|  |  |
| --- | --- |
| 1 | Vẽ và trình bày cách Spring MVC xử lý request     * Khi có một request đến thì DispatcherServlet (Front controller) sẽ đón nhận. * DispatcherServlet sẽ sử dụng HandlerMapping để xác định gửi requesr đến controller tương ứng. Sau đó controller sẽ làm việc với model để kiểm tra và lấy dữ liệu. * Sauk hi có kết quả thì controller sẽ gửi trả dữ liệu cho Front controller và dữ liệu sẽ được render đến View Template tương ứng. Và View Template sẽ được gửi trả về Front Controller, rồi gửi về Client. |
| 2 | **Trình bày cơ chế Dependency Injection trong Spring**   * Dependency injection là một cơ chế để triển khai IoC. * Là cơ chế giảm sự phụ thuộc giữa 2 Module (ex: SumController, SumService), 2 module này sẽ không phụ thuộc trực tiếp vào nhau mà thông qua abstract (ISumService) |
| 3 | **Có bao nhiêu cách để thực hiện Dependency Injection? Trình bày?**  - Constructor injection: các dependency đc cung cấp thông qua constructor của class.  - Setter injection: Các dependency (biến phụ thuộc) sẽ được truyền vào 1 class thông qua các setter method (hàm setter).  - Interface injection: Dependency sẽ cung cấp một Interface, trong đó có chứa hàm có tên là inject. Rồi bạn muốn inject gì đó thì gọi cái hàm inject(xx) ra. |
| 4 | **Framework là gì? Framework khác Library chỗ nào?**  - Frame work là các ứng dụng phần mềm có tính trừu tượng (bộ khung) mà có các quy tắc có sẵn lập trình viên sẽ dựa vào để phát triển ứng dụng của mình.  - Diểm khác biệt lớn nhất giữa Framework và Library đó chính là cơ chế “Inversion of Control”  • Với Library: Ứng dụng nắm quyền điều khiển (control)  • VớiFramework: Framework nắm quyền điều khiển |
| 5 | **Spring Framework là gì ?**   * Spring Framework là một framework ứng dụng và bộ chứa đảo ngược điều khiển cho nền tảng Java. Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: Dependency Injection và lập trình hướng khía cạnh ( Aspect Oriented Programming ). |
| 6 | Lợi ích của Spring Framework?   * Bài 1 |
| 7 | **Nguyên lý đảo ngược quyền điều khiển (Inversion of Control) là gì ?**  -IoC là một nguyên lý trong phát triển phần mềm, trong đó việc điều khiển các đối tượng hoặc các thành phần của hệ thống được thực hiện bởi framework hoặc các container  - IoC cho phép framework nắm giữ quyền điều khiển luồng thực thi của hệ thống và gọi các mã nguồn khác |
| 8 | **Bean là gì?**  Spring Bean là các object trong Spring Framework, được khởi tạo thông qua Spring Container. Bất kỳ class Java nào cũng có thể là Spring Bean nếu nó được cấu hình và khởi tạo thông qua container bằng việc cung cấp các thông tin cấu hình (các file config .xml, .properties..) |
| 9 | Trong Spring có bao nhiêu Bean Scope?   * Có 5 scope được định nghĩa cho Spring Bean: * Singleton: Chỉ duy nhất một thể hiện của bean sẽ được tạo cho mỗi container. Đây là scope mặc định cho spring bean. Khi sử dụng scope này cần chắc chắn rằng các bean không có các biến/thuộc tính được share. * Prototype: Một thể hiện của bean sẽ được tạo cho mỗi lần được yêu cầu(request) * Request: giống với prototype scope, tuy nhiên nó dùng cho ứng dụng web, một thể hiện của bean sẽ được tạo cho mỗi HTTP request. * Session: Mỗi thể hiện của bean sẽ được tạo cho mỗi HTTP Session * Global-Session: Được sử dụng để tạo global session bean cho các ứng dụng Portlet.   Trong 5 scope trên thì 3 scope cuối chỉ dùng trong ứng dụng web. |
