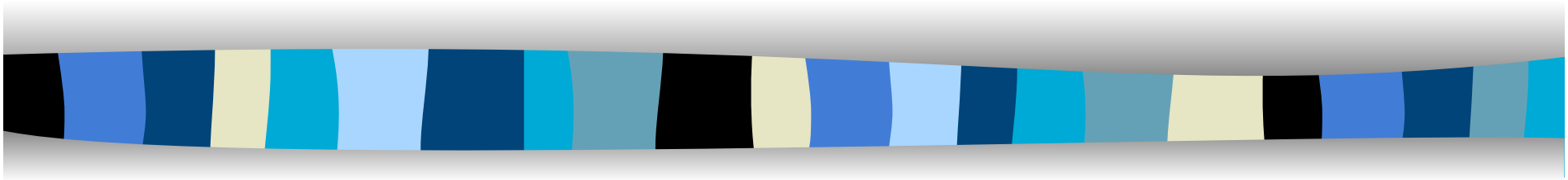
INFSCI 2470
Interactive System Design



Lecture 1: Introduction

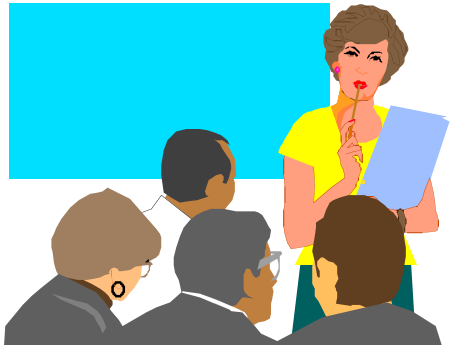
Peter Brusilovsky

INFSCI 2470 and MSIS

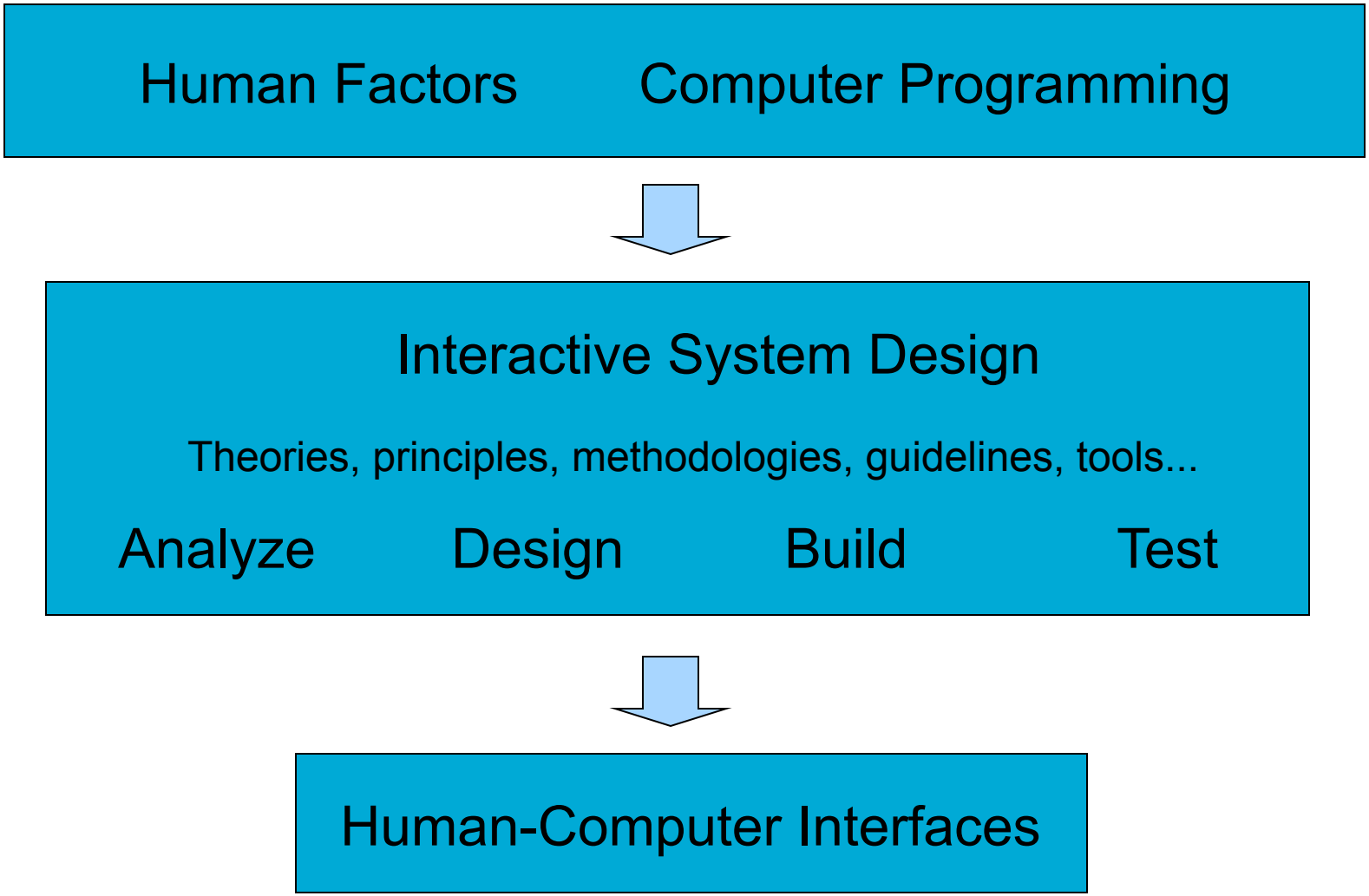
Humans

(Information) Systems

Interface



Body of Knowledge



What do we need to know?

- Foundation knowledge
  - Human factors, Systems, Programming
- Conceptual knowledge about HCI
  - Interfaces, paradigms, examples
- Conceptual knowledge on HCI design
  - Theories, principles, methodologies, guidelines, tools...
- Practical skills on HCI design
  - How to design, how to build, how to test

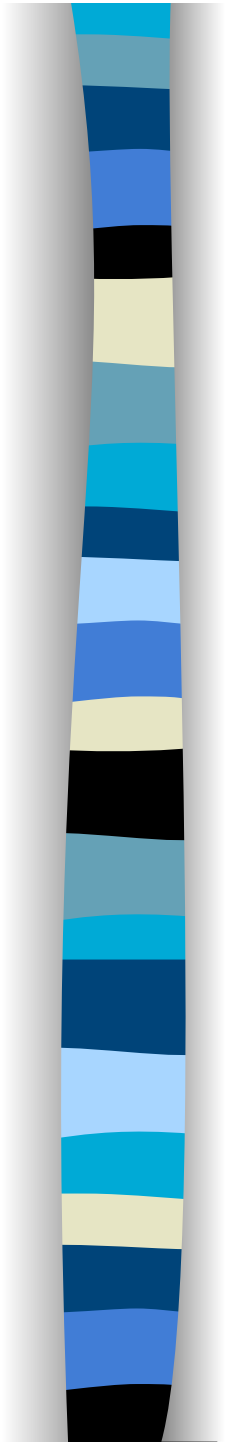
What we are going to learn?

- We will not learn what you are supposed to know: Foundation knowledge
  - Human factors, Systems, Programming

    BUT
  - We will review the human factors in the context of ISD
  - Get an introduction to GUI/Servlet part of Java programming
- We will not be able to learn about **all** kinds of past- and next-generation interfaces, but we will examine some categories (Web, VR, AR..)

Why we need to learn it?

- For any information scientist
  - Key part of the whole picture
- For software developers/testers
  - Role of the HCI in systems
- For HCI/Usability engineers
  - Key skills required for the job
- For Web professionals
  - Important for design/development/evaluation
  - Part of Web and Database Systems Track



How we will learn it?

