|  |  |
| --- | --- |
| This lesson suggests several small programming projects for enhancing the *TurnOffLights* app.    Hints and suggestions are provided.    **Objectives**: In this lesson you will learn to :   * better navigate the App Inventor online programming platform; * deepen your understanding of event-driven programming; * develop your understanding of how variables are incremented; * develop a better understanding of conditional *if* blocks and apps that make decisions. | LightsOutPart2.png  ***Click the image to watch video*** |

## 

## 

## Getting Ready

For this lesson you can use the project you created in the previous lesson. Start up [App Inventor](http://ai2.appinventor.mit.edu/) and after opening your *TurnOffLights* project, rename it *TurnOffLightsV2*, for Version 2 -- or something similar to that. Then complete the programming exercises described below.

# 

## Mini Project Ideas and Hints

Here are some creative project ideas. Working in pairs with your programming partner, complete all of them. You are encouraged to discuss your ideas for how to solve these problems with the instructor and with your partner and other students.

1. Personalize the app by changing its ***socially useful theme*** to one that you like. This will require you to add your own images for the *Canvas* background and the *ImageSprite.*
2. To make this more like a real game, add a scoring feature to the app, modifying the UI so it can display changes to the user's score as it changes. Hint: Use what you learned in the Paint Pot app about incrementing a variable to implement the score feature.
3. **If/else algorithm:** Add a winning score feature that stops the game and congratulates the user when they reach a certain score. Stopping the game should include stopping the ImageSprite from jumping around. (Possible enhancement: Use a *TextToSpeech* component and have it say something when the player reaches a certain score. Or use a *Notifier* component to pop up and congratulate the player.)
4. **Procedural abstraction**: Add a Reset button to the app that allows the player to restart the game after it's been stopped. Define a ***resetGame*** procedure to encapsulate the tasks involved in resetting the game. These would typically include setting the score back to 0 and getting the *ImageSprite* to start moving again. In addition to calling the procedure from the reset button event handler, it should be called from the *Screen1.Initialize* handler. Make sure you use good naming conventions when you add a button to the app.
5. Create one or more of your own enhancements or variations for this app. Here are some ideas:

* You could also implement keeping track of the number of misses -- i.e., the number of times the player failed to touch the *ImageSprite* -- and factor this into your scoring algorithm.
* **Challenging.** Add a second image to the app to turn the game into one that forces the player to distinguish between "good" and "bad" choices. For example, if your app had a nutrition theme, perhaps one image could be picture of a healthy food item and the other a not-so-healthy item. Perhaps the player's score decreases if they touch the "bad" choice.
* **Challenging:** Change the speed of the mole when the player reaches a certain score. (Hint: Recall that in this app, the Mole's speed is controlled by the Clock timer.)

Nice work! Now reflect on what you learned in this lesson and then retry the interactive exercises with the goal of answering each question correctly.

# Reflection: for your Portfolio

Create a page named ***Turn Off Lights Projects*** under the *Creative Projects* category of your portfolio and answer the following questions.

1. Describe the purpose of each enhancement that you added to your app. Give brief descriptions of the enhancements and provide screen shots of important blocks and describe how you used them to solve certain programming problems.
2. When the user touches an *ImageSprite*, both the Canvas.Touched and ImageSprite.Touched events are triggered. This is important for more complex games. For instance, suppose there are "good" and "bad" sprites in your game. If you hit one, you earn a point. If you hit the other, you lose two points. If you hit the Canvas and don't hit the ImageSprite, you lose 1 point. How would you code this?
3. How do you speed up the movement of the ImageSprite? What is the fastest it could move?

# Exercises

To practice your skills and test your knowledge try some [interactive exercises](https://course.mobilecsp.org/mobilecsp/unit?unit=23&lesson=53).