Project Bellerophon

多目的創作集団 G.K.F　長月翼

2024 年 5 月 29 日

目次

1

Project Bellerophon について

1.1 Project Bellerophon へようこそ

Project Bellerophon(以下 Bellerophon) は，スクリプトを組むことで SRPG(シミュレーションロールプレイングゲーム) の作成を行える Unity を用いたフレームワークプロジェクトです。

1.2 Bellerophon の構成

```
ProjectBellerophon.exe: Bellerophon 本体。ゲームを起動する際はこの exe を起動。
data: ユニットデータやマップデータなど各種定義情報を格納。
  character: キャラクターデータを格納。
  common: 汎用定義情報を格納。
    settings.json: カーソルやシステムメニューの構成などシステムの基本設定。
  map: マップデータを格納。
  message: 戦闘時の各種メッセージデータを格納。
  tiles: マップデータに用いられるタイルデータを格納。
  unit: ユニットデータを格納。
resource: 画像や音声など各種素材を格納。
  image: 各種データで使用される画像ファイルを格納
    unit: ユニットデータで使用される画像ファイルを格納
    back: 背景画像として使用される画像ファイルを格納
    character: キャラクターデータで使用される画像ファイルを格納
    map: タイルデータで使用される画像ファイルを格納
    system: 汎用定義情報で使用される画像ファイルを格納。
  music: BGM としてループ再生される音声データを格納。
  sound: 効果音として一度だけ再生される音声データを格納。
script: ゲームの進行を定義するスクリプトファイルを格納。
  initialize.json: ゲームを起動し、一番最初に実行されるスクリプト定義。
  system.json: 標準ではシステムメニューから実行されるスクリプト定義。

（以下、Unity 既定の構成ファイルのため無視してください）
UnityCrashHandler64.exe
UnityPlayer.dll
D3D12
MonoBleedingEdge
ProjectBellerophon_Data
```

1.3 ファイル形式

Bellerophon では，定義情報に使えるファイル形式として JSON 形式と YAML 形式の二つに対応しています。画像は PNG 形式や TGA 形式、BMP 形式など多種の形式、音声は AIF 形式、WAVE 形式、MP3 形式、OGG 形式の 4 つの形式に対応しています。

2
データ定義

2.1 システム設定

▼ 表 2.1　システム設定

debug_mode	デバッグモード。 false:Off、true:On
tool_mode	各種ツールをタイトルに表示するか否か。 false: 非表示、true: 表示
cursor	カーソル定義。
id	カーソルタイルセット名。”cursor”固定
pixels	カーソルタイルの基準ピクセル。標準は 32(px)
tile[]	カーソルタイル定義。
name	カーソルタイル名
base_layer	基本レイヤー
resource	画像ファイル名
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
overlay_back	装飾レイヤー (後)
resource	画像ファイル名
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
overlay_front	装飾レイヤー (前)
resource	画像ファイル名
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
map_cursor[]	マップカーソル指定。
move_area[]	移動領域指定。
attack_area[]	攻撃領域指定。
group_marker	グループマーカー指定。
self_marker	自機マーカー指定。
target_marker	目標マーカー指定。
system_menu[]	システムメニュー定義。
text	メニュー表示テキスト。
bold	テキストの太字表示。 false:Off、true:On
italic	テキストのイタリック表示。 false:Off、true:On
name	メニュー選択時に実行されるスクリプトのクラス名。
method	メニュー選択時に実行されるスクリプトのメソッド名。

