Курсова робота

з курсу «Основи програмування»

Тема: Розроблення та використання структур даних.

BellBot

**1. Визначення функціональних та нефункціональних вимог до програми, яка буде розроблятися.**

Опис функціональних вимог:

1. Функція: автоматичні повідомлення.

* Опис: ця функція забезпечує автоматичні відповіді користувачу для затвердження певної інформації(наприклад, повідомлення про отримання замовлення).
* Вхідні дані: замовлення, зроблене користувачем і його підтвердження.
* Джерело даних: користувач.
* Вихідні дані: повідомлення від бота.
* Пункт призначення повідомлення: особистий діалог з користувачем.
* Передумова: правильно введене замовлення з відповідними назвами страв для замовлення.

2. Функція: відображати категорії для вибору.

* Опис: ця функція забезпечує надання користувачу зручну можливість вибрати товар з переліку можливих. Користувачу буде надано перелік кнопок з категоріями, завдяки чому він швидко зможе перейти з категорії в категорію, або повернутись назад.
* Вхідні дані: натиснута кнопка користувачем.
* Джерело даних: користувач.
* Вихідні дані: немає вихідних даних, натомість користувачу буде надана наступна категорія для вибору товару. Після завершення всіх категорій буде виведено список існуючих пропозицій для користувача.
* Пункт призначення: особистий діалог з користувачем.
* Передумова: натиснута кнопка з назвою категорії користувачем для відповідних анступних дій.

3. Функція: прийом замовлення.

* Опис: ця функція забезпечує остаточне замовлення користувачем. Всі замовлення користувачем будуть виведені на екран для підтвердження або спростування замовлення. Як результат використовує функцію 1, яка буде виводити автоматичне повідомлення згідно з діями користувача.
* Вхідні дані: остаточне замовлення користувача та його підтвердження або скасування.
* Джерело даних: користувач.
* Вихідні дані: інформація введена користувачем
* Пункт призначення: особистий діалог з користувачем
* Передумова: замовлення, введене належним чином

Опис нефункціональних вимог:

1. Вимоги до інтерфейсу:

* Апаратні інтерфейси:
* Інтерфейси програмного забезпечення:

- для Python середовища використовується PyCharm Community 2017.3.2, що є одним з найпопулярніших та найкращих програм для цієї мови програмування. Зручний для користування, оскільки дає безліч підказок розробнику, надає швидкий перехід між файлами, дає змогу приємнати веб-розробку за допомогою flask чи django.

- сам Viber бот буде мати стандартний вигляд особистого діалогу у Viber з переважаючим фіолетовим кольором. Присутні всі функції, які є влаштовані у Viber.

* Комунікаційні інтерфейси:

- передача даних з Viber в нашу програму.

- передача замовлення на пошту.

2. Вимоги виконання:

- відповідь користувачу повинна проводитись не довше ніж за 5 секунд після насилання запиту на відповідь.

- кожен час відповіді повинен бути відслідкований для майбутньої подифікації програми в разі виникнення проблем.

3. Операційні вимоги:

- продуктивність: система повинна обслуговувати декілька користувачів водночас.

- надійність: система повинна бути стійкою до різних типів поломок, тому потрібно вести журнал всіх операцій.

- безпека та конфіденційність: вся фінансова та особиста інформація не повинна бути надана третім лицям .

- база даних: база даних може змінюватись лише розробником, ні в якому разі користувачем, не надаючи йому доступ для внесення будь яких змін.

**2.Опис можливостей пакетів json, xml, html.**

**JSON**

JSON(JavaScript Object Notation) – формат передачі даних, стильований під мову JavaScript, який також підтримується у Python.

