天地鳴動：プロトタイプ主要システム設計書

1：ゲームサイクルとフローシステム

1.1:GameManagerクラス
ゲームの進行状態、フェイズ管理、シーン遷移などを統括するシングルトンクラス

・処理
・ゲーム全体の初期化と終了処理
・GamePhase（出撃、戦闘、ステージクリア、ゲームオーバー）の管理と遷移
・シーン管理（タイトル、マップ選択、戦闘、リザルト）
・各フェイズにおけるUIの表示/非表示制御
・セーブ/ロードシステムの呼び出し
・クラス
・CurrentPahse：現在のゲームフェイズ(enum GamePhase)
・CurrentStageId：現在のステージID(int)
・メソッド
/InitializeGane();ゲーム全体の初期化
/ChangePhase(GamePhase newPhase);フェイズを切り替える
/LoadScene(String SceneName);シーンをロード
/SaveGame();ゲームをセーブする
/LoadGame();ゲームをロードする

1.2:GamePhase Enum
ゲームの各フェイズを定義する列挙型
/Title;タイトル画面
/Deployment;出撃フェイズ
/Battle;戦闘フェイズ
/StageClear;ステージクリア
/GameOver;ゲームオーバー
/Loading;ロード中

2：バトルシステム
ターン制シミュレーションRPGの核となるシステム群

2.1:BattleManagerクラス
戦闘全体の進行、ターン管理、ユニットの行動順序などを統括するシングルトンクラス

・処理
・戦闘の初期化と終了処理
・ターン開始/終了処理
・ユニットの行動順序管理
・勝利/敗北条件の判定とゲームオーバー処理
・不確定要素システムとの連携
・ダイクストラ法によるユニットの移動命令の中継
・クラス
/CurrentTurn;現在のターン数(int)
/ActiveUnit;現在行動中のユニット(Unitクラスインスタンス)
/AllUnits;現在マップ上に存在する全ユニットのリスト(List<Unit>)
・メソッド
/StartBattle();戦闘を開始する
/EndTurn();ターンを終了し、次のターンの処理を開始する

/CheckWinCondition();勝利条件を判定する
/CheckLoseCondition();敗北条件を判定する
/ApplyUncertaintyEffect();不確定要素の効果を適用する
/MoveUnit(Unit unit, Vector2Int targetPosition)指定ユニットをターゲット位置へ移動させる
(内部でDijkstraPathfinderを呼び出す)
/PerformAttack(Unit attacker, Unit defender)ユニット間の攻撃処理実行する

2.2:Unitクラス
ゲーム内の各ユニット(プレイヤー、敵)の情報を管理するクラス

・処理
・ユニットの基本ステータス管理
・現在のHP、移動力、経験値の管理
・装備品、特性、現在の行動状態(行動済みか否か)の管理
・レベルアップ、経験値獲得処理
・クラス
/UnitId;ユニットのユニークID(string)
/UnitName;ユニット名(string)
/CurrentHP;現在のHP(int)
/MaxHP;最大HP(int)
/CurrentMovementPoints;現在の移動力(ダイクストラ法計算で参照される)(int)
/AttackPower;攻撃力(int)
/Skill;技(int)
/Speed;速さ(int)
/Defense;防御力(int)
/CurrentEXP;現在の経験値(int)
/CurrentLevel;現在のレベル(int)
/UnitType;(enum UnitType)ユニットの特性(enum UnitType)
/EquippedWeapon;(Weapon)装備中の武器(Weaponクラスインスタンス)
/CurrentGridPosition;(Vector2Int)マップ上の現在のグリッド座標(Vector2Int)
/HasActedThisTurn;今ターン行動済みか(bool)
・メソッド
/UpdatePosition(Vector2Int newPosition);ユニットのグリッド座標を更新する
/ConsumeMovementPoints(int cost);移動ポイントを消費する
/TakeDamage(int damage);ダメージを受ける
/GainExperience(int exp)経験値を獲得し、レベルアップ処理を行う
/ApplyBuff(BuffType type, int value)バフ効果を適用する
/ApplyDebuff(DebuffType type, int value)デバフ効果を適用する
/ResetAction();ターン開始時に行動済み状態をリセットする

2.3:UnitType Enum
ユニットの特性を定義する列挙型

・Infantry;歩兵
・Aquatic;水棲
・Flying;飛行
・Cavairy;騎馬
・Heavy;重兵
・Archer;弓兵
・Mountain;山賊

