Java Joke API with Eclipse Tutorial

Contents

Time required: 60 minutes

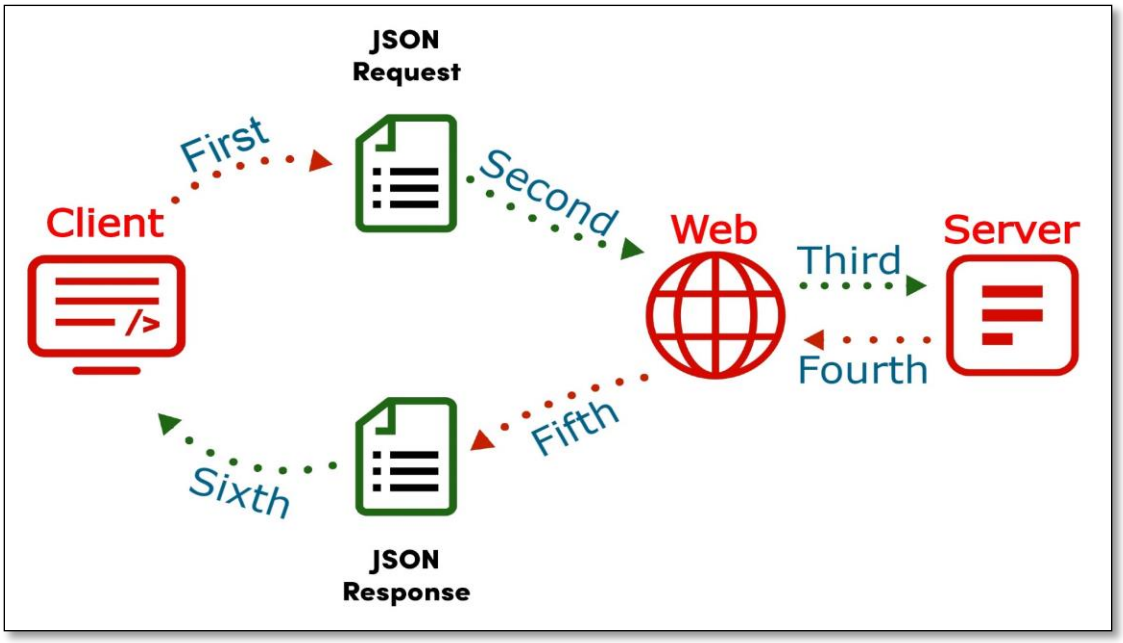
- Comment each line of code as shown in the tutorials and other code examples.
- Follow all directions carefully and accurately.
- Think of the directions as minimum requirements.
- Please read all the directions before beginning the assignment.

What is an API?

What is an API? This is what Wikipedia says.

"An **application programming interface** ('API') is a computing interface that defines interactions between multiple software intermediaries. It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees. An API can be entirely custom, specific to a component, or designed based on an industry-standard to ensure interoperability. Through information hiding, APIs enable modular programming, allowing users to use the interface independently of the implementation."

An API is a software intermediary that allows two applications to talk to each other. In other words, an API is the messenger that delivers your request to the provider that you're requesting it from and then delivers the response back to you.



Web API's are hosted on webservers. Instead of a web browser asking for a webpage (like how most interactions with the internet is done today) your program asks for data. This data is usually returned to your program in a JSON format. JSON stands for JavaScript Object Notation and is a lightweight text-based data-interchange format.

To get data to our program we make a request to a web server. The webserver then replies with our desired data and a status code. Sometimes an API key is needed to be sent along with the request to successfully return data.

There are many different ways to call information from an API. One of the most common are | **get** | requests. You use a get request each time you browse to a web site.

For example, to request data to a Python script using the extra functionality gained from the request library the statement below would be used.

```
response = requests.get('http://api_example.com/example.json')
```

This data object can be named anything, response was used as the identifier for this example. This API doesn't exist. This is the format they tend to take.

Data from the web can tell us the exact weather situation of any place on the globe, the location of faces on an image we provide or the number of astronauts currently floating far, far out in space. Any information freely accessible on the World Wide Web is literally within our grasp from within our programs.

Why Use a Web API?