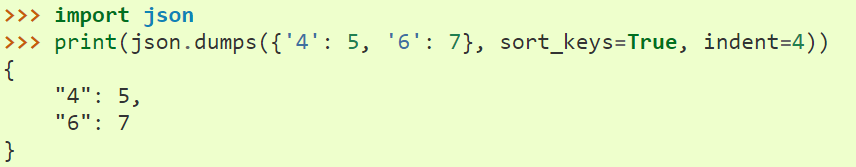
Є багато застосувань у пакету json. За його допомогою можна виконувати різні операції в першу чергу для покращення читабельності коду.

Широко використовується для веб розробки, в такому ключі від доречний в задачах обміну даними як між браузером і сервером. Також використовуєтсья у базах даних для зберігання складних структур. У json використовуються такі форми як об’єкти, списки, булеві значення, рядки символів та інші.

Але оскільки json використовує синтаксис мови JavaScript, то це зачасту створює низку проблем безпеки, тому що використовується функція eval(), яка не має попередньої перевірки. Проте, варто зазначити, що json займає менше місця в порівнянні з його основним конкуретом по форматах обміну даними – xml, що робить його на даний момент більш широко використовуючою мовою.

Повертаючись до застовування цього пакету у Python, можна навести декілька найбільш популярних прикладів методів, які є присутні в цьому пакеті. Серед них такі, як dumps(), який дозволяє відформатувати виведення тексту на екран; loads(), який десереалізує об’єкт в Python. Також можна закодовувати та розкодовувати інформацію за допомогою таких функцій як JSONDecoder та JSONEncoder.

Приклад json коду написаному на Python:



**XML**

Інтерфейс Python для виконання XML є погрупований в xml пакетах. Виконуючі модулі xml не є безпечні проти зловмисно створених даних, які сконструйовані нападниками і можуть виконувати атаки по серверу, доступатись до локальних даних, створювати звязки з іншими пристроями або обходити різні паролі. Все це може принести неабияку шкоду користувачу, тому використовуючи цей пакет, потрібно були особливо обережним.

Проте, було створено два пакети для захисту даних користувача – defusedxml та defusedexpat.

Defusedxml це python пакет з модифікованими підкласами всі stdlib xml посилачів, щоб запобігти будь якісь потенційно небезпечній операції. Використання цього пакету рекомендується для будь якого сервер коду, який пересилає ненадійні xml дані. Пакет також має розширену документацію про різні напади на систему та як хїх запобігти.

Defusedexpat постачає нас покращеним pyexpat модулем, який має протидію від розширених атак. Defusedexpat модуль все ще дозволяє величезну і налаштовану кількість розширених суб’єктів протидії. Покращення можуть бути включені в деякі майбутні реалізації Python.

Ось приклад як парсити xml в Python:



**HTML**

HTML(HyperText Markup Language) – стандартна мова розмітки веб-сторінок в інтернеті. Більшість веб-додатків та сторінок створюються за допомогою мови html.

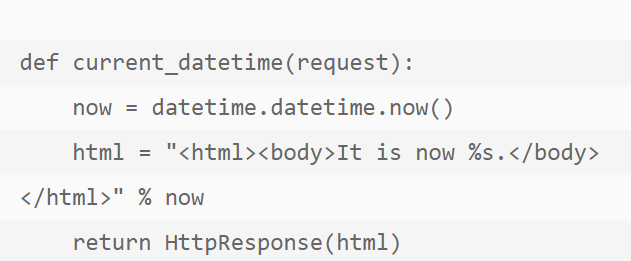
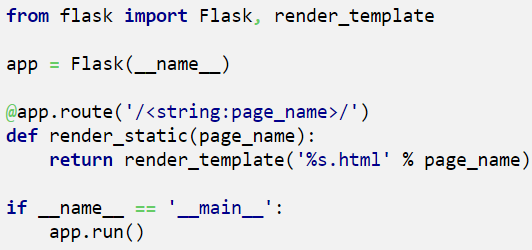
Загальна структура html розмітки:



Модуль HTMLParser служить для інтегрування текстових файлів сформатованих в html виді в Python.

