

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Part 4 – Score and a Timer

Using our GUI builder

At last we want to display score and a timer during the game. We need a way to display text on the screen – we need a GUI. Mind, this is not a GUI builder tutorial so the GUI created will be rather minimal. We will just modify our MainScreenGui a bit.

There is one step we need to do before approaching this section. We need to copy over the images that we are going to use in our GUI. Browse out to games/ExampleWackAMole/data/images folder and copy over all the files to your games/WackAMole/data/images folder.

Run TGB and press F10. Select mainScreenGui from the dropdown menu in the middle (as shown in Figure 4.1).

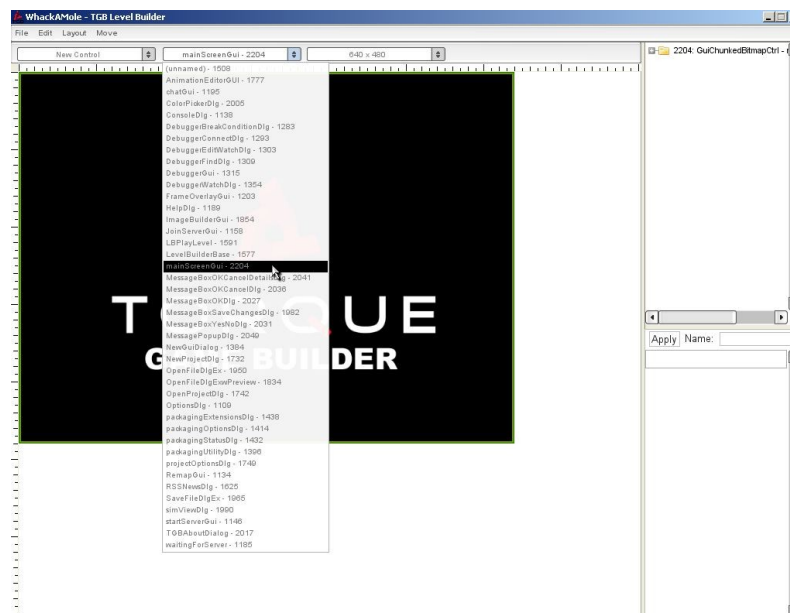


Figure 4.1

In the tree view to the right you can see that mainScreenGui consists of a GuiChunkedBitmapCtrl and a t2dSceneWindow contained in it. Select the GuiChunkedBitmapCtrl by clicking on it in the tree view. Then scroll down the parameter list to the bottom right and change its bitmap reference from logoblack.png to molebackground.png (as shown in Figure 4.2).



Figure 4.2

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Adding our board image

Then add a new GuiBitmapCtrl from the “New Control” dropdown, in the top left of your screen (as shown in Figure 4.3), and place it in top left of your GUI. Select boardalpha.png as bitmap the same way you did in the background above.

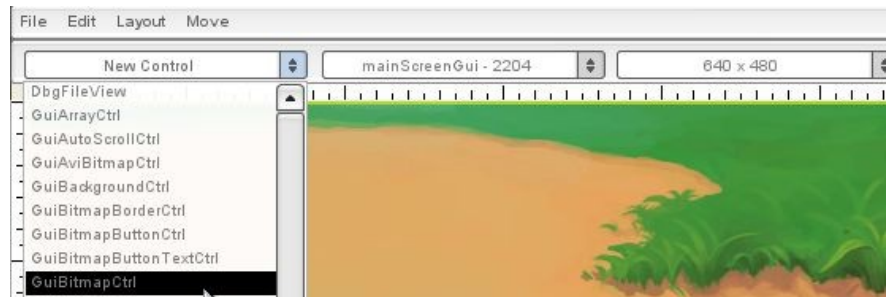


Figure 4.3

Adding text to our

Now we can add some text to our board. Start out by right clicking on the board and the board should then be surrounded by a yellow, green, and blue border (as shown in Figure 4.4). This means if we add any new controls (via the “New Control” dropdown) they will be added as a child control to our board. Now click the “New Control” dropdown and add a GuiTextCtrl. With it selected, look for the “text” parameter on the bottom right panel and type in “Whacked:” and then press enter (as shown in Figure 4.5).

