


form 기반의 인증 // 프론트 엔드가 담당

jsp 말고 vue나 react 는 다른 기술이 있다 security는

KB금융그룹 | 국민의평생 금융파트너  백엔드로 ajax로 하는 것은 jwt

설정이 90%다

템플릿만 잘 짜면  
개꿀

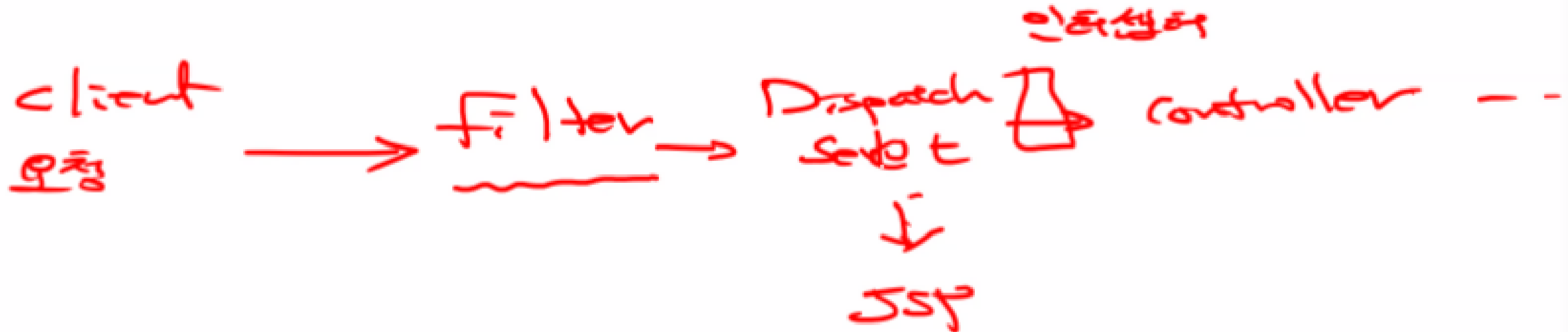
2025년 상반기 K-디지털 트레이닝

# Spring Web Security 소개

security는  
서비스와 독립적으로  
하는게  
제일 베스트이다.

기존에는 filter이나 인터셉터에서 인증 보안을 체크했음.

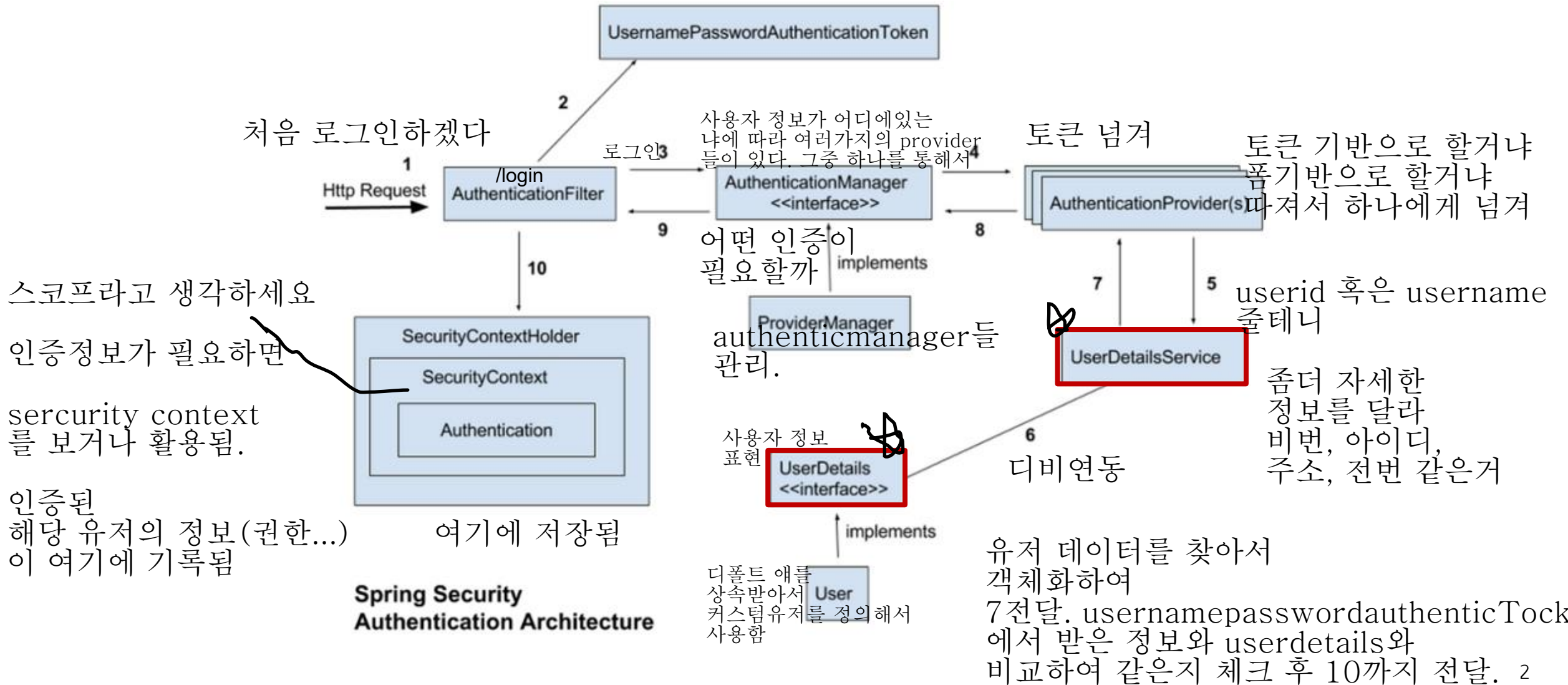
**[KR] IT's Your Life** 오늘 배우는 기술에서 보안 필터는 기존에 있던 필터보다 앞에서 동작한다



