Here's a concise summary of the three Spring MVC annotations:

**1. @RequestMapping**

* **Purpose**: Maps HTTP requests to handler methods in a controller.
* **Usage**: Can be used to define the URL pattern, HTTP method type (GET, POST), and additional properties like headers, params, etc.
* **Example**:

@RequestMapping("/home")

public String home() {

return "home";

}

**2. @RequestParam**

* **Purpose**: Binds request parameters (query or form) to method parameters.
* **Usage**: Extracts single values from HTTP requests (query params or form fields).
* **Example**:

@RequestMapping("/greet")

public String greet(@RequestParam("name") String name) {

return "Hello, " + name;

}

**3. @ModelAttribute**

* **Purpose**: Binds request parameters to a model object (typically a POJO or JavaBean).
* **Usage**: Used for binding form data to an object or adding common attributes to the model.
* **Example**:

@RequestMapping("/submit")

public String submitForm(@ModelAttribute User user) {

return "success";

}

**1. @RequestBody**

* **Purpose**: Binds the HTTP request body to a method parameter.
* **Usage**: Used to automatically deserialize JSON, XML, or other data formats from the request body into Java objects.
* **Example**:

@RequestMapping(value = "/create", method = RequestMethod.POST)

public String createUser(@RequestBody User user) {

// process user object

return "User created";

}

**2. @PathVariable**

* **Purpose**: Extracts values from URI path variables in the URL.
* **Usage**: Used to bind parts of the URI to method parameters.
* **Example**:

java

@RequestMapping("/user/{id}")

public String getUser(@PathVariable("id") int userId) {

return "User ID: " + userId;

}

**3. @CookieValue**

* **Purpose**: Binds a cookie value to a method parameter.
* **Usage**: Used to extract the value of a specific cookie from the request.
* **Example**:

@RequestMapping("/getCookie")

public String getCookieValue(@CookieValue("userSession") String userSession) {

return "Session ID: " + userSession;

}

These annotations allow you to access request body data, path variables, and cookie values in a clean and efficient way within Spring MVC controllers.

**@RequestHeader**

* **Purpose**: Binds a specific HTTP request header to a method parameter.
* **Usage**: Used to access HTTP headers in the request.
* **Example**:

java

Copy code

@RequestMapping("/header")

public String getHeader(@RequestHeader("User-Agent") String userAgent) {

return "User-Agent: " + userAgent;

}

**2. @ResponseBody**

* **Purpose**: Indicates that the return value of a method should be written directly to the HTTP response body.
* **Usage**: Used for sending data as JSON, XML, or other formats (commonly used in REST APIs).
* **Example**:

java

Copy code

@RequestMapping("/user")

@ResponseBody

public User getUser() {

return new User("John", "Doe");

}

**3. @ModelAttribute**

* **Purpose**: Binds request parameters to a model object, typically used in forms.
* **Usage**: Can be used for both binding form data to an object and adding attributes to the model.
* **Example**:

java

Copy code

@RequestMapping("/submit")

public String submitForm(@ModelAttribute User user) {

return "formSuccess";

}

These annotations enable you to access request headers, send response data directly, and bind form data or attributes to model objects in Spring MVC.