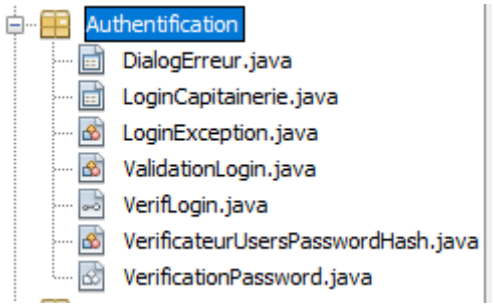
Explication et code correspondant à un login



Voici mon package contenant les fenêtres « Dialog » et les différentes classes/interfaces utiles au Login.

Lorsqu'un utilisateur démarre l'application, il se retrouve sur la Capitainerie (inutile car la plupart des boutons sont disable). Seulement le bouton « Se connecter » et « Format Date » sont actifs. L'utilisateur, lorsqu'il veut se connecter, doit appuyer sur le bouton « Se connecter ». Une fenêtre modale s'ouvre avec deux champs (Login et Password). L'utilisateur rentre ses données de login et clique sur le bouton de connexion.

```
if (this.BoutonConnexion.getText().equalsIgnoreCase("Se connecter"))
        {
            try
            {
                this.Sub = new VerificateurUsersPasswordHash (Login, Password);
                if (!Sub.isValid(Password, Login))
                    throw new LoginException();
                else
                {
                    if (Sub.findPwd(Login, Password))
                    {
                        System.out.println ("Mot de passe trouvé");
                        FenetreCapitainerie.MenuLogout.setEnabled(true);
                        FenetreCapitainerie.BoutonLire.setEnabled(true);
                        FenetreCapitainerie.MenuAmarrage.setEnabled(true);
                        FenetreCapitainerie.MenuBateau.setEnabled(true);
                        FenetreCapitainerie.MenuPersonnel.setEnabled(true);
                        FenetreCapitainerie.MenuFichierLOG.setEnabled(true);
                        FenetreCapitainerie.MenuLogin.setEnabled(false);
                        FenetreCapitainerie.BoutonDemarrerServeur.setEnabled(true);
                        FenetreCapitainerie.MenuNewUser.setEnabled(true);
                        this.setVisible(false);
                    }
                    else
                        throw new LoginException();
                }
            }
            catch (LoginException e)
            {
            }
        }
```

Je récupère son log et son pwd, je vérifie si le texte du bouton de connexion est bien en « Se connecter » (voir plus tard). Si oui, je crée un objet « VerificateurUsersPasswordHash » avec son Login en paramètre.

```
public VerificateurUsersPasswordHash (String log){
        super ();
        hash = new HashMap<>();
        repertoire = System.getProperty("user.dir"); // Je récupère les propiétés System
(Portabilité)
        separateur = System.getProperty("file.separator"); // Je récupère les propiétés
System (Portabilité)
        File dir = new File (repertoire + separateur + "Capitainerie" + separateur +
"Users");
        boolean dirExist = dir.exists(); // Je vérifie si un repertoire USER existe
        if (!dirExist)
        {
            try
            {
                hash.put(log, "1234"); // Si il n'existe pas, je crée le dossier +
fichier "Nomutilisateur.data" avec son login et un mdp 1234
                System.out.println ("Aucun User existe : création du sous dossier avec un
admin.");
                dir.mkdirs();
                fichier = new File (dir + separateur + "Users" + log + ".data");
                try (FileOutputStream fos = new FileOutputStream (fichier);
ObjectOutputStream oos = new ObjectOutputStream (fos))
                {
                    oos.writeObject(hash);
                    oos.flush();
                }
            }
            catch (FileNotFoundException ex){
                System.out.println ("Fichier "+fichier+" non-trouvé lors de l'écriture
dans le constructeur Dialogin");
            }
            catch (IOException ex) {
                System.out.println ("IOException Users"+fichier+" lors de l'écriture dans
le constructeur Dialogin");
            }
        }
```

