Instructor example

Web University by Harsha Vardhan

What is model binding in ASP.NET CORE?

Controllers and views need to work with data that comes from HTTP requests. For example, routes may provide a key that identifies a record, and posted form fields may provide model properties. The process of converting these string values to .NET objects could be complicated and something that you have to do with each request. Model binding automates and simplifies this process.

The model binding system fetches the data from multiple sources such as form fields, route data, and query strings. It also provides the data to controllers and views in method parameters and properties, converting plain string data to .NET objects and types in the process.

Example:

Let's say you have the following action method on the PostsController class:

```
[HttpGet("posts/{id}")]
public ActionResult<Post> GetById(int id, bool archivedOnly)
```

And the app receives a request with this URL:

`http://yourapp.com/api/Posts/5?ArchivedOnly=true`

After the routing selects the action method, model binding executes the following steps.

Locate the first parameter of GetByID, an integer named id, look through the available sources in the HTTP request and find id = "5" in route data.

Convert the string "5" into an integer 5.

Find the next parameter of GetByID, a boolean named archivedOnly.

Look through the sources and find "ArchivedOnly=true" in the query string. It ignores the case when matching the parameters to the strings.

Convert the string "true" into boolean true.

Some other examples of attributes include:

1. [FromQuery] - Gets values from the query string.

2. [FromRoute] - Gets values from route data.
3. [FromForm] - Gets values from posted form fields.
4. [FromBody] - Gets values from the request body.
5. [FromHeader] - Gets values from HTTP headers.

How validation works in ASP.NET CORE MVC and how they follow DRY principle?

ASP.NET CORE MVC supports the DRY (Don't Repeat Yourself) principle where you specify the behavior once and it reflects at multiple places in the application. Model Validation is one such example where you can specify validation rules by using data annotations in the model class and enforce the rules everywhere else (in controller and view) in the application.