- Lectures
  - Class lectures
  - Resarch seminars
- Readings
  - Course books
  - Additional sources
- Assignments and projects
  - Design, program, evaluate interfaces

Books

- William Newman and Michael Lamming
  - Interactive System Design
  - *How to get the book*
- Clayton Lewis and John Rieman
  - Task-Centered User Interface Design
- Additional Books (Design and Testing)
  - Norman
  - Nielsen
- Reading books with Knowledge Sea II

Course Work

1. Individual Homework 1: CourseWeb (5pt)
2. Individual Homework 2: Java Graphics (10pt)
3. Pair Homework 3: Servlets and forms (15pt)
4. Pair Homework 4: GUI programming (15pt)
5. Group Interface Evaluation Project (25pt)
6. Group Final Project
    1. Design part (20pt)
    2. Final Project (50pt)

A Few Words on Grading

- Final grade: percentage of total points
  - < 50% corresponds to F, 50-62.5 is D range, 65.5-75 is C range, 75-87.5 is B range, and 87.5-100 is A range.
- To get full credit submit homework before or on the due date
- Attendance and Activity count
  - Be active in forums, answer questions
- Integrity
- Group contribution

Course Tools

- All information will be provided via course Web site `http://www.sis.pitt.edu/~peterb/2470-121/`
- The complete list of tools is provided on Tools section of this site
- *CourseWeb* and *Knowledge Tree* will be used as the main learning support tools
- *Java and Java IDE* will be used for developing interactive systems
- Other tools will be introduced later

Blackboard (CourseWeb)

- CourseWeb system will be used for:
  - Posting announcements (WATCH IT!)
  - Posting course materials, assignments, and quizzes
  - Learning about and communicating with each other (including group communication)
  - Asking questions and getting answers
  - Submitting assignments and projects
  - Viewing grades and feedback

Knowledge Tree

- http://adapt2.sis.pitt.edu/kt/
- Knowledge Tree will be used for:
  - Access to course electronic books
  - Posting lecture slides and videos
  - Access to annotated Java examples

Communication

- To you
  - Watch closely the CourseWeb site for announcements.
  - Check your CIS mail regularly - most important and urgent information will be send to it
  - Send peterb@pitt.edu an e-mail (subject: 2470)
- From you
  - If a question is not personal (an answer could be useful for others) - ask via forum
  - If it is a personal question - ask me or TA by e-mail (can do it from CourseWeb too)
- Office Hours

Assignment 1 (Due next week)

- Try CourseWeb visible features, ask questions, answer questions
- Answer to the user profile questionnaire
- Send your “information card” to Home Page forum
  - Picture, Information about you, Status / goals of your study, Links
  - Goal - present your information for your peers considering to invite you to group projects
- Create a comment to this post commenting your design. Comment design of others
- Post a “hot topic” suggestion/link. Comment on suggestions of your classmates

Interactive System Design

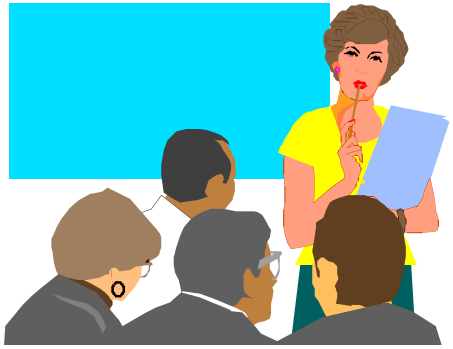
- What are interactive systems?
- Why do we design them?
- How do we know if we've succeeded?
- What happens if we fail?
- How do we maintain a track record of success in design?
- What is the design process?

What are interactive systems?

