isotai
@isotai
2018 年 05 月 13 日に投稿
ユーザー登録時のメール認証機能の実装方法
https://qiita.com/isotai/items/f810493dd192e0597f3a

Java
spring-boot

はじめに
web サービスを作成した際に実装したメール認証機能の流れを紹介します。
必要な操作は DB 登録と重複のチェックくらいで以外とシンプルです。
ただ、当初は実装方法のイメージを持つまで時間がかかってしまったので、
実装の流れを中心に紹介していきたいと思います。

全体の流れ：
1. (ユーザー)メールアドレスとパスワードを入力
2. (システム)UUID を生成してユーザー情報と一緒に一時テーブルに保存
3. (システム)ユーザーに認証 URL を送付
4. (ユーザー)メールアドレスから認証 URL をクッリク
5. (システム)UUID に該当する一時テーブルのユーザーを、正式にユーザー登録

1.メールアドレスとパスワードを入力
Ajax でも Post でも好きな方法でユーザー情報をサーバーサイドに渡してください。

スクリーンショット 2018-05-13 17.20.36.png

2.3.ユーザー情報を一時テーブルに保存 & ユーザーに認証 URL を送付
以下のような流れで処理していきます。

入力されたユーザー情報を受け取る
入力されたメールアドレスがすでに登録されていないか確認
確認が取れた場合、UUID と一緒に一時テーブルにユーザーを保存
認証 URL を生成してユーザーにメール送付します。認証 URL は/validate/id=UUID としています。
ユーザーが URL をクリックしたことを識別するには、
URL に一時テーブルに保存したユーザーと紐づく情報を追加しておく必要があります。
バッティングしたり予測できてしまうと、他人の一時ユーザーを認証できてしまうので、
UUID を使用します。

RegisterUserController

```
boolean isMember = memberRepository.existsByUsername(user);


        if(!isMember){
        String vali = UuidUtil.generateUUID();
        BCryptPasswordEncoder passEncoder = new BCryptPasswordEncoder();




        try {
            TmpMember tmpMember = new TmpMember(user, passEncoder.encode(pass), displyname, vali);
            tmpMemberRepository.saveAndFlush(tmpMember);
        } catch (Exception e) {
            e.printStackTrace();
            //status = "エラー：DB 保存失敗";
            return status;
        }

        String IPadnPort = myIP.getYourIP();
        String from = "送信元のメールアドレス";
        String title = "Tobidemo アカウント確認のお願い";
        String content = displyname + "さん" + "¥n" + "¥n" + "以下のリンクにアクセスしてアカウントを認
証してください" + "¥n"
                +"http://" + IPadnPort
                + "/validate"+ "?id=" + vali ;

        try {
            SimpleMailMessage msg = new SimpleMailMessage();

            msg.setFrom(from);
            msg.setTo(user);
            msg.setSubject(title);// タイトルの設定
            msg.setText(content); // 本文の設定
            mailSender.send(msg);
        } catch (Exception e) {
            e.printStackTrace();
            //status = "エラー：メール送付失敗";
            return status;
        }
```

```
            status = "ok";
            }
            return status; //ng

        }
        ;
        return status; //ng

    }
```

4.メールアドレスから認証 URL をクッリク
以下のようなメールがユーザーに送付されます。

スクリーンショット 2018-05-13 17.36.51.png

5.UUID に該当する一時テーブルのユーザーを正式にユーザー登録
ユーザーが URL をクリックしてアクセスした場合、id= の UUID を受け取ります。
受け取った UUID が一時テーブルに保存されているか確認します。
確認が取れた場合、認証済みのユーザー情報を保管するテーブルに、
登録し直します。
その後、サービスのログインページにリダイレクトしています。

ValidateUserController.java

```
@CrossOrigin
    @RequestMapping(value = "/validate", method = RequestMethod.GET)
    public String validate(RedirectAttributes redirectAttributes,ModelAndView mav, @RequestParam("id") String
id) throws Exception {

        String isRegisterd = "false";
        boolean isExist = tmpMemberRepository.existsByValidation(id);



        //System.out.println(isExist);

        if (isExist) {
            try {
                TmpMember tmp = tmpMemberRepository.findByValidation(id);
                String username = tmp.getUsername();
                String displyname = tmp.getDisplyname();
                String password = tmp.getPassword();



                Member member = new Member();
```

```
                member.setDisplyname(displyname);
                member.setPassword(password);
                member.setUsername(username);

                memberRepository.saveAndFlush(member);

                isRegisterd = "true";

            } catch (Exception e) {
                // TODO 自動生成された catch ブロック
                e.printStackTrace();
                 isRegisterd = "false";
            }



        }
        redirectAttributes.addFlashAttribute("isRegisterd", isRegisterd);
         return "redirect:/edit/begin";
    }
```

まとめ
最低限の機能ですが、意外と簡単にメール認証を実装することができました。
自前のメール認証ロジックなので、おかしなところがあるのかもしれないのですが、個人開発なので動くこと重視です！