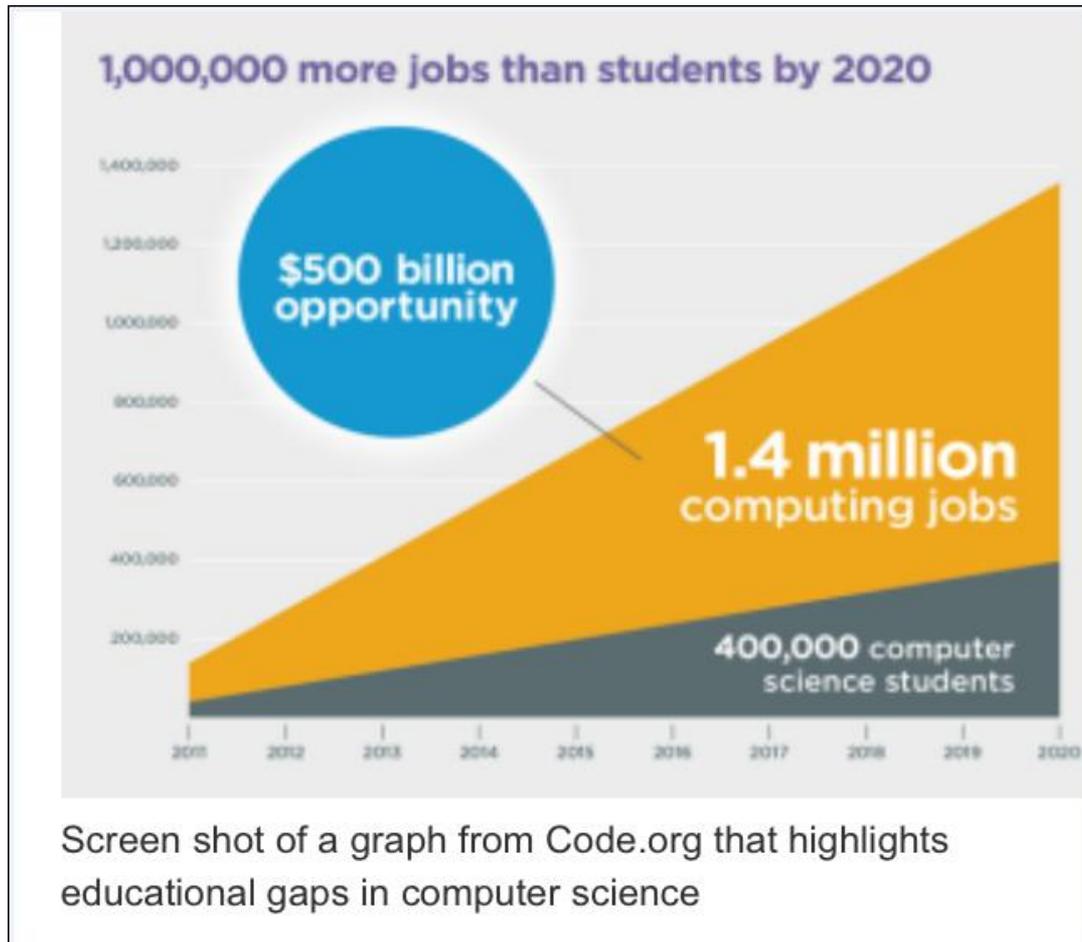


# Recruiting and Retaining Women in IT

- Background assumptions
  - Women are underrepresented in IT careers
  - Women may offer different strengths and needs than men as related to IT careers
  - The US may have as many as 1 million unfilled IT jobs in the next decade, based on current projections (2012 [STEM education report to the President](#) ?)(quoted by Philip Guo and Code.org)

# Code.org



# The Big Question:

- Is there a need, void, or opportunity in Madison to change our current, local environment, both via education and via career support, for women in IT?
- Especially targeting women software developers?
- If yes, how to implement?

Do you have a need to stay current  
in technology?

(Is that need currently well met?)

What is your ideal learning environment?

Would you attend CodeChix  
events?