settings.json

```
{
  "debug_mode": true,
  "tool_mode": true,
  "cursor": {
    "id": "cursor",
    "pixels": 32,
    "tile": [
      {
        "name": "sample_cursor",
        "base_layer": {
          "resource": "UI.png",
          "pos": [{ "x": 0, "y": 192 }],
          "color": { "r": 128, "g": 128, "b": 128, "a": 128 }
        },
        "overlay_back": {
          "resource": "UI.png",
          "pos": [{ "x": 0, "y": 192 }],
          "color": { "r": 128, "g": 128, "b": 128, "a": 128 }
        },
        "overlay_front": {
          "resource": "UI.png",
          "pos": [{ "x": 0, "y": 192 }],
          "color": { "r": 128, "g": 128, "b": 128, "a": 128 }
        }
      }
    ]
  },
  "map_cursor": ["sample_cursor"],
  "move_area": ["sample_cursor"],
  "attack_area": ["sample_cursor"],
  "group_marker": "sample_cursor",
  "self_marker": "sample_cursor",
  "target_marker": "sample_cursor",
  "system_menu": [
    {
        "text": "サンプルメニュー",
        "bold": false,
        "italic": false,
        "name": "system",
        "method": "sample_method"
    }
  ]
}
```

settings.yml

```
debug_mode: true
tool_mode: true
cursor:
  id: cursor
  pixels: 32
  tile:
    - name: sample_cursor
      base_layer:
        resource: UI.png
        pos:
          - {x: 0, 'y': 192}
        color: {r: 128, g: 128, b: 128, a: 128}
      overlay_back:
        resource: UI.png
        pos:
          - {x: 0, 'y': 192}
        color: {r: 128, g: 128, b: 128, a: 128}
      overlay_front:
        resource: UI.png
        pos:
          - {x: 0, 'y': 192}
        color: {r: 128, g: 128, b: 128, a: 128}
map_cursor:
  - sample_cursor
move_area:
  - sample_cursor
attack_area:
  - sample_cursor
group_marker: sample_cursor
self_marker: sample_cursor
target_marker: sample_cursor
system_menu:
  - text: サンプルメニュー
    bold: false
    italic: false
    name: system
    method: sample_method
```

2.2 キャラクターデータ

▼ 表 2.2　キャラクターデータ

id	キャラクター ID。
name	キャラクター名。
image	キャラクター顔画像。
status	キャラクター能力値
concentration	集中。攻撃の命中力に影響します。
reaction	反応。攻撃に対する回避力に影響します。
ability	技量。攻撃の致命力に影響します。
perception	知覚。攻撃に対する致命回避力に影響します。
intention	意思。攻撃の威力に影響します。
endurance	耐久。攻撃に対する防御力に影響します。
expertise	熟練。戦闘に直接的な関係はありません。
sp	ＳＰ。
suitability	地形適正。S、A、B、C、D、E の六段階で設定し、E は全く適正が無いことを表します。
space	宇宙適正。
air	空中適正。
ground	地上適正。
underwater	水中適正。
growth	成長値
concentration	1 レベル毎に上昇する集中。
reaction	1 レベル毎に上昇する集中。
ability	1 レベル毎に上昇する技量。
perception	1 レベル毎に上昇する知覚。
intention	1 レベル毎に上昇する意思。
endurance	1 レベル毎に上昇する耐久。
expertise	1 レベル毎に上昇する熟練。
sp	1 レベル毎に上昇するＳＰ。
suitability[]	地形適正の変化
lv	地形適正が変化するレベル。
space	変化後の宇宙適正。
air	変化後の空中適正。
ground	変化後の地上適正。
underwater	変化後の水中適正。
special[]	次期バージョンで実装

キャラクターデータ (JSON 形式)

```
{
 "character": [
  {
   "id": "test_0001",
   "name": "Test Character 0001",
   "image": "None.png",
   "status": {
    "concentration": 10,
    "reaction": 10,
    "ability": 10,
    "perception": 10,
    "intention": 10,
    "endurance": 10,
    "expertise": 10,
    "sp": 50
   },
   "suitability": {
    "space": "E", "air": "E", "ground": "A", "underwater": "D"
   },
   "growth": {
    "concentration": 1,
    "reaction": 1,
    "ability": 1,
    "perception": 1,
    "intention": 1,
    "endurance": 1,
    "expertise": 1,
    "sp": 2,
    "suitability": [
     {
      "lv": 5, "space": "D", "air": "D"
     },
     {
      "lv": 10, "ground": "S"
     }
    ]
   },
   "special": []
  }
 ]
}
```