| 10 | **@Autowire là gì?**   * **@Autowired** là cơ chế tiêm đối tượng vào trong class. * khi dùng @Autowired thì nó sẽ lấy đối tượng có sẵn trong ApplicationContext để tiêm vào class |
| 11 | **@Component có ý nghĩa gì?**   * Class được đánh dấu với annotation @Component thì sẽ được tạo thành 1 bean, khi dự án Spring được chạy nó sẽ quét qua và tạo 1 bean cho class đó. |
| 12 | **Trình bày ý nghĩa của Front Controller**   * Tiếp nhận request từ client, sử dụng cơ chế HandlerMapping để gửi Request đến Controller tương ứng. Sau đó tiếp nhận dữ liệu trả về từ Controller, render sử liệu đến View Template tương ứng, rồi nhận kết quả từ View Template và trả về Client. * Spring MVC cung cấp các annotation như @Controller và @RestController để khai báo các controller |
| 13 | **Trình bày ý nghĩa của ModelAndView Class**   * ModelAndView đại diện cho một view cùng với các dữ liệu sử dụng trong view đó |
| 14 | **Trình bày ý nghĩa của ModelMap Interface**   * Lớp con của Linked Hash Map * Giống Model và ModelAndView đều dùng để gửi dữ liệu từ controller lên views nhưng có thêm cái các phương thức của Map |
| 15 | **Trình bày ý nghĩa của ViewResolver Interface**  ViewResolver là cơ chế để xử lý tầng view của Spring MVC  ViewResolver ánh xạ tên của view sang đối tượng view tương ứng. Được xác định dựa trên prefix, suffix  Có nhiều triển khai khác nhau của ViewResolver:  • XmlViewResolver: Giúp xem file viết bằng XML  • UrlBasedViewResolver: Giúp ánh xạ trực tiếp URL tới tên logic của view. |
| 16 | **Phân biệt POST với PUT thường sử dụng để làm gì?**   * Phương thức POST để tạo dữ liệu còn PUT thì để cập nhật lại dữ liệu. * PUT chính là phương thức bảo toàn dữ liệu nên khi chúng ta ấn submit bao nhiêu lần đi chăng nữa thì dữ liệu cũng chỉ tính như chúng ta submit 1 lần. Còn đối với POST thì chúng ta submit quá nhiều lần thì có thể nó sẽ cho ra những kết quả khác nhau. |
| 17 | **@RequestMapping làm gì?**   * Annotation @RequestMapping được sử dụng để ánh xạ các request tới các action tương ứng của controller * @RequestMapping bao gồm một số thuộc tính để ánh xạ đến:   • URL  • HTTP Method  • Các tham số  • Các headers  • Media types   * @RequestMapping có thể áp dụng cho lớp hoặc phương thức |
| 18 | **Trình bày cơ chế Data Binding trong Spring**   * Data Binding là cơ chế liên kết dữ liệu đầu vào (hoặc đầu ra) với các đối tượng model * Sử dụng Data Binding, các form đều được liên kết với một đối tượng biểu diễn dữ liệu ở phía sau * Khi tương tác với form, dữ liệu trên form sẽ được tự động chuyển đổi thành các thuộc tính của đối tượng liên kết với nó * Data Binding hỗ trợ chuyển đổi dữ liệu (data conversion) và validate dữ liệu |
| 19 | **Thuộc tính consumes trong các Request Mapping là gì ?**   * **consumes** định dạng kiểu dữ liệu đầu vào (xml, JSON) * **produces** định dạng kiểu dữ liệu đầu ra (xml, JSON) |
| 20 | **Thymeleaf là gì?**  Thymeleaf là một Java template engine (bộ công cụ)dùng để xử lý View hoạt động trên HTML, XML, Javascript, CSS ,text và RAW. |
