Spring Web Annotation

Аннотации

**@ControllerAdvice** – это аннотация указывает что класс является контроллером для обработки исключения в SpringBoot.

**Аннотация принимает несколько необязательных параметров:**

* **defaultValue** - значение по умолчанию для использования в качестве запасного варианта.
* **name** - имя файла cookie для привязки.
* **required** - требуется ли файл cookie.
* **value** - псевдоним для name().

**@ControllerAdvice**

**public class DefaultAdvice {**

**@ExceptionHandler(BusinessException.class)**

**public ResponseEntity<Response> handleException(BusinessException e) {**

**Response response = new Response(e.getMessage());**

**return new ResponseEntity<>(response, HttpStatus.OK);**

**}**

**}**

**@CookieValue –** это аннотация которая указывает на то, что параметр метода контроллера привязан к HTTP-файлу cookie.

**Аннотация принимает несколько необязательных параметров:**

* **defaultValue -** значение по умолчанию для использования в качестве запасного варианта.
* **name -** имя файла cookie для привязки.
* **required -** требуется ли файл cookie.
* **value** - псевдоним для name().

**@GetMapping("/profile")**

**public String profilePage(@CookieValue(name = "color") String color) {**

**return "You are using " + color + " mode.";**

**}**

**@CrossOrigin** – это аннотация позволяет выполнять запросы между разными источниками к аннотированным методам контроллера.

**Аннотация принимает несколько необязательных параметров:**

* **allowCredentials –** должен ли браузер отправлять учетные данные, такие как cookies, вместе с междоменными запросами на аннотированную конечную точку.
* **allowedHeaders -** список заголовков запроса, которые разрешены в реальных запросах, возможно "\*", чтобы разрешить все заголовки.
* **exposedHeaders –** список заголовков ответов, к которым пользовательский агент разрешит клиенту доступ при фактическом ответе, отличных от "простых" заголовков, т.е.
* **maxAge -** максимальный возраст (в секундах) длительности кэша для ответов перед запуском.
* **methods -** список поддерживаемых методов HTTP-запроса.
* **originPatterns -** альтернатива origins(), поддерживающая более гибкие исходные шаблоны.
* **origins -** список источников, для которых разрешены запросы из разных источников.
* **value –** псевдоним для origins().

**@CrossOrigin**

**@GetMapping("/{id}")**

**public Mono<Account> retrieve(@PathVariable Long id) {**

**// ...**

**}**

**@DeleteMapping** — это аннотация для сопоставления HTTP DELETE запросов с конкретными методами обработки.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - псевдоним для RequestMapping.consumes().
* **headers** - псевдоним для RequestMapping.headers().
* **name** - псевдоним для RequestMapping.name().
* **params** - псевдоним для RequestMapping.params().
* **path** - псевдоним для RequestMapping.path().
* **produces** - псевдоним для RequestMapping.produces().
* **value** - псевдоним для RequestMapping.value().

**@DeleteMapping(value = "/clients/{id}")**

**public ResponseEntity<?> delete(@PathVariable(name = "id") int id) {**

**final boolean deleted = clientService.delete(id);**

**return deleted**

**? new ResponseEntity<>(HttpStatus.OK)**

**: new ResponseEntity<>(HttpStatus.NOT\_MODIFIED);**

**}**

**@ExceptionHandler** — это аннотация для обработки исключений в определенных классах обработчиков и /или методах обработчика.

**Аннотация принимает несколько необязательных параметров:**

* **Value -** исключения обрабатываются аннотированным методом.

**@ExceptionHandler(NotFoundException.class)**

**public ResponseEntity<ErrorMessage> handleException(NotFoundException exception) {**

**return ResponseEntity**

**.status(HttpStatus.NOT\_FOUND)**

**.body(new ErrorMessage(exception.getMessage()));**

**}**

**@GetMapping** — это аннотация для сопоставления HTTP GET запросов с конкретными методами обработки.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - псевдоним для RequestMapping.consumes().
* **headers** - псевдоним для RequestMapping.headers().
* **name** - псевдоним для RequestMapping.name().
* **params** - псевдоним для RequestMapping.params().
* **path** - псевдоним для RequestMapping.path().
* **produces** - псевдоним для RequestMapping.produces().
* **value** - псевдоним для RequestMapping.value().

