Handout

Your First Meteor 1.2 Application
Lesson 8 - Step 3
David Turnbull

Introduction

By default, the players in our list are ranked by the time they were added to the players list collection, rather than being ranked by their scores. To fix this, we'll modify the player helper function.

```
'player': function() {
    return PlayersList.find({}, {sort: {score: -1}});
}
```

By using curly braces, we're explicitly stating that we want to retrieve all of the data from the `PlayersList` collection. This is the default behavior of the find function, but by passing through the curly braces as the first argument, we can pass through a second argument where sorting rules are defined.

By passing through a value of `-1`, we can sort the documents in descending order. This means players will be sorted from the highest score to the lowest score. If we passed through a value of `1`, players would be sorted from the lowest score to the highest score. But what happens if two players have the same score?

Rank Alphabetically

Suppose, we have Bill and Bob players with the same score. Bill should be ranked above Bob because, alphabetically, but at the moment,this won't happen if Bob was added to the collection before Bill. To fix this, pass the `name` field into the sort method, but this time with a value of 1:

```
'player': function() {
    return PlayersList.find({}, {sort: {score: -1, name: 1}});
}
```