Spring Kakao Login

Spring Security 2

Kakao Login

> Kakao Login API
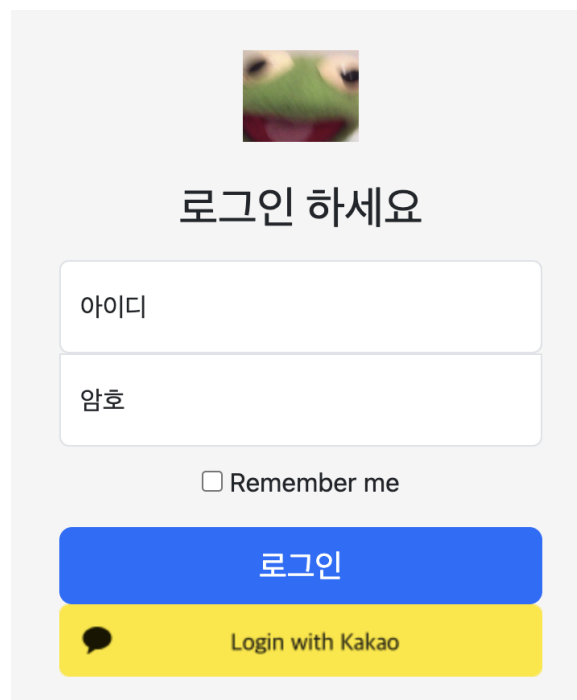
Kakao Developers

카카오 API를 활용하여 다양한 어플리케이션을 개발해보세요. 카카오 로그인, 메시지 보내기, 친구 API, 인공지능 API 등을 제공합니다.

k https://developers.kakao.com/

kakao developers

로그인 디자인 설정

```
<a href="javascript:kakaologin()">
  <img src="/resources/images/kakao_login_medium_wide.png">
</a>
```

로그인 하세요

아이디

암호

☐ Remember me

로그인

● Login with Kakao

로그인 시 넘겨 받을 항목 처리

옛날방식

```
<!-- Kakao SDK -->
<script src="https://developers.kakao.com/sdk/js/kakao.js"></script>

<script>
/* kakaologin 옛날 방식 */
Kakao.init('JAVASCRIPT_KEY');

function kakaologin() {
```

```
    window.Kakao.Auth.login({
      /* kakaologin 시 넘겨 받을 항목 */
      scope:'profile_nickname, account_email, gender',
      success: function (authObj) {window.Kakao.API.request({url:'/v2/user/me',
        success:res => {
          const nickname = res.kakao_account.profile.nickname;
          const email = res.kakao_account.email;
          const gender = res.kakao_account.gender;

          /* 콘솔로 확인 */
          /* console.log(nickname);
          console.log(email);
          console.log(gender); */

          kakaoProcess(nickname, email, gender);
        }
      })}
    })
}
```

Kakao Login 성공 여부 따라 사이트에서 로그인 처리

```
<!-- AJAX JQuery -->
<script src="https://code.jquery.com/jquery-3.6.3.min.js" integrity="sha256-pvPw+upLPUjgMXY0G+8O0xUf+/Im1MZjXxxgOcBQBXU=" crossorigin=

<script>
/* kakao 승인 절차 ajax */
function kakaoProcess(nickname, email, gender){
  $.ajax({
    type:'POST',
    url:'/users/kakao',
    data:{nickname:nickname,
      email:email,
      gender:gender
    },
    beforeSend:function(xhr){
      /* Spring Security로 인해 전송 전에 헤더에 csrf 값을 넣어야한다 */
      xhr.setRequestHeader('${_csrf.headerName}', '${_csrf.token}');
    },
    success:function(result){
      alert('카카오 로그인 성공, 메인 화면으로 이동합니다.');
      window.location.assign('/');
    },
    error:function(request, status, error){
      alert('카카오 로그인 실패, 최초 회원 가입 필요');
      window.location.assign('/users/joinkakao')
    }
  })
}
</script>
```

Mapper

user_SQL.xml

```
<select id="select_email"
  parameterType="String"
  resultType="com.carshop.users.User">
  <![CDATA[
    SELECT *
    FROM users
    WHERE username = #{email}
  ]]>
</select>
```

Repository

```
User existUsername(String email);
---------------------------------------------------------------------------
@Override
public User existUsername(String email) {
  return this.sqlSessionTemplate.selectOne("users.select_email", email);
}
```

Service

```
User existUsername(String email);
--------------------------------------------------------------------------------
@Override
public User existUsername(String email) {
  return userRepository.existUsername(email);
}
```

Controller

```
@RequestMapping("/kakao")
public String loginCheckKakao(@RequestParam Map<String, Object> auth) {

  String email = (String) auth.get("email");

  User user = this.userService.existUsername(email);

  if(user!=null) {
    System.out.println("디비에 회원정보 있음 즉 이미 회원");
    return "/login";//존재하는 페이지로 지정 > success
  } else {
    System.out.println("디비에 회원정보 없음 즉 회원 아님");
    return "";//error 발생시키기
  }
}

@GetMapping("/joinkakao")
public String joinkakaoForm(@ModelAttribute("NewUser") User user) {
  return "users/joinkakao";
}

@PostMapping("/joinkakao")
public String submitkakaoForm(@ModelAttribute("NewUser") User user) {
  //스프링은 반드시 password를 암호화하여 저장해야만 로그인 가능
  String encodedPassword = bCryptPasswordEncoder.encode(user.getPassword());
  user.setPassword(encodedPassword);
  userService.setNewUser(user);
  return "redirect:/login";
}
```

Spring Security Login Session

> Kakao Login 성공 시 session 처리

```
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.context.SecurityContext;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.web.context.HttpSessionSecurityContextRepository;
------------------------------------------------------------------------------------
if(user!=null) {
  //System.out.println("디비에 회원정보 있음 즉 이미 회원");
  List<GrantedAuthority> list = new ArrayList<GrantedAuthority>();
  list.add(new SimpleGrantedAuthority("ROLE_USER"));

  SecurityContext sc = SecurityContextHolder.getContext();

  sc.setAuthentication(new UsernamePasswordAuthenticationToken(user, null, list));

  HttpSession session = req.getSession(true);

  session.setAttribute(HttpSessionSecurityContextRepository.SPRING_SECURITY_CONTEXT_KEY, sc);

  return "/login";//존재하는 페이지로 지정 > success
}
```