Smalltalk Programming
Lesson 22

In the last lesson you added the ability to keep track of the score. In order for the score to be useful it needs to be used in the shooter game. Today, you will make changes to make the score work with your shooter game.

1. Create the new method below for your `Score` class.

displayPoints

```
    self
        string: points printString
        fontName: #Atlanta
        size: 22
```

2. The method `displayPoints` is what will actually be used to display the score on the shooter game screen. Look at each line of code and see if you can figure out what it might be doing.

3. You will want your score to update every time points are added. Make the following change to `Score>>points:` to do this.

points: anInteger

```
    points := points + anInteger.
    self displayPoints
```

4. Now add the changes to `Score>>initialize`.

initialize

```
super initialize.
points := 0.
self lock.
self textColor: Color green.
self displayPoints.
self extent: 90 @ 0
```

5. These changes will set the color of the text, display the points for the screen, and set how much space the score will take up on the screen. The "lock" message keeps someone from being able to edit (type over) the score.

6. Now that your score object is ready to be used, let's make it a part of `ShooterGame`. Like you did for the `Ship` object instance, you will also want to add an *instance variable* for your `Score` object instance. Instead of adding the *score* instance variable to the class code, let's have Smalltalk add the instance variable to the code for you when you save the `ShooterGame>>initializeScore` method below. Note that a window will popup saying "Unknown variable: score!" since it does not yet exist. Select "declare instance" to let Smalltalk know how you want to use *score* in the code below.

initializeScore

```
score := Score new.
score position: self position + (10 @ 10).
self addMorph: score
```

7. Whenever a new shooter game is started, you will want the score object to be included. Adding the following change to

`ShooterGame>>initialize` will create an instance of score for use in the game.
initialize

```
super initialize.
self position: 100 @ 100.
self extent: 640 @ 480.
self color: Color black.
self setNameTo: 'Shooter Game'.
self initializeShip.
self initializeScore.
self initializeEnemies
```

8. You can test your code by using explore morph on your running shooter game. Remember to select ShooterGame before sending these messages.

```
self initializeScore.
score points: 100.
```

9. Test your score by adding points to your score display. Send the `points:` message to `score` a number of times. What do you notice?

10. Now try sending the message “score points: -100.” What do you notice? What might this be useful for?

11. Try shooting a few enemies. What happens with the score? Why?

12. What else needs to be done for your shooter game?

13. Save and Quit your Smalltalk image.