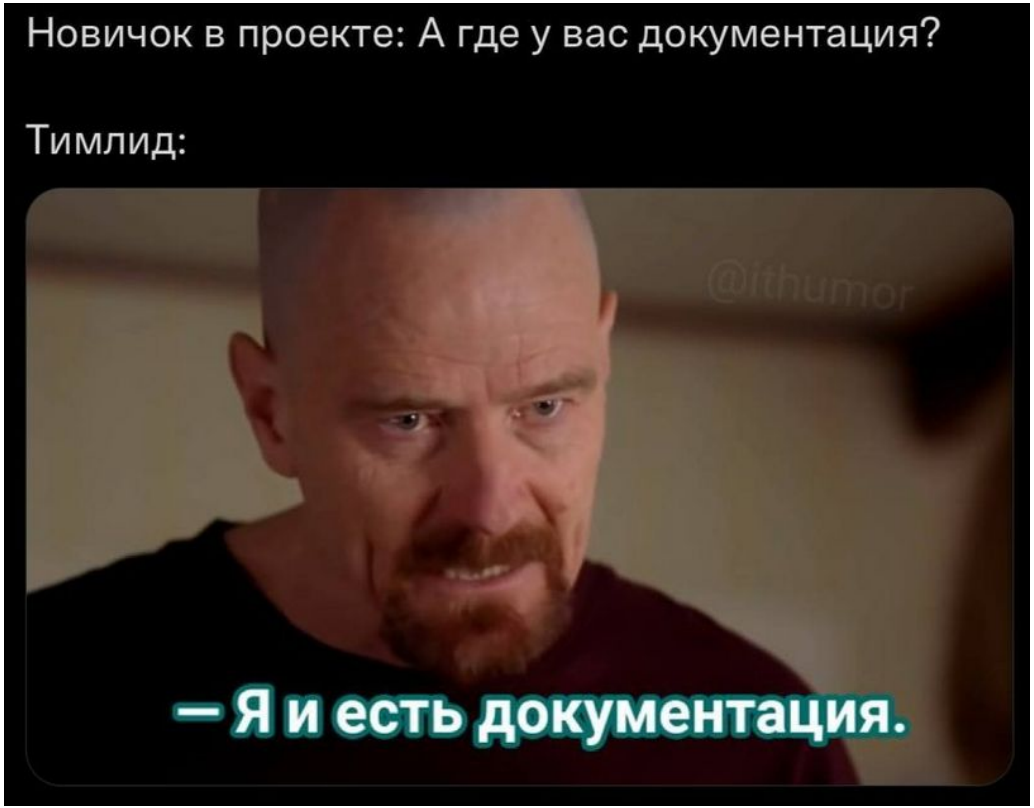
Инструменты автодокументирования

Лекция №7, СибГУ, БИН-19-1, ИСИС, Сафиуллина Н.Ф.

Введение


Новичок в проекте: А где у вас документация?
Тимлид:
@ithumor
— Я и есть документация.

За и против документации

Против:

– нужно выбрать систему

– увеличивается время разработки одной функциональной единицы

– надо поддерживать актуальность

– всегда есть исходный код вместо документации

За:

– легче обучать новичков

– проще делать доработки и писать ТЗ

– спасает от изобретения “велосипедов”

– отсутствие документации станет причиной негативных эффектов

Ловушка будущего

– код превратится в легаси код, работать с которым без документации станет испытанием

– технические задания будут в виде хотелок, не учитывающих то, как есть сейчас

– команда разработки будет тратить много ресурсов консультируя пользователей

Виды документации

1) Техническое задание
2) Пользовательские истории, users story
3) Комментарии в коде
4) Сценарии
5) Руководство пользователя
6) Руководство разработчика
7) Документ адресный справочник или маршрутизатор

```
# Соглашение №0102, ТЗ 012,
# мы не можем проводить операции в валюте,
# отличающейся от валюты
# указанной в договоре с клиентом

if c.val == ag.val:
    # если валюта счета == валюта договора,
    # то провести транзакцию
    do_transaction()
else:
    # иначе: отклонить транзакцию
    rollback_tran()
```

Инструменты автодокументирования

Doxygen – инструмент для создания документации из источников C++, C, Objective-C, C#, PHP, Java, Python, IDL, Fortran, D, VHDL.

