# Block Shift

**Part 1** Due in class on **11/18**
**Part 2** Due before class on **11/25**
**Part 3** Due before class on **12/4**
**Final** Due before class on **12/11**

For this project, you will create a single player puzzle game. This project is intended to help you increase your skills in the following areas:

- Resourcefulness and Persistence
  - Searching for answers online, bookmarking commonly used webpages
  - Asking for help from Ira, the TA, or other students when you're stuck
- Logical Thinking
  - Carefully thinking through step-by-step logic for the program
- Using Unity
  - Creating new projects, files, and assets
  - Attaching scripts to objects
- Using GitHub
  - Storing and sharing your program as it changes over time
- Publishing a game to your own website
  - How to create a basic website and the tools required
  - FTP, basic text editor, Unity
- Organization
  - Well-organized and well-named files, folders, assets, and variables
  - Nicely formatted code with matching {} and indentation
  - Comments to help understand the code
- Types of variables
  - int, float, bool
- Data structure: Arrays, including 2D arrays
  - What they are, how to create them and modify them
- Mathematical expressions
  - Comfortable with adding, subtracting, multiplying and dividing
- Conditional expressions
  - if(), else if(), else()
- Iterative statements
  - for() loops
- Animations and Coroutines
- Tuning variables and JSON files

Wow, that's a lot! It's amazing what you'll be able to learn in just a few weeks of effort!

**What To Hand In**

For each part of this project, **deliver your work as follows**:

a) Within Unity, make a WebGL build (File -> Build Settings)
b) Upload the .html file, Build folder, and TemplateData folder to your website.
c) Upload your project files (especially .cs) to GitHub.
d) Find your tab on the spreadsheet below and add your website and GitHub links for the appropriate part of the project:
https://docs.google.com/spreadsheets/d/1q7FT29Xah-0eGqp2oTqzsDv3wMB8kwn8l3peHw38CpU
a) Post in the #cs181_handin channel that you completed each part.

**Part 1**
Read the game design document at the end of this assignment. By the deadline, prepare a short (~4 minute) presentation to give in class. Update your slides here: https://docs.google.com/presentation/d/1j25uZpjA4rvSv_hNXgEVjMIaxO4JdqyR58dPp6g5nN0

The presentation should include:

1) A description of what your main Update() loop will do.

2) A list of each method you intend to implement with arguments, return values, and a one sentence description of what it does (i.e. the method header). For example:
   a.
```
// this method will check if the player should score
void CheckScore()
```
   b.
```
// Starting from the given position in the grid, this
// method will return how many cubes in a row match
// the given cube's color
int FindColorMatchInRow (int x, int y)
```

You will be evaluated on the quality of your content as well as the polish of your presentation. You may collaborate with any number of people for this part of the project, as long as you credit them appropriately. If you want to present in combination with someone else, that's fine too, as long as you notify Ira in advance and all people have a full understanding of what's being presented.

You may work with a partner, but ensure you can write all the code individually. If you finish this section have extra time, start programming your game!

**Part 2**
By the deadline, make significant progress toward the final game, as described in the game design document below. Here are some features that should be completed, and parenthetical features should be done soon:

- Grid of cubes
- Grid is populated with random colors (without 3 in a row)
- Pushers exist (and can move around)
- (Action phase exists and players can destroy cubes)
- (Pushers push in order, but blocks teleport instead of animate)
- Matches are recognized, scored, and removed from grid
- New cubes appear in grid instantly (or maybe animate nicely)

**Part 3**
By the deadline, the game should be nearly complete, as described in the game design document below. In addition to all the features in Part 2, here are additional features that should be done:

- Start Screen with Play button
- Game Summary screen with buttons and info
- UI on screen giving player feedback
- Loot and Combos
- Animations

**Final**
By the deadline, the game should be complete and bug free, as described in the game design document below. Spend the final week implementing any missing features and debugging problems. In a game this complicated, there may be strange bugs. Ensure that your game works properly in all cases and there are no bugs. Include a link to your game and GitHub repository in the spreadsheet on your tab, as usual.

**A Note about the Game Design**

The description below is something that you might get from a game designer, if you were working at a game company. Once you read it and start thinking about it, if you have questions about the design, you may feel free to ask questions to clarify design intent by messaging Ira.

The game below is complicated! If it's too hard for you to implement in ~40 hours of work, please employ some of the simplifications listed at the end of the document. It's much better to have a simplified version of the game working 100% than the full version of the game that doesn't actually work!

