Lecture 03
Java APIs
Strings and IOs

Ananda Gunawardena

Java APIs

- Think Java API (Application Programming Interface) as a super dictionary of the Java language.
  - It has a list of all Java packages, classes, and interfaces; along with all of their methods, fields and constructors.
  - `java.lang` intrinsic classes (`String`, etc)
  - `java.io` reading and writing
  - `java.util` Java Collection Framework and utility classes
  - `javax.swing` GUI
- Think Java API as the interface to manipulate Java classes as black boxes
  - It tells you how to use Java classes but little how they are implemented
- Quick Demo

Java APIs

- Any serious Java programmers should use the APIs to develop Java programs
- Best practices of using APIs
  - Download APIs to your local computer from http://java.sun.com/j2se/1.5/download.html
  - Treat it as a dictionary for reference, instead of a book from front to back

Strings and `String`

- Strings are an inevitable part of any programming task.
    - Printing messages (e.g. instant messaging with friends, output debugging information)
    - Representing data (e.g. student names)
    - Referring to files on disk (e.g. “c:\\file.txt”)
- Intuitively, think strings as a sequence of character
- In Java, `String` has an array of `char` as its internal representation.

String Properties

- `String` is a class, not a primitive type.
  - It has fields, constructors, and methods.

- Examples
  - Construct a string

    ```
    String s = "Hello world";
    ```
  - Get the length of the string

    ```
    int length = s.length();//length = 11
    ```
  - Access a part of the string

    ```
    String sub = s.substring(6);//"world"
    char c = s.charAt(1); //'e'
    ```
  - Concatenate strings

    ```
    String t = " of java";
    String s1 = s + t; //"Hello world of java"
    ```

String Properties

- `String` is immutable.
  - Once created, it cannot be changed. There is no such a method `setChar(char c)`
  - Why? Security reasons. (The detailed explanation is complicated and not useful in this course. Doing the following exercises is more helpful in understanding the concept.)

String Properties

- Q: What are the values of charArray1 and charArray1 at the end of the code

```
char[] charArray1 = {'a', 'b', 'c'};
char[] charArray2 = charArray1;
charArray1[1] = 'i';

//charArray1 = {'a', ‘i', 'c'}
//charArray1 = {'a', ‘i', 'c'}
```

String Properties

Q: What are the values of s1 and s2 at the end of the code

```
String s1 = "abc";
  String s2 = s1 ;
  s1 = s1 + "def";

  // s1 = "abcdef"
  //s2 = "abc"
```

String Main Usage

- Comparing strings
  - Q: Are “word” and “work” equal?
  - Which is bigger?

```
String s1 = "word";
 String s2 = "work";
 System.out.println(s1.equals(s2)); //false
System.out.println(s1.compareTo(s2)); // -3, meaning
"word” < "work"
// two strings are compared lexicographically. The
result is a negative integer if this s1
lexicographically precedes s2. The result is a positive
integer if s1 lexicographically follows s1. The result
is zero if the strings are equal;
```

String Main Usage

- Check if two strings have the same prefix and postfix
  - Q: Does “hello world” starts with “hell”?

```
String s = "hello world";
 System.out.println(s.startsWith("hell"));//true
 System.out.println(s.endsWith("world"));//true
```

- Convert cases
  - Q: I want to turn “hello world” into upper cases.

```
String sUpper = s.toUpperCase();//"HELLO WORLD"
```

String Main Usage

- Finding the index of a substring
  - Q: I want to know the position of o’s in the string

```
 String s = "Hello world";
 s.indexOf("o");//4, return -1 if the argument string
not found
 s.lastIndexOf("o");//7
```

String main usage – Special characters

Use \ to proceed the special characters

```
Tab  \t
New line \n
Carriage return \r
Single quote \’
Double quote \”
Backslash \\
```

Q: What is output of the strings?

```
System.out.println("a\nb\tc“);
//a
//b  c
```

Q: How do you represent the file path “c:\data\file.txt”?

```
String filePath = “c://data//file.txt”
```

String Advanced Usage – breaking words apart

Q: I want to break "Hello world of java“ into "Hello” “world” “of” “java“

```
String s = "Hello world of java";
    StringTokenizer st = new StringTokenizer(s);
    while (st.hasMoreTokens()) {
      System.out.println(st.nextToken());
    }

//Hello
//world
//of
//java
```

String Advanced Usage – breaking words apart

Q: I want to break "Hello, world of java“ into "Hello” “world” “of” “java” (this is a comma in the string)

```
String s = "Hello, world of java";
    StringTokenizer st = new StringTokenizer(s);
    while (st.hasMoreTokens()) {
      System.out.println(st.nextToken());
    }

//Hello,
//world
//of
//java
```

