Exponential Back Off Application

Execution, Setup and Deployment Instructions

stickermule

Author: Denis Bell
Email: denisdbell@gmail.com

Overview

This is a ruby based application which implements the exponential back off algorithm. Exponential back off is a standard error handling strategy for network applications in which a client periodically retries a failed request with increasing delays between requests.

Executing The Application

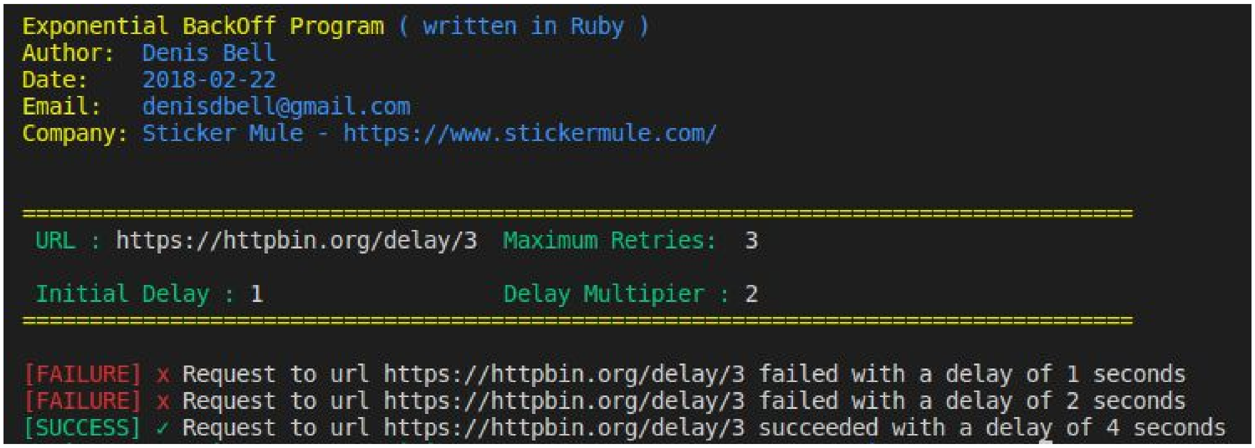
The instructions below describes the various methods which can be used to execute the application:

Run using the “docker run” command

To execute the application using docker run, type the following command in your terminal:

```
docker run -t denisdbell/backoff:1.0
```

You should see the following output displayed to stdout:



```
Exponential BackOff Program ( written in Ruby )
Author:  Denis Bell
Date:    2018-02-22
Email:   denisdbell@gmail.com
Company: Sticker Mule - https://www.stickermule.com/


===================================================================================
 URL : https://httpbin.org/delay/3  Maximum Retries:  3

 Initial Delay : 1                  Delay Multipier : 2
===================================================================================

[FAILURE] x Request to url https://httpbin.org/delay/3 failed with a delay of 1 seconds
[FAILURE] x Request to url https://httpbin.org/delay/3 failed with a delay of 2 seconds
[SUCCESS] ✓ Request to url https://httpbin.org/delay/3 succeeded with a delay of 4 seconds
```

As shown above, the application is executed with default values for the following variables:

1. **URL** - This is the url which will be requested by the application. The default value is **https://httpbin.org/delay/3**.

2. **Maximum Retries** - This variable represents the amount of times the specified URL will be requested. The default value is **3**.

3. **Initial Delay** - This is the delay in seconds that will be used to make the initial request to the specified url. The default value is **1** second.

4. **Delay Multiplier** - This variable is used to exponentially increase the delay value each time a failed request is made. The default value is **2**.

The value of the above variables can be easily changed by passing new values to the **docker run** command, see an example below:

```
docker run -t denisdbell/backoff:1.0
> https://httpbin.org/delay/5 \ #URL
> 4 \ #Maximum Retries
> 2 \ #Initial Delay
> 3 \ #Delay Multiplier
```

Individual parameters can also be set. In the following example only the **url** is set:

```
docker run -t denisdbell/backoff:1.0
https://httpbin.org/delay/3
```

Note: Parameters which are not set will use their default values.