2.4:Weaponクラス

武器の情報を管理するクラス
・処理
　　・武器の基本ステータス管理
　　・武器経験値、レベルアップ管理
・クラス
　　/WeaponId;武器のユニークID(string)
　　/WeaponName;武器名(string)
　　/WeaponType;武器の種類(enum WeaponType)
　　/AttackPower;攻撃力(int)
　　/HitRate;命中率(int)
　　/Range;攻撃範囲(int)
　　/CurrentWeaponEXP;現在の武器経験値(int)
　　/CurrentWeaponLevel;現在の武器レベル(int)
・メソッド
　　/GainWeaponExperience(int exp);武器経験値を獲得し、レベルアップ処理を行う

2.5:WeaponType Enum
武器の種類を定義する列挙型

・Sword;剣
・Axe;斧
・Lance;槍
・Bow;弓

2.6:戦闘計算式

・武器相性：剣＞斧＞槍の三すくみ
　　・有利な場合：命中+15%
　　・不利な場合：命中-15%
・命中率計算：（攻撃側の技*2 + 武器の命中）-（防御側の速さ*2 + 地形回避率）
・回避率計算：攻撃側の命中率に対する、防御側の実質的な回避率
・追撃：攻撃側の速さ - 防御側の速さ≧4の場合、追撃が発生2回攻撃
・ダメージ計算：（攻撃力 + 武器攻撃力）-（防御力 + 地形防御力ボーナス）

3：マップシステム
ゲームマップの生成、管理、地形効果などを扱うシステム。ダイクストラ法で使用されるマップデータを提供する

3.1:MapManagerクラス
ゲームマップの生成、管理、地形効果の適用などを統括するシングルトンクラス

・処理
　　・ステージIDに基づいてマップデータをロードし、マップを生成する
　　・各グリッドの地形タイプ、ユニットの配置状況を管理
　　・地形による移動コスト、回避率、防御力ボーナスなどを提供
　　・不確定要素による地形変化を適用する
　　・DijkstraPathfinderが必要とするマップ情報(タイルごとの移動コスト、通行可能性)を提供
・クラス
　　/GridSize;マップのグリッドサイズ(Vector2Int)
　　/TileGrid;各グリッドの情報を格納する2次元配列(Tile[,])
・メソッド
　　/LoadMap(int stageId);指定されたステージIDのマップデータをロードし、グリッドを生成
　　/GetTileAt(Vector2Int position);指定座標のタイル情報を取得する

/GetMovementCost(Vector2Int position, UnitType unitType);指定されたグリッドとユニットタイプに応じた移動コストを返す。DijkstraPathfinderがこのメソッドを利用する
/IsValidGridPosition(Vector2Int position);指定された座標がマップ範囲内か判定する
/ApplyTerrainEffect(Vector2Int position);指定座標の地形効果を適用する
/ChangeTerrain(Vector2Int position, TerrainType newType)特定のグリッドの地形タイプを変更する

3.2:Tileクラス
マップの各グリッドの情報を管理するクラス
・処理
・グリッドの座標、地形タイプ、上に存在するユニットの参照を保持
・地形ごとの防御力ボーナス、回避ボーナスを提供する
・クラス
/GridPosition;グリッドの座標(Vector2Int)
/TerrainType;地形の種類(enum TerrainType)
/OccupyingUnit;そのグリッドに存在するユニット(Unitクラスインスタンス、ない場合は null)
・メソッド
/GetDegenseBonus();防御力ボーナスを返す
/GetEvadeBonus();回避ボーナスを返す

3.3:TerrainType Enum
地形の種類を定義する列挙型
/Plain;平原
/Forest;森
/Mountain;山
/Desert;砂漠
/Water;水場
/River;川
/Snow;積雪
/Flooded;水害
/Landslide;土砂
/Paved;舗装

