KAZAKH KBTU BRITISH
TECHNICAL
UNIVERSITY

BACKEND FRAMEWORK (DJANGO)

Lesson 9

Logging

Why logging is important?

Levels of Log Message

1. **Debug** — used to give detailed information
2. **Info** — used to confirm that things are working as expected
3. **Warning** — used to indication that something unexpected happened
4. **Error** — used to tell that software has not been able to perform some function
5. **Critical** — used to tell that software itself may be unable to continue running

```
import logging

logging.basicConfig(
    filename='example.log',
    level=logging.DEBUG)

logging.debug('debug')
logging.info('info')
logging.warning('warning')
logging.error('error')
logging.critical('critical')
```

```
import logging

logging.basicConfig(
    filename='example.log',
    filemode='w',
    level=logging.DEBUG)
```

```
logging.debug('debug')
logging.info('info')
logging.warning('warning')
logging.error('error')
logging.critical('critical')
```

```
import logging

logging.basicConfig(
    filename=‘example.log’,
    filemode=‘w’,
    format='%(levelname)s:%(message)s',
    level=logging.DEBUG)

logging.debug('debug')
logging.info('info')
logging.warning('warning')
logging.error('error')
logging.critical(‘critical')
```

https://docs.python.org/3/library/logging.html#logrecord-attributes

Logging Variable Data

Logging components

1. Loggers — expose the interface that application code directly uses
2. Handlers — send the log records to the appropriate destination.
3. Filters — determines which log records to output
4. Formatters — specify the layout of log records in the final output

Handlers

1. StreamHandler
2. FileHandler
3. methods
    1. setLevel()
    2. setFormatter()
    3. addFilter() / removeFilter()

https://docs.python.org/3/howto/logging.html#handler-basic

https://docs.python.org/3/howto/logging.html#useful-handlers

Formatters

1. logging.Formatter
2. %(asctime)s *default value*: "**%Y-%m-%d %H:%M:%S**"

https://docs.python.org/3/howto/logging.html#formatters

Configuring Logging

```
import logging

logger = logging.getLogger(‘simple_example')
logger.setLevel(logging.DEBUG)

ch = logging.StreamHandler()
ch.setLevel(logging.DEBUG)

formatter = logging.Formatter('%(asctime)s - %(name)s - %
(levelname)s - %(message)s')

ch.setFormatter(formatter)

logger.addHandler(ch)

logger.debug('debug message')
logger.info('info message')
logger.warning('warn message’)
logger.error('error message’)
logger.critical('critical message')
```

Logging in Django

```
LOGGING = {
    'version': 1,
    'disable_existing_loggers': False,
    'formatters': {
        'verbose': {
            'format': '%(levelname)s %(asctime)s %(message)s',
        },
    },
    'handlers': {
      ‘test_file': {
            'level': 'DEBUG',
            'class': 'logging.FileHandler',
            'filename': 'test.log',
            'formatter': 'verbose'
        },
    },
    'loggers': {
      'test': {
            'handlers': ['test_file'],
            'level': 'DEBUG',
        },
    },
}
```

File Uploading

FileField, ImageField

1. upload_to
2. validators

FileField, ImageField

```
1. from rest_framework.parsers import FormParser, MultiPartParser, JSONParser
2. parser_classes = (MultiPartParser, FormParser, JSONParser,)
```

Settings configuration

1. *settings.py*
   - MEDIA_URL = '/media/'
   - MEDIA_ROOT = os.path.join(BASE_DIR, ‘media’)
2. *urls.py*
   - from django.conf import settings
   - from django.conf.urls.static import static
   - urlpatterns = […] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

Questions?