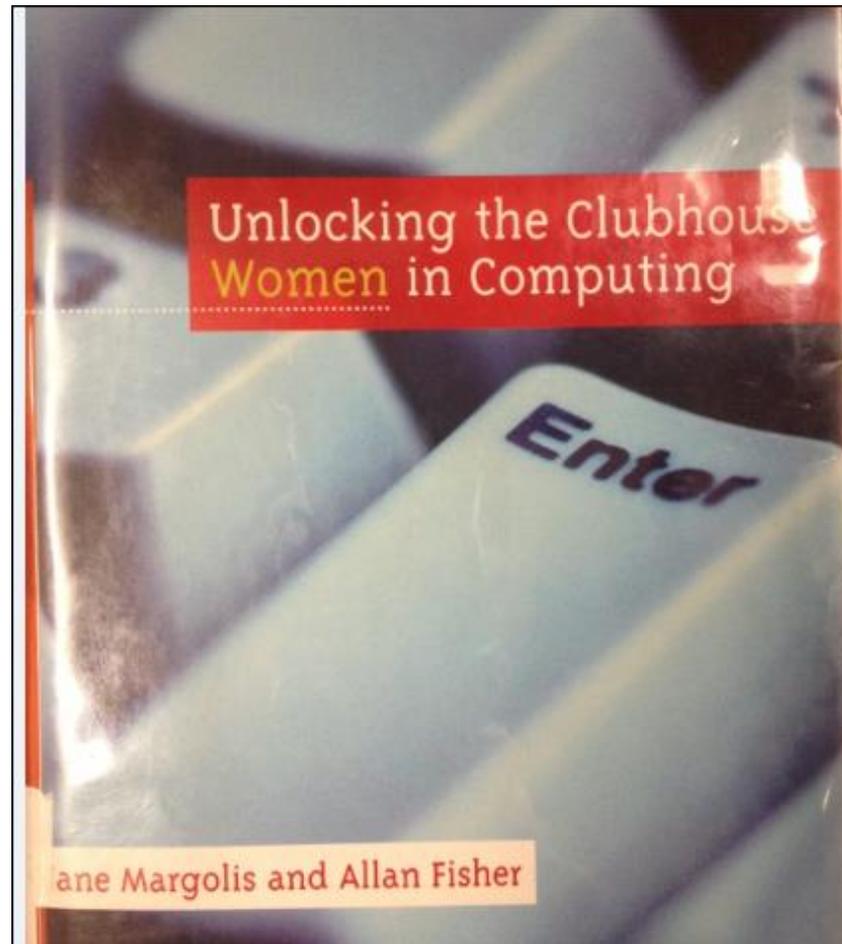
# My story

- Non-computer science major
- Y2K
- estgi/UWEX, DOA, DOT, WTCS
- 15 years as software developer

# Recent activities/ attempts

- Unsatisfying job/ looking for more meaning
- Technovation Challenge
- Meetup: Women in Tech
- MATC: IT Bootcamp for Girls
- Meetup: Women Coding For A Cause
- CodeChix.org
- Hacker School

# Unlocking the Clubhouse



# Unlocking the Clubhouse

- Published 2002 by MIT
- Based on Carnegie Mellon University (Pennsylvania, USA) computer science majors from 1995-1999
- Found trends, drew conclusions
- Implemented changes in high schools
- Implemented changes in colleges

# Self-confidence

- Girls enter college at the peak of self-confidence
- Self-confidence erodes very rapidly

# The Erosion of Confidence

- “Researchers on gender and math and science have found that self-confidence, not ability, is the significant difference between male and female science students.” (p. 81, Clubhouse)
- Declines more in selective schools
- Declines more for women than men
- “...Especially severe in historically male-dominated fields.”

# Small injuries hurt women more

- “Much prior research shows that female students in technical disciplines, perhaps partly because of their ‘outsider-ness,’ are especially vulnerable to poor teaching, inhospitable teaching environments (such as large classes), and unhelpful faculty.” (p. 83, Clubhouse)

# Persistence Roller Coaster

- “During our research we were often surprised by which students stayed in the program and which left. Especially in the first two years, many women ride an emotional roller coaster of certainty and doubt from term to term, indeed from week to week, and whether they decide to finish the ride or get off before it ends is unpredictable.” (p. 93, Clubhouse)

