6.170 P-set #1

Current Version 1.1

Owner: aphyer

Citations: A lot of general ideas were stolen from the jQuery lab. No other collaboration.

Design Challenges:

* Choosing a representation for the gameboard:
  + I used an array-of-arrays to represent the board, rather than creating a custom class, because I felt an array-of-arrays had all the functionality I wanted.
  + Each element was the number “0” or “1”. I chose to store these, rather than storing *true* and *false*, to let me count the # of adjacent live cells as simply as possible (just by adding all their values together).
  + I used a 19x19 board; one large enough to store interesting patterns, with a clearly-defined center square to center symmetrical shapes on.
  + I treated any square outside these boundaries as being dead, and did not track any patterns that spread beyond those boundaries. I did this because:
    - Treating squares outside the boundaries as alive results in squares on the boundaries randomly coming to life, which is bad.
    - Tracking patterns that spread beyond the boundaries would require me to continually increase the size of the tracking array. In particular, a single glider (a shape that moves in one direction forever) would result in a perpetually increasing board size. I wanted to avoid that.
    - A 19x19 board is large enough to demonstrate most interesting shapes.
* Minimizing graphics dependence:
  + Everything about the board other than printing it was encapsulated in routines with no dependence on graphics.
  + Printing the board to the screen was done by the subroutine *printBoard(board)*, which took as its input the nested-array structure of a board and printed it to the screen.