You could use static data which you have downloaded or created yourself. Data used this way with Java would work perfectly fine. Here are some examples of when APIs are more useful than static data sets.

- When the data is changing quickly, and you need the data to be accurate and up to date. Planes are currently in the air or the location of the international space station, the data is constantly changing.
- When only a small piece of a much larger data set is desired.
- When you are taking advantage of already completed, repeated computations. Sources such Facebook, Spotify and Google already have a ton of data, which have undergone serious calculations, which you can access easily with an API request.

In cases like the ones above, a Web API is the right solution. Using an API will save time and effort over doing all the computation ourselves or constantly refreshing our desired data.

Status Codes

Status codes are shorthand explanations of what happened when you requested for data and are very useful when it comes to problem solving. After requesting an API there are a number of situations that can occur. Each situation has a three digit numerical status code associated with it. The web server will return status codes to your computing device every single time they receive an API request. This allows for robust error handling.

To see the status code of your request type this line into your code where the placeholder text | response | is the user-defined variable name decided by you in your script line when data from the API is requested.

```
print(response.status_code)
```

All the potential responses that could be printed to the Python IDLE Shell can be seen below. An example of asking for the status code in script can be seen in the Astronaut in Outer Space Script.

- 200 -

This means that everything worked, the resource was requested, and the desired data was returned successfully. It will most likely be returned to as a JSON file. JSON file (longhand is JavaScript Object Notation format) is a lightweight method to store simple data structures and objects. It is primarily used for data-interchange between a web application and a server. This is because when exchanging data between a browser and a server, the data can only be text.

This is the code that is most often used to verify the API request. If any other code is returned, the request is considered in error.

- 301 -

This means, after the resource was requested, the server redirected your request to a different endpoint. This can happen when an organization switches domain names or an endpoint name is changed. The data may or may not get returned successfully.

- 400 -

This means, after the resource was requested, the server stated you made a bad request. When requesting something you must send information to gain access to the particular data you desire. This status code appears when you do not send along the right data and you would not have the data you desire.

- 401 -

This means, after the resource was requested, the server stated that you are not authenticated. When requesting something you must send authentication information to gain access to the particular data you desire. This status code appears when you do not send the right credentials to access an API. In this case you would not have the data you desire.

- 403 -

This means, after the resource was requested, the server stated you do not have the right permissions to see it. Some data online is locked meaning access to them is forbidden. In this case you would not have the data you desire.

- 404 -

This means, after the resource was requested, the server stated that the resource you attempted to access was not found on the server. Perhaps the data has been moved, deleted or was not there to begin with. In this case you would not have the data you desire.

Public API's

There are hundreds of public API's available. This website contains a list of some of them.

https://github.com/public-apis/public-apis

Official Joke Web API

Who doesn't need a good joke to brighten their day? This is a simple API that supplies a variety of random jokes.

We are going to create a Java program to get a joke and display it.

Before we do that, we have a few new concepts to go through.

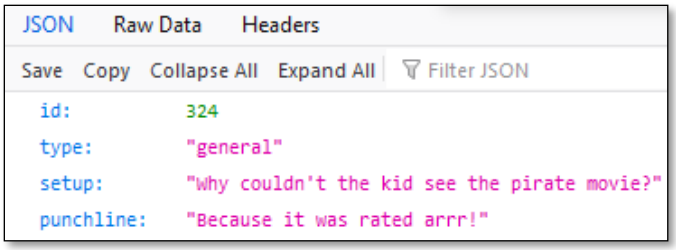
What is JSON?

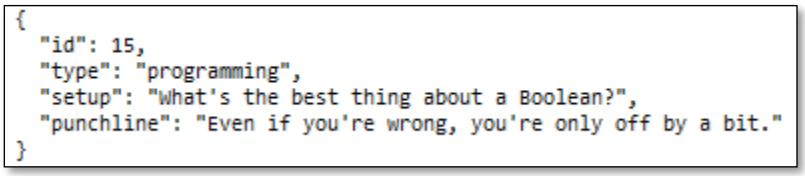
The Official Joke API is a free https-based joke Web API which provides a JSON response.