Ось кілька прикладів методів, які присутні та найбільш вживані в модулі HTMLParser: feed(data) – передає текст в форматі str; close() – примусове закриття файлу; reset() – скидає всі невиконані дані; getpos() – повертає номер рядка, на якому знахоиться користувач; чимоло методів, які починаються з приставки handle\_ і виконують різні функції. Вся інформація про цей парсер є доступна в офіційній документації Python.

Два найбільш популярні та поширені способи інтегрування html в мову програмування Python – це Flask та Django. За допомогою цих програм, можна писати back end для веб-сторінок. Ось приклади використання цих програм у Python:



За допомогою html можна робити безліч операцій для покращення інтерфейсу сторінки такі, як колір, тим носія, розмір в пікселях, дату, час скрипт, текстові рядки, таблиці даних, списки та багато іншого.

Цей бот працюватиме з різними типами даних, отримуючи їх за допомогою двох основних API, що дадуть змогу надалі визначати необхідний функціонал програми.

DeLuxe Food Delivery API – відповідатиме за отримання даних про користувача з json файлу.

Перелік даних , отримуваних від DeLuxe Food Delivery API:

* Номер телефону користувача \* (рис.1);
* Ім’я користувача \* (рис.1);
* Місце доставки \* (рис.1);
* Категорії товарів (рис.2)
* Підкатегорії товарів (рис.2)
* Окремі артикули (рис.3)

Viber Python Bot API – відповідатиме за роботу з користувачем у службі Viber, де, в основному, відбуватиметься робота з користувачем.

Перелік даних , отримуваних від Viber Python Bot API:

* Viber user ids – array of strings
* Повідомлення – list;
* Підпис – string;
* Web server URL – string;
* Унікальне ID повідомлення – string;
* Медіа – URL string(тільки формату jpeg);
* Чи користувач вже підписаний – Boolean;
* Локація – об’єкт класу Location

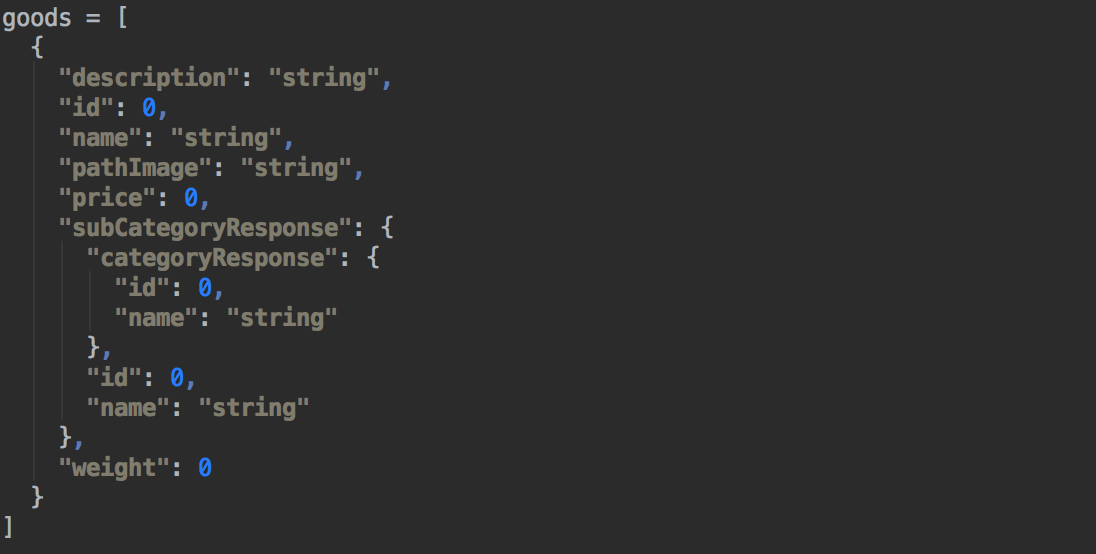
\*обов’язково вводиться при замовленні



(рис.1)



(рис. 2)



(рис.3)