



VALID
SENSE

Member List

Leader / プランナー / キャラクター	城脇 礼奈	3年 スーパーゲームクリエイター専攻
Planner /	増田 惟沙	3年 ゲーム企画・シナリオ専攻
Mechanical engineer / プランナー	和田 涼太	3年 ゲーム企画・シナリオ専攻
Designer / UI デザイナー	田島 美於	3年 ゲームグラフィックデザイナー専攻
	杉森 風太	2年 スーパーゲームクリエイター専攻
Programmer /	菅原 渉	3年 スーパーゲームクリエイター専攻
	チンラクケン	3年 ゲームプログラマー専攻
	キムハクチャン	2年 ゲームプログラマー専攻

Special thanks

東京ダンス・俳優 & 舞台芸術専門学校の方

- 01** 中山 真佳 キャラクター「サイト」担当
- 02** 新井 啓介 キャラクター「タクタイル」担当
- 03** 池田 華子 キャラクター「スマイル」担当
- 04** 朝倉 天音 キャラクター「テイスト」担当
- 05** 三枝 祐輝 キャラクター「ヒア」担当

卒業生

Designer / Live2D モデラー / UI デザイナー	リキギョウ リュウハイジ
Planner /	宮崎 千優 秋葉 智哉



妨害して、譜面を奪う。 対戦型音楽ゲーム



企画書 城脇礼奈
使用ソフト / CLIP STUDIO PAINT PRO

VALID
SENSE

『ヴァリッド・センス』は2021年度から
2022年前期の完成を目指して制作された
音楽ゲームです。
しかし私自身の実力不足により完成を延期、
新しいメンバーで再出発することとなりました。
辛いこと、折れてしまいそうになることは
沢山ありましたが、結果こうして皆様に
遊んでいただけるところまで形にすることができ、とても嬉しく、また、自身の成長と
自信への大きな糧となりました。
共に制作に励み支えてくれたメンバーと
サポートして下さった講師の先生方、そして
Valid Sense を遊んでくれた全ての方々に
心からの感謝を申し上げます。

城脇 礼奈



イラスト発注書

ジャケットイラスト発注書制作

増田 惟沙

使用ソフト / Excel

No.	名前	ID	横幅(X)pixel	高さ(Y)pixel	解像度	備考	受注者	担当者	確認者	状態
	ジャケットイラスト③soldier_jacket		1920pixel	1080pixel	350dpi	わからない点、不備がありましたら、増田まで連絡をお願いします。		増田惟沙		

詳細 SOLDIERのジャケットイラスト発注書です。画面サイズのものと、正方形にトリミングしたものと、PNGで納品してください。人物と背景のレイヤーを分けておいてください。

参考

背景は光路が行きかうネオンな交差点で、その中心にサイトさんがいるイメージです。

光路①

背景と作曲者のフォントは下のフォントを使用してほしいですが、タダかったら変えても構いません。

題名の下に作曲者

上のようなバキのフレームをお願いします。
バキはネオン調にしてください。

花が無っていたらかっこいいかなって思ふんですけど、ごちゅつきそうならなくてもいいです。
入れる場合は少し発光してると嬉しいです。

サイトさんがモーヴの手？を取り、こららを振り向いている感じでお願いします。
壁の毛の位置から背中が見えてたらえらかかなって思ふんですけど、マイナチだったら思しらやってもいいです。

使用フォント FoglihtenNo04 Font

フォントダウンロードリンク <https://www.fontspace.com/foglihtenn04-font-f13878>

ジャケットイラスト『Soldier』発注書

大まかなイメージイラストは自分で描きました。そこにフォントの指定や画像を載せることで、デザイナーにより伝わりやすい発注書を目指しました。

ジャケットイラスト 『RetroQuest』発注書

FIREWALL INVADER

イラストの元ネタは、「Firewall・Invader」という2年前期にリーダーと共に制作したゲーム。元作品の要素も入れつつ今作品のデザインに合うよう心掛けました。

発注物連絡表						参考素材			
No.	名前	ID	受注者	担当者	確認者	状態			
	ジャケットイラスト④firewall_invader		増田惟沙						

詳細 retro questのジャケットイラスト発注書です。画面サイズのものと、正方形にトリミングしたものと、PNGで納品してください。

参考

ドット絵でおねがいします。
曲の歌名はfirewall・invaderのロゴ画像に書いてください。

**FIREWALL
INVADER**
ファイアーウォール・インベーダー

曲名はバナーの中に書いてください。
歌は曲名に書がほんでいら背景にしてください。

左側には下のバスターくんの服を着たスマイルちゃんを描いてください。

バナー

曲名バナーの下や囲りに歌詞バナーを描いてください。

歌詞バナーをクリックするような感じでマウスを配置してください。

左側には下のラブくんの服を着たアイストちゃんを描いてください。



ゲーム仕様書

仕様書制作

城脇 礼奈 / 増田 惣沙 / 和田 涼太

使用ソフト / Excel

ゲームプロジェクト『Valid Sense』仕様書

最低限必要な部分に、逐一必要な箇所や新しい仕様を追記していきました。
その時々で手の空いているプランナーが担当し、一人当たりの負担を抑え
効率的に進めることができました。

INDEX | 納品表 / スケジュール | 概要 | 操作&筐体 | 画面構成要素 | ルール | 画面遷移 | キャラクター | 妨害スキル | メインゲーム | UI | 譜面 | BGM / SE | チュートリアル

ゲームルール

全体のルール

スキルを使って相手の譜面を妨害しながら、相手レーンを自分の色で全て塗りつぶして勝利する

勝利条件

3回勝負中2回勝利する

キャラクター選択

一曲目
(+リザルト)二曲目
(+リザルト)

三曲目

最終リザルト
(勝者発表)

- プレイヤーが使用するキャラクターを選ぶ。
※キャラの途中変更はできない。
- ※両者が同キャラを選択時2Pカラー化するか、選べなくするかは全体の進行をみて検討

- プレイヤー1がプレイ曲を選ぶ。
※相手の選ぶ難易度によって、
レーン奪取ノーツ数
スキルゲージ上昇率が変化する。
- 一定数コンボを達成するごとにスキルゲージが溜まっていき、スライドで発動。
相手のプレイを妨害する。
詳細はシート「妨害スキル」を参照。
- 曲のラスサビor1分を切るとレーンの色が隠され、曲終了直前に勝利したプレイヤー側のレーンに光るタップノーツが出現。
タップすると相手の最終防衛レーンを押し切る形で勝利演出が入る。
- リザルト表示後、二曲目へ

- プレイヤー2がプレイ曲選び、一曲目と同じようにプレイする。
- お互いが一勝一敗の場合はリザルトの後三曲目へ。
- どちらかが二連勝の場合そのまま最終リザルト
画面全体が勝者側の選択したキャラのイメージカラーに切り替わり終了画面へ。

- 不利な側(奪われたレーン数が多かった方)がプレイ曲を選ぶ。
- 曲のラスサビor1分を切るとレーン全体の色が灰色になってわからなくなり、曲終了直前に勝利したプレイヤー側のレーンに光るタップノーツが出現。
- 最終的なリザルト画面が表示され、画面全体が勝者側の選択したキャラのイメージカラーに切り替わり終了。
※次スタート画面に表示されるのもこの色
詳細は「システム詳細」を参照。

システムについての詳細はシート『システム詳細』を参照

タイミング判定とエフェクト

エフェクトは、ノーツの種類によらず、すべて共通となります

判定	上昇率(100%中)	判定(±ms)	エフェクト
Briliant	—	20(幅40※前後)	BRILLIANT BRILLIANTという文字と共に 赤色のエフェクトが出現する。 キャラクターによる色の変化はない。
Great	—	40(幅80)	GREAT GREATという文字と共に 単色のエフェクトが出現する。レーン (キャラクター)によって色が変化する。
Good	—	100(幅200)	GOOD GOODという文字と共に単色のエフェクト が出現するが、キラキラは出ない。 GREAT同様に色が変化する。
Poor	—		判定バーを過ぎると、ノーツがフェードアウトして消える。

