|  |  |  |  |
| --- | --- | --- | --- |
| 애너테이션의 종류 조사 | | | |
| 순서 | **애너테이션** | **정의** | **패키지** |
| 1 | **@Component** | **개발자가 직접 작성한 Class를 Bean으로 등록하기 위한 애노테이션 이다.** | **org.springframework.stereotype.Component;** |
| 2 | **@Bean** | **개발자가 직접 제어가 불가능한 외부 라이브러리 등을 Bean으로 만들려고 할 때 사용** | **org.springframework.context.annotation.Bean;** |
| 3 | **@Autowired** | **속성,setter,생성자 에서 사용하며, Type에 따라 알아서 Bean을 주입해준다.** | **org.springframework.beans.factory.annotation.Autowired;** |
| 4 | **@Inject** | **@Autowired와 비슷한 기능을 한다.** | **javax.inject-x.x.x.jar 파일이 추가** |
| 5 | **@Controller** | **Spring의 Controller를 의미한다.** | **org.springframework.stereotype.Controller;** |
| 6 | **@RestController** | **@Controller 에 @ResponseBody가 추가된 것으로, 컨트롤러 클래스의 각 메서드마다 @ResponseBody를 추가할 필요가 없어진다.** | **org.springframework.stereotype.Controller;** |
| 7 | **@Service** | **비즈니스 로직을 수행하는 Class라는 것을 나타내는 용도이다.** | **org.springframework.stereotype.Service;** |
| 8 | **@Repository** | **DAO class에 쓰이며, DataBase에 접근하는 method를 가지고 있는 Class에서 쓰인다.** | **org.springframework.stereotype.Repository;** |
| 9 | **@Cacheable** | **method앞에 지정해서 해당 method를 최초에 호출하면 캐시에 적재하고 다음부터는 동일한 method호출이 있을 때 캐시에서 결과를 가져와 return하므로 methode 호출 횟수를 줄여주는 애노테이션이다.** | **org.springframework.cache.annotation.Cacheable;** |
| 10 | **@CachePut** | **캐시를 업데이트하기 위해서 메서드를 항상 실행하게 강제하는 애노테이션이다.** | **org.springframework.cache.annotation.CachePut;** |
| 11 | **@RequestHeader** | **요청에서 특정 HTTP헤더 정보를 추출할 때 사용한다.** | **org.springframework.stereotype** |
| 12 | **@PathVariable** | **현재의 URI에서 원하는 정보를 추출할 때 사용하는 애노테이션이다.** | **org.springframework.stereotype** |
| 13 | **@CookieValue** | **현재 사용자의 쿠키가 존재하는 경우 쿠키의 이름을 이용해서 쿠키의 값을 추출한다.** | **org.springframework.stereotype** |
| 14 | **@Required** | **Setter method에 적용해주면 Bean생성시 필수 프로퍼티 임을 알린다.** | **org.springframework.beans.factory.annotation.Required;** |
| 15 | **@Qualifier(“BeanID”)** | **@Autowired와 같이 쓰이며 같은 타입의 Bean 객체가 있을 때 해당 아이디를 적어 원하는 Bean이 주입될 수 있도록 한다.** | **rg.springframework.beans.factory.annotation.Qualifier;** |
| 16 | **@Resource** | **@Autowired와 마찬가지로 Bean 객체를 주입해준다. 이때 차이점은 Autowired는 타입으로, Resource는 이름으로 연결해준다.** | **org.springframework.context.annotation. Resource;** |
| 17 | **@PreDestory** | **객체를 제거하지 전에 해야할 작업을 수행하기 위해 사용한다.** | **javax.annotation** |
| 18 | **@PropertySource** | **해당 프로퍼티 파일을 Environment로 로딩하게 해준다.** | **org.springframework.context.annotation.PropertySource;** |
| 19 | **@Lazy** | **Class가 로드될 때 스프링에서 바로 Bean등록을 마치는 것이 아니라 실제로 사용될 때 로딩이 이뤄지게 하는 방법이다.** | **org.springframework.context.annotation.Lazy;** |
| 20 | **@Value** | **properties에서 값을 가져와 적용할 때 사용한다.** | **org.springframework.beans.factory.annotation.Value;** |
| 21 | **@Valid** | **유효성 검증이 필요한 객체임을 지정한다.** | **org.springframework.validation.annotation.Validated;** |
| 22 | **@RequestBody** | **요청이 온 데이터를 바로 Class나 model로 매핑하기 위한 애노테이션이다.** | **org.springframework.stereotype** |
| 23 | **@Primary** | **@Autowired ,@Component 와 같이 사용되며 객체 생성에 우선권을 부여 한다.** | **org.springframework.context.annotation.Primary;** |
| 24 | **@Scheduled** | **정해진 시간에 실행해야 하는 경우에 사용한다.** | **org.springframework.scheduling.annotation.Scheduled;** |
| 25 | **@Configuration** | **xml문서의 시작 부분에 있는 <bean>를 대신하는 것으로, 설정하고자 하는 클래스 선언부 앞에 사용한다.** | **org.springframework.context.annotation.Configuration;** |