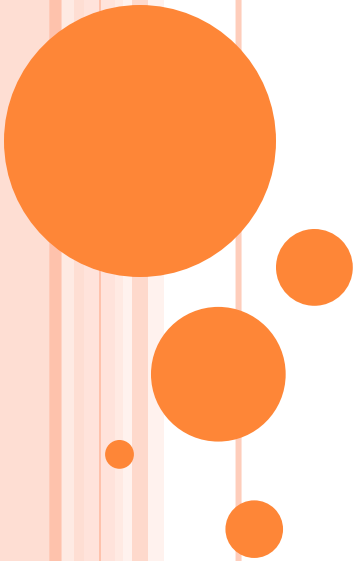


# Лекція 4. Getting Started

**Глибовець А.М.**

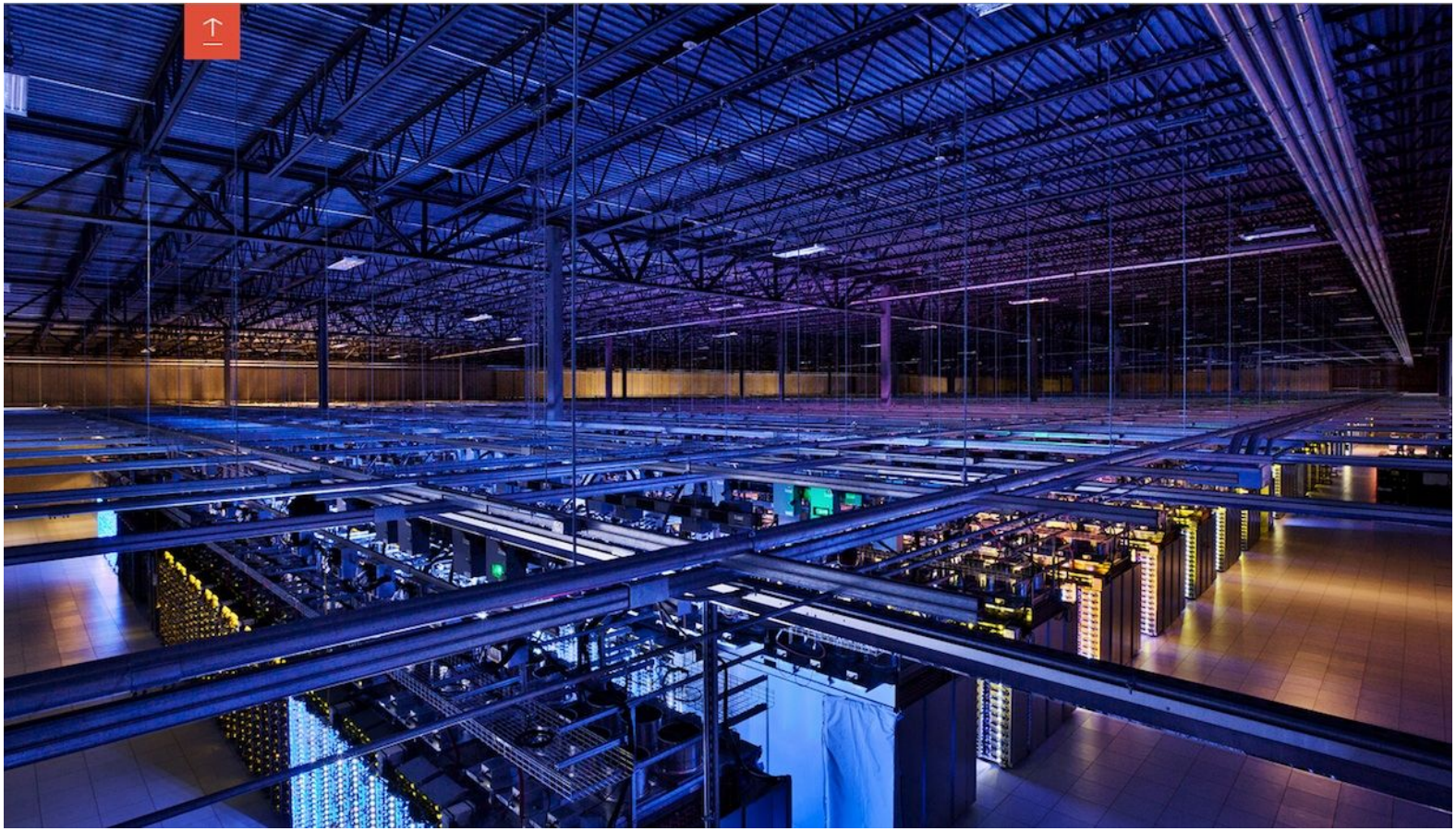
# Вступ

- Сьогодні ми згадаємо основи роботи з Google App Engine
  - подивимося на Google App Engine development site
  - створимо проект
  - подивимося на Cloud Endpoints
  - напишемо код для App Engine

# App Engine Development Cycle

- Для роботи вам необхідно мати Google акаунт
- Як ви пам’ятаєте Google надає всі засоби для розробки, deploy і запуску в Cloud вашого застосування
- Основна задача яку вирішує Google для вас – масштабування вашого застосування

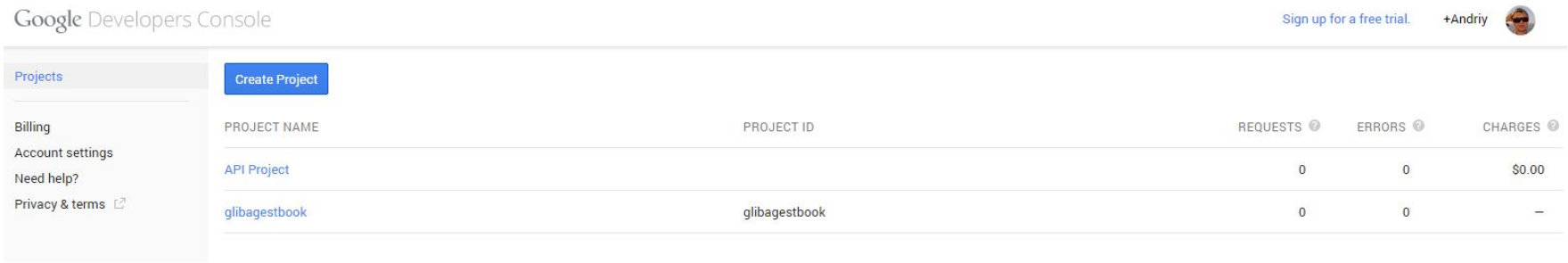
# Google Data Centers



http://blog.popflys.com/where-the-internet-lives/

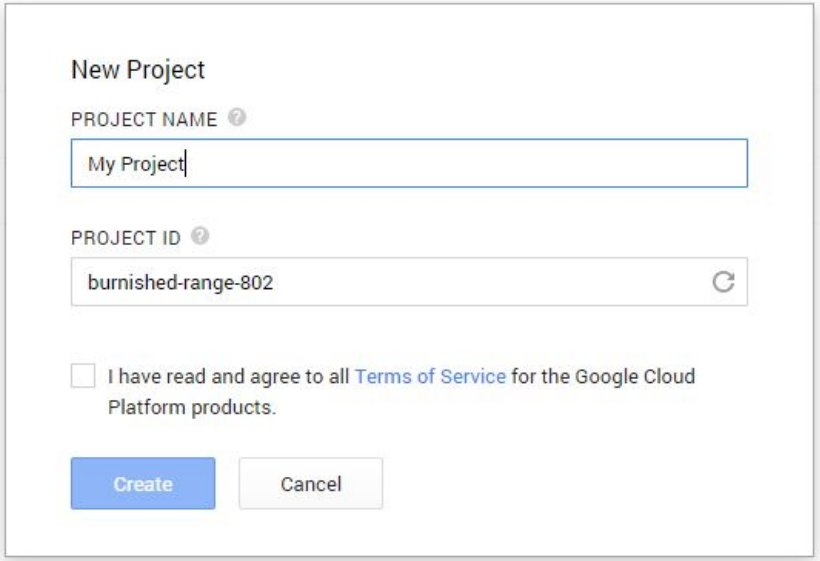
# New Project

- Для створення проекту ви можете скористатися консоллю
- https://console.developers.google.com

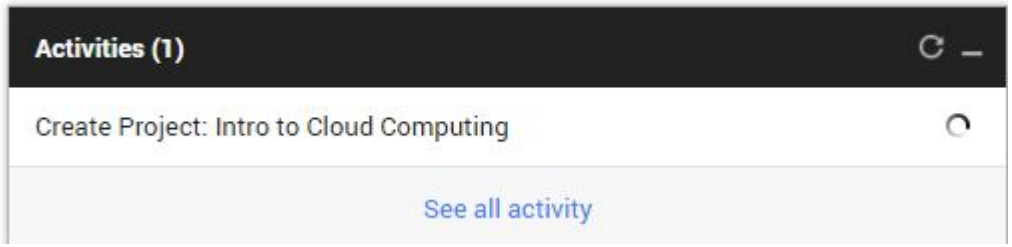


# New Project

- Створимо новий проект.
- Нам необхідно вказати
  - Project name – ім'я вашого проекту
  - Project ID – назва проекту, що буде використовуватися в URL
    - http://<ProjectID>.appspot.com



# New Project




Project ID: **glibovet-andrii-cloud1** Project Number: 399774280412

Estimated charges this month: $0.00

## Project Dashboard

**Take the Compute Engine quickstart**

Use NodeJS and MongoDB on Compute Engine to create a two-tier to-do list application. Duration: ~15 minutes.