ミス判定の演出

判定バーを過ぎると、ノーツがフェードアウトして消え、COMBOのところにMISSエフェクトが表示される。

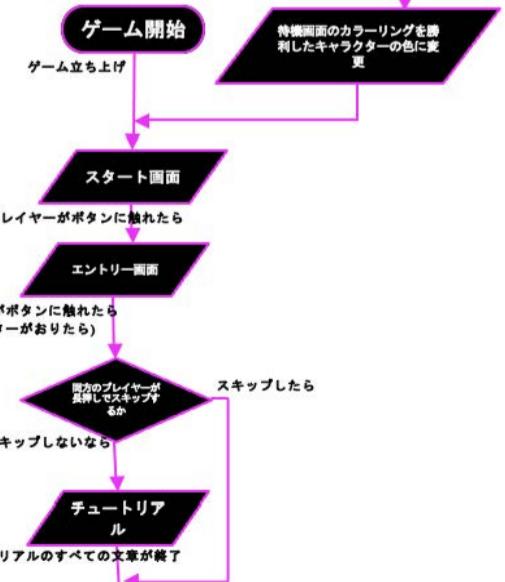
MISS

ミスしたノーツのあるレーンの色が0.3秒デフォルト(灰色)に変化して戻る。

曲選択画面



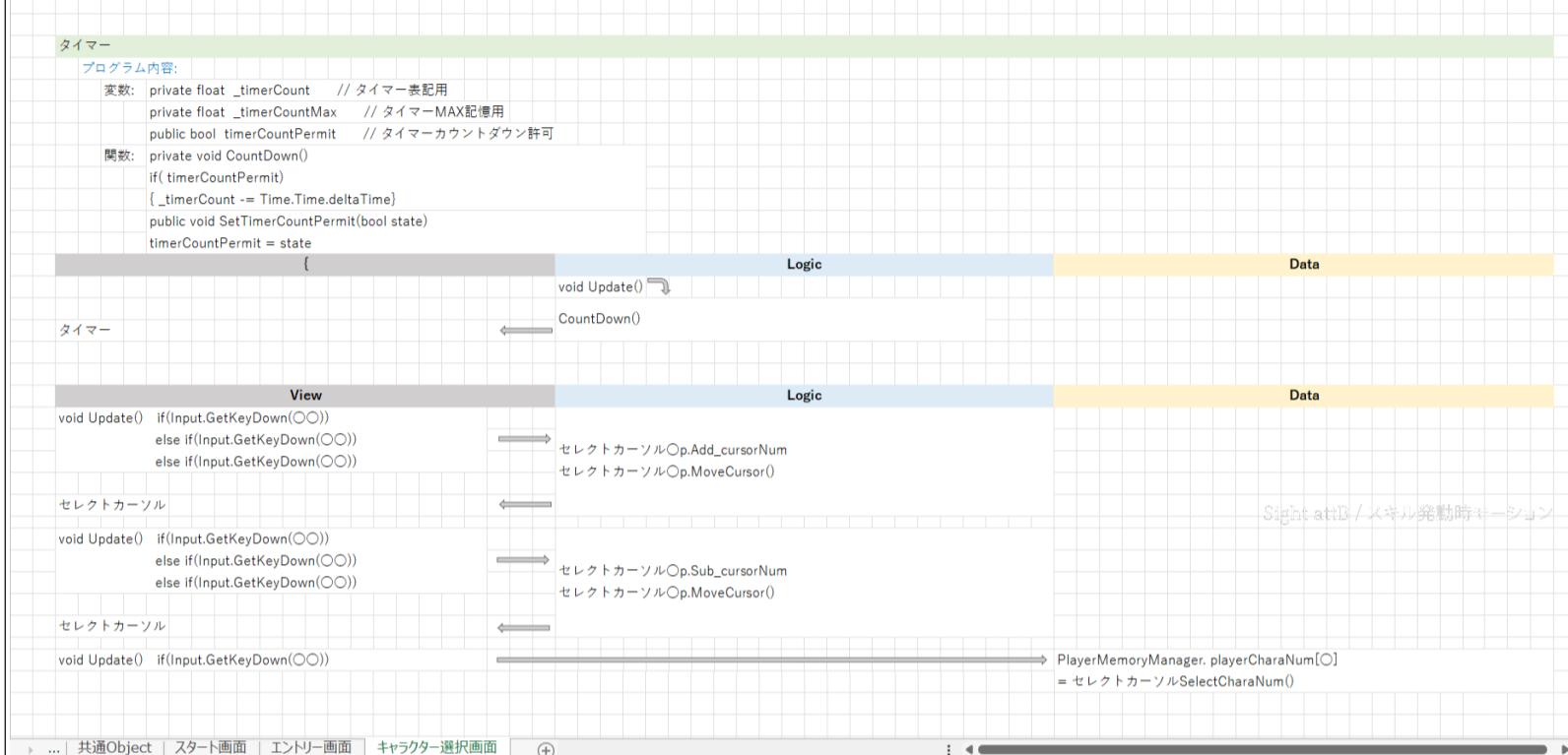
フロー





プログラマー / タスク管理
役割分担書類, プロジェクト引き継ぎ書類
菅原 涼
使用ソフト / Excel
制作期間 / 3日

キャラクター選択画面



役割分担書類

「View」「Logic」「Data」で、プログラマーの役割分担を予定していたため、先に制作していた「要素と簡単な機能をまとめた書類」を元に、より細かく相互関係や参照するタイミングに重点を置いた書類です。細かくした分制作に時間がかかってしまいましたが、作業の割振り時にどのようなプログラムを制作して欲しいかを伝えるのに役に立ちました。

メインゲーム画面

・各プレイヤースキル発動object

プログラム内容:

```
変数: • double 次のパネル入力待ち時間 / • int 入力パネルカウント
      • bool パネル入力待ち
      • bool スキル実行可能
関数: • 自分のキャラ側の一番端の下段パネルを押したら、「次のパネル入力待ち時間」中「パネル入力待ち」をtrueにして、その時間中、相手側隣のパネルを押したら、「次のパネル入力待ち時間」中「パネル入力待ち」をtrueにして、その時間中、前に押したパネルの相手側隣のパネルを押したら、自身の「スキルチャージ変数」が100以上かつ「スキル実行可能」がtrueなら、自キャラに対応した「スキル関数」を実行し、自キャラの「攻撃モーション」と、相手キャラの「ダメージモーション」を実行。
もし、「次のパネル入力待ち時間」以内に次のパネルを押せなかったら、「入力パネルカウント」と「パネル入力待ち」をリセットする
• 引数1つで、「スキル実行可能」falseにして、引数分時間経過後に「スキル実行可能」をtrueにする
```

・各キャラスキルobject

プログラム内容:

```
変数: • double 妨害UI表示時間
      • float 加減速数値 / • double 加減速妨害時間
      • double 毒化妨害時間
      • double 判断妨害時間 / • double 判断妨害倍率
関数: • 相手レーン部分に「妨害UI」を表示し、「妨害UI表示時間」後の非表示にする。
「各プレイヤースキル発動object」の「スキルクールタイム関数」の引数に「妨害UI表示時間」を渡して実行
      • ランダムで加速か減速を決定し、スキル発動時「現在経過時間」 + 「加減速妨害時間」の時間以内に、「適正判定時間」を迎えるノーツには、「加減速数値」を参考に、speedの数値を変更する。
「各プレイヤースキル発動object」の「スキルクールタイム関数」の引数に「加減速妨害時間」を渡して実行
      • 「現在経過時間」 + 「加減速妨害時間」の時間以内に、「適正判定時間」を迎えるノーマルノーツには、
```

プロジェクト引き継ぎの際の事前書類

プログラムを本格的に始める前に、制作途中のトラブルを少なくしてより効率の良い方法を整理するために制作いたしました。
仕様書をしっかり読み込み、細かく要素出しをした結果、普段以上に先を見通しながら細かい部分への配慮ができ、至らぬ点が多いながらもスムーズなプロジェクト進行にてても役立ちました。



ゲームプログラム

ノーマルノーツ判定処理

菅原 渉

使用ソフト / Unity 2021.2.15f1 , Visual Studio 2022

制作期間 / 1 時間

Assembly-CSharp

```
432 // /// <summary> 判定を元に、スコアの加算やエフェクトなどの判定の処理を行う
433 // 2 個の参照
434 private void Judge(long time, int line, int usePlayer)
435 {
436     // Brilliantの判定時間 / スキルによる倍率以下なら
437     if (time <= brilliantJudge / judgeLeverage[usePlayer])
438     {
439         OtherThanPoorJudge(line, usePlayer, (int)JudgeType.Brilliant);
440         NotesJudgeEffect((int)JudgeType.Brilliant, line, usePlayer);
441     }
442     // Greatの判定時間 / スキルによる倍率以下なら
443     else if (time <= greatJudge / judgeLeverage[usePlayer])
444     {
445         OtherThanPoorJudge(line, usePlayer, (int)JudgeType.Great);
446         NotesJudgeEffect((int)JudgeType.Great, line, usePlayer);
447     }
448     // Goodの判定時間 / スキルによる倍率以下なら
449     else if (time <= goodJudge / judgeLeverage[usePlayer])
450     {
451         OtherThanPoorJudge(line, usePlayer, (int)JudgeType.Good);
452         NotesJudgeEffect((int)JudgeType.Good, line, usePlayer);
453     }
454     // poor判定
455     else
456     {
457         PoorJudge(line, usePlayer);
458     }
459 }
```

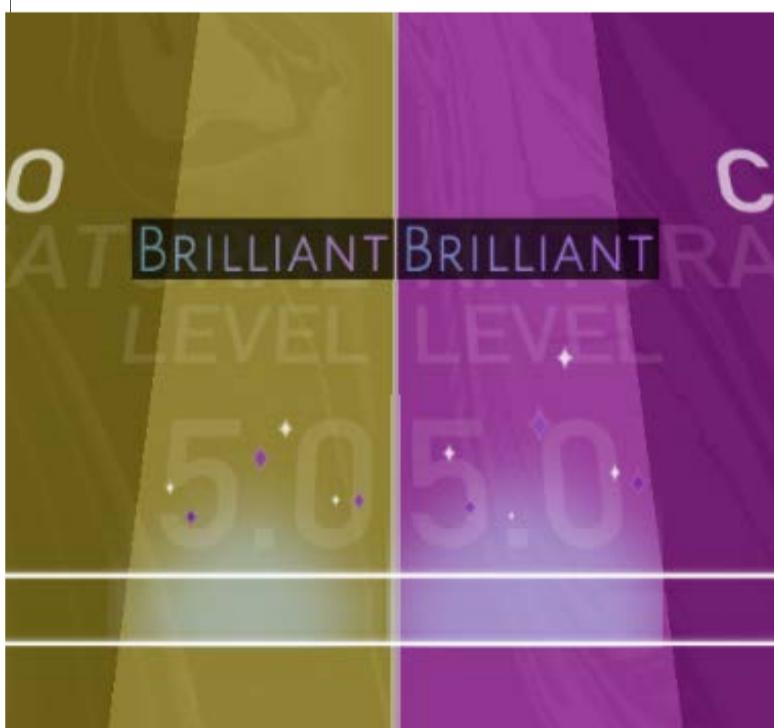
Sight andB / スキル表現時のーション

行: 478 文字: 5 S

ノーマルノーツ判定処理

ノーマルノーツとホールドノーツで使用する判定処理です。渡された time と、各判定に決められた値を比べて判定しています。

各種ノーツで使用するために、共通の処理をまとめたものになっています。各種ノーツごとの特殊な処理は、この関数を呼び出す関数にまとめてあります。



最高判定の BRILLIANT は
派手な文字と共に、虹色に光る
エフェクトが表示される

GOOD より上位の GREAT 判定は
キャラクターカラーの
エフェクトが表示される





ゲームプログラム

ノーマルノーツ判定分岐処理

菅原 渉

使用ソフト / Unity 2021.2.15f1 , Visual Studio 2022

制作期間 / 30 分 ~1 時間

PlayerInput.cs

```
1435 // <summary> ノーマルノーツの判定処理先の分岐
1439 3 個の参照
1440 void SimpleNotesJudge(int line, int usePlayer)
1441 {
1442     // タンパリン
1443     SEPlayer.instance.SEOneShot(1);
1444 
1445     // ノーツが poison ノーツかそうでないかで、判定を変える
1446     if (nextJudgeNotes[line].isPoison)
1447     {
1448         // 判定用関数で、poison ノーツの判定をする
1449         notesJudge.PoisonNotesJudgement(
1450             // criwear の時間 - 判定バー到達予定時間 を絶対値化して引数に設定
1451             (long)Mathf.Abs(nextJudgeNotes[line].time - MusicData.Timer), line, usePlayer);
1452     }
1453     else
1454     {
1455         // 判定用関数で、ノーマルノーツの判定をする
1456         notesJudge.SimpleNotesJudgement(
1457             // criwear の時間 - 判定バー到達予定時間 を絶対値化して引数に設定
1458             (long)Mathf.Abs(nextJudgeNotes[line].time - MusicData.Timer), line, usePlayer);
1459     }
1460 
1461     // 干渉するノーツを次にする
1462     lineNotesNum[line]++;
1463     // 次のノーツデータを格納
1464     SetNextNotesData(line);
1465 }
```