Voici le constructeur de l'objet. Je vais tout d'abord vérifier si le répertoire « Users » existe (s'il y déjà un utilisateur d'encodé). Si le répertoire n'existe pas, je vais alors en créer un et y ajouter un premier utilisateur. Les données de l'utilisateur seront son login et son mdp, stockés sous la forme d'une hashmap (clé + valeur). La clé sera son nom d'utilisateur et la valeur de cette clé, son mot de passe.

```
        else // Si un fichier USER existe
        {
            try
            {
                System.out.println ("Au moins le User admin existe déjà, lecture du
fichier correspondant au User encodé.");
                fichier = new File (dir + separateur + "Users" + log + ".data"); // Je
crée un fichier correspondant a celui du log
                try (FileInputStream fis = new FileInputStream (fichier);
ObjectInputStream ois = new ObjectInputStream (fis))
                {
                    this.hash = (HashMap) ois.readObject(); // je récupere dans ce
fichier le hasmap
                }
            }
            catch (FileNotFoundException ex) {
                System.out.println ("Fichier Users"+log+".data non-trouvé lors de la
lecture");
            }
            catch (IOException ex){
                System.out.println ("IOException Users"+log+".data lors de la lecture");
            }
            catch (ClassNotFoundException ex){
                System.out.println ("ClassNotFoundException Users"+log+".data lors de la
lecture");
            }
        }
```

Si un répertoire « Users » existe déjà, je vais alors vérifier si un fichier portant le nom « users+.data » existe dans ce même répertoire. Si oui, je l'ouvre et stocke sa valeur dans une Hashmap (this.hash).

```
if (!Sub.isValid(Password, Login))
                    throw new LoginException();
                else
                {
                    if (Sub.findPwd(Login, Password))
                    {
                        System.out.println ("Mot de passe trouvé");
                        FenetreCapitainerie.MenuLogout.setEnabled(true);
                        FenetreCapitainerie.BoutonLire.setEnabled(true);
                        FenetreCapitainerie.MenuAmarrage.setEnabled(true);
                        FenetreCapitainerie.MenuBateau.setEnabled(true);
                        FenetreCapitainerie.MenuPersonnel.setEnabled(true);
                        FenetreCapitainerie.MenuFichierLOG.setEnabled(true);
                        FenetreCapitainerie.MenuLogin.setEnabled(false);
                        FenetreCapitainerie.BoutonDemarrerServeur.setEnabled(true);
                        FenetreCapitainerie.MenuNewUser.setEnabled(true);
                        this.setVisible(false);
                    }
                    else
                        throw new LoginException();
                }
```

Si aucune exception n'a été déclenchée, je vais alors vérifier si le mot de passe rentré par l'utilisateur est valide (pas de caractères bizarres, une longueur de login raisonnable, etc.).

```
    public boolean isValid (String Password, String log)
    {
        if (log.length() <= 1){
            System.out.println("Le login doit faire plus de 2 caracteres !");
            return false;
        }

        String pattern= "^[a-zA-Z]*$";
        if(!log.matches(pattern)){
            System.out.println("La login ne peut contenir que des lettres et vous \naviez
entrés : " + log);
            return false;
        }

        if (Password.length() < 3){
            System.out.println("Le mot de passe ne peut être plus petit que 8 caractères
\net vous en aviez entrés : " + Password.length());
            return false;
        }
        else
            return true;
    }
```

Si les données sont valides, je vais maintenant comparer les deux hashmaps pour vérifier la correspondance des mots de passe.

```
@Override
    public boolean findPwd(String log, String pwd) {
        return hash.containsKey(log) && hash.containsValue(pwd);    // Je cherche dans la
hashmap la valeur correspondant à la clé LOG
    }
```

Si une exception est déclenchée comme le fichier user qui n'existe pas, le login pas valide, le mot de passe pas valide -> Je lance une LoginException qui va créer une fenêtre Dialog.

```
class LoginException extends Exception
{
    public LoginException()
    {
        DialogErreur dia = new DialogErreur(null, true);
        dia.setVisible(true);
    }
}
```