Take the quickstart

**Boost your app with a Google API**

Use Google APIs with your app to harness the power of Google's services and technologies

Enable an API

**Launch click-to-deploy software**

Deploy popular open stacks on Google Compute Engine just by filling out a simple form

Try Click to Deploy

**Try BigQuery with population data**

Run queries against huge public data sets to see how BigQuery can help you analyze your own data

Try BigQuery

**Create a storage bucket**

Store your unstructured data safely and with high availability using Google Cloud Storage

Create bucket

# Підтримка різних пристроїв

- В нас з вами дуже багато пристроїв, що ми б хотіли підтримувати в якості клієнтів нашого застосування
  - назвіть мені ці клієнти

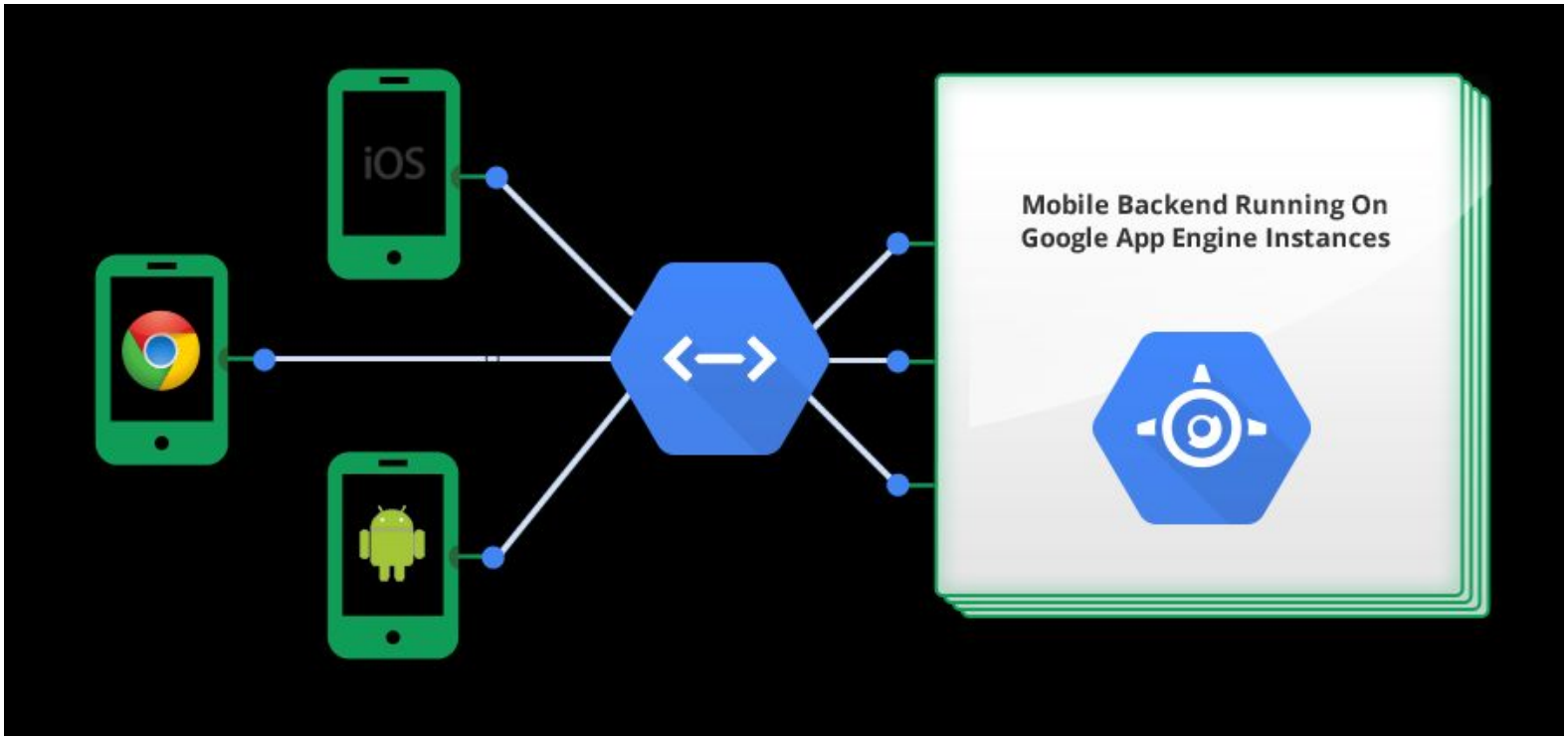
# Підтримка різних пристроїв

- Ми б хотіли написати наше застосування так, що б воно однаково могло працювати з усіма можливими пристроями.
  - Чи можна так зробити?
  - Як ви думаєте як це можна зробити?

# Підтримка різних пристроїв

- Google запровадив Cloud Endpoints для цих цілей
- Ви можете створити єдиний API в своєму застосуванні, а потім використовуючи Cloud Endpoints згенерувати API для різних пристроїв