# Block Shift

Designer: Ira Fay

Revision History

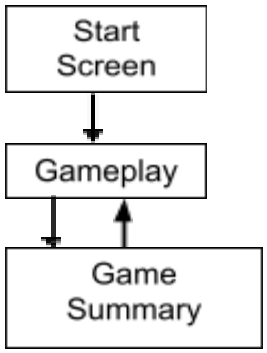
| Version | Date | Changes |
|---|---|---|
| 1.0 | 11/16/13 | Initial Version |

*Block Shift* is a single-player puzzle game (match 3 variant), playable in a web browser or on smartphones. The player uses a mouse and keyboard to control the game, or simply touches the screen on a smartphone.

We will be making this game as a tech demo, so art isn't a concern. If we have time later, we will add art to make it prettier.

The game has three states:

1) Start screen
2) Gameplay
3) Game Summary




**Start Screen**
When players first start the game, they should see a very large [Start] button. When they click it, the game starts.

**Gameplay**
The game is played on an 8 x 5 grid of large colored cubes, and takes place over 15 turns (tunable). Each turn has the following three phases:

1) Planning
2) Action
3) Resolution

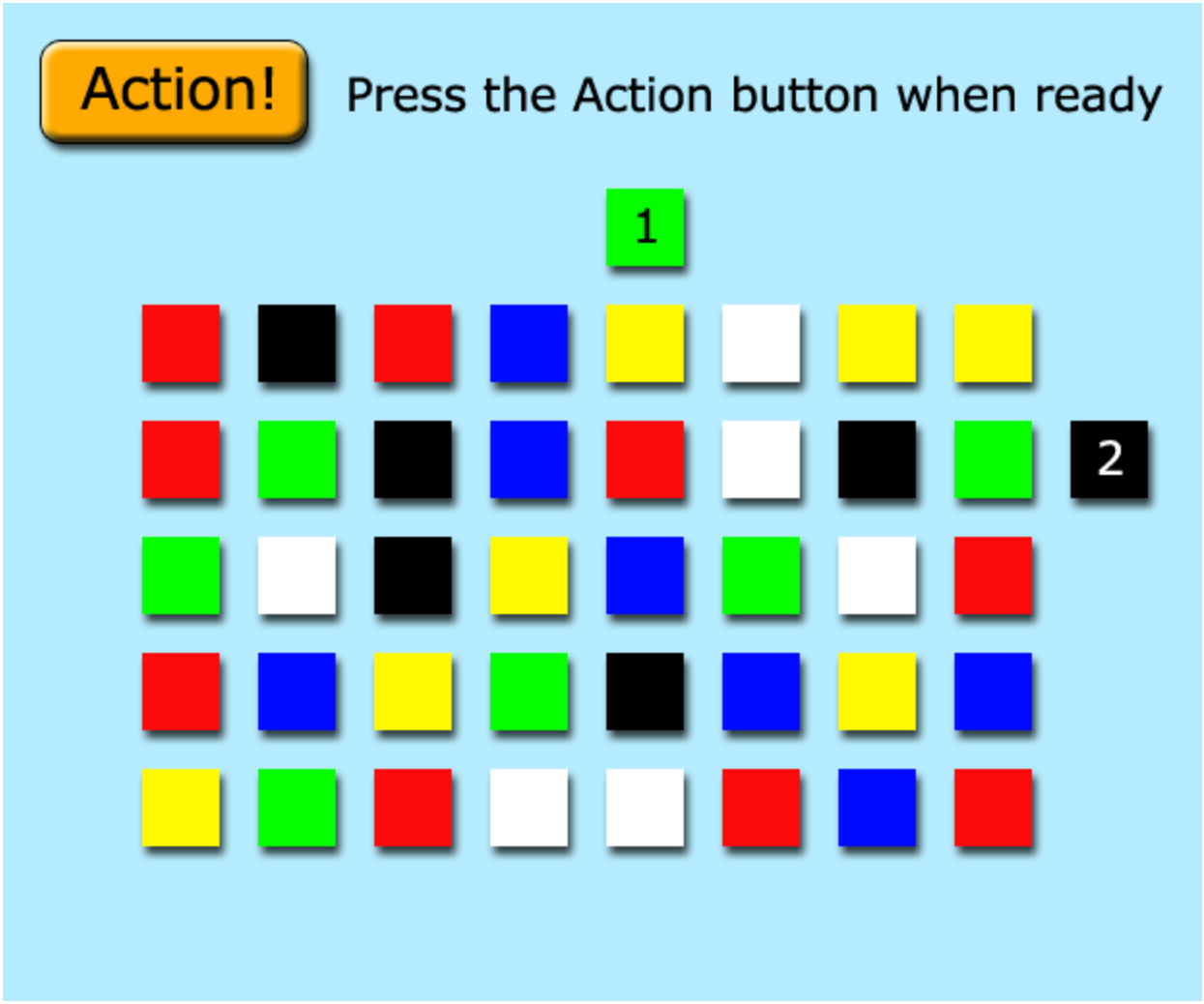
Players may take as long as they want during the planning phase. When players are done planning, they may trigger the start of the action phase. The action phase lasts 4.0 seconds (tunable). The resolution phase shows the results of the previous phases, and lasts as long as necessary for the animations to complete. Then the next planning phase begins, etc. Each phase is described in more detail below.

