CS6452: Prototyping Interactive Systems

Class Syllabus and Tentative Schedule

CS6452 provides an introduction to design, prototyping, and implementation of interactive systems for human-centered computing. The course focuses on core concepts in computer science and the implications of those concepts for interactive systems.

Week	Lecture Topic	Assignment
1 1/10/05	Introduction • Course goals, structure and context • Prototyping defined • Kinds of prototyping (lo-fi, hi-fi) • Practicum: working in the development environment	• Develop a set of lo-fi prototypes to serve as the basis for the UI to be created in Module I
2 1/17/05	**NO CLASS!** MLK Holiday	
3 1/24/05	**Module I**: Asynchronous Programming • Modularity and decomposition of problems • Thinking asynchronously ○ Common idioms • Refresher on O-O programming • Refresher on Jython collection types • Refresher on Swing GUI programmingß	• Create functional, interactive GUI approximating behavior of lo-fi prototype • Demonstrate decomposition of functionality into callbacks

4
 	
1/31/05
	Module I: Asynchronous Programming (cont'd)
	• Under the hood: polling, blocking, and threads
	• Common patterns
	  ○ Table-based dispatch
	  ○ Command objects
	  ○ Threading idioms (work pools, transients)	

5
 	
2/7/05
	Module I: Asynchronous Programming (cont'd)
	• Examples of non-GUI event-based programming
	  ○ Timers (and interrupts)
	  ○ Notification services
	  ○ Collaborative applications
	• In-class presentations	

6
 	
2/14/05
	Module II: Distributed Applications
	• "Paper Prototyping" for network applications
	• Everything (almost) you ever wanted to know about networks
	  ○ Internet Addressing
	  ○ Sockets
	  ○ Protocol layering
	  ○ Marshaling
	• Common patterns
	  ○ Client-server
	  ○ Peer-to-peer
		• Create a protocol diagram for an instant messaging application
		• Integrate GUI front-end with network backend
		• Demonstrate proficiency witih common networking APIs (sockets)

7	**Module II**: Distributed Applications (cont'd) • The seven fallacies of distributed computing	

7
 	
2/21/05
	Module II: Distributed Applications (cont'd)
	• The seven fallacies of distributed computing
	• Dealing with errors
	  ○ Local exceptions
	  ○ Exceptions passed over the wire
	  ○ Exceptions caused by the network
	• Exception handling in Jython	

8
 	
2/28/05
	Module II: Distributed Applications (cont'd)
	• Other styles of distributed applications
	  ○ Web applications and HTTP
	  ○ Tuplespaces
	  ○ Programming as if bandwidth mattered
	• In-class presentations	

9
 	
3/7/05
	Module III: Data Management
	• Logging
	• Storing data in a database
	  ○ Data storage
	  ○ Data structure
	  ○ Data management
		• Extend IM system to implement data logging to a database
		• Demonstrate querying to access chat history, usage data

10
 	
3/14/05
	Module III: Data Management
	• Getting data out of a database
	  ○ Querying
	  ○ Query syntax	

11 3/21/05	**NO CLASS!** Georgia Tech Spring Break	
12 3/28/05	**Module III**: Data Management	
13 4/4/05	**Module IV**: Intelligence • Recommender and learning systems	• TBD
14 4/11/05	**Module IV**: Intelligence • Using a web service to do your work	
15 4/18/05	**Module IV**: Intelligence	

16
 	
4/25/05
	LAST CLASS!
	• Alternative Architectures
	• Research Topics in Prototyping
	  ○ Supporting informality: Denim
	  ○ Prototyping speed interfaces: Suede
	  ○ Prototyping web interfaces: Designer's Outpost
	  ○ Hardware prototyping for non-hardware types: Phidgets, Making Things
	• Practical Tools for Prototyping
		• Completed project due!

The Class

One of the core goals of this class is to give students experience in rapidly constructing interactive systems. Thus, the *prototyping* in the course title is largely focused around providing HCC students with enough understanding of computer science to be able to rapidly create prototype interactive systems (rather than, say, non-interactive paper prototypes). After completing this class, students should have the skills to begin more complex programming projects (either for other courses, or in service of the computational portfolio requirement), be able to understand and communicate the software architectural design choices and tradeoffs made in the construction of a system, and be able to communicate with computer scientists.

The bulk of the class is divided into a number of *module*s, each of which explores a particular topic or approach to building interactive systems. These modules cover concepts that will be useful (necessary?) for completing the class project.

The Project

The class project is the creation of a prototype instant messaging/chat system. The creation of this piece of software will demonstrate the concepts covered in the individual class modules—we will discuss the techniques needed the build each part of the system in lectures, and implement each part in lockstep.

Prior to starting the first module, students will paper prototype the user interface to the system. In the first module, each student will develop an interactive GUI that approximates the functionality described in that student's paper prototype. In the second module we will extend the GUI mockup into an interactive system, by adding network functionality. The contents of the third and fourth modules are currently TBD.

In-Class Presentations

While each student should largely work individually on the project, I do encourage collaborative discussion of approaches, and how best to take advantage of the architectures presented in class. Toward this end, we will formalize these discussions in a series of in-class presentations by students. Each presentation will be a short (5-10 minute, depending on class size) summary of how the student approached a design problem related to the class project, the tradeoffs that were made, and so on.

One goal with the in-class discussions is to give students practice in *explaining and presenting* an architectural design space. The ability to defend a set of design choices is essential for HCC students' computational portfolio requirement

Course Grading Criteria

- Functionality of class project 35%
- Demonstrated knowledge of concepts taught in class 25%
- In-class presentations 10%
- Class attendance and participation 10%

Textbooks

There is no required textbook for the class. However, depending on your level of Jython/Python knowledge, you may want to look over the available books and invest in a reference. Note that, for most of the purposes of this class, a Python book will serve you just as well as a Jython-specific book.

A few books you might want to look at include:

Jython Essentials, Samuele Pedroni and Noel Rappin
This is one of the few Jython-specific books. It focuses mostly on how Jython differs from "normal" Python, and how Java integrates with Jython. It's probably best for those who already know either Python or Java; it's less good as an introductory tutorial.

Python Cookbook, Alex Martelli and David Ascher
This book contains a whole ton of short Python examples (how to use databases from Python, how to use the network from Python, etc.) Because it's Python-specific (as opposed to Jython-specific), it doesn't cover Swing GUI programming though. Still, a very useful set of examples.

Python Standard Library, Frederick Lundh
This is basically a reference to all of the libraries that come with Python. Almost all of these work just the same in Jython. Fewer examples than the *Python Cookbook*, but still useful. My guess is that, for this class, you wouldn't make use of most of the libraries defined here. If you go on to do more Jython/Python programming, though, this is an excellent reference.

Python Pocket Reference, Mark Lutz
This is a small (pocket-sized) and cheap ($10) guide to Python syntax and operators. It's hard to beat for the money.

There are quite a few Python books on the market, and more are coming out all the time. I have by no means had the time to evaluate them all. If you find one that you think is particularly good, please let me know!

General Administrivia

Instructor: Keith Edwards, keith@cc.gatech.edu, TSRB 348
Course URL: http://www.cc.gatech.edu/classes/AY2005/cs6452_spring

A note on email: my spam filtering is liable to reject mail from free email services such as HotMail or Yahoo. If you have one of these, I'd advise you use your GT or CoC account instead to contact me.

A note on telephoning: Georgia Tech's phone and voicemail service sucks. Thus, the phone is not a great way to get in touch with me. Leaving voicemail is even worse, since I never check it. Please send email, or just drop by, instead of leaving a message.