String Advanced Usage – breaking words apart

Q: I want to break "Hello, world of java“ into "Hello” “world” “of” “java“

```
 String s = "Hello, world of java";
    StringTokenizer st = new StringTokenizer(s, ", ");
    //”, “ is the delimiter. Notice we need to supply all the
    possible delimiters in the string. “,” and a space “ “
    while (st.hasMoreTokens()) {
      System.out.println(st.nextToken());
    }
```

```
Hello
world
of
java
```

String Advanced Usage – Concatenate strings repetitively

- Q: Suppose you have a list of 100 strings. You want to concatenate them into one string?
  Solution 1

```
List stringList = new ArrayList();
  for (int i = 0; i < 10; i++) {
    stringList.add("string" + i);
  }

  String finalString = "";
  for (Iterator iter = stringList.iterator();      iter.hasNext();
) {
    String s = (String) iter.next();
    finalString += s;
  }
//work but inefficient for large strings and large amount of
  concatenation because String is immutable. Intuitively, Java
  needs to maintain an a old string, a new string, and a
  concatenate string in each iteration.
```

String Advanced Usage – Concatenate strings repetitively

- Q: Suppose you have a list of 100 strings. You want to concatenate them into one string?
  Solution 2

```
List stringList = new ArrayList();
  for (int i = 0; i < 10; i++) {
    stringList.add("string" + i);
  }

  StringBuffer sb = new StringBuffer(); //StringBuffer is
mutable. Recommended for a large amount of string
concatenation.
  for (Iterator iter = stringList.iterator(); iter.hasNext(); )
{
    String s = (String) iter.next();
    sb.append(s);
  }
```

String Advanced Usage – String and OOP

- Q: I have just written a class. How can I represent the class with a string?

```
public class Point {
    double x;
    double y;
    Point(double x, double y) {
      this.x = x;
      this.y = y;
    }
}

Point p = new Point(2,3);
System.out.println(p);   //Point@360be0
```

String Advanced Usage – String and OOP

- Q: I have just written a class. How can I represent the class with a string?
  - Overwrite toString()

```
public class Point {
    double x;
    double y;
    Point(double x, double y) {
      this.x = x;
      this.y = y;
    }
  public String toString() {
      return "x = " + x + ", y = " + y;
    }
}

Point p = new Point(2,3);
System.out.println(p);    //x = 2.0, y = 3.0
```

String Advanced Usage – Convert between strings to primitives

- Q: I want to convert an integer 123 to a string and convert a string “123” to an integer
  - Use `String.valueOf(), Integer.parseInt()`

```
int i = 123;
String s = i + “”;
String t = String.valueOf(i);
String u = Integer.toString(i); //s = t = u = 123

int j = Integer.parseInt(s); // j = 123


//Extension - Double.parseDouble();
```

Java IO

Java IO

- Most programs need to interact with the outside world by reading and writing files from/to a hard disk
  - Microsoft Office
  - Matlab
  - Eclipse
  - Your future programs…
- Java provides a comprehensive package to deal with file input and output.

Java 5 Scanner Class

java.util

Class Scanner

How to read an integer from stdin

```
Scanner sc = new Scanner(System.in);
int i = sc.nextInt();
```

How to read a set of long integers from a file

```
Scanner sc = new Scanner(new FileReader("file.txt"));
while (sc.hasNextLong()) {
    System.out.println(sc.nextLong());
}
```

Scanner

How to read a set of strings(words) from a file

```
Scanner sc = new Scanner(new FileReader("file.txt"));
while (sc.hasNext()) {
    System.out.println(sc.next());
}
```

Scanner

Some useful scanner methods

boolean **hasNext()**

Returns true if this scanner has another token in its input.

boolean **hasNext(String pattern)**

Returns true if the next token matches the pattern constructed from the specified string.

boolean **hasNextDouble()**

Returns true if the next token in this scanner's input can be interpreted as a double value using the nextDouble() method.

Other IO methods

Java IO

- Q: How do I output something to the screen?
  - To give user the result, or debug the code
  - By using `System.out.println()`

```
      Object anObject = new Object();
      String myAnswer = "no";
      int i = 42;
      System.out.println("Hello, World of Java");
      System.out.println("An object is " + anObject);
      System.out.println("The answer is " + myAnswer + " at
this time.");
      System.out.println("The answer is " + i + '.');

 //Hello, World of Java
 //An object is java.lang.Object@18d107f
 //The answer is no at this time.
 //The answer is 42.
```

Java IO

- Q: How do I read input from the keyboard?
  - To get input from the user from a consol instead of a GUI
  - By using `System.in.read()`

```
int b = 0;
try {
        b = System.in.read();//returns an integer
} catch (Exception e) {
        System.out.println("Caught " + e);
}
System.out.println("Read this data: " + (char)b);
```

```
// Read this data: 1       (after I typed 1)
// Read this data: 1       (after I typed 123)
// Read this data: a       (after I typed a)
//This gives you the ability to read one byte at a time.
```

