



# IN SHORT

The TL;DR version of my life. Everything you need to decide if this resumé belongs on your desk or not.

## SEEKING WORK INVOLVING

End-to-end user experience design, text analysis and raw data mining, teaching and team leading.

## DEGREES

### Master of Arts in Classics

- ♣ Princeton University
- ♣ Conferred 25 Sep 2010

- ♣ Final GPA 4.0
- ♣ All graduate examinations passed

### Bachelor of Arts with General Honors Classical Studies with Special Honors

- ♣ University of Chicago
- ♣ Conferred 10 Jun 2006
- ♣ Final GPA 3.99

- ♣ Phi Beta Kappa (Beta of Illinois), Spring 2005
- ♣ Student Marshal
- ♣ The Classics Prize

## SELECTED PRESENTATIONS

- ♣ “Unicode and you: or, how to do Greek text properly and why it matters,” at *The First Classics and Technology Colloquium*, Princeton University, April 16, 2011.
- ♣ “Software development for the iPhone,” guest lecture for Brian W. Kernighan, *COS 333: Advanced Programming Techniques*, Princeton University, March 11, 2010.
- ♣ “Lexidium: Software development and the classics,” at *iPhone Apps - The New High Tech Gold Rush?*, Princeton University, November 12, 2009.
- ♣ “Computer technology and textual criticism: new approaches for the 21st century,” at *The Future of the Ancient: Making Classics Relevant*, Ohio State University, May 2, 2009.

## PUBLISHED SOFTWARE

### Mac OS X

- ♣ *Andromeda*, Greek and Latin data-mining tool, published 2010, [goldibex.com/andromeda](http://goldibex.com/andromeda)

### iOS

- ♣ *Lexidium*, Apple App Store (iOS), published 2009, [goldibex.com/lexidium](http://goldibex.com/lexidium)
- ♣ *Lexiphanes*, Apple App Store (iOS), published 2009, [goldibex.com/lexiphanes](http://goldibex.com/lexiphanes)

Total iOS sales: **13,782** (as of Sept. 2011)

## LANGUAGES SPOKEN

- ♣ **Computer** fluent in C, Objective-C/Cocoa, Javascript, Perl, Python; very competent in C++; some PHP, Ruby underway
- ♣ **Human** fluent in Latin and ancient Greek; some French, German, Italian, and modern Greek (dual US and EU citizenship)
- ♣ **Design** CSS through version 3, HTML5, familiar with major templating languages



# CREATIONS

The coolest things I've built, both hardware and software, from the whimsical to the lifesaving.

## *ANDROMEDA*

In 2010 I planned to write my doctoral dissertation at Princeton on data mining the electronic corpus of Greek literature, a database called TLG. The TLG was designed in 1974. It uses an obsolete *text-streaming* format that divides documents into blocks of 8k and marks the difference between text and metadata segments by setting the sign bit. I wrote a C library that internally converts TLG text to Apple's native `NSAttributedString` and made it the basis of a data-mining program for the Mac called *Andromeda*.

## *LEXIDIUM AND LEXIPHANES*

In 2008 my then-fiancée attended a summer program in Rome run by the Pope's Latin secretary. The only book they used was a large Latin lexicon they were expected to have on hand at all times. Luckily a consortium had already converted the tome to XML. I produced a very rough app for Windows Mobile 6 that permitted basic lookups on dictionary entries, and Donna was spared a fifteen-pound burden. That Christmas I bought an iPhone, and in early 2009 I learned Objective-C and Cocoa and released my dictionary tools for the iPhone, *Lexidium* for Latin and *Lexiphanes* for Greek.

In late 2010 I added the ability to look up dictionary words (like *metuo*, to fear) from inflected forms (*metuant*, let them be afraid). This was a nontrivial programming task that required me to implement a finite-state transducer in C (ISO C99). I plan on releasing this software (called *Parsimonious*) as a library.

## EXPLODING COMPUTER AND *DICK CHENEY'S DUCK HUNT*

Every year, the University of Chicago hosts a Scavenger Hunt legendary for the size of the teams and the difficulty of the items. In 2004, item #28 on the list read, "have a computer combust through nothing but its own internal workings." Among other modifications, I removed the power supply's safety fuses and reverse-biased the large filter and smoothing capacitors on the DC side of the bridge rectifier. When powered on the computer produced a shower of sparks and burst into flame. It earned 32 points.

In 2006, item #177 was "a playable copy of Dick Cheney's Duck Hunt... for the NES." Other teams produced a Flash version of the game, but I hacked the original NES ROM in 6502 assembly with new graphics and jokes about the Vice President's BAC. As luck would have it, a teammate had a reprogrammable NES cartridge, and we presented our copy on a vintage NES. It earned (.22 x 100) points.

## GPS RECEIVER

In 2002, long before these devices were found in every phone and car in America, I wrote the control software in assembly language for a GPS receiver that my father had built. The receiver was designed around the 80c562, a compatible Philips extension of the common Intel 8051 microcontroller, and a small industrial GPS device that transmitted raw data via RS232. I found the project challenging because I had to hand-code low-level routines that I used to take for granted, such as *malloc* and *memcpy*.