**@GetMapping("/hello")**

**public String handle() {**

**return "Hello WebFlux";**

**}**

**@InitBinder** — это аннотация которая идентифицирует методы, инициализирующие WebDataBinder, которые будут использоваться для заполнения аргументов объекта command и form аннотированных методов обработчика.

**Аннотация принимает несколько необязательных параметров:**

* **value -** имена атрибутов команды / формы и / или параметров запроса, к которым должен применяться этот метод инициализации-связывания.

**@InitBinder**

**public void initBinder(WebDataBinder binder) {**

**SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");**

**dateFormat.setLenient(false);**

**binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, false));**

**}**

**@Mapping** — это аннотация которая указывает на аннотацию веб-отображения.

**@MatrixVariable** — это аннотация которая указывает, что параметр метода должен быть привязан к паре имя-значение в сегменте пути. Поддерживается для RequestMapping аннотированных методов обработки.

**Аннотация принимает несколько необязательных параметров:**

* **defaultValue** - значение по умолчанию для использования в качестве запасного варианта.
* **name** - имя переменной матрицы.
* **pathVar** - имя переменной URI path, в которой расположена переменная matrix, если это необходимо для устранения неоднозначности (например
* **required** - требуется ли переменная matrix.
* **value** - псевдоним для name().

**@GetMapping("/pets/{petId}")**

**public void findPet(@PathVariable String petId, @MatrixVariable int q) {}**

**@ModelAttribute** — это аннотация которая привязывает параметр метода или возвращаемое значение метода к именованному атрибуту модели, доступному для веб-просмотра. Поддерживается для классов контроллеров с @RequestMapping методами.

* **Аннотация принимает несколько необязательных параметров:**
* **binding** - позволяет отключить привязку данных непосредственно к @ModelAttribute параметру метода или к атрибуту, возвращаемому из @ModelAttribute метода, оба из которых предотвратили бы привязку данных для этого атрибута.
* **name** - имя атрибута модели для привязки.
* **value** - псевдоним для name().

**@ModelAttribute**

**public void populateModel(@RequestParam String number, Model model) {**

**model.addAttribute(accountRepository.findAccount(number));**

**}**

**@PatchMapping** — это аннотация для отображения HTTP PATCH запросов на конкретные методы обработки.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - псевдоним для RequestMapping.consumes().
* **headers** - псевдоним для RequestMapping.headers().
* **name** - псевдоним для RequestMapping.name().
* **params** - псевдоним для RequestMapping.params().
* **path** - псевдоним для RequestMapping.path().
* **produces** - псевдоним для RequestMapping.produces().
* **value** - псевдоним для RequestMapping.value().

**@PatchMapping(value = "/{userId}", consumes = APPLICATION\_JSON\_VALUE, produces = APPLICATION\_JSON\_VALUE)**

**public UserResponse update(@PathVariable Integer userId, @RequestBody CreateUserRequest request) {**

**return userService.update(userId, request);**

**}**

**@PathVariable** — это аннотация которая указывает, что параметр метода должен быть привязан к переменной шаблона URI. Поддерживается для RequestMapping аннотированных методов обработчика.

**Аннотация принимает несколько необязательных параметров:**

* **name** - имя переменной path для привязки.
* **required** - требуется ли переменная path.
* **value** - псевдоним для name().

**@GetMapping("/users/{userId}")**

**public User getUser(@PathVariable(required = false) String userId) {**

**return user;**

**}**

**@PostMapping** — это аннотация для сопоставления HTTP POST запросов с конкретными методами обработки.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - псевдоним для RequestMapping.consumes().
* **headers** - псевдоним для RequestMapping.headers().
* **name** - псевдоним для RequestMapping.name().
* **params** - псевдоним для RequestMapping.params().
* **path** - псевдоним для RequestMapping.path().
* **produces** - псевдоним для RequestMapping.produces().
* **value** - псевдоним для RequestMapping.value().

