



Rapport de sécurité sous Symfony 4.4.3

Sommaire

- Introduction3
- Security.yaml.....3
- Definition 4
- SecurityController4
- Création d'un administrateur4
- Autorisation4
- Voters 5

Introduction

Symfony intègre un système d'authentification sécurisé afin de protéger l'accès à certaines pages ainsi qu'à certaines méthodes. En voici une présentation.

Security.yaml

Ce fichier de configuration permet de définir tous les paramètres nécessaires au bon fonctionnement du security-bundle de Symfony

```
1 security:
2   encoders:
3     App\Entity\User:
4       algorithm: bcrypt
5     # https://symfony.com/doc/current/security.html#where-do-users-come-from-user-providers
6   providers:
7     doctrine:
8       entity:
9         class: App\Entity\User
10        property: username
11   firewalls:
12     dev:
13       pattern: ^/(_(profiler|wdt)|css|images|js)/
14       security: false
15     main:
16       anonymous: ~
17       pattern: ^/
18     form_login:
19       login_path: /login
20       check_path: /login_check
21       always_use_default_target_path: true
22       default_target_path: /
23     logout:
24       path: /logout
25       target: /
26
27   # activate different ways to authenticate
28   # https://symfony.com/doc/current/security.html#firewalls-authentication
29
30   # https://symfony.com/doc/current/security/impersonating\_user.html
31   # switch_user: true
32
33   # Easy way to control access for large sections of your site
34   # Note: Only the *first* access control that matches will be used
35   access_control:
36     - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
37     - { path: ^/users, roles: ROLE_ADMIN }
38     - { path: ^/, roles: IS_AUTHENTICATED_FULLY }
39     # - { path: ^/admin, roles: ROLE_ADMIN }
40     # - { path: ^/profile, roles: ROLE_USER }
41
```

Exemple d'un fichier security.yaml

Definition

- encoders : C'est ici que nous précisons l'algorithme d'encodage des mots de passe.
- providers : détermine la classe et l'attribut utilisé pour l'authentification.
- firewalls : ce sont les routes prises en compte par le système d'authentification pour activer et définir son fonctionnement. On observe ici qu'il n'est pas nécessaire d'être connecté pour accéder au profiler (interface de debugging), en revanche pour ce qui concerne le site en production on y définit les routes de connexion et de déconnexion.
- access_control : définit les routes nécessitant ou non l'usage de l'authentification.

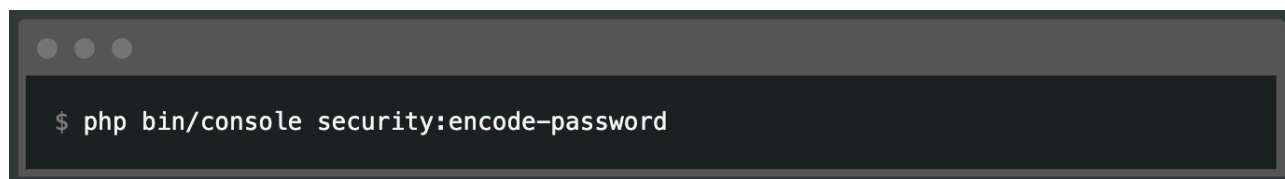
SecurityController

Afin de permettre l'authentification, ce contrôleur définira une méthode login pour répondre au formulaire de connexion, les autres routes concernant l'authentification seront définies dans le fichier config/routes.yaml.

Création d'un administrateur

La mise en place d'un administrateur peut se faire à l'aide de deux procédures distinctes, Soit en passant par l'interface utilisateur si elle est développée, dans le cas de todo&co, la route « user_create » via son url à condition d'être connecté et d'être soi-même un administrateur;

Soit en passant par la base de données, il sera cependant nécessaire d'inscrire un hash et non le mot de passe de l'utilisateur, ceci peut être effectué à l'aide des commandes fournies par Symfony

A terminal window with a dark background and light gray text. It shows a command prompt with a dollar sign followed by the command 'php bin/console security:encode-password'.

```
$ php bin/console security:encode-password
```

Autorisation

Une fois l'authentification effectuée, reste à savoir si l'utilisateur connecté a droit ou non d'accéder à la page demandée, c'est alors que nous utiliserons l'une des deux méthodes fournies avec le security-bundle de Symfony :

Soit `isGranted()` en lui précisant les droits minimum nécessaires,

Soit `denyAccessUnlessGranted` en lui précisant un paramètre de droit nécessaire et si souhaité, un message en cas d'erreur.

Voters

Si la vérification se fait via la méthode précédemment expliqué nécessite un paramètre, la logique correspondant a ce paramètre peut quant a elle être défini dans une classe Voter correspondant au contrôleur l'utilisant. Ce nouveau fichier se trouve dans :

➡ `src/Security/Voter/`.

Cette classe se compose assez facilement puisqu'elle étends de la classe Voter qui défini 2 méthodes, l'une vérifie que le paramètre et l'objet sont ceux attendues et l'autre retourne une méthode créé en son sein en fonction du paramètre récupéré.