3.4:DijkstraPathfinderクラス
ダイクストラ法を用いて、開始地点から到達可能な全てのマスへの最短コストを計算するクラス

・処理
・MapManagerからマップ情報を取得し、ダイクストラ法アルゴリズムを実行する
・指定された移動力内で到達可能な全てのマスのリストとその最小移動コストを返す
・クラス
・なし（メソッド中心のユーティリティクラス）
・メソッド
/FindReachableTiles(Vector2Int startPos, Unit unit);ダイクストラ法アルゴリズムを実行し、指定された移動力(unit.CurrentMovementPoints)内で到達可能な全てのマス(Dictionary<Vector2Int,int>)とその最小移動コストを返す。UIでの移動範囲表示や実際の移動パス計算に利用
/GetPathToTarget(Vector2Int startPos, Vector2Int targetPos, Unit unit);
FindReachableTilesの結果から、指定された目標地点への最短経路(List<Vector2Int>)を再構築して返す

3.5PathNodeクラス(DijkstraPathfinder内部用)
DijkstraPathfinderが経路計算に用いるノードオブジェクト

・処理
・グリッド座標、累積コスト、親ノードの情報を保持
・クラス
/GridPosition;グリッド座標(Vector2Int)
/CostFromStart;スタート地点からの実コスト(int)
/Parent;経路再構築のための親ノード(PathNode)
/IsWalkable;そのマスが通行可能かの判定(bool)(MapManagerから取得するため、不要であれば削除)

4：不確定要素システム
ターン経過によって天候や地形を変化させるシステム

4.1:UncertaintyManagerクラス
不確定要素の発生、適用、兆候の管理を統括するシングルトンクラス

・処理
・特定のターン経過またはイベントによって不確定要素を発生させる
・発生する不確定要素の種類をランダムまたは設定されたロジックで決定
・不確定要素によるマップ（地形）やユニット（ステータス）への影響を適用する
・不確定要素の変化の兆候を生成し、UIManagerを通じて表示
・クラス
/CurrentWeather;現在の天候タイプ(enum WeatherType)
/NextUncertaintyEventTurn;次の不確定要素が発生するターン(int)
・メソッド
/GenerateUncertaintyEvent();不確定要素イベントを生成する
/ApplyWeaterEffect(WeatherType weather);天候効果をマップ全体または指定範囲に適用する
/PredictUncertainty();不確定要素の変化の兆候を生成し、プレイヤーに提示する
/GetAffectedUnits(WeatherType weather);特定の天候によって影響を受けるユニットのリストを返す

4.2:WeatherType Enum
天候の種類を定義する列挙型
/Clear;晴れ
/HeavyRain;豪雨
/Blizzard;吹雪
/Sandstorm;砂嵐
/Drought;日照り
/Tornado;竜巻

6：UI/UXシステム
プレイヤーとのインタラクション、情報表示を担うシステム

6.1:UIManagerクラス
ゲーム内のUI要素の表示、更新、プレイヤーからの入力処理を統括するシングルトンクラス

・処理
・HUD（HPバー、時間表示、ターン数表示など）の管理
・メニュー画面（ユニットのステータス、インベントリ、オプションなど）の表示と制御
・ダイクストラ法で計算された移動可能範囲の表示

・ダイクストラ法で計算された経路の視覚的な表示
・ユニットの選択、移動範囲表示、攻撃対象選択などのインタラクション処理
・不確定要素の兆候、メッセージなどの表示
・プレイヤーの入力を受け取り、BattleManagerにイベントを通知

・クラス

マップ画面時
/MapHPBarUI;マップ画面用のHPバーのUIコンポーネント
/UnitInfoPanel;ユニット詳細情報パネル
/TurnCounterText;ターン数表示テキスト
/UncertaintyForecastText;不確定要素の予測表示テキスト
/MoveHighlightPrefab;移動可能マスをハイライトするPrefab
/PathLineRenderer;経路を描画するためのLineRendererコンポーネント
/TimeDisplayUI;時間表示のUIコンポーネント
/PhaseDisplayUI;フェイズ表示のUiコンポーネント
/CursorInfoPanel;カーソル位置にユニットがいるかの可否と地形情報を表示するパネル
/UnitStatusPanel;選択中のユニットの現在のステータスを表示するパネル
/TerrainInfoPanel;カーソル地点の地形の詳細情報を表示するパネル
/MenuIconUI;各種メニューへのアクセスを示すアイコンのUIコンポーネント
戦闘画面時
/AttackerStatusUI;戦闘時の攻撃側のステータスを表示するUIコンポーネント
/DefenderStatusUI;戦闘時の防御側のステータスを表示するUIコンポーネント
/BattlePredictionUI;戦闘結果の予測（ダメージ、命中率）を表示するUIコンポーネント
/EquippedWeaponInfoUI;戦闘時の装備中の武器情報を表示するUIコンポーネント

