### 如何写自定义验证类

创建一个自定义约束类

<?php

#Acme/DemoBundle/Validator/Constraints/ContaintsAlphanumberic.php

namespace Acme\DemoBundle\Validator\Constraints;

use Symfony\Component\Validator\Constraint;

class ContainsAlphanumeric extends Constraint

{

//只需要定义一个消息的属性；

public $message = 'he string "%string%" contains an illegal character: it can only contain

letters or numbers.';

}

//创建一个真正用来验证的类；默认名称对应是 类名+Validator(eg: ContainsAlphanumericValidator.php),如果不使用默认名称就需要在约束类中重写validatedBy方法

<?php

#Acme/DemoBundle/Validator/Constraints/ContainsAlphanumericValidator.php

namespace Acme\DemoBundle\Validator\Constraints;

use Symfony\Component\Validator\Constraint;

use Symfony\Component\Validator\ConstraintValidator;

class ContainsAlphanumericValidator extends ConstraintValidator

{

/\*\*

\* 可以注入container;

\*/

public function \_\_construct()

{

}

public function validate($value, Constraint $constraint)

{

//具体的验证方法

if (!preg\_match('/^[a-zA-Za0-9]+$/', $value, $matches)) {

$this->context->addViolation(

$constraint->message,

array('%string%' => $value))

);

}

Symfony2 2.5 的写法

$this->context->buildViolation($constraint->message)->setParameter('%string%', $value)->addViolation();

\*/

}

}

使用；直接指向这个约束；

*# src/Acme/BlogBundle/Resources/config/validation.yml*Acme\DemoBundle\Entity\AcmeEntity:\_x000D\_ properties:\_x000D\_ name:\_x000D\_ - NotBlank: ~\_x000D\_ - Acme\DemoBundle\Validator\Constraints\ContainsAlphanumeric: ~

把自定义的约束做为一个服务（可以增加参数；注入服务就可以操作数据库等服务）

services:\_x000D\_ validator.unique.your\_validator\_name:\_x000D\_ class: Acme\DemoBundle\Validator\Constraints\ContainsAlphanumericValidator tags:\_x000D\_ - { name: validator.constraint\_validator, alias: alias\_name }

此时就可以使用别名来验证(重写validateBy方法)

##Acme/DemoBundle/Validator/Constraints/ContainsAlphanumericValidator.php

public function validatedBy(){\_x000D\_ return 'alias\_name';

}