Humans

(Information) Systems

Interface



Support for human activity

Interface: Old Vision

- Place: Interface is an add-on to the main system functionality, a secondary issue
- Metaphor: Interface is a packaging box, the system is the content of the box.
- Approach: Design the system and then wrap in into the interface (waterfall model)
- Requirements: Interface should look good, large investments in developing an interface are not wise, hiring expensive professional is is pointless

Interface: Modern Vision

- Place: Interface is the core. The needs define the interface. The interface defines the system
  - From the user point of view the interface *is* the system. A good system may be ruined by a poor interface
- Metaphor: Still frame of a building
- Approach: Start with user/task analysis, design the interface part in parallel with the core functionality
- Requirements: good engineering, solid investments, high-quality professionals

Interactive System Design

- Why we design them?
  - To resolve situation of concern
- How do we know if we've succeeded?
  - By testing whether the situation is resolved
    - But we can't do it during design
  - By by measuring or predicting usability
- We will be using *usability engineering approach* to interactive systems design

Usability Factors

- The **speed of performance** of the activity, which affects **how many people** are needed to perform it
- The **incidence of errors** while performing the activity
- The user's ability to **recovery from errors** that occur
- The magnitude of the user's task in **learning to use** the system

Usability Factors

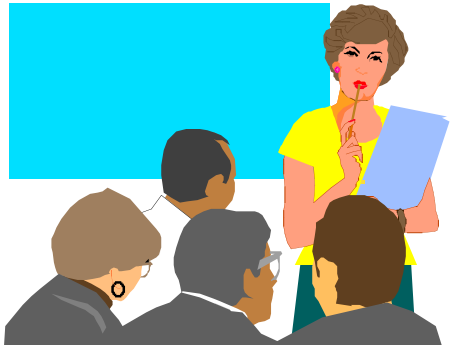
- The user's **retention of learned skills**
- The user's ability to **customize** the system to suit their way of working or the situation of use
- The ease with which people can **reorganize activities** supported by the system—their own activities and other people's
- Users' **satisfaction** with the system

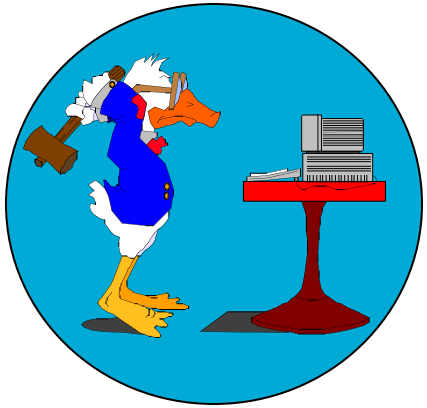
What Happens if we Fail?

Humans

(Information) Systems

Interface





Examples of Failure

- London Ambulance Service, 1992
  - The number of exception messages increased rapidly to such an extent that staff were unable to clear the queue. As the exception message queue grew the system slowed. The situation was made worse as unrectified exception messages generated more exception messages. With the increasing number of “awaiting attention” and exception messages it became increasingly easy to fail to attend to messages that had scrolled off the top of the screen
- Therac-25 X-Ray machine
- USS Vincennes’ Aegis weapons system

How do we maintain a track record of success?

- Approach design as an engineering activity
- Appreciate and study the science part of it “hard knowledge”
- Learn from successful and failed cases
- Collect cases of success, process cases into methods and practices
- Learn from similar engineering disciplines

Introduction: Interfaces

- How do people interact in the process of work? Example: Building a log house.
  - A subject needs to cut trees to build a log house - using other humans and tools
    - No interaction - a set of orders
      - Simple task or expert workers
    - Interactive supervision (giving commands)
    - Do the work yourself using powerful tools (direct manipulation with a chainsaw :)

Introduction: Interfaces

No interface: Batch processing

Command/Dialogue Interface

Graphical User Interface (GUI)