Raw API data is typically in a text file in JSON or XML format. For our API projects, we will use JSON.

JavaScript Object Notation (JSON) is an open standard text-based file and data interchange format. It uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types.

Go to https://official-joke-api.appspot.com/jokes/random to see some raw JSON data. This view is using Firefox.





This is the raw JSON file seen using Google Chrome.

{"id":79,"type":"general","setup":"What time is it?","punchline":"I don't know... it keeps changing."}

Download and Install Eclipse

Eclipse is an open source IDE for Java and other languages. It includes the capability to drag and drop components for Java GUI program much like Visual Studio does for C#.

1. Go to https://www.eclipse.org/downloads/
2. Download the 64-bit installer.
3. Double click the downloaded executable, **eclipse-inst-jre-win64.exe**
4. The **Eclipse Installer** starts up. Click **Eclipse IDE for Enterprise Java Developers**.
5. Click **Install**. Accept any license agreements.
6. Click **Launch** to launch Eclipse.
7. Choose a directory as a workspace, c:\eclipse-workspace for example. Make sure this is a directory you can locate later. All of your Eclipse projects will be stored there.

Create Application

1. In **Eclipse →** close the Welcome window.
2. Click **File → New → Project**. Go to **Java → Java Project**. Click **Next**.
3. Project Name: **OfficialJokeAPI**. Click **Next**.

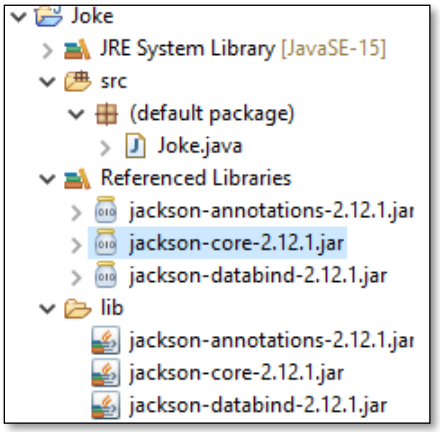
4. Java Settings: Uncheck **Create module-info.java file** → Click **Finish.**
5. Open the **OfficialJokeAPI** project:
6. Right click the **src** folder, **New** → **Class**.
7. Name: **JokeApp**
8. Which method stubs would you like to create? **public static void main(String[] args)** Click **Finish**. The file opens automatically in **Source** View.

Jackson JSON Libraries

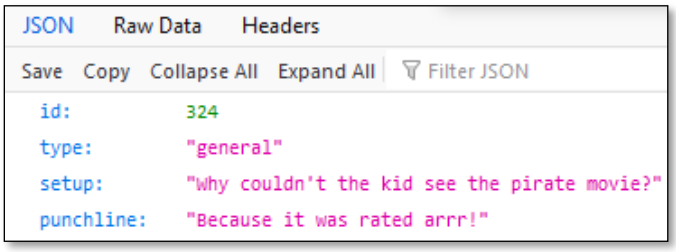
We need to add three libraries to parse out the JSON data into Java classes.

We will be retrieving data from the web in JSON format. Java doesn't have any native JSON parsing. We are going to add three libraries to parse out the JSON data into Java classes.

1. Download these three files.
    a. jackson-annotations-2.16.1.jar
    b. jackson-core-2.16.1.jar
    c. jackson-databind-2.16.1.jar
2. In the Joke project in Eclipse: Right Click **Joke → New → Folder**
3. Name the new folder: **lib**
4. Copy and paste the three jar files into this folder in Eclipse.
5. Right Click each jar file **→ Build Path → Add to Build Path**
6. Your project folder should look like this.



JSON to POJO



Go to https://official-joke-api.appspot.com/jokes/random You should see the above JSON file with different data.

To convert the JSON data into a Java class, we need to create a class modeled after this data. You can see the first field is an integer, the rest are Strings.

Here is a website that will do this for you.

