Overriding Default FOSUserBundle Forms

-
  - Overriding a Form Type

Overriding a Form Type¶

The default forms packaged with the FOSUserBundle provide functionality for registering new user, updating your profile, changing your password and much more. These forms work well with the bundle's default classes and controllers. But, as you start to add more properties to your `User` class or you decide you want to add a few options to the registration form you will find that you need to override the forms in the bundle.

Suppose that you have created an ORM user class with the following class name, `AppBundle\Entity\User`. In this class, you have added a `name` property because you would like to save the user's name as well as their username and email address. Now, when a user registers for your site they should enter in their name as well as their username, email and password. Below is an example `$name` property and its validators.

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

use FOS\UserBundle\Model\User as BaseUser;
use Doctrine\ORM\Mapping as ORM;
use Symfony\Component\Validator\Constraints as Assert;

class User extends BaseUser
{
    /**
     * @ORM\Id
     * @ORM\Column(type="integer")
     * @ORM\GeneratedValue(strategy="AUTO")
     */
    protected $id;

    /**
     * @ORM\Column(type="string", length=255)
     *
     * @Assert\NotBlank(message="Please enter your name.",
groups={"Registration", "Profile"})
     * @Assert\Length(
     *     min=3,
     *     max=255,
     *     minMessage="The name is too short.",
     *     maxMessage="The name is too long.",
     *     groups={"Registration", "Profile"}
     * )
     */
    protected $name;

    // ...
}
```

Note

By default, the Registration validation group is used when validating a new user registration. Unless you have overridden this value in the configuration, make sure you add the validation group named Registration to your name property.

If you try and register using the default registration form you will find that your new `name` property is not part of the form. You need to create a custom form type and configure the bundle to use it.

The first step is to create a new form type in your own bundle. The following class inherits from the base FOSUserBundle `fos_user_registration` type using the form type hierarchy and then adds the custom `name` field.

```
1
2
3
4
```

```php
<?php
// src/AppBundle/Form/RegistrationType.php

namespace AppBundle\Form;

use Symfony\Component\Form\AbstractType;
use Symfony\Component\Form\FormBuilderInterface;

class RegistrationType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options)
    {
        $builder->add('name');
    }

    public function getParent()
    {
        return 'FOS\UserBundle\Form\Type\RegistrationFormType';

        // Or for Symfony < 2.8
        // return 'fos_user_registration';
    }

    public function getBlockPrefix()
    {
        return 'app_user_registration';
```

```
    }

    // For Symfony 2.x
    public function getName()
    {
        return $this->getBlockPrefix();
    }
}
```

Note

If you don't want to reuse the fields added in FOSUserBundle by default, you can omit the `getParent` method and configure all fields yourself.

Now that you have created your custom form type, you must declare it as a service and add a tag to it. The tag must have a `name` value of `form.type` and an `alias` value that is the equal to the string returned from the `getName` method of your form type class. The `alias` that you specify is what you will use in the FOSUserBundle configuration to let the bundle know that you want to use your custom form.

Below is an example of configuring your form type as a service:

- *YAML*

```
1
2
3
4
5
6
```

```
# app/config/services.yml
services:
    app.form.registration:
        class: AppBundle\Form\RegistrationType
        tags:
            - { name: form.type, alias: app_user_registration }
```

- *XML*

Finally, you must update the configuration of the FOSUserBundle so that it will use your form type instead of the default one. Below is the configuration for changing the registration form type in YAML.

```
1
```

```
2
3
4
5
6
7
8
# app/config/config.yml
fos_user:
    # ...
    registration:
        form:
            type: AppBundle\Form\RegistrationType
            # if you are using Symfony < 2.8 you should use the type name
instead
            # type: app_user_registration
```

Note how the `alias` value used in your form type's service configuration tag is used in the bundle configuration to tell the FOSUserBundle to use your custom form type.