Validating Forms

Built-in Validators

Form Level Validation

Custom Validator

Field Level Validation

Validating Forms - Form Level Validation

- In forms.py

```
from django import forms
from django.core.exceptions import ValidationError

class ContactForm(forms.Form):

    subject = forms.CharField(max_length=100, required=True)
    message = forms.CharField(widget=forms.Textarea)
    sender = forms.EmailField()
    cc_myself = forms.BooleanField(required=False)


    def myclean(self):
     cleaned_data = super(ContactForm, self).clean()

     email = cleaned_data.get('sender')
     dom = 'gmail.com'

     if dom not in email:
         print('here')
         raise ValidationError('incorrect domain')
```

Validating Forms - Form Level Validation

- In views.py

```
from django.shortcuts import render
from django.http import HttpResponse

from .forms import ContactForm

# Create your views here.
def page(request):

    if request.method == 'POST':

        form = ContactForm(request.POST)
        if form.is_valid():
            form.myclean()
            return HttpResponse('thanks')

    else:

        form = ContactForm()
        context = {'form': form}

        return render(request, 'home.html', context)
```

Validating Forms - Form Level Validation

- In home.html

```
<form method='POST'>
{% csrf_token %}
{{ form }}

<input type="submit" value="submit">
</form>
```

Validating Forms - Field Level Validation

- In forms.py

forms.py

```
from django import forms
from django.core.exceptions import ValidationError

class ContactForm(forms.Form):

    subject = forms.CharField(max_length=100, required=True)
    message = forms.CharField(widget=forms.Textarea)
    sender = forms.EmailField()
    cc_myself = forms.BooleanField(required=False)


    def field_validation(self):
        email = self.cleaned_data.get('sender')
        dom = 'gmail.com'
        if dom not in email:
            raise ValidationError('field validation failed')
```

Validating Forms - Field Level Validation

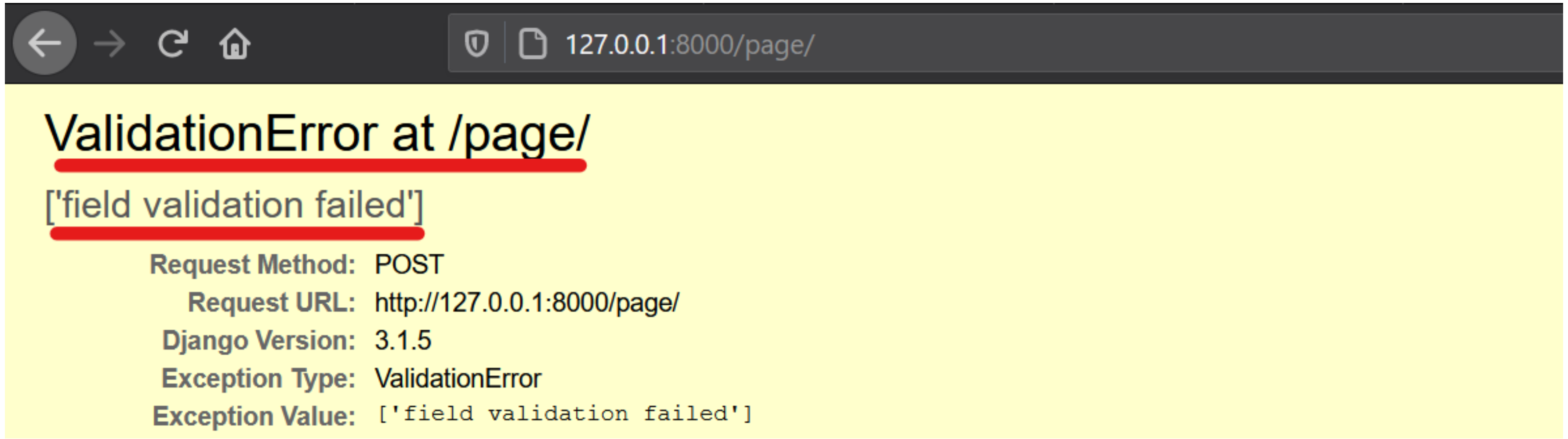
- In forms.py

views.py

```
from django.shortcuts import render
from django.http import HttpResponse

from .forms import ContactForm


def page(request):
    if request.method == 'POST':
        form = ContactForm(request.POST)
        if form.is_valid():
            form.field_validation()
            return HttpResponse('thanks')

    else:
        form = ContactForm()
        context = {'form': form}
        return render(request, 'home.html', context)
```

Validating Forms - Field Level Validation

- If sender has any mail domain other than gmail.com



127.0.0.1:8000/page/

ValidationError at /page/

['field validation failed']

```
 Request Method:  POST
    Request URL:  http://127.0.0.1:8000/page/
 Django Version:  3.1.5
 Exception Type:  ValidationError
Exception Value:  ['field validation failed']
```

Forms - Built-in fields

Built-in Fields

- BooleanField
- CharField
- ChoiceField
- TypeChoiceField
- DateField
- DateTimeField
- DecimalField
- DurationField
- EmailField
- FileField
- FilePathField
- FloatField
- ImageField
- IntegerField
- JSONField
- URLField

