

# Spring Security Özelleştirmeleri

# Spring Security Özelleştirmeleri

- Spring Security oldukça **esnek** ve uygulamalara göre **özelleşmeye açık** bir framework'tür
- Kimliklendirme ve yetkilendirme **süreçlerinin hemen her adımı** uygulamaya özgü gereksinimlere göre özelleştirilebilir
- Custom login form, custom AuthenticationProvider, Custom UserDetailsService ve UserDetails sınıfları, Filter zincirinde değişiklik **en yaygın özelleştirme örnekleridir**

# Custom Login Form

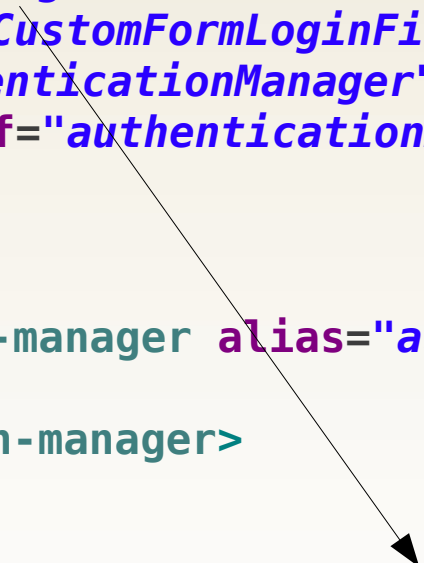
- Örneğin, login formunda kullanıcı adı, şifrenin yanı sıra birde parola sorulabilir
- Ya da captcha özelliği eklenmesi istenebilir
- Bu gibi ihtiyaçlar genellikle **UsernamePasswordAuthenticationFilter** sınıfından extend ederek karşılanabilir
- **UsernamePasswordAuthenticationFilter.attemptAuthentication()** metodu **override** edilerek özelleştirme yapılabilir

# Custom Login Form

```
<beans>
  <bean id="customizedFormLoginFilter"
        class="x.y.z.CustomFormLoginFilter">
    <property name="authenticationManager"
              ref="authenticationManager"/>
    ...
  </bean>

  <security:authentication-manager alias="authenticationManager">
    ...
  </security:authentication-manager>

  <security:http>
    <security:custom-filter ref="customizedFormLoginFilter"
                          position="FORM_LOGIN_FILTER"/>
  </security:http>
</beans>
```



## AuthProvider

- Uygulamaya özel, örneğin, kullanıcının eriştiği istemci makinanın belirli bir IP grubunda olması gibi Authentication kontrolleri eklenebilir
- Bu gibi durumlarda **DaoAuthProvider** sınıfını extend etmek, ya da **AuthProvider** arayüzünü implement etmek gerekebilir
- **DaoAuthProvider.additionalAuthenticationChecks()** metodu override edilerek bu kontroller yapılabilir


# Custom AuthenticationProvider

```
<beans>
  <security:authentication-manager>

    <security:authentication-provider
      ref="customAuthProvider"/>

  </security:authentication-manager>

  <bean id="customAuthProvider"
    class="x.y.z.CustomAuthenticationProvider">
    ...
  </bean>
</beans>
```



# Filter Zincirine Ekleme

- Örneğin, belirli durumlar için kimliklendirmenin bypass edilmesi istenebilir
- Ya da kullanıcı authenticated iken login sayfasına eriştiğinde ana sayfaya yönlendirilmesi istenebilir
- Yeni bir **Filter yazılması** ve **springSecurityFilterChain'e** eklenmesi gerekebilir

# Filter Zincirine Ekleme

```

<beans>
  <bean id="authByPassFilter" class="x.y.z.AuthByPassFilter">
    ...
  </bean>

  <bean id="mainPageRedirectFilter"
        class="x.y.z.MainPageRedirectFilter">
    ...
  </bean>

  <security:http auto-config="true">
    ...
    <security:custom-filter ref="authByPassFilter"
                           before="FORM_LOGIN_FILTER" />

    <security:custom-filter ref="mainPageRedirectFilter"
                           after="SECURITY_CONTEXT_FILTER" />

  </security:http>
</beans>

```



# Custom UserDetails ve UserDetailsService

- Örneğin, uygulama kendine ait bir User tipini kullanmak isteyebilir
- Ya da Authentication bilgisinin bir kısmı DB'den diğer bir kısmı LDAP veya başka bir realm'den gelebilir
- Bu gibi durumlarda **UserDetails** ve **UserDetailsService** arayüzlerinin implement edilmesi gerekecektir

# Custom UserDetails ve UserDetailsService

```
public class CustomUser implements UserDetails {  
    ...  
}
```

```
@Service  
@Transactional  
public class CustomUserDetailsService implements UserDetailsService {  
  
    @PersistenceContext  
    private EntityManager entityManager;  
  
    @Override  
    public UserDetails loadUserByUsername(String username)  
        throws UsernameNotFoundException {  
        try {  
            CustomUser user = entityManager.createQuery(  
                "from User u where u.username = :username", CustomUser.class)  
                .setParameter("username", username).getSingleResult();  
            return user;  
        } catch (Exception ex) {  
            throw new UsernameNotFoundException(  
                "User not found with username :" + username, ex);  
        }  
    }  
}
```

# Custom UserDetails ve UserDetailsService

```
<beans>
  <security:authentication-manager>

    <security:authentication-provider
      user-service-ref="customUserDetailsService">

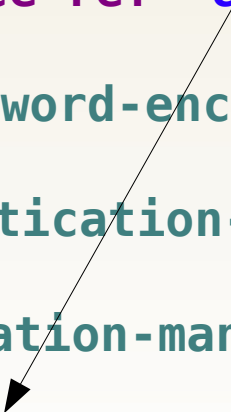
      <security:password-encoder hash="sha"/>

    </security:authentication-provider>

  </security:authentication-manager>

  <bean id="customUserDetailsService"
    class="x.y.z.CustomUserDetailsService">

  ...
</bean>
</beans>
```



# İletişim

- **Harezmi** Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- <http://www.java-egitimleri.com>
- [info@java-egitimleri.com](mailto:info@java-egitimleri.com)

