
Tanja Blascheck
Aviz, Inria Saclay, France

Main Research Interests
• Information Visualization and Visual Analytics
• Eye Tracking, Interaction Logs, Think-Aloud
• Evaluation

Your Questions for the Community
• How can visualization experts help?
• How can eye tracking experts help?
• What are common scenarios/questions/studies?
• What (other) biometric data could be of interest?

Your Ideas for the Community
• Studying program language design using biometric data
• Combining eye tracking with interaction logs and think-aloud
• Analyzing multimodal data with visualizations
• Integrating visualizations in program language design

tanja.blascheck@inria.fr
Interested in:
• Program editor interface design
• Recording and analysing programming activity

Questions/Ideas:
• How do you programmatically analyse programming activity?
• Why don’t all professionals use an IDE?
• Should languages be designed with IDEs in mind?
MICHAEL COBLENZ

- Fourth-year PhD student, Computer Science Department, Carnegie Mellon University
- Former Apple software engineer (8 years)
- Research: human-centered programming language design and evaluation
  - Immutability; blockchain programming language
- How can we design languages with powerful features that are as usable as possible in the long term?
Igor Crk
icrk@siue.edu

• Interest:
  Indicators of human performance measurable by brain activity (EEG)
  • Expertise(?)
  • Task difficulty/effort(?)

• Questions:
  How are you accounting for variability of expertise?
  Is there a need for something like a standard set of tasks against which study participants are grouped?
  What might something like that look like?
  Is it even reliably doable?
John Daughtry: Google

Recently
Google web APIs developer experience infrastructure

Currently
Istio - open platform to connect, manage, & secure microservices
Endpoints - Web API management on Google Cloud

Long-term Interest
How all types of APIs of all types are designed
- Broad conceptualization of API, including libraries, SDKs, and web
- Broad conceptualization of design, including surface/policy/deployment/ops

Questions
Learning a language is in large part learning its APIs
API design is large part of enterprise software engineering
Config is a crucial part of APIs (e.g., causes most outages)
Fabian Deitelhoff, PhD Student

• Social Navigation & Guidance
  • Programming education (students)
  • Eye tracking & behavioral data of learners
  • Aggregate this data and use it (hidden gaze sharing)
  • Try to help learners with data-based learning hints

• Topics of high interest
  • Towards better visual languages (debugger, units of measure, …)
  • Educational programming languages
  • Live programming
  • Combination of (visual) languages and hardware (Mindstorms)
Rob DeLine, Microsoft Research

I’ve done programming language research (years ago).

I’ve studied developers and other software engineers for 25 years.

Currently, I’m working on tools to teach machine learning to kids and makers.

Food for thought

Methodology: The effect size of PL design is small, while individual variability is high.

Safety gap: More non-experts programmers putting software everywhere (IoT).

The AI Renaissance: How much software will be trained rather than written?

Adoption: Why are so many languages designed by non-PL-experts? C, JS, HTML, R, ...
My work applies theories from the **learning sciences** and techniques from **HCI** in the domain of **computing education**.

**Food for Thought:**

- What makes programming cognitively difficult? (And what's unique about it wrt any other domain?)

- How do languages and tools shape how we interact with code? (And how might we better leverage that while learning?)

- In what ways do novices and experts reason about code differently? (And what does that say for how we teach?)

**Relevant Projects:**

- Cognitive load and expertise effects in type system usage
- Learning to program as a non-native English speaker
- Engagement and creativity in novice web design
- K-12 computing teacher training
Andrew T. Duchowski
Clemson University

- My Main Research Interests
  - eye tracking, graphics, HCI, visual perception

- My questions for the Community
  - which languages are worth knowing / teaching?
    - Python, SWIFT, C++, Objective C?
  - what is the latest in many-core / multi-core programming?
    - OpenMP? OpenCL?

- Ideas for improvement of programming language design
  - make parallelism more accessible, esp. GPU
    - make OpenCL / GLSL easier to use
Scott Fleming

Main Research Interests
• HCI of programming
• Programming tools and environments
• Program comprehension
• Information foraging theory for SE
• Code navigation

Questions for the Community
• What benchmarks for language evaluation?
• What language training/expertise should we assume?
• Practical info about eye tracking and other useful biometrics

Ideas for the Community
• Usability engineering ideas (e.g., ease-of-use vs. ease-of-learning)
• Theories of human behavior (e.g., information foraging theory, cognitive load theory)
Questions for the community
In thinking about young learners (ages: 5-18), and the teachers of those learners....

