**～レーシングゲーム～**

今回作ったのは、俯瞰視点からのレーシングゲームである。

周囲を杭で囲まれた広いフィールドにはチェックポイントが存在する。

その全てをできるだけ早く回り、タイムを競うゲームとなっている。

難易度は3種類あり、チェックポイントの数・配置が異なっている。

一度出したハイスコアはリトライしても残っており、

自分の記録を塗り替えていくチャレンジをすることも可能だ。

**実行方法**

車の画像、車の影の画像、背景の画像、タイトル画像、メーター画像の入った

imgフォルダをクラスファイルと同じディレクトリにおき、プログラムを実行する

**操作方法**

～タイトル画面～



上下キーまたはマウスのクリックで

難易度のボタンを選択

Zキーで決定

～ゲームクリア画面～

スペースキーでリトライ

エンターキーでタイトルに戻る

～ゲーム画面～



左右キーでハンドル

Zキーでアクセル

Xキーでブレーキ兼バック

スペースキーでリトライ

エンターキーでタイトルに戻る

**クラスの説明**

**Carクラス**

車の動き、描写をさせるクラスである

車の描写関連

・車を全方向にむかせて操作するために、画像の回転を扱う事の出来るGraphics2Dを導入した。

・車の画像の描写と同時に少しずれた位置に車の影の画像を描写することで車に立体感が生まれた。

・車のスピードが上がるにつれて描写位置を車後ろ方向にずらすことで、車を操作しやすくした。

車の挙動関連

・ブレーキと、バックを同じキーで担わせることでゲームとして操作しやすくした。

・左右キーを押すことで車の角度を変えていく方法をとった。なお、曲がる角度は速度に反比例するようにして、速いスピードでは曲がりにくくした。

・障害物や壁に当たった時の挙動として、いきなり止めるのではなく、急ブレーキをかけたように止まるよう、あたり判定を作成した

・スピードが出ている場合、アクセル、ブレーキいずれも踏んでいない状態でも段々とスピードが下がるようにしたことで動きに現実味を持たせた。

**Objectクラス**

フィールドの周囲を囲む杭の描写を行うクラスである

・杭はGraphics2Dを用いて、端の丸い太い直線にした。

・杭の長さ、角度は車の位置を使って、一点透視図法で計算し求めた。これによって

ゲーム画面の奥行きをより演出することができた。

下の図がその計算のメカニズムである。

birdeye:視点の高さ

leng:杭の高さ

k:車（視点）の中心から杭までの長さ

x:杭の一点透視上での杭の長さ

**Rockクラス**

フィールドの中に配置する、障害物としての杭の描写を行うクラスである

・杭の描写メソッドはObjectクラスをそのまま継承した。

・このクラスを作ることで複数の障害物を扱うことが可能となった。

（今回はフィールドの中心に１つ配置するだけにしている）

**Checkpointクラス**

ゲーム上で難易度を左右させるのがこのチェックポイントの数である。

このクラスでは描写、車が通ったかどうかを判定するメソッドが組み込まれている

・このクラスにはすでに通ったかどうかを確認するためboolean型でフラグを用意してある。この状態から、チェックポイントの色を変えることで視覚的に分かりやすくした