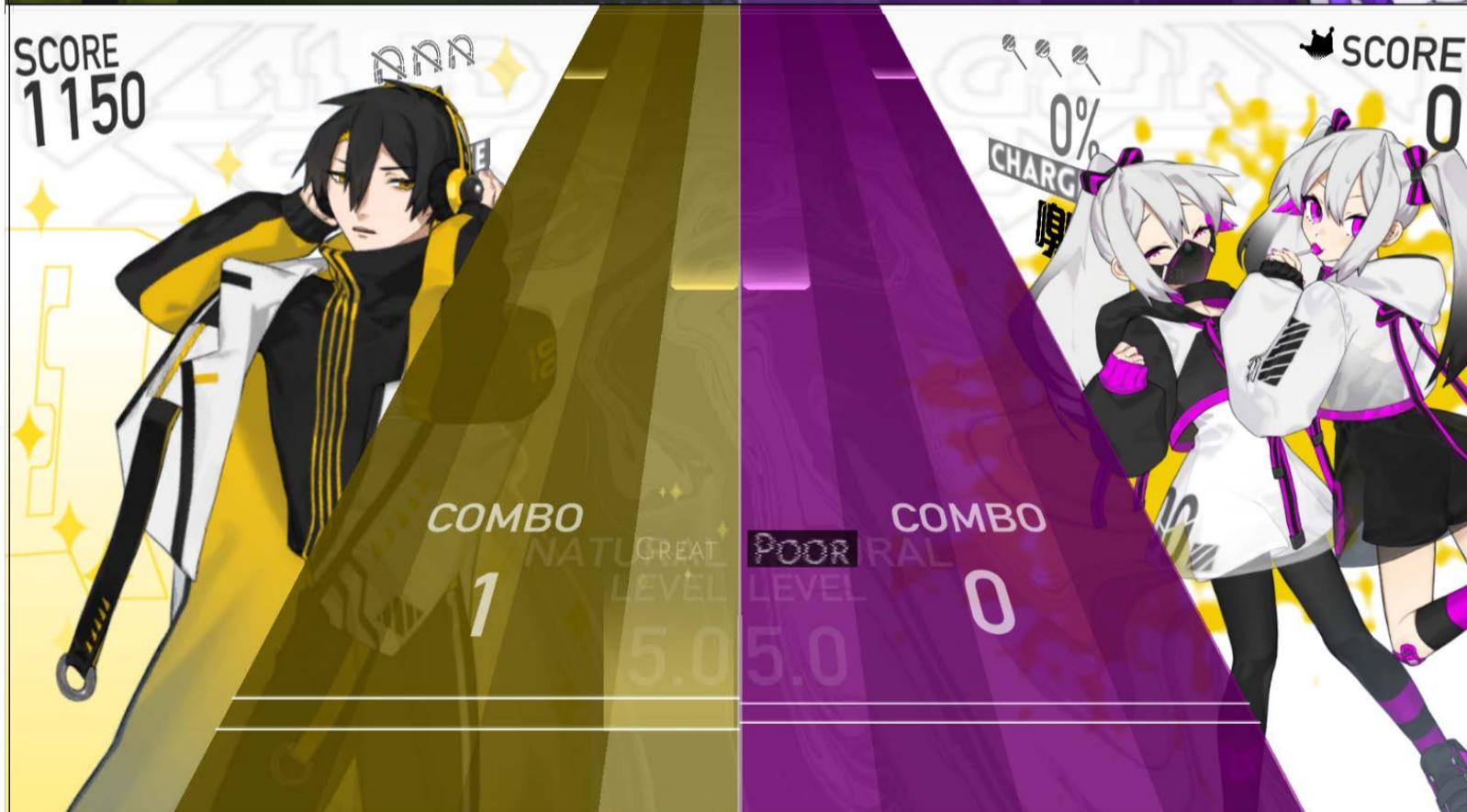
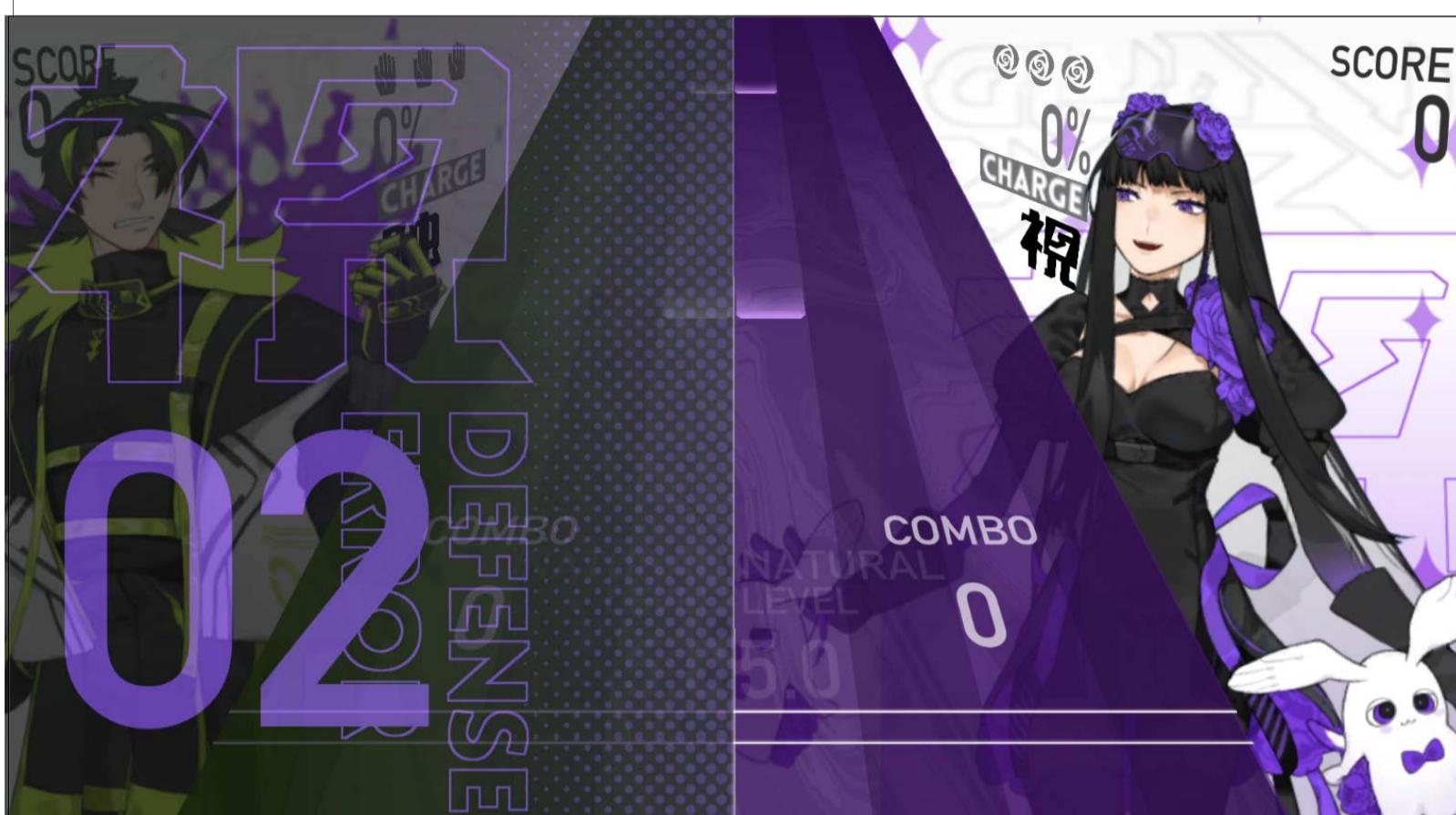
ノーマルノーツ判定分岐処理

プレイヤーの入力から、ノーマルノーツの判定先を決めるスクリプトです。キャラクタースキルによって判定結果が変わる物がある為、その状態かどうかを判断し、それによって、呼び出す関数を変えております。

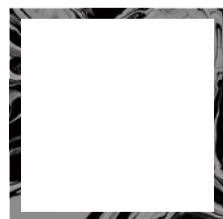
▼プレイヤー入力前（左側ノーツは poison ノーツ）

▼入力後。両ノーツ最高判定のタイミングだが、判定が逆になる
poison ノーツ側はミス判定処理がされている





[GitHub](#) ▶



プログラマー
スガハラ ワタル
菅原 渉



ゲームプログラム
譜面データ、譜面コンバーター
チンラクケン
使用ツール / Visual Studio Code
制作期間 / 1週間

ノーマルノーツ判定分歧処理

譜面データが JSON フォーマットで構成されていて、
楽曲データとノーツ生成時に必要なデータが含まれています。
プランナーが譜面制作ツールで制作した譜面を
JSON フォーマットに変換します。

Assets > Resources > BeatmapData > 02.json > songdata

```

1  {
2     "songdata": {
3         "name": "Liquid Dubstep",
4         "songnum": 1,
5         "artist": "k",
6         "bpm": 145,
7         "offset": 6100
8     },
9     "difflist": {
10        "natural": {
11            "level": 5,
12            "notelist": [
13                { "type": 0, "line": 2, "time": 6100 },
14                { "type": 0, "line": 2, "time": 6927 },
15                { "type": 0, "line": 2, "time": 7548 },
16                { "type": 0, "line": 1, "time": 9410 },
17                { "type": 0, "line": 1, "time": 10237 },
18                { "type": 0, "line": 1, "time": 10858 },
19                { "type": 0, "line": 2, "time": 12720 },
20                { "type": 0, "line": 2, "time": 13548 },
21                { "type": 0, "line": 3, "time": 14168 },
22                { "type": 0, "line": 1, "time": 16031 },
23                { "type": 0, "line": 0, "time": 16858 },
24                { "type": 0, "line": 3, "time": 17479 },
25                { "type": 2, "line": 1, "time": 19341, "endtime": 20789 },
26                { "type": 2, "line": 2, "time": 22651, "endtime": 24100 },
27                { "type": 0, "line": 2, "time": 25962 },
28                { "type": 0, "line": 2, "time": 26375 },
29                { "type": 0, "line": 1, "time": 26789 },
30                { "type": 0, "line": 2, "time": 27410 },
31                { "type": 0, "line": 2, "time": 28031 },
32                { "type": 0, "line": 1, "time": 28444 },
33                { "type": 0, "line": 1, "time": 29272 },
34                { "type": 0, "line": 2, "time": 29686 },
35                { "type": 0, "line": 2, "time": 30100 }
36            ]
37        }
38    }
39 }
```



ゲームプログラム
音源管理 (CRIWARE)
チンラクケン
使用ツール / Visual Studio Code , CRI ADX LE , CRI Atom Craft
制作期間 / 1週間

音源管理 (CRIWARE)

ゲーム内の音源を管理するため CRIWARE を導入しています。
CRI Atom Craft でゲーム内の音源の編集を行っています。

Atom Browser

Path to Search	D:/Tech... Reload Info
ACF Files	BGMData/BGM.acf
Cue Sheet	
BGMcue	
PreviewCue	
ResultBGMcue	
SEcue	
TitleBGM	
tutorialCue	
CueSheet_Hear	
CueSheet_Sight	
CueSheet_Smell	
CueSheet_Tactile	
Cue Name	Cue ID
ChartreuseGreen	0
LiquidDubstep	1
RetroQuest	2
Soldier	3
Tutorial	4
Show Private Cue	Stop All
Cue Information	
Cue ID	0
Cue Name	ChartreuseGreen
User Data	
Create GameObject Add Component Update Component	

▲ CRIWARE(ゲーム内)

CRI Atom Craft LE Ver.3.46.02 (built on Jul 21 2021)

プロジェクトツリー

- CriAtomCraftTV3Root
 - BGM
 - ユーザー設定
 - 全体設定
 - DSP/FX設定
 - バスマップ
 - カテゴリ
 - REACT
 - ボリュームミキタグループ
 - ワークユニットツリー
 - ワークユニット
 - BGM
 - リファレンス
 - キュー・シート・フォルダ
 - BGM
 - BGMcue (5/5)
 - ChartreuseGreen
 - Track_Chartreuse Green
 - LiquidDubstep
 - Track_liquid dubstep
 - マテリアルツリー
 - BGM_Materialinfo
 - マテリアルルート・フォルダ
 - Chartreuse Green.wav
 - retroquest.wav
 - liquid dubstep.wav
 - soldier.wav
 - Tutorial.wav