| 21 | **Sử dụng lặp trong Thymeleaf như thế nào? Sử dụng điều kiện trong Thymeleaf như thế nào?**   * Vòng lặp: th: each * Điều kiện: th: if, th:unless or th:switch, th:case |
| 22 | **Formatter là gì ? Converter là gì ?**   * Converter và Formatter hỗ trợ chuyển đổi kiểu dữ liệu này sang kiểu dữ liệu khác * Tuy nhiên, kiểu dữ liệu nguồn của Formatter là String, trong khi đó converter có thể làm việc với bất cứ kiểu dữ liệu nguồn nào. Formatter phù hợp hơn với tầng web (web-tier), còn converter thì có thể sử dụng ở bất cứ tầng nào. |
| 23 | **ORM là gì ?**   * ORM (Object-Relational Mapping) là kỹ thuật liên kết giữa các đối tượng trong lập trình với các đối tượng trong CSDL |
| 24 | **JPA là gì? Spring Data JPA là gì ?**   * JPA (Java Persistence API ) là đặc tả của Java dành cho các thao tác với dữ liệu * Spring Data JPA là một module của Spring Framework. Mục đích của Spring Data JPA là giảm thiểu việc thực hiện quá nhiều bước để có thể implement được JPA.   •Spring Data Repository định nghĩa một interface chính tên là Repository. Interface này nắm bắt entity cần quản lý và kiểu dữ liệu id của entity đó.  •Interface CrudRepository kế thừa từ interface Repository, chứa các phương thức thông dụng dành cho các thao tác CRUD.  •Interface PagingAndSortingRepository cung cấp các phương thức hỗ trợ choviệc phân trang và sắp xếp các entity. |
| 25 | **Hibernate là gì ? Hibernate làm gì với database**   * Hibernate là một trong những ORM Framework, dung để triển khai ORM. * Hibernate giúp lập trình viên Java có thể map các object (POJO) với database, và hỗ trợ thực hiện các khái niệm lập trình hướng đối tượng với cơ dữ liệu quan hệ. * Ngoài ra Hibernate giúp lưu trữ và truy vấn dữ liệu quan hệ mạnh mẽ và nhanh. Hibernate cho phép truy vẫn dữ liệu thông qua Java Persistence API (JPA) hoặc bằng HQL hoặc bằng SQL thuần (Native SQL). |
| 26 | **Liệt kê một số annotation của hibernate?**  - @Entity  - @Table  - @Column  - @Id  - @GeneratedValue  - @ManyToOne  - @OneToMany  - @OneToOne  - @ManyToMany |
| 27 | **Entity là gì?**   * Entity là đối tượng đại diện cho dữ liệu ở trong ứng dụng * Entity sẽ được ánh xạ (mapping) tới một bảng trong CSDL * Trong JPA, một entity cần tuân thủ:   • Được gắn với annotation javax.persistence.Entity  • Có một constructor là publicvà không có tham số  • Không được khai báo final  • Các biến đối tượng cần được khai báo là private, protected hoặc ở mức package-private |
| 28 | **Persistence Context & Entity Manager là gì?**  - Entity Manager cung cấp các phương thức thao tác với các entity(CRUD)  - Persistence quản lý các entity |
| 29 | **Câu lệnh truy vấn động là gì? Câu lệnh truy vấn tĩnh là gì?** |
| 30 | Trạng thái của Entity bao gồm những gì?(bỏ) |
| 31 | **Có bao nhiêu loại mapping trong Hibernate?**  @OneToMany  @ManyToOne  @OneToOne  @ManyToMany |
| 32 | **SessionFactory là gì?**   * Là một interface giúp tạo ra session kết nối đến database bằng cách đọc các cấu hình trong một file xml |
| 33 | **Validation dữ liệu là gì?**   * Validation thực hiện đánh giá/ xác minh tính hợp lệ của dữ liệu đầu vào. * Các tầng validate dữ liệu   + UserInterface(UI): Html, JS.  **+ Business layer: Java, PHP**  + Data layer : database |
