**Тема 2.4.5. Представления и маршрутизация – ключевые элементы Django**

В Django за обработку запросов пользователя отвечают представления (views, размещаются в файле views.py). В них реализованы функции, которые принимают данные запроса в виде объекта request и генерируют ответ пользователю (response) в виде HTML страниц.

Развитие файла views.

```
from django.http import HttpResponse


def index(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>Главная</h2>")


def about(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>О сайте</h2>")


def contact(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>Контакты</h2>")
```

Сопоставим функциям в файле views маршруты в файле urls.py.

Переменная urlpatterns (шаблон URL) содержит эти сопоставления:

```
from django.contrib import admin
from django.urls import path
from firstapp import views

urlpatterns = [
    path('', views.index),
    path('about', views.about),
    path('contact', views.contact),
    path('admin/', admin.site.urls),
]
```

Запустим сервер: python manage.py runserver

В браузере перейдем по адресу http://127.0.0.1:8000

Попадаем на главную страницу сайта.

По адресу http://127.0.0.1:8000/about попадем на страницу **О сайте.**

По адресу http://127.0.0.1:8000/contact попадем на страницу **Контакты.**

**Примечание**

При использовании функции path() запрошенный путь должен абсолютно точно соответствовать указанному в маршруте адресу URL. Иначе есть вероятность возникновения ошибки.

В пакете django.urls имеется альтернативная функция re_path(). Она позволяет задавать адреса URL посредством регулярных выражений.

```
urlpatterns = [
        path( ' ', views.index),
        re_path(r '^about',  views.about),
        re_path( '^contact',  views.contact),
        ]
```

```
urlpatterns = [
    path('', views.index),
    re_path(r'^about', views.about),
    re_path(r'^contact', views.contact),
]
```

Выражение ^about указывает на то, что адрес может не точно соответствовать, а только начинаться с about.

Основные элементы регулярных выражений:

| Оператор | Описание |
|---|---|
| `.` | Один любой символ, кроме новой строки `\n`. |
| `?` | 0 или 1 вхождение шаблона слева |
| `+` | 1 и более вхождений шаблона слева |
| `*` | 0 и более вхождений шаблона слева |
| `\w` | Любая цифра или буква (`\W` — все, кроме буквы или цифры) |
| `\d` | Любая цифра [0-9] (`\D` — все, кроме цифры) |
| `\s` | Любой пробельный символ (`\S` — любой непробельный символ) |
| `\b` | Граница слова |
| `[..]` | Один из символов в скобках (`[^..]` — любой символ, кроме тех, что в скобках) |
| `\` | Экранирование специальных символов (`\.` означает точку или `\+` — знак «плюс») |
| `^` и `$` | Начало и конец строки соответственно |
| `{n,m}` | От `n` до `m` вхождений (`{,m}` — от 0 до `m`) |
| `a\|b` | Соответствует `a` или `b` |
| `()` | Группирует выражение и возвращает найденный текст |
| `\t`, `\n`, `\r` | Символ табуляции, новой строки и возврата каретки соответственно |

Подробнее по ссылкам :

https://docs.python.org/3.8/library/re.html документация по регулярным выражениям в Python 3.

https://tproger.ru/translations/regular-expression-python/ практическое применение регулярных выражений.

**Передача параметров в функции представления**

**Определение параметров через функцию re_path()**

Дополним файл views.py функциями:

# функция принимает дополнительный параметр productid

def products(request, productid):

output = ”<h2>Продукт № {0} </h2>”.format(productid)

return HttpResponse (output)

# функция принимает дополнительные параметры id,name

def users(request, id, name):

output = ”<h2>Пользователь </h2><h3>id: {0}

Имя: {1}</h3>”.format(id,name)

return HttpResponse (output)

```
def products(request, productid):
    """
    :param request:
    :param productid:
    :return:
    """
    output = "<h2>Продукт № {0}</h2>".format(productid)
    return HttpResponse(output)


def users(request, id, name):
    """
    :param request:
    :param id:
    :param name:
    :return:
    """
    output = "<h2>Пользователь </h2><h3>id: {0} Имя: {1}</h3>".format(id,
name)
    return HttpResponse(output)
```