キャラクターデータ (YAML 形式)

```yaml
character:
  - id: test_0001
    name: Test Character 0001
    image: None.png
    status:
      concentration: 10
      reaction: 10
      ability: 10
      perception: 10
      intention: 10
      endurance: 10
      expertise: 10
      sp: 50
    suitability:
      space: E
      air: E
      ground: A
      underwater: D
    growth:
      concentration: 1
      reaction: 1
      ability: 1
      perception: 1
      intention: 1
      endurance: 1
      expertise: 1
      sp: 2
      suitability:
        - lv: 5
          space: D
          air: D
        - lv: 10
          ground: S
    special: []
```

2.3 ユニットデータ

▼ 表 2.3　ユニットデータ

id	ユニット ID。
name	ユニット名。
machine	機械ユニットか否か。false: 生物ユニット、true: 機械ユニット
display	ユニット画像。
base_resource	ユニットマップタイル画像。
pixels	ユニットマップタイルの基準ピクセル。標準は 32(px)
base_image[]	ユニットマップタイルの基本タイルレイヤー
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
front_image[]	ユニットマップタイルの装飾タイルレイヤー
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
status	キャラクター能力値
accuracy	精度。攻撃の命中力に影響します。
maneuver	機動。攻撃に対する回避力に影響します。
power	出力。攻撃の威力に影響します。
armor	装甲。攻撃に対する防御力に影響します。
reduction	軽減。攻撃に対する致命回避力に影響します。
move	移動。1 ターンに移動出来る距離に影響します。
hp	ＨＰ。
en	ＥＮ。
suitability	地形適正。S、A、B、C、D、E の六段階で設定し、E は全く適正が無いことを表します。
space	宇宙適正。
air	空中適正。
ground	地上適正。
underwater	水中適正。
special[]	次期バージョンで実装
weapon[]	攻撃手段
name	攻撃手段名。
attack_type	攻撃種別。short(近接)、long(射撃)、mix(複合)
range	射程。
min	最短射程。
max	最長射程。
suitability	地形適正。S、A、B、C、D、E の六段階で設定し、E は全く適正が無く攻撃不可であることを表します。
space	宇宙適正。
air	空中適正。
ground	地上適正。
underwater	水中適正。
accuracy	精度。攻撃の命中力に影響します。
critical	致命力。
atk	威力。
bullets	装弾数。
special[]	次期バージョンで実装

ユニットデータ (JSON 形式)

```
{
    "unit": [
        {
            "id": "test_0001", "name": "Test Unit 0001", "machine":true,
            "display": "type_c4.png", "base_resource": "type_c4.png", "pixels": 128,
            "base_image": [
                {
                    "pos": [ { "x": 0, "y": 0 } ],
                    "color": { "r": 255, "g": 255, "b": 255, "a": 255 }
                }, {
                    "pos": [ { "x": 0, "y": 384 } ],
                    "color": { "r": 255, "g": 255, "b": 255, "a": 255 }
                }, {
                    "pos": [ { "x": 0, "y": 768 } ],
                    "color": { "r": 255, "g": 255, "b": 255, "a": 255 }
                }, {
                    "pos": [ { "x": 0, "y": 1152 } ],
                    "color": { "r": 255, "g": 255, "b": 255, "a": 255 }
                }
            ],
            "status": {
                "accuracy": 30, "maneuver": 30, "power": 100, "armor": 300,
                "reduction": 100, "move": 4, "hp": 5000, "en": 300
            },
            "suitability": { "space": "E", "air": "D", "ground": "A", "underwater": "D" },
            "special": [],
            "weapon": [
                {
                    "name": "テスト格闘",
                    "attack_type": "short", "range": { "min": 1, "max": 1 },
                    "suitability": { "space": "E", "air": "E", "ground": "A", "underwater": "D" },
                    "accuracy": 0, "critical": 0, "atk": 2000, "bullets": 0,
                    "special": []
                }, {
                    "name": "テスト武装",
                    "attack_type": "long", "range": { "min": 3, "max": 5 },
                    "suitability": { "space": "B", "air": "B", "ground": "A", "underwater": "D" },
                    "accuracy": 0, "critical": 0, "atk": 2500, "bullets": 12,
                    "special": []
                }
            ]
        }
    ]
}
```

