Spring Security Testing With JUnit

Maven Dependencies

```xml
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-test</artifactId>
    <version>4.1.4.RELEASE</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-test</artifactId>
    <version>4.0.0.RELEASE</version>
    <scope>test</scope>
</dependency>
```

Notice that, above dependencies will be used when test phase because we set dependency's **scope as test**

Example

```java
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = {
     "classpath:/applicationContext.xml",
     "classpath:/mvc-dispatcher-servlet.xml",
     "classpath:/spring-security.xml"})
     @WebAppConfiguration
     public class SpringSecurityTest {

         private MockMvc mockMvc;
         @Autowired
         private WebApplicationContext wac;
         private MockMultipartFile file;

         @Before
         public void setup() throws Exception {
             this.mockMvc= MockMvcBuilders
             .webAppContextSetup(wac)
             .apply(springSecurity()).build();

             initMockMultipartFile();
         }

         private void initMockMultipartFile() throws IOException {
             File thisClassFile = new File("C:\test.txt");
             file = new MockMultipartFile("file[]", thisClassFile.getName(),
             "text/plain",IOUtils.toByteArray(new FileInputStream(thisClassFile)));
         }
```

```java

    @Test
    public void testFormLoginWithWrongPassword() throws Exception {
        Map<String, Object> sessionAttrs = new HashMap<String, Object>();
        sessionAttrs.put("rand1", "5");
        sessionAttrs.put("rand2", "5");
        mockMvc.perform(post("/admin/login")
        .param("username", "myuce")
        .param("password", "test")
        .param("captcha", "10")
        .with(csrf())
        .sessionAttrs(sessionAttrs))
        .andExpect(status().isFound())
        .andExpect(redirectedUrl("/admin/index?error=true"));
    }

    @Test
    public void testAdminLogout() throws Exception {
        mockMvc.perform(logout("/yonetim/logout"));
    }

    @Test
    @WithMockUser(username = "myuce", roles = {"USER", "ADMIN"})
    public void testFileUpload() throws Exception {
        mockMvc.perform(MockMvcRequestBuilders.fileUpload("/admin/home/file-upload")
        .file(file)
        .with(csrf()))
        .andExpect(status().is(200))
        .andDo(print())
        .andExpect(content().string("success"));
    }

    @Test
    public void loginAuthenticationToken() throws Exception {
        assertNotNull(getToken());
    }

    @Test
    public void testSendMailWithoutToken() throws Exception {
        assertEquals(401, sendMail(null).getStatus());
    }

    @Test
    public void testSendMail() throws Exception {
        String token = getToken();
        assertNotNull(token);
        MockHttpServletResponse response = sendMail(token);
        assertEquals(200, response.getStatus());
        assertEquals("true", response.getContentAsString());
    }

    private String getToken() throws Exception {
        return getClientLoginHttpResponse().getHeader("token");
    }

    private MockHttpServletResponse getClientLoginHttpResponse() throws Exception {
        return mockMvc.perform(post("/login")
        .param("username", "myuce")
        .param("password", "19871987"))
        .andExpect(status().isOk())
        .andDo(print())
        .andReturn()
        .getResponse();
    }

    private MockHttpServletResponse sendMail(String token) throws Exception {
        return mockMvc.perform(MockMvcRequestBuilders.fileUpload("/send-mail")
            .file(file)
            .param("token", token)
            .param("to", "test@gmail.com")
            .param("title", "ŞÖMSDLFMSD")
            .param("content", "sfşmsdfa"))
            .andReturn()
            .getResponse();

        }
    }
```

For more information https://spring.io/blog/2014/05/07/preview-spring-security-test-method-security