05-431 / 05-631
Software Structures for User Interfaces (SSUI)

Brad Myers
Human Computer Interaction Institute

Fall, 2022

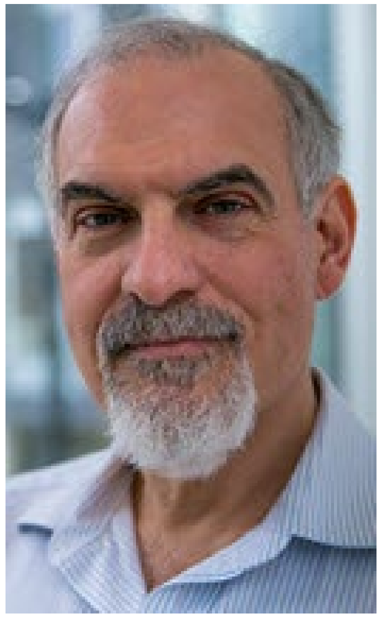
Course:

- Course web page:
  - Temporarily: http://www.uicourse.org/
  - http://www.cs.cmu.edu/~bam/uicourse/05631fall2022
- Schedule / Syllabus:
  - http://www.cs.cmu.edu/~bam/uicourse/05631fall2022/schedule.html
- Tuesday / Thursday, 3:05PM - 4:25PMin GHC 4102
- Lab sessions: Fridays, 3:35PM - 4:25PM in WEH 4623

NEW!

Instructor

- Brad Myers
  - Human Computer Interaction Institute
  - Office: Newell-Simon Hall (NSH) 3513
  - Office Phone: 412-268-5150
  - E-mail: bam@cs.cmu.edu
  - http://www.cs.cmu.edu/~bam
- TA: Ángel Alexander Cabrera
  - PhD student in HCI
  - https://cabreraalex.com/
  - Email: acabrera@andrew.cmu.edu



Office Hours

- Homeworks are due on Tuesdays and Thursdays before class
  - See: https://www.cs.cmu.edu/~bam/uicourse/05631fall2022/homeworks.html
- Tentative:
  - Alex's office hour: Fridays at 11:00am - 12:00noon in NSH A408
  - My office hour: Wednesdays at 11:00am - 12:00noon in NSH 3513
- Or post questions on Piazza – will usually answer right away

Acknowledgements

- Thanks to PhD students Michael Liu for consultations on content, homeworks and web page design
- Michael wrote most of the original reference implementations for the homeworks
- Corey Emery – TA in 2020
  - Helped develop most of the homeworks
- Clara Cook – TA in 2021
  - Also contributed

- Also discussions with Profs. Scott Hudson, Jen Mankoff, Christian Kaestner and many others

Lab sessions by Alex Cabrera

- Fridays, 3:35PM - 4:25PM in WEH 4623
- Will provide practice on what is required for the homeworks
  - Will be helpful for homeworks
- Content *may* be on the tests
  - So not optional

Recordings

- This course is *in-person*
- But all lectures and lab sessions will be on Zoom and recorded
  - Links will be on Canvas since *not public*
    - Do not share recordings publicly or on the open web
- However, not promising the recordings will be perfect
  - Issue of class questions, writing on the board, etc. not being recorded well
- For your own personal use if sick or for reviewing

Other HCI Courses

- We will ***not*** cover UI *design*
  - Homeworks specify the designs
  - You need to do software design and implementation to match the spec.
  - *NOT: "explore UI design from a more cs perspective" – preliminary questionnaire*
- We will ***not*** cover UX / HCI *Methods*
  - Not how to *evaluate* a user interface, needs analysis, or *iterative prototyping*
- Not redundant with any other HCI course
- **Other** courses that provide an **overview** of HCI or teach **methods**:
  - 05-391/891 Designing Human Centered Software (DHCS) – *best overview*
  - 05-410/610 User-Centered Research and Evaluation (UCRE) – *UX methods*
  - 05-430/630 Programming Usable Interfaces (PUI) – *iterative prototyping*
    - **SSUI is an approved alternative for PUI for MHCI students**
  - 05-863 Introduction to Human Computer Interaction for Technology Executives

