Slide 1

Introduction to UML

Slide 2

What is UML?

- "Unified Modeling Language".
- From originators' Web site:

  "method for specifying, visualizing, and documenting the artifacts of an object-oriented system under development"
- From *UML Distilled*:

  "family of graphical notations, backed by single meta-model, that help in describing and designing software systems, particularly software systems built using the object–oriented style"
- Many things to many people, used in different ways.

Slide 3

Why Use UML?

- Large and complex systems are difficult for humans to understand. Abstracting out key features ("modeling") and representing pictorially helps.
- Prior to UML, much experimentation with different ways of modeling object-oriented systems. Goal of UML was to unify these.
- A key point — all elements of UML are easy to draw, allow representing ideas at different levels of detail. There are tools specifically designed to work with UML diagrams (of various kinds), but diagrams can be drawn without them.

Slide 4

What You (Probably) Already Know About UML

- In PAD II we use UML "class diagrams" to show things about the classes in a Java program:
  - For each class, its name, attributes (variables), and behavior (methods).
  - Inheritance relationships among classes.
  - Associations between classes — e.g., a ShoppingList class might have an array of ItemAndPrice objects.
- Can become complex, but usually useful in visualizing overall structure of program design.

Slide 5

What UML Can Represent

- Version 2.0 of standard includes at least 12 different types of diagrams:
  Class, Sequence, Object, Package, Deployment, Use Case, State Machine, Activity, Communication, Composite Structures, Component Diagrams, Interaction Overview, Timing
- Quoting from a presentation by Dr. Lewis: “All the diagrams can look pointless or confusing if you don’t understand them and use them well.” With practice, however, their value starts to become more apparent.
- Most likely to be useful in this course are class diagrams and use case diagrams.

Slide 6

Use Cases, My (First) Two Cents’ Worth

- My first reaction — “pointless and confusing”. But with repeated exposure, they’re starting to look more sensible.
- One articles I found on the Web describes use cases as “stories about how the system is supposed to work.”
- Other sources say use cases are as much about writing prose (in a somewhat structured format) as about diagrams. Several of them mention that different formats have been proposed, and there’s not one right way, but you should do whatever seems to work.

Slide 7

Use Cases, Some Basic Ideas

- The idea is to define system/solution requirements in terms of
  - Actors — system users, sometimes system components.
    (ATM example: Actors are customers and technicians.)
  - "Use cases" — each represents one thing the system is supposed to do for an actor (or actors).
    (ATM example: Use cases are "withdraw money", "deposit money", "check balance", and "run diagnostics".)
- If there is common functionality among use cases, can factor it out — "includes" or "uses".
- If a particular use case has some specialized versions (e.g., error situations), can split out — "extends".

Slide 8

Use Cases, My (Second) Two Cents' Worth

- First step in designing something — figure out / decide what it's supposed to do.
- Use cases can help with that.