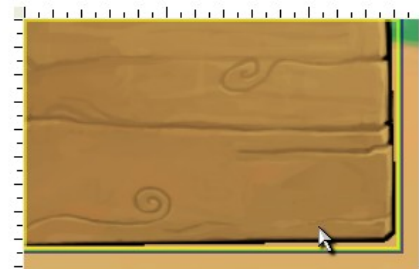


Figure 4.4

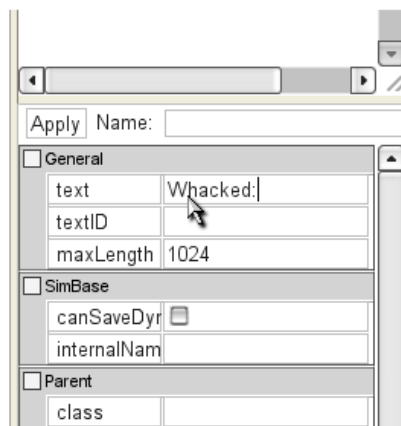


Figure 4.5a



Figure 4.5b

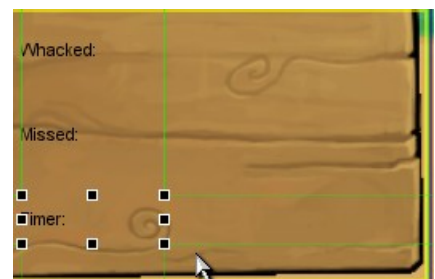


Figure 4.6

Now repeat that step for two more GuiTextCtrls. Second the second one's text to “Missed:” and set the third one's text to “Timer”. Drag and place them so they stack (as shown in Figure 4.6).

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Next you can select each one and change its “Profile” (in the lower right hand panel) to “GuiBigTextProfile” (as shown in Figure 4.7). You may have to resize each of the text controls for the text to properly fit. If we want to manipulate these text controls from script, we must give them names, we can do this by selecting each text control and enter a name in the “Name” property (middle right of your screen, right next to the “Apply” button) (as shown in Figure 4.8). Enter these three names, each for the appropriate text control: “MoleGuiWhacked”, “MoleGuiMissed”, “MoleGuiTimer”. Our final step to finish with our text controls is to ensure our text size is right. To accomplish this we can change the “HorizSizing” and “VertSizing” both to “relative” (as shown in Figure 4.9).