Соответственно изменим файл urls.py:

from django.urls import re_path

from firstapp import views

urlpatterns = [

re_path(r’^products/(?P<productid>\d+)/’, views.products),

re_path(r’^users/(?P<id>\d+)/(?P<name>\D+)/’, views.users),

]

Здесь ?P<>  - представление параметра в шаблоне адреса

Общий формат определения параметра:

(?P<имя_параметра>регулярное выражение)

```
urlpatterns = [
    path('', views.index),
    re_path(r'^about', views.about),
    re_path(r'^contact', views.contact),
    re_path(r'^products/(?P<productid>\d+)/', views.products),
    re_path(r'^users/(?P<id>\d+)/(?P<name>\D+)/', views.users),
]
```

Запустим локальный сервер разработки:

python manage.py runserver

далее:

http://127.0.0.1:8000/products/5 (запрос данных о продукте №5, будет возвращена страница с информацией о продукте №5)

http://127.0.0.1:8000/users/7/Иван/ (запрос о пользователе с идентификационным номером 7 по имени Иван, будет возвращена страница с информацией о пользователе с идентификационным номером 7 по имени Иван)

НО если адрес будет введен неточно и/или данные по запросу не будут найдены, то пользователю будет выдано сообщение об ошибке.

Чтобы это предотвратить, можно использовать параметры по умолчанию.

Тогда вместо сообщения об ошибке будут возвращены данные по умолчанию.

**Определение параметров по умолчанию**

def products(request, **productid=1**):

```
        output = ”<h2>Продукт № {0} </h2>”.format(productid)

        return HttpResponse (output)
```

При этом в файле urls.py надо будет прописать еще маршрут:

re_path(r’^products/$’, views.products).

Итоговый вид файла urls.py:

```
from django.contrib import admin

from django.urls import path

from django.urls import re_path

from firstapp import views

urlspatterns = [

        path(‘’,views.index),

        re_path(r’^about’, views.about),

        re_path(r’^products/$’, views.products),

        re_path(r’^products/(?P<productid>\d+)/’, views.products),

        re_path(r’^users/(?P<name>\D+)/’, views.users),

]
```

```
urlpatterns = [
    re_path(r'^products/$', views.products),
    path('', views.index),
    path('about', views.about),
    path('contact', views.contact),
    path('admin/', admin.site.urls),
    re_path(r'^products/(?P<productid>\d+)/', views.products),
    re_path(r'^users/(?P<id>\d+)/(?P<name>\D+)/', views.users),

]
```

**Определение параметров через функцию path()**

Формат параметров функции **path():**

<спецификатор:название_параметра>

Django предоставляет спецификаторы:

- str  - любой текст, кроме символа (/)
- int – любое целое положительное число
- slug  -последовательность буквенных символов ASCII, цифр, дефиса и символа подчеркивания.
- uuid – идентификатор UUID
- path – любая строка, включая символ подчеркивания

Для файла views.py:

```
from django.shortcuts import render

from django.http import HttpResponse

def index(request):

    return    HttpResponse(«<h2>Главная</h2>»)

def about(request):

    return    HttpResponse(«<h2>О сайте</h2>»)

def contact(request):

    return    HttpResponse(«<h2>Контакты</h2>»)

def products(request, productid=1):

        output = ”<h2>Продукт № {0} </h2>”.format(productid)

        return HttpResponse (output)

def users(request,id,name):

        output = ”<h2>Пользователь </h2><h3>id: {0}

         Имя: {1}</h3>”.format(id,name)

         return HttpResponse (output)
```

Содержание файла urls.py с использованием спецификаторов параметров функции path() будет следующим:

```
from django.contrib import admin

from django.urls import path

from django.urls import re_path

from firstapp import views

urlspatterns = [

        path(‘ ’, views.index),

        re_path(r’^about’, views.about),

        re_path(r’^contact’, views.contact),

        path(’products/<int:productid>/’, views.products),

        path(’users/<int:id>/<str:name>/’, views.users),

]
```

