Class API

口 a class API can contain three different types of methods:
1. constructors
2. class methods - have keyword static in method header
3. instance methods - method header exactly like class methods except no static

Constructors

口 purpose is to initialize the object
口 give initial values to the object attributes
口 invoked using the new operator when an object is instantiated
口 method name is the same as the class name
口 have no return type (not even void)
Default Constructor

✴ the default constructor is a constructor that takes no arguments
✴ initializes object attributes to default values

EX: Square Class

Square Class API
Constructors

public Square()

Default constructor to create a Square that is red, 30 pixels in size, is located at (60, 50), and is not initially visible.

Invoking Constructors

✴ invoked with the new operator
✴ EX:

Square s1 = new Square();

invokes default Square constructor

Multiple Constructors

✴ a class may have more than one constructor
✴ the constructors must take different arguments
✴ Java invokes the correct constructor based on the arguments you give it
**EX: Person Class**

**Person Class API**

**Constructors**

```java
public Person()
```

Constructor for a `Person` named “John Doe” who is 0 years old. Initially this `Person` is alive.

```java
public Person(String name, int age)
```

Constructor for a `Person` with the specified name and age. Initially this `Person` is alive.

---

**Invoking Constructors**

EX:

```java
Person p1 = new Person();
Person p2 = new Person("Mary", 20);
```

---

**Person Class**

- API, compiled class file `Person.class`, and an example program `PersonApp.java` are on the class website
- spend some time looking at this example
- note, for instance, that some `Person` instance methods return values
  - work just like class methods, except require an object to invoke

---

**Standard Class Library**

- the Java Standard Class Library provides hundreds of useful classes for your use
- comes free with Java
- FYI: a link to the class APIs is on the website Related Links page
Packages

- The Java Standard Class Library classes are organized into packages
- Classes in the java.lang package are available to all programs by default
  - Includes the classes Math and System

String Class

- String is actually a class in the java.lang package, not a data type
- Every String is an object
- An abbreviated String class API is on the website

String Class

- Because Strings are so important in writing programs, Java gives Strings two unique capabilities
  - Concatenation using the + operator
  - Abbreviated instantiation syntax (see next slide)
- No other classes have these capabilities

String Instantiation

- We have initialized Strings using the following syntax:
  
  ```java
  String s = “abc”;
  ```
- This is really special abbreviated syntax for:
  
  ```java
  String s = new String(“abc”);
  ```
- These statements are equivalent
- Both create a String object containing “abc”
Length Method

- the length instance method returns how many characters are in a String object
- it has the method header:
  ```java
  public int length()
  ```
- EX:
  ```java
  String str = new String("ab");
  int n;
  n = str.length();
  • n would have the value 2
  ```

charAt Method

- the charAt instance method takes a single integer argument, and returns the character at that index in the String
- it has the method header:
  ```java
  public char charAt(int index)
  ```

charAt Method

- the first character in a String is at index 0
- the last character is at length() - 1
- EX:
  ```java
  String s = new String("HELLO");
  char c;
  c = s.charAt(1);
  • c would have the value ’E’
  ```

Comparing Objects

- objects cannot be compared using the regular relational operators (==, >, etc.)
- classes usually provide instance methods for comparing objects of that class
Equals Method

✴ the `String` class provides the `equals` instance method that returns whether or not two `Strings` have the same value
✴ it has the method header:
  ```java
  public boolean equals(String s)
  ```

EX:
```java
String a = new String("joe");
String b = new String("mary");
if (a.equals(b))
  Output.showMessage("same");
else
  Output.showMessage("diff");
```

• would output "diff"
✴ note that `a.equals(b)` and `b.equals(a)` have the same value

Class Availability

✴ there are three ways to make a class available to use in your programs:

  1. put compiled `.class` file in same folder with your program
     • EX: `Output`, `Square`, `Person`
  2. classes in `java.lang` package are automatically available
     • EX: `Math`, `String`
  3. classes in all other packages require an `import` statement

Import Statement

✴ syntax:
  ```java
  import packageName.className;
  ```

 Java keyword   dot operator
✴ put `import` statements just below your header comments, and just above your class header
DecimalFormat Class
• the DecimalFormat class in the java.text package lets you format doubles for output
• requires the following import statement:
  import java.text.DecimalFormat;
• an abridged DecimalFormat class API is given on the website

DecimalFormat Class
• to use, you must first create a DecimalFormat object with the desired format
• the constructor has the method header:
  public DecimalFormat (String formatStr)
  • the format string argument specifies how the object will format doubles
  • a format string of “0.00” indicates a format of two digits to the right of the decimal place

Format Method
• the format instance method has the following method header:
  public String format (double num)
• the method takes a single real number as its argument, and returns that number as a formatted String

DecimalFormat EX
DecimalFormat df = new DecimalFormat("0.00");
String s;
s = df.format(1.123456789);
Output.showMessage(s);

will output 1.12
DecimalFormat EX

The program Format.java contains another DecimalFormat example. Note the required import statement above the class.

Module 19 Vocabulary

- constructor
- default constructor
- Java Standard Class Library
- package
- import statement

Questions?

Email Kevin at sahrk@sou.edu