ユニットデータ (YAML 形式)

```yaml
unit:
  - id: test_0001
    name: Test Unit 0001
    machine: true
    display: type_c4.png
    base_resource: type_c4.png
    pixels: 128
    base_image:
      - pos:
          - { x: 0, 'y': 0 }
        color: { r: 255, g: 255, b: 255, a: 255 }
      - pos:
          - { x: 0, 'y': 384 }
        color: { r: 255, g: 255, b: 255, a: 255 }
      - pos:
          - { x: 0, 'y': 768 }
        color: { r: 255, g: 255, b: 255, a: 255 }
      - pos:
          - { x: 0, 'y': 1152 }
        color: { r: 255, g: 255, b: 255, a: 255 }
    status:
      { accuracy: 30, maneuver: 30, power: 100, armor: 300,
        reduction: 100, move: 4, hp: 5000, en: 300 }
    suitability: { space: E, air: D, ground: A, underwater: D }
    special: []
    weapon:
      - name: テスト格闘
        attack_type: short
        range: { min: 1, max: 1 }
        suitability: { space: E, air: E, ground: A, underwater: D }
        accuracy: 0
        critical: 0
        atk: 2000
        bullets: 0
        special: []
      - name: テスト武装
        attack_type: long
        range: { min: 3, max: 5 }
        suitability: { space: B, air: B, ground: A, underwater: D }
        accuracy: 0
        critical: 0
        atk: 2500
        bullets: 12
        special: []
```

2.4 メッセージデータ

▼ 表 2.4　メッセージデータ

id	キャラクター ID。
attack[]	攻撃時。
message[]	メッセージ内容。
hit_under	命中率が指定値以下の場合。
hit_over	命中率が指定値以上の場合。
hp_under	ＨＰが指定値以下の場合。
hp_over	ＨＰが指定値以上の場合。
guard[]	防御時 (戦闘処理の都合により未実装)
message[]	メッセージ内容。
hit_under	命中率が指定値以下の場合。
hit_over	命中率が指定値以上の場合。
hp_under	ＨＰが指定値以下の場合。
hp_over	ＨＰが指定値以上の場合。
avoid[]	回避成功時。
message[]	メッセージ内容。
hit_under	命中率が指定値以下の場合。
hit_over	命中率が指定値以上の場合。
hp_under	ＨＰが指定値以下の場合。
hp_over	ＨＰが指定値以上の場合。
damage[]	被ダメージ時。
message[]	メッセージ内容。
hit_under	命中率が指定値以下の場合。
hit_over	命中率が指定値以上の場合。
hp_under	ＨＰが指定値以下の場合。
hp_over	ＨＰが指定値以上の場合。
destroy[]	被撃墜時。
message[]	メッセージ内容。
hit_under	命中率が指定値以下の場合。
hit_over	命中率が指定値以上の場合。
hp_under	ＨＰが指定値以下の場合。
hp_over	ＨＰが指定値以上の場合。

メッセージデータ (JSON 形式)

```
{
  "messages": [
    {
      "id": "test_0001",
      "attack": [
        {
          "message": ["テスト攻撃だよ！"]
        }, {
          "message": ["いっけー！"]
        }, {
          "hit_under": 30, "message": ["当たって！！"]
        }, {
          "hit_over": 80, "message": ["余裕余裕〜♪"]
        }
      ],
      "guard": [
        {
          "message": ["来る……！"]
        }, {
          "hit_over": 80, "message": ["守りを固めるしか……！"]
        }
      ],
      "avoid": [
        {
          "message": ["回避成功！"]
        }, {
          "hit_under": 30, "message": ["そんな攻撃は当たらない！！"]
        }, {
          "hit_over": 80, "message": ["奇跡的な回避！"]
        }
      ],
      "damage": [
        {
          "message": ["うげっ"]
        }, {
          "hp_under": 30, "message": ["やばいっ！"]
        }
      ]
    }
  ]
}
```