Direct manipulation Interface

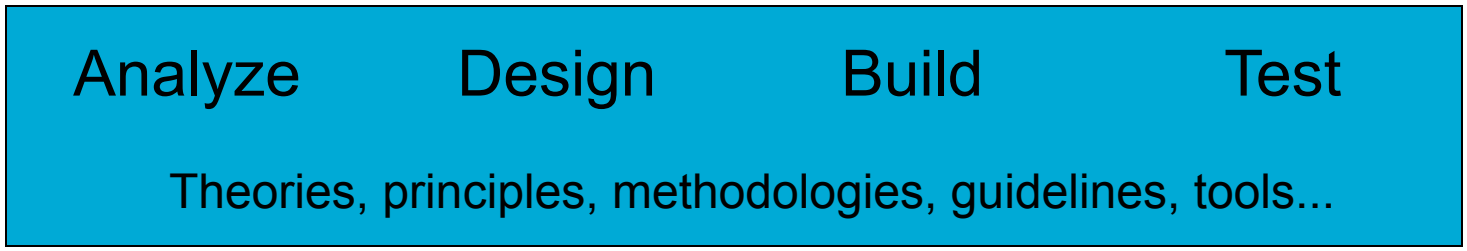
NL and Speech

Virtual reality

Gestures

Commands/Dialogue Direct manipulation

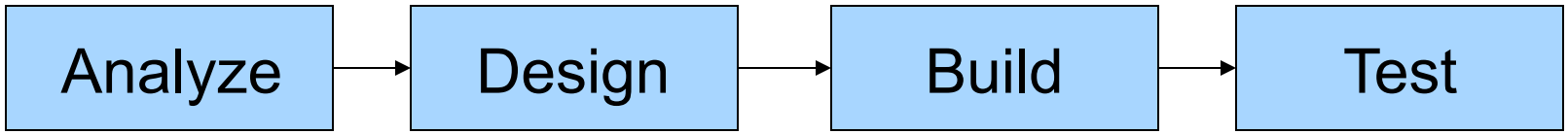
Introduction: Design Overview



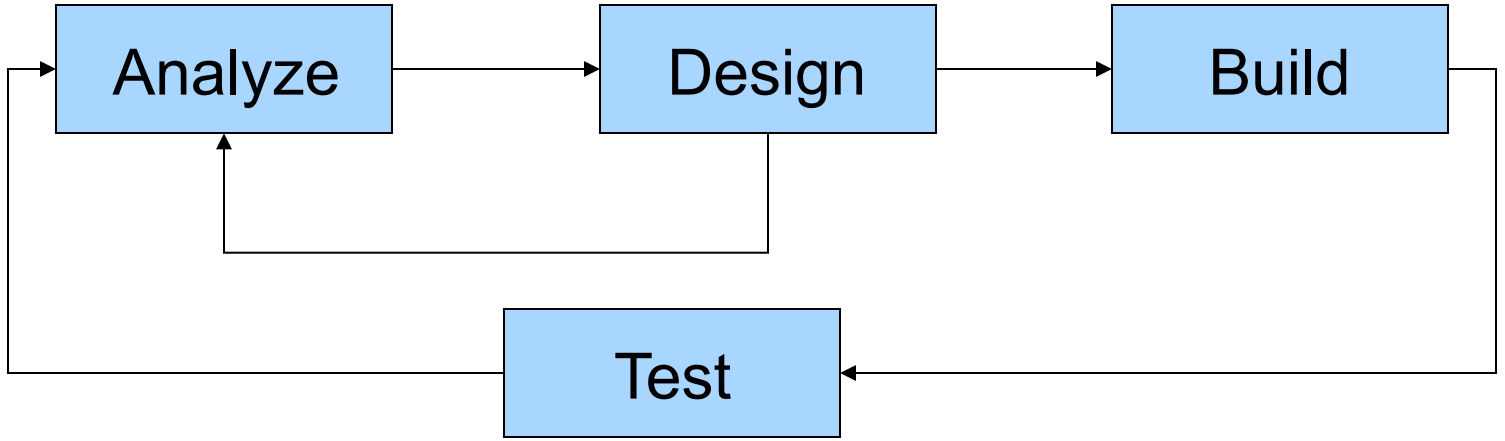
- Iterative nature of design
- How we can afford it
- Methodologies, guidelines?
- Practical methodology : Task-centered design

Iterative Design

Feed-forward (open loop):



The way it should be:



How We Can Afford It?