# The culture, not the individual, must change

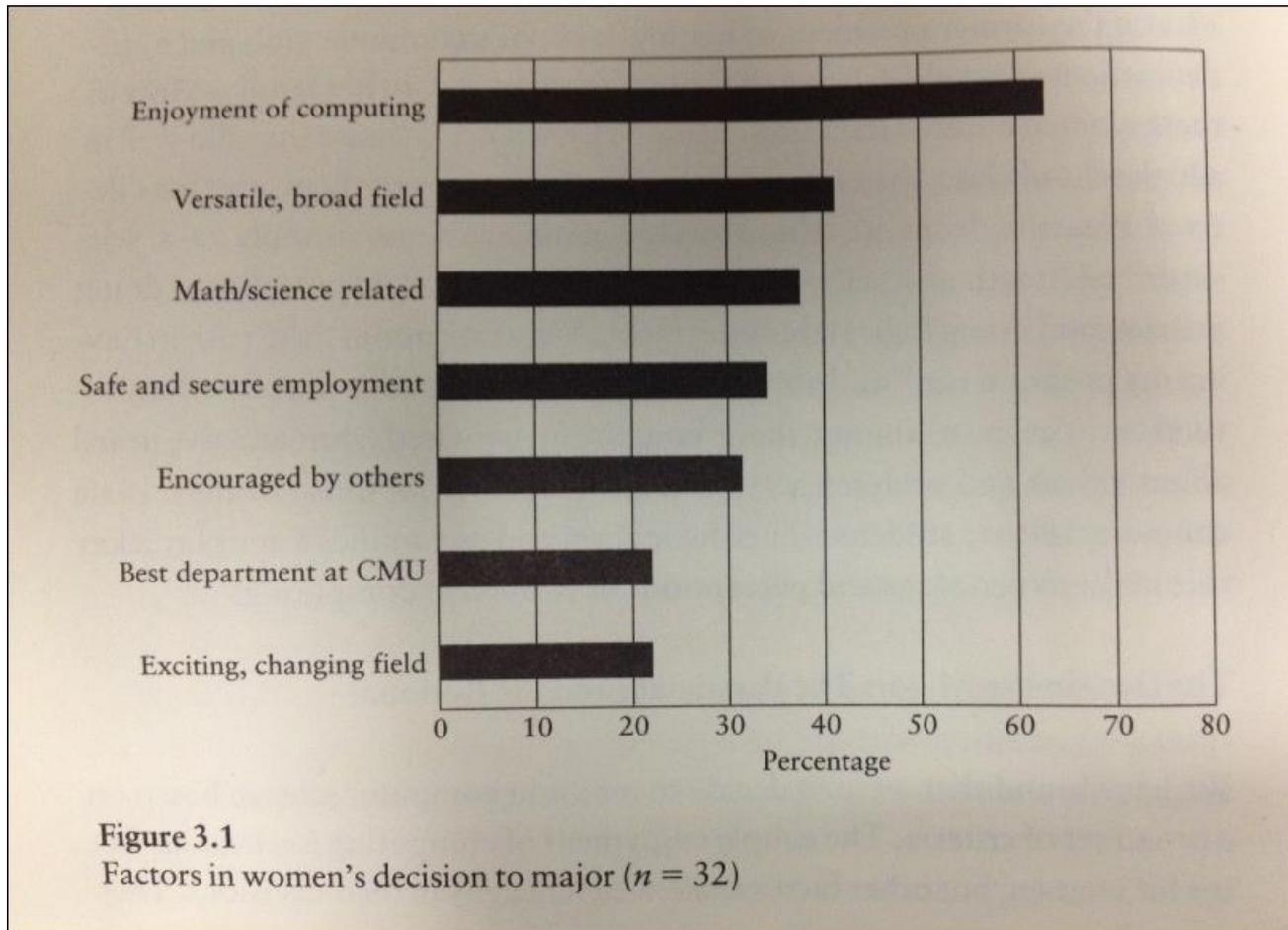
- “Attrition of women from computer science has been a significant problem...nationwide.”
- Women transfer out “at more than twice the rate of male students”
- (p. 91, Clubhouse)

- “While it may be tempting to assume that the difficulties expressed by women who leave the program are somehow unique to them, in fact the majority of women in the program, both those who leave and those who stay, express similar dissatisfaction with their peers, the culture of the discipline, and the teaching.” (p. 91, Clubhouse)

- “The persisters go through the same processes of self-doubt, fear, and anxiety as the leavers.” (p. 91, Clubhouse)

