# Добавление курса BTC. Динамический импорт модулей.

| **№ урока:** | 7 | **Курс:** | Python Practice |
|---|---|---|---|

| **Средства обучения:** | Интерпретатор Python, virtualenv, текстовый редактор. |
|---|---|

## Обзор, цель и назначение урока

Цель урока: реализовать API для получения курса BTC: BTC => UAH, BTC => RUB, BTC => USD. Добавить название модуля в таблицу курсов для динамического импорта нужного модуля в зависимости выбранного курса валют.

## Изучив материал данного занятия, учащийся сможет:

- Закрепить навыки работы с библиотекой requests.
- Закрепить навыки парсинга данных в разнообразном формате.
- Ознакомиться со стандартной библиотекой importlib для динамического импорта модулей.
- Использовать importlib библиотеку в коде проекта golden-eye.

## Содержание урока

1. Краткое резюме того, что мы имеем на данный момент. Обзор дальнейших действий.
2. Обзор нужных API для получения курса BTC.
3. Реализация API путем наследования от базового класса _Api — быстро и просто благодаря проведенному ранее рефакторингу.
4. Рассмотрение неудобств текущего подхода при обновлении определенного курса валют.
5. Варианты решения — хранить модуль в базе, вместе с курсом.

## Резюме

- Для получения новых курсов валют BTC => UAH, BTC => RUB, BTC => USD нам необходимо реализовать несколько новых API. Для курса BTC => UAH будем использовать уже реализованное API ПриватБанка. Необходимо только добавить парсинг нужного значения, модифицировать скрипт.
- Для получения курса валют BTC => RUB и BTC => USD будем использовать API https://api.cryptonator.com/api/ticker/, url для получения курсов — https://api.cryptonator.com/api/ticker/btc-rub и https://api.cryptonator.com/api/ticker/btc-usd
- Добавим новый модуль cryptonator в package API и добавим реализацию API путем наследования класса _Api.
- Добавим отправку запроса и обработку ответа в новом модуле.
- При текущем подходе обновления курса валют, нам необходимо точно знать, в каком API модуле реализована логика получения каждого конкретного курса, что не совсем удобно. Удобнее было бы, если бы мы просто указывали, какой именно курс мы хотим сейчас обновить, а система сама бы уже знала, какой модуль API отвечает за это обновление.
- Самый простой вариант для реализации подобной логики — это добавление в базу данных информации о том, какой модуль отвечает за обновление того или иного курса валют. Добавим поле module в таблицу CurrencyRate, not null.
- Теперь система уже будет хранить информацию о том, какой именно модуль API нужно использовать при обновлении курса, осталось только добавить динамический импорт нужного модуля, создание класса Api и вызова в нем метода update_rate.



CyberBionic systematics

CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- Стандартная библиотека importlib позволяет импортировать модули динамически, то есть без явного указания инструкции import. Функция importlib.import_module ожидает обязательным параметром название модуля, осуществляет импорт этого модуля и в качестве результата выполнения возвращает сам объект модуля. У этого объекта, как и при обычном импорте, доступны описанные в модуле классы, функции, переменные и т.п.
- Второй, необязательный параметр функции importlib.import_module — это package, в котором находится модуль.
- Добавим динамический импорт модулей в код проекта, проверим на тесте для курса валют USD => RUB.

## Закрепление материала

- Для чего нужно логирование в БД? В чем отличия от логирования в файл?
- Какая минимальная информация нужна при сохранении логов общения с удаленными API?
- Как привязать индекс к полю БД с помощью библиотеки peewee?
- Как вывести на печать трейсбек исключения с помощью traceback библиотеки? Как получить строку с трейсбеком ошибки?

## Дополнительное задание

Задание
Модифицировать тесты для получения всех остальных курсов валют, проверить работоспособность проекта после внесенных изменений, корректность работы.

## Самостоятельная деятельность учащегося

Задание 1
Добавить в API cryptonator проверку ответа, а именно - проверку на то, что параметры "base" и "target" соответствуют переданным валютам. Добавить проверку, что ответ содержит все ожидаемые поля.

Задание 2
Структура тестов проекта имеет вид:

```
tests:
        __init__.py
        core_test.py
        api_test.py
run.py
importer.py
```

Необходимо добавить код в модуль importer, а именно: динамический импорт модулей api_test, core_test, вызов метода main_test класса Test, который описан в обоих модулях. Примерный вид класса Test для модуля api_test:

```
class Test():
    def main_test(self):
        print(f"{__name__}.Test.main_test")
        print("Testing api...")
```

## Рекомендуемые ресурсы

https://docs.python.org/3/library/importlib.html#importlib.import_module



CyberBionic Systematics ® 2018
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com