・描画色を半透明にすることで、背景との調和がとれ、見た目的なクオリティが高まった。

**Buttonクラス**

タイトル画面で難易度を選択するためのボタンの描写、マウスの判定を行うクラスである

・画面に円を描くことという点で類似していることからCheckpointクラスを継承した。

実際にはx,yの要素を引き継いで使うだけにとどまった。

・ボタンの色は難易度それぞれに違う色を割り当てるため、クラス内で値は入れなかった。

・ボタンの選択状態で透明度を切り替え、３つのボタンのうちどれが選択されているか分かりやすくした

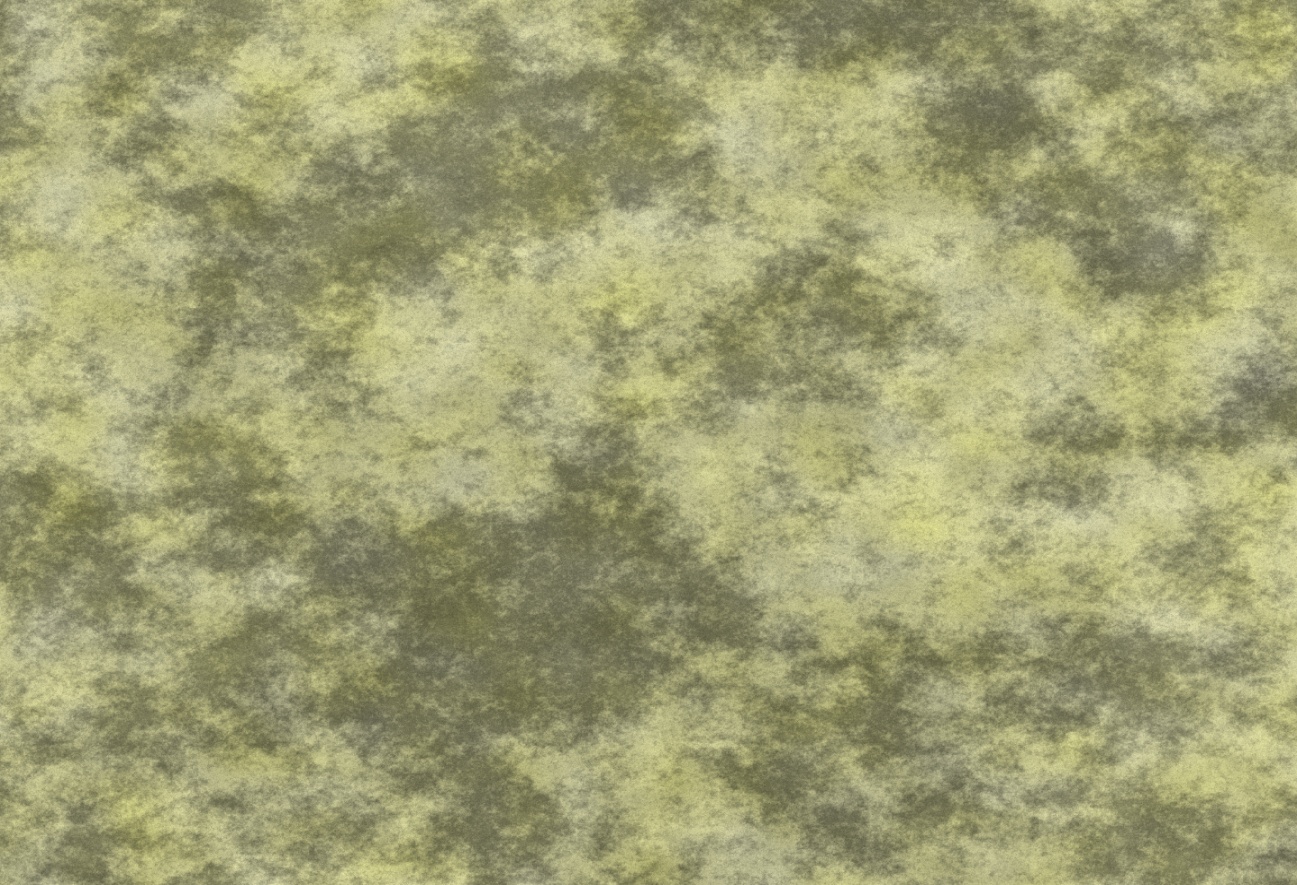
**RacingGameクラス（メインクラス）**

ゲーム進行に必要なすべての処理を行っている。

このゲームの描画の特色として、アプレットとフィールドのサイズの関係がある。

車が動ける全体のフィールドを一画面に描写し、その中で車を動かすのではなく

車を画面の中心に大きく描き、背景を車の動きに合わせて動かすことで

上空からカメラで撮っているような構図となり

ゲームをする上で車を運転している

という感覚を高めることができる。

アプレットサイズは480×360

ゲームでの背景画像サイズは2880×1960

（実際に車が走ることのできるフィールドは2400×1600）

となっており、描画をする際にアプレットに対してマイナス方向に背景画像の描写開始地点を動かすことでこれを実現している。

その他、車以外の杭やチェックポイントなどの描画もすべてこの考えに則っている。

よって、すべての描写は車のフィールド上での位置に左右されるため、描画処理には引数として車の座標を渡している。

**描画関連**

**タイトル画面**

・タイトルメニューの背景としてゲームの背景画像を動かして描写することで、タイトル画面に動きが生まれた

・ハイスコアを難易度別に一覧表示することで、自分の最高記録を分かりやすくした

・描写する文字のほとんどを画像として読み込むことで凝った演出ができた

**ゲーム画面**

・ミニマップを導入し、フィールド上での車の位置、障害物の位置、チェックポイントの位置・状態を描写することにより、今回のようにフィールド全体が見えない方式のレースゲームでも、遊びやすくなった

・ゲーム画面のアクセントとして、車のスピードを表示するメーターを描写した。

・画面の中心からまだ回っていないチェックポイントまでの方向を矢印として描写することで、ゲームのテンポを飛躍的に向上させることができた

・ゲームのUIとして、画面の４隅に必要な情報を配置することで、操作の邪魔にならないよう留意した。

**ゲームクリア画面**

・最後のチェックポイントを通った時、通った瞬間にクリア画面に移行してしまうのではなく、そのままブレーキがかかり、止まったところでクリア画面に移行するようにすることで動きに現実味を持たせた。

・ハイスコアを更新したときだけNew Record!!という表示をすることで、ハイスコアの更新への意味を高めた。

**内部処理関連**

このゲームでの難易度設定は、チェックポイントの数と配置によって行われるが、

プログラムではtxtファイルからチェックポイントの数、座標を読み込めるようにした。

今回、学内のパソコンで実行するにあたって、おそらくセキュリティ関連の問題からかtxtファイルを読み込むことはできなかったが、読み込むメソッドとしては完成したのでソースコードに残してある。

この方式をとることで、コードをいじらなくても難易度を変えることが可能となり、ゲームとしての将来性を持たせることができた。

**今回用いた画像について**

車の画像や、背景の砂漠風の画像をはじめすべての画像は自分で作成した。

タイトルのフォントに関してはPricedownフォントを用いた。

このゲームはRockStar Games社のGTA(Grand Theft Auto)シリーズをコンセプトとしている。