ゲームタイトル：「城を防衛シロ！！」

ファイルについて

- ExeFile フォルダ
  - CastleDiffender.exe：ゲームを実行できる exe ファイル
  - audio フォルダ：音声ファイルのフォルダ
  - pictures フォルダ：画像ファイルのフォルダ
  - savedata フォルダ：各ステージのハイスコアが書かれたファイルを保存するフォルダ
  - stageData フォルダ：各ステージのマップ情報が書かれたファイルを保存するフォルダ
- CastleDiffender フォルダ：ソースコードを保存するフォルダ
- CastleDiffender.sln：visual studio のソリューション
- Library フォルダ：DirectX のライブラリ用のフォルダ

開発について

開発期間：（2016 年 3 月 21 日～2016 年 4 月 20 日）約 1 ヶ月

開発人数：1 人（DirectX 以外のコードは全て自作です.）

自身の担当範囲：DirectX 以外のプログラム全般

動作環境：DirectX9 以上（Library フォルダからライブラリを参照するようにしているため，DirectX のインストールは必要ないと思われます）

起動方法：ExeFile フォルダ内の，CastleDiffender.exe が実行ファイルです．ゲーム中の操作方法については下記の「操作方法」項目に記載しています.

アピールポイント：下記の「アピールポイント」項目に記載しています.

開発者名：阿部拳之

使用言語：C++

使用 API：DirectX9

開発環境：Visual Studio 2015

ゲームルール

ステージ中央にある城に向かって攻めてくる敵を，勇者を操作することで倒して城を守りきりましょう！

城や勇者の体力が 0 になった場合はゲームオーバです.
一定数の敵を倒すとステージクリアとなり，残り時間をスコアとして競います.

ゲーム詳細

- ジャンルについて
  2D アクションとタワーディフェンスを混ぜたようなゲームです.
  ややこしい戦略なんて考えずに，アクションで敵を倒して城を守りぬきましょう.

- 勇者について
  プレイヤーは，勇者を操作することでゲームをプレイします.
  勇者は，移動や攻撃，必殺技（炎），ガードをすることができます.
  勇者は敵に攻撃されることでダメージを受けます．一定以上ダメージを受けるとゲームオーバとなってしまいます.
  自分や城を守りつつ敵を攻撃するして城を守りぬきましょう.

- 敵について
  敵は城を攻め落とすために軍勢となって攻めてきます.
  城に向かって進行してきますが，勇者やバリケードや城に近づくと，対象に向って攻撃をはじめます.
  敵の強さは全 3 種類で，色がノーマル→赤→青の順に強くなっています．また，それよりもさらに強い敵も存在します，攻撃は遅いですが威力や体力が高いので注意して戦うようにしましょう.

- マップについて
  ステージ中央には守る対象となる城があり，その周りには岩とバリケードが配置されています.
  敵と味方（勇者）関係なく岩に向かって進むことはできません.
  バリケードは勇者のみが通過することができ，敵はバリケードを攻撃して破壊するまではバリケードがある部分を通過することができません.

- 城について
  城はステージの中央に位置しています．このゲームの目標は，「敵から城を守りぬくこと」です.
  城は敵から攻撃を受けるとダメージを受けます．勇者同様，一定以上ダメージを受けるとゲームオーバとなってしまいます.

- バリケードについて

バリケードは敵に攻撃されることでダメージを受けます．一定以上ダメージを受けるとバリケードが破壊されてしまいます．
勇者がバリケードに攻撃した場合，バリケードを修理することができます．

- ステージについて
  ステージは，今回は全 3 ステージを用意しました．ステージ 1 が最も簡単で，ステージ 2→ステージ 3 と難しくなっていきます．
  各ステージをクリアするとステージの残り時間がスコアとして表示されます．ハイスコアを目指して頑張りましょう！