メッセージデータ (YAML 形式)

```yaml
messages:
  - id: test_0001
    attack:
      - message:
          - テスト攻撃だよ！
      - message:
          - いっけー！
      - hit_under: 30
        message:
          - 当たって！！
      - hit_over: 80
        message:
          - 余裕余裕〜♪
    guard:
      - message:
          - 来る……！
      - hit_over: 80
        message:
          - 守りを固めるしか……！
    avoid:
      - message:
          - 回避成功！
      - hit_under: 30
        message:
          - そんな攻撃は当たらない！！
      - hit_over: 80
        message:
          - 奇跡的な回避！
    damage:
      - message:
          - うげっ
      - hp_under: 30
        message:
          - やばいっ！
```

2.5 タイルデータ

▼ 表 2.5　タイルデータ

id	タイルセット ID。
pixels	タイルの基準ピクセル。標準は 32(px)
tile[]	タイル定義。
name	タイル ID
collision	床として機能するか否か。false: 床機能無し、true: 床機能あり
suitability[]	地形毎の移動コスト定義
move_type	対象地形。space: 宇宙、air: 空中、ground: 地上、underwater: 水中
cost	移動コストの基準値。
base_image[]	マップタイルの基本タイルレイヤー
resource	マップタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
rules[]	ルールタイル定義
output	ルール種別。single: オートタイル、random: ランダムタイル
resource	ルールタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
neighbors[]	ルールタイルの隣接タイル定義。この条件を満たす場合に設定されたタイル画像が使用されます。
pos	隣接タイルの座標。
flg	隣接タイルの状態。1: 同タイル、2: 同タイル以外
perlin	ランダムタイルのノイズ設定。
overlay_back[]	マップタイルの装飾タイルレイヤー (背面)
resource	マップタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
rules[]	ルールタイル定義
output	ルール種別。single: オートタイル、random: ランダムタイル
resource	ルールタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
neighbors[]	ルールタイルの隣接タイル定義。この条件を満たす場合に設定されたタイル画像が使用されます。
pos	隣接タイルの座標。
flg	隣接タイルの状態。1: 同タイル、2: 同タイル以外
perlin	ランダムタイルのノイズ設定。
overlay_front[]	マップタイルの装飾タイルレイヤー (前面)
resource	マップタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
color	カラー情報を 0〜255 の RGBA で指定
rules[]	ルールタイル定義
output	ルール種別。single: オートタイル、random: ランダムタイル
resource	ルールタイル画像。
pos[]	アトラステクスチャの位置情報。画像サイズは pixels を縦横それぞれ 3x2 にしたものが設定されます。
neighbors[]	ルールタイルの隣接タイル定義。この条件を満たす場合に設定されたタイル画像が使用されます。
pos	隣接タイルの座標。
flg	隣接タイルの状態。1: 同タイル、2: 同タイル以外
perlin	ランダムタイルのノイズ設定。