タイムライン

インスペクター

Chartreuse Green.wav

名前 値

- 名前 Chartreuse Green.wav
- コメント
- ルード ピカルドに含めない False
- エコーコーナー・ファイル・サイズ 0
- ウェーブフォームリージョン 漂れループを無効化 False
- ループエンドで再生停止 False
- ボリューム 1.00
- ボリュームランダム幅 0.00
- EGアタック 0
- EGホールド 0
- EGディケイ 0
- EGラスティング 1.00
- EGリリース 0
- キー ポリューム / ピッチ * パスエンド ボイス エバロープ
- フィルター バ (B,A) 3Dオブジェクト テイリング セクター
- スタイル ラウド : フラッシュ ブースト Sync プロパティ
- ChartreuseGreen カテゴリ
- カタログリキューブライオリティ 値 0 タイプ 後着優先
- カタログリミット リミット数 1 タイプ
- カーラーデータ
- カーラーコメント

ログ

21:53:11.176 :読み込み時間 "0:00:00.091"
21:53:11.176 :プロジェクトの読み込みを完了しました。

インスペクター AISC ポイントリスト ワークユニットツリー REACT マテリアル 使用 ログ スタートページ

タイムライン キューキャッシュ (DSP/バス設定) リストエディター プロジェクトツリー マテリアルツリー パラメーターパレット レベルメーター ブックマーク タイムカーブ

◀ CRI Atom Craft



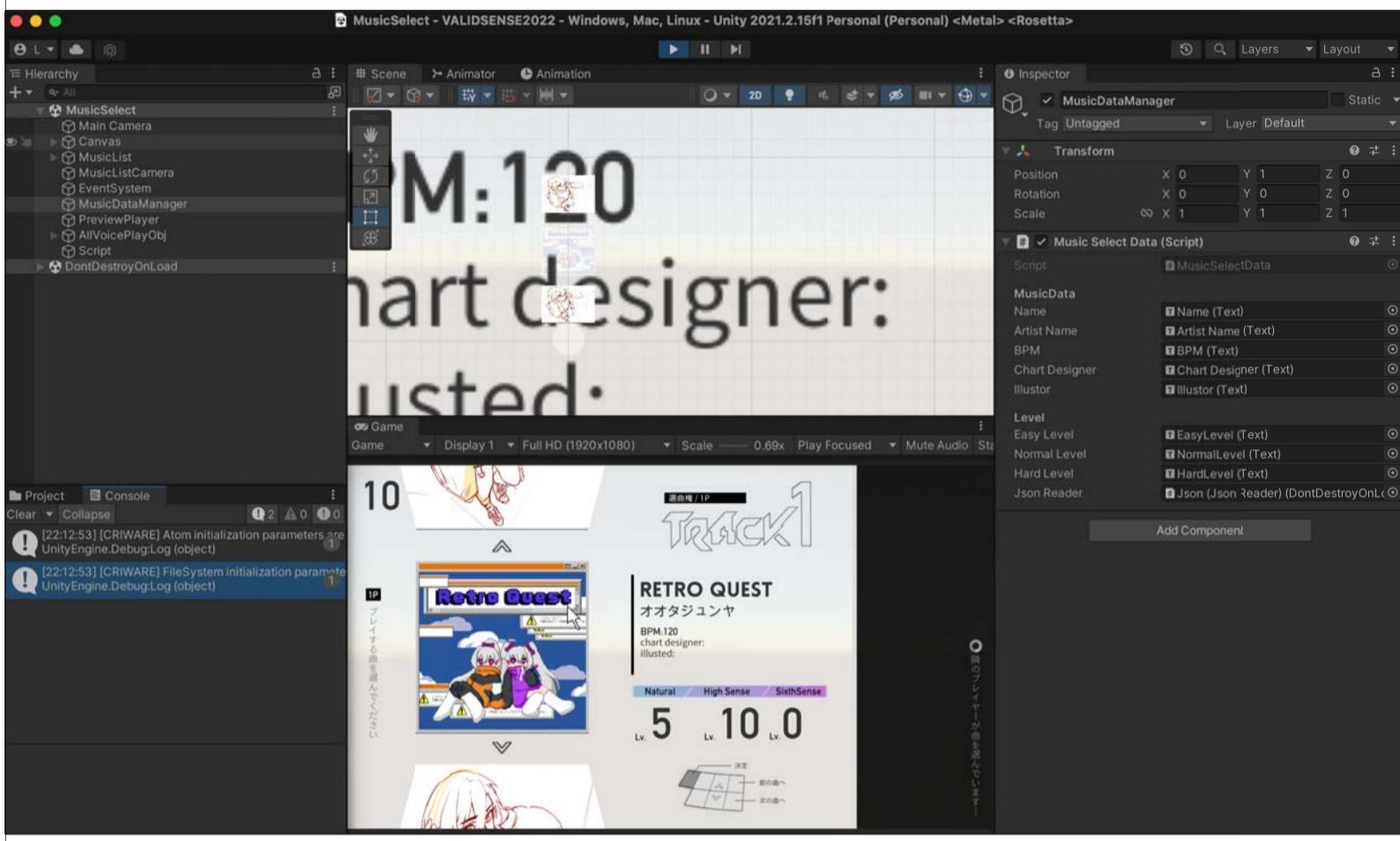
ゲームプログラム

曲選択画面

チンラクケン

使用ツール / Visual Studio Code , Unity 2021.2.15f1

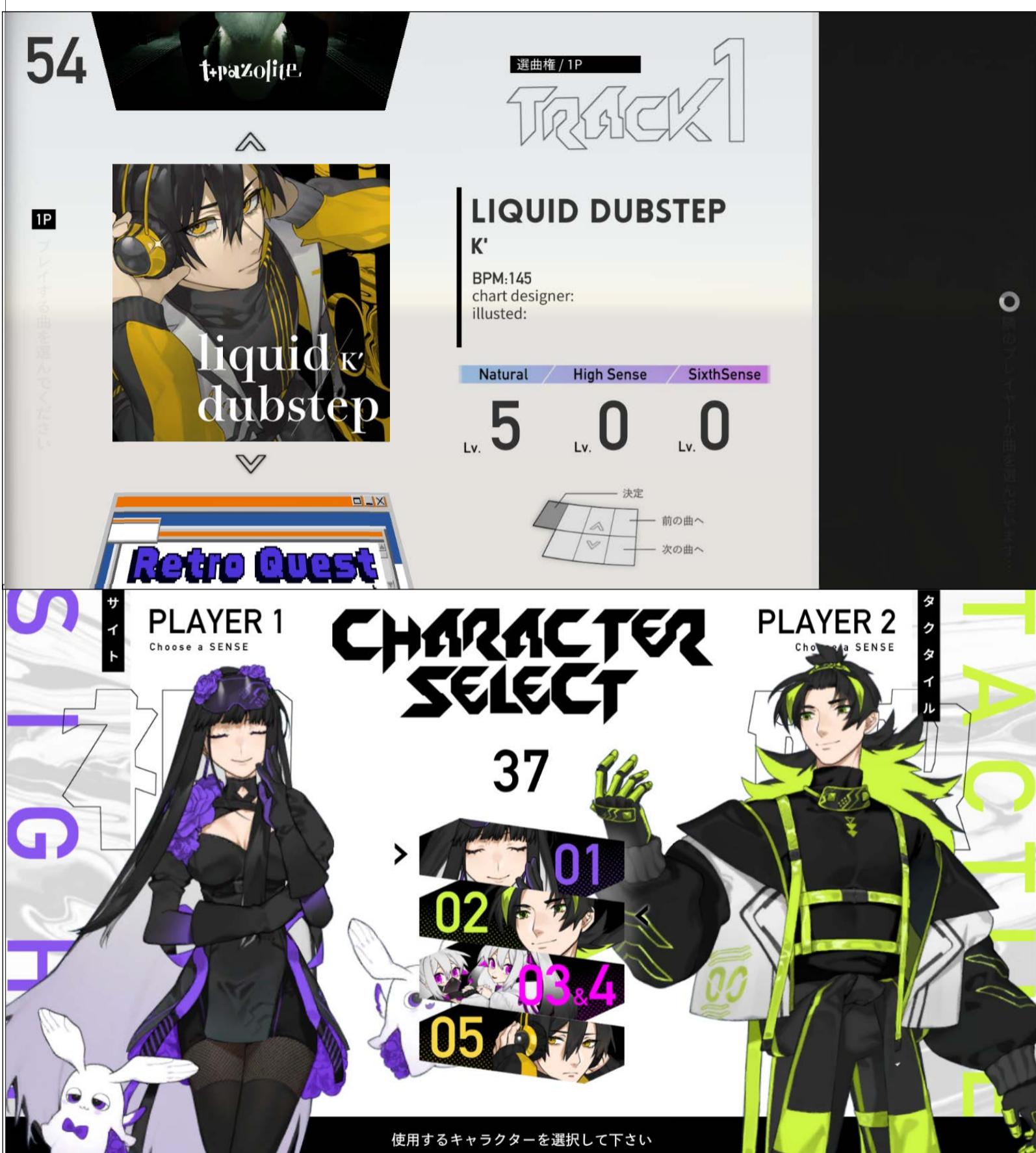
制作期間 / 1ヶ月



曲選択画面

曲選択画面のアニメーションのスクリプトです。
譜面データから楽曲データを取得しています。画面の表示は4曲基準で作られていますが、5曲以上でも正常に表示されるように調整しています。

```
71 private IEnumerator ScrollUp()
72 {
73     //スクロール可能の判定
74     if(isScrolling) { yield break; }
75
76     //スクロール開始
77     isScrolling = true;
78     ScrollUpNumChange();
79     //スクロールSE
80     SEPlayer.instance.SEOneShot(5);
81
82     //矢印を非表示
83     Arrow[0].SetActive(false);
84     Arrow[1].SetActive(false);
85
86     //スクロールアニメーション
87     yield return ScrollUpAnimation();
88
89     //矢印を表示
90     Arrow[0].SetActive(true);
91     Arrow[1].SetActive(true);
92
93     //曲データの表示変換
94     jsonReader.SendMessage("ChangeJson",nowMusicNum);
95     //曲を流す
96     MusicPreviewPlayer.instance.MusicPlay(nowMusicNum - 1);
97     //スクロール完了
98     isScrolling = false;
99 }
```



GitHub ▶



プログラマー

チン ラクケン

陳 樂謙



ゲームプログラム

難易度選択カーソル移動スクリプト

キムハクチャン

使用ソフト / Unity 2021.2.15f1 , Visual Studio 2022

制作期間 / 2時間～3時間

```
// Update is called once per frame
void Update()
{
    //1Pがボタンを押すとスクリプトが作動する
    if (canPlayer1Input)
    {
        if (Input.GetKeyDown(KeyCode.Q) || Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.E) || Input.GetKeyDown(KeyCode.R)
            || Input.GetKeyDown(KeyCode.S) || Input.GetKeyDown(KeyCode.D) || Input.GetKeyDown(KeyCode.F))
        {
            player1_IsMove = true;
            SEPlayer.instance.SEOneShot(6);
        }
    }

    //エントリー画面がspeed*時間の速度だけ下がる
    if (player1.transform.localPosition.y > -50 && player1_IsMove)
    {
        player1.transform.localPosition -= new Vector3(0, speed * Time.deltaTime, 0);
    }

    //エントリーが画面外に出ないように調整
    else if (player1_IsMove)
    {
        player1_IsMove = false;
        player1.transform.localPosition = new Vector3(player1.transform.localPosition.x, -50, 0);
    }
}
```

