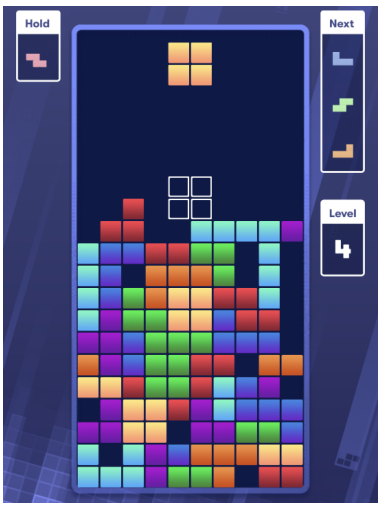
「3Dテトリス（3DTetris）」設計案

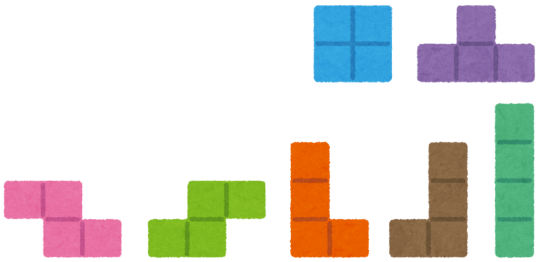
概要

タイトル	3Dテトリス（3DTetris）
ジャンル	パズル
概要・特徴	某人気パズルゲームのテトリスの3Dバージョンで、ステージを360°見渡すことができる
ルール	制限時間内のスコアを競う
ゲームオーバー条件	ブロックを上限まで積み上げてしまったとき
ゲームクリア条件	制限時間終了まで、ブロックの高さを制限内に抑える
ゲームサイクル	1. ゲームタイトル画面にてゲームスタート演出を行う 2. 「Game Start」ボタンが押されたらゲーム開始 3. ゲームオーバーならゲームオーバー演出を行い「1」へ遷移 4. 制限時間終了でゲームクリア演出を行う（スコアを表示） 5. 「1」へ遷移（以降ループ）
備考	• ブロック数の制限は10×20個 • 画面右上に以降3つまでのブロックが表示される • 1つだけブロックをホールドでき、ホールドしたブロックは画面左上に表示される • マウスカーソルは常に非表示 • ブロックの落下速度は常に一定 • 1列消化すると、10ポイントが貰える

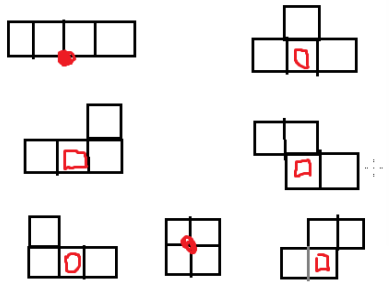
ゲーム画面モデル



ブロックの種類と色



ブロックの回転軸



操作方法

左右移動	「→」or「←」
降下速度UP	「↓」
ホールド・使用	「↑」
視点移動	マウス

右に90°回転	右クリック
左に90°回転	左クリック

スクリプト

スクリプト名	処理内容	備考
GameData.cs	• ~~ゲーム全体のデータの管理~~	• ~~シングルトン~~
BlockController.cs	• ~~左右移動~~ • ~~降下速度UP~~ • ~~ホールド・使用~~ • ~~回転~~ • ~~ブロック自身の情報を管理・制御~~	• ~~各ブロックにアタッチ~~
BlockDataSO.cs	• ~~ブロックのプレファブの管理~~ • ~~ブロックのイメージ画像の管理~~	
BlockGenerator.cs	• ~~ブロックのランダムな生成~~	
BlockManager.cs	• ~~ブロックの消化~~ • ~~ホールドされたブロックのデータの管理~~ • ~~ホールドされたブロックの使用~~ • ~~ゴーストの生成~~	• ~~シングルトン~~
UIManager.cs	• ~~ゲームスタート演出~~ • ~~ゲームオーバー演出~~ • ~~ゲームクリア演出~~ • ~~スコア表示の更新~~ • ~~制限時間の表示~~ • ~~以降3つのブロックのイメージの表示~~ • ~~ホールドされているブロックの表示~~	• ~~シングルトン~~

SoundDataSO.cs	• ~~音のデータの管理~~	
SoundManager.cs	• ~~音の取得~~ • ~~音の再生~~ • ~~音の停止~~	• ~~シングルトン~~
GhostController.cs	• ~~ゴーストの可視化~~ • ~~ゴーストの移動~~	

実装手順

1. ~~GitHubとSourceTreeのリポジトリを作成~~（9/6に実装済み）
2. ~~ステージ・ブロック等を作成~~（9/6に実装済み）
3. ~~Cinemachineを利用した視点移動の実装~~（9/6に実装済み）
4. ~~GameData.csを作成~~（9/6に実装済み）
5. ~~BlockController.csを作成~~（9/6に実装済み）
6. ~~BlockDataSO.csを作成~~（9/6に実装済み）
7. ~~BlockGenerator.csを作成~~（9/6に実装済み）
8. ~~BlockManager.csを作成~~（9/7に実装済み）
9. ~~UIManager.csを作成~~（9/8に実装済み）
10. ~~GameManager.csを作成~~（9/8に実装済み）
11. ~~SoundDataSOを作成~~（9/9に実装済み）
12. ~~音のインポートと設定を行う~~（9/9に実装済み）
13. ~~SoundManager.csを作成~~（9/9に実装済み）
14. ~~GhostController.csを作成~~（9/9に実装済み）