It lacked a map display, having only a 4x20 character VFD on which it displayed location data, but the receiver saved my father's life when he took his small boat out on Lake Michigan and got swamped in the middle of a storm. He had memorized the coordinates of a safe harbor and made his way to safety in zero visibility, on instruments only. That was my first lesson in how important coding discipline could be.



# WHAT I CAN DO

The abilities I will bring to your team.

## RAPID SKILL DEVELOPMENT

I am already conversant in a wide variety of programming languages and idioms as listed above. Unless your shop runs on COBOL or Forth, odds are I already have enough know-how to get started now.

I also learn new development environments and languages quickly. I had Python by the horns in two weeks, and in a month and a half I was writing decorator factories and resolving questions about method resolution order in other peoples' APIs. Even Objective-C/Cocoa, which is renowned for its complexity and often confounds programmers with its differences in idiom and syntax from C++ , I was able to grasp relatively rapidly. Thanks to my existing skill in C I was able to master it in about three months.

Of course, actually *writing* code can sometimes play a smaller role in the development process, especially for mature apps. Fortunately, my background in both hardware and software has given me a strong facility in debugging, even where the only available tools are a disassembler and memory viewer. This can be critical when developing against proprietary or nonstandard libraries that have underdeveloped or even absent symbol files.

## TEAMWORK AND GROUP GUIDANCE

I believe that the greatest weakness in my portfolio is that I have never worked with a group of developers larger than two or three at a time. That said, I do have experience working with and directing a group toward a common objective from having taught at Princeton, where the students are voracious learners and expect continuous personal engagement with their teachers. In the fall semester of 2009, I had charge of three small seminar groups of 8 to 10 students each, in which I promoted discussion and interaction between students and provided timely and constructive feedback on assignments. Most critically, I was able to use all my information channels to identify and respond to what the students *were'n't* learning well.

My students' course evaluations speak for themselves. "Fantastic, calm, and easy to speak to. Great preceptor, who tried his best to engage the class, even when it was a monday [sic] morning." "He was one of the best preceptors I've had because of his intelligence, mastery of the material, and willingness to help his students." "He was very good and encouraged student participation without making anyone feel uncomfortable or pressured to speak. He was very responsive to questions and always available for help." (The full, authenticated transcript of evaluations is available on request.)

## ADAPTIVE BRANCH PREDICTION

I have a knack for picking winners in emerging technologies. Though I don't always see what's coming ahead of time, when I do throw my hat in the ring I am seldom wrong. In 2002, when most other college freshmen with cell phones had a Nokia or Motorola, I was using the Handspring VisorPhone, the prototype of the Treo smartphone, to read my email from anywhere I could get a signal. I correctly predicted the decline of Perl in large production environments in the mid-2000s, and I was on top of the explosive growth of the mobile app market in early 2009. Currently I believe that we should be preparing for the rise of native object database and persistence frameworks as serious alternatives to both RDBMS and structured storage.



# WHY I'M HERE

From the basement workshop to the ivory tower and on to Silicon Valley.

I often get asked, usually by people firmly on one side or the other, how I went from being a software designer to a scholar of ancient Greek and Latin and then back again. It's a long story, so I'll try to be brief.

By the time I was nine years old I already had the truth tables of Boolean logic memorized and I'd written two Apple II games. In seventh grade I built a mockup of a "smart card" application using Visual Basic and some creative image editing.

All this is totally understandable if you know something about my father, who was an old-school electronics engineer. If it was worth doing in his household, it was worth doing in wire wrap. His day job was hand-winding the superconducting coils for MRI machines, and when he got home he rebuilt vintage vacuum-tube radios for fun (tubes were "valves" in his worn-down British accent) and AM transmitters to spook his children with messages from Skeletor. The man lived and breathed electronics.

When I got older, we formed a two-man team. He did the hardware and I did the software. I had barely any formal training in development or programming, but between my dad's coaching and a knack for software development I never really hit any barriers.

My path to Greek and Latin was somewhat more labyrinthine. I found out pretty quickly that I loved to read, but I had no exposure to the ancient classics until college, where I picked up Greek and Latin and found a mentor in a faculty member looking to push the classics into the 21<sup>st</sup> century. Between my enthusiasm for the material and an extremely positive experience with the intellectual life in college, I decided to continue my study in classics at Princeton.

After a couple of years spent stuffing my head with Homer and Aeschylus and writing monographs on obscure archaic Latin historians, the old itch for software development resurfaced. Fed up with the lack of software support for people in my trade, I developed a variety of tools to make my colleagues' work faster and easier, but I ran into trouble when I started proposing computationally-oriented projects for my doctoral dissertation. Whether it was the use of machine learning to resolve outstanding questions of textual authorship or data-mining the poetic corpus to find evidence of orality in lyric poetry, my faculty kept shrugging me off, and the mantra "wait until you have tenure" rapidly became a fuse-blower for me.

The critical moment (that's *kairos* in ancient Greek) came when I helped organize and presented at a technology workshop for my department. The turnout was underwhelming and the people who would have been directing my dissertation were unreceptive to say the least.

So, after four years of graduate work and an M.A. to show for it, I'm ready to move on. I don't for a moment regret my time in Princeton, which was some of the happiest of my life. Having learned how huge the gulf was between what I want to do with my life and what I was able to do in academics, I'm now pursuing outlets for my creative impulse that welcome disruptive technology.