# Google Cloud Endpoints




https://cloud.google.com/appengine/docs/java/endpoints/

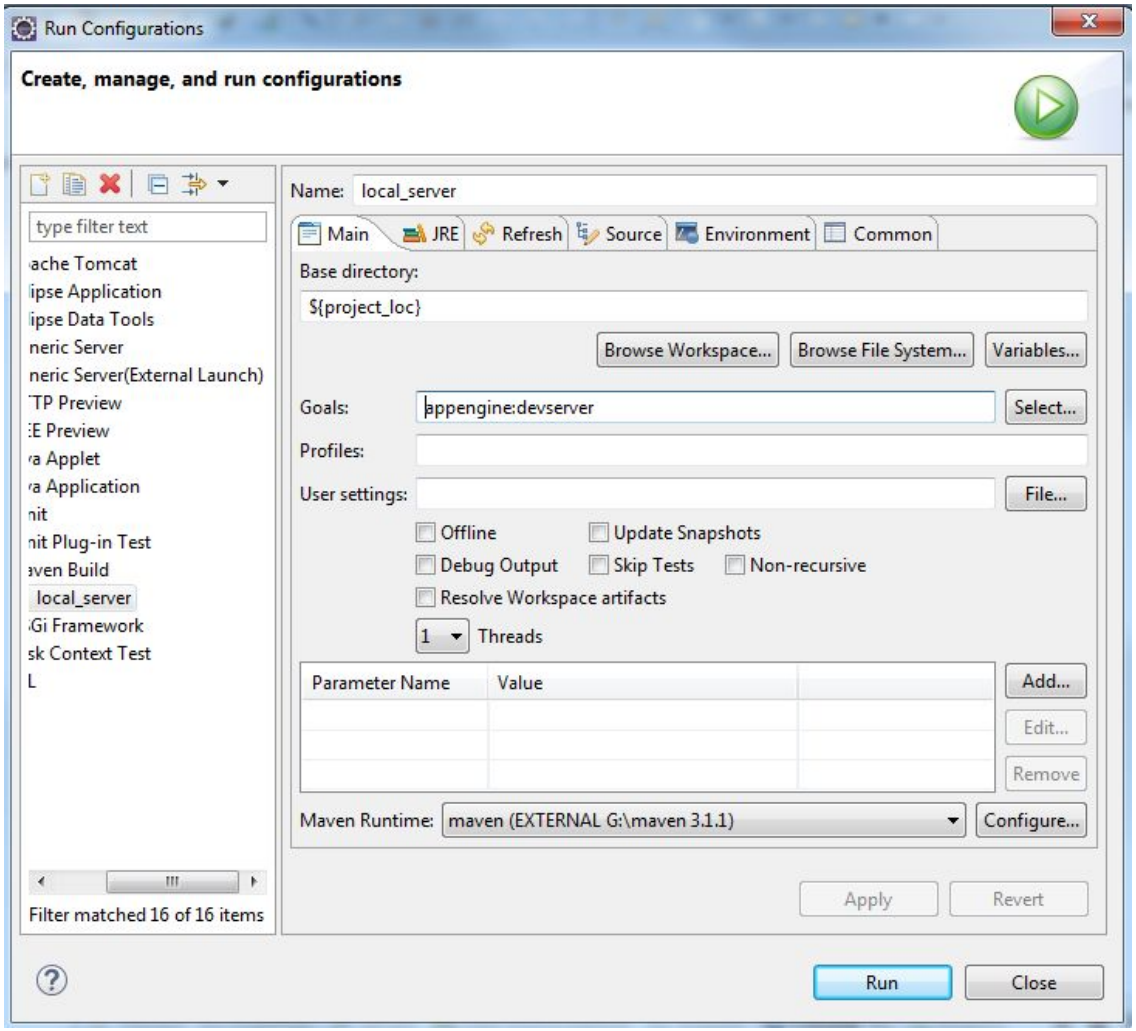
# Налаштування

- Ви маєте інсталювати:
  - Java JDK 1.7
  - Maven
  - Eclipse EE
  - **Не маєте встановлювати App Engine SDK**
  - Для цих цілей ми будемо використовувати Maven

# Перший проект

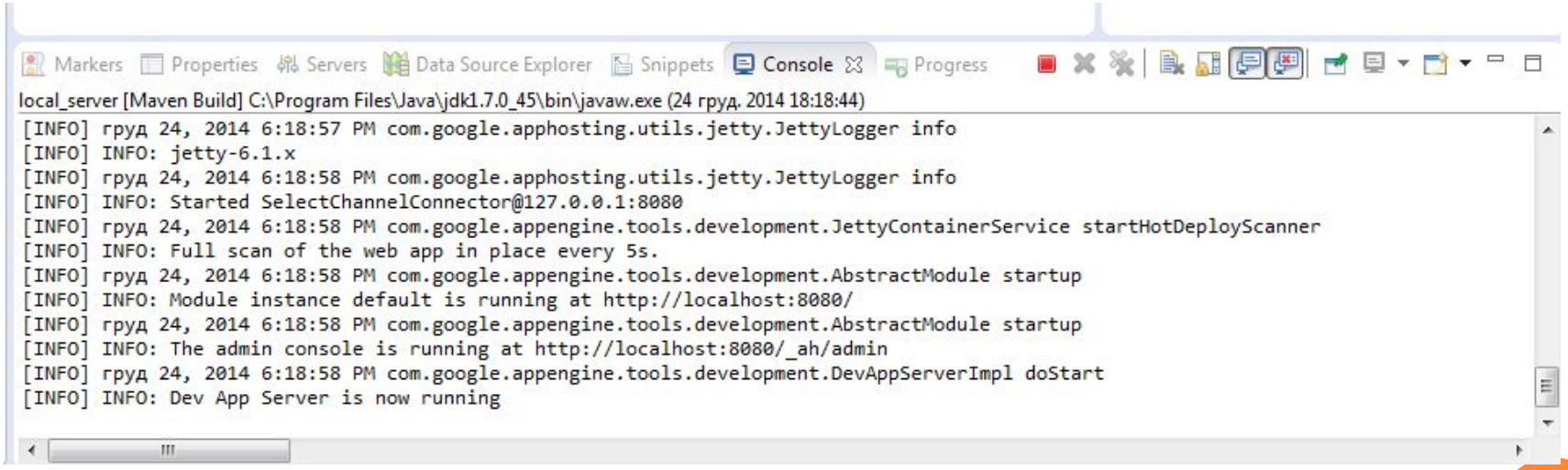
- Подивимося на простий проект
- Для того, що б імпортувати приклади до лекції в Eclipse ви маєте зробити наступну річ
  - Eclipse > File > Import
  - Existing Maven projects
- Для запуску:
  - Run As
  - Run Configurations
  - new launch configuration
  - Set the goal to appengine:devserver для запуску на локальному комп'ютері, або appengine:update для завантаження в App Engine
  - В Maven Runtime, ви маєте використовувати External реалізацію Maven, а не Embedded версію

# Запуск


Run Configurations
Create, manage, and run configurations
type filter text
ache Tomcat
ipse Application
ipse Data Tools
neric Server
neric Server(External Launch)
TP Preview
E Preview
a Applet
a Application
nit
nit Plug-in Test
aven Build
local_server
Gi Framework
sk Context Test
L
Filter matched 16 of 16 items
Name: local_server
Main
JRE
Refresh
Source
Environment
Common
Base directory:
${project_loc}
Browse Workspace...
Browse File System...
Variables...
Goals: appengine:devserver
Select...
Profiles:
User settings:
File...
Offline
Update Snapshots
Debug Output
Skip Tests
Non-recursive
Resolve Workspace artifacts
1
Threads
Parameter Name
Value
Add...
Edit...
Remove
Maven Runtime: maven (EXTERNAL G:\maven 3.1.1)
Configure...
Apply
Revert
Run
Close

# Запуск

- Перший запуск буде достатньо довгим
- Maven завантажить всі потрібні бібліотеки
- Після цього запуститься локальний сервер



# Запуск

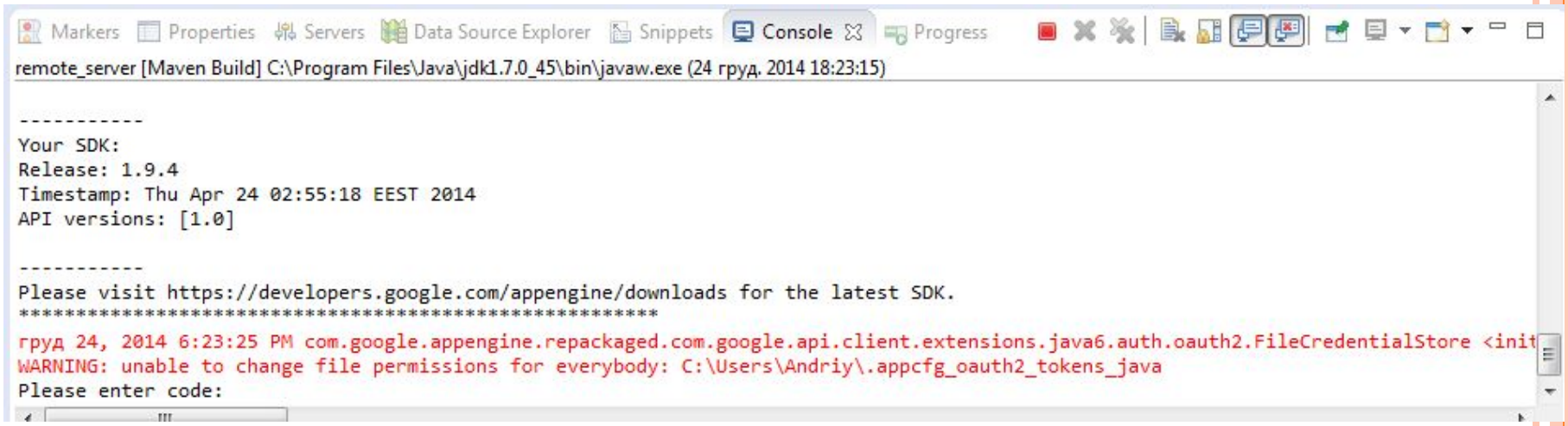
- Зайдіть і подивіться на наш результат

# Запуск

- Це просте застосування але з кодом ми розберемося пізніше
- Тепер давайте завантажимо наше застосування в Хмару
- Для цього створіть аналогічну конфігурацію
- Єдине замініть goals на appengine:update