# Spring Web Security 설정

개발자가 관여하는 파트는 userdetailservice와 userdetail를 만들어적용하면돼 나머지는 만들어져있음  
인터페이스를 구현하거나 기존 클래스를 상속받아 완성시키면 돼

## 스프링 시큐리티 아키텍처



## ✓ 프로젝트 생성

- Template: MyBatisSpringLegacy
- name: secserver

## ✏ settings.gradle ↴

```
rootProject.name = "secserver"
```

## build.gradle ✓

```
...
ext {
    junitVersion = '5.9.2'
    springVersion = '5.3.37'
    ...
    springSecurityVersion='5.8.13' ✓
}
```

```
dependencies {
    ...
```

// 보안

```
implementation("org.springframework.security:spring-security-web:${springSecurityVersion}")
implementation("org.springframework.security:spring-security-config:${springSecurityVersion}")
implementation("org.springframework.security:spring-security-core:${springSecurityVersion}")
implementation("org.springframework.security:spring-security-taglibs:${springSecurityVersion}")
...
}
```

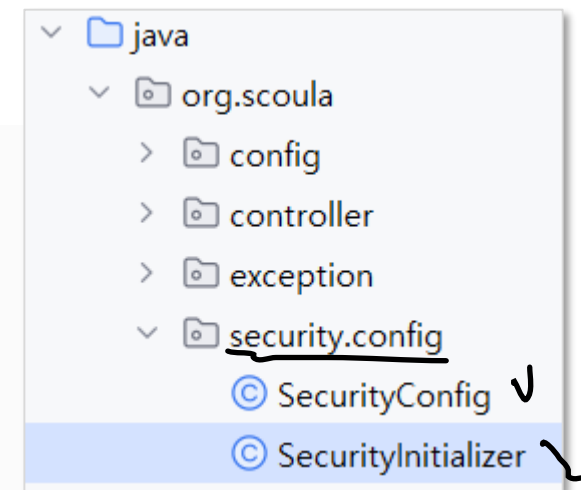
} servlet용

jsp용

sec태그 활용

## SecurityInitializer.java

```
package org.scoula.security.config;  
  
import org.springframework.security.web.context.AbstractSecurityWebApplicationInitializer;  
  
public class SecurityInitializer extends AbstractSecurityWebApplicationInitializer {  
필수  
}
```



security filter의 연결 순서 등을 지정하는  
필수적인 요소.  
내용이 없어도 만들어 봐야함

## SecurityConfig.java

```
package org.scoula.security.config;
```

```
@Configuration
```

```
@EnableWebSecurity
```

```
@Log4j2
```

```
public class SecurityConfig extends WebSecurityConfigurerAdapter {
```

```
    @Override
```

```
    public void configure(HttpSecurity http) throws Exception {
```

