**Lesson 25. Telegram Bot**

Третя чверть нашого курсу позаду і як завжди маємо перевірити себе та з’ясувати свій прогрес, що зрозумілось добре, а над чим ще необхідно зупинитись додатково. Як ми твердо переконані, перевірити себе можливо тільки працюючи самостійно і саме цьому присвячений даний урок.

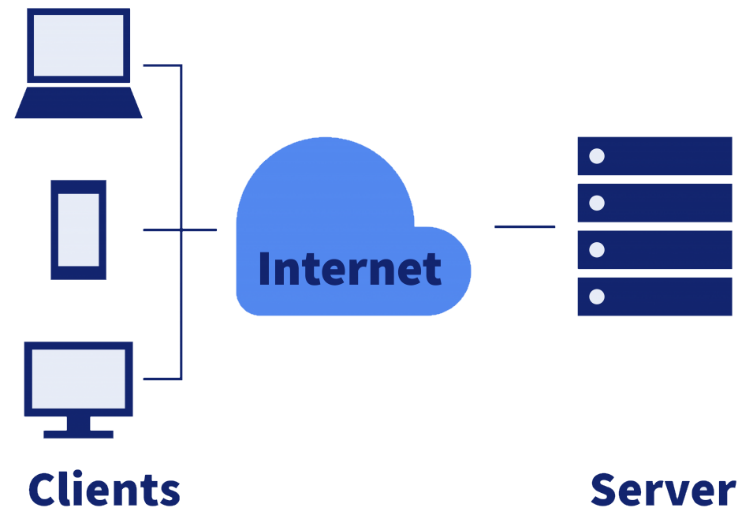
Робота складається із двох частин:

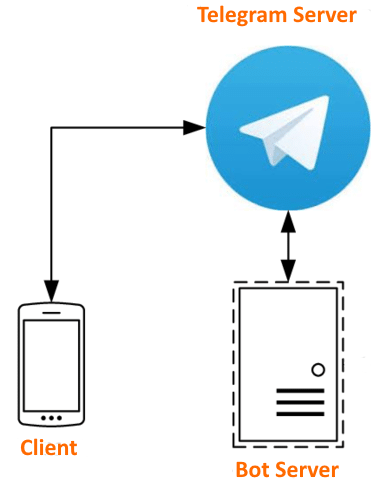
**Clients and Servers**

Перш ніж говорити про те, як працюють телеграм боти, необхідно трошки детальніше розібратись із тим, як працює інтернет загалом.

Усі пристрої підключені до мережі інтернет можна поділити на **клієнти** і **сервери**. Клієнти – всі користувачі інтернету, а сервери – машини, що їх обслуговують.

Коли ми намагаємось потрапити на якусь веб-сторінку (наприклад instagram), наш комп'ютер (планшет, смартфон тощо) надсилає запит на сервер, що зберігає цю веб-сторінку. Сервер обробляє запит та надсилає назад веб-сторінку нашому комп'ютеру, де ми її переглядаємо за допомогою веб-браузера.



Цей процес дозволяє нам переглядати веб-сторінки, відправляти електронні листи та користуватися багатьма іншими інтернет-сервісами.

У випадку із Telegram-ботами сервера матимемо цілих два:

* Telegram-сервер – для зв’язку із клієнтами (вже готовий, створений самим телеграмом);
* Bot-сервер – для забезпечення головної логіки та функціоналу бота (маємо запрограмувати самі, та налаштувати зв’язок із Telegram-сервером).

Для того, щоб створити сервер телеграм бота на своєму комп’ютері, необхідно його спочатку зареєструвати в Telegram. Це можна зробити за допомогою батька всіх ботів – @BotFather (<https://t.me/BotFather>).

**Create New Bot**

Перейшовши за посиланням, необхідно в меню команд знайти команду ***/newbot*** і далі дотримуючись інструкцій вказати для бота назву та **username**.

При успішній реєстрації BotFather видасть вам спеціальний і дуже секретний ключ (**token**) для доступу до вашого бота. Саме ці значення (token та username) знадобляться для подальшої розробки бота.

**Libraries**

Тест складається із 20 питань, вартість правильної відповіді на кожне – 1 бал.

Після проходження тесту порахуйте кількість балів поділивши кількість правильних відповідей на 2 і заокругливши в більшу сторону.

**Завдання 1**. (5 балів) Cтворіть клас **User**, що відображає користувача програми чи веб-сайту. Даний клас повинен мати наступні **поля**:

* *printInfo* (виводить інформацію про юзера у консоль в наступному вигляді: *User [Jane Doe, email: someemail@gmail.com]*).

Клас повинен мати наступні **конструктори**:

* З одним одним параметром, через який при створенні об’єкта вказується email юзера;
* З трьома параметрами, через які вказуються всі три поля при створення об’єкта.

Протестуйте роботу методів, створивши **два об’єкти класу** за допомогою двох різних конструкторів.

**Завдання 2**. (5 балів) Створіть клас **Employee** (працівник), що унаслідує клас User. Даний клас повинен додатково мати поля:

* *specialty* (спеціальність працівника);
* salary (заробітна плата за місяць, грн).

Клас повинен мати наступні **методи**:

* *get* i *set* для кожного із власних полів;
* *setSalary* (повинен встановлювати нове значення заробітної плати не менше, аніж мінімальна сума – 8000 грн, ця ж сума є значенням за замовчуванням);
* *getAnnualSalary* (повертає річну заробітну плату працівника, грн);
* *raiseSalary* (піднімає заробітну плату на певну кількість відсотків, які вказуються в параметрі методу);
* *printInfo* (успадкований від User, але виводить інформацію про працівника у консоль в наступному вигляді: *Employee [John Doe, email: example@email.com, specialty: manager, salary: 9600.0 UAH]*).

Клас повинен мати тільки один **конструктор**:

* З одним одним параметром, через який при створенні об’єкта вказується email;

Створіть **один об’єкт класу** і протестуйте роботу усіх його власних, або переписаних методів.

**Additional Task**

**Завдання 3**. Створіть клас **Customer** (клієнт), що унаслідує клас User. Даний клас повинен додатково мати одне поле:

* *balance* (кількість коштів на рахунку, грн, дефолтне значення – 0, не може мати значень менше 0).

Клас повинен мати наступні **методи**:

* *getBalance* (повертає поточне значення коштів на рахунку);
* *debit* (додає певну кількість коштів на рахунок);
* *credit* (знімає певну кількість коштів з рахунку, якщо запитаних коштів недостатньо, то вони не знімаються взагалі, а в консоль виводиться відповідне повідомлення);
* *printInfo* (успадкований від User, але виводить інформацію про працівника у консоль в наступному вигляді: *Customer [Jack Doe, email: customer@gmail.com, balance: 300.0 UAH]*).

Клас повинен мати тільки один **конструктор**:

* З одним одним параметром, через який при створенні об’єкта вказується email;

Тепер може порахувати бали і зробити висновки:

**Additional Task**

**Test Part** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / (20)

**Practice Part** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / (10)

**Result** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / (30)

**Висновок.** Підкресліть той варіант, що на вашу думку є найближчий до правди:

* Все круто, мій результат ідеальний;
* Я майже з усім справився/справилась, трохи допрацюю і все буде ок;
* Ну таке… Я багато чого не розумію, треба попрацювати, щоб наздогнати;
* Все погано, я засмучений/засмучена, здається, самостійно працювати не можу.