Chris Hardcastle

Technical Test Submission (part 2 of 2)

This work was prepared and written by Chris Hardcastle in response to the brief below. The work was briefed as a technical test.

Brief

This work is the result of my answer to the technical test as briefed below:

Using the Google Map Geocoding API (documentation here: https://developers.google.com/maps/documentation/geocoding/start), write some code, that will be able to, as a minimum:

a) Accept an address as a command line argument. b) Call the API with the given address. c) Get the response. d) Parse the response, and extract the: 1. full postal code, if it exists, or partial postal code, if it exists 2. The place id. 3. The latitude and longitude of the given address. e) Append the extracted data to a CSV file.

Some pointers:

1. Although the developer docs say you need a key, you should be able to call the API without needing a key, for example, if you open this link in your browser, it should work:

https://maps.googleapis.com/maps/api/geocode/json?address=8+Cambridge+Road,Hove

If you are unable to query it without a key, then feel free to code as if someone should provide their key to the program.

2. Your code should be well documented.

3. Your code should demonstrate good coding principles.

4. You are free to code it however you wish, but we would like some notes explaining any decisions you made e.g. structure of your code, why you chose to use a particular library, design pattern, and so on.

Assumptions

A number of assumptions were made on the brief:

- The script should handel cases where no results were found.
- The script could terminate with a small message as a response.
- The script doesn’t need to terminate with JSON or behave in a RESTful way.
- No frontend is required for this work.
- Some test driven development was allowable.
- Where appropriate, test data didn’t have to be accurate.
- Empty search results are not added to the CSV file.
- It was not necessary to maintain PSR-2 code compliance.

Installation

1. Download or clone the work from https://github.com/chardcastle/address-lookup.git
2. Navigate to the project directory
3. Install composer and run composer install to import the necessary dependancies.
4. Ensure write permissions to the data folder.

Usage

Open your command line editor or terminal.

1. Search for an unknown address

> *php api.php "xxxxxxx"*

2. Search for a valid address

> *php api.php "10 downing street, London"*

Any other uses should invoke the default (help) response.

In the event of a successfully matched address (2), the following information is appended to a CSV file located `data/results.csv`:

- PostCode
- PlaceId

- Latitude
- Longitude

The test (csv) file is the result of automated testing and may not event exist unless the tests have been run.

You’re welcome to delete the csv files at any point, they will make a fresh return the next time the script runs.

Test instructions

Install and run composer in the same document root directory, this provides everything required to run:

> *phpunit tests/*

At this present time of writing: 3 tests and 7 assertions (all) passed in 473 milliseconds, using 15.75MB of memory.

Any feedback is gratefully received. Please visit www.chrishardcastle.co.uk for more information.

Thanks for your interest in my work and taking a look at technical test result.

The first part of the technical test has already been submitted.
This is the second half which completes the test.
Thanks for reviewing my answers | Chris Hardcastle |