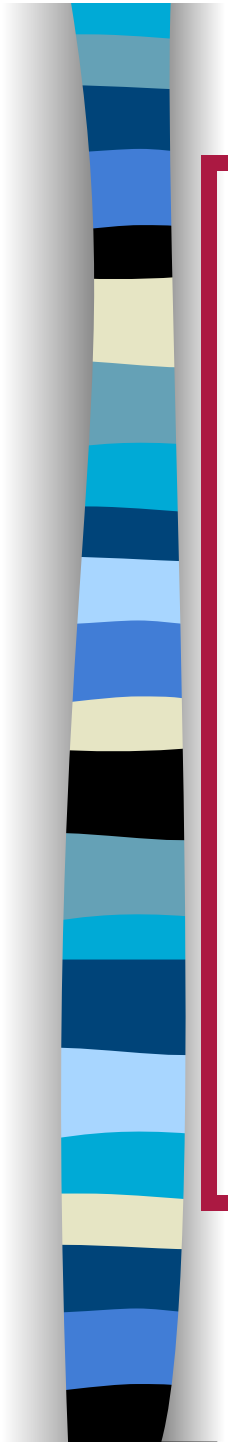
- Incremental increase of complexity and cost in design, building and testing
- Design/Build: more expensive objects
  - Idea ⇨ Draft ⇨ Mock-up ⇨ Prototype …
- Testing: more users, more advanced tests
  - Talk with a colleague ⇨ Discuss in a team ⇨ Perform usability analysis ⇨ Test with 1-2 users ⇨ Test with a variety of users ...

Why Iterative Design?

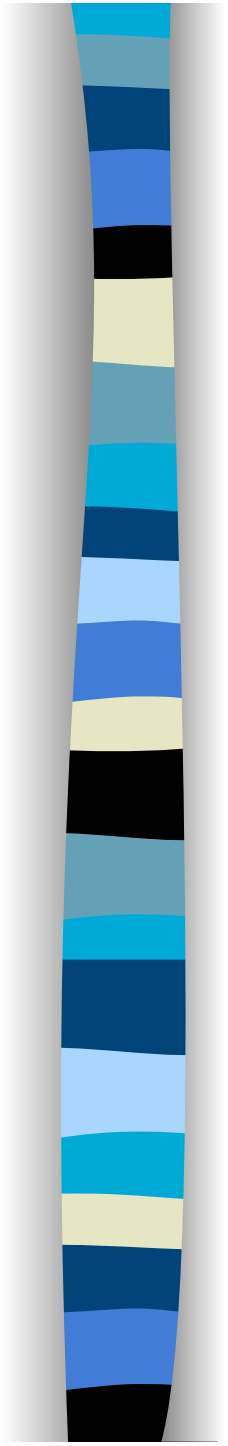
- Mantra of Iterative design:

  The user is not like me

- Users are diverse, you are just one
- Users are not experts (in computer systems, interfaces, etc) -- you are
- Users do not know the background, reasons, way of thinking behind the system being design -- you do

