HTML Analyzer

A simple web-application which allows the user to conduct some analysis of an html web page. It provides a simple interface where user can enter a html page url and submit. The application processes the given html url and returns the results of analysis in a simple tabular fashion.

Getting Started

Follow the instructions below to setup and build the project in our local machine.

Prerequisites

This project is build using the below tools. Please ensure you have the below prerequisites satisfied

```
1. JDK 1.8 or above
2. Gradle 4.1 or above
```

Open source libraries or plug-ins used

This project uses the below open source libraries, frameworks and plugins.

```
1. Spring Boot gradle plug-in 1.5.7.RELEASE
2. Jsoup 1.10.3
3. Guava:23.1-jre
4. Pagination.js 2.0.8
```

Installing

Clone and run the project using below commands.

```
unzip the html-analyzer.zip and then use your favorite ide to load the project by s
electing the build.gradle file.
```

If you are using a IDE for example Intellij then, import project and choose build.gradle file. For more information you can check out the link https://spring.io/guides/gs/intellij-idea/
(https://spring.io/guides/gs/intellij-idea/)

Testing

Use the below command to run automated test suite

```
./gradlew test
```

Running

Use the below command to build the project and run from the project root directory. The project will build and run in a embedded tomcat instance.

```
./gradlew bootRun
```

If the build is successfull then you should see something like this in the console

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v1.5.7.RELEASE)

2017-10-09 20:39:14.172  INFO 73274 --- [           main] htmlanalyzer.Application
                : No active profile set, falling back to default profiles: default
2017-10-09 20:39:14.358  INFO 73274 --- [           main] ationConfigEmbeddedWebApp
licationContext : Refreshing org.springframework.boot.context.embedded.AnnotationCo
nfigEmbeddedWebApplicationContext@2c1b194a: startup date [Mon Oct 09 20:39:14 CEST
2017]; root of context hierarchy
2017-10-09 20:39:16.678  INFO 73274 --- [           main] s.b.c.e.t.TomcatEmbeddedS
ervletContainer : Tomcat initialized with port(s): 8080 (http)
2017-10-09 20:39:16.706  INFO 73274 --- [           main] o.apache.catalina.core.St
andardService   : Starting service [Tomcat]
2017-10-09 20:39:16.708  INFO 73274 --- [           main] org.apache.catalina.core.
StandardEngine  : Starting Servlet Engine: Apache Tomcat/8.5.20
2017-10-09 20:39:16.972  INFO 73274 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[local
host].[/]       : Initializing Spring embedded WebApplicationContext
2017-10-09 20:39:16.972  INFO 73274 --- [ost-startStop-1] o.s.web.context.ContextLo
ader            : Root WebApplicationContext: initialization completed in 2644 ms
2017-10-09 20:39:17.161  INFO 73274 --- [ost-startStop-1] o.s.b.w.servlet.ServletRe
gistrationBean  : Mapping servlet: 'dispatcherServlet' to [/]
2017-10-09 20:39:17.172  INFO 73274 --- [ost-startStop-1] o.s.b.w.servlet.FilterReg
istrationBean   : Mapping filter: 'characterEncodingFilter' to: [/*]
2017-10-09 20:39:17.173  INFO 73274 --- [ost-startStop-1] o.s.b.w.servlet.FilterReg
istrationBean   : Mapping filter: 'hiddenHttpMethodFilter' to: [/*]
2017-10-09 20:39:17.173  INFO 73274 --- [ost-startStop-1] o.s.b.w.servlet.FilterReg
istrationBean   : Mapping filter: 'httpPutFormContentFilter' to: [/*]
2017-10-09 20:39:17.173  INFO 73274 --- [ost-startStop-1] o.s.b.w.servlet.FilterReg
istrationBean   : Mapping filter: 'requestContextFilter' to: [/*]
2017-10-09 20:39:17.729  INFO 73274 --- [           main] s.w.s.m.m.a.RequestMappin
gHandlerAdapter : Looking for @ControllerAdvice: org.springframework.boot.context.e
mbedded.AnnotationConfigEmbeddedWebApplicationContext@2c1b194a: startup date [Mon O
```