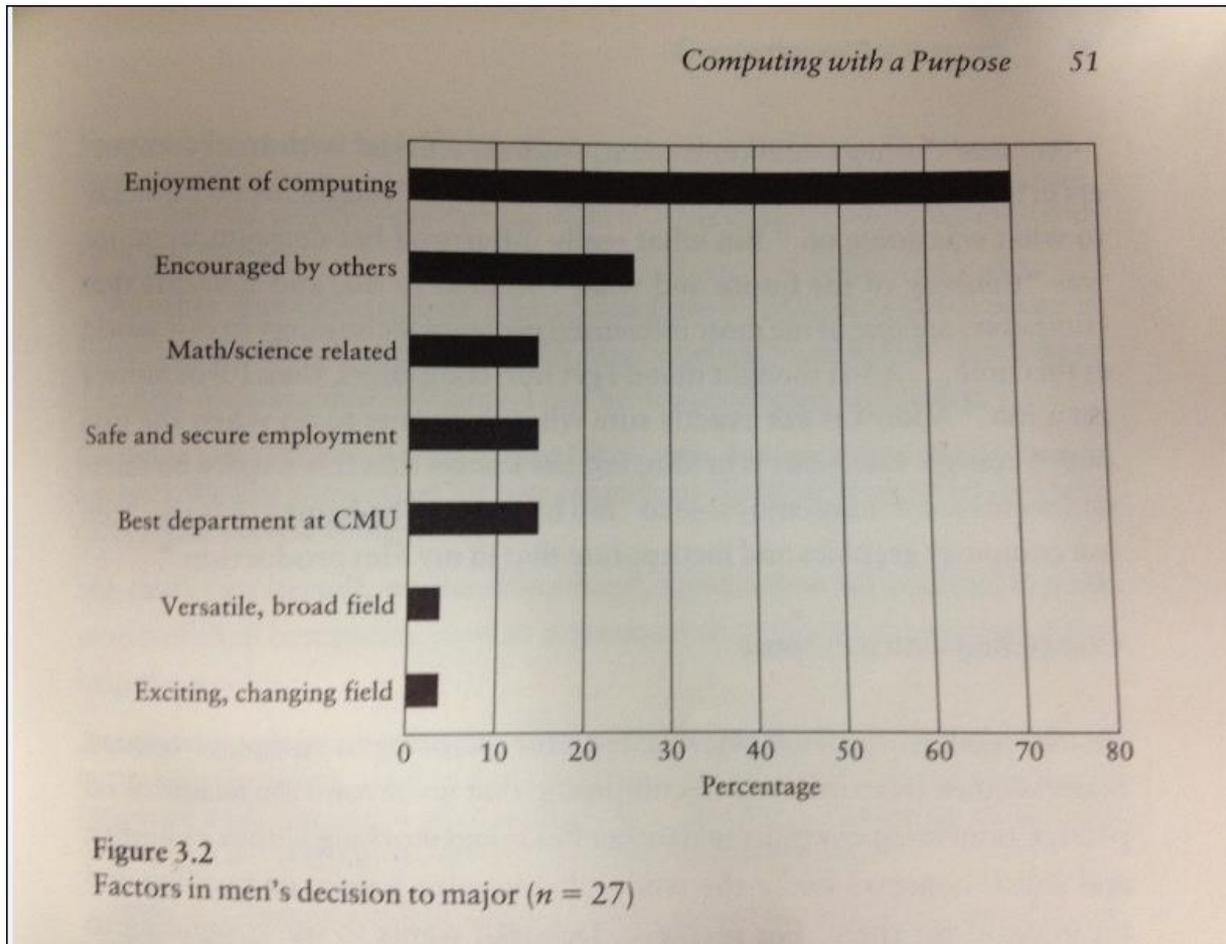
How can we help women  
persist in IT?

# Women's reasons for pursuing CS major



- It's helpful to ask WHY women and men pursue computer science.
- “Most women take a large number of factors into account: five of the seven categories we tabulated were mentioned by at least 30% of the women” (p. 50, Clubhouse)

# Men's reasons for pursuing CS major



- “In contrast, the only motivation listed by at least 30% of the men (in fact, by 70%) is the enjoyment of computing.” (p. 51, Clubhouse)

# Strengths brought to IT by females

- “In response to the perception by teachers that their students (largely male) tended to focus too much on the speed of their programs and the speed with which they were able to write them, we also provided a list... of additional goals of software design.” (p. 121, Clubhouse)
- [From the chapter in which high school teachers learned how to better attract and retain female students in computer science classes.]

