

COMP102 2011: Java Documentation

Brief and partial documentation of UI and other Java classes and methods

UI class:

```
// Text output
public void clearText()           // Clears the text pane
public void print(anything val)   // Prints val with no newline
public void println(anything val) // Prints val and newline
public void println()            // Prints a newline
public void printf(String format, ...) // Prints the format string, inserting the remaining
                                       // arguments at the %'s in the format string :
                                       // %s for Strings .
                                       // %d for ints, (%3d: use at least 3 characters ),
                                       // %4.2f for 2dp doubles,
                                       // Use \n for newline

// Text input
public String askString(String question) // ask the user for a line of text
public int askInt(String question)       // ask the user for an integer
public double askDouble(String question) // ask the user for a number
public String askToken(String question)  // ask the user for a single token
public boolean askBoolean(String question) // ask the user for a yes/no or true/false value

public String next()                 // Read and return next token of the user's input
public boolean nextBoolean()         // Read next token of input (must be a yes/no or true/false )
public double nextDouble()           // Read next token of input, return as double. (must be a number)
public int nextInt()                 // Read next token of input, return as int. (must be an integer )
public String nextLine()             // Read and return rest of input line
public boolean hasNext()             // Returns true iff there is another token in the user input.
public boolean hasNextBoolean()      // Returns true if next token of input is yes/no or true/false
public boolean hasNextDouble()       // Returns true if next token of input is a number
public boolean hasNextInt()          // Returns true if next token of input is an integer
public boolean hasNextLine()         // Returns true

// Graphics output
public void clearGraphics()           // Clears the graphics pane
public void setColor(Color c)        // Change the colour for later drawing commands
public void repaintGraphics()         // Updates and redisplay graphics pane
```

```

// draw outlines of shapes
public void drawLine(double x1, double y1, double x2, double y2)
public void drawRect(double left, double top, double width, double height)
public void drawOval(double left, double top, double width, double height)
public void drawArc(double left, double top, double width, double height, double startAngle, double arcAn
public void drawString(String s, double x, double y)
public void drawImage(String filename, double x, double y)
public void drawImage(String filename, double x, double y, double width, double height)

// draw solid shapes (not outlines)
public void fillRect (double left , double top, double width, double height)
public void fillOval (double left , double top, double width, double height) // Draws solid oval
public void fillArc (double left , double top, double width, double height, double startAngle, double arcAn

// erasing and inverting : all the drawXXX commands also have an eraseXXX and invertXXX form

// Other input
public void addButton(String name, UIButtonListener controller) // Add a button to input panel
public void addTextField(String s, UITextFieldListener obj) // Add a textField to input panel
public void addSlider(String s, double min, double max, UISliderListener obj) // Add a slider to input pa
public void addSlider(String s, double min, double max, double init, UISliderListener obj)
public void setMouseListener(UIMouseListener obj) // Set mouseListener for Graphics pane
public void setMouseMotionListener(UIMouseListener obj) // Set mouseMotionListener for Graphics pane

// Misc
public void sleep(double millis) // pause program for specified time ( milliseconds ).
public void initialise () // ensure UI window has been initialised
public void quit () // delete UI window; usually halts the program.

```

Trace class:

```

public void print (anything val) // If Trace visible , prints val with no newline
public void println (anything val) // If Trace visible , Prints val and newline
public void println () // If Trace visible , Prints a newline
public void printf (String format, ...) // If Trace visible , Prints formatted string

```

Integer class:

```
public static final int MAX_VALUE // The largest possible int: 2(31)-1  
public static final int MIN_VALUE // The smallest possible int: -231
```

// Typical usage:

```
int max = Integer.MIN_VALUE; // find maximum of a file of integers  
while (scan.hasNextInt()){  
    int num = scan.nextInt();  
    if (num > max){  
        max = num;  
    }  
}
```

Double class:

```
public static final double MAX_VALUE // The largest possible double: just under 2(1024)  
public static final double MIN_VALUE // The smallest possible positive nonzero double  
public static final double POSITIVE_INFINITY // positive infinity (greater than any number)  
public static final double NEGATIVE_INFINITY // negative infinity (less than any number)  
public static final double NaN // The double that is 'Not a Number'
```

// Typical usage:

```
double min = Double.POSITIVE_INFINITY; // find minimum of an array of doubles  
for (int i=0; i<numbers.length; i++){  
    if (numbers[i] < min){ min = numbers[i]; }
```

Math class:

```
public static double sqrt(double x) // Returns the square root of x  
public static double min(double x, double y) // Returns the smaller of x and y  
public static double max(double x, double y) // Returns the larger of x and y  
public static double abs(double x) // Returns the absolute value of x  
public static int min(int x, int y) // Returns the smaller of x and y  
public static int max(int x, int y) // Returns the larger of x and y  
public static int abs(int x) // Returns the absolute value of x  
public static double random() // Returns a random number between 0 and 1.0  
public static double hypot(double dx, double dy) // Returns sqrt(dx*dx + dy*dy)
```

// Typical usage:

```
double diagonal = Math.sqrt(Math.sqrt(wd) + Math.sqrt(ht));  
if ( Math.random()<0.1) { ... // do something with probability 0.1  
int size = (int)(Math.random()*50); // a random integer between 0 and 49.
```

