2 The anatomy of class definitions

Main concepts to be covered

- instance variables
- constructors
- methods
- parameters
- assignment statements
- conditional statements

Ticket machines – an external view

- Exploring the behavior of a typical ticket machine.
  - Use the *naive-ticket-machine* project.
  - Machines supply tickets of a fixed price.
    - How is that price determined?
  - How is 'money' entered into a machine?
  - How does a machine keep track of the money that is entered?

Ticket machines – an internal view

- Interacting with an object gives us clues about its behavior.
- Looking inside allows us to determine how that behavior is provided or implemented.
- All Java classes have a similar-looking internal view.

Basic class structure

```
public class TicketMachine
{
    Inner part of the class omitted.
}
```

The outer wrapper of TicketMachine

```
public class ClassName
{
    Instance variables
    Constructors
    Methods
}
```

The contents of a class

Instance variables

- Instance variables store values for an object.
- They are also known as *fields*.
- Use the *Inspect* option to view an object's instance variables.
- Instance variables define the state of an object.

```
public class TicketMachine
{
    private int price;
    private int balance;
    private int total;

    Further details omitted.
}
```

visibility modifier, type, variable name

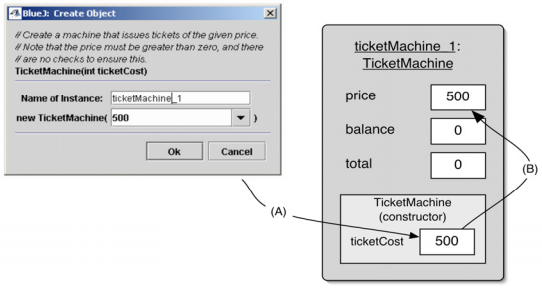
```
private int price;
```

Constructors

- Constructors initialize an object.
- They have the same name as their class.
- They store initial values into the instance variables.
- They often receive external parameter values for this.

```
public TicketMachine(int ticketCost)
{
    price = ticketCost;
    balance = 0;
    total = 0;
}
```


Passing data via parameters



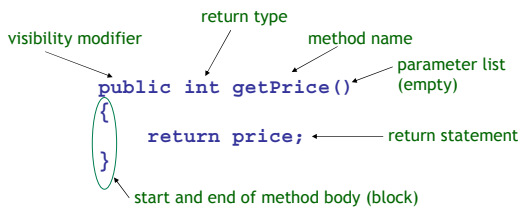
Assignment

- Values are stored into instance variables (and other variables) via assignment statements:
  - *variable* = *expression*;
  - `price = ticketCost;`
- A variable stores a single value, so any previous value is lost.

Accessor methods

- Methods implement the behavior of objects.
- Accessors provide information about an object.
- Methods have a structure consisting of a header and a body.
- The header defines the method's *signature*.
  `public int getPrice()`
- The body encloses the method's statements.

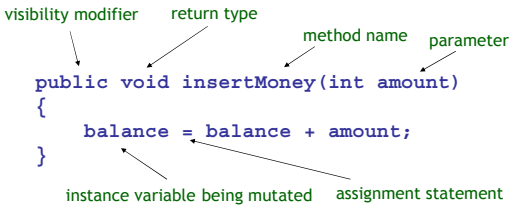
Accessor methods



Mutator methods

- Have a similar method structure: header and body.
- Used to *mutate* (i.e., change) an object's state.
- Achieved through changing the value of one or more instance variables.
  - Typically contain assignment statements.
  - Typically receive parameters.

Mutator methods



Printing from methods

```
public void printTicket()
{
    // Simulate the printing of a ticket.
    System.out.println("####################");
    System.out.println("# The BlueJ Line");
    System.out.println("# Ticket");
    System.out.println("# " + price + " cents.");
    System.out.println("####################");
    System.out.println();

    // Update the total collected with the balance.
    total = total + balance;
    // Clear the balance.
    balance = 0;
}
```

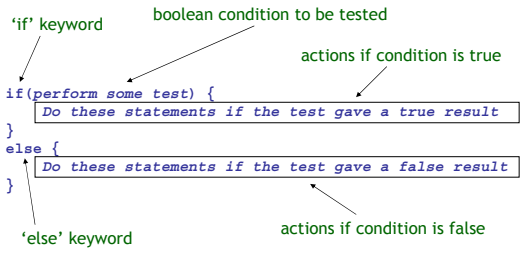

Reflecting on the ticket machines

- Their behavior is inadequate in several ways:
  - No checks on the amounts entered.
  - No refunds.
  - No checks for a sensible initialization.
- How can we do better?
  - We need more sophisticated behavior.

Making choices

```
public void insertMoney(int amount)
{
    if(amount > 0) {
        balance = balance + amount;
    }
    else {
        System.out.println("Use a positive amount: " +
                           amount);
    }
}
```

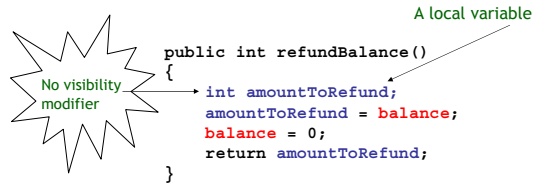

Making choices



Local variables

- Instance variables are one sort of variable.
  - They store values through the life of an object.
  - They are accessible throughout the class.
- Methods can include shorter-lived *local* variables.
  - They exist only as long as the method is being executed.
  - They are only accessible from within the method.

Local variables



Review

- Class bodies contain instance variables, constructors and methods.
- Instance variables store values that determine an object's state.
- Constructors initialize objects.
- Methods implement the behavior of objects.

Review

- Instance variables, parameters and local variables are all variables.
- Instance variables persist for the lifetime of an object.
- Parameters are used to receive values into a constructor or method.
- Local variables are used for short-lived temporary storage.

Review

- Objects can make decisions via conditional (if) statements.
- A true or false test allows one of two alternative courses of actions to be taken.