| 34 | **Trình bày cách triển khai validate dữ liệu trong Spring**  - Có 2 cách triển khai validate:  + Dùng các anotation để validate: @NotNull, @NotBlank ,@NotEmplty, @Size, @Min, @Max…  + Custom Validate (Tự định nghĩa)  B1: implement Validator vào Dto  B2: Định nghĩa method validate  B3: gọi method validate ở controller để kiểm tra dữ liệu |
| 35 | **Binding Result là gì?**  BindingResult là đối tượng của Spring mà giữ kết quả xác thực và ràng buộc và chứa các lỗi có thể xảy ra (đặt bên cạnh đối tượng cần valid) |
| 36 | AOP là gì? AOP gồm những yếu tố nào? - Aspect Oriented Programming (AOP)– lập trình hướng khía cạnh(Bổ trợ cho OOP) – để giải quyết cho công việc được lặp đi lặp lại và không liên quan đến nghiệp vụ của project như: logging, transaction, security **- AOP gồm những yếu tố nào**  **+** Aspect: Khai báo 1 khối độc lập với OOP  + Advice: Các hành động mà AOP dùng để quan sát OOP  + Joinpoint: Các điểm gây ra các hành động ở advice  + Pointcut: Tập hơn các Joinpoint có khả năng gây ra các hành động ở advice |
| 37 | **Các loại Advice?**  • Before Advice: thực thi trước join point.  • After Returning Advice: thực thi sau khi joint point hoàn thành bình  thường  • After Throwing Advice: thực thi nếu joint point kết thúc do có ngoại  lệ được tung  • After (finally) Advice: thực thi sau khi joint point hoàn thành bất kể dưới điều kiện nào  • Around Advice: thực thi cả trước và sau joint point |
| 38 | **JoinPoint là gì?**  + Joinpoint: Các điểm gây ra các hành động ở advice |
| 39 | **Nêu cách triển khai AOP mà bạn dùng trong dự án**   * Định nghĩa 1 Aspect: đặt @Aspect trên đầu class * Định nghĩa 1 Pointcut * Khai báo advice |
| 40 | **Xử lý ngoại lệ trong Spring Web MVC như thế nào?**   * Sử dụng try-catch. Phương pháp này thường để xử lý các ngoại lệ tại request (Controller). * Sử dụng @ExceptionHandler để ủy quyền cho front controller xử lý ngoại lệ. * Sử dụng AOP . Afterthrowing * Sử dụng HandlerExceptionResolver. Phương pháp này được dùng để xử lý ngoại lệ tại servlet. * Sử dụng trang báo lỗi. Phương pháp này được dùng để bắt các fatal error hay các ngoại lệ sinh ra ngoài phạm vi của Spring. |
| 41 | **Cookie là gì?**   * Cookie là các tệp văn bản được lưu trữ trên Client side và chúng được giữ cho mục đích theo dõi sử dụng. Tập lệnh máy chủ gửi một bộ Cookie đến trình duyệt. Ví dụ: tên, tuổi hoặc số nhận dạng,…. Trình duyệt lưu thông tin này trên máy khách để sử dụng trong tương lai. * Khi trình duyệt lần sau gửi bất kỳ yêu cầu nào đến máy chủ web thì nó sẽ gửi thông tin Cookie đó đến máy chủ và máy chủ sử dụng thông tin đó để nhận dạng người dùng. * Thành phần dùng để lưu trữ dữ liệu và được lưu ở server, khi tắt trình duyệt thì vòng đời |
| 42 | **Phân biệt Session và Cookie**   |  |  |  | | --- | --- | --- | |  | **Session** | **Cookie** | | **What** | Là 1 tệp dữ liệu được lưu trữ bất kì kiểu dữ liệu nào. Bộ nhớ tối đa là 128MB theo mặc định | Chỉ lưu trữ được kiểu String .Kích thước Cookie tối đa là 4KB | | **When** | Một Session kết thúc khi người dùng đóng trình duyệt | Cookie kết thúc tùy thuộc vào thời gian bạn đặt cho nó | | **Where** | Lưu trữ ở server & tồn tại ở JSESSIONID | Lưu trữ ở Client , trình duyệt | | **Security** | Không thể chỉnh sửa | Có thể chỉnh sửa được | |
