

Django, newforms..

Григорий Петухов, Руну, Омск, 21 июня 2008г.

Обработка форм.

- Рендеринг формы
- Валидация
- Рендеринг с ошибками
- Сохранение данных
- DRY

Рендеринг формы.

```
<form method="post">
  <table>
    {{ form }}
    <tr><td><input type="submit" /></td></tr>
  </table>
</form>
```

```
...
<tr>

<th><label for="id_password">Password:</label></th>
  <td><input id="id_password" type="password"
name="password" maxlength="Password" /></td>
</tr>
```

Варианты рендеринга.

- `{{ form }}` = `{{ form.as_table }}`
- `{{ form.as_p }}`
- `{{ form.as_ul }}`
- `{{ form.some_field }}`

View

```
@render_to('some_template.html')
def auth(request):
    if 'POST' == request.method:
        form = SomeForm(request.POST)
    else:
        form = SomeForm()
    if form.is_valid():
        form.save()
        return HttpResponseRedirect('...')
    return {'form': form}
```

Привязка к данным.

- Unbound: `SomeForm()`
- Bound:
 - `SomeForm({'a': 'b'})`
 - `SomeForm({})`
 - `SomeForm(request.POST)`

Валидация.

- Валидная форма
 - `form.is_valid()` is True
 - `form.cleaned_data`
 - `form.errors == {}`
- Невалидная форма
 - `form.is_valid()` is False
 - `form.cleaned_data` не существует
 - `f.errors: {'field_name': ['Some error']}`

Базовая валидация.

- `forms.CharField()`
- `forms.CharField(required=False)`
- `forms.ImageField()`
- `forms.CharField(max_length=33)`

Метод поля clean

```
class Field(object):
    def __init__(self, required=True, ...):
        self.required = required
        ....
    def clean(self, value):
        if self.required and value in EMPTY_VALUES:
            raise ValidationError(
                self.error_messages['required'])
        return value
```

```
class CharField(object):
    def clean(self, value):
        super(CharField, self).clean(value)
        ....
```

Метод формы clean_FOO

```
class FooForm(forms.Form):
    login = models.CharField(max_length=50)

    def clean_login(self):
        if 'login' in self.cleaned_data:
            login = self.cleaned_data['login']
            if 'z' in login:
                raise forms.ValidationError('Do
not use Z!')
            return login
```

Метод формы clean

```
class FooForm(forms.Form):
    login = models.CharField(max_length=50)

    def clean_login(self):
        if 'login' in self.cleaned_data:
            login = self.cleaned_data['login']
            if 'z' in login:
                raise forms.ValidationError('Do
not use Z!')
            return login
```

Порядок валидации.

1. Методы `clean` у каждого поля
2. Методы `clean_FOO`
3. Метод `clean` формы

Результат: `cleaned_data`

Снова View

```
@render_to('some_template.html')
def auth(request):
    if 'POST' == request.method:
        form = SomeForm(request.POST)
    else:
        form = SomeForm()
    if form.is_valid():
        form.save()
        return HttpResponseRedirect('...')
    return {'form': form}
```

Сохранение формы

```
def save(self):  
    login = self.cleaned_data['login']  
    password = self.cleaned_data['password']  
    user = User(login=login, password=password)  
    return user
```

View для редактирования.

```
@render_to('some_template.html')
def auth(request, user_id):
    user = User.objects.get(pk=user_id)
    if 'POST' == request.method:
        form = SomeForm(request.POST, user=user)
    else:
        form = SomeForm(initial={'login': user.login})
    if form.is_valid():
        form.save()
        return HttpResponseRedirect('...')
    return {'form': form}
```

Форма для редактирования.

```
class UserForm(forms.Form):  
    login = form.CharField()  
  
    def __init__(self, *args, *kwargs):  
        self.user = kwargs.get('user')  
        super(UserForm, self).__init__(*args, **kwargs)  
  
    def save(self):  
        login = self.cleaned_data['login']  
        user = form.save_instance(self, user)  
        return user
```

Автогенерация форм.

- Устарели:
 - `form_for_instance`
 - `form_for_model`
- Актуально:
 - `ModelForm`

ModelForm

```
class UserForm(ModelForm):  
    class Meta:  
        model = User
```

```
class UserForm(ModelForm):  
    password = CharField(required=False)  
  
    class Meta:  
        model = User  
        fields = ['username', 'password']
```

ModelForm - редактирование.

```
@render_to('some_template.html')
def auth(request, user_id):
    user = User.objects.get(pk=user_id)
    if 'POST' == request.method:
        form = SomeForm(request.POST, instance=user)
    else:
        form = SomeForm(instance=user)
    if form.is_valid():
        form.save()
        return HttpResponseRedirect('...')
    return {'form': form}
```

Конец.