05-830: Advanced UI Software (AUIS)

- I used to teach an “Advanced UI Software” (AUIS) class for PhD students (05-830)
  - Research oriented – lots of reading of research papers
  - How to *build* UI toolkits
  - Always had a tiny enrollment
- Will be a section of SSUI next year – Fall’2023
  - Recommend that all PhDs wait and take it next year

Course History

- 05-431/631 Software Structures for User Interfaces (SSUI)
  - also Software Architectures for UIs (SAUI)
  - One of the first HCII courses
  - I taught it in 2001
  - Focused on Java programming
- 430/630 PUI added later, with prototyping/iterative design
- Fewer CS students in MHCI/BHCI programs
- All students wanted PUI content
- “SSUI" then became a lab in PUI – now just called “Advanced Lab”
- Fall’2020 – started over since SSUI needed for new HCI Major
  - Purely virtual
- This is the third time taught
  - First time it has a lab session – to provide more support on the programming aspects

Readings

- Schedule & readings:
  - https://www.cs.cmu.edu/~bam/uicourse/05631fall2022/schedule.html
  - Will link to the slides for each lecture
  - Course schedule is tentative
  - Note required readings – only a few that aren't part of homeworks
  - Lots of "Recommended" and "Optional"
    - Recommended - Good background on that topic; what you *should* know
    - Optional – other interesting or historically important readings
  - CMU-only and ACM DL, use CMU network or VPN

Homeworks

- Homeworks
  - https://www.cs.cmu.edu/~bam/uicourse/05631fall2022/homeworks.html
- Take Home Midterm
  - Was popular in previous years, so doing it again
- Take Home Final test
  - Not cumulative, just on the second half lectures
- See homework policies
    - Due before class on the scheduled date, homeworks can be turned in late for a penalty, homeworks are individual, no cheating, turn in on Canvas & GitHub
- Schedule + Homeworks = Syllabus
- Questions using Piazza: piazza.com/cmu/fall2022/05431631
- Checkout and turn-in using Github classroom
  - Has everyone used Github?

Final Project

- Create your own
- Can be a reimplementation or novel & publishable
- Will be in groups
- Will have about 1 month
- Lots of topic ideas will be on the final projects page

What is this class about? "Software Structures for User Interfaces"

- "**User Interfaces**" (UI)
  - The part of an application or device that a person sees or interacts with
  - Everything the user can see, the look-and-feel, all behaviors, speed of interaction, etc.
- "**Software Structures**"
  - How the UI is **implemented** = "Front end programming"
  - Software patterns / architectures across many platforms
    - Not just what is popular today
  - Complexities unique to UIs

What Will This Class Cover?

- Key principles & structures for implementing UIs
- Most have stood the "test of time"
  - Some invented in the 1980s and still used today
- Key tradeoffs and design decisions for the implementations
- Will continue to be useful with the next library & language

But what specifically?

- (See the schedule and homework list)
- HTML, CSS, JavaScript, React
  - Their models, principles, hierarchies, intersections and differences
  - (But you need to learn the details on your own)
- Input handling
  - Click, double-click, drag, touch, multiple fingers, other sensors
- 2D output
  - Canvas vs. retained object model, details of 2D graphics, refresh
- Implementing Undo
- Connecting to a backend, using web services, cloud database
- Other Implementation details: resources, geometry management
- Constraints and data bindings
- Model-view-controller and other architectures
- Interactive UI builders & prototypers
- Implementing for accessibility
- Specialized UI tools: visualization, 3D, AR/VR, conversational UIs, UbiComp, animation

