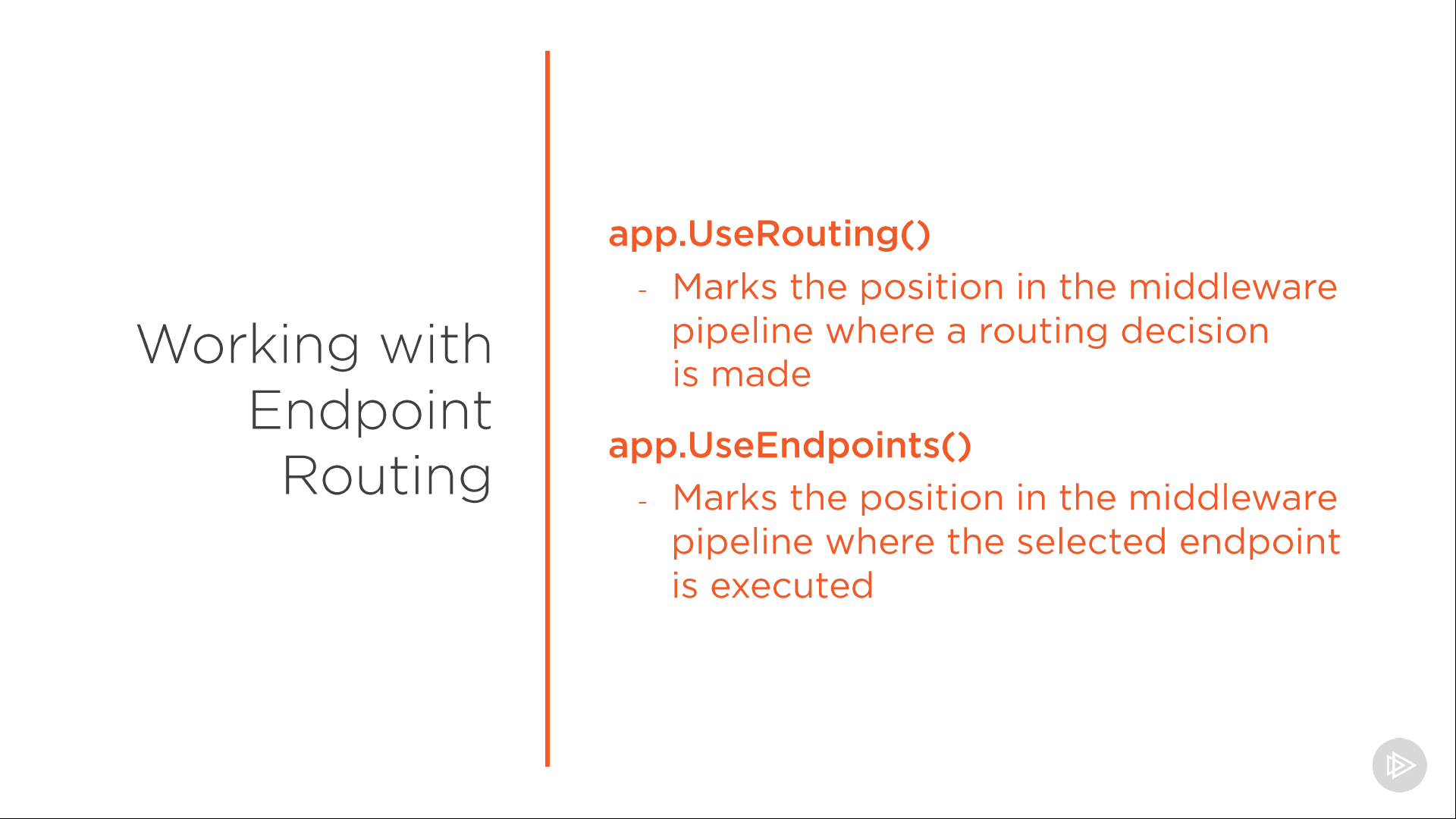
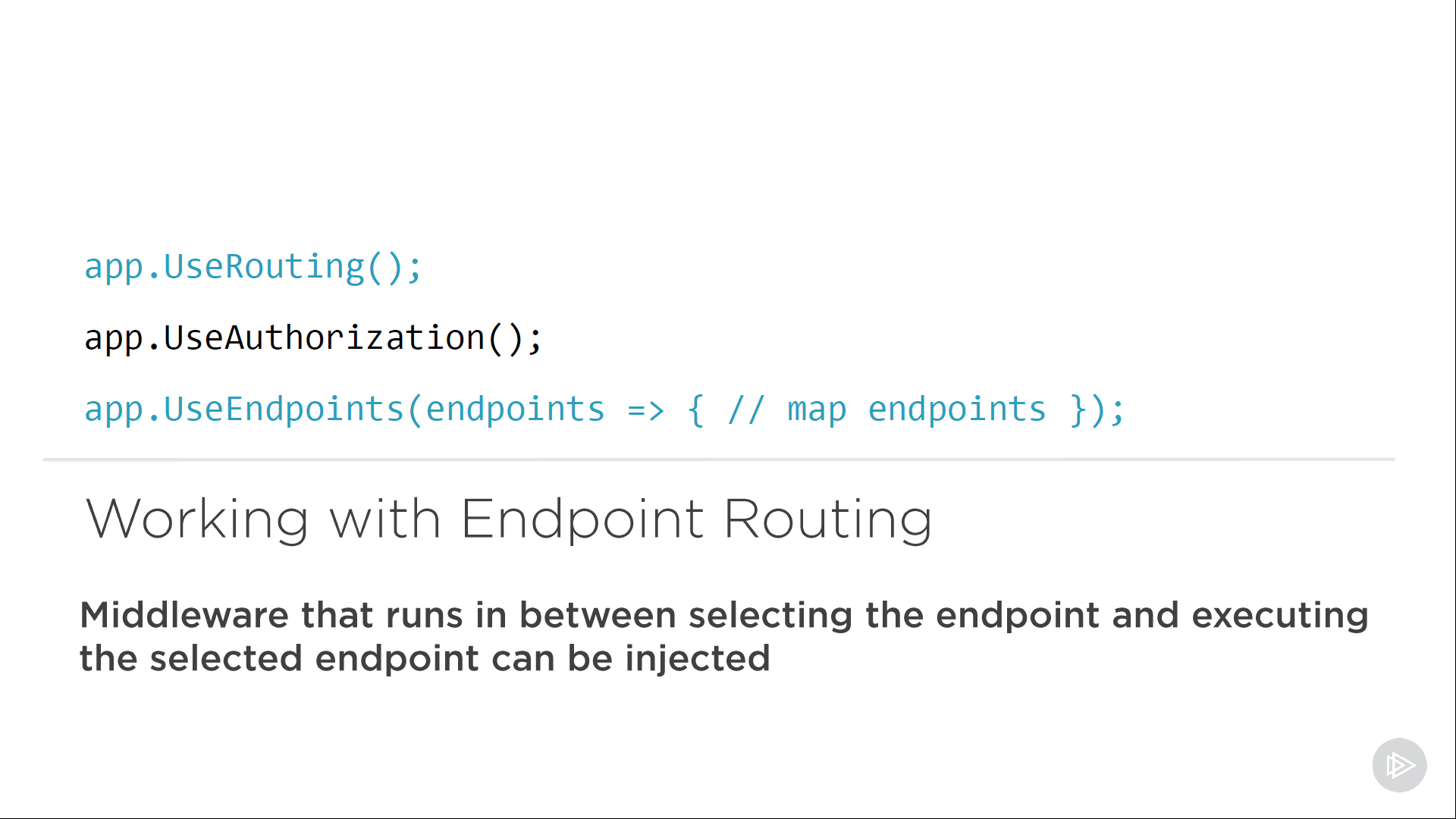


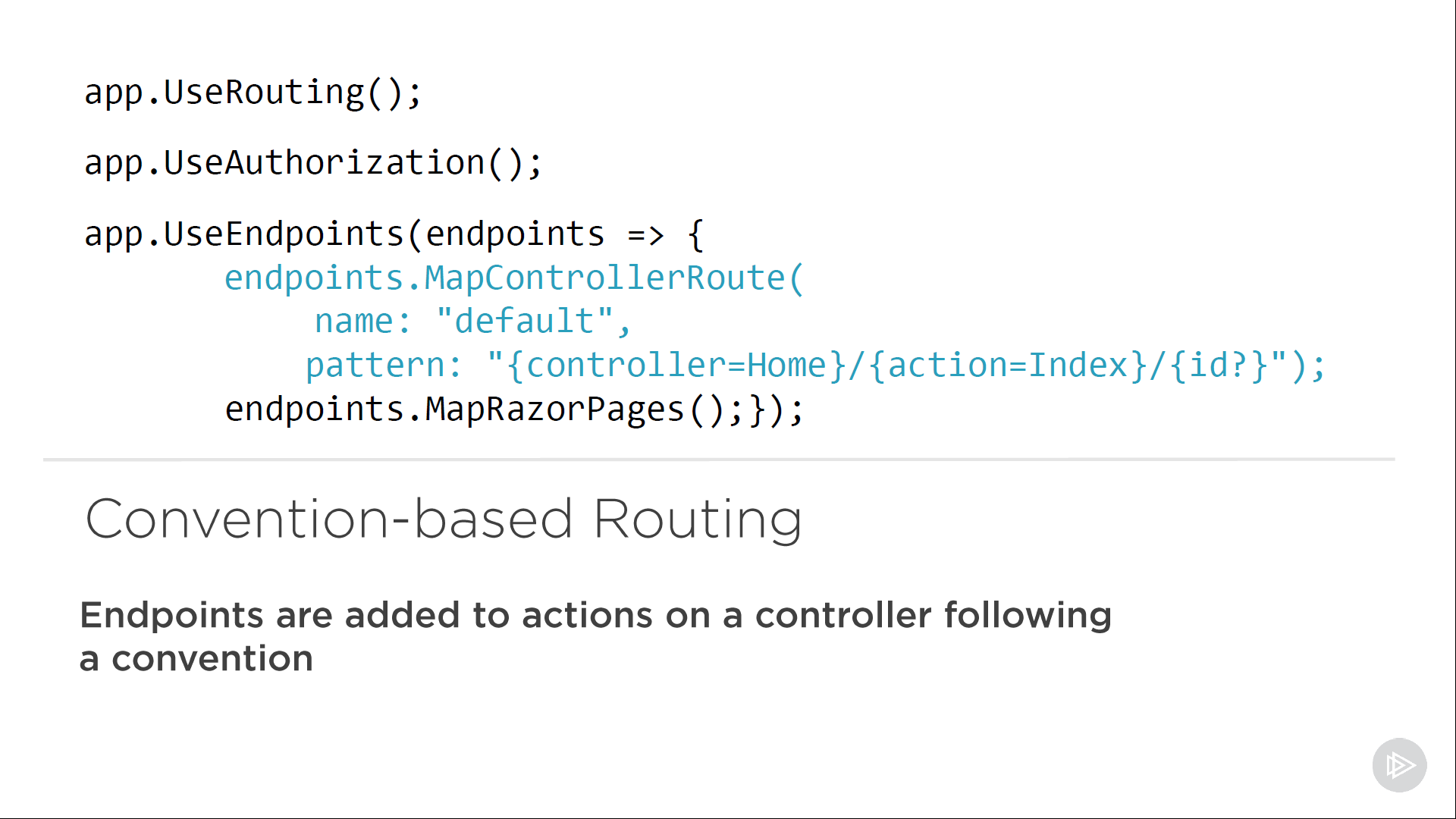
* Routing matches a request URI to an action on a controller.
* So once we send an HTTP request, the MVC framework parses the URI and tries to map it to an action on a controller.
* In ASP.NET Core 3, the preferred way to set this up is through **endpoint routing**.
* To set that up, two pieces of middleware must be injected in the request pipeline, app.UseRouting and app.UseEndpoints.
* app.UseRouting marks the position in the middleware pipeline where a routing decision is made. In other words, where an endpoint is selected.
* app.UseEndpoints marks the position in the middleware pipeline where the selected endpoint is executed.