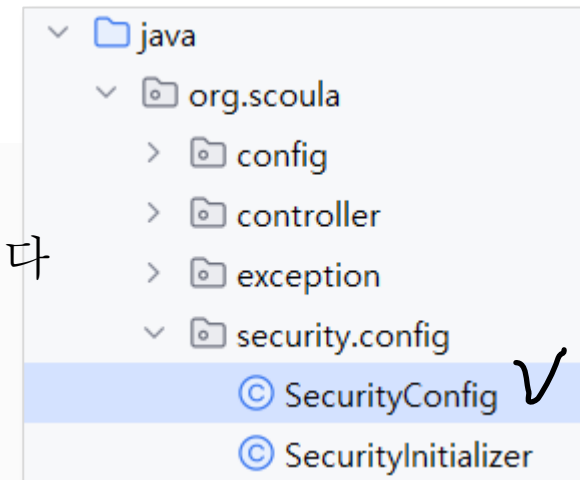
```
    }
```

```
}
```

접미어 어댑터가 붙으면  
인터페이스를 기본적으로 구현한 클래스.

접미어 adapter가 붙으면  
디폴트 메서드가 구현되어있다  
오버라이드할거 몇개만  
다시 정의하면 된다.

실질적인 설정 객체



## config/WebConfig.java

```
@Override
```

```
public Class<?>[] getRootConfigClasses() {
```

```
    return new Class[] { RootConfig.class, SecurityConfig.class };
```

```
}
```

추가

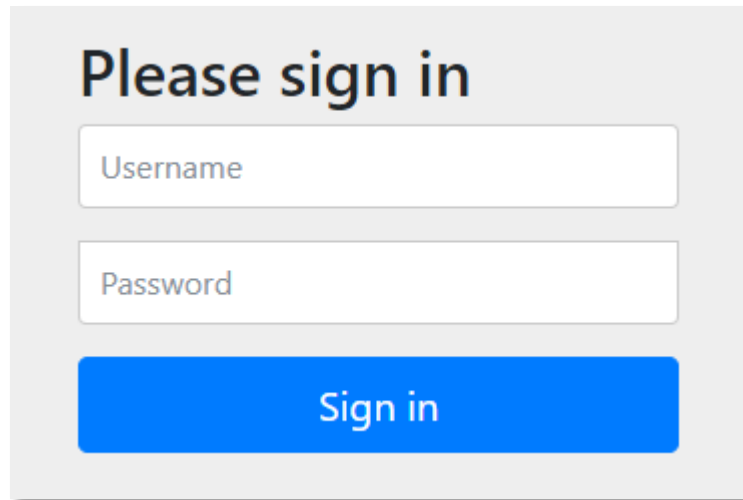
기본 설정이 모든 요청에서  
인증이 됐는지 확인

안되어있으며지정 페이지로 리다이렉트함

## ✓ 실행

- 모든 페이지는 보안정책에 따라 접근
- 스프링 시큐리티가 설정되면
  - 디폴트 : 모든 페이지는 로그인을 해야 접근 가능
  - /login 페이지로 리다이렉트됨