| 43 | **Nêu cách triển khai Session mà bạn dùng trong dự án** -Tạo: Dùng @SessionAttributes kết hợp với @ModelAttribute khi cả 2 trùng tên thì session sẽ được tạo ra-Sử dụng:+Dùng @SessionAttribute đặt làm tham số của từng phương thức+Dùng @SessionAttributes trên đầu các class controller mà muốn sử dụng session |
| 44 | **Web Service là gì? Lấy ví dụ về web service**   * Là dịch vụ bao gồm các tiêu chuẩn và giao thức để các hệ thống và ứng dụng khác có thể trao đổi được với nhau * Ví dụ: Google Maps API là một dịch vụ web có thể được sử dụng bởi các trang web để hiển thị bản đồ bằng cách chuyển tọa độ tới đó. |
| 45 | **Phân biệt Web Service và Web truyền thông** |
| 46 | **SOAP là gì? RESTful là gì?**  Các loại web service cơ bản   * SOAP (Simple Object Access Protocol) là giao thức sửdụngXML để định nghĩa dữ liệu dạng thuần văn bản(plain text) và truyền dữl iệu thông qua HTTP. * REST (Representational State Transfer) là một kiểu cấu trúc cung cấp các quy tắc để xây dựng web service.   • REST định nghĩa dữ liệu dưới dạng XML hoặc JSON (JavaScript Object Notation) và truyền thông qua mạng internet sử dụng giao thức HTTP.  • Các web service xây dựng dựa trên REST được gọi là RESTful, chúng chủ yếu nhằm xử lý các hoạt động CRUD (Create/ Read/ Update/ Delete) trên dữ liệu. |
| 47 | **jQuery là gì? Ajax là gì?**   * jQuery là thư viện của Java Script   $(selecter).event();   * Ajax là kỹ thuật ở JQuery dùng để xử lý bất đồng bộ * Asynchronous JavaScript and XML, là một nhóm các công nghệ phát triển web được sử dụng để tạo các ứng dụng web giàu tính tương tác, nhanh hơn và mượt mà hơn với sự giúp đỡ của XML, HTML, CSS và JavaScript. |
|  | **Nêu ví dụ về cách bạn triển khai Ajax trong dự án của bạn**  $.ajax({         headers: {             'Accept': 'application/json',             'Content-Type': 'application/json'         },         type: "POST",         data: JSON.stringify(newSmartphone),         //tên API         url: "/smartphones",         //xử lý khi thành công         success: successHandler(Đây là để gọi lại function nha)         error: failHandler(Đây là để gọi lại function nha)     });  headers: Quy định kiểu dữ liệu gửi đi và trả về  type: Phương thức truyền đi  data: Nội dùng truyền đi  url: đường dẫn cần truyền tới  success: Xử lý khi thành công  error: Xử lý khi thất bại |
| 50 | **I18n và L10n là gì?**   * Internationalization: Là quá trình thiết kế một ứng dụng phần mềm để nó có thể đáp ứng được nhiều ngôn ngữ và khu vực khác nhau mà không cần thay đổi kỹ thuật. * Localization: Là quá trình điều chỉnh phần mềm đã được quốc tế hóa cho một ngôn ngữ hoặc một khu vực cụ thể bằng cách chỉ định ngôn ngữ hoặc khu vực sau đó dịch văn bản. |
| 51 | **Nêu cách triển khai I18N và I10N trong dự án Spring**  - Tạo class AppConfiguration implements WebMvcConfigurer  @Override  -addInterceptors(): tự động phân tích tham số lang đi kèm các request để xác định bản địa hiện tại  @Bean  -MessageSource: setBaseName(tên file.properties)  -LocaleResolver: setDefaultLocale(new Locale( lấy thông tin về bản địa từ session của người dùng))  Tạo file.properties điền các message cần đặt tên  ->Qua html gọi message bằng #{message} |
