PLEASE DO NOT WRITE ON THIS REFERENCE AS WE WILL REUSE THEM

Includes, in this order… **General**: String, ArrayList, Random,

Comparisons: Comparable, Comparator

JavaFX: Application, Stage, Scene, Node, Shape, Rectangle, Ellipse, Arc, Line, Polygon, Text, Group, Pane, Hbox/Vbox, BorderPane, Insets, Button, CheckBox, RadioButton, TextField, EventHandler, ActionEvent, MouseEvent, AnimationTimer

Multithreading: Thread, Runnable, Lock, ReentrantLock, Condition

Networking: ServerSocket, Socket, DataInputStream, DataOutputStream, InputStream, OutputStream

class String

Hierarchy: Object \ String

Constructors

`String()`
Creates the empty String

`String(String original)`
Creates a String that is the copy of the original parameter String

Methods

`char`	**`charAt(int index)`** Returns the character at the specified index
`int`	**`compareTo(String anotherString)`** Compares two Strings
`String`	**`concat(String str)`** Concatenates the specified string to the end of this string.
`boolean`	**`equals(String str)`** Compares this String to another string.
`boolean`	**`equalsIgnoreCase(String str)`** Compares this String to another string ignoring case considerations.
`int`	**`indexOf(int ch)`** Returns the index within this string of the first occurrence of the specified character.
`int`	**`indexOf(int ch, int fromIndex)`** Returns the index of the first occurrence of the specified character, starting the search at fromIndex.
`int`	**`indexOf(String str)`** Returns the index of the first occurrence of the specified substring.
`int`	**`indexOf(String str, int fromIndex)`** Returns the index of the first occurrence of the specified substring, starting at the specified index.
`int`	**`lastIndexOf(int ch)`** Returns the index within this string of the last occurrence of the specified character.
`int`	**`lastIndexOf(String str, int fromIndex)`** Returns the index within this string of the last occurrence of the specified substring, searching backward starting at the specified index.
`int`	**`length()`** Returns the length of this string.

String	**replace(String target, String replacement)** Replaces each substring that matches the target String with the specified literal replacement String.
String	**substring(int beginIndex)** Returns a string that is a substring of this string.
String	**substring(int beginIndex, int endIndex)** Returns a string that is a substring of this string.
String	**toLowerCase()** Returns a String with all characters converted to lower case.
String	**toUpperCase()** Returns a String with all characters converted to upper case.

class ArrayList<E>

Hierarchy: Object \ AbstractCollection<E> \ AbstractList<E> \ ArrayList<E>

Constructors

ArrayList<E>() Constructs an empty list with an initial capacity of ten.
ArrayList<E>(int initialCapacity) Constructs an empty list with the specified initial capacity.
ArrayList<E>(ArrayList<E> original) Creates a copy of the parameter ArrayList.

Methods

boolean	**add(E element)** Appends the parameter element to the end of the ArrayList
void	**add(int index, E element)** Inserts the specified element at the specified position in this list.
void	**clear()** Removes all of the elements from the list
E	**get(int index)** Returns the element at the specified position in this list.
int	**indexOf(Object o)** Returns the index of the first occurrence of the specified element in this list, or -1 if not in the list.
boolean	**isEmpty()** Returns true if the list contains no elements.
E	**remove(int index)** Removes the element at the specified position in this list.
boolean	**remove(Object o)** Removes the first occurrence of the specified element from this list, if it is present.
E	**set(int index)** Replaces the element at the specified position in this list with the specified element.
int	**size()** Returns the number of elements in the listj.
void	**sort(Comparator<? Super E>) c** Sorts this list according to the order induced by the specified Comparator.
String	**toString()** Returns a string representation. The string representation consists of the collection’s elements in order, enclosed in square brackets “[]”. Adjacent elements are separated by the characters “, “

class Random

Hierarchy: Object \ Random

Constructors

`Random()`
Creates a new random number generator.