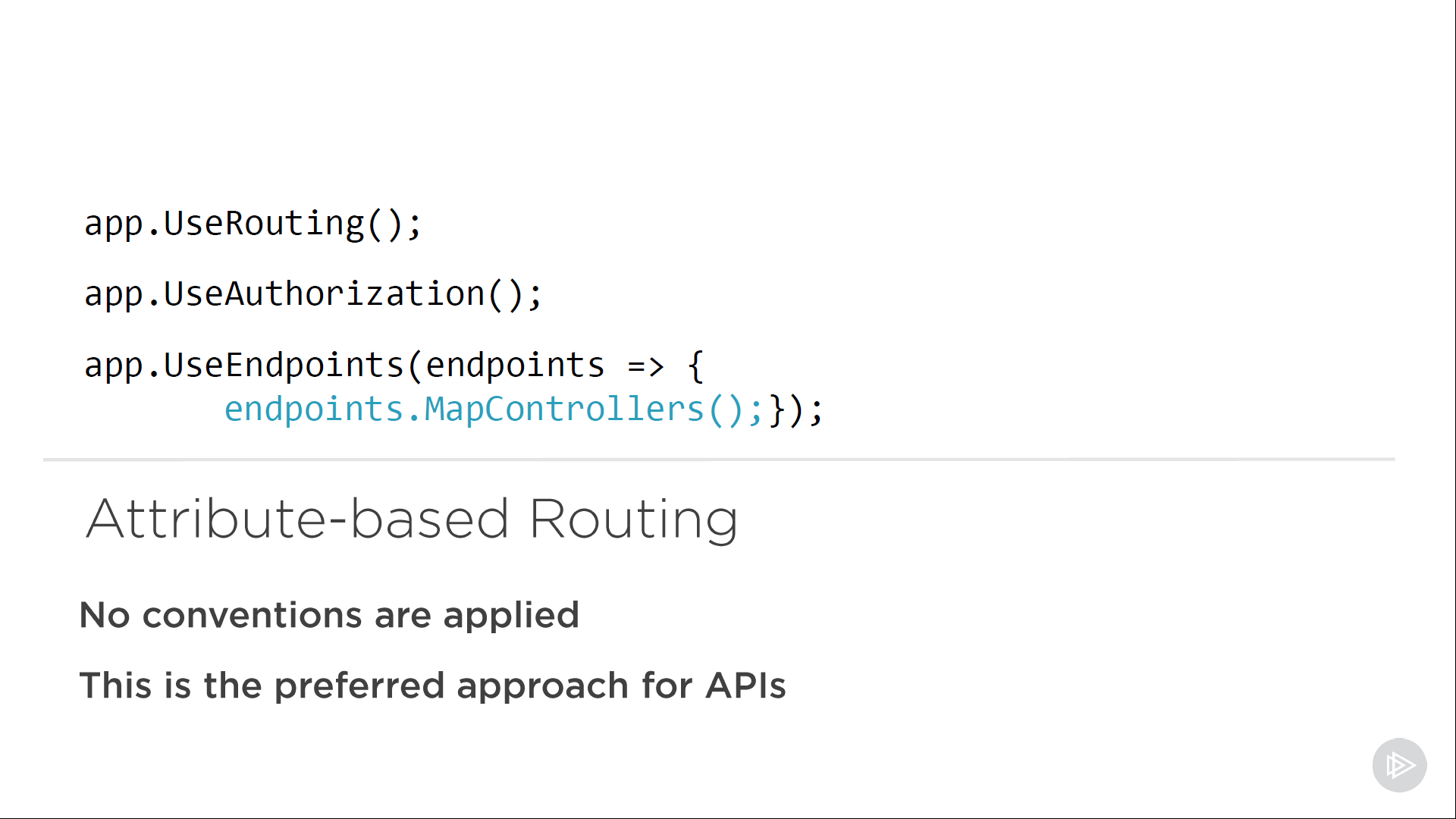
* In older ASP.NET Core versions, routing was setup when calling into useMVC or useRouting.
* The advantage of endpoint routing over that old approach is that we can now inject middleware that runs in between selecting the endpoint and executing the selected endpoint. In other words, we can inject pieces of middleware that know which endpoint was selected and can potentially select a different one. These can then see the selected endpoint if any and just use that, or they can select a different endpoint.
* A common example of this is the authentication middleware, as you can see on screen. It runs between UseRouting and UseEndpoints. We still need to map the endpoints through.



