Лабораторна робота № 7. Створення Application Programming Interface для хмарного додатку

Мета роботи: створити простий HTTP API для хмарного додатку з лабораторної роботи № 4. API має задовольняти REST вимогам побудови інтерфейсів. Продемонструвати роботу API на прикладі 3-4 методів. Кожен метод перевірити з клієнту за допомогою curl або інших аналогічних програм.

Задача: використати код програми з лабораторної роботи № 4 та побудувати приклад 3-4 методів API, які будуть задовольняти REST вимогам. Для передачі даних у методах використовувати JSON або XML формат.

Короткі теоретичні відомості

REST (скор. англ. Representational State Transfer, «передача стану подання») — підхід до архітектури мережевих протоколів, які забезпечують доступ до інформаційних ресурсів. Був описаний і популяризований у 2000 році Роєм Філдінгом (Roy Fielding), одним із творців протоколу HTTP. Найвідомішою системою, побудованою переважно за архітектурою REST, є сучасна Всесвітня павутина.

Дані повинні передаватися у вигляді невеликої кількості стандартних форматів (наприклад HTML, XML, JSON). Мережевий протокол (як і HTTP) повинен підтримувати кешування, не повинен залежати від мережевого прошарку, не повинен зберігати інформацію про стан між парами «запит-відповідь». Стверджується, що такий підхід забезпечує масштабування системи і дозволяє їй еволюціонувати з новими вимогами.

Антиподом REST є підхід, заснований на виклику віддалених процедур (Remote Procedure Call, RPC). Підхід RPC дозволяє використовувати невелику кількість мережевих ресурсів з великою кількістю методів і складним протоколом. При підході REST кількість методів і складність протоколу

суворо обмежені, що призводить до того, що кількість окремих ресурсів має бути великою.

JSON (англ. JavaScript Object Notation, укр. об'єктний запис JavaScript, вимовляється джейсон) — це легкий формат обміну даними між комп'ютерами. JSON базується на тексті, і може бути з легкістю прочитаним людиною. Формат дозволяє описувати об'єкти та інші структури даних. Цей формат головним чином використовується для передачі структурованої інформації через мережу (завдяки процесу, що називають серіалізацією). Розробив і популяризував формат Дуглас Крокфорд.

JSON знайшов своє головне призначення у написанні веб-програм, а саме при використанні технології AJAX. JSON виступає як заміна XML під час асинхронної передачі структурованої інформації між клієнтом та сервером. При цьому перевагою JSON перед XML є те, що він дозволяє складні структури в атрибутах, займає менше місця і прямо інтерпретується за допомогою JavaScript в об'єкти.

Приклад опису даних у форматі JSON.

```
{
   "firstName": "Іван",
   "lastName": "Коваленко",
   "address": {
       "streetAddress": "вул. Грушевського 14",
       "city": "Київ",
       "postalCode": 21000
   },
   "phoneNumbers": [
       "044 123-1234",
       "050 123-4567"
   ]
}
```

Для створення “REST-backend-додатку” можна використати кілька наявних бібліотек, таких яких Restlet Framework [6]. Обмежень на інструментарій для студентів не встановлюється. Але найбільш “природним” є використання інструментарію самого Google - Google Cloud Endpoints [7]. Далі наводяться деякі настанови з відкритої документації Google Cloud Endpoints (https://cloud.google.com/appengine/docs/java/endpoints/helloworld-java-maven#building_and_running_locally).

Перш ніж приступити до розробки REST-додатку в GAE, слід налаштувати Ваше хмарне середовище:

1. Встановити Java 7 SDK та Maven 3.1 + (https://cloud.google.com/appengine/docs/java/tools/maven#configuring_java)
2. Створити проект Cloud Platform Console та налаштувати його квоти.