# Strengths brought to IT by females

- “These ‘ilities’ are rarely appreciated by novice programmers, but almost always outrank speed in the real world and happen to be in line with many girls’ connections with computing:” (p. 121, Clubhouse)
  - Compatibility: working ... with other programs
  - Durability: outlasting changes in surrounding systems

# Strengths brought to IT by females

- Extensibility: making it easy to add new features and functions
- Maintainability
- Portability
- Readability
- Reliability
- Usability
- Utility: providing a useful solution to a problem

# Strengths brought to IT by males

- Magnetic Attraction
- “...a remarkably consistent picture emerges: more boys than girls experience an early passionate attachment to computers, whereas for most girls attachment is muted and is ‘one interest among many.’ These attachment differences help to shape students’, parents’, and teachers’ expectations that boys and men, *not* girls and women, will excel in and enjoy computing.” (p. 16, Clubhouse)

# What helps keep women in IT?

- Broaden culture to be more than 'male hacker'
  - “For some students, the image of a computer science major as someone who is myopically obsessed with computers is a perfect fit.” (p. 65, Clubhouse)
- Broaden culture to accept a more diffuse, less single-minded attachment to IT as a legitimate way of engaging with IT professionally

# Hacker definition

- “New Hacker’s Dictionary... computer geek as ‘withdrawn, relationally incompetent, sexually frustrated and desperately unhappy when not submerged in his or her craft.’... having ‘relatively little ability to identify emotionally with other people.’” (p. 67, Clubhouse)

- “It is important to remember that women who do not throw themselves into the computer world might not be inferior to men, but that sacrificing everything to computers might not be something that a psychologically healthy human being does. Perhaps men and women alike would be better off if some jobs and hacker cultures did not require giving up the rest of their lives.” (p. 73, Clubhouse)

# What helps keep women in IT?

- Broaden the culture so women don't feel they need to change who they are to fit in.
- 'Contextualize' IT – show why it matters:  
“To spark and engage girls' interest and engagement in computing, we believe that computer science must be viewed as a fully human discipline that, while highly technical, is linked to other arenas and people.” (p. 120, Clubhouse)

# Strengths in certain cultures

- In India, more than half the students in IT classes and receiving IT degrees are female
- In the US, women receive a much smaller percentage of the IT degrees
- According to Unlocking the Clubhouse:
  - “...a supportive learning community is critically important for the success of minority students in math and science. Seeking answers to the high failure rate of African American students studying calculus at the University of California at Berkeley, Treisman observed that Asian American students formed social communities in which they helped each other... ” (p. 105, Clubhouse)

# (Unlocking the Clubhouse)

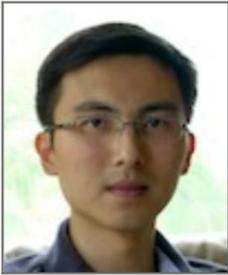
(last slide of direct quotes)

- He also found that most African American math students he studied were highly motivated, worked hard, and studied long hours but that even the best-prepared among them were failing.
- What stood out between the Asian and African American students was not a difference in motivation, preparation, or family support **but in integrating studying and learning into social lives.**
- African Americans were academically isolated and did not congregate into learning social communities the way the Asian students did.
- Instead, their academic interests and social interests were separate while they worked hard (**and somewhat unproductively**) on their own.
- (quoted directly from Unlocking the Clubhouse, p. 105-106)

How can we create supportive  
learning environments?

# Silent Technical Privilege

Philip Guo



Assistant Professor of  
Computer Science  
University of Rochester

<http://pgbovine.net/tech-privilege.htm>

As an Asian male student at MIT, I fit society's image of a young programmer. Thus, throughout college, nobody ever said to me:

- “Well, you only got into MIT because you're an Asian boy.”
- (while struggling with a problem set) “Well, not everyone is cut out for Computer Science; have you considered majoring in bio?”
- (after being assigned to a class project team) “How about you just design the graphics while we handle the backend? It'll be easier for everyone that way.”
- “Are you sure you know how to do this?”

## PROGRAMMERS AREN'T SUPERHEROES

---

One trite retort is, “Well your friend should've been tougher and not given up so easily. If she wanted it badly enough, she should've tried again, even knowing that she might face resistance.”

