Understanding Class Definitions

T. M. Murali

21st January 2004
Review: Objects and Classes

- **Object**: represents “things” from the real world or from some problem domain.
- **Class**: represents all objects of a type.
- We can send messages to objects by invoking their methods.
- Methods can have 0 or more parameters that pass additional information for the object to use when executing the method.
- Parameters have types (e.g., boolean, int, string).
- We can create multiple instances of objects from a single class.
- All instances of a class have the same attributes.
- Each object has its own set of values, called the state of the object.
Today: Examine Source Code

- Class definition: fields, constructors, and methods
- Methods: signatures, parameters, return values, and local variables
- Scope and lifetime of a variable
- Some other Java syntax (very similar to Perl)
Example: Naive Ticket Machine

- Explore the behavior of a typical ticket machine.
- Use the naive-ticket-machine project.
- Machine supplies tickets of a fixed price.
- How do we determine that price?
- How do we enter money into the machine?
- How does the machine keep track of the money that is entered?
Constructors

- A **constructor** initialises an object.
- It has the same name as the class.
- It stores initial values into the fields.
- It can receive external parameter values to do so.
Methods implement an object’s behaviour.

**Accessors** provide information about an object.

Methods consist of a header and a body. The header defines the method’s **signature**. The body encloses the method’s statements.

**Mutators** are methods that change an object’s state by modifying the value of one or more fields.
Reflecting on the \textit{naive-ticket-machine class}

- Its behavior is inadequate in several ways:
  - No checks on the amounts entered.
  - No refunds.
  - No checks for a sensible initialization.

- How can we achieve more sophisticated behaviour?
Local Variables

▸ Fields are one category of variable.
  ▸ They store values through the life of an object: lifetime.
  ▸ They are accessible throughout the class: scope.

▸ Methods can include shorter-lived local variables.
  ▸ Their lifetime is the period during which the method is being executed.
  ▸ Their scope is limited to the method.
Summary of Today’s Class

- Fields (instance variables)
- Writing comments
- Constructors
- Scope and lifetime
- Accessor (get) and mutator (set) methods
- Method signatures
- Local variables
- Printing
- Source code and compilation