School of Electronics and Computer Science

Faculty of Physical and Applied Sciences

University of Southampton

Date: January 9, 2012

COMP3001 Group Scripting Coursework - Team B

Team Members:

Dexter Lowe (dl10g09@ecs.soton.ac.uk) James Skinner (dl10g09@ecs.soton.ac.uk) Adam Thomas (dl10g09@ecs.soton.ac.uk) Bahman Asadi (dl10g09@ecs.soton.ac.uk) Peter West (dl10g09@ecs.soton.ac.uk) Alex Parker (dl10g09@ecs.soton.ac.uk) Tom Smith (dl10g09@ecs.soton.ac.uk)

http://3001Blokus.appspot.com

Contents

1 Description of Prototype Functionality

1.1 Blokus

Blokus is an abstract strategic board game for 4 players. Also a 2 player variant exists where each player controls 2 colours. Blokus is played on a 20x20 board using 21 unique polyominoes. On their turn users select a piece from their piece-tray on the left and move it to where they want to place it on the board. Client-side validation is used to highlight the area the piece would cover to show whether a position is valid or invalid as appropriate. If the user wishes to rotate or flip the a piece they either hover over it in their piece-tray and use the control halo that appears or once picked up they can use left and right to rotate or 'h' and 'v' to flip horizontally and vertically respectively. Once the user has decided where to place it they click once more to place the piece on the board.

No piece may overlap with any other piece, the first piece placed for each player must use one of the 4 corner squares. Finally all subsequent pieces must always share a corner with an existing piece of the same colour but must never share a flat side with a piece of the same colour.

If a player cannot place a piece then their turn is skipped. Play continues until no player can place any piece. Once the game is over the score is calculated by subtracting 1 point for each square on each remaining unplaced piece. 15 points are awarded if all pieces were placed, plus an extra 5 if the last piece placed was the monomino. The user's statistics are updated in their profile.

1.2 Authentication

When a user first navigates to the Blokus app they are given a guest user account so that they can play a quick game immediately. If they wish to store their statistics they must register or login. We have chosen to provide Facebook and Google authentication however as some people do not like to use these open authentication systems we have also provided the facility to have a Blokus account without needing to use social authentication.

1.3 Game Modes

Once logged in or just as a guest the user must then decide whether they wish to play a 2 or 4 player game. They click the appropriate button or the quick game button if they don't mind, and once there are enough people waiting a game is created and joined. The user can also choose to host a private game with their friends by selecting private game and defining if it is a 2 or 4 player game. Then the user is given a unique id which they can tell their friends which will put them all in the same game once the 2 or 4 player limit is reached.

2 List of Tools and Techniques Used

2.1 Tools

- Various Text Editors with syntax highlighting for general development. No language specific IDEs were used to reduce platform incompatibility issues and reduce unnecessary files.
  - Sublime Text 2 (All)
  - Vim (Unix Based Environments)
  - Notepad++ (Windows Environments)
- Google Chrome Developer Tools were used for Javascript debugging.
- Python 2.7 was used as it is the version used on the Google App Engine.
- The Google App Engine SDK 1.6.1 was used as the base platform.
- The Google App Engine was used as the hosting service.
- Mercurial (hosted on BitBucket) was used for collaborative source control. Mercurial was selected over SVN due to its superior branching and the local repository.
  - Mercurial (Command Line) (All)
  - TortoiseHG (Windows Environments)
- The Django framework was used both to simplify development and encapsulate some GAE SDK commands such as deploying to the app engine.
- Facebook was used for group contact and organisation.
- Google Docs were used for collaborative authoring and document sharing.
- Mozilla Firefox, Google Chrome, Chromium, IE9 and Safari were used for testing.
- JSON was used to aid in the simplification of client/server communication and used to pass objects around in Javascript.
- AJAX was used to create dynamic forms using asynchronous Javascript.
- cron was used to garbage collect completed or abandoned games and inactive temporary users from the database.

2.2 Techniques

We used a RAD (Rapid Application Development) approach to development having regular meetings and aiming to have a prototype with more functionality at each one. We also used Facebook and Google Docs to ensure that everyone knew the current state and design of the system.

Temporary data was created on the client side and a DEBUG page to emulate many server functions so that front-end development could continue even if the back-end was not working. In this way the various modules of the system were kept separate as far as possible.

We used BitBucket issue tracking to keep track of errors so that issues were not forgotten and everyone could see what needed to be done. This helped in ensuring development time was not wasted and task distribution.

3 Relevant Statistics