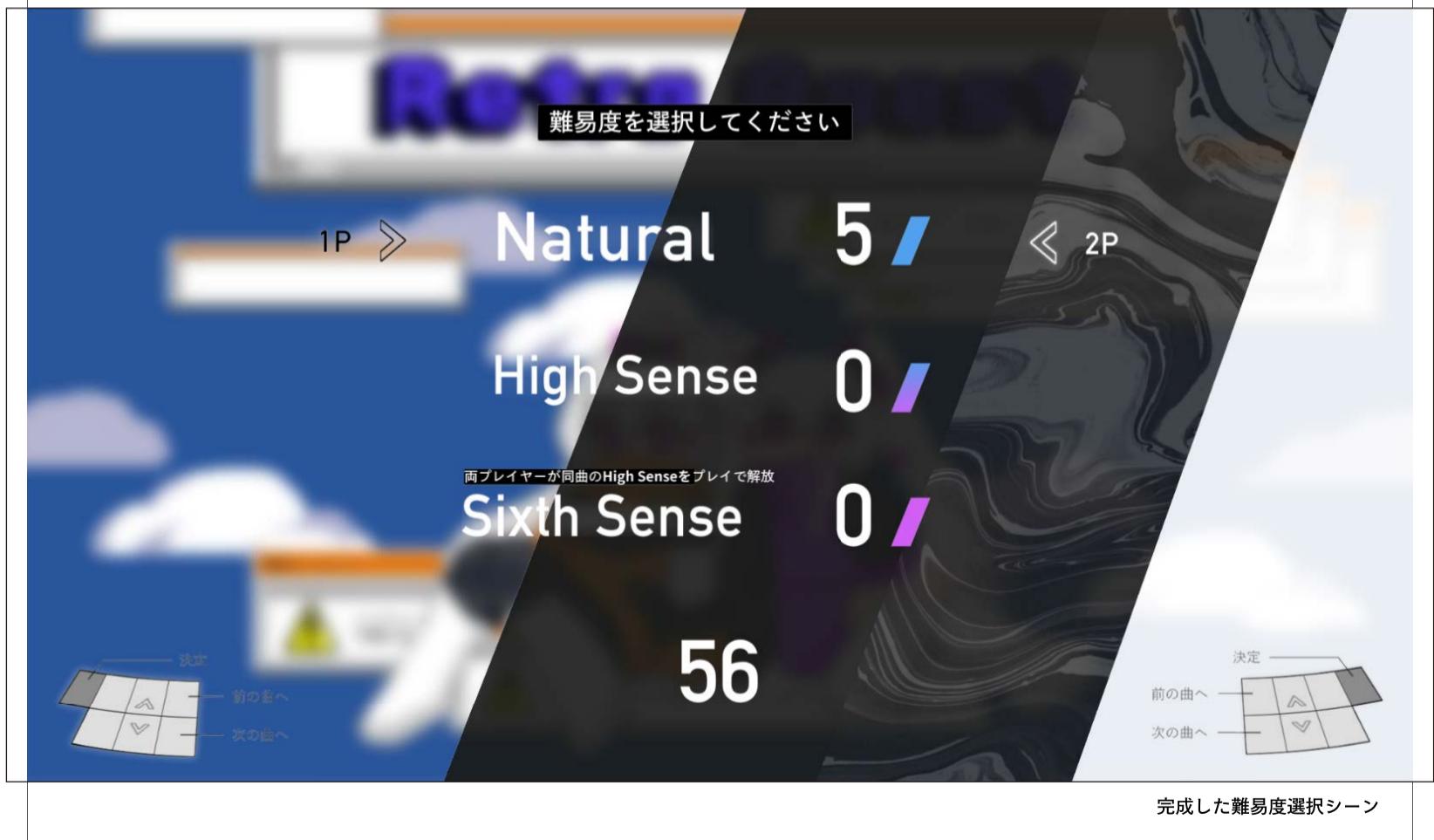
Sight and / スキル実験時モーション

難易度選択カーソル移動スクリプト

プレイヤーが難易度選択する時作動するスクリプトです。

決定のボタンを押す前には上、下で難易度選択が可能です。

決定したらカーソルは動きません。



完成した難易度選択シーン



ゲーム紹介 PV

PV 制作

杉森 風太

使用ソフト / Adobe After Effects

制作期間 / 3 時間

Youtube ▶



再生はこちら

The screenshot shows the Adobe After Effects interface with a project titled "VALID_SENSE_2022_PV_1.0". The composition panel displays a character with long black hair and purple flowers. The properties panel on the right shows various settings for the selected layer, including transform, color, and effects. The timeline at the bottom shows keyframes for different layers over time.

ゲーム紹介 PV

個人的にもグリッジ関連のトランジションなどはよく作るので、自分とは相性が良かったと思います。





ゲーム UI エフェクト

MISS エフェクト制作

杉森 風太

使用ソフト / Adobe After Effects

制作期間 / 1 時間

Youtube ▶



再生はこちら



MISS エフェクト

コンボが途切れた際に表示されるエフェクト。
自分が日頃から使っているエフェクトパターンを使いました。



ゲーム UI エフェクト

スキル被弾時エフェクト制作

杉森 風太

使用ソフト / Adobe After Effects

制作期間 / 4 時間

Youtube ▶



再生はこちら

The screenshot shows the Adobe After Effects interface with a composition named 'render stardust'. In the Project panel, there are several layers including 'stardust birth', 'render stardust', 'clear mat', and 'mat'. The Timeline panel shows keyframes for the 'stardust' layer. The Effect Controls panel displays a complex node graph for the 'stardust' effect, starting with an 'Emitter' node which feeds into a 'Particle' node, then an 'Auxiliary' node, and finally another 'Particle' node which is connected to a 'Force' node.



スキル被弾時エフェクト

スキルが発動した際相手キャラクターの背後に表示されるエフェクト。
CC Mr. Mercury を使わず、Stardust で制作しました。



コントローラー

設計・制作

和田涼太

使用ツール / SolidWorks, Arduino IDE

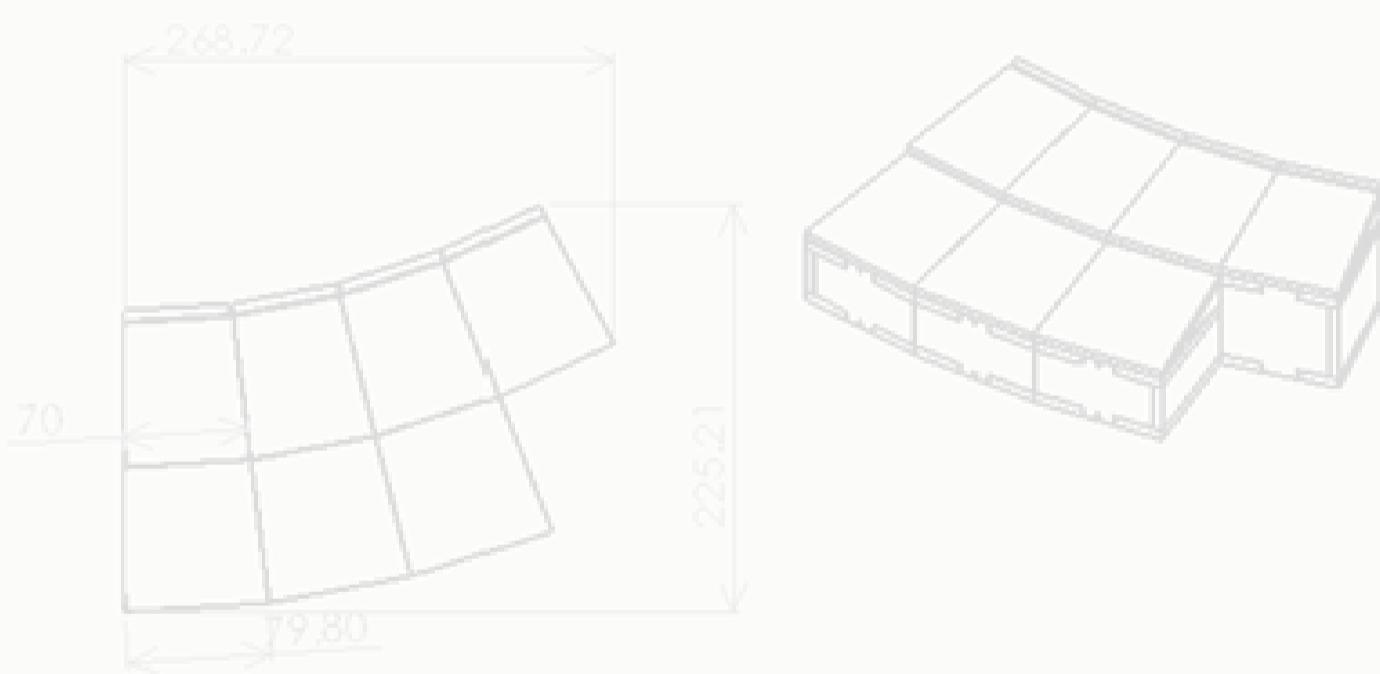
制作期間 / 設計 3ヶ月 組み立て 1週間 制御 1ヶ月



専用コントローラー

デザインの案をもらって、それを形にするのがとても大変でした。最終的にきちんと形になって動作するようになったので安心しています。

操作に合わせて LED が光るのもこだわったポイントです。



80



譜面制作

ゲーム収録楽曲譜面制作

和田涼太

使用ツール / PaletteWorks Editor

制作期間 / 1譜面 : 約4時間 ×12

The screenshot shows the PaletteWorks Editor software interface. On the left, there's a vertical toolbar with icons for different note shapes and a BPM selector. The main area is a grid-based music score editor with tracks labeled #22, #23, #24, and #25 on the y-axis. Each track contains horizontal bars of various colors (yellow, green, blue, red) representing musical notes. To the right of the grid are several panels: 'Control' (with playback buttons and scroll mode settings), 'Basic Information' (title, artist, composer fields), 'Statistics' (tap, flick, slide, hold, etc. counts), and 'Music Information' (file browser, volume sliders for master and SE).



PV動画コンテ

動画コンテ制作

和田涼太

使用ソフト / Adobe After Effects

制作期間 / 1週間

The screenshot shows the Adobe After Effects interface. The top bar has various tools and panels like Project, Composition, and Timeline. The main composition window displays a character with purple hair and a black outfit, standing next to two white bunnies. A text box with the Japanese characters 'ゲーム画面' (Game Screen) and 'サイトを映す' (Show Site) is overlaid on the screen. The bottom part of the interface shows the Timeline panel with multiple layers: 'サイトを映す' (Site to Show), 'サイト' (Site), 'Sight,Choice.wav', 'Sight,Joke.wav', 'ST,Choice.wav', and 'ST>Welcome.wav'. Each layer has its duration and position on the timeline.

The screenshot shows the Adobe After Effects interface again, but this time it's displaying a character selection screen. It features a character with purple hair and a black outfit, with the text 'CHARACTER SELECT' and '58' above them. Below the preview window is the Timeline panel, showing the same layers as the previous screenshot. The preview window also shows a small thumbnail of the character and the text 'CHARACTER SELECT'.