Swagger – инструменты Swagger облегчают работу по созданию и поддержке вашей документации по API.

Sphinx – позволяет легко создавать интеллектуальную и красивую документацию.

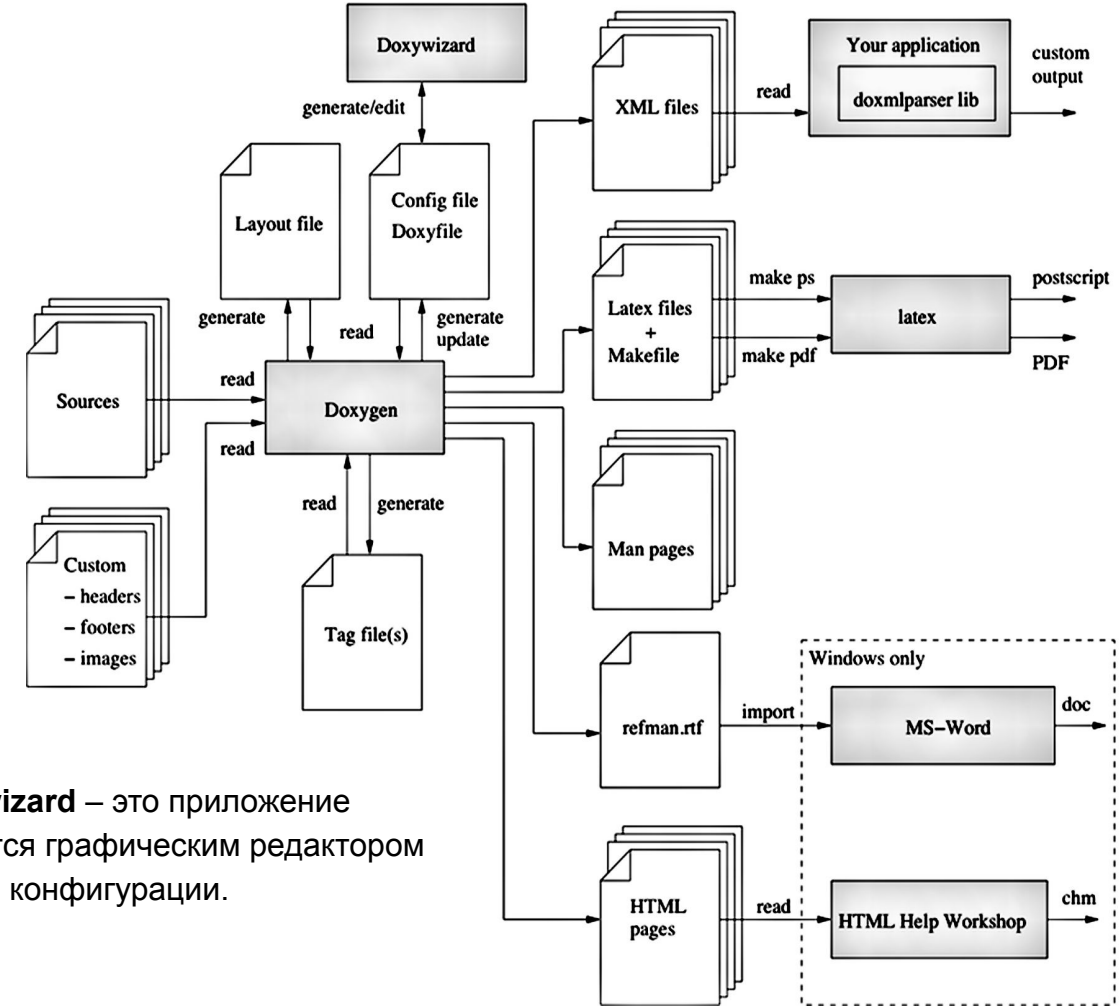
Pydoc – модуль стандартной библиотеки Python, автоматически генерирует документацию из модулей Python.