These sorts of remarks also aggravate me. Writing code for a living isn't like being in the Navy SEALs hunting down international terrorists or [simultaneously shooting three pirates in the head at sea while they were pointing a gun at a civilian](#). Programming is seriously *not* that demanding, so you shouldn't need to be a tough-as-nails superhero to enter this profession.

## CONCLUSION

---

Here's a thought experiment: For every white or Asian male expert programmer you know, imagine a parallel universe where they were of another ethnicity and/or gender but had the exact same initial interest and aptitude levels. Would they still have been willing to devote the over ten thousand hours of deliberate practice to achieve mastery in the face of dozens or hundreds of instances of implicit discouragement they will inevitably encounter over the years? Sure, some super-resilient outliers would, but many wouldn't. Many of us would quit, even though we had the potential and interest to thrive in this field.

# Hacker School

[www.hackerschool.com](http://www.hackerschool.com)

- Creates supportive learning environment by:
  - Social rules
    - **No feigning surprise**
    - **No well-actually's**
    - **No back-seat driving**
    - **No subtle sexism**
  - Providing a vision of ideal learner
    - *Be rigorous*
    - *Strive for greatness*
    - *Reflect on your progress*
  - Provides suggestions for learning

# (Hacker School)

## Social rules

- Another way we try to remove obstacles to learning is by having a small set of *social rules*. These rules are intended to be lightweight, and to make more *explicit* certain social norms that are normally *implicit*. Most of our social rules really boil down to "don't be a jerk" or "don't be annoying." Of course, almost nobody sets out to be a jerk or annoying, so telling people not to be jerks isn't a very productive strategy. That's why our social rules are designed to curtail specific behavior we've found to be destructive to a supportive, productive, and fun learning environment.

# (Hacker School)

## **No feigning surprise**

- The first rule means you shouldn't act surprised when people say they don't know something. This applies to both technical things ("What?! I can't believe you don't know what *the stack* is!") and non-technical things ("You don't know who RMS is?!"). Feigning surprise has absolutely no social or educational benefit: When people feign surprise, it's usually to make them feel better about themselves and others feel worse. And even when that's not the intention, it's almost always the effect. As you've probably already guessed, this rule is tightly coupled to our belief in the importance of people feeling comfortable saying "I don't know" and "I don't understand."

# (Hacker School)

## No well-actually's

- A well-actually happens when someone says something that's almost - but not entirely - correct, and you say, "well, actually..." and then give a minor correction. This is especially annoying when the correction has no bearing on the actual conversation. This doesn't mean Hacker School isn't about truth-seeking or that we don't care about being precise. Almost all well-actually's in our experience are about grandstanding, not truth-seeking. (Thanks to [Miguel de Icaza](#) for originally coining the term "well-actually.")

# (Hacker School)

## **No back-seat driving**

- If you overhear people working through a problem, you shouldn't intermittently lob advice across the room. This can lead to the "too many cooks" problem, but more important, it can be rude and disruptive to half-participate in a conversation. This isn't to say you shouldn't help, offer advice, or join conversations. On the contrary, we encourage all those things. Rather, it just means that when you want to help out or work with others, you should fully engage and not just butt in sporadically.

# (Hacker School)

## **No subtle sexism (1)**

- Our last social rule bans subtle sexism, racism, homophobia, etc. This one is different from the rest, because it's often not a specific, observable phenomenon ("well-actually's" are easy to spot because they almost always start with the words, "well, actually...").
- Hacker School is not a place to publicly *debate* whether comment X is sexist, racist, etc. If you see something that's unintentionally sexist, racist, homophobic, etc. at Hacker School you're welcome to point it out to the person who made the comment, either publicly or privately, or you can ask one of the faculty to say something to that person. Once the initial mention has been made, we ask that all further discussion move off of public channels. If you are a third party, and you don't see what could be biased about the comment that was made, feel free to talk to faculty. Please don't say, "Comment X wasn't homophobic!" Similarly, please don't pile on to someone who made a mistake.

# (Hacker School)

## **No subtle sexism (2)**

- We want Hacker School to be a space with as little bigotry as possible in it. Therefore, if you see sexism, racism, etc. outside of Hacker School, please don't bring it in. So, for example, please don't start a discussion on the mailing list of the latest offensive comment from Random Tech Person Y.
- Why don't we want public discussions of sexism, racism, etc. at Hacker School? For many people, especially those who may have spent time in unpleasant environments, these conversations can be very distracting. At Hacker School, we want to remove as many distractions as possible so everyone can focus on programming. There are many places in the world to discuss and debate these issues, but there are precious few where people can avoid them. We want Hacker School to be one of those places.

