MONOGAME

Making games with C#

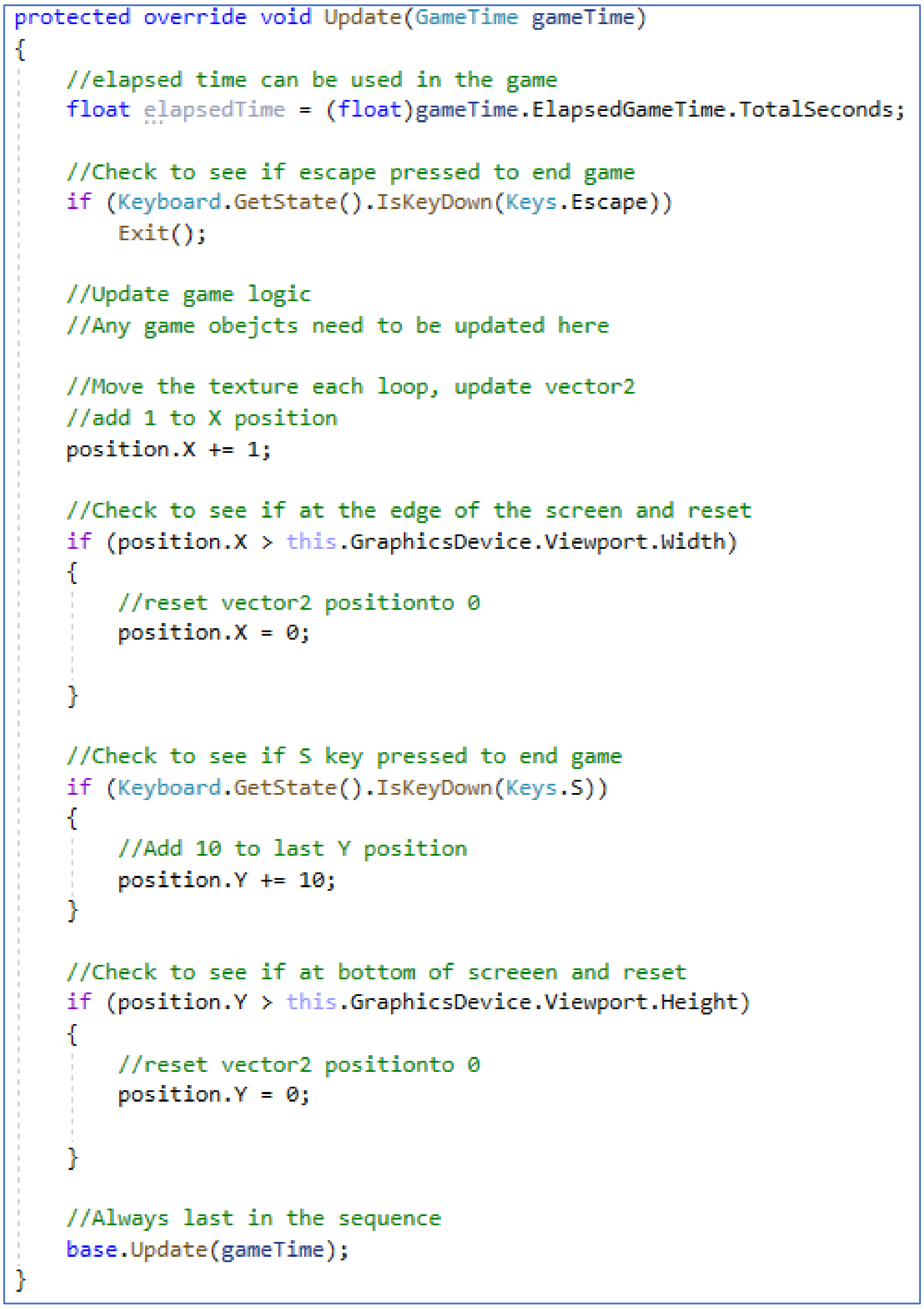
Getting user input

Getting the user to control items on the screen is an important part of any game, otherwise it is just an animation.

Adding keyboard input

Many games use keyboard input to control the characters, the code checks whether a key has been checked and performs an action.

Final Keyboard Update code



```
protected override void Update(GameTime gameTime)
{
    //elapsed time can be used in the game
    float elapsedTime = (float)gameTime.ElapsedGameTime.TotalSeconds;

    //Check to see if escape pressed to end game
    if (Keyboard.GetState().IsKeyDown(Keys.Escape))
        Exit();

    //Update game logic
    //Any game obejcts need to be updated here

    //Move the texture each loop, update vector2
    //add 1 to X position
    position.X += 1;

    //Check to see if at the edge of the screen and reset
    if (position.X > this.GraphicsDevice.Viewport.Width)
    {
        //reset vector2 positionto 0
        position.X = 0;

    }

    //Check to see if S key pressed to end game
    if (Keyboard.GetState().IsKeyDown(Keys.S))
    {
        //Add 10 to last Y position
        position.Y += 10;
    }

    //Check to see if at bottom of screeen and reset
    if (position.Y > this.GraphicsDevice.Viewport.Height)
    {
        //reset vector2 positionto 0
        position.Y = 0;

    }

    //Always last in the sequence
    base.Update(gameTime);
}
```

Keyboard input

Keyboard input is nice any easy, MonoGame provides everything that you need. The best place to do this is in the update method, as the keyboard status needs to be checked during the game loop.