<http://localhost:8080/login>



디폴트 로그인 페이지

✓ 인증      몇가지 용어

401에러

- 자신을 증명하는 것

- 자기 스스로가 무언가 자신을 증명할 만한 자료를 제시

제시해야함

✓ 인가

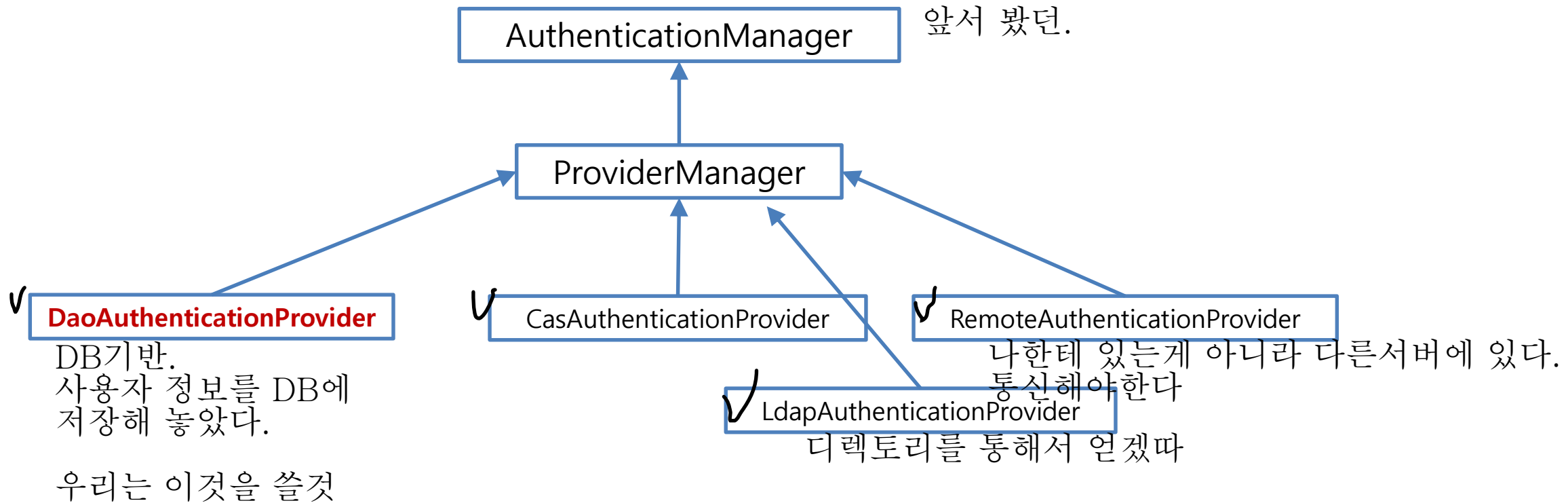
403에러

- 남에 의해서 자격이 부여됨



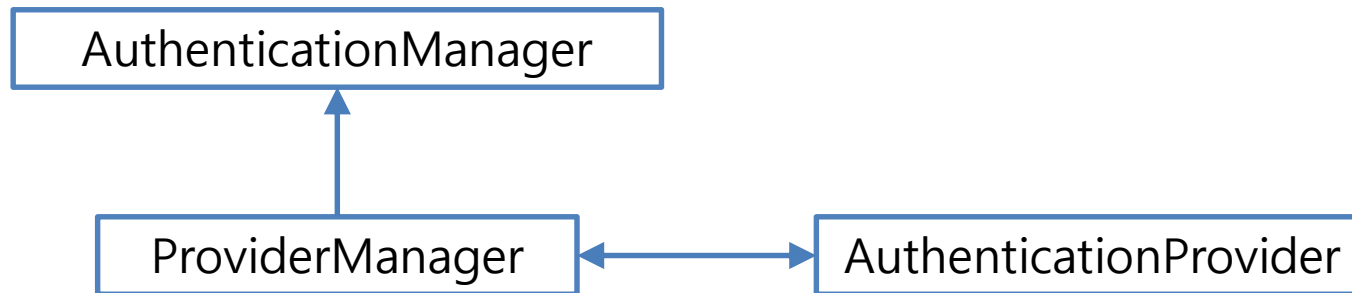
## ✓ AuthenticationManager(인증 매니저) 상속관계

- 인증 담당하며, 다양한 방식의 인증을 처리



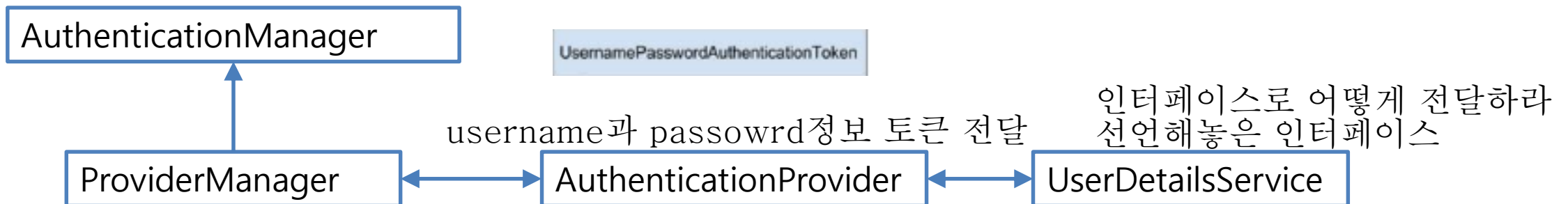
## ✓ ProviderManager

- 인증에 대한 처리를 AuthenticationProvider라는 타입의 객체를 이용해 처리



## ✓ AuthenticationProvider(인증 제공자)

- 실제 인증 작업을 진행
- UserDetailsService가 인증된 정보에 권한에 대한 정보를 구성



- ✓ 스프링 시큐리티를 커스터마이징 하는 방식 2가지 방식
  - AuthenticationProvider를 직접 구현하는 방식 1
  - 실제 처리를 담당하는 UserDetailsService를 구현하는 방식 2 이 권장되고 우리가 할 방법