**Cube Colors**
Each cube in the grid has a color. The valid colors are:

1) Black
2) Blue
3) Green
4) Red
5) Yellow
6) White

**Grid Setup**
The grid is 8 x 5. At the beginning of the game, the grid should be setup with a random selection of colored cubes, but there shouldn't be any group of 3 or more cubes of the same color next to each other (horizontally or vertically).

**Planning Phase**
At the start of each planning phase, 2 "pusher" cubes appear outside the grid, on the edge of it. See sample screenshot below. These pusher cubes should be clearly labeled 1 and 2. The pushers have a random color (and could be the same color).




During the planning phase, the player may use the keyboard to adjust the locations of the 2 pusher cubes. There are 8 valid locations above the grid, 8 below the grid, 5 to the left of the grid, and 5 to the right of the grid. The corners of the grid aren't valid locations for the pusher cubes.

The pusher cubes should always remain aligned with the grid; pick one key to move Pusher 1 clockwise, and another to move it counter clockwise. Pick two more keys for Pusher 2.

If the player tries to put both pusher cubes in the same location, the pusher that was there first remains, and the other pusher goes to the next valid location.

Once the player is done moving the pusher cubes, the player can press the [Action] button, which starts the Action phase.

**Action Phase**
Unlike the Planning phase, the Action phase is timed. It lasts 4.0 seconds (tunable). When the Action phase begins, the following things happen:

1) The Action button disappears, and turns into a large count down timer (starting at 4.0 seconds).
2) The instructional text changes to read, "You may click cubes to destroy them"

During the Action phase, the player may click/touch any cube, including the two pushers. If the player touches a cube, it instantly disappears. Note that players cannot move pushers to new locations during the Action phase, though they can click a pusher to make it disappear. Players may click/touch as many cubes as they want during the Action phase. On a touch device, multitouch should be allowed.

When the timer reaches 0.0, the Action phase ends and the Resolution phase starts.

**Resolution Phase**
When the Resolution phase starts, the player cannot click/touch any more cubes. The player simply watches the results of their decisions. The following things happen in order:

1) The countdown timer disappears, as well as the instructional text.

2) Assuming Pusher 1 wasn't destroyed during the Action phase, it moves one space into the grid, pushing any cubes one space, as appropriate.
    a. For example, if Pusher 1 is above the 5$^{th}$ column, all cubes in the 5$^{th}$ column move down one space, and Pusher 1 ends up in the top row of the 5$^{th}$ column.
    b. It's possible that this movement will cause a cube to end up outside the grid (at the bottom of the 5$^{th}$ column).
    c. If the player clicked a cube in the 5$^{th}$ column during the Action phase, then the pusher would only push as many cubes as appropriate, and then fill in the empty space.
    d. The player should actually see the pusher cube moving, and the pushing shouldn't happen instantly. The total animation should take 1.5 seconds (tunable).

3) After Pusher 1 has finished it's pushing, Pusher 2 moves the same way.
    a. If Pusher 1 and Pusher 2 start opposite each other, and no cubes were clicked during the Action phase, Pusher 2 will undo what Pusher 1 did, and both Pushers will end up outside of the grid!

4) Any cube that is currently outside the grid is collected as loot. See the Loot section for more details. The looted cubes should fade out, or move off screen, and it should take 1.0 seconds (tunable). If the player gets no loot, then skip this step (and take 0 time).

5) Within the grid, score each group of 3-or-more same colored cubes in a row or column. The cubes in the group must be adjacent to score (no intervening cubes). Scored cubes should fade out, and it should take 0.5 seconds (tunable). If multiple groups of cubes are scoring, fade out one group, then another group in order, until all groups have faded out in sequence (do not fade out all groups at the same time).

6) Now, all cubes that have empty space below them (due to scoring, or the Action phase) fall to the bottom of the grid until all spaces are filled in. Add random colored cubes to the top of the grid as needed. This should take 1.0 seconds (tunable). See Bejeweled for an example of how new cubes should fall into the grid. (https://www.youtube.com/watch?v=bfMYDJQ5i10)

7) Once all cubes have filled in, go back to Step 5-6 if any additional scoring should happen. Continue repeating until no more scoring needs to occur.

Once the Resolution phase is done, go to the Planning phase. The [Action] button and the appropriate instruction text should reappear, as well as two random pusher cubes.