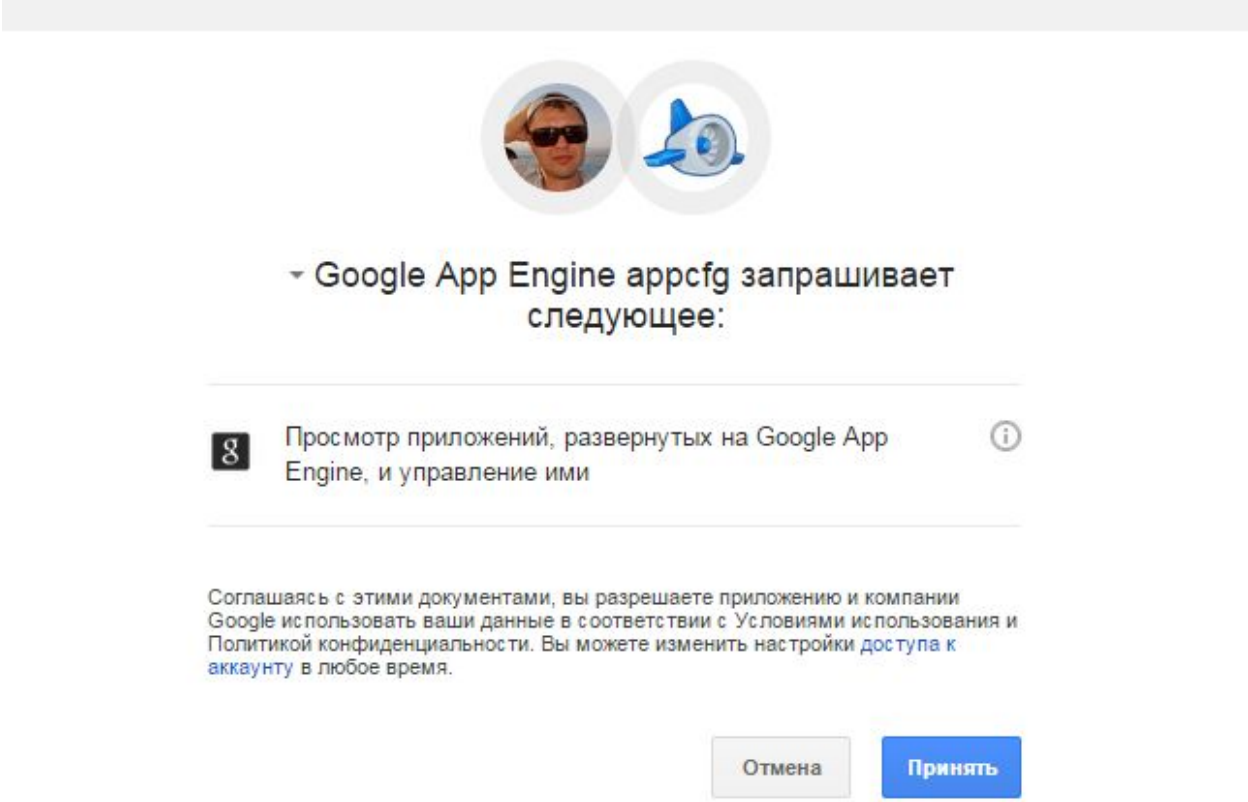
# Update в хмару

- Після першого запуску ви побачите в консолі

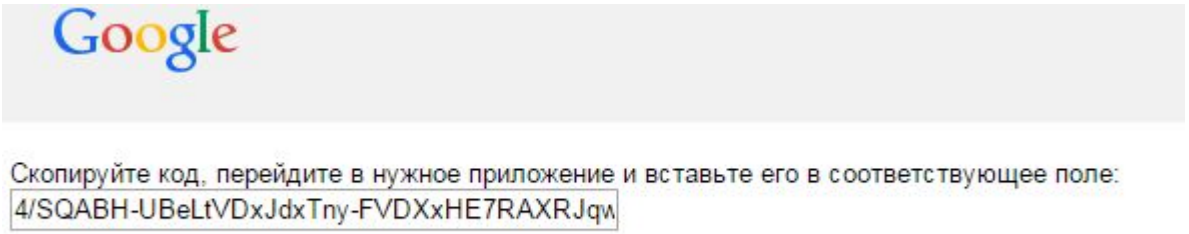


# Update в хмару

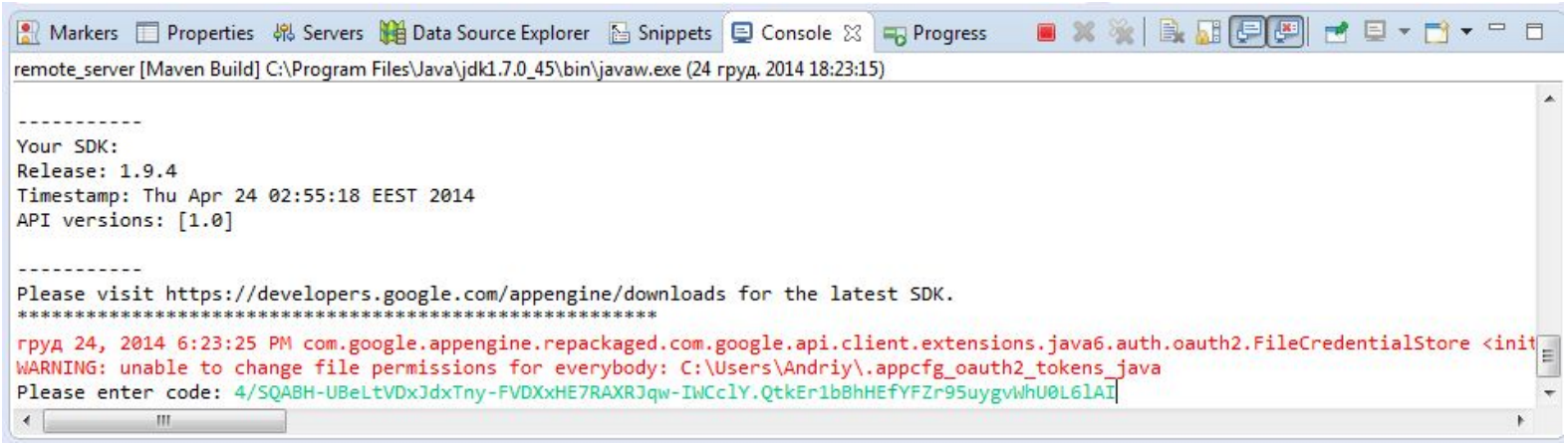
- В вас відкриється браузер



# Update в хмару

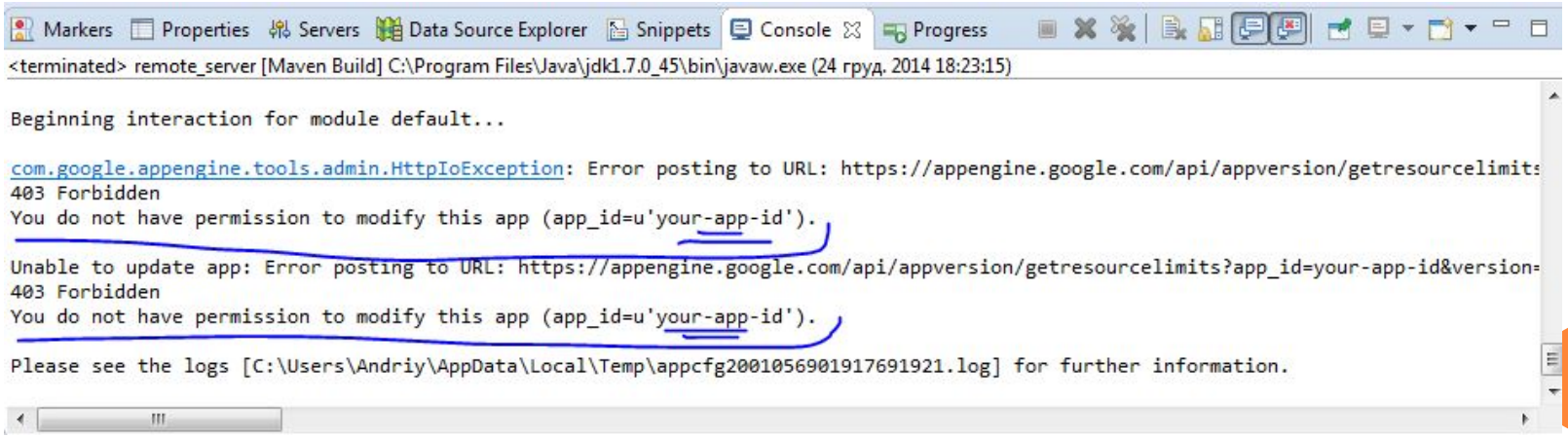


# Update в хмару



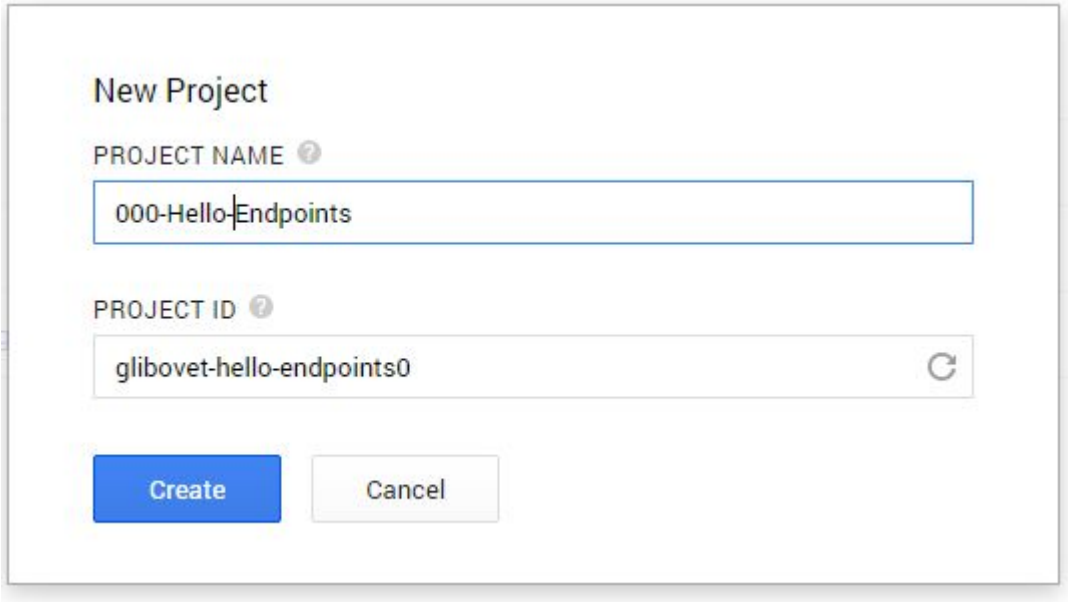
# Update в хмару

- Я отримав помилку
- Звісно, бо в мого застосування не такий id



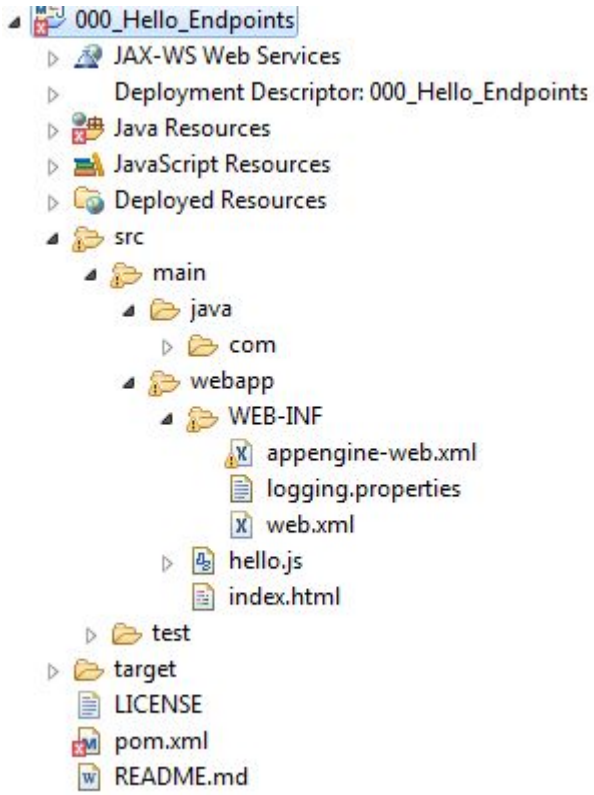
# Update в хмару

- Створю новий проект в App Engine

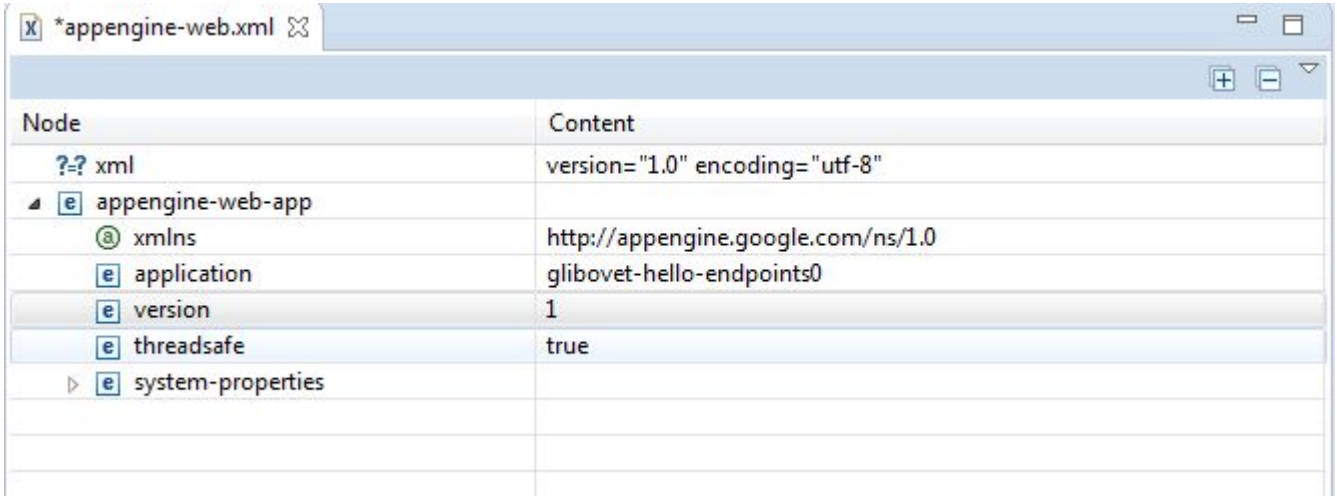


# Update в хмару

- Тепер виправимо файл appengine-web.xml

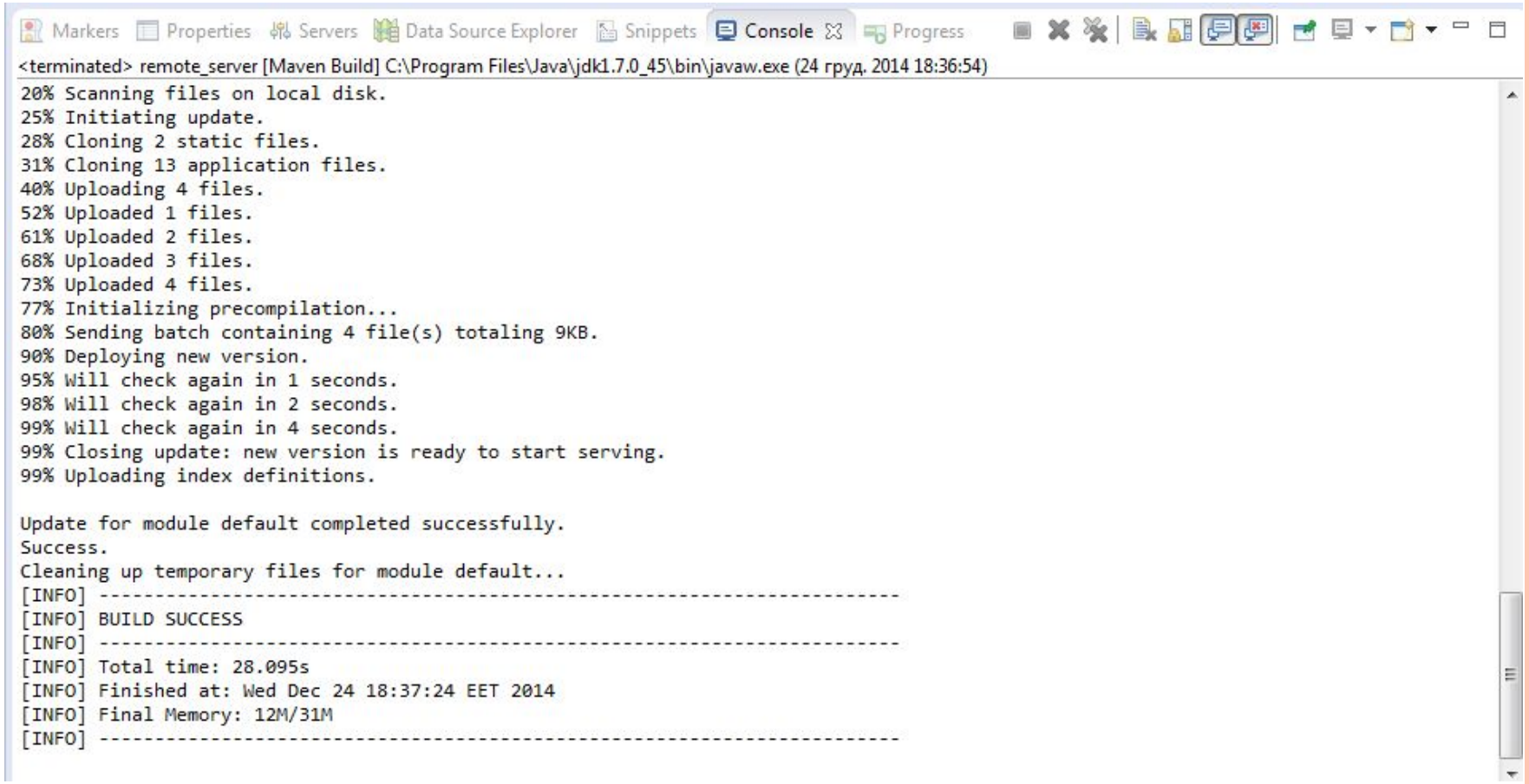


