Appendix 6: The Jack OS API

The Jack language is supported by a library of eight classes that extend the language's capabilities. This standard library can be viewed as a basic operating system.

Math

A library of commonly needed mathematical functions.

`function int multiply(int x, int y)`: Returns the product of `x` and `y`. When a Jack compiler detects the multiplication operator '`*`' in the program's code, it handles it by invoking this method. Thus the Jack expressions `x*y` and `Math.multiply(x,y)` return the same value.

`function int divide(int x, int y)`: Returns the integer part of `x/y`. When a Jack compiler detects the division operator '`/`' in the program's code, it handles it by invoking this method. Thus the Jack expressions `x/y` and `Math.divide(x,y)` return the same value.

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

`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

The `String` class represents strings of `char` values, and supports commonly needed string processing services.

`constructor String new(int maxLength)`: Constructs a new empty string with a maximum length of `maxLength` and initial length of 0.

`method void dispose()`: Disposes this string.

`method int length()`: Returns the number of characters in this string.

`method char charAt(int j)`: Returns the character at the `j`-th location of this string.

`method void setCharAt(int j, char c)`: Sets the character at the `j`-th location of this string to `c`.

`method String appendChar(char c)`: Appends `c` to this string's end, and returns this string.

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

`method int intValue()`: Returns the integer value of this string, until a non-digit char is detected.

`method void setInt(int val)`: Sets this string to hold a representation of the given value.

`function char backSpace()`: Returns the backspace character.

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

`function char newLine()`: Returns the newline character.

Array

In the Jack language, arrays are instances of the OS class `Array`. Once declared, the array entries can be accessed using the syntax `arr[i]`. Jack arrays are not typed: Each array entry can hold a primitive data type or an object type, and different entries can have different types.

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

`method void dispose()`: Disposes this array.

Output

A library of functions for displaying characters. The class assumes a character-oriented screen consisting of 23 rows (indexed 0...22, top to bottom) of 64 characters each (indexed 0...63, left to right). The top left character location on the screen is indexed (0,0). Each character is displayed by rendering on the screen a rectangular image, 11 pixels high and 8 pixels wide (which include margins for inter-character and inter-row spacing). The bitmap images of all the characters can be found by looking up the given code of the `Output` class. A visible cursor, implemented as a small filled square, indicates where the next character will be displayed.

`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)`: Displays the character at the cursor location, and advances the cursor one column forward.

`function void printString(String s)`: Displays the string starting at the cursor location, and advances the cursor appropriately.

`function void printInt(int i)`: Displays the integer 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

A library of functions for displaying graphical shapes on the screen. The Hack physical screen consists of 256 rows (indexed 0..255, top to bottom) of 512 pixels each (indexed 0..511, left to right). The top left pixel on the screen is indexed (0,0).

`function void clearScreen()`: Erases the entire screen.

`function void setColor(boolean b)`: Sets the current color. This color will be used in all the subsequent draw*Xxx* function calls. Black is represented by `true`, white by `false`.

`function void drawPixel(int x, int y)`: Draws the (`x`,`y`) pixel, using the current color.

`function void drawLine(int x1, int y1, int x2, int y2)`: Draws a line from pixel (`x1`,`y1`) to pixel (`x2`,`y2`), using the current color.

`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`), using the current color.

`function void drawCircle(int x, int y, int r)`: Draws a filled circle of radius `r`$\leq$181 around (`x`,`y`), using the current color.

Keyboard

This class supports 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. Recognizes all the values in the Hack character set (see appendix 5). These include the characters newline (128, return value of `String.newline()`), backspace (129, return value of `String.backspace()`), left-arrow (130), up-arrow (131), right-arrow (132), down-arrow (133), home (134), end (135), page-up (136), page-down (137), insert (138), delete (139), `ESC` (140), and `F1-F12` (141-152).

`function char readChar()`: Waits until a key is pressed on the keyboard and released, then displays the corresponding character on the screen, and returns the character.

`function String readLine(String message)`: Displays the `message` on the screen, reads from the keyboard the entered string of characters until a newline character is detected, echoes (prints) the string on the screen, and returns the string. Also handles user backspaces.

`function int readInt(String message)`: Displays the `message` on the screen, reads from the keyboard the entered string of characters until a newline character is detected, displays the string on the screen, and returns its integer value, until the first non-digit character in the entered string is detected. Also handles user backspaces.

Memory

This library provides two different services: direct access to the computer's main memory (RAM), and allocation and recycling of memory blocks. The Hack RAM consists of 32,768 words, each holding a 16-bit binary number.

`function void peek(int address)`: Returns the value of `RAM[address]`.

`function void poke(int address, int value)`: Sets `RAM[address]` to the given `value`.

`function Array alloc(int size)`: Finds an available RAM block of the given `size`, and returns a reference to its base address.

`function void deAlloc(Array o)`: De-allocates the given object, which is cast as an array. In other words, makes the RAM block that starts in this address available for future memory allocations.

Sys

A library that supports various program execution services.

`function void halt()`: Halts the program execution.

`function void error(int errorCode)`: Displays the error code, using the format "`ERR<errorCode>`", and halts the program's execution.

`function void wait(int duration)`: Waits approximately `duration` milliseconds, and returns.