Methods

`double`	**`nextDouble()`** Returns the next pseudorandom, uniformly distributed double value between 0.0 and 1.0 from this random number generator's sequence.
`int`	**`nextInt(int bound)`** Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.

interface Comparable<T>

Methods

`int`	**`compareTo(T other)`** Compares this object with the specified object for order. Negative implies this object should be ordered first. Positive that the parameter should be ordered first.

interface Comparator<T>

Methods

`int`	**`compare(T a, T b)`** Compares this paramete objects to determine order. Negative implies the *a* object should be ordered first. Positive that the *b* object should be ordered first.

abstract class Application

Hierarchy: Object \ Application

Methods

`void`	**`init()`** The application initialization method.
`static void`	**`launch(String... args)`** Launch a standalone application.
`abstract void`	**`start(Stage primaryStage)`** The main entry point for all JavaFX applications.
`void`	**`stop()`** This method is called when the application should stop.

class Stage

Hierarchy: Object \ Window \ Stage

Methods

`void`	**`close()`** Closes the Stage.
`void`	**`setResizable(boolean value)`** Sets the value of the property resizable.

void	**setScene(Scene value)** Specify the scene to be used on this stage.
void	**setTitle(String text)** Sets the value of the property title.
void	**show()** Attempts to show this Window by setting visibility to true

class Scene

Hierarchy: Object \ Scene

Constructors

Scene(Parent root)
Creates a Scene for a specific root Node.

Scene(Parent root, double width, double height)
Creates a Scene for a specific root Node with a specific size.

Methods

ObservableList<Node>	**getChildren()** Gets the list of children of this Parent.
void	**setFill(Paint value)** Sets the value of the property fill.

abstract class Node

Hierarchy: Object \ Node

Methods

double	**getLayoutX()** Gets the value of the property layoutX.
double	**getLayoutY()** Gets the value of the property layoutY.
boolean	**isVisible()** Gets the value of the property visible
void	**setLayoutX(double value)** Sets the value of the property layoutX.
void	**setLayoutY(double value)** Sets the value of the property layoutY.
boolean	**setMouseTransparent(boolean value)** Sets the mouseTransparent property
void	**setOnMouseClicked(EventHandler<MouseEvent>)** Sets the mouse clicked event handler
void	**setOnMouseDragged(EventHandler<MouseEvent>)** Sets the mouse dragged event handler
void	**setOnMouseMoved(EventHandler<MouseEvent>)** Sets the mouse moved event handler
void	**setOnMousePressed(EventHandler<MouseEvent>)** Sets the mouse pressed event handler
void	**setOnMouseReleased(EventHandler<MouseEvent>)** Sets the mouse released event handler
void	**setVisible()** Sets the value of the property visible

abstract class Shape

Hierarchy: Object \ Node \ Shape

Methods

`Paint`	**`getFill()`** Gets the value of the property fill.
`Paint`	**`getStroke()`** Gets the value of the property stroke.
`double`	**`getStrokeWidth()`** Gets the value of the property strokeWidth
`void`	**`setFill(Paint value)`** Sets the value of the property fill.
`void`	**`setStroke(Paint value)`** Sets the value of the property stroke.
`void`	**`setStrokeWidth(double value)`** Sets the value of the property strokeWidth.

class Rectangle

Hierarchy: Object \ Node \ Shape \ Rectangle

Constructors

`Rectangle(double width, double height)`
Creates a new instance of Rectangle with the given size.

`Rectangle(double x, double y, double width, double height)`
Creates a new instance of Rectangle with the given position and size.

Methods

	`getX, setX, getY, setY, getWidth, setWidth, getHeight, setHeight` All defined in the usual way

class Ellipse

Hierarchy: Object \ Node \ Shape \ Ellipse

Constructors

`Ellipse(double radiusX, double radiusY)`
Creates a new instance of Ellipse with the given size.

`Ellipse(double centerX, double centerY, double radiusX, double radiusY)`
Creates a new instance of Ellipse with the given position and size.