ゲーム収録楽曲譜面制作

カッコいいものができるか不安でしたが、最終的にとても良いものに仕上げてもらえたようで良かったです。

キャラクターが皆とても魅力的なので、PVにもぜひ注目してみてください。



イラスト発注書

ジャケットイラスト発注書制作
和田 涼太
使用ソフト / Excel

ジャケットイラスト発注書

楽曲を選択する際のアイコンの発注書です。色をカラーコードで指定したり、参考画像を添付したり、デザイナーが理解しやすいよう尽力しました。
とっても良いイラストを描いていただけて良かったです。

No.	名前	DX(横向き)	形式	横幅(X)/px	高幅(Y)/px	画素	所有者	権利
1	シャトルトリューパーズ チャートル・ジャケット	Chartruese_Green_jacket	PSD	2994	2994	350	右端部の際に表示される商品のアイコン。別途ジャケットに使用するライクです。	権利

「キャラクターの部分」と「曲名と作曲者名」と「背景」の部分をレイヤーを分けた状態で納品してください。全3つ。

参考

今回参考にしてほしい画像です。
トバゾライト氏のアルバム
「シャルトリューズグリーン」のジャケットです。
キャラの「横団」「ポーズ(腕組)」もそのまま、
背景の「檻の壊れた牢獄」「赤い垂れ幕」
の要素も用いて製作してください。
「檻の壊れ方」を、ヴァリッド・センスのロゴの形に
寄せてください。
完整性を保つようにではなく、やんわりと形が
似ているというようなイメージでお願いします。

曲名と作曲者名を画像内に乗せてください。
おおよそで構わないでの、下記のカラーコードを参考にしてください。

#7ff00
曲名「Chartreuse Green」
作曲者名「t+pazolite」

になります。そのまま使用してください。
フォントは、ポップで、硬くなりすぎず、可愛くてコミカルなものを使用してください。
→詳細は、城脇と相談し、最終決定をしてください。

右図のようなイメージで、
トバゾライト氏の位置に、タクタイルを
腕組みのポーズで配置し、
服装は「スーツ」にしてください。
尚、手の義手はそのまま着用させてください。

元画像のパンダの位置に、
マスコットキャラクター「モブ」
を参考画像のように配置してください。



◀完成品



ゲームプログラム

エントリースクリプト

キムハクチャン

使用ソフト / Unity 2021.2.15f1 , Visual Studio 2022

制作期間 / 1 時間

```
// Update is called once per frame
void Update()
{
    // If a button is pressed, the script runs
    if (canPlayer_Input)
    {
        if (Input.GetKeyDown(KeyCode.Q) || Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.E) || Input.GetKeyDown(KeyCode.R)
            || Input.GetKeyDown(KeyCode.S) || Input.GetKeyDown(KeyCode.D) || Input.GetKeyDown(KeyCode.F))
        {
            player1_IsMove = true;
            SEPlayer.instance.SEOneShot(6);
        }
    }

    // The entry screen moves down at speed * time
    if (player1.transform.localPosition.y > -50 && player1_IsMove)
    {
        player1.transform.localPosition = new Vector3(0, speed * Time.deltaTime, 0);
    }

    // Adjusts the entry screen so it doesn't go off the screen
    else if (player1_IsMove)
    {
        player1_IsMove = false;
        player1.transform.localPosition = new Vector3(player1.transform.localPosition.x, -50, 0);
    }
}
```

Sight and Skill Transition Motion

エントリースクリプト

プレイヤーが Key を押すとエントリー画面が作動するスクリプトです。
エントリーが画面外に出ないように調整するスクリプトも含まれています。



完成したタイトル・エントリーシーン



GitHub ▶



プログラマー

キム ハクチャン
金 學燦



ゲーム UI, グラフィックデザイン
ジャケットイラスト制作
田島 美於
使用ソフト / CLIP STUDIO PAINT PRO
制作期間 / 1週間

通常曲「Retro Quest」ジャケットイラスト
前作であるファイヤーウォール・インベーダーに
使用した楽曲のため世界観をそのままに、
スマイルとテイストを描きました。よく見ると、
立ち絵同様にテイストは飴を咥えています。



ゲーム UI, グラフィックデザイン
UI グラフィック制作
田島 美於
使用ツール / Adobe Illustrator
制作期間 / 1週間



キャラクタースキルアイコン
キャラクターのモチーフをもとにアイコンを作成しました。
キャラによって線のサイズが同じになるように調整しました。
シンプルですがパッと見てわかりやすいアイコンにしています。



クラウン

メインゲーム中、優勢な方に表示されるクラウンアイコンです。
王冠の形とハーフトーンのサイズを細かく確認して作成しました。





ゲーム UI エフェクト

画面遷移アニメーション制作

杉森 風太

使用ソフト / Adobe After Effects

制作期間 / 3時間

Youtube ▶



再生はこちら



画面遷移アニメーション

AE 的なカッコよさよりも、シュットとしたアニメーションを意識して作りました。



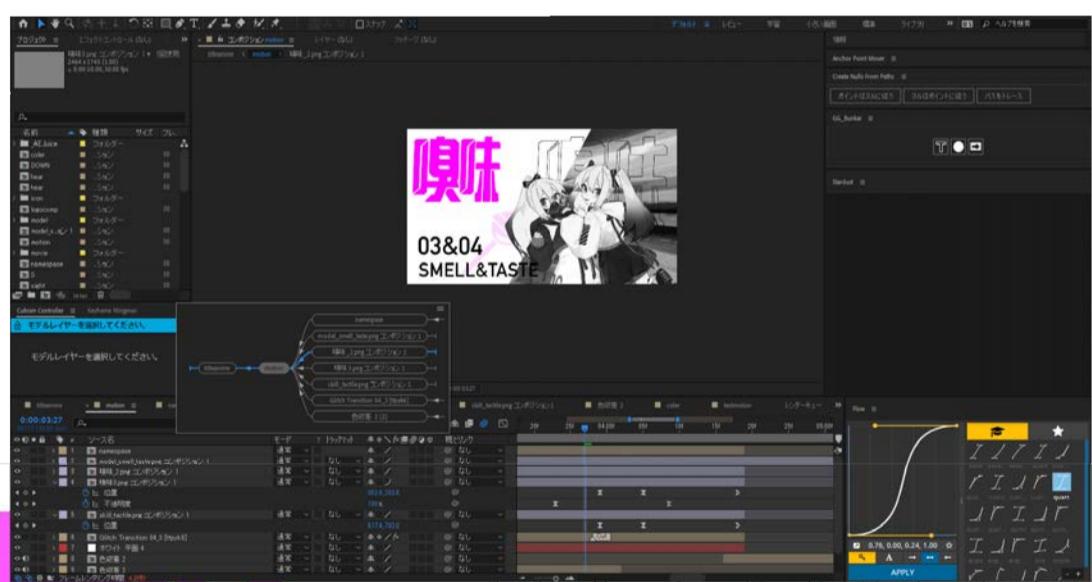
ゲーム内ムービー

タイトルロゴアニメーション

杉森 風太

使用ソフト / Adobe After Effects

制作期間 / 6時間



タイトルロゴアニメーション

リーダーから何度も

チェックバックをもらいながら
完成させました。

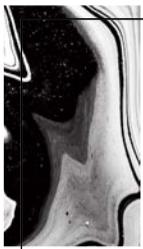
大変でしたが満足できる
ものができました。

Youtube ▶



再生はこちら





声優発注

台本制作
宮崎 千優
使用ソフト / Word
制作期間 / 2週間

1. タイトルの読み上げ

(「ワリッド・センス」ですが、読み方「バリッド・センス」にしてください。
ナレーションっぽくaidaさん)

001 サイズ

「対戦型音楽ゲーム「Wild Sense」
(サイトは紹介さん要を加えつつ、余韻にかつてよさが残るような爽やかな
感じで言つてください)」

002 タタタイフ

「対戦型音楽ゲーム「Wild Sense」
(設定のなんちやっぽを加えつつ、明るい感じでハキハキと。
余韻にかつてよさが残るような爽やかな感じで言つてください)」