- How does PL design affect the development of one's ability to “think computationally”?
- Related: Does the level of abstraction in a language help or hinder development of one's ability to devise computational solutions to problems?
- Developmentally appropriate PL considerations: When thinking about “learnability” how should we take into account the broader context in which most of our young learners are learning (i.e. school)?
- Do we need to bridge a gap between languages for learners and professional languages? (yes)
- Mathematics: friend or foe?
- Are IDEs as or more important for learning than the language itself?
Reiner Hähnle

- Full Professor @ TU Darmstadt
- Leads Software Engineering Group

Research Interests

- Formal Specification and Verification (KeY Prover for Java)
- Debugging, Program Understanding (Symbolic Execution Debugger)
- Executable Modeling Languages (ABS)

Questions & Ideas

- Language Design for Analyzability / Verifiability
- Concurrent Programming optimized for Humans (not Machines)
Matthias Hauswirth
USI, Lugano, Switzerland
http://www.inf.usi.ch/faculty/hauswirth/

Research Topics:
Profiling (Performance)
Dynamic Analysis
Learning to Program
Repository Mining

Sabbatical: SRI Education (Learning Sciences)

Evaluate Collaboratory
http://evaluate.inf.usi.ch/

Open Letter to PC Chairs
http://evaluate.inf.usi.ch/letter-to-pc-chairs

Artifact Evaluation
http://evaluate.inf.usi.ch/artifacts/aea

Truth, The Whole, Nothing But
Pragmatic Guide to Assessing Empirical Evals

SIGPLAN Empirical Evaluation
http://sigplan.org/Resources/EmpiricalEvaluation/
"How does a language for non professional developers look like?"

End User Programming

"Is text the best paradigm?"

Felienne Hermans
TU Delft

Programming education

"How to design languages and tools for learning?"

"What can we learn from other fields?"
Johannes C. Hofmeister

- Python
- Good Software

Person-Centered Programming

- Why is there always bad code, but noone ever writes bad code?
- Why does visiting someone’s code feel like visiting someone else’s bedroom?
- Emotions, Intimacy, Knowledge Representation, Humanistic Psychology
- More at http://brains-on-code.org

Brain by A. Varghese @ The Noun Project
Maya Israel
University of Illinois-Urbana Champaign

• I study elementary & middle school CS ed with a focus on:
  • Integration into mathematics instruction
  • Universal Design for Learning (UDL) and metacognitive strategies that support students who struggle

• I use mixed-method research to study student learning, persistence, and collaborative problem solving.

• Questions of Interest:
  • In what ways do different programming languages either support or impede learning for students with learning disabilities?
  • Which pedagogical approaches promote engagement and learning for struggling learners during programming activities?
Ciera Jaspan

- Software Engineer at Google in Developer Tools
  - Tech Lead Manager of Engineering Productivity Research
  - Identifying inefficiencies in our development tools and processes
- Helping developers make tradeoffs
  - Which tool/language to use
  - Which design to implement
  - How many resources to provide for a project
  - Whether and when to do a migration
- When we create new language features, what is the cost of upgrading?
  - And what is the cost of staying still?
- How do we meet developers where they (and their code) already are?
Research interests

- theory of programming languages, including human factors
- education research to improve my own teaching
- development of evidence-based practice
- unfortunate tendency to delve into foundations
  - research methodology
  - philosophy of computing, philosophy of science
  - foundations of mathematics

Questions that bother me

- How to avoid a replication crisis?
- How to learn from the empirical evidence?
Andy J. Ko
Associate Professor, The Information School, University of Washington

• I’ve studied **programming** for ~20 years: how people learn it and do it, how to make it better

• Lately, I’ve been using **HCI** methods and **Software Engineering** techniques to answer **Computing Education** questions

• How much do programming languages matter?

• How much does evidence about PL matter?

• Can PL design compensate for poor skills?
I study how developers interact with code and design new programming tools.

- program comprehension, reverse engineering, code navigation, debugging, software design, microtask programming
- programming as asking and answering questions

How do we design programming experiences, rather than programming languages?

How do programming styles mediate the productivity impact of language features?

How do we measure the cost in complexity, as well as the potential benefit, in adding a new feature to a programming language?
Andrew Macvean: API Usability @ Google

Background
● UX Researcher. API Usability. Google for ~4 years
● Primary focus area is ‘web APIs’ and their ecosystem (client libraries, documentation, etc)

Primary Research Interests
● API Design Process
● Understanding and measuring API Usability
  ○ Scale
  ○ Recent focus on mining large datasets
● Developer segmentation

Questions, Ideas, Hopes
● Shared datasets of development activity
● Taxonomy of heuristics for human centered API/Programming language design
● Developer archetypes and models
Research Interests: Software Engineering, Maintenance, and Evolution. Program Comprehension, Analysis, Transformation, Open Source Software, Mining Software Repositories, Software Visualization

srcML - An infrastructure for the exploration, analysis and manipulation of source code

Languages for:
- Searching and exploring source code (srcQL)
- Transforming and manipulating source code (srcTL)

Questions: How to evaluate a language? How to compare two languages? Which is better? Why? Appears to be a difficult problem to address in general.