Methods

	`getCenterX, setCenterX, getCenterY, setCenterY, getRadiusX, setRadiusX, getRadiusY, setRadiusY` All defined in the usual way

class Arc

Hierarchy: Object \ Node \ Shape \ Arc

Constructors

`Arc(dble centerX, dble centerY, dble radiusX, dble radiusY, dble startAngle, dble length)`

Creates a new instance of Arc with the given values. Angle is defined in degrees. **The length is the angle length.** Dble is short for double

Methods

`getCenterX, setCenterX, getCenterY, setCenterY, getRadiusX, setRadiusX, getRadiusY, setRadiusY, getStartAngle, setStartAngle, getLength, setLength`

All defined in the usual way. **The length is the angle length.**

class Line

Hierarchy: Object \ Node \ Shape \ Line

Constructors

`Line(double startX, double startY, double endX, double endY)`

Creates a new instance of line

Methods

`getStartX, setStartX, getStartY, setStartY, getEndX, setEndX, getEndY, setEndY`

All defined in the usual way.

class Polygon

Hierarchy: Object \ Node \ Shape \ Polygon

Constructors

`Polygon(double... Points)`

Creates a new instance of Polygon.

Methods

`ObservableList<Double> getPoints()`

Gets the coordinates of the Polygon vertices.

class Text

Hierarchy: Object \ Node \ Shape \ Text

Constructors

`Text(String text)`

Creates a new instance of Text.

`Text(double x, double y, String text)`

Creates a new instance of Text.

Methods

`getX, getY, setX, setY, getText, setText`

All defined in the usual way.

class Group

Hierarchy: Object \ Node \ Parent \ Group

Constructors

`Group()`
Creates a new Group

`Group(Node... children)`
Creates a new Group consisting of children

Methods

`ObservableList<Node> getChildren()`
Gets the list of children of this Parent.

class Pane

Hierarchy: Object \ Node \ Parent \ Region \ Pane

Constructors

`Pane()`
Creates a new Pane layout

`Pane(Nodes... children)`
Creates a new Pane layout

Methods

`ObservableList<Node> getChildren()`
Gets the list of children of this Parent.

`void` **`setPrefHeight(double value) / setPrefWidth(double value)`**
Sets the value of the property prefHeight/prefWidth.

class HBox/VBox

Hierarchy: Object \ Node \ Parent \ Region \ Hbox/Vbox

Constructors

`HBox(double spacing) / Vbox(double spacing)`
Creates a new Hbox/Vbox layout

`HBox(double spacing, Nodes... children) / Vbox(double spacing, Nodes... Children)`
Creates a new Hbox/Vbox layout

Methods

`ObservableList<Node> getChildren()`
Gets the list of children of this Parent.

`void` **`setAlignment(Pos value)`**
Sets the value of the property alignment.

`static void setMargin(Node child, Insets value)`
Sets the margin for the child when contained by an hbox/vbox.

`void` **`setPadding(Insets value)`**
Sets the value of the property padding.

class BorderPane

Hierarchy: Object \ Node \ Parent \ Region \ BorderPane

Constructors

BorderPane()
Creates a new BorderPane layout

Methods

ObservableList<Node> getChildren()	
	Gets the list of children of this Parent.
static void setAlignment(Node child, Pos value)	
	Sets the alignment for the child when contained by a border pane
static void setMargin(Node child, Insets value)	
	Sets the margin for the child when contained by an hbox/vbox.
void	**setCenter, setLeft, setRight, setBottom, setTop (Node value)** Sets the element for the specified area of the BorderPane
void	**setPadding(Insets value)** Sets the value of the property padding.

class Insets

Hierarchy: Object \ Insets

Constructors

Insets(double topRightBottomLeft)
Constructs a new Insets instance with same value for all four offsets.

