Helper Function

At the moment, our leaderboard template only contains some static text. To fix this we're going to create a **helper function** that is attached to template and allows us to execute code from inside the interface.

To begin, we'll take an old approach to creating helper functions. This approach is deprecated, meaning it's no longer officially supported. By the time you're watching this video, it might not work at all, but this older approach is easier to teach and to understand. It allows us to ease into the non-deprecated approach, which we'll talk about in a moment. Inside the JavaScript file write the following code inside the `isClient` conditional:

```
Template.leaderboard.player
```

This is the deprecated syntax for creating a helper function andit can be broken down into three parts:

- The `template` keyword searches through the templates in our Meteor project. We only have the one template at the moment, but a complete project would probably have many more.
- The `leaderboard` keyword is a reference to the leaderboard template we created earlier. This function then, will be attached to this specific template.
- The `player` keyword is the name we're giving to this function. We can however, name it whatever we want. The name is simply something we'll reference from inside the HTML file.

To attach code to this helper, associate it with a function and inside return some static text:

```
Template.leaderboard.player = function() { return 'test' }
```

Then, place the following code inside the leaderboard template:

```
{{player}}
```

Here, we're using another Spacebars tag, as evidenced by the double curly braces. But notice that we're not using a greater than symbol, and that's because we're not including a template. Instead, we're referencing the name of the `player` function that we just created. This is how we include a helper function in a template.

Testing it Out

If we save the file and switch back to the browser, we can see that the text being returned by the `player` function now appears within the interface. If the text doesn't appear, something is either wrong with the code, or this approach to creating helpers has been removed from Meteor.

Now that you know the old way of creating helper functions, we can discuss the new way. Delete the helper function we just created and replace it with

```
Template.leaderboard.helpers
```

This is similar to the code we wrote a moment ago, but to be clear, we are not creating a helper function named `helpers`. Instead, the `helpers` keyword allows us to define a number of helper functions within a single block of code. Just pass through a pair of curly braces, and between these braces, we can define helper functions in the JSON format:

```
Template.leaderboard.helpers = {
    player: function() { return 'test' }
}
```