



UC Berkeley
Computer Science
Lecturer SOE
Dan Garcia

CS10 : YOU'LL LOVE IT!
**Watch the student
testimonials about CS10,
what it means to them,
and how it has changed
their lives. Inspiring!**

inst.eecs.berkeley.edu/~cs10/

CS10 The Beauty and Joy of Computing

Lecture #1 Welcome; Abstraction

2012-01-18



CS10 in one slide

▪ Big Ideas of Programming

- Abstraction
- Algorithms (2)
- Recursion (2)
- Functions-as-data, λ (2)
- *Programming Paradigms*
- *Concurrency*
- *Distributed Computing*

▪ Beauty and Joy

- “CS Unplugged” activities
- All lab work in pairs
- Two 3-week projects in pairs
 - Of their own choice!!
- One blog
 - Of students’ own choice!!

▪ Big Ideas of Computing

- HowStuffWorks
 - 3D Graphics
 - Video Games
 - Computational Game Theory
- Research Summaries
 - AI
 - HCI
- Apps that Changed the World
- Social Implications of Computing
- Saving the World with Computing
- How Twitter Works (guest lecture)
- Cloud Computing
- Limits of Computing
- Future of Computing



Format & Textbooks

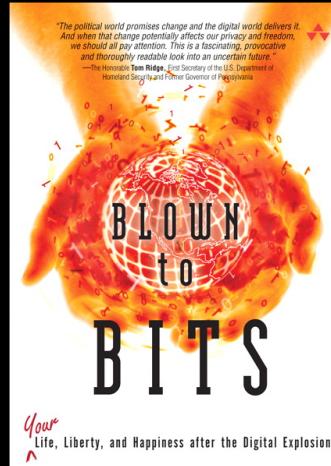
- Format (7 hrs/wk * 14 wks)

Mon	Wed	Fri
Lecture	Lecture	Discussion
Lab	Lab	
Lab	Lab	

- Selected Reading

- Taken from great book ("Blown to Bits" by Abelson, Ledeen & Lewis) + articles + videos
- Current events EVERY DAY (e.g., IBM's Watson vs Jeopardy)

- All resources FREE
 - Even clickers!



HAL ABELSON • KEN LEDEEN • HARRY LEWIS

contributed articles

Data generated as a side effect of game play also solves computational problems and trains AI algorithms.

BY LUKE VON AHN AND LAURA DABISH

Designing Games With A Purpose

MANY TASKS ARE trivial for humans but continue to challenge computers, such as common computer programs. Traditional computational approaches to solving such problems focus on improving artificial-intelligence algorithms. Here, we advocate a different approach: harnessing the power of the human brain through computer games. To aid this goal, we present general design principles for the development of games that we call "games with a purpose," or GWAs, in which people, as a side effect of playing, perform tasks computers are trained to do.

The Entertainment Software Association (www.theesa.com/facts_gamer.asp) has reported that more than 100 million Americans are spending their free time playing computer and video games in the U.S. Indeed, by age 21, the average American has spent more than 10,000 hours playing video games—equivalent to five years of work at a full-time job of 40 hours per week.

26 COMMUNICATIONS OF THE ACM / APRIL 2012 / VOL. 55 / NO. 4

IS ABSTRACTION THE KEY TO COMPUTING?

Why is it that some software engineers and computer scientists are able to produce clean, elegant designs and programs, while others cannot? Is it possible to improve these skills through education and training? Critical to these questions is the notion of abstraction.

By JEFF KRAMER

For over 30 years, I have been involved in teaching and research in computer science and software engineering. My primary experience has been in programming, distributed systems, distributed algorithms, concurrency, and software design. All these courses require that students are able to perform problem solving, conceptualization, modeling, and analysis of complex systems. In the last few years, students are clearly able to handle complexity and to produce elegant models and designs. The same students are also able to cope with the complexities of distributed algorithms, the applicability of various modeling notations, and other subtle issues.

77

contributed articles

"Digital fluency" should mean designing, creating, and sharing, not just browsing, chatting, and interacting.