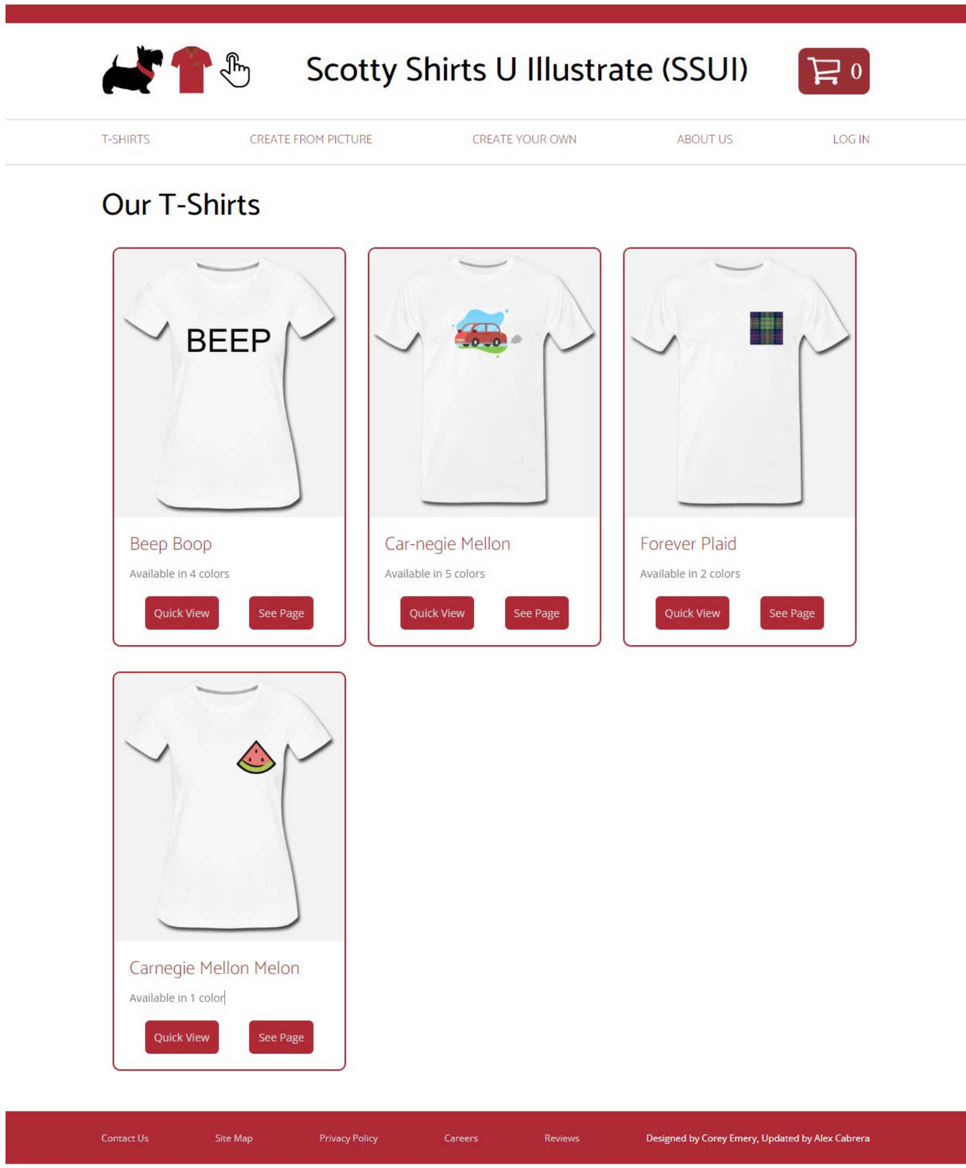
} homeworks

People’s Backgrounds

- Based on your survey answers from the Google Form
- Questions mostly matched what we will cover

Homework 1

- Assigned today, due 9/13/2022 at 3:05pm ET (two weeks from today)
  https://www.cs.cmu.edu/~bam/uicourse/05631fall2022/HW1/
- Build a dynamic website to sell T-shirts
  - Dynamic = some parts of pages created using your JavaScript code
- Detailed specification (no UI design needed)
- Only needs to run on Chrome on regular size screens
  - Not “responsive”



Why is This Topic Important?

