Building a Scorekeeper

Alice 3 Tricks in Mini Trivia (1/4)

0 → -5 → 20 → 85

By Vicki Zhang
Duke University
Under the direction of
Professor Susan Rodger
June 2017
About the Mini Trivia Challenge

Mini Trivia is a four-question game created by Vicki Zhang. The author explains four useful topics involved in separate tutorials:

1. scorekeeper
2. asking user for an answer
3. billboard
4. multi-layered object clicking

This tutorial explains how to build a scorekeeper. Start by downloading Mini Trivia_Challenge 1 Scorekeeper_Start
To get started

Find and create a new `TextModel`.

Set `name` to `score`, and initial `value` to `0`. 
Drag the text onto a clear spot (shown on next page) using the **Default** handle style.

Resize it by
- changing the data directly
- or using the **Resize** handle style.
Tweak it to roughly this size and location.

Color could be changed, if so inclined:

When satisfied, click on

Paint =

Edit Code
One new Property for Score

• Goal: Create a property used to store the value of current score
• Purpose: To access the value easily
• How: Add new property for TextModel
  • Value type: whole number
  • Name: currentScore
  • Initializer: 0
Note that we are enabling ALL text models this property and the procedures that we will create.
After you added the `currentScore` property, Alice \textit{automatically creates} a procedure (\texttt{setCurrentScore}) and a function (\texttt{getCurrentScore}). We will use them soon.

`currentScore` is a property of the `TextModel`, and used a \textit{parameter of this procedure}. 

\[\texttt{this setCurrentScore} \quad \texttt{currentScore: ???}\]

\[\texttt{this getCurrentScore}\]
Three new **Procedures** for Score

1. **setScore**
2. **addScore(#)***
3. **reduceScore(#)***

**Notes💡:**

- addScore(5) will add 5 to the score, and reduceScore(10) will subtract 10 from the score.
- The # sign in the procedure is a placeholder demanding an input.

Consider: If addScore did not have this input, Alice would not know how much to add. This is called a *parameter*. 
Set Score

• Add TextModel **Procedure** and name it **setScore**

• This procedure is created for the purpose of setting the content of display
Set Score

Drag in the **setValue** procedure

>>Select “custom TextString”

>>Press “Ok” directly. The result is shown below.

Then click the tiny triangle on the right and select

“” + ??? >>

whole number >>

currentScore
text:

- (current value)
- "hello"

Custom TextString...

- "hello"

DecimalNumber

- WholeNumber

- SThing

- 0
- 1
- 2
- 3

Custom WholeNumber...

- this.currentScore
- newScore
Set Score Completed

declare procedure setScore

Do in order

\texttt{this.setValue} + = \texttt{this.currentScore}
Add Score

• Create a second **Procedure** for the TextModel
• Name it **addScore**
• Add a parameter for this procedure
  • **so that Alice knows **how much** to add**
  • Name the parameter **howMuch**
Add Score

• Drag in the `setCurrentScore` procedure and select `currentScore` as the parameter in the dropdown menu.

• We want to add to the `currentScore` by `howMuch`

• The resulting formula should be:

  ```
  setCurrentScore to (currentScore + howMuch)
  ```

• Process shown on next page.
The point of addScore is to update the value of currentScore and update the score displayed on the screen by calling setScore.
Reduce Score

• This is merely a numerically flipped version of `addScore`.
• Try it yourself first!
• Step-by-step guide on the next slide.
Reduce Score

• Create a second Procedure for the TextModel
• Name it reduceScore
• Add a parameter for this procedure
  • **so that Alice knows how much to reduce**
  • Name the parameter howMuch
Reduce Score

• Drag in the `setCurrentScore` procedure and select `currentScore` as the parameter in the dropdown menu.

• We want to subtract `howMuch` from the `currentScore`

• The resulting formula should be:
  ```plaintext```
  `setCurrentScore` to `(currentScore-howMuch)`

• Process shown on next page.
The point of `reduceScore` is to update the value of `currentScore` and update the score displayed on the screen by calling `setScore`. 
Scorekeeper Completed!

- Now we just have to plug in the procedures to appropriate places in the game.
- You will have comments in Question A, Question C, Question D, and initializeEventListeners for guidance.
Example: Question A

Step 1:
Go to **scene** and you should find the scene procedures.

Select **questionA** procedure.
Example: Question A

Step 2: Select on the left tool bar TextModel—this.score, which is the visualized form of our scorekeeper.
Example: Question A

• We can now see all procedures of TextModel on the left, including the ones we created.
• Drag in addScore into the doTogether box, either below or above the comment.
• Select custom whole number and type in 10.
Example: Question A

• Similarly, drag reduceScore in. Make sure you place it at the same level of the comment, i.e. under “else” and outside of do together.

• Select custom whole number and type in 5.
Try it out!

• Run the game and click on the hammer, which corresponds to Question A.

• Comments for Questions B, C and D are found, respectively, in initializeEventListeners, questionC, questionD.

Finish them all for a complete game!