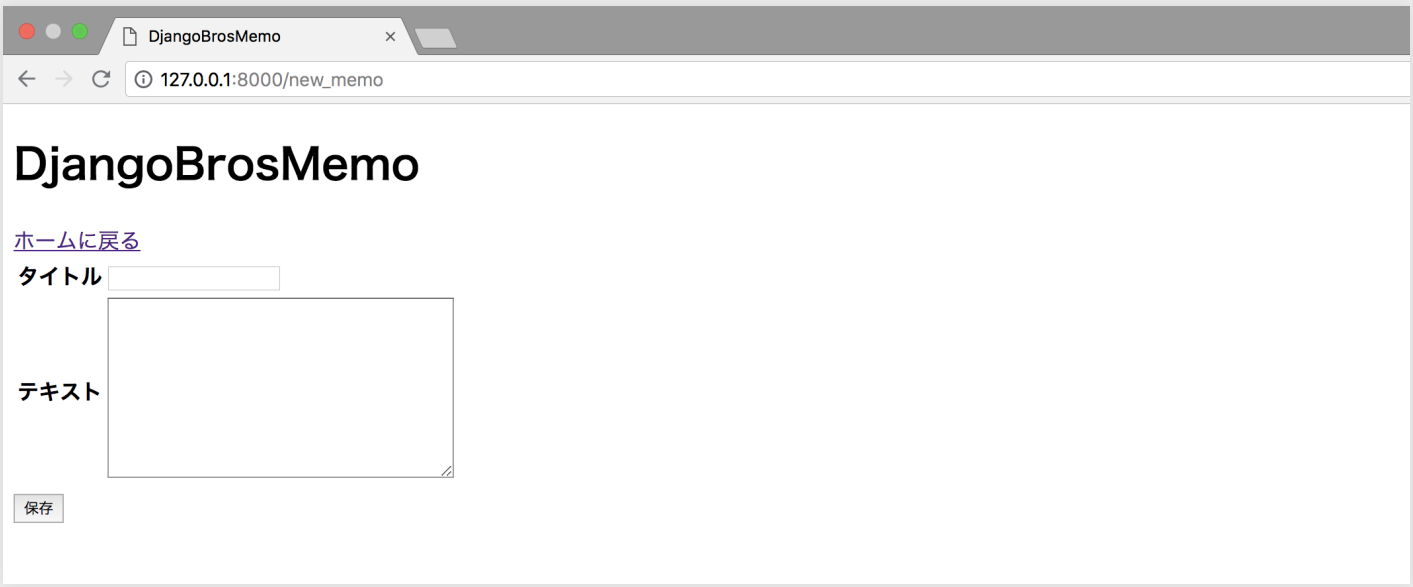
# フォームを作ろう

このレッスンでは、フォームを作成してユーザーが新しいメモを投稿できるようにします。「保存」ボタンを押すとmethod=POSTのリクエストが送られるように設定するんでしたよね。




必要な手順は以下の通りです。

1. フォームを作成する
2. フォームをnew_memo.htmlで表示する
3. フォームから受け取った情報を保存する

# 1. フォームを作成する

まずは、メモを投稿できるフォームを作りましょう。今回はフォーム用のファイルを作成します。ファイルはどこに作成してよいですが、今回はappディレクトリ直下にforms.pyというファイルを作成します。

~/memo/app

```
$ touch forms.py
```

forms.pyを開いて以下のように編集してください。

~/memo/app/forms.py

```
from django.forms import ModelForm
from .models import Memo

class MemoForm(ModelForm):
    class Meta:
        model = Memo
        fields = ['title', 'text']
```

１行目で、Djangoがデフォルトで用意している**ModelForm**というものをインポートし、これを継承させてMemoFormという名前のフォームを作っています。ModelFormは各々のモデルに対応したフォームを作ってくれます。今回の場合は、`model=Memo`とすることでMemoモデルに対応したフォームを生成しています。

Memoモデルは`title`や`text`というフィールドを持っていますので、これらに対応する入力欄があるフォームができるのです。`fields`の部分で表示する入力欄を定義しています。`fields = ['title']`とすると、タイトルだけ入力できるフォームとなります。

もちろん、モデルをベースとしないプレーンなフォームを作成することもできます。お問い合わせフォームなどの、モデルとは関係のないフォームが必要なときはプレーンなフォームを使用します。今回のようにモデルと関連づいたフォームを作成したいときは、ModelFormを継承するのが便利です。ModelForm以外のフォームの作り方は別の機会に紹介します。

## 2. フォームをnew_memo.htmlで表示する

続いて、今作成したMemoFormをテンプレートで表示させましょう。まずは、views.pyを以下のように編集します。

~/memo/app/views.py

```
from .forms import MemoForm

def new_memo(request):
    form = MemoForm
    return render(request, 'app/new_memo.html', {'form': form })
```

forms.pyファイルからMemoFormをインポートして、変数formに代入しています。そして、render関数の第３引数でそれをテンプレートに渡すように設定しています。これで、new_memo.htmlでは、{{ form }}という記述でフォームが表示されるようになります。

new_memo.htmlはこのようになります。

~/memo/app/templates/app/new_memo.html

```
<div>
  <a href="{% url 'app:index' %}">ホームに戻る</a>
</div>

<form action="{% url 'app:new_memo' %}" method="POST">{% csrf_token %}
  {{ form.as_p }}
  <button type="submit" class="btn">保存</button>
</form>
```

