Software Developer

Tech Task - Ores

Objective

- To evaluate the applicant’s technical skills via the produced project
    - Architecture design choices
    - Focus on readability, maintainability and extensibility of the solution
    - Applying good development practices
- To evaluate the applicant’s ability to follow a list of requirements during implementation
- To evaluate the applicant’s orientation towards product and user
    - Attention to detail and user experience
    - Focus to the fun factor of the game

Game Requirements

- Boxes stack vertically in columns and have different colors
- Periodically, a new column of (random) boxes pushes all the columns sideways
- When a box is clicked, all adjacently connected boxes of the same color disappear
    - Boxes are considered adjacent if they are vertically or horizontally next to each other
    - All adjacent boxes should disappear, not only the immediately adjacent
- If there is a vertical gap between two boxes, top boxes should collapse down filling the empty spaces
- If there is a horizontal gap between two columns, boxes will collapse towards the spawn zone
- Opposite to the spawn zone there is an end zone
    - The game ends when a box reaches the end zone
- Check Ores for details and inspiration - http://www.miniclip.com/games/ores/
    - You may be creative in terms of start conditions, difficulty, timings, visuals, ...

Rules

- You must develop the game in C++ or Objective-C
- You must use libSDL for rendering (http://www.libsdl.org/)
    - You may use additional sdl libraries to handle image and text loading and rendering or audio playback
- You may use any free images or sounds to improve the quality of your game
- You **may not** use any external game engine (cocosx, unity, udk, etc...)
- Delivery consists of an archive file (“**delivery.zip**”) containing:
    - A **prebuilt version of the game** (preferably targeting OSX, but Windows is ok)
        - It should not be necessary to install any external players or libraries to run the application
    - **All of the source code and asset files** necessary to compile and run the game
        - Do not deliver any temporary or other files that are not necessary for the above purposes
    - A **readme.txt** file with any information you would like to share with the technical reviewers

Evaluation

- Both the prebuilt and the code delivered will be evaluated
    - The readme file will also be taken into consideration during the review
- Evaluation of the prebuilt delivered:
    - How the delivery matches against the specification (validate how well requirements are met)
    - Feeling when playing the game (FPS drops, noticeable bugs and fun-factor)
- Evaluation of the code delivered:
    - Readability, maintainability and extensibility of the project
        - Focus on the overall structure and organization of the project
        - Proper execution of good programming practices and code quality
    - Memory considerations