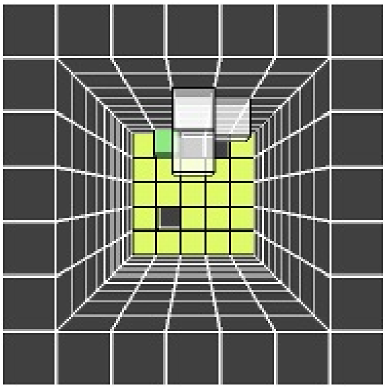
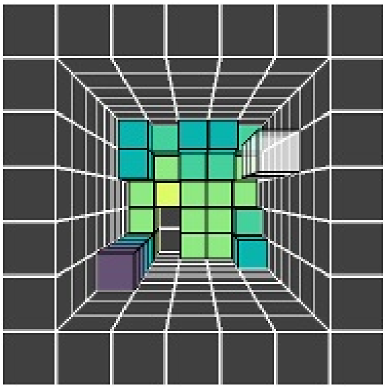
Lisp in Summer Projects Submission

Submission Date	2013-10-20 17:52:01
Full Name	E. Cayenne Geis
Country	USA
Project Name	Blocks in Space
Type of software	gui app
General category	game
LISP dialect	Clojure
GitHub URL	https://github.com/cayennes/blocks-in-space/tree/lisp
Did you start this project?	Yes, all the code is written by me
Project Description	I want to describe my project in this form.
Purpose	This is a tetris-like game in three dimensions instead of two. Also instead of increasing falling speed, the blocks increase in complexity as the game progresses.
Function	This is a tetris-like game in three dimensions instead of two. Also instead of increasing falling speed, the blocks increase in complexity as the game progresses.
Motivation	xblockout is fun but it only runs on linux. Writing a similar game in Clojure provides the the opportunity not just to run it on more platforms, but to make a better game. The fun challenge of the game is working with the block shapes, not in moving them quickly. So in this game that's what progresses instead of the speed.
Audience	This is a game for anyone who thinks that trying to manipulate complex shapes in three dimensions is a fun game; currently also only people that are comfortable installing leiningen and running from source.
Methodology	The graphics are powered by the opengl capabilities of the quil library. Actual game state is stored in atoms; everything else is immutable.

	Falling blocks are represented as a center point and a shape that is a set of coordinates in relation to that center point. This makes them very natural to manipulate. There is a function to turn that into a set of coordinates, which is again very natural to display or to work with in the context of the rest of the game state. The sequence of possible shapes to add as the game progresses is programmatically generated in a seque - so at least the next shape is always generated in parallel to the game. The method of generating additional shapes works by creating a large list of possibilities that includes many duplicates and then (mostly*) filtering out the duplicates. It is not a problem to have some inefficiency in this process since once the game starts it is done in parallel with the gameplay which is never otherwise hardware-limited. An orientation is chosen for each shape in a way that maximizes the amount of it that is clear on first sight - as flat as possible, with the side with more space visible. *I didn't do enough normalizing in the function that gives orientations scores and so this filtering process is imperfect.
Conclusion	I succeeded in making a game that I think is fun to play. I was very happy with how easy Clojure made it to generate an arbitrarily long list of different shapes of blocks, and to do that in parallel with gameplay so that it doesn't cause a delay in the experience. I plan to do at least a little polishing whenever I happen to get back to this. I definitely want to fix the block-uniqueness issue. After that I want to put more effort in figuring out if there's a workaround for the opengl-related issues that I experienced when trying to make a stand-alone jar, as I think that's what it would take to have something that could reach a wider audience. If I can fix that, then I might tackle the other things on the missing-features list in the README.
Build Instructions	N/A
Test Instructions	This is not a significant test suite, but what there is can be run with $ lein test It fails (see bug description below.)
Execution Instructions	$ lein run See README.md for play instructions
Describe any bugs or caveats	In a few cases, when an additional shape should enter the list of those possible, a previous shape in a new starting orientation becomes possible instead.

Screen shots	 3d-block.png  colored-heights.png
Official	I have read rules and have abided by them. I am 18 years of age or older. I am not living in Brazil, Quebec, Saudi Arabia, Cuba, Iran, Myanmar (Burma), North Korea, Sudan, or Syria.