操作説明

- タイトル画面での操作について
  Z キー：ステージ選択へ
  X キー：操作説明へ
  E キー：ゲーム終了

- ステージ選択画面での操作について
  上下キー：ステージを選択
  Z キー：ステージを決定，ステージスタート
  X キー：タイトル画面へ

- ステージ中の操作について
  上下左右キー：移動
  A キー：攻撃
  S キー：必殺技
  D キー：ガード

- ゲームクリア時の操作について
  Z キー：ステージをリトライ
  X キー：タイトル画面へ
  E キー：ゲーム終了

- ゲームオーバー時の操作について

Z キー：ステージをリトライ

X キー：タイトル画面へ

E キー：ゲーム終了

アピールポイント

- 操作が簡単でテンポよく流れるゲームになっている点
- 衝突判定を極力つかわないようにして計算量を落とした点
  - 敵が城やバリケードと隣接しているかどうかを，マップ情報を利用することで判定するようにしています.
- 敵の AI の設計をランダム性を入れることでゲームバランスが取りやすい点
- ゲームエンジンのコードから全て自作をした点

プログラムについて

- ゲームエンジンのコード
  - winmain.cpp

    最も上の階層のクラスで，メインループを実行します.
  - graphics.cpp

    スプライトなどを描画するためのクラスです.

    描画に関する情報を扱ったり，テクスチャを読みこんだり，描画処理を行うことができます.
  - textureManager.cpp

    テクスチャを扱うクラスです.

    画像ファイルの読み込みなどを行います.
  - image.cpp

    画像を描画するクラスです.

    画像の描画やアニメーションの更新を行うことができます.

    ゲーム内のオブジェクトはすべて画像があると考えられるので，このクラスを継承することを推奨します.
  - entity.cpp

    エンティティ（実体）を表すクラスです.

    衝突判定に関する処理を行うことができます.

    Image クラスを継承しているので画像やアニメーションを持つ実体です.

このクラスを継承したクラスは体力を持ち，他のエンティティとの衝突判定を行うことができます.

- character.cpp

  Entity クラスを継承した，動作を行うキャラクターの基本となるクラスです.

  このクラスは継承することでのみ使用可能で，オブジェクトの生成はできません.

  Entity に向いている方向や状態（移動中，攻撃中など）を追加することによって，キャラクターの基本形を完成させています.

- audio.cpp

  オーディオを再生，停止するクラスです.

  メインクラスがこのクラスのオブジェクトを保持し，オーディオの切り替えを行います.

- Input.cpp

  ユーザーからの入力（マウスやキーボード）を処理するクラスです.

  このクラスがユーザーから入力を受け取ることによって，入力に応じた処理を行えるようになります.

- MessageDialog.cpp

  ユーザーへのメッセージダイアログを表すクラスです.

  予期せぬエラーが起こった場合などに，ユーザーへのメッセージを表示することができます.

  ゲームエンジンに組み込んでいますが，今回のゲームでは特に使用はしていません.

- InputDialog.cpp

  ユーザーの入力用のダイアログボックスを表すクラスです.

  ユーザーの入力した文字列などを表示することができます.

  ゲームエンジンに組み込んでいますが，今回のゲームでは特に使用はしていません.

- console.cpp

  デバッグ用のコンソールを表すクラスです.

  「`」で表示させることができ，特定のコマンドを打つことでゲーム内の状態を確認することができます.

- gameError.h

  ゲームエンジンによってスローされる Error クラスです.

  何らかのエラーが起こった場合，このクラスを通して Error が投げられます.

- dashboard.cpp

  ダッシュボード（棒グラフ，スイッチなど）を表すクラスです.

  このクラスを利用することによって棒グラフやスイッチをシステムグラフィックスとして使用することができます.

- rect.cpp

  四角形を描画するクラスです.

  四角形をシステムグラフィックとして使用することができます.

- text.cpp

  スプライトベースのテキストを描画するクラスです.

  スプライトベースのテキストを使用して文字列を表示することができます.