# (Hacker School)

## Why have social rules?

- The goal isn't to burden Hacker School with a bunch of annoying rules, or to give us a stick to bludgeon people with for "being bad." Rather, these rules are designed to help all of us build a pleasant, productive, and fearless community.
- If someone says, "hey, you just feigned surprise," or "that's subtly sexist," don't worry. Just apologize, reflect for a second, and move on. It doesn't mean you're a "bad" person, or even a "bad" Hacker Schooler. As we said above, these rules are meant to be *lightweight*. We've all done these things before. In fact, we originally adopted a no well-actually policy for our company because Nick and Dave well-actually'd each other *all the time*.

# Traits of an awesome Hacker Schooler

Here are three principles we believe in. If you're living up to these principles, you're doing well at Hacker School.

- *Be rigorous.* Understand how and why your code works. Understand your tools. If you're working with a framework (like Sinatra or Flask), learning to use it is just scratching the surface. Go deeper. Learn how it works.
- *Strive for greatness.* You're all at Hacker School because we believe you can be great programmers. Becoming great takes a lot of work. All of us who work at Hacker School are trying to become great, too. We don't think we're there yet.
- *Reflect on your progress.* We're all getting better at programming, but we should also be getting better at learning. Reflecting looks different for different people, but we recommend two primary things. First, write a blog! Even if no one else reads it, writing prose is a great way to crystallize concepts in your mind and deepen your understanding. Second, get code review! It's so much easier to get better when you're getting feedback and advice.

# If you're relatively new to programming

- *Choose a language and stick to it.* Python or Ruby is probably best; JavaScript and many others are also ok.
- *Avoid large frameworks* like Rails and Django until you've got a decent grasp on the language you're using.
- *Write lots of code.* The specific code you write is less important than that you write lots of code.
- *Don't worry about choosing the "perfect" project.* It's easy to let the perfect be the enemy of the good when it comes to project selection.
- *Have your code reviewed regularly*, ideally by someone who knows the language you're working in well.
- [\*Pair program\*](#), ideally with people who know the language you're working in well.
- *Develop a good mental model of your code.*
- *Become a systematic debugger.*
- *Write small programs from scratch.*
- *Give yourself progressively larger challenges.* For example, write a project you think will take an hour, then an afternoon, then a day, then two days...
- *Become comfortable with your tools*, but don't go overboard yak-shaving. Learn to use git, GitHub, an editor (vim, emacs, Sublime, TextMate, all are great), and your language's debugger.
- *Avoid distractions.*

# If you already have a good grasp of at least one language

- *Learn a second (or third) language well*, ideally one that's very different from the one you know best (e.g., if you know Python, it's probably more educational to learn Go or C or Scheme before you learn Ruby).
- *Have your code reviewed regularly.*
- *Review other people's code regularly.*
- *[Pair](#) with people of all skill levels.* You can learn a lot pairing with people at or above your level, and pairing with people who aren't as experienced (having to explain things to someone else forces you to deepen and clarify your understanding of them).
- *Find the things you've always been a little scared of* – e.g., multithreaded programming – and learn those (assuming they actually interest you).
- *Avoid distractions.*

# If you're really experienced

- *Contribute to open source projects*, especially well-established ones with high code-quality standards.
- *Consider starting your own open source project*. Two good ways to find ideas: Scratch your own itch, or factor out code you've written multiple times previously into its own library.

# (last slide directly quoting HackerSchool.com)

- **Social rules**
- Another way we try to remove obstacles to learning is by having a small set of *social rules*. These rules are intended to be lightweight, and to make more *explicit* certain social norms that are normally *implicit*. Most of our social rules really boil down to "don't be a jerk" or "don't be annoying." Of course, almost nobody sets out to be a jerk or annoying, so telling people not to be jerks isn't a very productive strategy. That's why our social rules are designed to curtail specific behavior we've found to be destructive to a supportive, productive, and fun learning environment.