Найпростіший варіант ознайомлення з процесом розробки REST-додатку в GAE – це вивчити простий приклад, який доступний на git-репозиторії:

```
git clone
https://github.com/GoogleCloudPlatform/appengine-
endpoints-helloworld-java-maven
```

Структура файлів проекту має такий вигляд:

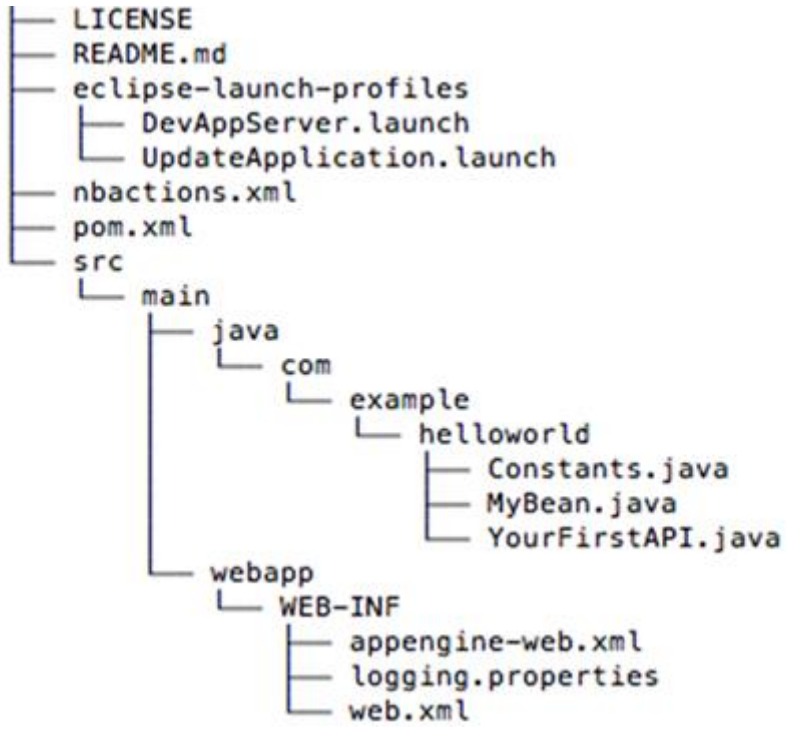

Рис. 5.1. Структура файлів проекту хмарного додатку з REST

Серед іншого, інтерес представляють такі файли:

- pom.xml – налаштування для Maven
- appengine-web.xml – налаштування додатку
- web.xml – “мепінг” для backend-сервлету
- MyBean.java – JavaBean, що використовується для пересилання даних через endpoints.
- YourFirstAPI.java - функціонал бекенду.

Характерний фрагмент файлу MyBean.java:

```
public class MyBean {
  private String myData;
  public String getData() {
    return myData;
  }
  public void setData(String data) {
    myData = data;
  }
}
```

Об’єкт JavaBean використовується для передачі даних з бекенду до клієнта.

Характерний фрагмент файлу YourFirstAPI.java:

```
/** An endpoint class we are exposing */
@Api(name = "myApi",
    version = "v1",
  namespace = @ApiNamespace(ownerDomain = "helloworld.example.com",
        ownerName = "helloworld.example.com",
        packagePath = ""))

/** A simple endpoint method that takes a name and says Hi back */
@ApiMethod(name = "sayHi")
public MyBean sayHi(@Named("name") String name) {
  MyBean response = new MyBean();
  response.setData("Hi, " + name);
  return response;
}
```

Анотація Api задає конфігурацію бекенду. Анотація ApiMethod дозволяє сконфігурувати окремі методи інтерфейсу. У відкритому доступі (в т.ч. на ресурсах Google Cloud Platform) є чимало прикладів з розгортання REST-додатків, крок за кроком.

Завдання

1. Розробити опис методів API.
2. Підготувати опис формату даних у методах.
3. Підготувати приклади клієнтських запитів у системі curl або аналогічних.
4. Розробити код програми хмарного додатку.
5. Виконати тестування запитів API для роботи з хмарним додатком.

Зміст звіту

1. Мета роботи.
2. Завдання роботи.
3. Оформлення результатів роботи.
4. Опис методів API, опис прикладів запитів у системі curl або аналогічних.

5. Протокол тестування методів API хмарного додатку.

6. Висновки.

Контрольні питання

1. Якими вимогами має керуватися розробник для побудови REST API ?

2. Які переваги має формат JSON? Які є недоліки?

3. У чому перевага побудови API на базі HTTP протоколу?