Smalltalk Programming
Lesson 21

In the last lesson you added the ability to destroy enemies. Now would be a good time program the ability to keep score of your victories. Today, you will begin creating the code needed for keeping track of the score.

1. There are 4 things needed for your score. What 4 things would be needed for scoring?

2. A score needs to; 1) receive points for when an enemy is destroyed, 2) to add the points received to the current score, 3) to display the score points, and 4) an easy thing to forget – it needs to remember the points for the score.

3. What will be needed in order to remember the points of a score? Hint – the solution will be the same as that used for remembering the enemy direction.

4. If your answer was to use an *instance variable*, then great job! If not, that is still fine. These lessons are to help learn and this will be another good example for using instance variables.

5. Type and save the following code to create the new `Score` class.

```
TextMorph subclass: #Score
   instanceVariableNames: 'points'
   classVariableNames: ''
   poolDictionaries: ''
   category: 'ShooterGame'
```

6. Notice that the score is a subclass of `TextMorph`. `TextMorph` provides the `Score` class with the ability to display text (letters and numbers). Displaying numbers will be needed to show the score.

7. Once again you will need to create an initialize method for your Score instance.

```
initialize

    super initialize.
    points := 0
```

8. By now you can probably figure out each line of code. Do you know why the *points* instance variable needs to be assigned the value 0? What might happen if it is not set to 0?

9. A score will not be of much use if it does not provide its number (or value). Type and save the following Score method.

```
points

    ^ points
```

10. A score will also not be of much use if it there is no way to receive points. This will be a good place to also add the new points to the current points for a total score. Type and save the following Score method.

```
points: anInteger

    points := points + anInteger
```

11. Open a new workspace window to test your new code. Type the following lines and use the last 2 expressions a few times (Ctrl-d to “do it” or Ctrl-p to “print it”) to test and observe your new code.

```
a := Score new.
a points: 100.
a points.
```

13. After testing your code you can close the new workspace that you opened for this lesson.

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

15. Save and Quit your Smalltalk image.