# Recruiting Girls and Women to IT

**iridescent**



**tech** *novation*

[About](#) | [Curriculum](#) | [Forum](#) | [Register](#) | [Get Involved](#) | [Results](#) | [Resources](#)

**Technovation is the global technology entrepreneurship program for girls.**

1,374 alumnae · 19 countries

[Register Now for 2014 »](#)

<http://iridescentlearning.org/programs/technovation-challenge/>

# Technovation Challenge 2014

- Groups of 5 girls, aged 13-18
- One mentor (woman in IT field)
- One teacher or coach
- 12 weeks, February to May
- Build a mobile app from scratch (using AppInventor)
- Create business plan and 4-minute video pitch
- Winners get \$10,000 and have app built

# Retaining Women in IT

# What about adult women currently in IT careers?

- Do we have a retention problem?
- Do many women change careers, OUT of IT?
- Is the trend of “technologists” becoming “project managers” a symptom of a relatively unwelcoming, male-dominated field?
- Does the ‘persistence roller coaster’ continue?
- Does the fast pace of technological change impact women differently than men?

Can we create supportive learning environments?

# My Professional Goals:

- To keep my technical skills current enough that I have job security and flexibility
- To have easy access to my ideal learning environment

# My Ideal Learning Environment:

- **Supportive** (women, face-to-face, small teams, pair-programming, mentors)
- **Applied** (hands-on, real problem, 'hacking/coding sessions')
- **Local** (less expensive, easier to get to)
- **Shared leadership** (ideas and initiative comes from multiple individuals)
- **Modest time commitment** (4-6 hours per month?)
- **Minimal administrative overhead** (branch of codeChix?)

# CodeChix.org

- CodeChix is a CA non-profit public benefit organization for local women developers run by local women developers in a non-alpha, language/os agnostic, **supportive environment with an emphasis on face-to-face communication**. It is the first group of its kind and started in 2009 in San Jose, CA.

# CodeChix Events (monthly)

- **Hacking Sessions:** Usually held at coffee shops, we get together for a couple of hours to code. You're welcome to bring your own project, help out someone else, or just relax.
- **Workshops:** Spend between 2 to 5 hours with us on a weekend to learn about a state-of-the-art tool or programming language to improve your skills. Workshops are taught by volunteer industry professionals and CodeChix members.
- **Tech-Talks:** We invite coders and technical leads to talk to us about their cool projects and share their knowledge with us. These are usually on a weekday evening and last approximately 2.5 hours
- **Speaker Series:** Once a year, we invite speakers from executive levels or principal technical contributors to come and share their personal projects or product, show and go through their code, demo the project to us and do Q&A.

# CodeChix Skills Levels Defined

## **Newbies** (never coded, been away from coding for > 2 years)

Welcome to the group! If you are new to programming or have had a “gap” since you last programmed, it is highly recommended that you attend at least 2 of the monthly sessions. Bring your goals and ideas and talk to everyone to share what you are expecting to do and what timeframe and level you are looking to achieve. Also, if you have any specific languages, OS's that you are interested in. Note that all our organizers are volunteers and please be appreciative of their time, efforts and mentoring abilities by either buying them a cup of coffee/dessert/whatever-they-like and supporting them through showing up for their events and contributing whatever you can. If you are unclear how you can contribute, ask the organizers and offer what your skill set is. We need all kinds of skills in this group particularly help with marketing/sales and lining up speakers for our speaker series events. If you have your own websites, please link to our site to show support.

## **Intermediate** (coding for 3-6 years)

Welcome to the group! As intermediate we are very happy to have you join us. Since you are new to the group, please attend at least one monthly session and share what level you are at and what level you are expecting to get to and timeframe. As an intermediate coder, please consider being an organizer for one of our sessions – informal, hacking, marketing and/or speaker series. If you have your own websites, please link to our site to show support.

## **Advanced** (coding for 7+ years)

Welcome to CodeChix! We are honored to have you in our group and are eager to help you in whatever goal you wish to achieve. We would love to see you as often as possible at whichever session you choose. Please consider being an organizer for our hacking sessions. We, also, welcome you to be part of our speaker series and would love to have you showcase your projects/talk about them. So, please don't hesitate to contact the appropriate organizers for each session type. And, we would love to have you be a herald and public proponent of CodeChix and speak/blog/tweet on our behalf on all things code-related, if you are interested. It is a great way to get visibility and reach. If you have your own websites, please link to our site to show support.

Next steps?