Specifically, you must address the following rubric criteria:

* **Arrange the bricks into an organizational structure that promotes engagement with the animation**. The goal is to create a layout that is visually unique or compelling. Your layout should also include a style of bricks that makes the animation more interesting to watch. Here are some options to consider. You can also try an idea of your own.
  + Add texture or color to the different kinds of bricks.
  + Change the sizes of the bricks.
  + Add a manually controlled paddle to the bottom of the screen, using the brick item as a base.
* **Apply physics laws to the circles**. When a circle hits one side of the screen, the circle’s progress should change in some way. Before hitting the side of the screen, the circles move at a constant speed and have randomized movement. Once the circles bounce off one edge of the screen, you can change them to make the animation more engaging. Here are some options to consider. You can also try an idea of your own.
  + Change the speed of the circle.
  + Change the angle of trajectory so the circle follows physics laws instead of taking a randomized pattern. Changing the angle of trajectory means the circle would continue in the direction it was heading rather than moving backward.
  + Add friction to specific surfaces. Adding friction would affect the circle and slow its progress after it collided with the surface.
* **Alter the state of the bricks upon collision**. You need a code for something to happen when a circle collides with a brick. This means updating the code to alter the state of the bricks on collision. Here are some options to consider. You can also try an idea of your own.
  + Require the brick to take a certain number of hits before it disappears.
  + Change the color or texture of the brick when it is hit.
  + Combine the previous two options. Make the brick change its texture each time it is hit until it disappears. For example, you may choose to add cracks to the texture of the brick until it is destroyed.
* **Alter the state of the circles on collision**.  You need a code for something to happen when a circle collides with another circle. This means updating the code to alter the state of the circles on collision. Here are some options to consider. You can also try an idea of your own.
  + The two circles combine to become one larger circle.
  + The circles change their color or texture.
  + Both circles disappear once hit.
  + The circles spawn multiple smaller circles.
* **Explain the changes you made to the code**. Discuss the work you completed by focusing on the different tactics you used to create a fully realized 2D animation. What were the changes you chose to make? What was your intent behind them? How did you approach coding to successfully create this outcome?
* **Create code that follows a logical flow without syntax errors**. The code you create needs to be executable. All the code that is included needs to be reached by the execution. You don’t need to write everything as a single function. Your work should be well modularized.
* **Apply coding best practices in your creations**. Pay particular attention to the way you format and comment your code. Program code should be easy to read and follow industry-standard code formatting practices, such as indentation and spacing. The source code should be brief and clear. Use descriptive comments.