using Microsoft.Xna.Framework;

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

{

private Texture2D map;

private Texture2D border;

private Texture2D buttons;

// not used atm \\ private Texture2D market;

private Texture2D text;

private Texture2D market1;

private Texture2D market2;

GraphicsDeviceManager graphics;

SpriteBatch spriteBatch;

public Game1()

{

graphics = new GraphicsDeviceManager(this);

Content.RootDirectory = "Content";

//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");

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

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

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

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

market1 = Content.Load<Texture2D>("MarketPart1.png");

market2 = Content.Load<Texture2D>("MarketPart2.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(-5, 70, 500, 5), Color.White);

// Creating the borders beneath the buttons img \\

spriteBatch.Draw(border, new Rectangle(-5, 103, 500, 5), Color.White);

// Creating the borders beneath the map img \\

spriteBatch.Draw(border, new Rectangle(-5, 348, 500, 5), Color.White);

// Creating the borders beneath the text img \\

spriteBatch.Draw(border, new Rectangle(-5, 550, 500, 5), Color.White);

// <Creating the market1 img> \\

spriteBatch.Draw(market1, new Rectangle(0, -50, 450, 122), Color.White);

// <Creating the market2 img> \\

spriteBatch.Draw(market2, new Rectangle(0, 555, 450, 23), Color.White);

// not used atm \\ <Creating the market img> \\

//spriteBatch.Draw(market, new Rectangle(0, 0, 450, 120), Color.White);

// <Creating the buttons img> \\

spriteBatch.Draw(buttons, new Rectangle(0, 74, 450, 31), Color.White);

// <Creating the map img> \\

spriteBatch.Draw(map, new Rectangle(0, 108, 450, 240), Color.White);

// <Creating the text img> \\

spriteBatch.Draw(text, new Rectangle(0, 353, 450, 197), Color.White);

spriteBatch.End();

base.Draw(gameTime);

}

}

}