The first step is to setup the class an object to capture the keyboard state using the **KeyboardState** class.

```
//check the status of current keyboard state
KeyboardState state = Keyboard.GetState();
```

The next step is to find out whether the key you wanted was pressed and make something happen. In this example, each time Y is pressed the Y position is increased by 10.

```
//Check to see if at the bottom edge and reset
if (position.Y > this.GraphicsDevice.Viewport.Height)
{
    position.Y = 0;
}
```

More logic required

A bit more logic is required in the Update() method now, as the texture could be moved off the bottom of the screen. The code is similar to before , but this time changing **position.Y** and checking the screen height.

```
//Check to see if at the bottom edge and reset
if (position.Y > this.GraphicsDevice.Viewport.Height)
{
    position.Y = 0;
}
```

Mouse input

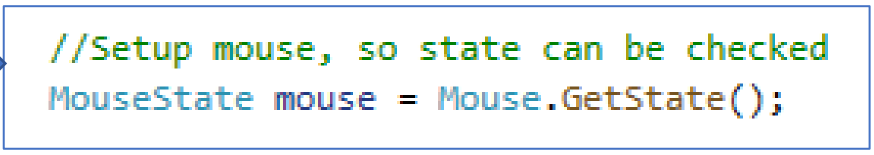
The keyboard is not the only way to get user input, there is also the mouse. This is very similar to using keyboard input.

Final Update code

```
protected override void Update(GameTime gameTime)
{
    //elapsed time can be used in the game
    float elapsedTime = (float)gameTime.ElapsedGameTime.TotalSeconds;

    //Check to see if escape pressed to end game
    if (Keyboard.GetState().IsKeyDown(Keys.Escape))
        Exit();

    //Update game logic
    //Any game obejcts need to be updated here

    //Move the texture each loop, update vector2
    //add 1 to X position
    position.X += 1;

    //Check to see if at the edge of the screen and reset
    if (position.X > this.GraphicsDevice.Viewport.Width)
    {
        //reset vector2 positionto 0
        position.X = 0;

    }

    //Check to see if S key pressed to end game
    if (Keyboard.GetState().IsKeyDown(Keys.S))
    {
        //Add 10 to last Y position
        position.Y += 10;
    }

    //Check to see if at bottom of screeen and reset
    if (position.Y > this.GraphicsDevice.Viewport.Height)
    {
        //reset vector2 positionto 0
        position.Y = 0;

    }

     //Setup mouse, so state can be checked
     MouseState mouse = Mouse.GetState();

     //Check if left mouse button pressed
     if (mouse.LeftButton == ButtonState.Pressed)
     {
         //Set X and Y to left click location
         position.X = mouse.X;
         position.Y = mouse.Y;
     }
```

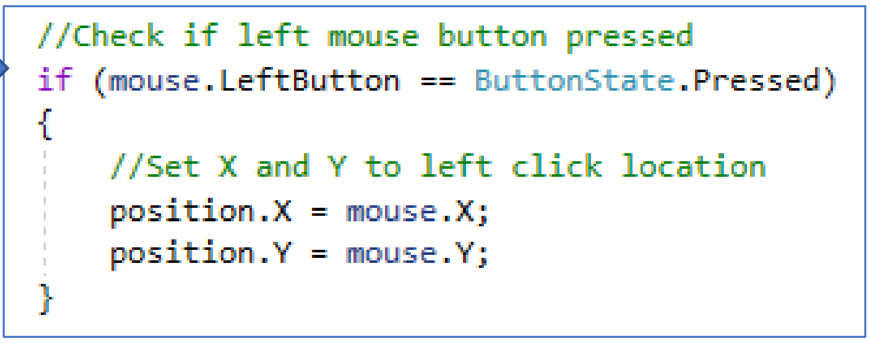
Mouse input

The first step is to setup the class an object to capture the keyboard state using the KeyboardState class.



```
//Setup mouse, so state can be checked
MouseState mouse = Mouse.GetState();
```

The next step is to find out whether the button you wanted was pressed and make something happen. In this example, each time the left mouse button is pressed the X and Y position changed to the mouse location.



```
//Check if left mouse button pressed
if (mouse.LeftButton == ButtonState.Pressed)
{
    //Set X and Y to left click location
    position.X = mouse.X;
    position.Y = mouse.Y;
}
```