# Update в хмару



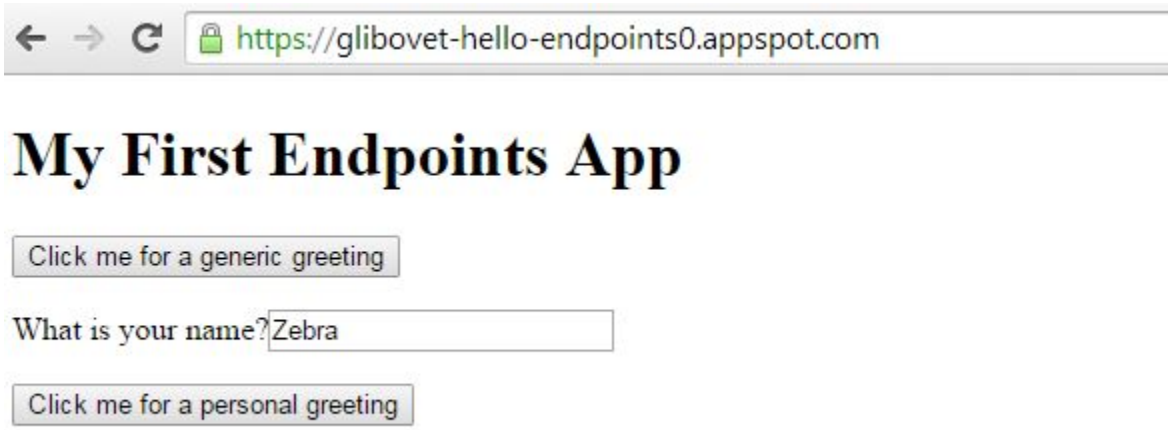
# Update в хмару

- Тепер знову запустимо наш проект на Update в хмару



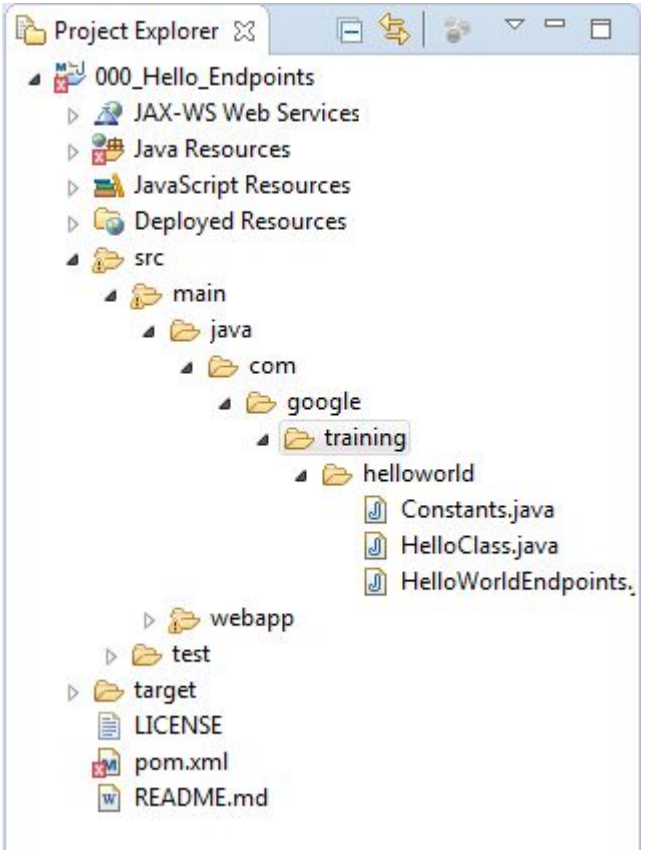
# Update в хмару

- Перевіримо:



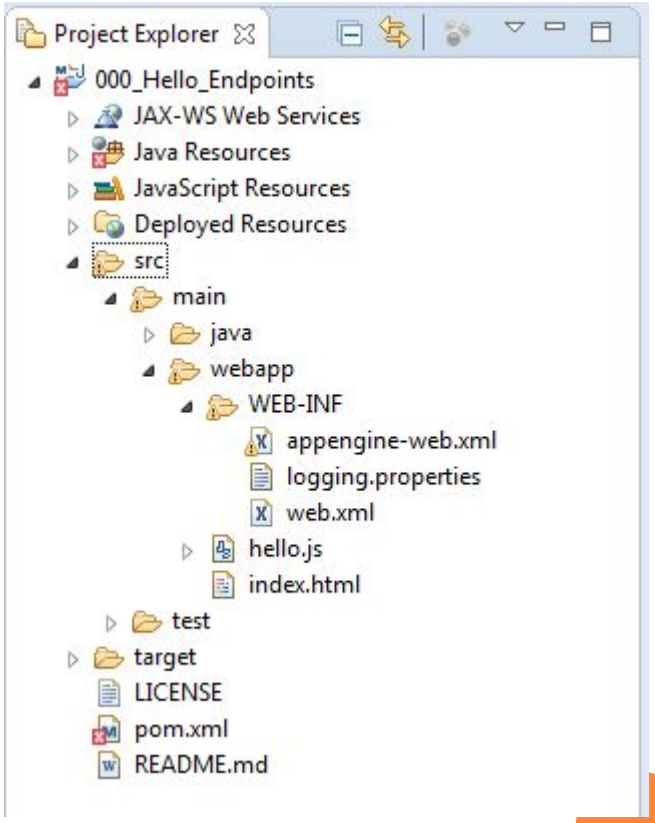
# Подивимося на проект

- Давайте подивимося на Endpoints
- Подивимося зміст цих трьох файлів



# Подивимося на проект

- Тепер подивимося на webapp
- Відкриємо web.xml



# Подивимося на проект

- Обов'язкова частина для використання Endpoints
- <servlet>
- <servlet-name>SystemServiceServlet</servlet-name>
- <servlet-class>com.google.api.server.spi.SystemServiceServlet</servlet-class>
- <init-param>
- <param-name>services</param-name>
- <param-value> com.google.training.helloworld.HelloWorldEndpoints </param-value>
- </init-param>
- </servlet>
- <servlet-mapping>
- <servlet-name>SystemServiceServlet</servlet-name>
- <url-pattern>/_ah/spi/*</url-pattern>
- </servlet-mapping>

# Подивимося на проект

- Налаштування безпеки, що говорить, що ми будемо використовувати https
- <security-constraint>
- <web-resource-collection>
- <web-resource-name>all</web-resource-name>
- <url-pattern>/*</url-pattern>
- </web-resource-collection>
- <user-data-constraint>
- <transport-guarantee>CONFIDENTIAL</transport-guarantee>
- </user-data-constraint>
- </security-constraint>