Java IO

- Q: How do I read a line from the keyboard?
  - Use `BufferedReader is = new BufferedReader(new InputStreamReader(System.in));`

```
      try {
          BufferedReader is = new BufferedReader(new
InputStreamReader(System.in));
          String inputLine;
          while ((inputLine = is.readLine()) != null) {
              System.out.println(inputLine);
          }
          is.close();
      } catch (IOException e) {
          System.out.println("IOException: " + e);
      }
 //  abc        (after I typed abc)
 //This gives you the ability to read one line at a time.
```

Java IO

- Quiz: How do I read a three-digit integer from the keyboard?
  - Use `Integer.parseInt()`

```
      try {
            BufferedReader is = new BufferedReader(new
InputStreamReader(System.in));
            String inputLine;
            while ((inputLine = is.readLine()) != null) {
               int value = Integer.parseInt(inputLine);//parse
the string into an integer
               System.out.println(value);
            }
            is.close();
      } catch (IOException e) {
            System.out.println("IOException: " + e);
      }
 // 123       (after I typed 123)
 //This gives you the ability to read in any primitives.
```

Java IO

- Q: How do I read a file from the hard drive?
  - To read all the data into memeory, manipulate it and output the result, common in engineering, statistics, machine learning applications
  - Use `BufferedReader is = new BufferedReader(new FileReader("c://data.txt"));`

```
    try {
      BufferedReader is = new BufferedReader(new
FileReader("c://data.txt"));
           String inputLine;
          while ((inputLine = is.readLine()) != null) {
              System.out.println(inputLine);
          }
          is.close();
      } catch (IOException e) {
          System.out.println("IOException: " + e);
      }

// This   (data.txt contains the same four lines of words)
//is
//the
//data
```

Java IO

- Q: How do I write data to the hard drive?
  - Use `PrintWriter out = new PrintWriter(new BufferedWriter(new FileWriter("c://output.txt")));`

```
 try {
        PrintWriter out = new PrintWriter(new
BufferedWriter(new FileWriter("c://output.txt")));
         out.println("output data");
        out.close();
      }
      catch (IOException e) {
        System.out.println("IOException: " + e);
      }
//you will find a file named output.txt under c:\ and it
  contains a line "output data“.
```

Java IO

- There are two kinds of files
  - Files of characters (texts)
  - Files of bytes (primitives, arrays, objects)
- java.io package contains

	Input	Output

Character streams	`*Reader*`
	  `BufferedReader`
	  `InputStreamReader`
	    `FileReader`
		`*Writer*`
		  `BufferedWriter`
		  `FileWriter`
		  `PrintWriter`

Byte streams	`*InputStream*`	`*OutputStream*`
		  `PrintStream`

Abstract classes are shown in *italics*.

Java IO

- Q: What is System.out and Sytem.in?
  - System.out is PrintStream, which has several println() methods for differnet data type.
  - Sytem.in is InputStream, which has only three read() methods, which are for integers only

- Q: Why do you wrap one class in another `BufferedReader is = new BufferedReader(new InputStreamReader(System.in));`?
  - System.in is an InputStream.
  - An InputStreamReader is a bridge from byte streams to character streams: It reads bytes and decodes them into characters.
  - Each invocation of InputStreamReader's read() causes one or more bytes to be read from the underlying byte-input stream. (inefficient)
  - BufferedReader read more bytes ahead from the underlying stream. (more efficient)

Java IO

- Q: What else can you do with IO?
  - Read and write binary data using DataInputStream and DataOutputStream
  - Read and write Java objects using using ObjectInputStream and ObjectOutputStream
  - Exchange data streams with other programming languages (e.g C++)
  - Read and write files from a Jar archive or a zip file.

- Q: They look fancy. How can I do it?
  - Study Java API when you need, and google.

Summary

- Java API is a dictionary of the Java language, and the interface for programmers to manipulate Java classes as black boxes.
- String is a class, and immutable.

  Construct a string `String s = "Hello world";`

  Get its length `s.length();`

  Get a substring `s.substring(6)`

  Concatenate strings `s + t or StringBuffer`

  Compare strings `s1.equals(s2), s1.compareTo(s2)`
  `s.startsWith("hell"), s.endsWith("world")`

  Convert cases `s.toUpperCase();`

  Finding the index of a substring `s.indexOf("o")`

  Break a string apart `StringTokenizer st = new StringTokenizer(s);`

  Convert a string to primitives `Integer.parseInt(s);`

  Convert a primitive to a string `4+ “”`

Summary

- Data can be in the form of characters or bytes.
- IO is comprehensive

  Output a message to screen

```
System.out.println("Hello, World of Java");
```

  Output text data to a file

```
PrintWriter out = new PrintWriter(new BufferedWriter(new
FileWriter("c://output.txt")));
```

  Read an integer from the consol

```
int b = System.in.read();
```

  Read a line from the consol

```
BufferedReader is = new BufferedReader(new
InputStreamReader(System.in));
```

  Read a file

```
BufferedReader is = new BufferedReader(new
FileReader("c://data.txt"));
```