Pdoc – простой консольный инструмент Python и библиотека для автоматического создания документации API для модулей Python.

Pyment – создание, обновление или преобразование docstring в существующих файлах Python.

Doxygen

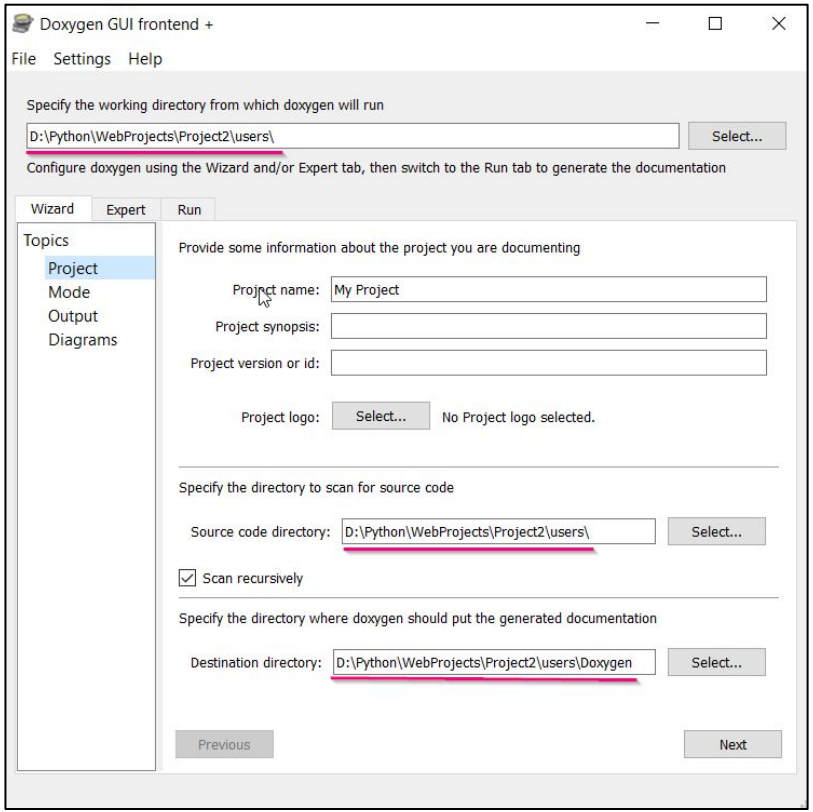
Doxygen — инструмент для создания документации из аннотированных источников C++, но он также поддерживает другие языки программирования: C, Objective-C, C#, PHP, Java, Python, IDL (разновидности Corba, Microsoft и UNO/OpenOffice).), Fortran и в некоторой степени D.



doxygen— это основная программа, которая анализирует исходный код и генерирует документацию.

doxywizard – это приложение является графическим редактором файла конфигурации.

Doxygen: настройка




```
doxygen -g <файл конфигурации>
```

Важные параметры файла настройки:

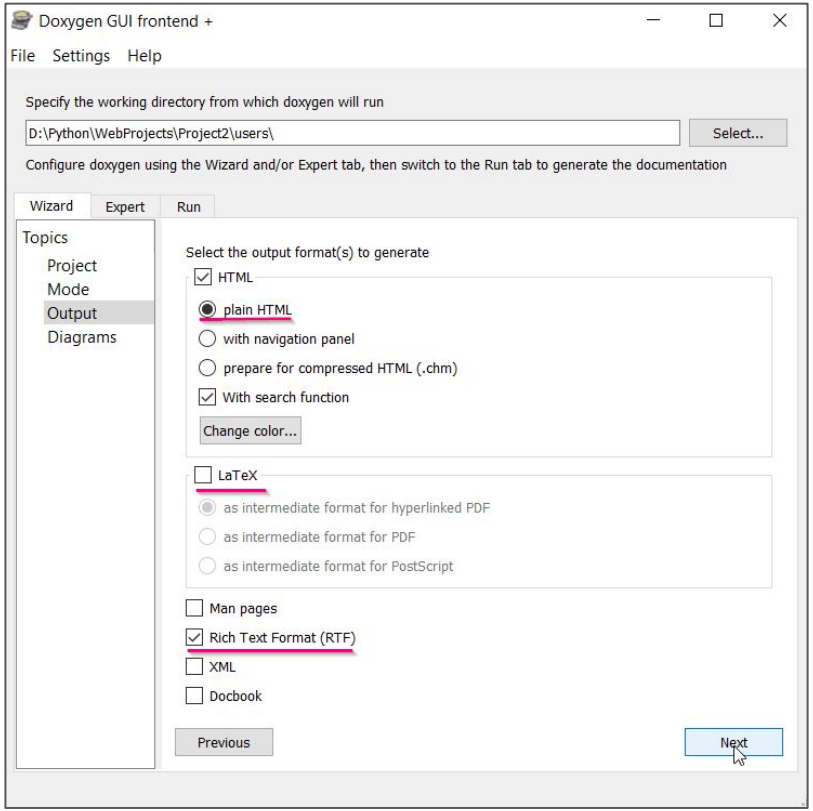
Specify the work directory from which doxygen will run – откуда взять

Source code directory – где код

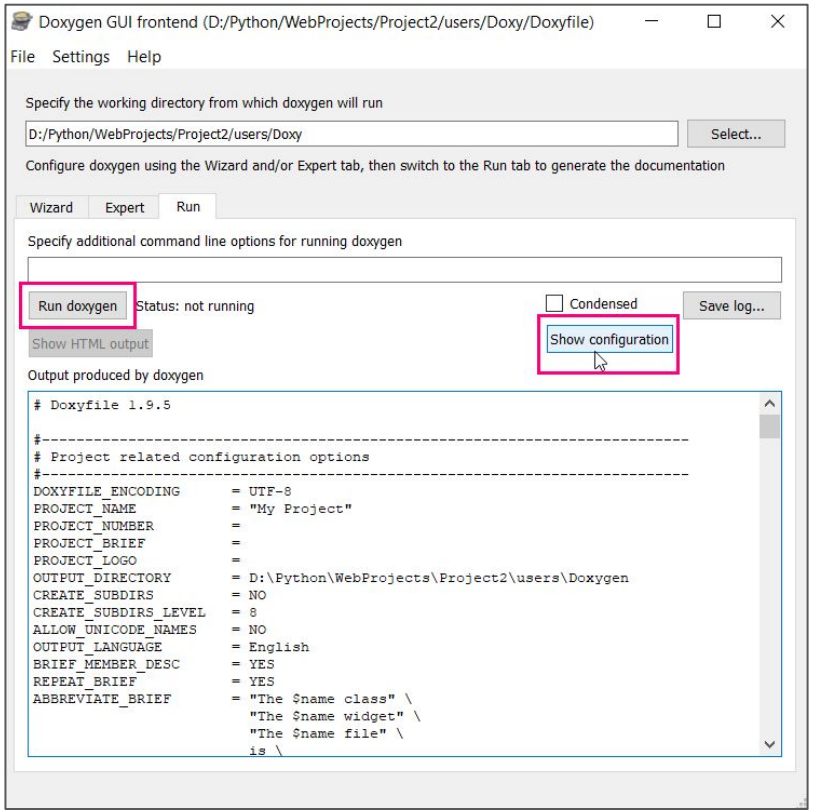
Scan recursively – перебирать папки рекурсивно