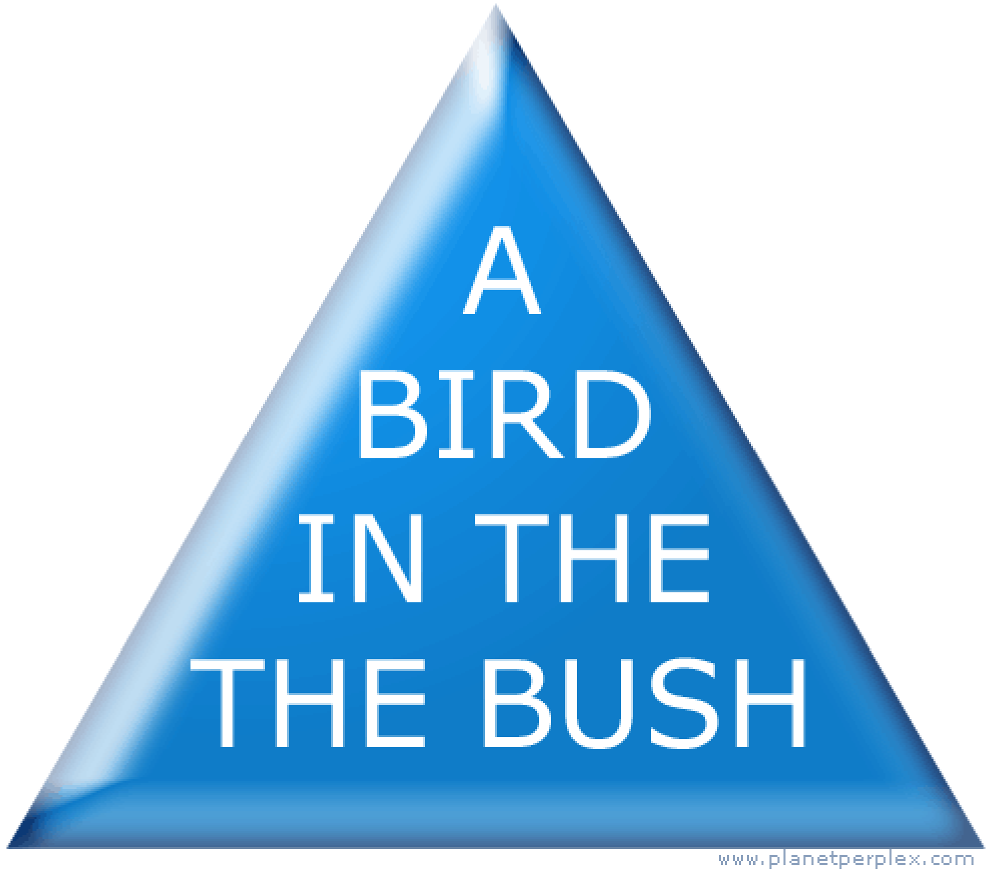
- Virtually all UIs are created using such Tools
- Previous research has influenced today's tools
  - Enormous impact!
- New tools are created all the time
  - E.g., Flutter for Dart language and mobile
  - Some are easier to use than others!
- Principles and architectures for good designs
  - Avoid "reinventing the wheel"
  - What are the "best practices" for these tools
- Modern UIs and Tools for web, phones, wearables, Smart TVs, etc. all use similar designs
  - Speech and conversational interfaces are different
- Research topic in ACM UIST, CHI
  - Also ICSE, SPLASH, PLATEAU, CHASE, many others

Why are User Interfaces Difficult to Design?

Why Hard to Design UIs?

- "It is easy to make things hard. It is hard to make things easy."
- No silver bullet
- Seems easy, common sense, but seldom done right
  - Once done right, however, seems "obvious"
- User Interface design is a creative process
- Designers have difficulty thinking like users
  - Often need to understand task domain
  - Can't "unlearn" something

Can’t Unlearn Something



Why Difficult, 2

- Specifications are always wrong:
  - "Only slightly more than 30% of the code developed in application software development ever gets used as intended by end-users. The reason for this statistic may be a result of developers not understanding what their users need."

