@RequestBody

이 어노테이션이 붙은 파라미터에는 http요청의 본문(body)이 그대로 전달된다.

일반적인 GET/POST의 요청 파라미터라면 @RequestBody를 사용할 일이 없을 것이다.

반면에 xml이나 json기반의 메시지를 사용하는 요청의 경우에 이 방법이 매우 유용하다.

HTTP 요청의 바디내용을 통째로 자바객체로 변환해서 매핑된 메소드 파라미터로 전달해준다.

```
@RequestMapping(value = "/ajaxTest.do")
public String ajaxTest(@RequestBody UserVO getUserVO) throws Exception {

  System.out.println(getUserVO.getId());

  return "test/login.tiles";
}
```

@ResponseBody

자바객체를 HTTP요청의 바디내용으로 매핑하여 클라이언트로 전송한다.

@ResponseBody 가 붙은 파라미터가 있으면 HTTP요청의 미디어타입과 파라미터의 타입을 먼저 확인한다.

* dispatcher-servlet.xml 의 <annotation-drvien> 태그 내에 선언하는 <message-converter> 에서 확인.

```
<mvc:annotation-driven>
    <mvc:message-converters>
        <bean class="org.springframework.http.converter.StringHttpMessageConverter">
            <property name="supportedMediaTypes">
                <list>
                    <value>text/html;charset=UTF-8</value>
                </list>
            </property>
        </bean>
    </mvc:message-converters>
</mvc:annotation-driven>
```

메세지 변환기 중에서 해당 미디어타입과 파라미터 타입을 처리할 수 있다면, HTTP요청의 본문 부분을 통째로 변환해서 지정된 메소드 파라미터로 전달해준다.

```
@ResponseBody
@RequestMapping(value = "/ajaxTest.do")
public UserVO ajaxTest() throws Exception {

  UserVO userVO = new UserVO();
  userVO.setId("테스트");
```

```
    return userVO;
}
```

즉, @Responsebody 어노테이션을 사용하면 http요청 body를 자바 객체로 전달받을 수 있다.

*** @RestController**

@Controller와는 다르게 @RestController는 리턴값에 자동으로 @ResponseBody가 붙게되어 별도 어노테이션을 명시해주지 않아도 HTTP 응답데이터(body)에 자바 객체가 매핑되어 전달 된다.

@Controller인 경우에 바디를 자바객체로 받기 위해서는 @ResponseBody 어노테이션을 반드시 명시해주어야한다.