Eugen: Welcome to this lesson out of Learn Spring where we're going to be taking a closer look at some of the core Spring MVC annotations.

Let's start with a primary annotations for actually defining our controllers. That is @Controller, and @RestController. Up until this point, we've been using these core annotations, for example when we implemented our first simple controller with RestController, but also when we implemented our [00:00:30] MVC application where we used @Controller. But what we haven't done is we haven't stopped and discussed thee exact differences between these two annotations. So first of all, let's do that here.

We're going to start with @Controller. This raw annotation doesn't make any assumptions about the style of application that we're building. So, it doesn't really matter if we're building a REST API, or a more traditional MVC style of application, controller [00:01:00] works for both of them. But if we are building a REST API, then typically what we want to do is we want to marshal our responses, our resources, directly to the http response body. That is usually how we're building a REST API. So, in order to do that, we will have to use the response body annotation. But, naturally, putting that annotation on each and every method gets repetitive, and that is exactly where RestController [00:01:30] comes in. This annotation simply bundles controller and response body together in a single annotation. So basically, we don't have to add response body manually each and every time. We get that by default using the RestController annotation.

All right, now that we've defined our controller, the usual next step here is of course to start defining mappings. That is where this core annotation comes in. Simply put, [00:02:00] RequestMapping helps us map one of our methods to an HTTP verb, a request [inaudible 00:02:07] and of course some other details. And more recently, Spring MVC introduced some shortcuts. Again with the same goal of simplification, we now have these shorthand annotations that we can use instead of the more open and low level RequestMapping annotation. And, as their name suggests, these simple [00:02:30] annotations aren't really doing anything other than deciding the verb for a RequestMapping. So instead of having to specify that manually, you just use the right annotation and that gets mapped to the right verb.

All right, enough theory. Let's jump right in and let's start doing some coding.

All right, so our goal here is to create a simple crowd REST API for our simple domain. We are of course going to be using Project. [00:03:00] Now, our starting point here is already a working controller since we have already done part of this work. And again, we are already using RestController here, as well as RequestMapping, and we have these two end points declared here. But using an annotation and really understanding it are two very different things. So, let's have a closer look at this RequestMapping here, which is defined not on an individual [00:03:30] method, but at a base control log level. That's interesting because when you think about, the annotation logically belongs on a method, so why do we have it here on the controller? Well, that is simply because we can actually combine multiple instances of this same annotation at multiple levels and

will allow Spring MVC to match them behind the scenes and form our final mappings at the method level.

[00:04:00] Now, that may seem simple, but it's hugely powerful and really contributes to the simplicity story of Spring. Simply put, when we define this at a control log level, it will get applied to all of the methods in this controller. So that's a baseline of common configuration which we can refine at the method level. For example, we're doing that here and here.

Now, what's even more [00:04:30] interesting in the way that we're using this refining process, where we have the base RequestMapping annotation at the control log level, and then we have these individual annotations on the methods. As I was saying, what's even more interesting is that we're not even using the RequestMapping actual annotation on the methods. We're using the shorthands and that works just as well.

But why does that work? It works because, again, these [00:05:00] annotations are shorthands for RequestMapping. For example, if we step into one of these, we see the actual RequestMapping annotation defined here. And incidentally, we also see that the default method is considered GET. So because these are shorthands, we could actually replace these with a raw annotation. So, for example, instead of GET mapping here, we could just as well write... [00:05:30] the annotation like this. So we don't have to use the shorthands, but of course they're cleaner and easier to read.