Destination directory – куда сохранить документацию

Doxygen: выбор формата документации

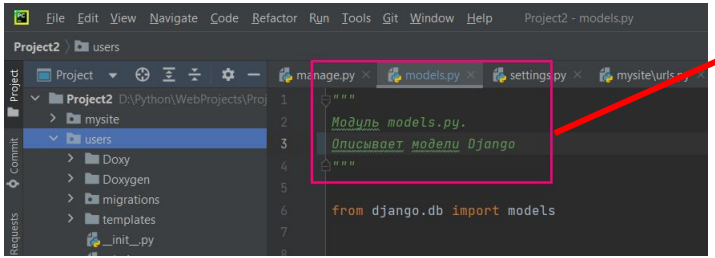


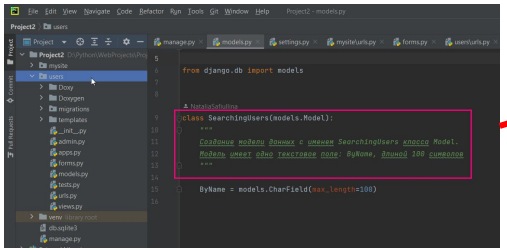
Запуск doxygen

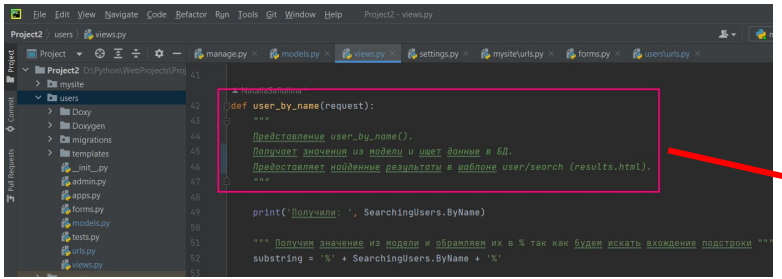



```
OUTPUT_DIRECTORY =
D:\Python\WebProjects\Project2\users\Doxygen

PYTHON_DOCSTRING = YES

INPUT = D:\Python\WebProjects\Project2\users

FILE_PATTERNS = *.c \
                *.py \
                *.pyw \

GENERATE_HTML = YES
HTML_OUTPUT          = html
HTML_FILE_EXTENSION = .html

GENERATE_RTF = YES
RTF_OUTPUT           = rtf

OUTPUT_LANGUAGE = Russian
```

Python docstring








```
"""
Модуль models.py.
Описывает модели Django
"""

from django.db import models

class SearchingUsers(models.Model):
    """
    Создание модели данных с именем
    SearchingUsers класса Model.
    Модель имеет одно текстовое поле:
    ByName, длиной 100 символов
    """

    ByName = models.CharField(max_length=100)
```

```
def user_by_name(request):
    """
    Представление user_by_name().
    Получает значения из модели и ищет данные в БД.
    Предоставляет найденные результаты
    в шаблоне user/search (results.html).
    """
```

Встроенные инструменты Python

```
print(user_by_name.__doc__)
```

```
    Представление user_by_name().
    Получает значения из модели и ищет данные в БД.
    Предоставляет найденные результаты в шаблоне user/search (results.html).
```

Любой объект кода Python имеет атрибут __doc__, который Python хранит с другими прочими параметрами этого объекта. Если мы оформляем комментарии в виде Docstring, то этот атрибут будет заполнен этими комментариями.

Встроенные инструменты Python: PyDoc и PyMent

```
D:\Python\>pip install pyment
D:\Python\WebProjects\Project2\users>pyment views.py
```

Pyment создаст файл: `views.py.patch:`

```
# Patch generated by Pyment v0.3.3

--- a/views.py
+++ b/views.py
@@ -12,8 +12,10 @@


 def get_name(request):
Представление get_name(). Отрисовывает форму и сохраняет введенные значения в
модель.
+
+    :param request:
+

     print('Получен метод запроса: ', request.method)
```