**@PostMapping**

**@ResponseStatus(HttpStatus.CREATED)**

**public void add(@RequestBody Person person) {**

**// ...**

**}**

**@PutMapping** — это аннотация для сопоставления HTTP PUT запросов с конкретными методами обработки.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - псевдоним для RequestMapping.consumes().
* **headers** - псевдоним для RequestMapping.headers().
* **name** - псевдоним для RequestMapping.name().
* **params** - псевдоним для RequestMapping.params().
* **path** - псевдоним для RequestMapping.path().
* **produces** - псевдоним для RequestMapping.produces().
* **value** - псевдоним для RequestMapping.value().

**@PutMapping(value = "/update/{id}")**

**public String update(@PathVariable("id") int countryId, @RequestBody Country country) {**

**return "Country updated successfully";**

**}**

**@RequestAttribute** — это аннотация для привязки параметра метода к атрибуту запроса.

**Аннотация принимает несколько необязательных параметров:**

* **name** - имя атрибута запроса для привязки.
* **required** - требуется ли атрибут запроса.
* **value** - псевдоним для name().

**@GetMapping("/")**

**public String handle(@RequestAttribute Client client) {**

**// ...**

**}**

**@RequestBody** — это аннотация указывающая на параметр метода, должна быть привязана к телу веб-запроса. Тело запроса передается через HttpMessageConverter для разрешения аргумента метода в зависимости от типа содержимого запроса. Необязательно, автоматическая проверка может быть применена путем аннотирования аргумента с помощью @Valid.

**Аннотация принимает несколько необязательных параметров:**

* **required** - требуется ли содержимое тела.

**@PostMapping**

**@ResponseStatus(HttpStatus.CREATED)**

**public void add(@RequestBody Person person) {**

**// ...**

**}**

**@RequestHeader** — это аннотация которая указывает, что параметр метода должен быть привязан к заголовку веб-запроса.

**Аннотация принимает несколько необязательных параметров:**

* **defaultValue** - значение по умолчанию для использования в качестве запасного варианта.
* **name** - имя заголовка запроса для привязки.
* **required** - требуется ли заголовок.
* **value** - псевдоним для name().

**@RequestHeader("Keep-Alive") long keepAlive) {**

**//...**

**}**

**@RequestMapping** — это аннотация для отображения веб-запросов на методы в классах обработки запросов с гибкими сигнатурами методов.

**Аннотация принимает несколько необязательных параметров:**

* **consumes** - сужает первичное сопоставление по типам носителей, которые могут использоваться обработчиком сопоставления.
* **headers** - заголовки отображаемого запроса, сужающие основное сопоставление.
* **method** - методы HTTP-запроса для сопоставления, сужающие основное сопоставление: GET, POST, HEAD, OPTIONS, PUT, PATCH, DELETE, TRACE.
* **name** - присвойте имя этому сопоставлению.
* **params** - параметры отображаемого запроса, сужающие первичное отображение.
* **path** - URI для сопоставления путей
* **produces** - сужает первичное сопоставление по типам носителей, которые могут быть созданы обработчиком сопоставления.
* **value** - первичное сопоставление, выраженное этой аннотацией.

**@RestController**

**@RequestMapping("/persons")**

**class PersonController {**

**@GetMapping("/{id}")**

**public Person getPerson(@PathVariable Long id) {**

**// ...**

**}**

**@RequestMethod** — это аннотация указывает на то что это перечисление методов HTTP-запроса. Предназначен для использования с RequestMapping.method() атрибутом RequestMapping аннотации.

**Аннотация принимает несколько необязательных параметров:**

**Enum.EnumDesc<E extends Enum<E>> -** DELETE, GET, HEAD, OPTIONS, PATCH, POST, PUT, TRACE.

**val info = RequestMappingInfo.paths("/user/{id}").methods(RequestMethod.GET).build()**

**@RequestParam** — это аннотация которая указывает, что параметр метода должен быть привязан к параметру веб-запроса.

**Аннотация принимает несколько необязательных параметров:**

* **defaultValue** - значение по умолчанию, используемое в качестве запасного варианта, когда параметр запроса не указан или имеет пустое значение.
* **name** - имя параметра запроса для привязки.
* **required** - является ли параметр обязательным.
* **value** - псевдоним для name().

