Thoughts on Level Editor implementation

1. Speaking with a combination of Dr. Lewis and Dr. Cline, the specific type of level editor that they would like to see implemented is a run-time end-user level editor. This means that they would like the customer to be able to create levels, preferably within the app itself (though Dr. Lewis mentioned it wouldn't necessarily even need to be done in Unity if the levels could be easily ported into the game).

2. The implementation that was considered in Summer 2017 was implementing a process by which new scenes (the unity asset type levels are currently built on) could be generated at run-time and saved to the game's assets.

3. The "Hooks" provided by Summer 2017 should allow this to be easily implemented given the following steps are completed:
    a. Find or create a way to save new scenes at run time
        i. The Summer 2017 group did very little research in this area as our term goals fell out of this area
        ii. There are some examples available on line of run-time end-user level creators for Unity that may serve as a reference
    b. Within the new scenes, create pre-tagged WinCon_Unchecked object prefabs (This is pretty basic code in Xamarin and could be pieced together from existing code)
        i. The selection of this could be implemented in a number of ways. We specifically mentioned checkboxes to Dr. Cline, but this may become crowded if enough win conditions are eventually added. Note that the location of the unchecked/checked boxes on screen is based on the suggestion of the customer, and can easily be changed.
        ii. Reference biorubebot-dev-master\Documents\WinConditionInstruction.txt for more information on Win Condition implementation

Thoughts on Modularity

1. Both Dr. Lewis and Dr. Cline have a great interest in adding in more modular behavior.
    a. Dr. Cline mentioned that a number of the game objects could actually react together in multiple ways and her end goal is to allow them to do so.
        i. This would require dividing out portions of the existing scripts and adding checks based on either the added win scenarios or the existence of other available game objects.
        ii. The implemented game objects represent only a minute fraction of the total objects that Dr. Cline would eventually like to implement. For this reason, it is suggested that the code be re-worked in general with an emphasis on object-oriented behavior on a piece by piece (read class by class) basis.
    b. Dr. Lewis's interest in modular behavior seems to stem from the ease of implementation of new ideas and pieces
        i. Refactoring the code as suggested above should aid in this
        ii. The implementation of a level editor should aid in this

Other thoughts and issues

1. Meet with Dr. Cline as soon as possible. She is the customer. It's her goals and your work achieving those that really matter. Additionally, understanding the biological concepts behind the existing gameplay along with the planned pieces is crucial in understanding how best to implement the modularity/OOP functionality, as well as give better direction for core architectural changes.

2. Every project member should go through the process of downloading and installing Unity in order to become familiar not just with the code, but also understanding (at least somewhat) how the game engine functions. It can be difficult at times understanding how the object on screen are tied to specific code, so it is important to become familiar with the Unity software as soon as possible. Also work with Dr. Lewis when updating to a newer version of Unity as currently utilized Unity functions may become deprecated or otherwise broken beyond v5.6.1f1 used for Summer 2017.
    a. We suggest first looking at each of the "WinCon_Unchecked" (uncheck boxes that have been renamed to match its corresponding win condition) objects on screen and exploring how the tag for that object is used to identify a point in the code where that biological event occurs. You can then see how once the event occurs, the object's tag is dropped and the unchecked box prefab is replaced with a "WinCon_Checked" (checked box) prefab. Understanding the ties between the objects on screen and the code underneath will greatly help the development process not only for improving the current game, but also when hooking in a Level Editor.

3. It would likely behoove you to try to get the project loaded on the iPad at every meeting. If a member of your group has an Apple Developer account, this should be easy. Otherwise, you'll need to work with Dr. Lewis to accomplish this goal. This will allow you to see what bugs exist early and address them as needed. **The game behaves differently in Unity and on PC than it does on the iPad or on a Mac.**

4. The current pathfinding system needs a lot of work.
    a. Items are currently "locked" together once they identify one another. This means that if one object cannot get to the other (for example, if you place an ATP outside of the cell membrane but there is an inactivated portion of the receptor inside, the receptor will never be activated even if other ATPs are available inside). This should be easily solved with a "time-out" feature that would allow both objects to reset.
    b. The collision system doesn't seem to work on all platforms. Have not identified a fix for this.