・メソッド

/UpdateHUD();HUD情報を更新する
/ShowUnitInfo(Unit unit);指定ユニットの詳細情報を表示する
/DisplayMovementRange(Dictionary<Vector2Int,int> reachableTiles);移動可能な範囲をマップ上にハイライト表示する
/DrawPath(List<Vector2Int> path);計算された経路をマップ上に描画する
/ClearHighlightAndPath();ハイライトと経路表示をクリアする
/DisplayAttackRange(List<Vector2Int> attackableTiles);攻撃可能な範囲をマップ上にハイライト表示する
/ShowForecast(string forecastMessage);不確定要素の予測メッセージを表示する
/OnTileClicked(Vector2Int clickedPosition);マップ上のタイルがクリックされた際の処理
/OnUnitSelected(Unit selectedUnit);ユニットが選択された際の処理
/UpdateMapScreemUI();マップ画面時のUI（HPバー、時間、フェイズ、カーソル情報、ユニットステータス、地形情報、メニューアイコン）を更新する
/ShowBattleScreenUI(Unit attacker, Unit defender, BattlePredictionData prediction, Weapon attackerWeaon);戦闘画面時のUI（両者のステータス、戦闘予測、装備中の武器情報）を表示する
/HideBattleScreenUI();戦闘画面UIを非表示にする
/UpdateCursorInfo(Unit unitOnCursor, Tile terrainInfo);カーソル地点のユニット情報と地形情報を更新する
/UpdateUnitStatus(Unit unit);選択中のユニットのステータスパネルを更新する
/ShowBattlePrediction(Unit attacker, Unit defender, BattlePredictionData prediction);戦闘予測UIを表示し、データを設定する

6：データ管理システム
ゲーム内の各種データを管理し、セーブ/ロードを行うシステム

6.1:DataManagerクラス
ユニット、武器、マップなどのマスターデータを管理し、セーブデータとの連携を行うシングルトンクラス

・処理
　　・CSVまたはXML形式のマスターデータをロードし、メモリ上に保持
　　・セーブデータの読み書き（PlayerPrefsまたはファイルI/Oを使用）
　　・データの一貫性と整合性の保証
・クラス
　　/UnitMasterData:ユニットのマスターデータ(Dictionary<string, UnitData>)
　　/WeaponMasterData;武器のマスターデータ(Dictionary<string, WeaponData)
　　/MapMasterData;マップのマスターデータ(Dictionary<int , MapData>)
・メソッド
　　/LoadAllMasterData();全てのマスターデータをロードする
　　/SaveGameData(GameSaveData saveData);ゲームデータをセーブする
　　/LoadGameData();ゲームデータをロードする
　　/GetUnitData(string unitId);指定IDのユニットマスターデータを返す
　　/GeeWeaponData(string weaponId);指定IDの武器マスターデータを返す

6.2:データクラス（構造体）
各マスターデータに対応するデータ構造を定義

/UintData:UnitId,UnitName,BaseHP,BaseAttackPower,..（マスターデータ用）..
/WeaponData;WeaponId,WeaponName,WeaponType,BasseAttackPower,HitRate,Range,...
（マスターデータ用）
/MapData;StageId,MapSize,InitialUnitPlacements,TerrainLayout,..（マスターデータ用）
/GameSaveData;CurrentTurn,PlayerUnitsData(List<UnitSaveData>),
EnemyUnitsData(List<UnitSaveData>),CurrentWeather,....（セーブデータ用）
/UnitSaveData;UnitId,CurrentHP,CurrentMovementPoints,CurrentEXP,EquippedWeapon,
GridPosition,HasActedThisTurn,...（ユニットのセーブデータ用）

７：サウンドシステム

7.1:AudioManager

８：視覚化システム

8.1:TileVisualizerクラス

8.2:UnitVisualizerクラス

９：入力システム

9.1:InputManagerクラス

9.2:InputType Enum

9.3:XboxControllerBinding

１０：技術要件と開発環境

・開発言語：C#
・開発環境：Visual Studio Code
・ゲームエンジン：Unity

・バージョン管理：Git
・データエディタ：CSVまたはXML形式でインポート/エクスポート可能
・AIツール：Gemini