Insets(double top, double right, double bottom, double left)
Constructs a new Insets instance with four different offsets.

class Button

Hierarchy: Object \ Node \ Parent \ Region \ Control \ Labeled \ ButtonBase \ Button

Constructors

Button()
Creates a new Button

Button(String text)
Creates a new Button containing the given text

Methods

void	**setOnAction(EventHandler<ActionEvent>)** Sets the value of the onAction property.
void	**setText(String text)** Sets the value of the text property.

class CheckBox

Hierarchy: Object \ Node \ Parent \ Region \ Control \ Labeled \ ButtonBase \ Checkbox

Constructors

`CheckBox()`
Creates a new CheckBox

`Button(String text)`
Creates a new CheckBox containing the given text

Methods

`boolean`	**`isSelected()`** Gets the value of the property selected.
`void`	**`setOnAction(EventHandler<ActionEvent>)`** Sets the value of the onAction property.
`void`	**`setSelected(boolean value)`** Sets the value of the property selected.
`void`	**`setText(String text)`** Sets the value of the text property.

class RadioButton

Hierarchy: Object \ Node \ Parent \ Region \ Control \ Labeled \ ButtonBase \ ToggleButton \ RadioButton

Constructors

`RadioButton()`
Creates a new RadioButton

`Button(String text)`
Creates a new RadioButton containing the given text

Methods

`boolean`	**`isSelected()`** Gets the value of the property selected.
`void`	**`setSelected(boolean value)`** Sets the value of the property selected.
`void`	**`setOnAction(EventHandler<ActionEvent>)`** Sets the value of the onAction property.
`void`	**`setText(String text)`** Sets the value of the text property.
`void`	**`setToggleGroup(ToggleGroup value)`** Sets the value of the property toggleGroup.

class ToggleGroup

Hierarchy: Object \ ToggleGroup

Constructors

`ToggleGroup()`
Creates a new ToggleGroup

Methods

`Toggle`	**`getSelectedToggle()`** Gets the selected Toggle.
`void`	**`selectToggle(Toggle value)`** Sets the selected Toggle.

class TextField

Hierarchy: Object \ Node \ Parent \ Region \ Control \ TextInputControl \ TextField

Constructors

`TextField()`
Creates a new TextField

`TextField(String text)`
Creates a new TextField containing the given text.

Methods

`String`	**`getText()`** Gets the value of the text property.
`void`	**`setText(String text)`** Sets the value of the text property.

interface EventHandler<T extends Event>

Methods

`void`	**`handle(T event)`** Invoked when a specific event of the type for which this handler is registered happens.

class ActionEvent

Hierarchy: Object \ EventObject \ Event \ ActionEvent

Methods

`void`	**`consume()`** Marks this Event as consumed. This stops its further propagation.
`Object`	**`getSource()`** The object on which the Event initially occurred.

class MouseEvent

Hierarchy: Object \ EventObject \ Event \ InputEvent \ MouseEvent

Methods

`void`	**`consume()`** Marks this Event as consumed. This stops its further propagation.
`Object`	**`getSource()`** The object on which the Event initially occurred.
`double`	**`getX(),  getY()`** Get the position of the event relative to the origin of the MouseEvent's source.

abstract class AnimationTimer

Hierarchy: Object \ AnimationTimer

Methods

`void`	**`handle(long now)`** This method needs to be overridden by extending classes.
`void`	**`start()`** Starts the AnimationTimers.
`void`	**`stop():`** Stops the AnimationTimers.

class Thread

Hierarchy: Object \ Thread

Constructors

`Thread(Runnable target)`
Allocates a new Thread object.