003 ディスク&スマイル

「せーの!」
「対戦型音楽ゲーム「Wild Sense」
(小声でココソ話しているみたいにお願いします)」

004 スマイル

「性格を少し抑えつけて、人カバべを見ながら「せーの」で読んでいるような
可愛らしさがあると良いです)」

005 ヒア

「対戦型音楽ゲーム「Wild Sense」
(陰気っぽさを加えつつ、聞き取りやすいようにハキハキと。
余韻にかつてよさが残るような爽やかで言つてください)」

4

WILD SENSE

セリフ 台本

脚本制作を担当しました。声優を起用した
プロジェクトに携わるのは初めてで、大変な
こと多くありましたが、キャラクターに声が
ついた瞬間はとても嬉しかったです。

宮崎 千優 城脇 礼奈

WILD SENSE

タイトルロゴ

城脇 礼奈

使用ソフト / Illustrator
制作期間 / 4時間



ゲーム UI , ビジュアルデザイン

UI , ゲームビジュアルデザイン

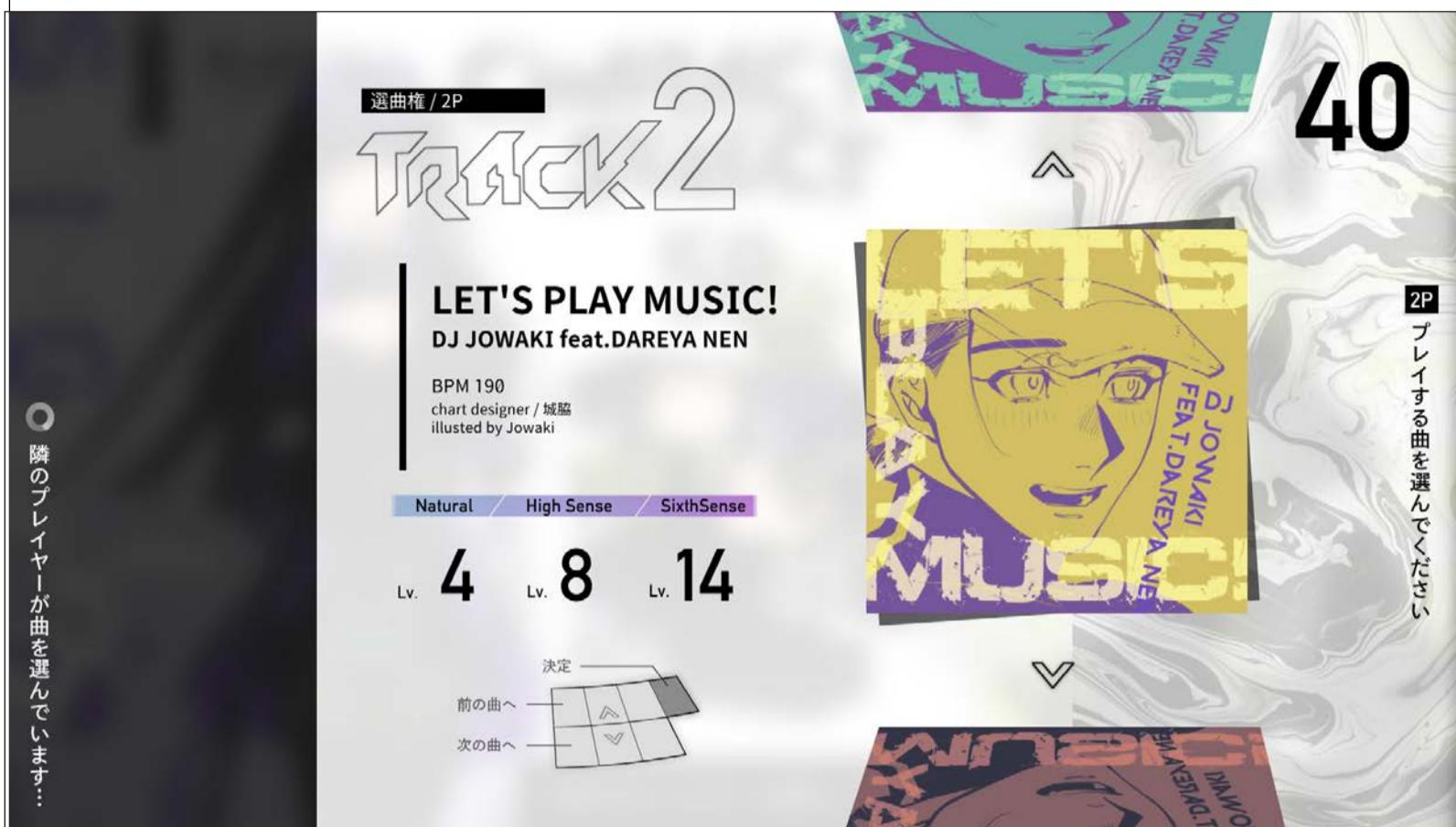
城脇 礼奈

使用ソフト / CLIP STUDIO PAINT PRO , Adobe Photoshop

各画面イメージ&UI制作

ゲームの各画面のビジュアルを全て担当し、個々の UI も制作しました。

最初の段階からイメージがあることで共有がしやすく、制作進行もスムーズにできたと思っています。



上：難易度選択画面 下：キャラクター選択画面 右上：難易度選択画面 右下：メインゲーム画面

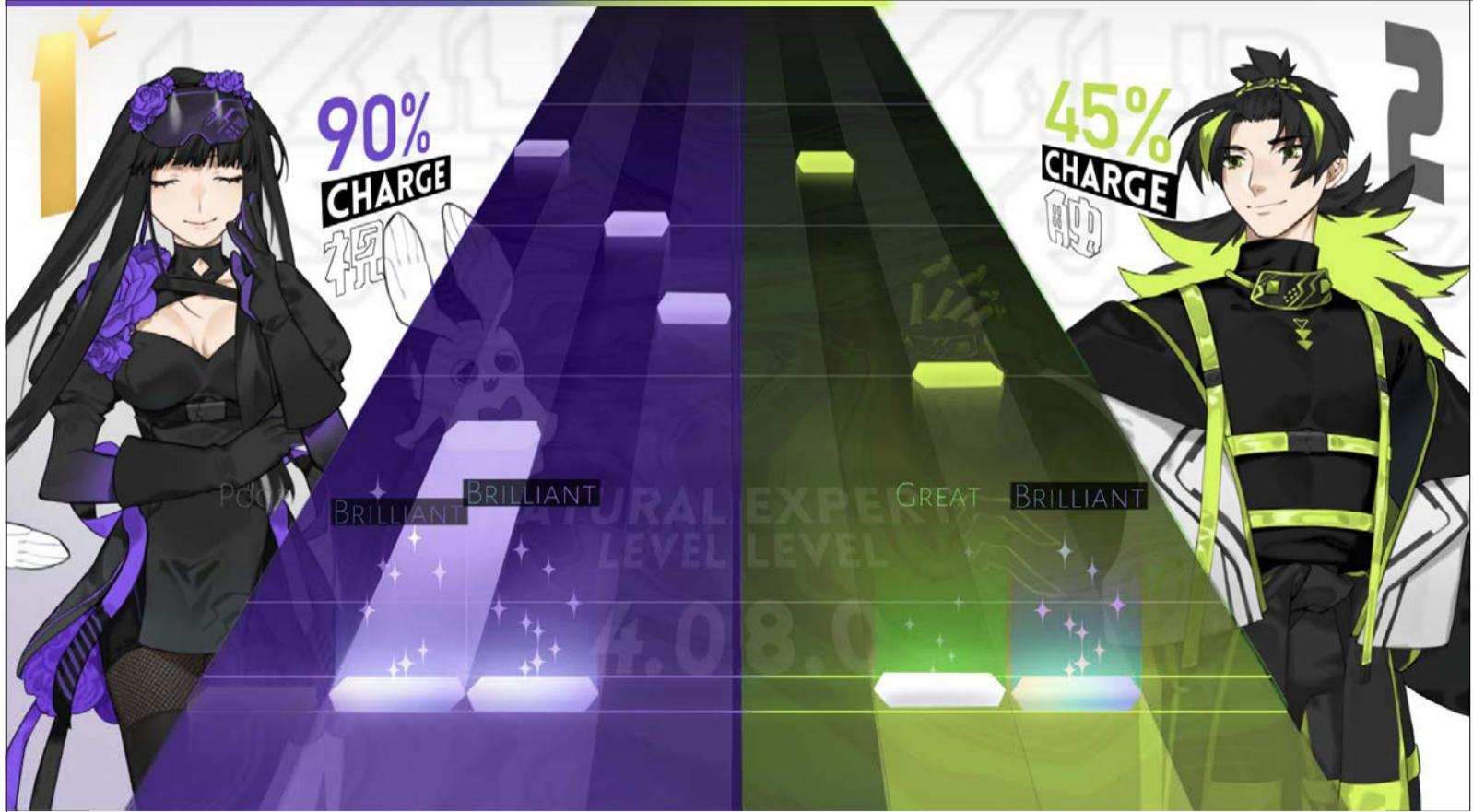
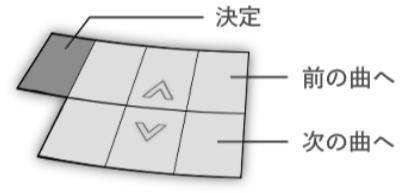


CHARACTER SELECT



> ○ > TRACK

Natural / High Sense / SixthSense





コントローラー

専用コントローラー内基板プログラム

和田 涼太

使用ソフト / Arduino IDE

制作期間 / 1ヶ月

専用コントローラー内基板のプログラム

LED テープの点灯処理とキーボード入力信号がそれほどラグを伴わずに反映することができたので安心しました。人の手が触れているかどうかの判定をする値を、自分で実験しながら割り出す必要があったので、それに苦労しました。

```

VS_Controller_Neo_Ver7_15
#include <CapacitiveSensor.h>
#include "Keyboard.h"
#include <Adafruit_NeoPixel.h>
const int LED_PIN = 13; // D13
const int LED_COUNT = 42; // LEDの数
Adafruit_NeoPixel pixels(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

/*
 * CapacitiveSense Library Demo Sketch
 * Paul Badger 2008
 * Uses a high value resistor e.g. 10 megohm between send pin and receive pin
 * Resistor effects sensitivity, experiment with values, 50 kilohm - 50 megohm. Larger resistor values yield larger sensor values.
 * Receive pin is the sensor pin - try different amounts of foil/metal on this pin
 * Best results are obtained if sensor foil and wire is covered with an insulator such as paper or plastic sheet
 */

CapacitiveSensor cs_2_3 = CapacitiveSensor(2,3); // 10 megohm resistor between pins 4 & 3, pin 2 is sensor pin, add wire, foil
CapacitiveSensor cs_2_4 = CapacitiveSensor(2,4); // 10 megohm resistor between pins 4 & 6, pin 6 is sensor pin, add wire, foil
CapacitiveSensor cs_2_5 = CapacitiveSensor(2,5);
CapacitiveSensor cs_2_6 = CapacitiveSensor(2,6);

CapacitiveSensor cs_2_0 = CapacitiveSensor(2,0);
CapacitiveSensor cs_2_9 = CapacitiveSensor(2,9);
CapacitiveSensor cs_2_10 = CapacitiveSensor(2,10);
//CapacitiveSensor cs_2_11 = CapacitiveSensor(2,11);

```



チュートリアルフラグリスト

フラグリスト制作

和田 涼太

使用ソフト / Excel

制作期間 / 3日

チュートリアルフラグリスト

ゲームのチュートリアルを構成するフラグリストです。細かい仕様をメンバーとすり合わせながら調整しました。最終的にはチュートリアルの動画コンテを作成し、プログラマーがタイミングを掴みやすいようにしました。