    -- Hugh Beyer and Karen Holtzblatt, "Contextual Design: A Customer-Centric Approach to Systems Design,“ *ACM Interactions*, Sep+Oct, 1997, iv.5, p. 62.
  - Need for prototyping and iteration

Why Difficult, 3

- Tasks and domains are complex
  - Word 1 (100 commands) vs. Word 2013 (>2000)
  - MacDraw 1 vs. Illustrator
  - BMW iDrive adjusts over 700 functions
- Adding graphics can make worse
  - Pretty ≠ Easy to use
- Can’t necessarily just copy other designs
  - Legal issues
- All design/development involves tradeoffs
  - Add Features
  - Test/Fix Bugs
  - Test/Fix usability
  - Time-to-market

Why are User Interfaces Difficult to Implement?

What are the most difficult kinds of programs, in general?

- What properties make a task difficult to program?

What are the most difficult kinds of programs, in general?

- What properties make a task difficult to program?
- *GUI programming has most of them!*

Why Are User Interfaces Hard to Implement?

- They are hard to design, requiring iterative implementation
  - Not the waterfall model: specify, design, implement, test, deliver
- They are reactive and are programmed from the "inside-out"
  - Event based programming
  - More difficult to modularize

Why Hard to Implement? cont.

- They generally require multi-processing
  - To deal with user typing; aborts
  - Window refresh
  - Window system as a different process
  - Multiple input devices
- There are real-time requirements for handling input events
  - Output 60 times a second
  - Keep up with mouse tracking
  - Video, sound, multi-media

Why Hard to Implement? cont.

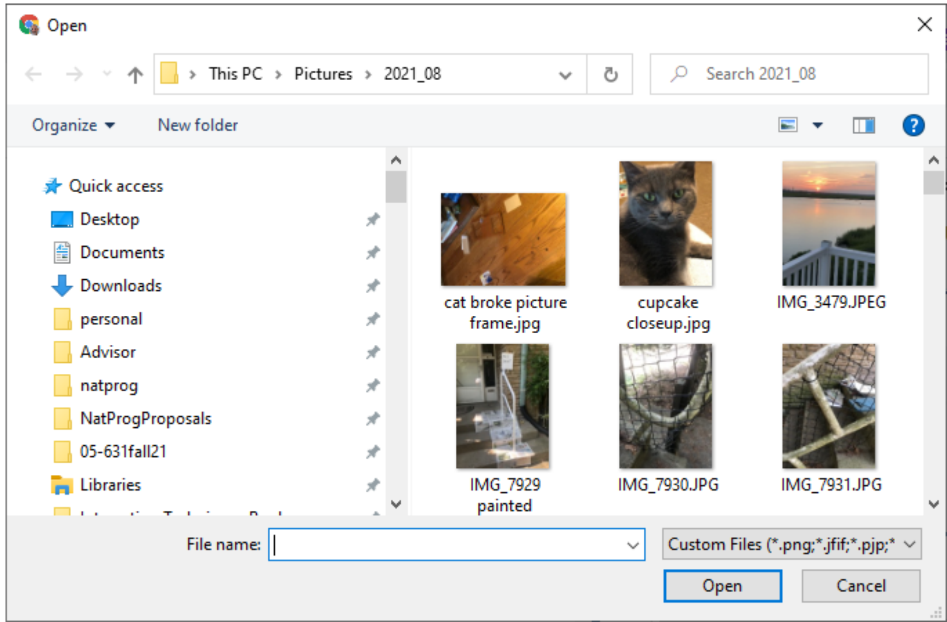
- Need for robustness
  - No crashing, on any input
  - Need for security, e.g., protect from SQL injection attacks
  - Helpful error messages and recover gracefully
  - Aborts
  - Undo
- Lower testability
  - No unit tests for user inputs
  - Few tools for regression testing
- Difficulty of Modularization
  - Much code goes into event handlers

