Урок 10

Модуль pickle

The pickle module implements a fundamental, but powerful algorithm for serializing and de-serializing a Python object structure.

pickle.dump(*obj*, *file*[, *protocol*])¶

pickle.load(*file*)

pickle.dumps(*obj*[, *protocol*])

pickle.loads(*obj*[, *protocol*])

what can be pickled

- None, True, and False
- integers, long integers, floating point numbers, complex numbers
- normal and Unicode strings
- tuples, lists, sets, and dictionaries containing only picklable objects
- functions defined at the top level of a module
- built-in functions defined at the top level of a module
- classes that are defined at the top level of a module
- instances of such classes whose __dict__ or the result of calling __getstate__() is picklable (see section *The pickle protocol* for details).

Установка php.

sudo apt-get install php5-fpm

```
server {
        listen 80 ;
        server_name best.local;
        root         /home/zarik/www/marriage_cms/web_best/;
        index   index.php;
        client_max_body_size   100m;
        location ~ \.php$ {
        # fastcgi_pass 127.0.0.1:9000;
        fastcgi_pass unix:/var/run/php5-fpm.sock;
        fastcgi_index  index.php;
        fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
        fastcgi_param  SCRIPT_NAME              $fastcgi_script_name;
        fastcgi_param  QUERY_STRING             $args;
        include     fastcgi_params;
        }
  location / {
        index index.php;
        try_files $uri /index.php?$args;
        } }
```

Логирование.

The cast of players¶

A Python logging configuration consists of four parts:

- Loggers
- Handlers
- Filters
- Formatters

log levels:

- `DEBUG`: Low level system information for debugging purposes
- `INFO`: General system information
- `WARNING`: Information describing a minor problem that has occurred.
- `ERROR`: Information describing a major problem that has occurred.
- `CRITICAL`: Information describing a critical problem that has occurred.

Логирование

```
import logging

# Get an instance of a logger
logger = logging.getLogger(__name__)

def my_view(request, arg1, arg):
        if bad_mojo:
            # Log an error message
            logger.error('Something went wrong!')
```

```
# Get an instance of a specific named logger
logger = logging.getLogger('project.interesting.stuff')
```

Логирование в django

- logger.critical()
- logger.error()
- logger.warning()
- logger.info()
- logger.debug()

```
# import the logging library

import logging

# Get an instance of a logger

logger = logging.getLogger(__name__)

logger.error('Something went wrong!')
```

Настройка логирования

```
LOGGING = {
    'version': 1,
    'disable_existing_loggers': True,

    'formatters': {
        'verbose': {
            'format': '%(levelname)s %(asctime)s %(module)s %(process)d %(thread)d %(message)s'
        },
        'simple': {
            'format': '%(levelname)s %(message)s'
        },
    },
```

```
        }
```

Настройка логирования

```
'handlers': {
        'null': {
            'level':'DEBUG',
            'class':'django.utils.log.NullHandler',
        },
        'console':{
            'level':'DEBUG',
            'class':'logging.StreamHandler',
            'formatter': 'verbose'
        },
        'mail_admins': {
            'level': 'ERROR',
            'class': 'django.utils.log.AdminEmailHandler'
        }
    },
```

Настройка логирования

```
'loggers': {
        'django': {
            'handlers':['null'],
            'propagate': True,
            'level':'INFO',
        },
        'django.request': {
            'handlers': ['console'],
            'level': 'ERROR',
            'propagate': False,
        },
        'main.views': {
            'handlers': ['console'],
            'level': 'DEBUG',
            'propagate': False,
        }
    }
```

Стандартные логеры в django

django – логер принимающий все сообщения. Сообщения не записываются непосредственно в этот логер.

django.request

Принимает сообщения связанные с процессом обработки запросов. 5XX ответы отправляют ERRORсообщения, 4XX ответы – WARNING сообщения.

django.db.backends

Сообщения связанные с взаимодействием кода с базой данной. Например, каждый SQL запрос создает DEBUG сообщение в этот логер.

class **AdminEmailHandler([***include_html=False***])**

Этот логер отсылает e-mail администраторам сайта с принятый сообщением.

Если сообщение содержит атрибут request, полная информация о запросе будет включена в e-mail.

Если сообщение содержит стек трассировки(*пер.* stack trace information), он будет включен в e-mail.

Работа с сессией и куками

To enable session functionality, do the following:

- Edit the `MIDDLEWARE_CLASSES` setting and make sure it contains

```
django.contrib.sessions.middleware.SessionMiddleware
```

By default, Django stores sessions in your database BUT

`SESSION_ENGINE`

```
django.contrib.sessions.backends.cache
django.contrib.sessions.backends.file
django.contrib.sessions.backends.signed_cookies
fav_color = request.session.get('fav_color', 'red')
request.session['user'] = pickle.dumps(user)
```

request.COOKIES.get('visits', '0') ; response.set_cookie('visits', visits + 1)

Работа с регулярными выражениями

```
import re

c = re.compile('ab*')
```

Pattern object

match() - Определить, начинается ли совпадение регулярного выражения с начала строки

search() - Сканировать всю строку в поисках всех мест совпадений с регулярным выражением

findall() - Найти все подстроки совпадений с регулярным выражением и вернуть их в виде списка

finditer() - Найти все подстроки совпадений с регулярным выражением и вернуть их в виде итератора

Match object

group() - Вернуть строку, сошедшуюся с регулярным выражением

start() - Вернуть позицию начала совпадения

end() - Вернуть позицию конца совпадения

span() - Вернуть кортеж (start, end) позиций совпадения

Свойство модели get_absolute_url

Define a `get_absolute_url()` method to tell Django how to calculate the canonical URL for an object.

```
def get_absolute_url(self):
    return "/people/%i/" % self.id

def get_absolute_url(self):
    from django.core.urlresolvers import reverse
    return reverse('people.views.details', args=[str(self.id)])
```