- textDX.cpp

  D3DXFont ベースのテキストを描画するクラスです.

  スプライトベースのテキストと違って，テキスト用の画像を用意する必要なしで使用ができます.

- game.cpp

  ゲームエンジンの基盤となるクラスです.

  ゲームのメインクラスはこのクラスを継承して作成する必要があります.

  ゲーム内のすべてのオブジェクトやシステムグラフィックスを扱う処理をします.

- 今回のゲーム独自のコード
  - brave.cpp

    プレイヤーが操作する勇者を表すクラスです.

    プレイヤーが操作をして移動，攻撃，必殺技，ガードを駆使して敵を倒します.

  - castle.cpp

    城を表すクラスです.

    敵によって攻撃され，体力が 0 になるとゲームオーバーです.

  - enemy.cpp

    雑魚敵を表すクラスです.

    バリケードを破壊しながら城に向かって進行します.

  - midBoss.cpp

    中ボスを表すクラスです.

    Enemy クラスを継承しているため，動作はほとんど雑魚敵と一緒です.

  - barricade.cpp

    バリケードを表すクラスです.

    城の周りに配置され，敵の攻撃によって破壊され，勇者の攻撃によって修理することができます.

  - map.cpp

    ステージ内のマップ情報を保持するクラスです.

    ステージ内の各グリッド毎に，城や岩，バリケードなど，どのオブジェクトがあるか（あるいはなにもないか）をマップ情報として所有します.

    マップ情報はステージごとに外部ファイルによって与えられます.

    マップ情報の詳細は map.h 内のコメントに示しています.

  - stage.cpp

    ステージ情報を管理するクラスです.

    勇者，敵，マップ情報，バリケード，城などステージに関する情報はここで管理します.

また，これらのオブジェクトの衝突判定処理やオブジェクトを 1 フレーム分更新するなどの処理も行います.

- fire.cpp

  勇者の必殺技によって発動される炎を表すクラスです.

  発射されると直進していき，敵にあたるか一定時間経つと消滅します.

- braveAttackCollision.cpp

  勇者による攻撃の当たり判定用のエンティティを出現させるクラスです.

  デバッグ時以外は不可視で，衝突判定のみ持ったエンティティです.

  このエンティティと敵やバリケードとの衝突判定が検知されると，対象はダメージを受けたり回復したりします.

- enemyAttackCollision.cpp

  敵による攻撃の当たり判定用のエンティティを出現させるクラスです.

  基本は BraveAttackCollision と同様ですが，勇者と敵との間に攻撃範囲の差別化を図るために分けて作成しています.

- castleDiffender.cpp

  ゲームのメインクラスです.

  1 フレームごとの更新や描画はこのクラスで行います.

- システムグラフィックスや画像を管理する今回のゲーム独自のコード
  - attackEffect.cpp

    勇者の攻撃時のアニメーションを表示するクラスです.

  - braveIcon.cpp

    勇者のアイコン画像を表すクラスです.

    勇者の大量に応じて表情が変化します.

  - castleIcon.cpp

    城のアイコン画像を表すクラスです.

  - hitEffect.cpp

    攻撃がヒットした時のアニメーションを表示するクラスです.

    攻撃がバリケードにヒットした場合などに発動します.

  - hpTextImage.cpp

    「HP」という文字の画像を表すクラスです.

    システムグラフィックスとして使用しています.

  - mpTextImage.cpp

    「MP」という文字の画像を表すクラスです.

    システムグラフィックとして使用しています.

- その他

- constants.h

  ゲーム内のパラメータや，画像・音声ファイル名などを定数として記述しています.

  いくつかのマクロ処理についても記述しています.

参考

音声ファイル元：「PANICPUMPKIN」

url：http://pansound.com/panicpumpkin/index.html

画像ファイル元：「VIPRPG 素材保管@wiki」

url：http://www54.atwiki.jp/viprpg_sozai/pages/1.html