Examples

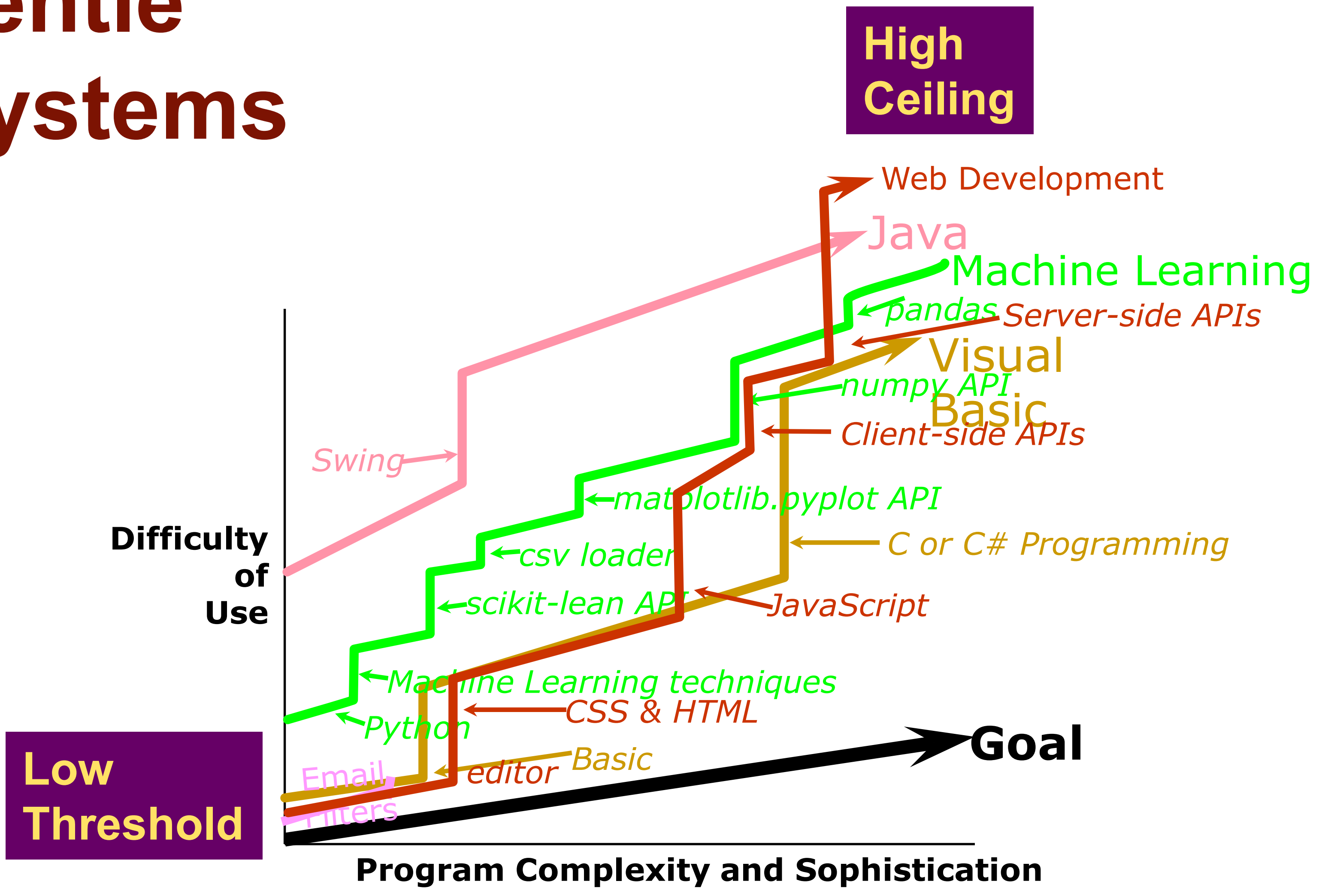
- Difference between displaying “hello” and displaying a blue rectangle
  - Easier with html/JavaScript!
- Difficulty to read a file name
  - Reading a text string from the **console** - `const input = prompt();`
  - Configuring and handling **built-in file dialog**

```
<input type="file"
   id="mypic" name="mypic"
   accept="image/png, image/jpeg">
```

  - **Creating** a new file dialog



Goal: Gentle Slope Systems


High Ceiling
Web Development
Java
Machine Learning
pandas
Server-side APIs
Visual Basic
numpy API
Client-side APIs
Swing
matplotlib.pyplot API
C or C# Programming
Difficulty of Use
csv loader
scikit-lean API
JavaScript
Machine Learning techniques
CSS & HTML
Python
Goal
Basic
editor
Email Filters
Low Threshold
Program Complexity and Sophistication