String class:

```
public int length() // Returns the length (number of characters) of the string
public boolean equals(String s) // Returns true if string has same characters as s
public boolean equalsIgnoreCase(String s) // String has same characters as s, ignoring their case
public String toUpperCase(String s) // Returns upper case copy of string
public String toLowerCase(String s) // Returns lower case copy of string
public boolean startsWith(String s) // Returns true if first part of string matches s
public boolean contains(String s) // Returns true if s matches some part of the string
public String substring(int j, int k) // Returns substring from index j to index k-1
public String trim(): // Returns copy of string with spaces at ends removed
public int indexOf(String s) // Returns the index of where s first matches
// Returns -1 if string does not contain s anywhere
```

// Typical usage:

// assume name, answer, and courseCode are all variables of type String

```
if ( answer.equals(name) ) { ...
System.println("Ans : "+ answer.toLowerCase());
if ( name.startsWith("Pe") ) { ...
System.println("Course number =" + courseCode.substring(4, 7));
```

Color class:

```
public Color(int red, int green, int blue) // Make a colour; arguments must be 0..255
public Color(float red, float green, float blue) // Make a colour; arguments must be 0.0 to 1.0
Color.gray, Color.blue, Color.red, // Some of the predefined colours
Color.green, Color.black, Color.white
```

// Typical usage:

```
Color col = new Color((float) Math.random(),(float) Math.random(),(float) Math.random());
```

Listener Interfaces:

MouseListener interface:

```
public void mousePerformed(String action, double x, double y)
// Called when mouse event happens.
// The action argument may be “pressed”, “released”, “clicked”, “moved”, or “dragged”
```

UIButtonListener interface:

```
public void buttonPerformed(String button) // Called when button pressed
```

UITextFieldListener interface:

```
public void textFieldPerformed(String field , String value) // Called when value entered in text field
```

UISliderListener interface:

```
public void sliderPerformed(String slider , double value) // called when slider changed
```

```
PrintStream class:                                     // Note, System.out is a PrintStream object
public PrintStream (File f)                             // Constructor, for printing to a file
public void close()                                     // Close the file (if it is wrapping a File object)
public void print (String s)                            // Prints s with no newline
public void print (int i)                              // Prints i with no newline
public void print (double d)                           // Prints d with no newline
public void println ()                                 // Prints a newline
public void println (String s)                         // Prints s followed by newline
public void println (int i)                            // Prints i followed by newline
public void println (double d)                        // Prints d followed by newline
public void printf (String format, ...)                // Prints the format string, inserting the remaining
                                                    // arguments at the %'s in the format string:
                                                    // %s for Strings.
                                                    // %d for ints, (%3d: use at least 3 characters),
                                                    // %4.2f for 2dp doubles,
                                                    // Use \n for newline
```

// Typical usage:

```
PrintStream output = new PrintStream(new File("data.txt"));
output.println (age);
output.println (height);
System.out.println("Hi, I'm " +name+ " (age: "+ age+ ", height: "+height+"m)");
```

Note: Although the UI is not a PrintStream, it shares all the print... methods and behaves the same way, but prints to the UI text pane.

File class:

```
public File (String fname) // Constructor. Creates a File object attached to the file named fname
public boolean exists() // Returns true if and only if the file already exists
```

// Typical usage:

```
File outFile = new File(UIFileChooser.save("File name"));
if ( outFile.exists() ){
    Scanner sc = new Scanner(outFile);
    while (sc.hasNext()){
        System.out.println(sc.nextLine());
    }
    sc.close();
}
```

Scanner class:

```
public Scanner (File f)           // Constructor, for reading from a file
public Scanner (InputStream i)    // Constructor. Note: System.in is an InputStream
public Scanner (String s)        // Constructor, for reading from a string
public boolean hasNext()         // Returns true if there is more to read
public boolean hasNextInt()      // Returns true if the next token is an integer
public boolean hasNextDouble()  // Returns true if the next token is a number
public String next()             // Returns the next token (chars up to a space/line)
public String nextLine()        // Returns string of chars up to next newline
                                   // (next and nextLine throw exception if no more tokens)
public int nextInt()            // Returns the integer value of the next token
                                   // (throws exception if next token is not an integer or no more tokens)
public double nextDouble()      // Returns the double value of the next token
                                   // (throws exception if next token is not a number or no more tokens)
public void close()             // Closes the file (if it is wrapping a File object)
```

// Typical usage:

```
Scanner scan = new Scanner(new File("data.txt")); // find total of numbers in a file of integers
double total = 0;
while (scan.hasNextDouble()) {
    total = total + scan.nextDouble();
}
scan.close();
```

Note: Although the UI is not a Scanner, it shares all the next... methods and behaves the same way, but gets input from the UI textpane.

UIFileChooser class:

```
public static String           // Ask user for an existing file
public static String open(String prompt)
public static String save()    // Ask user for a new or existing file
public static String open(String prompt)
```

// Typical usage:

```
String imgFileName = UIFileChooser.open("Choose image file");
File outFile = new File(UIFileChooser.save());
for (int i=0; i<count; i++) {
    outFile.println (data[i].toString ());
}
```