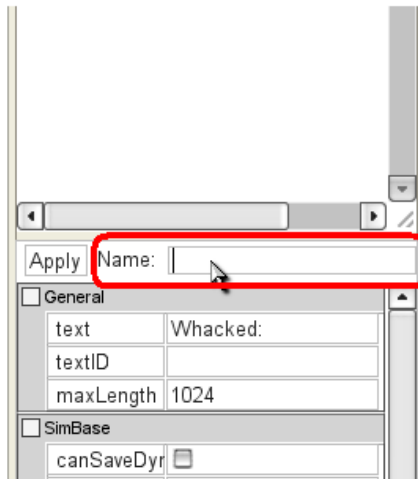


Figure 4.8

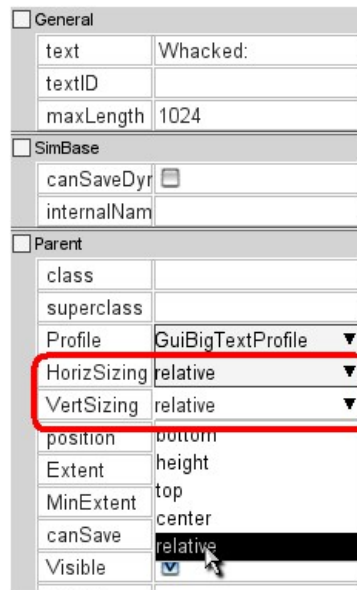


Figure 4.9

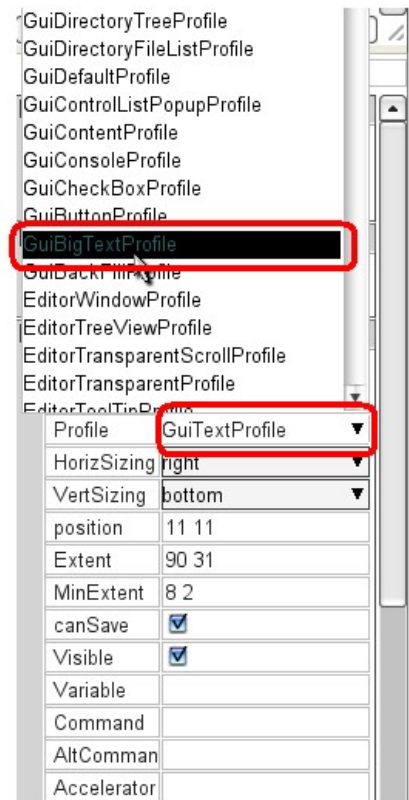


Figure 4.7

Select the t2dSceneWindow in the tree view on the right. Then click on Layout > Bring to Front. If we don't do that the new score board will obstruct the stop-level-panel the Level Builder places there and you wouldn't be able to get back to the Level Builder when testing your level. When you are done go to File > Save Gui and save the file as mainScreen.gui in /WhackAMole/Gui/. To exit the GUI builder and return to our level builder you can click on the center dropdown in the top section of your screen (as shown in Figure 4.10). Select the “Level Builder Base” GUI and then hit “F10” to exit the GUI Editor.

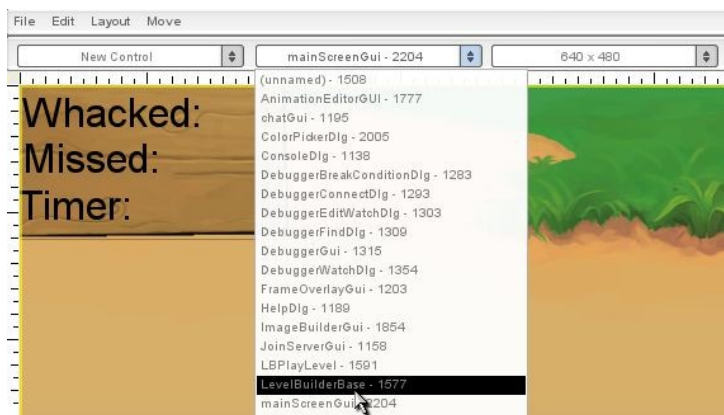


Figure 4.10

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Updating the GUIs

You should be presented with the whole new background when you click Run Game. What we have to do is update the GuiTextCtrls. That will happen mostly in moleLevel.cs. First we modify MoleLevel::onLevelLoaded to reset score and then we introduce a global variable that holds the number of seconds one game lasts.

```
$MOLE_LEVEL::timePerGame = 30; // seconds

function moleLevel::onLevelLoaded(%this)
{
    %this.respawnPointSet = new SimSet();

    // set the occupied counter to zero
    %this.spawnPointsOccupied = 0;

    %this.maxCreationInterval = $MOLE_FACTORY::maxCreationInterval;
    %this.minCreationInterval = $MOLE_FACTORY::minCreationInterval;
    %this.speedUp = $MOLE_FACTORY::speedUp;

    // Reset score
    %this.whackedCount = 0;
    %this.missedCount = 0;

    // Reset the GuiTextCtrls
    MoleGuiWhacked.setText("Whacked:" SPC %this.whackedCount);
    MoleGuiMissed.setText("Missed:" SPC %this.missedCount);

    // start the mole creation
    %this.startFactory();
}

//Then we'll add functions to increase score and update the GuiTextCtrls.
function moleLevel::incWhackedCount(%this)
{
    %this.whackedCount++;
    MoleGuiWhacked.setText("Whacked:" SPC %this.whackedCount);
}

function moleLevel::incMissedCount(%this)
{
    %this.missedCount++;
    MoleGuiMissed.setText("Missed:" SPC %this.missedCount);
}
```