**Определение параметров по умолчанию в функции path()**

Определим в функциях в файле views.py для параметров значения по умолчанию:

```
def products(request, productid=1):

    output = "<h2>Product № {0}</h2>".format(productid)

    return HttpResponse(output)


def users(request, id=1, name="Dirk"):

    output = "<h2>User</h2><h3>id: {0}  name: {1}</h3>".format(id, name)

    return HttpResponse(output)
```

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


def index(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>Главная</h2>")


def about(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>О сайте</h2>")


def contact(request):
    """
    :param request:
    :return:
    """
    return HttpResponse("<h2>Контакты</h2>")


def products(request, productid=1):
    """
    :param request:
    :param productid:
    :return:
    """
    output = "<h2>Продукт № {0}</h2>".format(productid)
    return HttpResponse(output)


def users(request, id=1, name="Dirk"):
    """
    :param request:
    :param id:
    :param name:
    :return:
    """
    output = "<h2>Пользователь </h2><h3>id: {0} Имя: {1}</h3>".format(id,
name)
    return HttpResponse(output)
```

В этом случае для каждой функции в файле urls.py надо определить по два маршрута:

```
from django.urls import path

from firstapp import views
```

```
urlpatterns = [
    path('products/', views.products),
    path('products/<int:productid>/', views.products),

    path('users/', views.users),
    path('users/<int:id>/<str:name>/', views.users),
]
```

Файл urls.py  после изменений:

```
from django.contrib import admin
from django.urls import path
from django.urls import re_path
from firstapp import views
urlspatterns = [
        path(‘ ’, views.index),
        re_path(r’^about’, views.about),
        re_path(r’^contact’, views.contact),
        path('products/', views.products),    # маршрут по умолчанию
         path('products/<int:productid>/', views.products),

          path('users/', views.users),      # маршрут по умолчанию
          path('users/<int:id>/<str:name>/', views.users),
]
```

```python
from django.contrib import admin
from django.urls import path, re_path
from firstapp import views


urlpatterns = [
    path('', views.index),
    re_path(r'^about', views.about),
    re_path(r'^contact', views.contact),
    path('products/', views.products),  # маршрут по умолчанию
    path('products/<int:productid>/', views.products),
    path('users/', views.users),  # маршрут по умолчанию
    path('users/<int:id>/<str:name>/', views.users),
]
```

Запустим локальный сервер разработки:

python manage.py runserver

и перейдем на страницы

http://127.0.0.1:8000/products/ и http://127.0.0.1:8000/users/

Убедимся, что при отсутствии параметров в строке запроса пользователю не выдается ошибка, а возвращаются ответные страницы, определенные по умолчанию.

**Параметры строки запроса пользователя**

Есть разные способы передачи параметров.

Параметры могут передаваться **через интернет-адрес (url):**

**http://localhost/index/6/Dirk/**

**а** могут передаваться **через строку запроса пользователя:**

**http://localhost/index/?id=5&name=Dirk/**

Параметры строки запроса пользователя указываются после символа

знак вопроса **-?.**

Каждый такой параметр – это пара «ключ-значение».

Например: в параметре id=7 id – это ключ, а 7 – это значение.

Параметры отделяются один от другого знаком амперсанд - &.

Чтобы получить параметры из строки запроса нужно использовать метод request.Get.get().

Переопределим в файле views.py функции products()  и users(), используя метод request.Get.get():

```
def products(request, productid):
    category = request.GET.get("cat", "")
    output = "<h2>Product № {0}  Category: {1}</h2>".format(productid,
category)
    return HttpResponse(output)