タイルデータ (JSON 形式)

```
{
  "tiles": [
    {
      "id": "ts_debugtiles", "pixels": 32,
      "tile": [
        {
          "name": "debug_block", "collision": false,
          "suitable": [
            { "move_type": "air", "cost": 1 },
            { "move_type": "ground", "cost": 1 }
          ],
          "base_layer": {
            "resource": "BLRP_DebugTileset.png",
            "pos": [ { "x": 192, "y": 0 } ],
            "color": { "r": 255, "g": 255, "b": 255, "a": 255 }
          },
          "overlay_back": {
            "resource": "BLRP_DebugTileset.png",
            "pos": [ { "x": 0, "y": 192 } ],
            "color": { "r": 255, "g": 255, "b": 255, "a": 255 },
            "rules": [
              {
                "output": "single", "resource": "BLRP_DebugTileset.png",
                "pos": [ { "x": 0, "y": 192 } ],
                "neighbors": [ { "pos": { "x": -1, "y": 0 }, "flg": 2 } ]
              }
            ]
          },
          "overlay_front": {
            "resource": "BLRP_DebugTileset.png",
            "pos": [ { "x": 0, "y": 192 } ],
            "color": { "r": 255, "g": 255, "b": 255, "a": 255 },
            "rules": [
              {
                "output": "random", "resource": "BLRP_DebugTileset.png", "perlin": 0.5,
                "pos": [ { "x": 0, "y": 96 }, { "x": 0, "y": 192 } ]
              }
            ]
          }
        }
      ]
    }
  ]
}
```

タイルデータ (YAML 形式)

```yaml
tiles:
  - id: ts_debugtiles
    pixels: 32
    tile:
      - name: debug_block
        collision: false
        suitable:
          - move_type: air
            cost: 1
          - move_type: ground
            cost: 1
        base_layer:
          resource: BLRP_DebugTileset.png
          pos:
            - { x: 192, 'y': 0 }
          color: { r: 255, g: 255, b: 255, a: 255 }
        overlay_back:
          resource: BLRP_DebugTileset.png
          pos:
            - { x: 0, 'y': 192 }
          color: { r: 255, g: 255, b: 255, a: 255 }
          rules:
            - output: single
              resource: BLRP_DebugTileset.png
              pos:
                - { x: 0, 'y': 192 }
              neighbors:
                - pos: { x: -1, 'y': 0 }
                  flg: 2
        overlay_front:
          resource: BLRP_DebugTileset.png
          pos:
            - { x: 0, 'y': 192 }
          color: { r: 255, g: 255, b: 255, a: 255 }
          rules:
            - output: random
              resource: BLRP_DebugTileset.png
              perlin: 0.5
              pos:
                - { x: 0, 'y': 96 }
                - { x: 0, 'y': 192 }
```

2.6 マップデータ

▼ 表 2.6　マップデータ

id	マップ ID。
music	マップ BGM。
max	マップサイズ。
tile_set	標準タイルセット ID。
init_tile	標準タイル ID。他のタイルが設定されていない場合、高さ 0 全面がこのタイルで初期化されます。
tiles[]	標準タイルセット ID。
pos	タイル座標。X、Y が面座標を表し、Z が高さを表します。
tile_set	タイルセット ID。
id	タイル ID。

マップデータ (JSON 形式)

```
{
  "maps": [
    {
      "id": "default_map",
      "music": "sample.ogg",
      "max": { "x": 10, "y": 10 },
      "tile_set": "ts_city_01", "init_tile": "concrete_01",
      "tiles": [
        {
          "pos": { "x": 0, "y": 0, "z": 0 },
          "tile_set": "", "id": "waterfront_none"
        }, {
          "pos": { "x": 0, "y": 1, "z": 0 },
          "tile_set": "", "id": "waterfront_none"
        }, {
          "pos": { "x": 2, "y": 10, "z": 0 },
          "tile_set": "ts_military_01", "id": "fence_01_inv"
        }
      ]
    }
  ]
}
```

マップデータ (YAML 形式)

```
maps:
  - id: default_map
    music: sample.ogg
    max:
      x: 10
      'y': 10
    tile_set: ts_city_01
    init_tile: concrete_01
    tiles:
      - pos:
          x: 0
          'y': 0
          z: 0
        tile_set: ''
        id: waterfront_none
      - pos:
          x: 0
          'y': 1
          z: 0
        tile_set: ''
        id: waterfront_none
      - pos:
          x: 2
          'y': 10
          z: 0
        tile_set: ts_military_01
        id: fence_01_inv
```

3
スクリプト

3.1 スクリプト制御

3.2 マップ制御

3.3 ユニット制御

3.4 カメラ制御

3.5 メッセージ制御

3.6 オーディオ制御