# Подивимося на проект

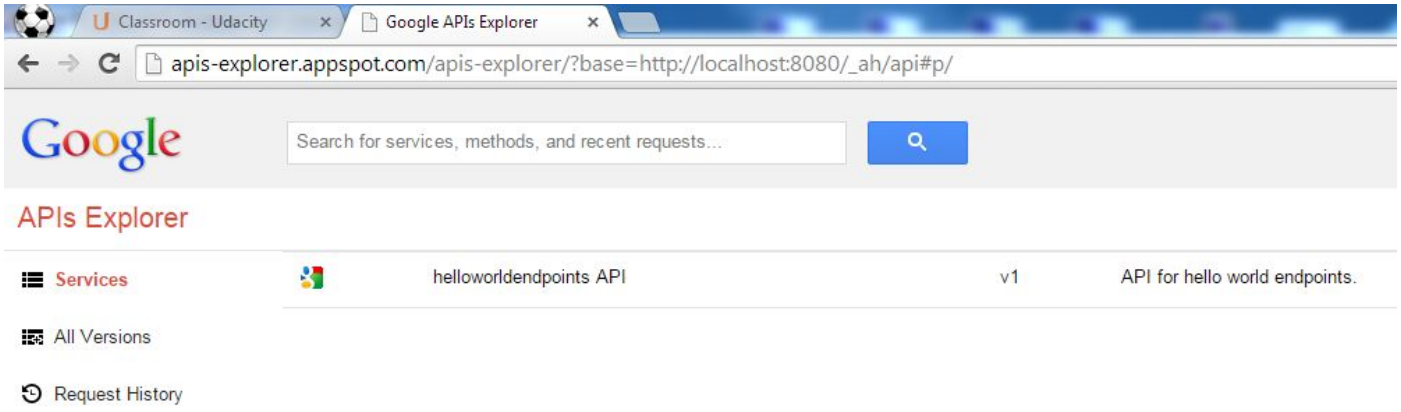
- Залишилося два файли
  - index.html
  - hello.js

# Endpoints

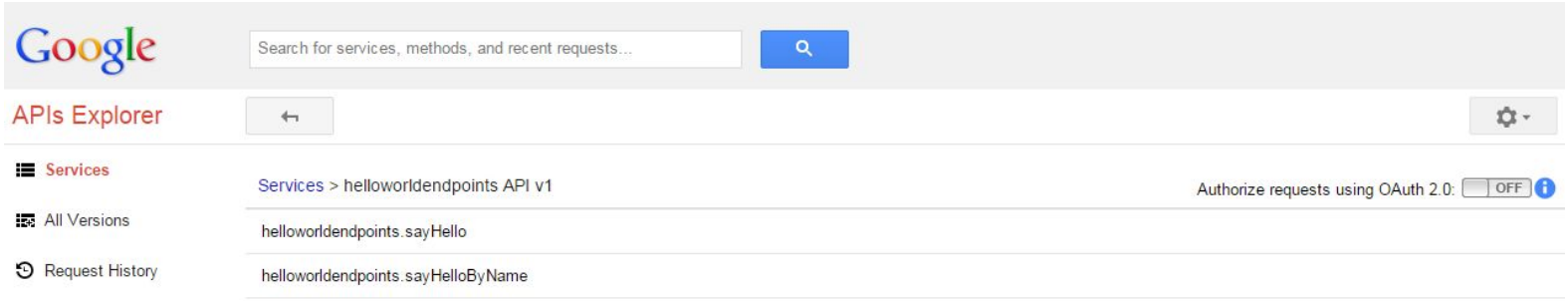
- Давайте докладно розглянемо файл HelloWorldEndpoints

# Використання

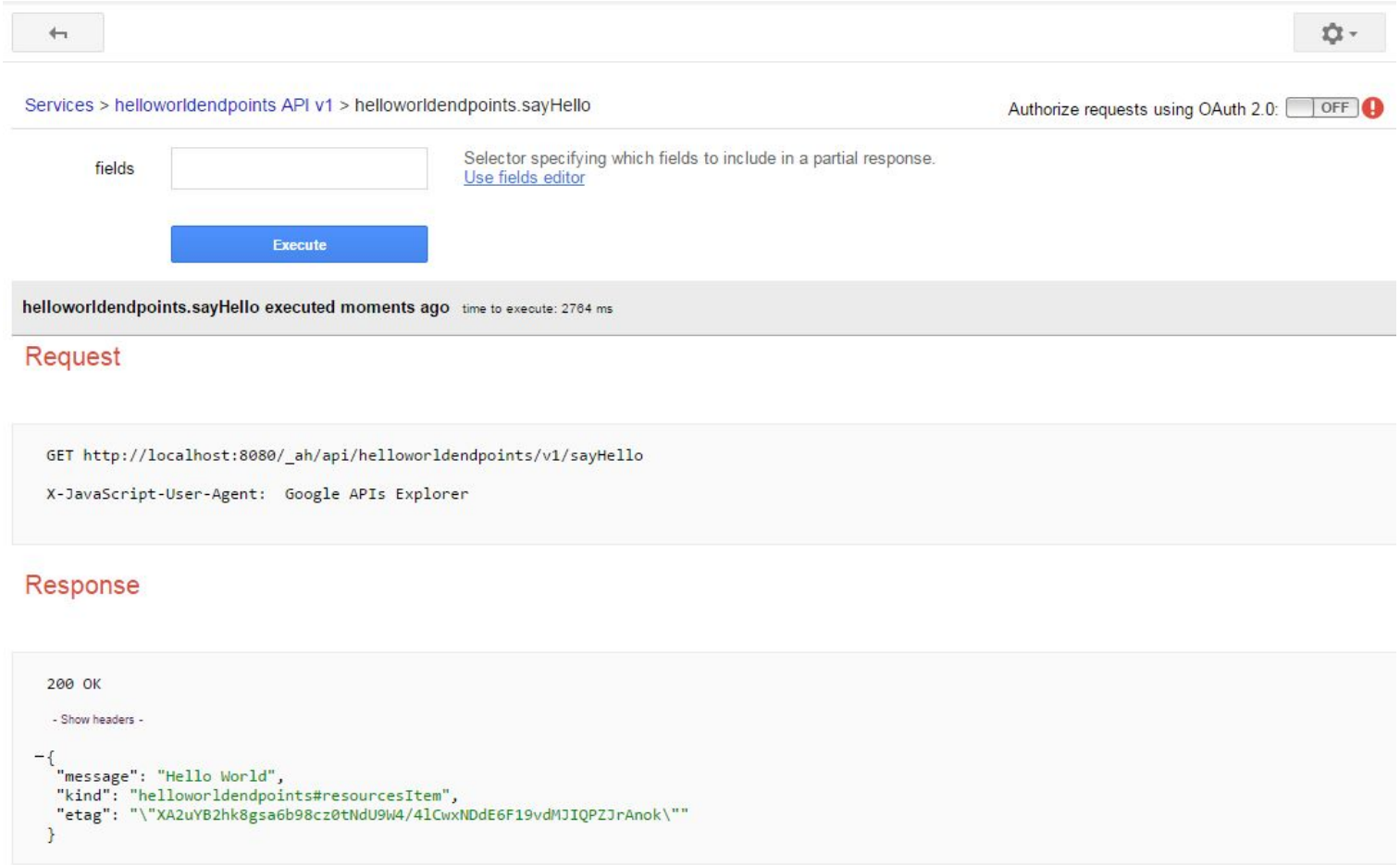
- Давайте запустимо наше застосування на локальному комп'ютері
- Ми запустимо панель управління api
  - http://localhost:8080/_ah/api/explorer