More:
https://docs.djangoproject.com/en/3.1/ref/forms/fields/#built-in-field-classes

Forms - Built-in fields

forms.py

```
from django import forms

class myForm(forms.Form):
    x1 = forms.BooleanField(required=False)
    x2 = forms.CharField(min_length = 2, max_length = 10,
required=False)

    x = (('s', 'small'),
        ('m', 'medium'))
    x3 = forms.ChoiceField(choices=x)

    x = (('1', '10'),
        ('1', '20'))
    x4 = forms.TypedChoiceField(choices=x,coerce=int)
```

Forms - Built-in fields

forms.py

Contd...

```
    x5 = forms.DateField(required=False)
    x6 = forms.DateTimeField(required=False)
    x7 = forms.DecimalField(required=False)
    x8 = forms.DurationField(required=False)
    x9 = forms.EmailField(required=False)
    x10 = forms.FileField(required=False)
    x11 =
forms.FilePathField(allow_folders=True,path=r'C:\Users\harulp663
\Desktop\practice\myproj', required=False)
    x12 = forms.FloatField(required=False)
    x14 = forms.IntegerField(required=False)
    x15 = forms.JSONField(required=False)
    x16 = forms.URLField(required=False)
```

Forms - Built-in fields

Home.html

```
<form method='POST' enctype="multipart/form-data">
{% csrf_token %}
{{ form }}

<input type="submit" value="submit">
</form>
```

Forms - Built-in fields

Views.py

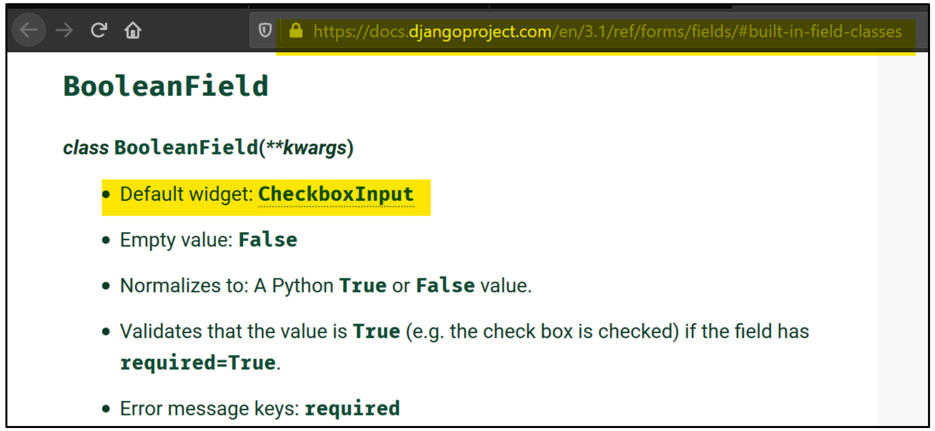
```
from django.shortcuts import render
from django.http import HttpResponse

from .forms import myForm


def page(request):
    if request.method == 'POST':
        form = myForm(request.POST,request.FILES)
        if form.is_valid():
            values = form.clean()
            print(values)
            return HttpResponse('check ur command prompt')
        else:
            return HttpResponse('not valid form')

    else:
        form = myForm()
        context = {'form': form.as_ul}
        return render(request, 'home.html', context)
```