Code Sample 4.1

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Now we only have to call `MoleLevel::incWhackedCount()` and `MoleLevel::incMissedCount()` whenever a mole is whacked or missed. We do that in `mole::onAnimationEnd()` (in the `mole.cs`):

```
function mole::onAnimationEnd(%this)
{
    if( %this.getAnimationName() != ("animMoleComeOut" @ %this.moleColor) )
    {
        // free the respawn point for other moles
        %this.respawnPoint.isOccupied = "";

        // decrement the occupied counter
        %this.sceneGraph.spawnPointsOccupied -= 1;

        // update score
        if( %this.getAnimationName() $= ("animMoleWhacked" @ %this.moleColor) )
            %this.sceneGraph.incWhackedCount();
        else
            %this.sceneGraph.incMissedCount();

        %this.sceneGraph.schedule( 1500, "spawnMole");
        %this.safeDelete();
    }
}
```

Code Sample 4.2

And there you have it: A simple scoring system.

Next we will update the timer. When the level is loaded, a new scenegraph is created for it. That means that we can keep a timer based on the “scene-time”, which should reflect the time the level is running. Let's use that for our timer code. We will use the `onUpdateScene()` callback (a “callback” is simply a function automatically called by the engine, that way we can place the function in script and respond to that event) of the scenegraph to update the `GuiTextCtrl` of the timer. Add this function to `moleLevel.cs`:

```
$MOLE_LEVEL::timePerGame = 60;

function moleLevel::onUpdateScene(%this)
{
    // Calculate how much time is left
    %timeLeft = $MOLE_LEVEL::timePerGame - %this.getSceneTime();

    // Round it to full seconds
    %timeLeft = mFloor( %timeLeft + 0.5 );

    // Update the GuiTextCtrl
    MoleGuiTimer.setText( %timeLeft );
}
```

Code Sample 4.3

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Creating an end screen GUI

Let's create another GUI that is displayed when time is up. Open the GUI builder (F10) and select File > New GUI... and create a new GuiControl called MoleEndScreenGui. Create a GuiBitmapCtrl with molewritingalpha.png as bitmap. Then create two GuiButtonCtrls and label them with PLAY AGAIN and QUIT. Set all HorizSizing and VertSizing fields to relative.

Set the Command field of the QUIT button to quit();. Then set the Command field of the PLAY AGAIN button to restartGame();. These functions will be called when the button is clicked on. Save the GUI under WhackAMole/gui/endScreen.gui.

Next create the restartGame() function in game.cs right under endGame().

```
function restartGame()
{
    Canvas.popDialog(MoleEndScreenGui);
    endGame();
    startGame( $levelForRestart );
}
```

Code Sample 4.4

Then modify startGame() so it saves the name of the last level loaded in \$levelForRestart and add a exec() call for endScreen.gui at the top of the game.cs file below to all of our other exec calls:

```
function startGame(%level)
{
    //exec game scripts
    exec("./mole.cs");
    exec("./moleLevel.cs");

    //exec our gui files
    exec("~/gui/endScreen.gui");

    // Set The GUI.
    Canvas.setContent(mainScreenGui);
    Canvas.setCursor(DefaultCursor);

    moveMap.push();

    sceneWindow2D.setUseObjectMouseEvents( true );

    if( isFile( %level ) || isFile( %level @ ".dso" ) )
    {
        sceneWindow2D.loadLevel(%level);
        $levelForRestart = %level;
    }
}
```

Code Sample 4.5

Torque Game Builder – Whack-A-Mole Tutorial Part 4

Adding a time check

Now add a time-check in `moleLevel::onUpdateScene()` (in our `moleLevel.cs`). When time is up then the scene is paused and the `MoleEndScreenGui` is pushed on the Canvas.

```
function moleLevel::onUpdateScene(%this)
{
    // Calculate how much time is left
    %timeLeft = $MOLE_LEVEL::timePerGame - %this.getSceneTime();

    // Time-check
    if( %timeLeft < 0 )
    {
        %this.setScenePause( true );
        Canvas.pushDialog( MoleEndScreenGui );
    }

    // Round it to full seconds
    %timeLeft = mFloor( %timeLeft + 0.5 );

    // Update the GuiTextCtrl
    MoleGuiTimer.setText( %timeLeft );
}
```

Code Sample 4.6