**Scoring**
Players score points for matching groups of same-colored cubes as follows:

| Consecutive Cubes | Points |
|---|---|
| 3 | 10 |
| 4 | 20 |
| 5 | 30 |
| 6 | 40 |
| 7 | 50 |
| 8 | 60 |

Players can also earn a combo bonus, if multiple scoring events happen within the same Resolution Phase. For example, if the player goes through steps 1-7, and then gets to Step 5 a second time, all scoring from that event should be worth 2x. If the player gets to Step 5 a third time, scoring from that event should be worth 3x (tunable).

| Combo | Multiplier |
|---|---|
| 2 | 2.0 |
| 3 | 3.0 |
| 4 | 4.0 |
| 5 | 5.0 |
| 6 | 6.0 |
| etc. | etc. |

**Loot**
The player collects loot by pushing cubes out of the grid. At most, the player can get 2 loot per turn. The player should be able to see how many of each color cube they collected.

Loot provides a score bonus. For each loot in a given color, the player earns a 10% score bonus when scoring that color cube. For example, if the player has 2 white cubes in loot and then scores a group of 3 white cubes, it would be worth 12 points instead of the usual 10. Loot and combo bonus stack. Loot applies immediately, including scoring cubes in that same Resolution phase.

**End of Game**
The game ends after 15 turns (tunable). Display the current turns remaining somewhere on screen.

**Game Summary**
Once the game ends, display the Game Summary screen. It should include:
- A big “Victory!” text.
- The player’s score.
- A big [Play Again] button that will return the players to Gameplay.

**Simplifications**
When getting started, or if you're feeling overwhelmed, try these simplifications:

- Use 1 pusher instead of 2.
- Skip the Action phase entirely.
- In the Resolution phase, don't animate the movement of the cubes. Show each step with a second or two delay, so the player can still see what's going on.
- Ingore combo bonuses
- Ingore loot bonuses

**Hints**
If you want to try managing the animations and movements yourself, you should read about Coroutines. They are useful and have funky syntax:

http://docs.unity3d.com/Manual/Coroutines.html

If you want to use code made by someone else to simplify the movement of objects, that's OK in this case, and you might want to check out iTween:

http://itween.pixelplacement.com/

I haven't used it, but it seems relatively good. The iTween.MoveTo() method may provide the vast majority of what you need.

**Useful Link on GUI in Unity**
There are several requirements to display information to the player on screen. This walkthrough Ira made might be useful: https://youtu.be/9aO0BvBLrys

Here's an older link too: https://learn.unity.com/tutorial/ui-components
And specifically #3 on UI Text. You can experiment within Unity by choosing GameObject -> UI -> Text.

**Challenge by Choice**
Do any or all of the following additional tasks. I've included categories below, but you can pick and choose however you want.

Game Design Complexity

1) During the planning phase, allow the two pusher cubes to end up in the same location. In that case, during the Resolution phase, Pusher 1 would push first, and then pusher 2 would push, effectively moving the blocks in that column or row two squares.
2) Change pusher 1 to a large cube that takes up a 2x2 space in the grid. It will push 2 spaces, and push 2 rows/columns at a time. Those large cubes will remain in the grid until pushed out, and whenever they move, they push blocks appropriately. If any part of a large cube ends up out of the grid, the entire cube is removed and it counts for 4 small cubes. Note: This could be quite hard to implement. I'm not sure how to do it off the top of my head, but it seemed fun!

Input Methods

- In the Planning Phase, instead of using the keyboard to move the pushers, allow the player to click and drag the pusher cubes around the edge of the grid. If the player clicks and drags so that a pusher cube isn't aligned with the grid, the pusher cube should snap to the nearest valid location when the player releases the mouse button. (This would allow the game to be played on a smartphone.)

Feedback to the Player

- Account for color blind players by adding a letter to the front of each colored cube, based on the color. Perhaps A, B, C, D, E, F.
- On the Start Screen, include some instructions about how to play
    - For extra challenge: When showing instructions, don’t just use text. Include images of some sort to communicate more clearly.
- Whenever players mouse-over any location (but before clicking on it), give a visual indication that it’s clickable (change the color a bit, enlarge it slightly, etc. Your choice of what to do, but do something.)
- When a cube gets scored, do something interesting visually. For example, displaying floating text that appears out of the cube with +10, or however many points were just scored. Or make the cube explode into a bunch of tiny bits.
- Somewhere on screen, display the player's current score.
    - Along with the player's current score, show the word "Score."
- On the Game Summary screen, show additional information like:
    - Total cubes that appeared of each color
    - High score
    - Scores from the 10 most recent games
    - A tip on how to play better
    - Anything else potentially interesting to the player

Sound Effects

- Play a sound effect (SFX) whenever interesting things happen in the game.
- Play music that loops, so there's constantly music playing during the game.

Visual Polish

- Make the game look prettier, by any means you deem appropriate.

Original project designed by Professor Ira Fay, with help from Sam Goodspeed.