AUTHENTIFICATION

L'authentification est la procédure qui détermine l'identité de celui qui visite l'application. Soit le visiteur n'est pas identifié, il est anonyme. Soit il s'est identifié auprès du formulaire de connexion, il est membre de l'application. Dans Symfony, c'est le firewall qui gère l'authentification. Le visiteur doit être un membre authentifié pour accéder à l'ensemble des pages, sinon le firewall le redirigera vers la page de connexion qui est la seule partie de l'application à ne pas être sécurisée.

La classe d'utilisateurs User

Dans l'application, les utilisateurs sont des instances de la classe `User`. Et pour que Symfony l'accepte comme classe utilisateur de la couche sécurité, elle implémente l'interface `UserInterface` avec cinq méthodes obligatoires.

```php
// src/AppBundle/Entity/User.php

use Symfony\Component\Security\Core\User\UserInterface;

class User implements UserInterface
{
    // l'identifiant de l'utilisateur au sein de la couche sécurité
    private $username;

    // le mot de passe
    private $password;
}
```

La configuration de l'authentification

Elle s'effectue dans le fichier security.yml qui se subdivise en plusieurs sections :

- **Encoders**

```yaml
# app/config/security.yml
security:
    # ...

    encoders:
        AppBundle\Entity\User: bcrypt
```

Un encodeur est un objet qui encode le mot de passe des utilisateurs de l'application. Ici c'est l'algorithme `bcrypt` qui est utilisé pour encoder les mots de passe fournis par l'entité `User`.

- **Providers**

```yaml
# app/config/security.yml
security:
    # ...

    providers:
        # le nom du fournisseur
        doctrine:
            # le type de fournisseur
            entity:
                # la classe de l'entité que le fournisseur utilise
                class: AppBundle:User
                # l'attribut de la classe qui sert d'identifiant
                property: username
```

Un provider est un fournisseur d'utilisateurs. Les firewalls s'adressent aux providers pour récupérer les utilisateurs et les identifier.

- **Firewalls**

```yaml
# app/config/security.yml
security:
    # ...

    firewalls:
        # le nom du pare-feu,
        # il s'agit juste d'un identifiant unique
        main:
            # toutes les URL commençant par « / »
            # (c'est-à-dire l'application entière)
            # sont protégées par le pare-feu
            pattern: ^/
            # la méthode d'authentification utilisée pour le pare-feu
            form_login:
                # la route du formulaire de connexion
                login_path: login
                # la route de validation du formulaire de connexion
                check_path: login_check
```

Lorsque le pare-feu initie le processus d'authentification, il redirige l'utilisateur vers `login`, soit la route du formulaire de connexion. Ce dernier renseigne alors son nom d'utilisateur et son mot de passe transmis ensuite vers `login_check`, soit la route de validation du formulaire de connexion où seront vérifiés les identifiants.

Le formulaire de connexion

```php
// src/AppBundle/Controller/SecurityController.php

namespace AppBundle\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;


class SecurityController extends Controller
{
    /**
     * @Route("/login", name="login")
     */
    public function loginAction(Request $request)
    {
        // Le service authentication_utils permet de récupérer le nom d'utilisateur
        $authenticationUtils = $this->get('security.authentication_utils');

        // et l'erreur dans le cas où le formulaire a déjà été soumis mais était invalide
        $error = $authenticationUtils->getLastAuthenticationError();
        $lastUsername = $authenticationUtils->getLastUsername();
```

Le formulaire de connexion (route `login`) est définie dans le code source, dans la méthode `loginAction()` du contrôleur `SecurityController`. Alors que pour la validation du formulaire de connexion (route login_check), le système de sécurité de Symfony gère lui-même l'authentification grâce au gestionnaire d'évènements. En cas de succès, le visiteur sera authentifié. En cas d'échec, l'application le renverra vers le formulaire de connexion pour qu'il réessaie.

La vue du formulaire

```twig
{# src/AppBundle/Resources/views/security/login.html.twig #}

{% block body %}
    {# Si le formulaire a une erreur, elle est affichée ici #}
    {% if error %}
        <div class="alert alert-danger" role="alert">{{ error.messageKey|trans(error.messageData,
    {% endif %}

    {# Le formulaire avec l'URL de la validation vers la route « login_check » #}
    <form action="{{ path('login_check') }}" method="post">
        <label for="username">Nom d'utilisateur :</label>
        <input type="text" id="username" name="_username" value="{{ last_username }}" />

        <label for="password">Mot de passe :</label>
        <input type="password" id="password" name="_password" />

        <button class="btn btn-success" type="submit">Se connecter</button>
    </form>
```