Introduction

Methods are useful for the sake of security, but we can also use them to reduce the amount of code we have to maintain. To demonstrate this, we are going to combine the increment and decrement events into a single method. This is possible because there's a lot of similar code shared between them. The only difference is that inside the increment event, we're passing a value of 5 into the `$inc` operator, while inside the decrement event, we're passing through a value of -5.

To begin, we'll focus on the increment event. Inside this event, remove the `update` function and replace it with a `Meteor.call` statement. Pass through a value of `updateScore` as the first argument. For the second argument, pass through the `selectedPlayer` variable.

Next, create the `updateScore` method inside the `Meteor.methods` block:

```
Meteor.methods = {
    updateScore: function(selectedPlayer) {
        PlayersList.update({_id: selectedPlayer}, {$inc: {score: 5}});
    }
}
```

To make the method more useful, introduce `scoreValue` variable:

```
Meteor.methods = {
    updateScore: function(selectedPlayer, scoreValue) {
        PlayersList.update({_id: selectedPlayer}, {$inc: {score: scoreValue}});
    }
}
```

Pass it as an argument via `Meteor.call`.

Meteor.call Statement

All we have to do is add the `Meteor.call` statement to the decrement event. But instead of passing through a value of 5 as a third argument, we can pass through a value of -5: both buttons will work as expected, but with a lot less code involved.

There are, however, a few problems we still need to solve. First we need to use a `check` function to make sure the `selectedPlayer` variable is a string. Second, we need to use a `check` function to make sure the `scoreValue` variable is a number. Both of these functions will prevent users from passing unwanted data into the method.

Third, we need to create a `currentUserId` variable inside the method, and create a conditional around the `update` function to prevent users from using this method if they're not logged into an account. And fourth, we need to pass the `currentUserId` variable into the `update` function so the logged in users can only update documents that belong to them:

```
Meteor.methods = {
    updateScore: function(selectedPlayer, scoreValue) {
        check(selectedPlayer, String);
        check(scoreValue, Number);
        if(currentUserId) {
            PlayersList.update({_id: selectedPlayer}, {$inc: {score: scoreValue}});
        }
    }
}
```