1. Go to https://codebeautify.org/json-to-java-converter
2. Click **Load Url**: Paste the Url of the JSON to be converted. Click **Load**.
3. The **Result** will be a class with the fields, setters, and getters.
4. Click the **Copy** button.
5. In the Joke project: Create a new Java public class named: **JokeClass**
6. Paste the code into this file. Change the class name to **JokeClass**.
7. Change the id field, getters, and setters data type from double to int.
8. Create a void **displayJoke()** method that displays the setup and punchline to the console as shown below.

```
JokeClass object
id = 86
type = general
setup = Did you hear about the bread factory burning down?
punchline = They say the business is toast.
```

9. Override the Object **toString()** method to display the object information as shown below.

```
Setup: Did you hear about the bread factory burning down?
Punchline: They say the business is toast.
```

Joke Class

Add the following import lines to the top of the **Joke.java** file.

```
/**
 * Java API to JSON to POJO
 * Filename: Joke.java
 * Written by:  William Loring
 * Written on:  02/14/2021
 * Revised:
 */

import java.io.IOException;
import java.net.URL;

// Import Jackson JSON libraries
import com.fasterxml.jackson.core.JsonGenerationException;
import com.fasterxml.jackson.databind.JsonMappingException;
import com.fasterxml.jackson.databind.ObjectMapper;

```

This imports the necessary java and Jackson JSON libraries needed for this project.

```java
/**
 * Java API to JSON to POJO
 * Filename: OfficialJokeAPI.java
 * Written by:  William Loring
 * Written on:  02/14/2021
 * Revised:
 */

import java.io.IOException;
// Simple Java library to retrieve http information from a URL
import java.net.URL;

// Import Jackson JSON libraries
import com.fasterxml.jackson.core.JsonGenerationException;
import com.fasterxml.jackson.databind.JsonMappingException;
import com.fasterxml.jackson.databind.ObjectMapper;

public class JokeApp {

    public static void main(String[] args) {
        try {
            // Create Jackson JSON object to read the raw JSON data
            // into POJO (Plain Old Java Objects)
            ObjectMapper objectMapper = new ObjectMapper();

            // URL for JSON source
            String url = "https://official-joke-api.appspot.com/jokes/random";

            // Read JSON source into JokeClass Object
            JokeClass jokeClass =
                    objectMapper.readValue(new URL(url), JokeClass.class);

            // Print out the jokeClass object
            System.out.println(jokeClass);

            // Display a joke at the console
            System.out.println(jokeClass.jokeToString());

        // e.printStackTrace() will print the exception
        // JSON Exception handling
        } catch (JsonGenerationException e) {
            e.printStackTrace();
        } catch (JsonMappingException e) {
            e.printStackTrace();
        // IO Exception handling
        } catch (IOException e) {
            e.printStackTrace();
        // Handle any other exceptions
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

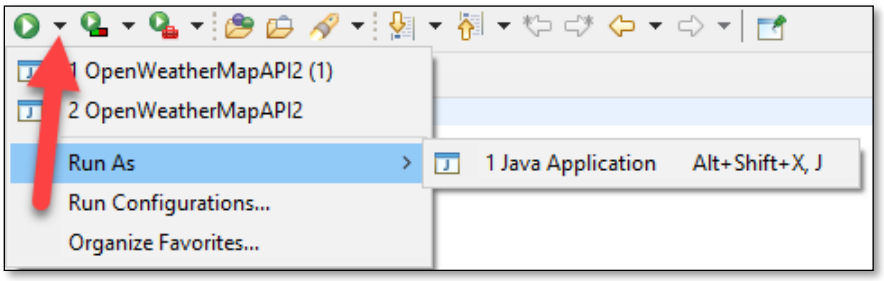
The above code does 3 things.

1. Creates a Jackson object to read JSON into POJO (Plain Old Java Objects).
2. Reads the https JSON into a **JokeClass** object.
3. Displays a joke.

Time for a Joke

Time to run your new application.

1. Click the project folder.
2. Click downward pointing arrow next to the green run button.
3. Click **Run As → Java Application**.



Example run as shown in the console at the bottom of Eclipse:

```
JokeClass object
id = 86
type = general
setup = Did you hear about the bread factory burning down?
punchline = They say the business is toast.

Setup: Did you hear about the bread factory burning down?
Punchline: They say the business is toast.
```

Assignment Submission

Zip up the Joke folder under your Eclipse workspace.

Attach the zip file to the assignment in Blackboard.