using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Content;

using Microsoft.Xna.Framework.Graphics;

using Microsoft.Xna.Framework.Input;

namespace MarketsScreen

{

/// <summary>

/// This is the main type for your game.

/// </summary>

public class Game1 : Game

{

// Loading in the textures \\

private Texture2D Map;

private Texture2D Border;

private Texture2D Buttons;

private Texture2D Text;

private Texture2D Market1;

private Texture2D Market2;

GraphicsDeviceManager graphics;

SpriteBatch spriteBatch;

public Game1()

{

graphics = new GraphicsDeviceManager(this);

Content.RootDirectory = "Content";

// Making the mouse visible \\

this.IsMouseVisible = true;

//Changing the window width and height\\

graphics.PreferredBackBufferWidth = 450;

graphics.PreferredBackBufferHeight = 600;

graphics.ApplyChanges();

}

/// <summary>

/// Allows the game to perform any initialization it needs to before starting to run.

/// This is where it can query for any required services and load any non-graphic

/// related content. Calling base.Initialize will enumerate through any components

/// and initialize them as well.

/// </summary>

protected override void Initialize()

{

// TODO: Add your initialization logic here

base.Initialize();

}

/// <summary>

/// LoadContent will be called once per game and is the place to load

/// all of your content.

/// </summary>

protected override void LoadContent()

{

// Create a new SpriteBatch, which can be used to draw textures.

spriteBatch = new SpriteBatch(GraphicsDevice);

// TODO: use this.Content to load your game content here

// not used atm \\ market = Content.Load<Texture2D>("market.png");

Market1 = Content.Load<Texture2D>("MarketPart1.png");

Buttons = Content.Load<Texture2D>("ButtonMarkets.png");

Map = Content.Load<Texture2D>("Tryoutmap.jpg");

Text = Content.Load<Texture2D>("TryoutText.png");

Market2 = Content.Load<Texture2D>("MarketPart2.png");

Border = Content.Load<Texture2D>("borderflat.png");

}

/// <summary>

/// UnloadContent will be called once per game and is the place to unload

/// game-specific content.

/// </summary>

protected override void UnloadContent()

{

// TODO: Unload any non ContentManager content here

}

/// <summary>

/// Allows the game to run logic such as updating the world,

/// checking for collisions, gathering input, and playing audio.

/// </summary>

/// <param name="gameTime">Provides a snapshot of timing values.</param>

protected override void Update(GameTime gameTime)

{

if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed || Keyboard.GetState().IsKeyDown(Keys.Escape))

Exit();

// TODO: Add your update logic here

base.Update(gameTime);

}

/// <summary>

/// This is called when the game should draw itself.

/// </summary>

/// <param name="gameTime">Provides a snapshot of timing values.</param>

protected override void Draw(GameTime gameTime)

{

GraphicsDevice.Clear(Color.NavajoWhite);

// TODO: Add your drawing code here

spriteBatch.Begin();

// Creating the border beneath the initial market img \\

spriteBatch.Draw(Border, new Rectangle(-1, 70, 452, 6), Color.White);

// Creating the borders beneath the buttons img \\

spriteBatch.Draw(Border, new Rectangle(-1, 105, 452, 6), Color.White);

// Creating the borders beneath the map img \\

spriteBatch.Draw(Border, new Rectangle(-1, 350, 452, 6), Color.White);

// Creating the borders beneath the text img \\

spriteBatch.Draw(Border, new Rectangle(-1, 555, 452, 6), Color.White);

// <Creating the market1 img> \\

spriteBatch.Draw(Market1, new Rectangle(0, 0, 450, 71), Color.White);

// <Creating the buttons img> \\

spriteBatch.Draw(Buttons, new Rectangle(0, 75, 450, 31), Color.White);

// <Creating the map img> \\

spriteBatch.Draw(Map, new Rectangle(0, 110, 450, 240), Color.White);

// <Creating the text img> \\

spriteBatch.Draw(Text, new Rectangle(0, 355, 450, 200), Color.White);

// <Creating the market2 img> \\

spriteBatch.Draw(Market2, new Rectangle(0, 560, 450, 40), Color.White);

spriteBatch.End();

base.Draw(gameTime);

}

}

}