HTMLのformタグで{{ form }}を囲うことで、MemoFormをフォームとして表示します。`.as_p`をつけることで入力欄ごとにpタグで囲われることになるので、改行されて綺麗に表示してくれるようになります。

actionの部分では、このフォームが投稿されたときに実行される処理を定義しています。「投稿されたとき」というのは、type="submit"のボタンが押されたときのことです。上の例だと、保存ボタンが押された時に"{% url 'app:new_memo' %}"に対してHTTPリクエストが送られることになります。この時、method="POST"とあるようにHTTPメソッドはPOSTなので、フォームで入力したデータも一緒にサーバーに送られます。'app:new_memo'とは、urls.pyでname='new_memo'となっているパスのことです。

`{% csrf_token %}`はセキュリティ対策のために必要なものです。これがないとエラーになるので気をつけましょう。

Titleが空欄の状態で保存ボタンを押すと「このフィールドを入力してください。」といったエラーメッセージが表示されるはずです。このMemoFormはMemoモデルに対応したフォームであり、MemoモデルのTitleフィールドは`blank=False`と設定されているためです。一方Textはブランクを許容していますので、空欄のまま投稿できます。これで、「ModelFormがモデルに対応したフォームを作ってくれる」ということがなんとなく理解できたのではないでしょうか。

まだ、サーバー側の処理は定義していませんので、保存ボタンを押してもページがリロードされるだけだと思います。次は、保存処理ができるように設定しましょう！

## 3. フォームから受け取った情報を保存する

ここまでで、フォームから入力されたデータをサーバーに送ることができました。あとは、サーバー側で受け取った情報を保存する処理ができれば完了です。

保存ボタンが押された時は、"{% url 'app:new_memo' %}"、つまりnew_memo関数が実行されます。new_memo関数は以下のように編集してください。

~/memo/app/views.py

```
from django.shortcuts import render, redirect

def new_memo(request):
    if request.method == "POST":
        form = MemoForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('app:index')
    else:
        form = MemoForm
    return render(request, 'app/new_memo.html', {'form': form })
```

まず、if文でHTTPリクエストのメソッドがPOSTの時と、それ以外(else)のときで実行する処理を切り分けています。それ以外の時というのは、つまりmethod=GETのときです。

method=GETのときは、特に変更を加えていないのでこれまで定義していた処理をそのまま実行します。つまり、http://127.0.0.1:8000/new_memoというURLに直接アクセスされたときやリンクで飛んできたときは、ただ単にフォームを表示させる処理だけを実行します。

続いて、request.method="POST"のときの処理について説明します。保存ボタンが押された時の処理ですね。

まず、`form = MemoForm(request.POST)`では、新しいMemoインスタンスを生成して変数formに代入してます。引数にrequest.POSTを指定していますが、request.POSTにはユーザーがフォームに入力した情報が含まれていますので、その情報をもとにMemoインスタンスを生成します。

例えばユーザーが、タイトル「買うもの」、テキスト「卵、ジュース」という内容で保存した場合、title="買うもの", text="卵、ジュース"のMemoインスタンスが生成されます。request.POSTを引数にとるだけでこのような処理ができてしまうのは、MemoFormがModelFormを継承して作られた特別なフォームだからです。

ただこの段階では、インスタンスは一時的に生成されただけでまだDBに保存されていません。form.save()のように、save()メソッドが呼び出されたタイミングで初めて保存されます。

`if form.is_valid():`では、生成されたMemoインスタンスが正しい値を持っているかを検証しています。Memoモデルは、タイトルが150文字より多かったりタイトルがblankではいけないなどの条件を指定していますが、インスタンスがこの条件を満たしているかを判定しているのです。

最後のredirect関数では、次に実行する処理を指定しています。render関数と似ていますが、render関数では表示するテンプレートファイルを指定するのに対して、redirect関数ではURLパスを指定します。書き方も、render関数では`'app/new_memo.html'`のようにスラッシュを使ってフォルダ階層を示すのに対して、redirect関数では、`'app:index'`のようにパスを指定する必要があり微妙に書き方が異なります。

今回の例では、メモを保存したらリダクレクトさせてindex関数を実行し、トップ画面を表示するように設計しています。トップ画にいくと、フォームから投稿したメモが追加されているのがわかるはずです。

redirect関数もrender関数同様インポートしないと使えませんので注意してくださいね。

これで、ユーザーもメモを登録できるようになりました！これまでは、管理権限のあるひとしかデータを操作できませんでしたが、フォームを使うことによってサイトを訪れたユーザーが動的な処理をできるようになりましたね。

このあとのレッスンでメモの削除と編集機能も実装しますが、このレッスンの内容が理解できていればとても簡単にできちゃいますよ！