Forms - Built-in widgets

Built-in widgets

Highlighted some widgets in the diagram

- Whenever a built-in field is used, there is a default widget used

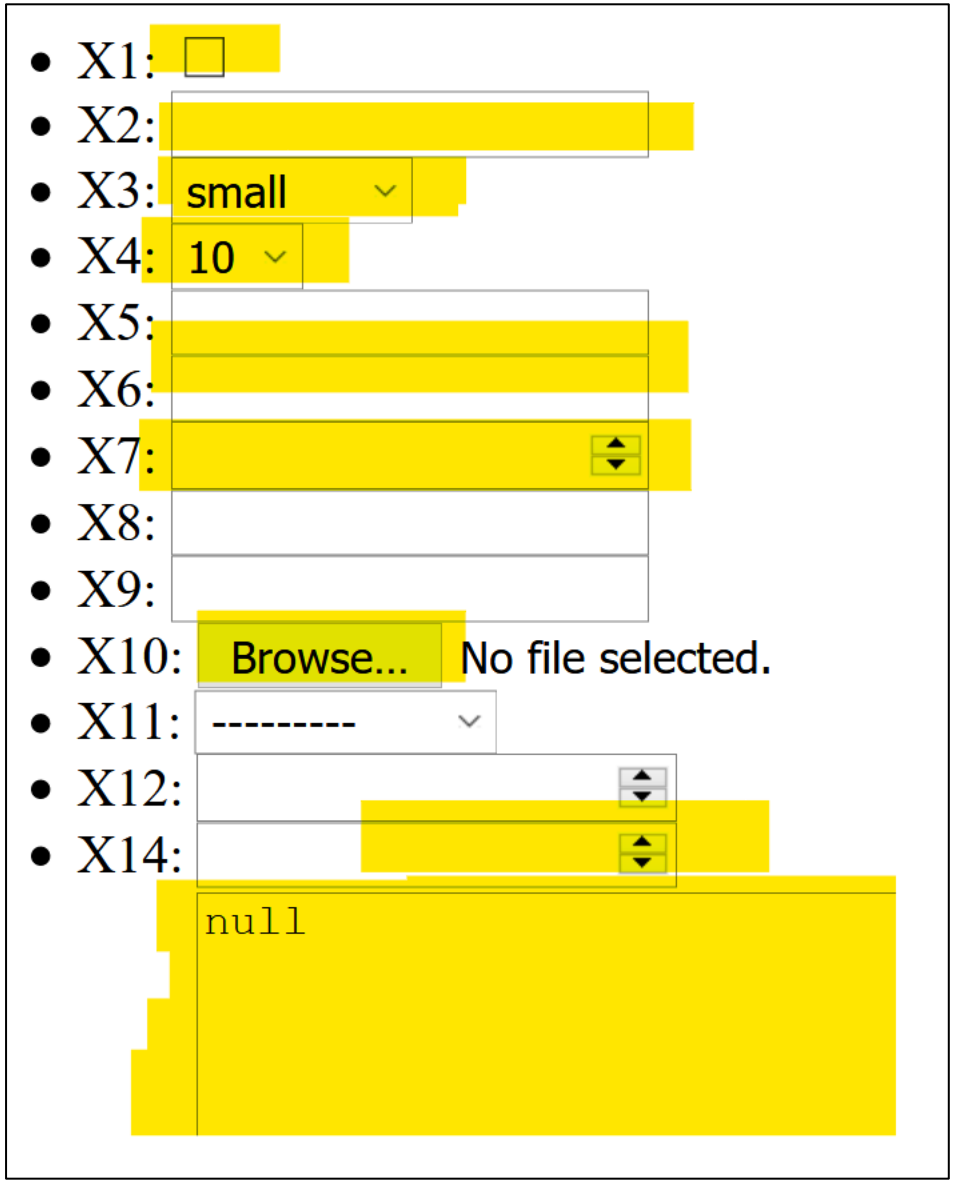


https://docs.djangoproject.com/en/3.1/ref/forms/fields/#built-in-field-classes

BooleanField

class `BooleanField(**kwargs)`

- Default widget: **CheckboxInput**
- Empty value: **False**
- Normalizes to: A Python **True** or **False** value.
- Validates that the value is **True** (e.g. the check box is checked) if the field has **required=True**.
- Error message keys: **required**



- X1: □
- X2:
- X3: small
- X4: 10
- X5:
- X6:
- X7:
- X8:
- X9:
- X10: Browse... No file selected.
- X11: ---------
- X12:
- X14:

null

Forms - Built-in widgets

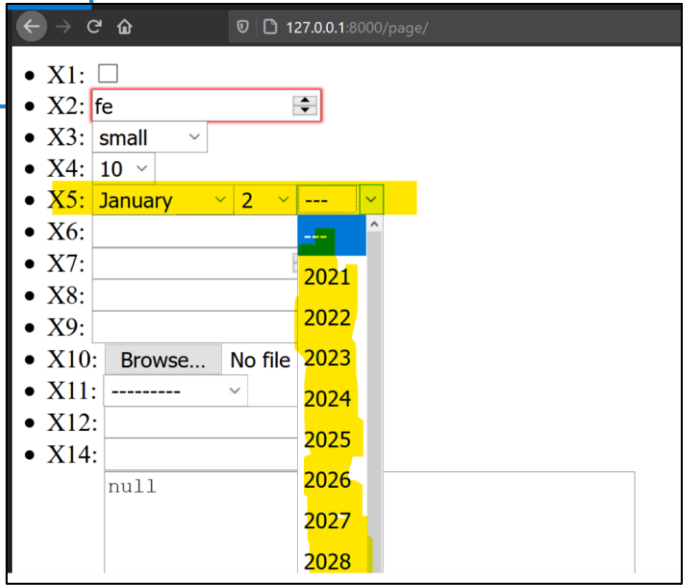
- **Widgets handling input of text**
  - TextInput
  - NumberInput
  - EmailInput
  - URLInput
  - PasswordInput
  - HiddenInput
  - DateInput
  - DateTimeInput
  - TimeInput
  - TextArea
- **Selector and checkbox widgets**
  - CheckBoxInput
  - Select
  - NullBooleanSelect
  - SelectMultiple
  - RadioSelect
  - CheckboxSelectMultiple
- **File upload widgets**
  - File Input
  - ClearableFileInput

More:
https://docs.djangoproject.com/en/3.1/ref/forms/widgets/#built-in-widgets

Forms - Built-in widgets

forms.py

```
from django import forms

class myForm(forms.Form):
    x5 = forms.DateField( required=False,
                          widget=forms.SelectDateWidget)
```



127.0.0.1:8000/page/

- X1: □
- X2: fe
- X3: small
- X4: 10
- X5: January 2 ---
- X6:
- X7:
- X8:
- X9:
- X10: Browse... No file
- X11: ---------
- X12:
- X14:

null