Methods

`static Thread`	**`currentThread()`** Returns a reference to the currently executing thread object.
`void`	**`interrupt()`** Tests whether the current thread has been interrupted.
`static boolean`	**`interrupted()`** Tests whether the current thread has been interrupted.
`boolean`	**`isAlive()`** Tests if this thread is alive.
`void`	**`join()`** Waits for this thread to die.
`static void`	**`sleep(long millis)`** Causes the currently executing thread to sleep for the specified number of milliseconds.
`void`	**`start()`** Causes this thread to begin execution; the JVM calls the run method of this thread.

interface Runnable

Methods

`void`	**`run()`** When an object implementing interface Runnable is used to create a thread, starting the thread causes the object's run method to be called in that thread.

interface Lock

Methods

`void`	**`lock()`** Acquires the lock
`Condition`	**`newCondition()`** Returns a new Condition instance that is bound to this Lock instance.
`void`	**`unlock()`** Releases the lock.

class ReentrantLock

Hierarchy: Object \ ReentrantLock

Constructors

`ReentrantLock()`
Creates a new ReentrantLock

interface Condition

Methods

void	**await()** Causes the current thread to wait until it is signalled or interrupted.
void	**signalAll()** Wakes up all waiting threads.

class ServerSocket

Hierarchy: Object \ ServerSocket

Constructors

ServerSocket(int port)
Creates a server socket, bound to the specified port.

Methods

Socket	**accept()** Listens for a connection to be made to this socket and accepts it.
void	**close()** Closes the socket.

class Socket

Hierarchy: Object \ Socket

Constructors

Socket(String host, int port)
Creates a stream socket and connects it to the specified port number on the named host

Methods

void	**close()** Closes the socket.
InputStream	**getInputStream()** Returns an input stream for this socket.
InputStream	**getOutputStream()** Returns an output stream for this socket.

class DataInputStream

Hierarchy: Object \ InputStream \ DataInputStream

Constructors

DataInputStream(InputStream in)
Creates a DataInputStream that uses the specified underlying InputStream.

Methods

int	**read(byte[] b)** Reads bytes from the contained input stream and stores them into the array b. Returns number of the bytes read.
boolean	**readBoolean()** Reads one byte and returns true if that byte is nonzero, false if that byte is zero.
byte	**readByte()** Reads one input byte and returns a byte value.
char	**readChar()** Reads two input bytes and returns a char value.

`double`	**`readDouble()`** Reads eight input bytes and returns a double value.
`float`	**`readFloat()`** Reads four input bytes and returns a float value.
`int`	**`readInt()`** Reads four input bytes and returns an int value.
`long`	**`readLong()`** Reads eight input bytes and returns a long value.
`short`	**`readShoart()`** Reads two input bytes and returns a short value.
`String`	**`readUTF()`** Reads in a string that has been encoded using a modified UTF-8 format.

class DataOutputStream

Hierarchy: Object \ OutputStream \ DataOutputStream

Constructors

`DataOututStream(OutputStream out)`

Creates a DataOutputStream that uses the specified underlying OutputStream.

Methods

`void`	**`flush()`** Flushes this data to the output stream
`void`	**`WriteBoolean(boolean v)`** Writes 1 byte to the underlying output stream as a boolean.
`void`	**`writeByte(byte v)`** Writes 1 byte to the underlying output stream as a byte.
`void`	**`writeChar(char v)`** Writes two bytes to the underlying output stream as a character.
`void`	**`writeDouble(double v)`** Writes 8 bytes to the underlying output stream as a double.
`void`	**`writeFloat(float v)`** Writes 4 bytes to the underlying output stream as a float.
`void`	**`writeInt(int v)`** Writes 4 bytes to the underlying output stream as an integer.
`void`	**`writeLong(long v)`** Writes 8 bytes to the underlying output stream as a long integer.
`void`	**`writeShort(int v)`** Writes 2 bytes to the underlying output stream as a short integer.
`void`	**`writeUTF(String str)`** Writes a string to the underlying output stream using modified UTF-8 encoding in a machine-independent manner.

class InputStream

Hierarchy: Object \ InputStream

Methods

`int`	`available()` Returns number of bytes that can be read from this input stream.
`void`	`close()` Closes this input stream and releases any system resources associated with the stream.

class OutputStream

Hierarchy: Object \ OutputStream

Methods

`void`	`close()` Closes this output stream and releases any system resources associated with this stream.
`void`	`flush()` Flushes this output stream and forces any buffered output bytes to be written out.