Установка DjDT

Сторонние приложения устанавливаются в виртуальное окружение через менеджер пакетов **pip**. Активируйте виртуальное окружение и выполните команду

```
pip install django-debug-toolbar==3.8.1
```

> После установки любого нового пакета выполняйте `pip freeze > requirements.txt`, тогда любой программист сможет легко установить зависимости, нужные в проекте.

После установки приложения зарегистрируйте и настройте приложение **DjDT** в файле *settings.py*:

```
# settings.py

...

INSTALLED_APPS = [
    # В проекте уже зарегистрировано несколько приложений.
    ...
    # Регистрируем новое приложение в проекте:
    # обязательно ниже, чем django.contrib.staticfiles.
    'debug_toolbar',
]

# MIDDLEWARE — список промежуточных программных слоёв, подключённых к проекту.
# DebugToolbarMiddleware будет обрабатывать информацию из запросов
# и отображать её в панели Django Debug Toolbar.
# Добавьте DebugToolbarMiddleware в самый конец списка.
MIDDLEWARE = [
    ...
    'debug_toolbar.middleware.DebugToolbarMiddleware',
]

# Добавьте в settings.py эту константу, чтобы DjDT знал,
# запросы с каких IP он должен обрабатывать.
INTERNAL_IPS = [
    '127.0.0.1',
]
```

Последний штрих: в головной файл *urls.py* добавьте новое правило для режима отладки:

```
# urls.py
# Импортируем информацию из настроек.

...

from django.conf import settings

urlpatterns = ...

# Если проект запущен в режиме разработки...
if settings.DEBUG:
    import debug_toolbar
    # Добавить к списку urlpatterns список адресов из приложения debug_toolbar:
    urlpatterns += (path('__debug__/', include(debug_toolbar.urls)),)
```

После подключения и настройки **DjDT** дизайн страниц проекта изменится: появится боковое меню с несколькими подразделами. Во вкладке SQL можно увидеть запросы, на основе которых формируется запрошенная страница.

Если Django Debug Toolbar показывает 404 ошибку на странице SQL

Некоторые версии Django Debug Toolbar не будут корректно работать, если в корневом *urls.py* подключение функции `static()` описано выше подключения дебаг-панели.

В этом случае надо подключать тулбар до подключения функции `static()`:

```
# корневой urls.py
...
# Подключаем дебаг-панель:
if settings.DEBUG:
    import debug_toolbar
    # Добавить к списку urlpatterns список адресов
    # из приложения debug_toolbar:
    urlpatterns += (path('__debug__/', include(debug_toolbar.urls)),)

# Подключаем функцию static() к urlpatterns:
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Большинство Django-программистов подключают **DjDT** в самом начале работы над проектом: это упрощает разработку и помогает избежать многих проблем. Обычному пользователю эта панель не нужна, поэтому **DjDT** настраивают так,

чтобы она была видна только в режиме разработки (при настройке `DEBUG = True` в *settings.py*).