| 52 | **Interceptor là gì?**  Được sử dụng để “chặn”, hay thao-tác-trước đối với request và thao-tác-sau đối với response để chuyển đổi hoặc sử dụng thông tin được chứa trong request hoặc response đó.. |
| 53 | **Bảo mật là gì? Cơ chế bảo mật trong Spring như thế nào?**  Là bảo vệ dữ liệu của bạn từ bất kỳ hình thức truy cập trái phép,bảo vệ dữ liệu an toàn trên môi trường trực tuyến. Trong Spring có cơ chế xác thực (Authentication) và cơ chế phân quyền (Authorization) |
| 54 | **Authentication là gì ? Authorization là gì ?**  - Authentication: là quá trình xác thực thông tin người dùng có được truy cập vào hệ thống  - Authorization: là quá trình phân quyền vói từng account người dùng. |
| 55 | **Liệt kê các cơ chế xác thực**  •HTTP Basic: là một phương thức để client cung cấp username và password khi thực hiện yêu cầu.  •Cookies(khi người dùng đăng nhập lần đầu thì sử dụng cooki để đăng nhập lại)  •Tockens(JWT JSON Web Token)  •Signature(JWT JSON Web Token)  •One-time password: có nghĩa là mật khẩu chỉ sử dụng một lần |
| 56 | **CSRF là gì**  -**CSRF:** Là cách tấn công bằng cách mượn quyền chứng thực từ người dùng để làm thay đổi trang web mà họ có account |
| 57 | **CORS là gì**  - Cross-Origin Resource Sharing (CORS) là một tiêu chuẩn về bảo mật có ở các web browser thế hệ mới.  - CORS là một cơ chế xác nhận thông qua Header của request. Cụ thể là trên Server sẽ nói với browser về quy định chấp nhận những request từ domain nào và phương thức ra sao (GET, POST, PUT, v.v..) |
| 58 | **Spring Boot là gì?**   * Là một module của Spring FrameWork, cung cấp 1 tính năng gọi là RAD (Rapid Application Development), dùng để tạo ra một ứng dụng có rất ít config, k cấu hình bất kì xml nào, có sử dụng quy ước về thiết kế theo cấu hình. |
| 59 | **Sự khác nhau giữa SpringBoot và SpringMVC là gì?**   |  |  | | --- | --- | | **Spring Boot** | **Spring MVC** | | * Tích hợp sẵn tomcat * Cung cấp sẵn thư viện * Tự động cấu hình * Không cần file config(xml; java) * Khó fix bug * Có class application và application.properties | * Không có tích hợp tomcat * Không cung cấp * Phải tự cấu hình * Cần phải có file config * Dễ fix bug hơn 1 tí nhưng vẫn khó * Không có | |
| 60 | **Cấu hình cho Spring Boot Tìm kiếm các Bean ở nhiều package khác nhau bằng cách nào?**  Dùng @ComponentScan đặt trên class  Dùng @Bean đặt trên method để khai báo các Bean |
| 61 | Trình bày Spring Security? @EnableWebSecurity làm gì?  - Authentication: là quá trình xác thực thông tin người dùng có được truy cập vào hệ thống  - Authorization: là quá trình phân quyền vói từng account người dùng. |
| 62 | Trình bày Spring Boot JPA? Cài đặt?  -Spring Boot JPA là một phần trong hệ sinh thái Spring Data, nó tạo ra một layer ở giữa tầng service và database, giúp chúng ta thao tác với database một cách dễ dàng hơn, tự động config và giảm thiểu code thừa thãi.  -Spring Boot JPA đã wrapper Hibernate và tạo ra một interface mạnh mẽ.  Chọn Spring Date JPA trong phần SQL |
| 63 | Để sử dụng interface JpaRepository cần cung cấp những thông tin nào?  Đối tượng và kiểu dữ liệu id của đối tượng đó. |
| 64 | Bạn có biết annotation @ConfigurationProperties? Trình bày?  Cho phép truy xuất các thuộc tính được đặt trong các tệp cấu hình 1 cách tự động và mạnh mẽ |