def users(request):
    id = request.GET.get("id", 1)
    name = request.GET.get("name", "Mike")
    output = "<h2>User</h2><h3>id: {0}  name: {1}</h3>".format(id, name)
    return HttpResponse(output)
```

Параметр productid будет передаваться в функцию products через интернет-адрес. Из строки запроса будет извлекаться значение параметра cat:

**category = request.Get.get("cat", "").**

**"cat"** - название параметра строки запроса,

**""** - значение по умолчанию на случай ошибки

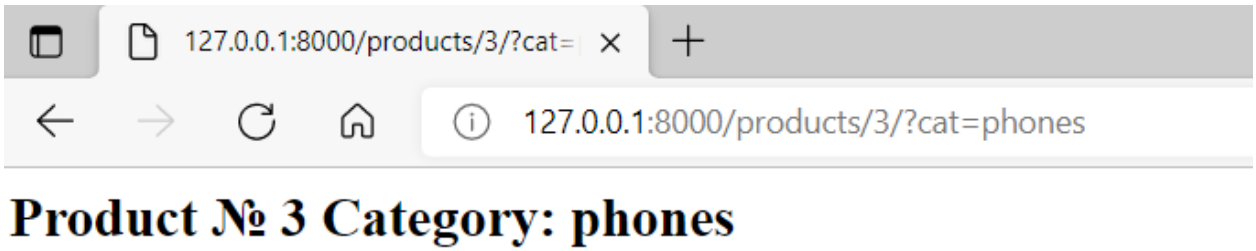
Значения параметров id и name в функции users извлекаются из строки запроса. Значения по умолчанию заданы: id=1, name=”Mike”.

В файле urls.py зададим маршруты:

```
urlpatterns = [
    path('', views.index, name='home'),
    path('products/<int:productid>/', views.products),
    path('users/', views.users),
]
```

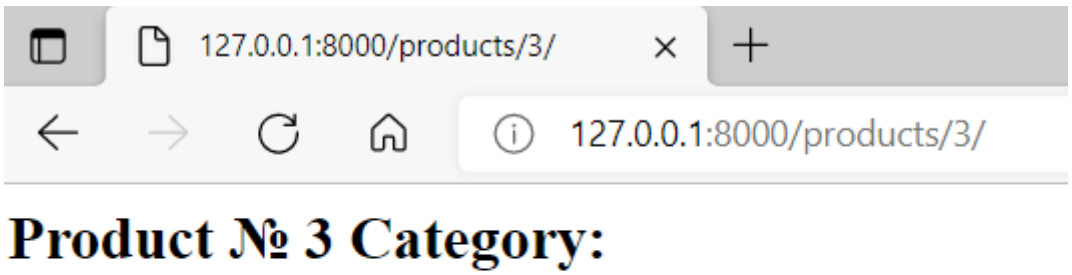
запустим отладочный сервер и обратимся к страницам products и users:

127.0.0.1:8000/products/3/?cat=phones

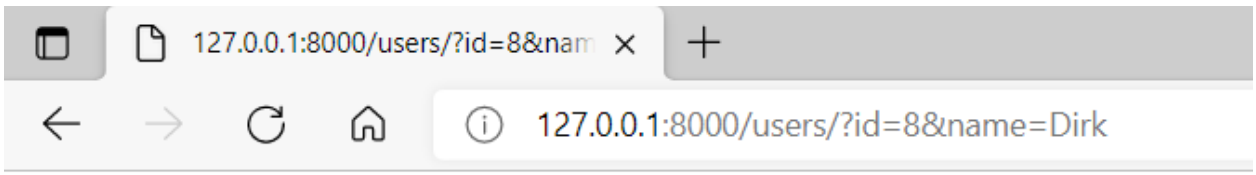



---

Вывод значений по умолчанию:



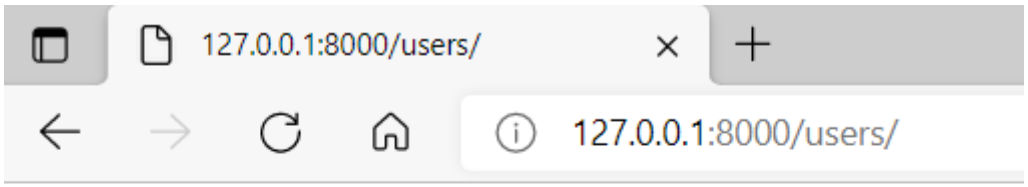

http://127.0.0.1:8000/users/?id=8&name=Dirk




**User**

**id: 8 name: Dirk**

127.0.0.1:8000/users/ - будут выведены значения по умолчанию:




**User**

**id: 1 name: Mike**