The Jack OS API

The Jack language comes with a collection of eight built-in classes that extend the language's capabilities. This standard library, which can also be viewed as a basic operating system, includes the following classes, all implemented in Jack:

Math

This class enables various mathematical operations.

function int **`abs`**(int x): returns the absolute value of x.

function int **`multiply`**(int x, int y): returns the product of x and y.

function int **`divide`**(int x, int y): returns the integer part of x/y.

function int **`min`**(int x, int y): returns the minimum of x and y.

function int **`max`**(int x, int y): returns the maximum of x and y.

function int **`sqrt`**(int x): returns the integer part of the square root of x.

String

This class implements the `String` data type and various string-related operations.

constructor String **`new`**(int maxLength): constructs a new empty string (of length zero) that can contain at most maxLength characters.

method void **`dispose`**(): disposes this string.

method int **`length`**(): returns the length of this string.

method char **`charAt`**(int j): returns the character at location j of this string.

method void **`setCharAt`**(int j, char c): sets the j-th element of this string to c.

method String **`appendChar`**(char c): appends c to this string and returns this string.

method void **`eraseLastChar`**(): erases the last character from this string.

method int **`intValue`**(): returns the integer value of this string (or the string prefix until a non-digit character is detected).

method void **`setInt`**(int j): sets this string to hold a representation of j.

function char **`backSpace`**(): returns the backspace character.

function char **`doubleQuote`**(): returns the double quote (") character.

function char **`newLine`**(): returns the newline character.

Array

This class enables the construction and disposal of arrays.

function Array **`new`**(int size): constructs a new array of the given size.

method void **`dispose`**(): disposes this array.

Output

This class allows writing text on the screen.

function void **`moveCursor`**(int i, int j): moves the cursor to the j-th column of the i-th row, and erases the character displayed there.

function void **`printChar`**(char c): prints c at the cursor location and advances the cursor one column forward.

function void **`printString`**(String s): prints s starting at the cursor location and advances the cursor appropriately.

function void **`printInt`**(int i): prints i starting at the cursor location and advances the cursor appropriately.

function void **`println`**(): advances the cursor to the beginning of the next line.

function void **`backSpace`**(): moves the cursor one column back.

Screen

This class allows drawing graphics on the screen. Column indices start at 0 and are left-to-right. Row indices start at 0 and are top-to-bottom. The screen size is hardware-dependant (in the Hack platform: 256 rows by 512 columns).

function void **`clearScreen`**(): erases the entire screen.

function void **`setColor`**(boolean b): sets a color (white=false, black=true) to be used for all further `drawXXX` commands.

function void **`drawPixel`**(int x, int y): draws the (x,y) pixel.

function void **`drawLine`**(int x1, int y1, int x2, int y2): draws a line from pixel (x1,y1) to pixel (x2,y2).

function void **`drawRectangle`**(int x1, int y1, int x2, int y2): draws a filled rectangle whose top left corner is (x1, y1) and bottom right corner is (x2,y2).

function void **`drawCircle`**(int x, int y, int r): draws a filled circle of radius r<=181 around (x,y).

Keyboard

This class allows reading inputs from a standard keyboard.

function char **`keyPressed`**(): returns the character of the currently pressed key on the keyboard; if no key is currently pressed, returns 0.

function char **`readChar`**(): waits until a key is pressed on the keyboard and released, then echoes the key to the screen and returns the character of the pressed key.

function String **`readLine`**(String message): prints the message on the screen, reads the line (text until a newline character is detected) from the keyboard, echoes the line to the screen, and returns its value. This function also handles user backspaces.

function int **`readInt`**(String message): prints the message on the screen, reads the line (text until a newline character is detected) from the keyboard, echoes the line to the screen, and returns its

integer value (until the first non-digit character in the line is detected). This function also handles user backspaces.

Memory

This class allows direct access to the main memory of the host platform.

function int **`peek`**(int address): returns the value of the main memory at this address.

function void **`poke`**(int address, int value): sets the contents of the main memory at this address to value.

function Array **`alloc`**(int size): finds and allocates from the heap a memory block of the specified size and returns a reference to its base address.

function void **`deAlloc`**(Array o): De-allocates the given object and frees its memory space.

Sys

This class supports some execution-related services.

function void **`halt`**(): halts the program execution.

function void **`error`**(int errorCode): prints the error code on the screen and halts.

function void **`wait`**(int duration): waits approximately *duration* milliseconds and returns.