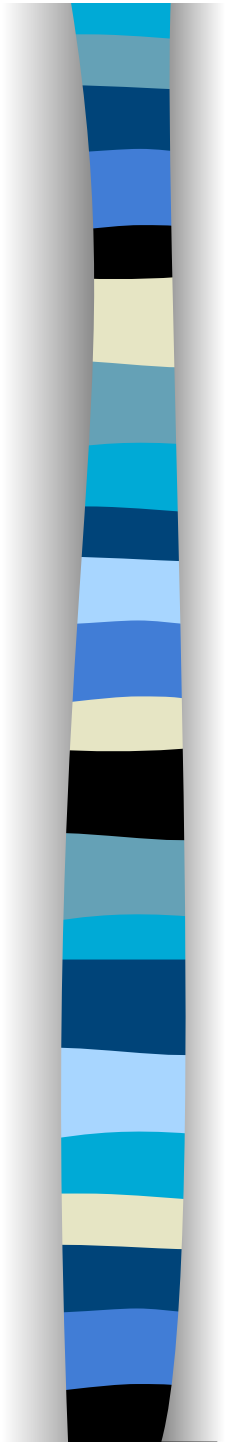


The user
is not like me



Introduction to Java

- Based on C and C++
- Developed in 1991 for intelligent consumer electronic devices
  - Market did not develop, project in danger of being cancelled
- Internet exploded in 1993, saved project
  - Used Java to create web pages with dynamic content
- Java formally announced in 1995
- Now used to create web pages with interactive content, enhance web servers, applications for consumer devices (pagers, cell phones)...



Programming with Java

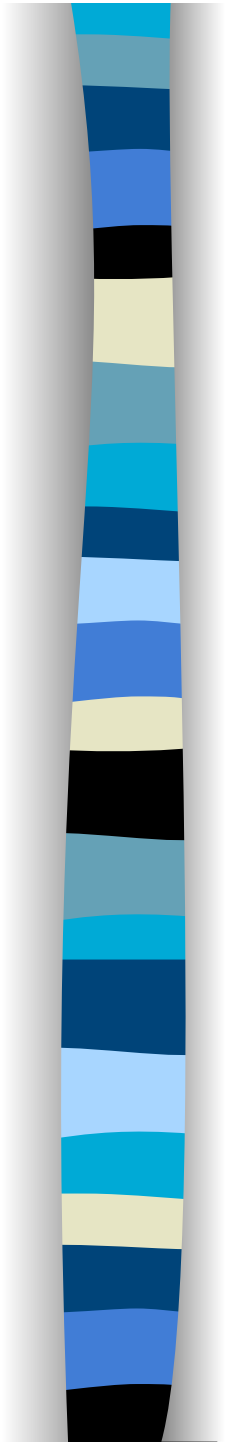
- Java programs
  - Consist of pieces called classes
  - Classes contain methods, which perform tasks
- Class libraries
  - Rich collection of predefined classes, which you can use
  - Also referred as Java API (Applications Programming Interface)
- Two parts to learning Java
  - Learning the language itself, so you can create your own classes (we are not going to do it!)
  - Learning how to use the existing classes in the libraries

Java Programming Cycle

- Edit
  - Use an editor to type Java program
  - **`vi`** or **`emacs`**, notepad, Jbuilder, Visual J++
  - **`.java`** extension
- Compile
  - Translates program into *bytecodes*, understood by Java interpreter
  - **`javac`** command: **`javac myProgram.java`**
  - Creates **`.class`** file containing *bytecodes*

Java Programming Cycle

- Load (applications)
  - Class loader transfers **`.class`** file into memory
    - Applications - run on user's machine
    - Applets - loaded into Web browser, temporary
  - Classes loaded and executed by interpreter with **`java`** command
  - **`java Welcome`**
- Load (applets)
  - HTML documents can refer to Java Applets, loaded by web browsers
  - **`appletviewer`** minimal browser, can only interpret applets
  - To load: **`appletviewer Welcome.html`**



Java Programming Cycle

- Verify
  - Bytecode verifier makes sure bytecodes are valid and do not violate security
  - Java must be secure - possible to damage files (viruses)
- Execute
  - JRE interprets program one bytecode at a time
  - Performs actions specified in program

Tools to use

- Old, Good Way (editor/compiler):
  - Oracle (SUN)'s SDK/JRE
    - Download from http://www.oracle.com/technetwork/java/javase/downloads/index.html
    - Use with some editor (crimson, pfe)
- IDE's
  - Free IBM Eclipse
  - Microsoft VS
  - Educational IDE's (Blue J, JGRASP…)