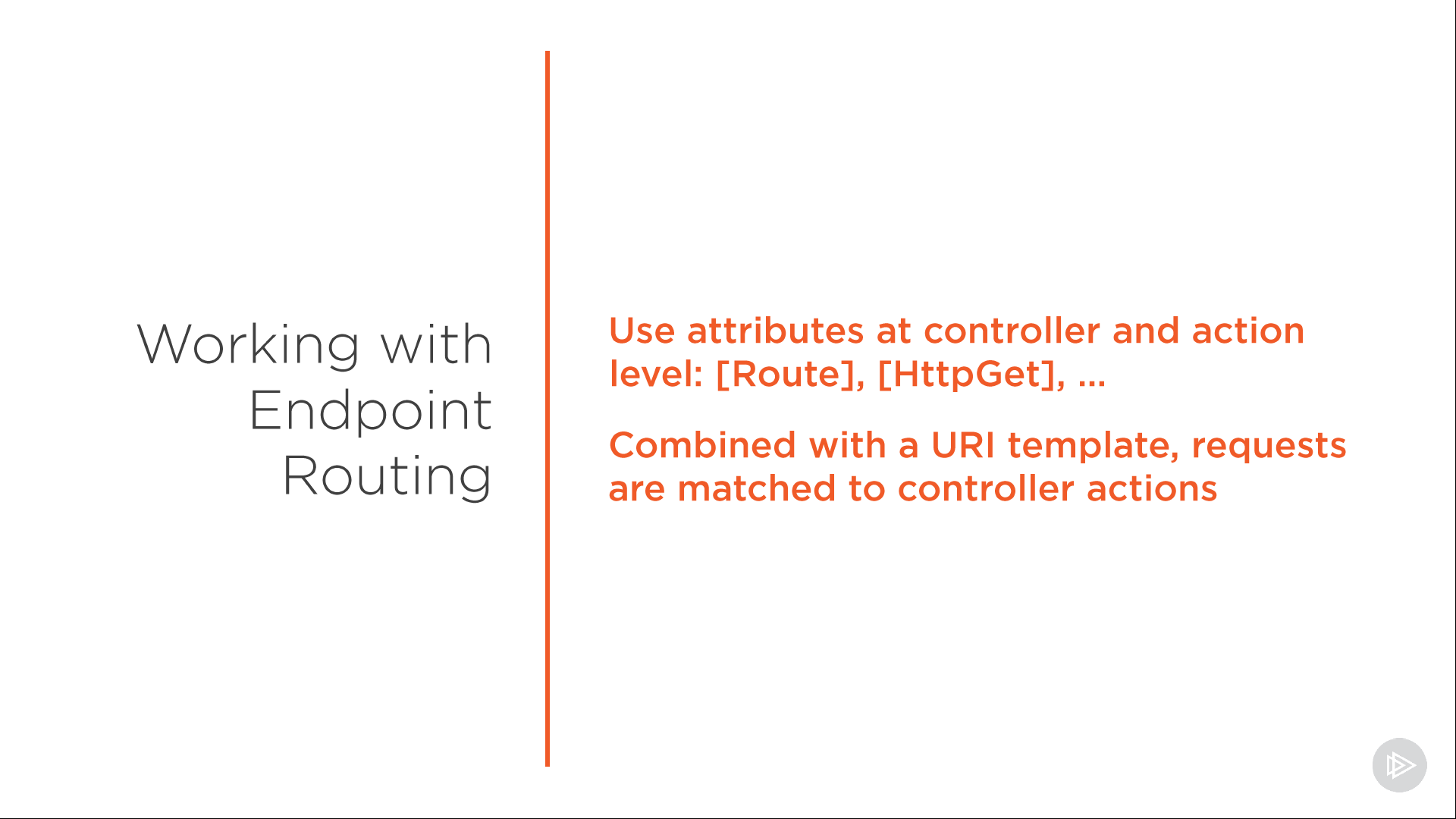
* There's two ways of setting these up,
  + convention based or
  + attribute based.
* Let's have a look at a convention-based example. Here we map URIs to actions in our controller following a convention. That's the pattern you see on screen.



* As you might guess from the code, for example, from the MapRazorPages statement, this type of routing is typically used for web applications. For APIs, the best practice is to use attribute-based routing.
* This is code from our current code base, which we got from the API template.



* MapControllers adds endpoints for our controller actions, but no routes are specified, i.e., there's no conventions applied. That means we have to set them up in another way, through attributes. Attribute-based routing, as the name implies, allows us to use attributes at controller and action level.



* We provide these with the URI template, and through that template and the attribute itself, a request is matched to a specific action on the controller. For this we use a variety of attributes depending on the HTTP method you want to match.