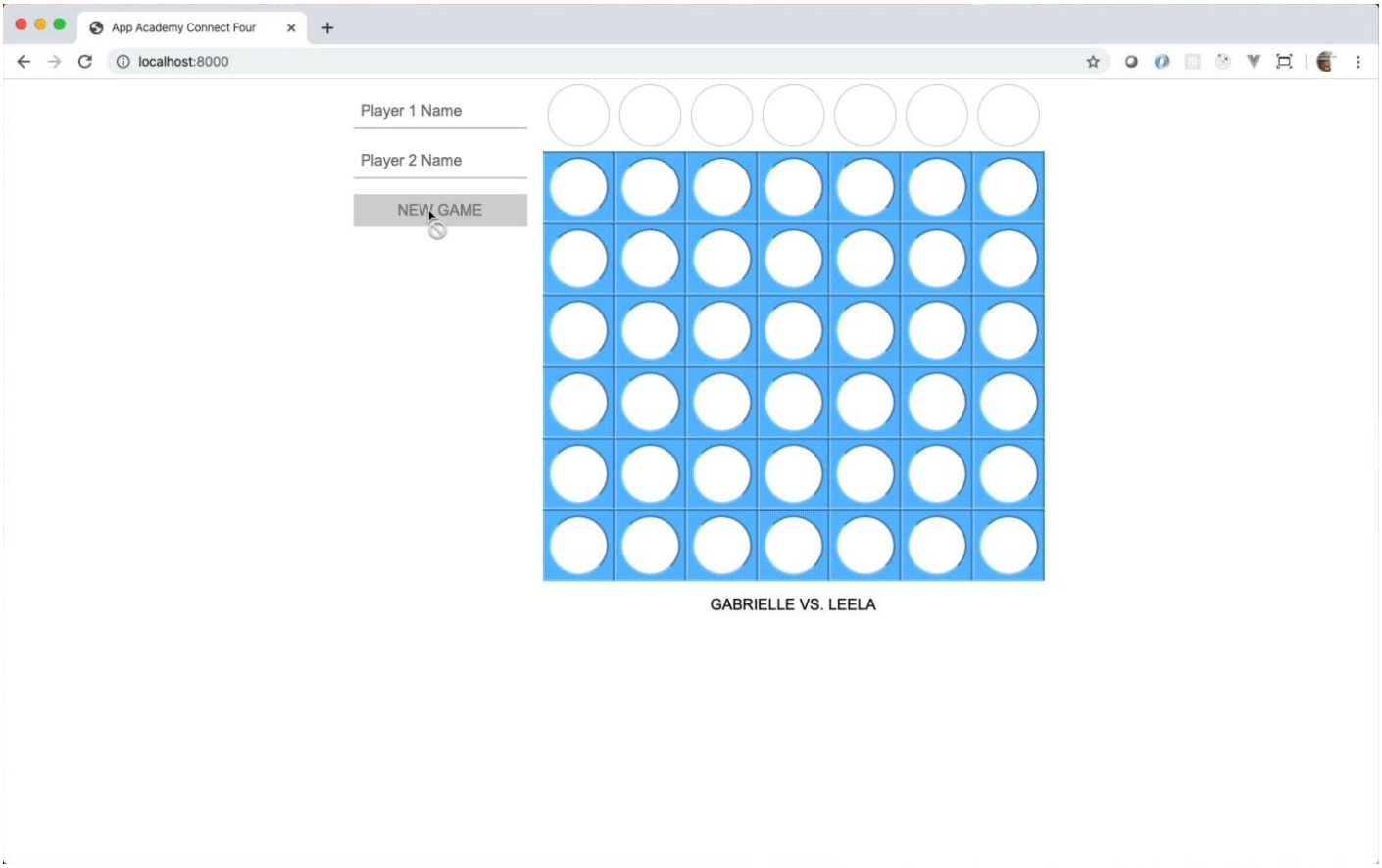
Full Columns



Now, you need to figure out if a column is full for two reasons:

1. **In the "model"**: You don't want to add a token to a column that is already full.
2. **In the UI**: You want to change the appearance of the click targets.

You need to do something about the behavior of a column. Since you have a `Column` class already, you can just do it in there! See how this object-oriented thing makes it easy to determine where to put new features!?!?

In the `Column` class:

- Depending on how you implemented it, just don't add the token if there is no available slot for it.
- Add a method named `isFull` and have it return `true` if there are no more available slots (that is, there are already six tokens in it).

Since the `updateUI` is going to need to know if a column is full and the `updateUI` method *only* knows about the `Game` object and none of the `Column` objects (because you're following the Law of Demeter with respect to your own code), add an `isColumnFull` method that takes a column index between `0` and `6`, inclusive, and returns the value of the `isFull` method invoked on the appropriate `Column` object stored in the `columns` array.

In the `updateUI` method, create a `for` loop that iterates over the values from `0` to `6`, inclusive. For each value:

- Select the element with the id of "column-«column index»".
- If the value returned from the `isColumnFull` method on the `Game` object is `true`, then *add* the "full" class to the element selected in the previous step.
- If the value returned from the `isColumnFull` method on the `Game` object is `false`, then *remove* the "full" class to the element selected in the previous step.