BY MITCHELL BRONFEN, JOHN MALONEY, ANDRÉS MENEZES, HERNANDEZ Z., NATALIA RUMY, EVELYN EASTMAN, ALICE BLOOM, ANTHONY BURGESS, JAY SILVER, BRIAN SILVERMAN, AND YASMIN KAPAI

Scratch: Programming for All

When Seymour Papert, Editor-in-Chief of *Communications*, invited us to submit an article, he recalled how he first learned about Scratch. "A colleague of mine was a painter," he said. "One night he tried to get her 10-year-old daughter interested in programming, and the only thing that appealed to her was Scratch." That's when we realized that when we set out to develop Scratch six years ago, We wanted to develop an approach to programming that would appeal to people who had no prior experience as programmers. We wanted to make it easy for everyone, of all ages, backgrounds, and interests, to program their own digital stories, music, art, games, and simulations, and share their creations with one another. Since the public launch in May 2007, the Scratch Web site has become the most popular site for children without online community, with people sharing,

80 COMMUNICATIONS OF THE ACM / APRIL 2012 / VOL. 55 / NO. 4



Peer Instruction

- Increase real-time learning in lecture, test understanding of concepts vs. details
- As complete a “segment” ask multiple choice question
 - 1-2 minutes to decide yourself
 - 2 minutes in pairs/triples to reach consensus. Teach others!
 - 2 minute discussion of answers, questions, clarifications



Piazza for {ask,answer}ing questions

The screenshot shows the Piazza interface. At the top, there's a navigation bar with the Piazza logo, a search bar, and a button to "Add Question/Note". On the right, a user profile for "Dan Garcia" is shown. Below the navigation bar, a sidebar on the left displays "Popular tags" and sections for "QUESTION FEED" and "FILTERS", with "This week" and "Last week" expanded. A question titled "When are TA / professor office hours?" is highlighted in yellow. The main content area shows the question and its response. The question was posted by an administrator and last updated by Luke Segars 2 days ago. The response is from the instructor, stating they haven't established office hours yet but will update soon. It was also last updated by Luke Segars 2 days ago. Below the response, there are buttons for "Good Answer!" and "Ask a Followup". A "Start off a Students' Response" button is also visible. At the bottom, there are sections for "AVERAGE RESPONSE TIME" (N/A), "SPECIAL MENTIONS" (Luke Segars answered the question), and "USERS ONLINE THIS WEEK" (3). A footer at the very bottom includes links to "About Piazza", "Privacy Policy", "Copyright Policy", "Terms of Use", and "Report a Bug", along with a copyright notice for 2011.



Abstraction

- **Detail removal**
 - “The act or process of leaving out of consideration one or more properties of a complex object so as to attend to others.”
- **Generalization**
 - “The process of formulating general concepts by abstracting common properties of instances”

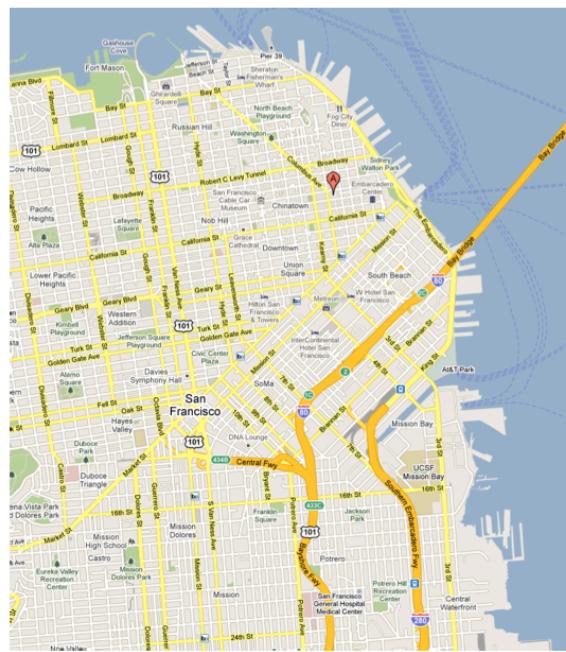


Henri Matisse “Naked Blue IV”