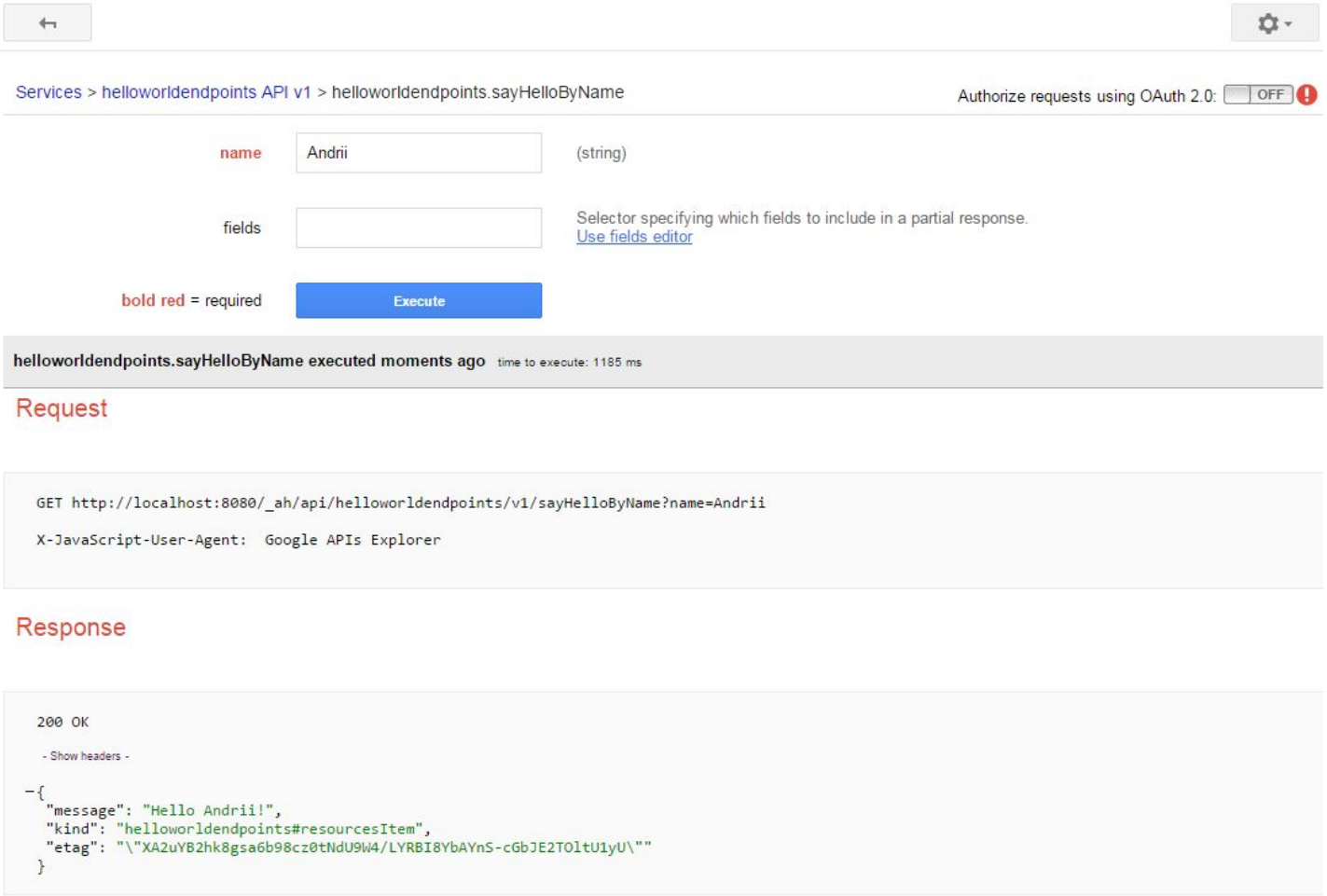
# Використання



# Використання



# Використання



# Використання

- Розберемося з фронтендом
  - index.html
  - Дуже важлива частина коду:
    - <script src="*hello.js*"></*script*>
    - <!-- Load the Google APIs Javascript client library -->
    - <!-- Then call the init function, which is defined in hello.js -->
    - <script src="*https://apis.google.com/js/client.js?onload=init*"></*script*>
- Весь інший код дуже простий
- Сама цікава частина:
  - onclick="will_be_set_after_endpoints_apis_loaded"

# Використання

- Давайте докладно розберемося з hello.js

# Нова функція

- Додамо нову функцію в Endpoints
  - greetByPeriod(name,period)
  - return HelloClass object
    - Good Morning Andrii!
    - Good Evning Andrii!

# Нова функція

- @ApiMethod(name ="greetByPeriod", path = "greetByPeriod", httpMethod = HttpMethod.***GET)***
- **public HelloClass greetByPeriod(@Named("name") String name,** @Named("period") String period){
  - **return new HelloClass(name,period);**
- }

# Нова функція

- Додамо нове поле в index.html та функцію в js
  - нам потрібне нове поле для вводу і нова кнопка

# Нова функція

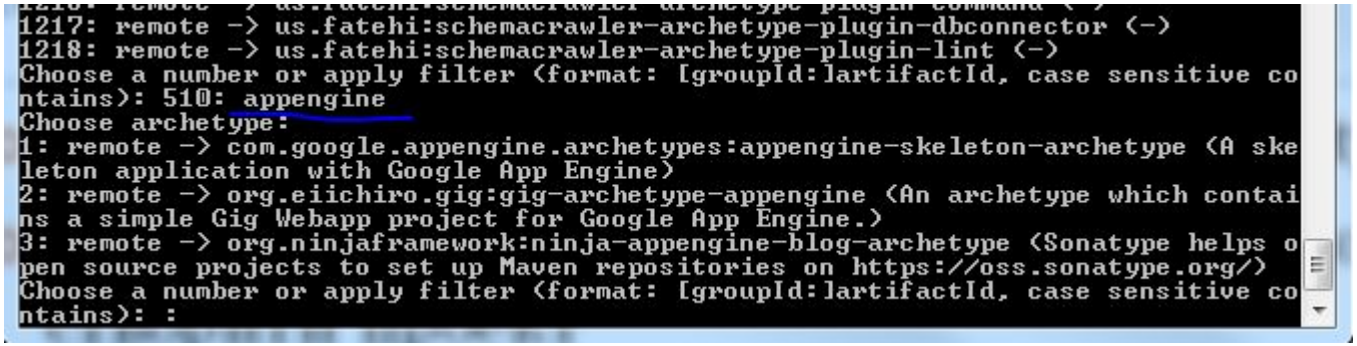
- **function greetByPeriod(){**
  - **var name = document.getElementById("name_field").value;**
  - **var period = document.getElementById("period_field").value;**
  - **var request = gapi.client.helloworldendpoints.greetByPeriod({'name': name, 'period':period});**
  - request.execute(sayHelloCallback);
- }

# Maven

- Ми з вами починали з проекту, що хтось створив для нас
- Що робити якщо такого проекту в нас немає?

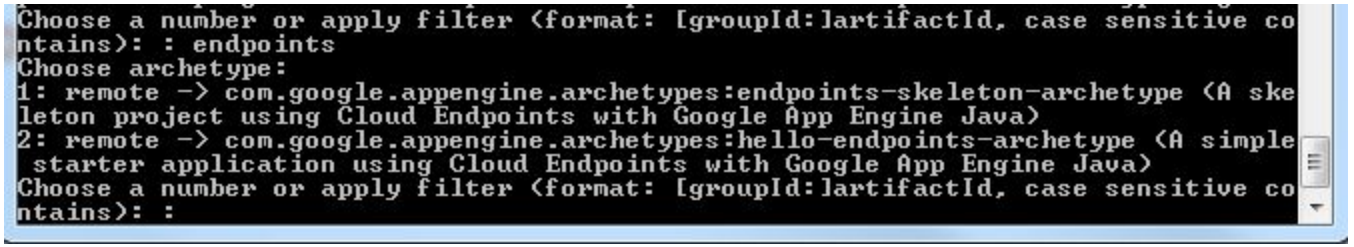
# Maven

- Ми можемо використати Maven і archetypes
- Відкрийте консоль в директорії де ви хочете створити проект
  - mvn archetype:generate
- Перший раз Maven багато чого завантажить
- Після цього вам буде запропоновано обрати архетип проекту
- Їх багато, тому ви можете відфільтрувати їх
- наберіть appengine



# Maven

- Введіть endpoints



# Maven

- Ми використаємо com.google.appengine.archetypes:hello-endpoints-archetype
- Таким чином ми обираємо номер 2
- Після цього нам пропонують версію, ми оберемо останню – 3
- Введемо groupId
  - ua.com.glybovets
- artifactId
- і далі по пунктам
- Тепер ми можемо імпортувати даний проект в Eclipse

# Google API

- Точно такий же Api використовує і Google для совїх сервісів
- Google Apis Explorer
- https://developers.google.com/apis-explorer/#p/
- Давайте знайдемо переклад і подивимося на його API

- Дякую за увагу