チュートリアル		1P側		2P側	
フラグ	起動タイミング	表示チケット	素材ID	フラグ	起動タイミング
チュートリアル開始	[tutorial] の文字とエフェクトが表示、チュートリアル開始			チュートリアル開始	[tutorial] の文字とエフェクトが表示、チュートリアル開始
ゲームシステム説明	フェードインで中央にテキストが③表示	このゲームは対戦型のリズムゲームです。操作は簡単で、音楽に合わせて手に多くの指で自分の色で塗っているプレイヤーの操作です。		ゲームシステム説明	フェードインで中央にテキストが③表示
プレイヤー1のレーンが発光	IPのレーンが発光し、3秒間説明が表示される。	これがIPの操作レーンです。選んだキャラクターによって色が異なります。		プレイヤー1のレーンが発光	IPのレーンが発光し、3秒間説明が表示される。
プレイヤー2のレーンが発光	2Pのレーンが発光し、3秒間説明が表示される。	これが2Pの操作レーンです。色が違いますね。		プレイヤー2のレーンが発光	2Pのレーンが発光し、3秒間説明が表示される。
操作条件の表示	操作条件が表示される	ノーマルノーツが発光されます。リーンが相手の色で塗られるときに合わせて手に多くの指で自分の色で塗っているプレイヤーの操作です。		操作条件が表示	操作条件が表示される
ノーマルノーツ説明	フェードインで中央にテキストが③表示	ノーマルノーツが発光されます。材料バーに合せて手を動かします。		ノーマルノーツ説明	フェードインで中央にテキストが③表示
ノーマルノーツ お手本フレイ	「3、2、1」のカウントの後自動でノーマルノーツがタップされる。	[お手本フレイ] 3,2,1		ノーマルノーツ お手本フレイ	「3、2、1」のカウントの後自動でノーマルノーツがタップされる。
ノーマルノーツ プレイ	Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!		Ready! 「3、2、1」のカウントの後自動でノーマルノーツがタップされる。 打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!
ホールドノーツ説明	フェードインで中央にテキストが③表示	ホールドノーツが発光されます。材料バーに合せて手を動かします。		ホールドノーツ説明	フェードインで中央にテキストが③表示
ホールドノーツ お手本フレイ	「3、2、1」のカウントの後自動でホールドノーツがタップされる。	[お手本フレイ] 3,2,1		ホールドノーツ お手本フレイ	「3、2、1」のカウントの後自動でホールドノーツがタップされる。
ホールドノーツ プレイ	Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!		Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!
スライドノーツ説明	フェードインで中央にテキストが③表示	スライドノーツが発光されます。材料バーに合せて手を動かします。		スライドノーツ説明	フェードインで中央にテキストが③表示
スライドノーツ お手本フレイ	「3、2、1」のカウントの後自動でスライドノーツがタップされる。	[お手本フレイ] 3,2,1		スライドノーツ お手本フレイ	「3、2、1」のカウントの後自動でスライドノーツがタップされる。
スライドノーツ プレイ	Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!		Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。3回成功するとクリアです。	Ready! 3,2,1 Nice!
リンクノーツ説明	リンクノーツについての説明と図が表示されます	リンクノーツの説明。キャラクターとリンクしてます。2人で最も合わせてタップ		リンクノーツ説明	リンクノーツについての説明と図が表示されます
リンクノーツ お手本フレイ	「3、2、1」のカウントの後自動でリンクノーツがタップされる。	[お手本フレイ] 3,2,1		リンクノーツ お手本フレイ	「3、2、1」のカウントの後自動でリンクノーツがタップされる。
リンクノーツ プレイ	Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。2回成功するとクリアになります。	Ready! 3,2,1 Nice!		Ready! 「3、2、1」のカウントの後プレイ! キャラクターが実際にプレイする。打てた場合は「Nice!」の文字が表示される。2回成功するとクリアになります。	Ready! 3,2,1 Nice!
キャラクタースキル説明	キャラクタースキルについての説明が表示される	ゲージがかかる。経済に消費になるスキルを使用できます。コンボヒートも減ります。		キャラクタースキルカウンター説明	IPのスキルが発動し、それに応じるカウントのチュートリアルが開始する。
キャラクタースキル使用1	Vボタンを押すとキャラクターのスキルを使用する。スキルは必ず命中する。命中! 表示時キャラクターは固定	命中!!		キャラクタースキルカウンター説明	IPのスキルが発動し、相手のスキンが変動します。
キャラクタースキル使用2	Vボタンを押すとキャラクターのスキルを使用する。スキルは必ずカウンターされる。命中! 表示時キャラクターは固定	命中!!		キャラクタースキル使用2	画面に[危険]が表示され、Vボタンを押すと2Pのスキルへのカウンターが必ず発動する。
キャラクタースキルカウンター説明	2Pのスキルが発動し、それに対するカウンターのチュートリアルが開始する。	危険!!		キャラクタースキル説明	キャラクタースキルについての説明が表示される。
キャラクタースキルカウンター説明1	2Pのスキルが発動し、音符を突ける。カウンターできない	今度はどちらかしら、スキルをカウンターしてしまましょう。危険!!		キャラクタースキル説明1	Vボタンを押すとキャラクターのスキルを使用する。スキルは必ず命中する。命中! 表示時キャラクターは固定
キャラクタースキルカウンター説明2	画面に[危険]が表示され、炎村特典Vボタンを押すと2Pのスキルへのカウンターが必ず発動	スキルは、Vボタンを押すと必ず命中します。「カウンター」できません。自身が不利になことがあります。		キャラクタースキル使用2	Vボタンを押すとキャラクターのスキルを使用する。使用後「命中」表示時キャラクターは固定
キャラクタースキル終了	チュートリアルが終了することを伝えるテキストが③表示	これでキャラクターは終了です。貴方のセンスを楽しんでいま		チュートリアル終了	チュートリアルが終了することを伝えるテキストが③表示されます。





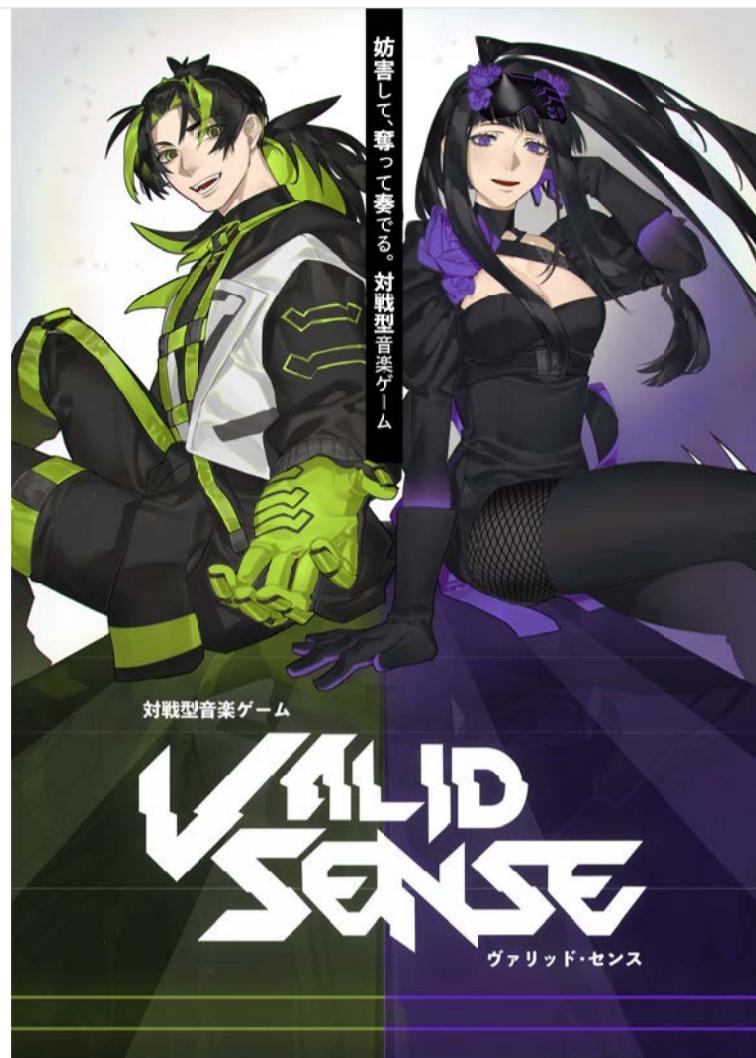
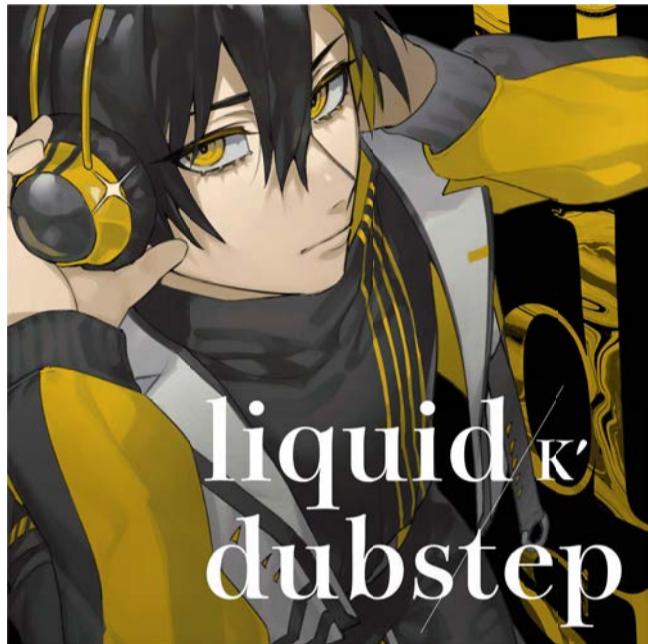
ポスター、イラスト

ポスター、ジャケットイラスト制作

城脇 礼奈

使用ソフト / CLIP STUDIO PAINT PRO

制作期間 / 6時間



『Liquid Dubstep』 ジャケットイラスト

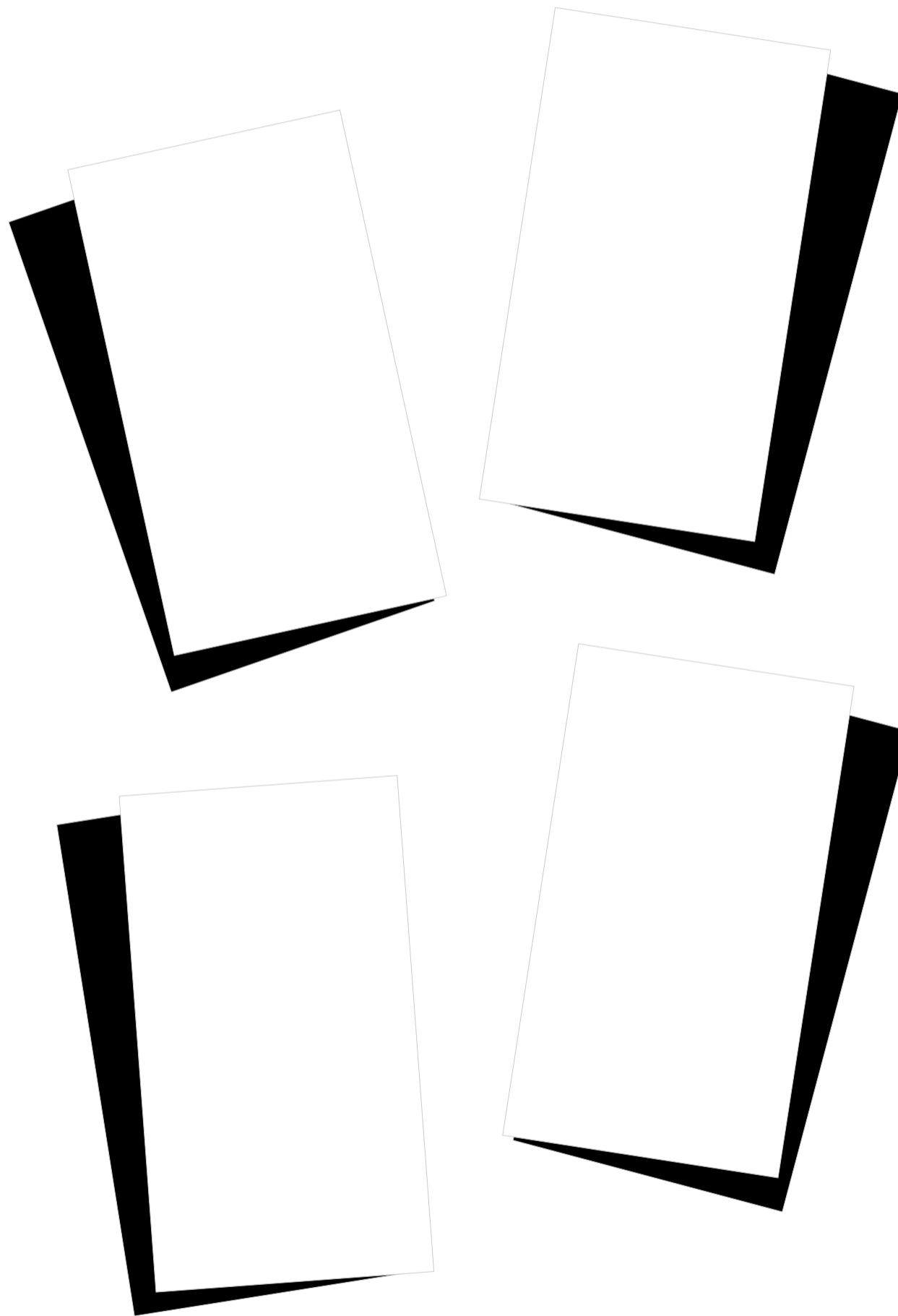
使用ソフト / CLIP STUDIO PAINT PRO

制作期間 / 4時間

**VALID
SENSE**



**VALID
SENSE**







**KILLID
SENSE**