**@GetMapping**

**public String setupForm(@RequestParam("petId") int petId, Model model) {**

**Pet pet = this.clinic.loadPet(petId);**

**model.addAttribute("pet", pet);**

**return "petForm";**

**}**

**@RequestPart** — это аннотация которая может использоваться для связывания части запроса "multipart / form-data" с аргументом метода.

**Аннотация принимает несколько необязательных параметров:**

* **name** - имя части в "multipart/form-data" запросе, к которому выполняется привязка.
* **required** - требуется ли эта часть.
* **value** - псевдоним для name().

**@PostMapping("/")**

**public String handle(@RequestPart("meta-data") Part metadata, (1)**

**@RequestPart("file-data") FilePart file) {**

**// ...**

**}**

**@ResponseBody** — это аннотация указывающая на возвращаемое методом значение, должна быть привязана к телу веб-ответа.

**@GetMapping("/accounts/{id}")**

**@ResponseBody**

**public Account handle() {**

**// ...**

**}**

**@ResponseStatus** — это аннотация помечает метод или класс исключений статусом code() и reason(), который должен быть возвращен.

**Аннотация принимает несколько необязательных параметров:**

* **code** - код состояния, который будет использоваться для ответа.
* **reason** - причина, которую следует использовать для ответа.
* **value** - псевдоним для code().

**@PostMapping**

**@ResponseStatus(HttpStatus.CREATED)**

**public void add(@RequestBody Person person) {**

**// ...**

**}**

**@RestController** — это аннотация которая сама по себе аннотируется с помощью @Controller и @ResponseBody.

**Аннотация принимает несколько необязательных параметров:**

* **value** - значение может указывать на предложение для имени логического компонента, которое должно быть преобразовано в компонент Spring bean в случае автоматически определяемого компонента.

**@RestController**

**public class HelloController {**

**@GetMapping("/hello")**

**public String handle() {**

**return "Hello WebFlux";**

**}**

**}**

**@RestControllerAdvice** – это аннотация указывает что класс является контроллером для обработки исключения которая сама по себе аннотируется с помощью @ControllerAdvice и @ResponseBody..

**Аннотация принимает несколько необязательных параметров:**

* **аnnotations** - массив аннотаций.
* **assignableTypes** - массив классов.
* **basePackageClasses** - типобезопасная альтернатива basePackages() для указания пакетов, в которых следует выбирать контроллеры, которые будут рекомендованы @RestControllerAdvice аннотированным классом.
* **basePackages** - массив базовых пакетов.
* **мalue** - псевдоним для basePackages() атрибута.

**@RestControllerAdvice**

**public class ExceptionApiHandler {**

**@ExceptionHandler(MethodArgumentTypeMismatchException.class)**

**public ResponseEntity<ErrorMessage> mismatchException(MethodArgumentTypeMismatchException exception) {**

**return ResponseEntity**

**.status(HttpStatus.NOT\_FOUND)**

**.body(new ErrorMessage(exception.getMessage()));**

**}**

**@SessionAttribute** — это аннотация для привязки параметра метода к атрибуту сеанса.

**Аннотация принимает несколько необязательных параметров:**

* **name** - имя атрибута сеанса для привязки.
* **required** - требуется ли атрибут session.
* **value** - псевдоним для name().

**@GetMapping("/")**

**fun handle(@SessionAttribute user: User): String {**

**// ...**

**}**

**@SessionAttributes** — это аннотация указывающая атрибуты сеанса, которые использует конкретный обработчик.

**Аннотация принимает несколько необязательных параметров:**

* **names** - имена атрибутов сеанса в модели, которые должны храниться в сеансе или каком-либо диалоговом хранилище.
* **types** - типы атрибутов сеанса в модели, которые должны храниться в сеансе или каком-либо диалоговом хранилище.
* **value** - псевдоним для names().

**@Controller**

**@SessionAttributes("pet")**

**class EditPetForm {**

**// ...**

**}**

**@ValueConstants** — это аннотация которая обозначает общие константы значений, разделяемые аннотациями привязки.

**Аннотация принимает несколько необязательных параметров:**

* **DEFAULT\_NONE** - константа, определяющая значение без значения по умолчанию - в качестве замены null, которое мы не можем использовать в атрибутах аннотации.