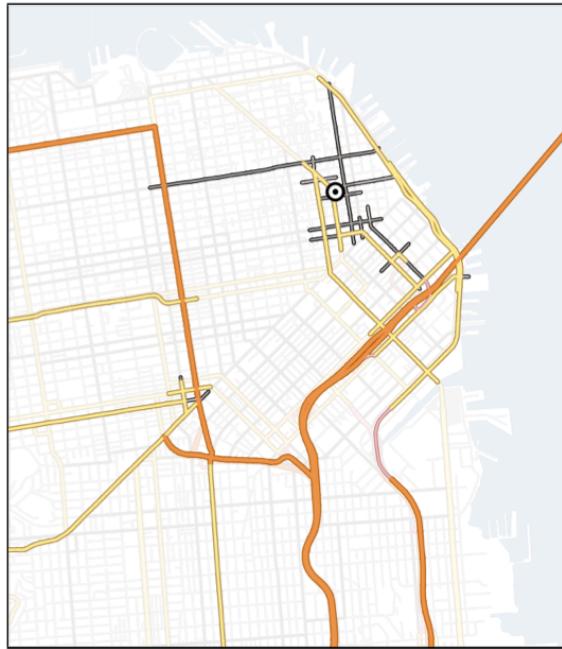
Garcia, Spring 2012



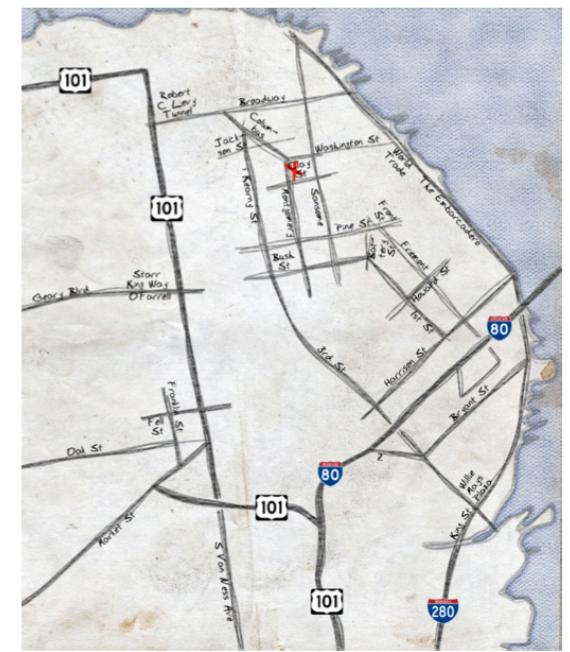
Detail Removal



General Purpose Online Map



Selected Roads



Our Result

Automatic Generation of Detail Maps
Maneesh Agrawala (UCB EECS), among others



Detail Removal (in CS10)

- You'll want to write a project to **simulate a real-world situation**, or play a game, or ...
- Abstraction is the idea that you **focus on the essence**, the cleanest way to map the messy real world to one you can build



The London Underground 1928 Map
& the 1933 map by Harry Beck.

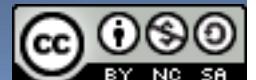


Generalization Example

- You have a farm with many animal kinds.
- Different food for each
- You have directions that say
 - To feed dog, put dog food in dog dish
 - To feed chicken, put chicken food in chicken dish
 - To feed rabbit, put rabbit food in rabbit dish
 - Etc...
- How could you do better?
 - To feed <animal>, put <animal> food in <animal> dish



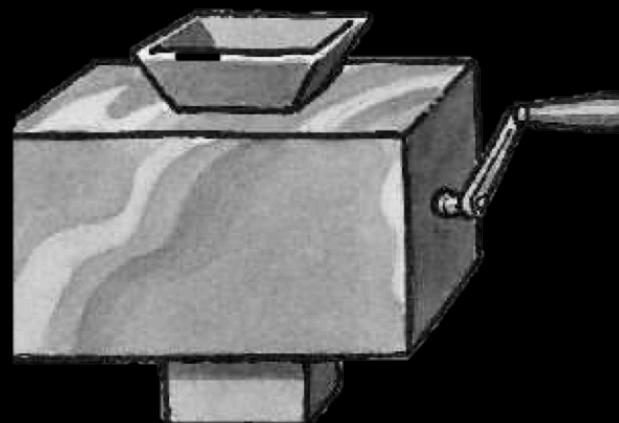
Garcia, Spring 2012



Generalization (in CS10)

- You are going to learn to write functions, like in math class:

$$y = \sin(x)$$



“Function machine” from *Simply Scheme* (Harvey)

- You should think about what inputs make sense to use so you don't have to duplicate code



Summary

- Abstraction is one of the big ideas of computing and computational thinking
- Think about driving. How many of you know how a car works? How many can drive a car?
Abstraction!



Someone who died in 1930 could still drive a car today because they've kept the same Abstraction!
(right pedal faster, left pedal slow)