```
ct 09 20:39:14 CEST 2017]; root of context hierarchy
2017-10-09 20:39:17.855  INFO 73274 --- [           main] s.w.s.m.m.a.RequestMappin
gHandlerMapping : Mapped "{[/rest/metadata]}" onto public htmlanalyzer.rest.model.M
etaData htmlanalyzer.rest.service.HTMLAnalyzeController.metadata(java.lang.String)
2017-10-09 20:39:17.857  INFO 73274 --- [           main] s.w.s.m.m.a.RequestMappin
gHandlerMapping : Mapped "{[/rest/links]}" onto public htmlanalyzer.rest.model.Link
s<htmlanalyzer.rest.model.Link> htmlanalyzer.rest.service.HTMLAnalyzeController.lin
ks(java.lang.String,int,int)
2017-10-09 20:39:17.860  INFO 73274 --- [           main] s.w.s.m.m.a.RequestMappin
gHandlerMapping : Mapped "{[/error]}" onto public org.springframework.http.Response
Entity<java.util.Map<java.lang.String, java.lang.Object>> org.springframework.boot.
autoconfigure.web.BasicErrorController.error(javax.servlet.http.HttpServletRequest)
2017-10-09 20:39:17.861  INFO 73274 --- [           main] s.w.s.m.m.a.RequestMappin
gHandlerMapping : Mapped "{[/error],produces=[text/html]}" onto public org.springfr
amework.web.servlet.ModelAndView org.springframework.boot.autoconfigure.web.BasicEr
rorController.errorHtml(javax.servlet.http.HttpServletRequest,javax.servlet.http.Ht
tpServletResponse)
2017-10-09 20:39:17.914  INFO 73274 --- [           main] o.s.w.s.handler.SimpleUrl
HandlerMapping  : Mapped URL path [/webjars/**] onto handler of type [class org.spr
ingframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-10-09 20:39:17.915  INFO 73274 --- [           main] o.s.w.s.handler.SimpleUrl
HandlerMapping  : Mapped URL path [/**] onto handler of type [class org.springframe
work.web.servlet.resource.ResourceHttpRequestHandler]
2017-10-09 20:39:18.022  INFO 73274 --- [           main] o.s.w.s.handler.SimpleUrl
HandlerMapping  : Mapped URL path [/**/favicon.ico] onto handler of type [class or
g.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-10-09 20:39:18.119  INFO 73274 --- [           main] oConfiguration$WelcomePa
geHandlerMapping : Adding welcome page: class path resource [static/index.html]
2017-10-09 20:39:18.593  INFO 73274 --- [           main] o.s.j.e.a.AnnotationMBea
nExporter        : Registering beans for JMX exposure on startup
2017-10-09 20:39:18.747  INFO 73274 --- [           main] s.b.c.e.t.TomcatEmbedded
ServletContainer : Tomcat started on port(s): 8080 (http)
2017-10-09 20:39:18.756  INFO 73274 --- [           main] htmlanalyzer.Application
                 : Started Application in 5.63 seconds (JVM running for 7.174)
```

How to use the application?

To check and ensure everything works as expected, open a browser window and hit the below url.

```
http://localhost:8080
```

You should see a page with a text box prompting for url to be entered. Enter the URL and click 'Analyze' as shown below.

```
screen-shot: /document/html-analyzer.png
```

The results are shown in a simple tabular fashion which includes below items

1. HTML Version

2. Page title
3. If the page has a login form or not
4. Total number of different heading types
5. Total number of links categorized as internal and external links. Internal links are the links which contain the same domain as the url queried for and external links are ones which are pointing to a different domain. ``` screen-shot: /document/html-metadata.png

```
6. Paginated information containing different links and if the links are reachable
 and a simple status message.
For example, showing information on what went wrong (error message or http status c
ode) while trying to access the link.
```

screen-shot: /document/html-links.png

```

## Assumptions

Following are some of the assumptions made during the development.

1. All the urls will be submitted with http:// or https:// as a prefix
2. The task descriptions says 'Number of hypermedia links' and 'Number of headings', so here I assume that it is not a typo and expected is only number (count) and not actual value.
3. I did not fully understand one of the optional part which says 'Consider the effect of re-direction', I was not sure what is required here and hence I have left this part untouched.
4. Latest version of modern browsers are to be supported. Behaviour of the application is undefined when old version of browser is used.

## Design decision

1. Taking my programming language strengths into consideration I have decided to use Java 8 and coded the sever side and majority of business logic in Java 8
2. Decided to use simple html, css and javascript to render the results in the browser. Client-side code is only rendering the screens and all the core business logic is served by the server side.
3. Spring boot framework is one of the most modern framework which offers a lot of out of the box features such as support for REST framework, embedded tomcat-server with minimal set of configuration and annotations and hence this is probably the first choice for server side development.
4. JSoup is a simple yet powerful library to parse html in java and hence I had no second thoughts using this.
5. For domain name matching in the url, I found Guava:23.1-jre library quite simple and fits the purpose.
```

6. For client side pagination I used a simple pagination.js plugin, its quite simple and easy to use.
7. For the optional part, I decided to perform pagination of links since performance is quite relevant for this part.
8. Pagination.js is a simple plug-in that can be used to implement pagination in the client side using jQuery.

Limitation

1. The application does not show and error message when a invalid url is provided as input.
2. For large html page containing hugh number of internal and external links, the performance of link retrieval can drop slightly considering the amount of data that has to be parsed in the html document. This can be fine-tuned to some extent by specifying the 'pageSize' parameter in the client size and also by reducing the time_out value required to check if the link is reachable or not.

Future improvements

The following improvements can be considered for future releases.

1. Show client-side error messages. For example when user enters invalid url.
2. Provide configurable parameters such as TIMEOUT or defining page size in client side etc.
3. Server side caching of links instead of repeatedly fetching the links for the same given url. The links more often does not change overnight hence it can be cached in the server side with a simple LRU caching mechanism using tools such as redis or key-value pair db such as aws dynamodb.
4. Retrieving links from jsoup api and its reachable information can be done as a async process running in a map-reduce framework to speedup the processing.
5. An automated batch job which processes all the most popular html urls on a daily basis and stores this in a no-sql database as a document which later can be retrieved using elastic search or solr full text search engine when user queries for a url. This can significantly speed up the data retrieval for links to check if it is reachable or not.