Current Research: Studying programmers via eye-trackers, Developing eye-tracking environments, Empirically examining program identifier naming wrt parts of speech, Automatic documentation and summarization of code
Amelia McNamara
Visiting Assistant Professor of Statistical and Data Sciences
Smith College
Northampton, MA USA

My goal is to make it easier for everyone to **do** and **understand** statistics (data science)

Main research interests:
- Statistical computing
- Statistics education
- Spatial statistics
- Data visualization

Questions for the community:
- Is programming language design different for DSLs?
- How can we better study the effects of language choices? In particular, how can we tease out how specific elements of existing languages do or don't support people?
Computer Science Education

• Cognitive Load
• Broadening Participation
• Increase Number of HS CS Teachers

I want a programming language that is easy for novices to learn and use. Better if it's also authentic.
• HCI II, School of Computer Science, CMU
• My research:
  – Research at the intersection of HCI and Software Engineering
  – Improving programming for expert, novice, and end-user programmers
  – Use HCI methods to better understand programmers’ real issues and barriers, across the entire life-cycle
  – Make programming easier and more correct by making it more natural
• Ideas/questions:
  – There are many user-centered methods. Which to use?
  – See our shared position paper
Lutz Prechelt
Freie Universität Berlin, Germany

- Former pertinent experimental research on:
  - Typing/type checking
  - Inheritance [*]
  - PSP
  - Design patterns
  - Comparison Java, C, C++, Perl, Python, Rexx, Tcl [*]
  - Comparison Java, PHP, Perl for web development [*]

- Main current interest:
  - Deciphering Pair Programming

- Question to group:
  - Are we making proper use of what evidence we have?

- Proposition:
  - We need more long-term research efforts!

- Other:
  - Review Quality Collector

[*] featured later
Sibylle Schupp
Professor
Hamburg University of Technology (TUHH)
Germany

Teaching
• Functional Programming
• Compiler Construction
• Software Engineering
• Software Verification & Testing

Research

- types, data-flow analysis
- timed automata, online model checking

- static, offline
- dynamic, online

Software quality

- GDPR (general data protection regulation)
- radiosurgery

Program language design
• Throwaway programs
• Uncertainties at source-code level

Programmers
• License to program in X
• Teaching only: PL-independent self-assessment for incoming students

Seminar
Questions & Ideas
Bonita Sharif, Associate Professor
Youngstown State University, Ohio, USA
bsharif@ysu.edu

• **Research Interests**: software engineering, *eye tracking*, program comprehension, *empirical studies*, software traceability, emotional awareness in development

• I study eye movement data from programmers working on software tasks such as bug fixes or feature additions.

• I am working on the design and development of the *iTrace infrastructure* that brings *eye tracking* into development environments to enable realistic eye-tracking studies using various software artifacts.

**Questions and Ideas**

• How do we make a language accessible to practitioners and educators?

• Do PL designers consider the human readability of a language during design?

• What is the impact of programming languages on programmer productivity?

• I am interested in contributing towards an evidence standard to support language design features.
Since I helped write the proposal, the objectives from there are essentially my own objectives for the meeting as well

- **Objective 1:** Form a community of scholars around programming language design based on evidence of programmer impact
- **Objective 2:** Create an organized set of priorities for new studies in the area
- **Objective 3:** Create an evidence standard to be used in human-centered studies on programming language design
- **Objective 4:** Discuss or create theories that can guide the community in understanding and framing the evidence gathered over time

My questions revolve especially about what kind of actions can be taken, especially, for objective 1 and 3. For example, what actions would need to be taken to establish the community in objective 1 and what specific procedures would we use to establish an evidence standard formally?
Walter F. Tichy  
walter.tichy@kit.edu  ps.ipd.kit.edu

Interests:

Software Engineering
Empirical Software Engineering
Parallel Programming
Programming for Everyone (in natural language)

Vision:

Close the semantic gap between what people can express and what computers can do.

Eliminate programming languages for end-user programming
P. Merlin Uesbeck  
University of Nevada, Las Vegas - United States

Ph.D. candidate with a focus on human-centered empirical studies on the impact of programming language design

Research interest:
• Evidence-based API design (specifically database access)
• Impact of polyglot programming

Question:
• What can we do to improve the evidence foundation of programming language design?

Ideas:
• Recommendations and guidelines for studies (design, execution, analysis, publication)
• Trial registration
Lea Verou

PhD student
MIT CSAIL
since 2014

W3C CSS Working Group,
former W3C staff
since 2012

Conference speaker/author
on Web development
since 2010

- What makes programming hard?
- How can we make Web programming accessible to novices?

research

Create interactive web applications
that manage, store and transform
data just by authoring HTML

https://mavo.io

leaverou@mit.edu
http://lea.verou.me