Run using “docker-compose”

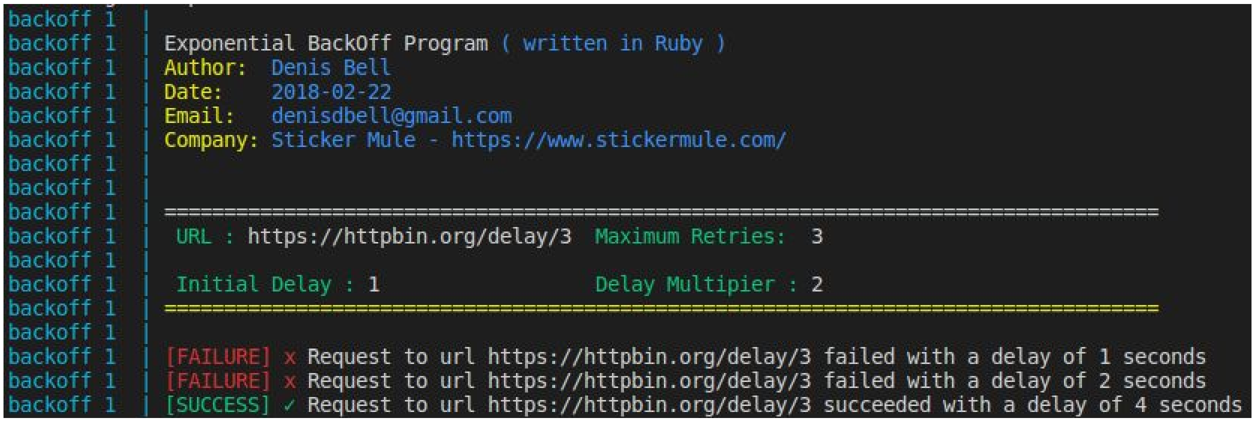
The **docker-compose.yml** file is located in the root directory of the project. It contains the configuration needed to execute the application.

```
version: '3'
services:
 backoff:
   build: .
   image: denisdbell/backoff:1.0
```

Navigate to the root directory of the application and type the following command to launch the application:

```
docker-compose up
```

You should see the following output displayed to stdout:



```
backoff 1  |
backoff 1  | Exponential BackOff Program ( written in Ruby )
backoff 1  | Author:  Denis Bell
backoff 1  | Date:    2018-02-22
backoff 1  | Email:   denisdbell@gmail.com
backoff 1  | Company: Sticker Mule - https://www.stickermule.com/
backoff 1  |
backoff 1  |
backoff 1  | ===================================================================================
backoff 1  |  URL : https://httpbin.org/delay/3  Maximum Retries:  3
backoff 1  |
backoff 1  |  Initial Delay : 1                  Delay Multipier : 2
backoff 1  | ===================================================================================
backoff 1  |
backoff 1  | [FAILURE] x Request to url https://httpbin.org/delay/3 failed with a delay of 1 seconds
backoff 1  | [FAILURE] x Request to url https://httpbin.org/delay/3 failed with a delay of 2 seconds
backoff 1  | [SUCCESS] ✓ Request to url https://httpbin.org/delay/3 succeeded with a delay of 4 seconds
```

Run using ruby

To execute the application using ruby, navigate to the root directory of the application and install the dependencies using the following command:

```
bundle install
```

Now run the application using the following command:

```
ruby lib/start_back_off.rb
```

Building And Pushing The Docker Image

When making modifications to the code for example bug fixes, enhancements etc, the docker image will need to be rebuilt and pushed to the docker hub repository. The **Dockerfile,** located in the root of the project directory, contains the information required to build the image. See the Dockerfile details below:

```
FROM ruby:2.5-alpine3.7

MAINTAINER Denis Bell <denisdbell@gmail.com>

RUN apk add --no-cache git

RUN mkdir /usr/app

COPY . /usr/app

WORKDIR /usr/app/

RUN bundle install

ENTRYPOINT ["ruby", "lib/start_back_off.rb"]
```

The image can be built using the **docker build** or **docker-compose build** commands. Both methods are shown below:

Building the image using the “docker build” command

Navigate to the root directory of the project and execute the following command to build the **backoff** image:

```
docker build . -t <docker hub username>/backoff:<version>
```

After the build process is complete. Push the image to the docker hub repository by using the following command:

```
docker push <docker hub username>/backoff:<version>
```

Building the image using “docker-compose”

Docker compose is the prefered method to build images because of its simplicity. Navigate to the root directory of the project and run the following command to build the **backoff** image:

```
docker-compose build
```

Note: Image name and version can changed in docker-compose.yml file

After the build process is complete. Push the image to the docker hub repository using the following command:

```
docker-compose push
```

Thanks for reading, Happy Coding!