IDGen Document

Table of Contents

[Overview: 1](#_Toc448507382)

[Requirement: 1](#_Toc448507383)

[Implementation: 2](#_Toc448507384)

[1) get ID: 2](#_Toc448507385)

[2) ListIDSelector: 2](#_Toc448507386)

[Test Request and Response: 2](#_Toc448507387)

[Success URL 2](#_Toc448507388)

[1) To save the selector to files, and generate the id we have the following url. 2](#_Toc448507389)

[2) To get the listIDSelector we have the following url. 3](#_Toc448507390)

[Failure URL: 3](#_Toc448507391)

[1) Error message for Special character. 3](#_Toc448507392)

[2) If the File path is missing. 4](#_Toc448507393)

[3) If ID Reaches maximum i.e[9999999999L]. 7](#_Toc448507394)

[File Contents 8](#_Toc448507395)

[Content of Tracker File: 8](#_Toc448507396)

[Content of Selector File: 8](#_Toc448507397)

[Content of Log File: 8](#_Toc448507398)

# Overview:

IDGen, provide automatically generated IDs based on a selector which is supplied by the caller using rest url, and stores the data into files.

Data storage for the IDGen takes place in text files,

1. IDGen\_Tracker.txt: Stores the selector value and its generated ID.
2. IDGen\_Selector.txt: Stores the caller and selector values if the tracker file does not have its record.
3. IDGen\_Log.txt: Stores the Log details.

# Requirement:

1. get ID: For every rest url request, increment the selector's id and returns it in json format.
2. ListIDSelector: ListIDSelector will return all callers and selectors from the “IDGen\_Selector.txt” file, and return it in json format.

# Implementation:

Controller class has two restful request.

1) get ID: For every rest url request, increment the selector's id and returns it in json format.

1. The rest url will contain caller and selector, for every url request the selector values is being checked with the “IDGen\_Tracker.txt” file selector value, and if the file contains the selector value the id of selector in “IDGen\_Tracker.txt” file will increment by 1.
2. If the selector value is not in “IDGen\_Tracker.txt” file, then the selector adds the caller value and selector value to the “IDGen\_Selector.txt” file. And add selector value with id 1 in “IDGen\_Tracker.txt” file.
3. “IDGen\_Log.txt” file will save the time date and the values of caller and selector with the id.

2) ListIDSelector: ListIDSelector will return all callers and selectors from the “IDGen\_Selector.txt” file, and return it in json format.

1. When the request comes for ListIDSelector, the “IDGen\_Selector.txt” file is read, and return the data as key and value pair of selector value and its caller value in json format.

# Test Request and Response:

Success URL***:***

### 1) To save the selector to files, and generate the id we have the following url.

url:

http://localhost:8080/getID/{CALLER}?selector={SELECTOR}

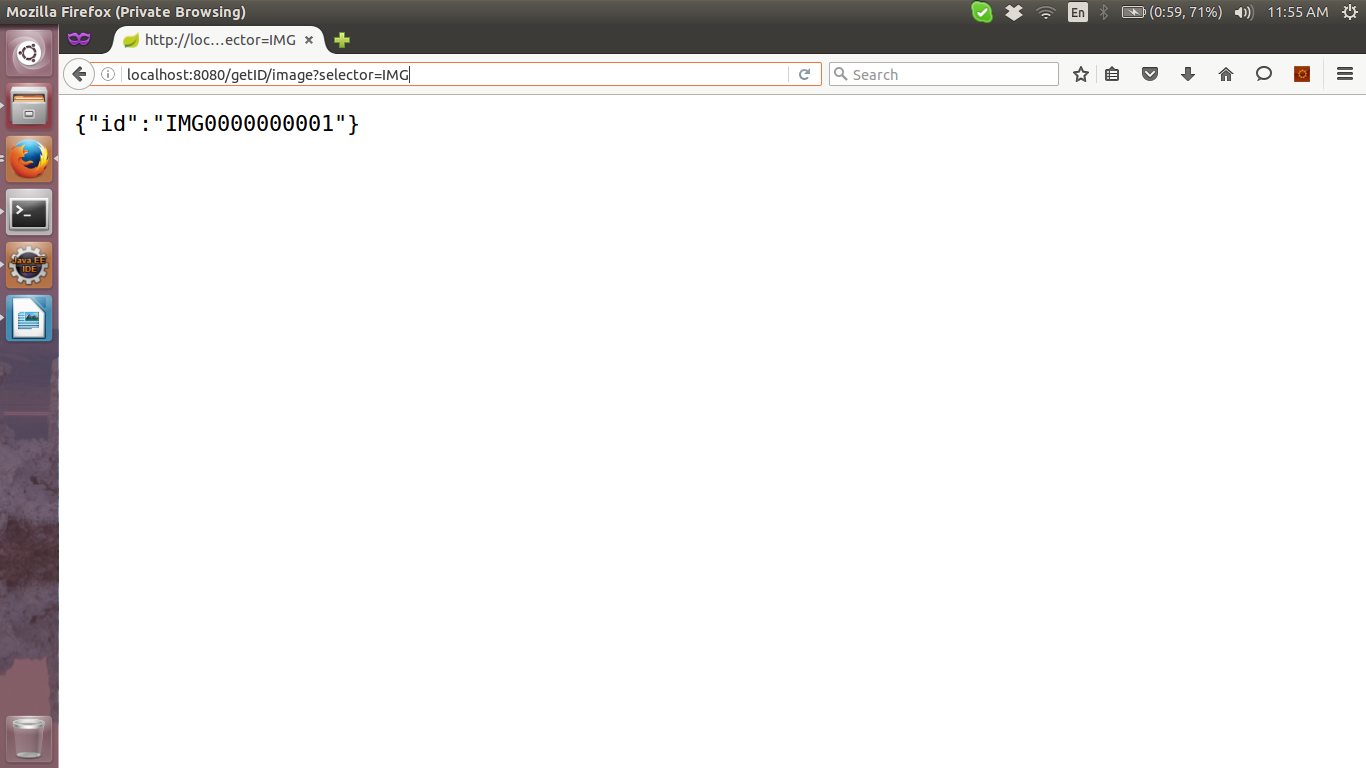
example :

Request: http://localhost:8080/getID/image?selector=IMG

Response will be: {"id":"IMG0000000001"}

Note: The application name can have 3 special characters (:,-,\_), rest all special characters gives us the error message.

**Screenshot**:

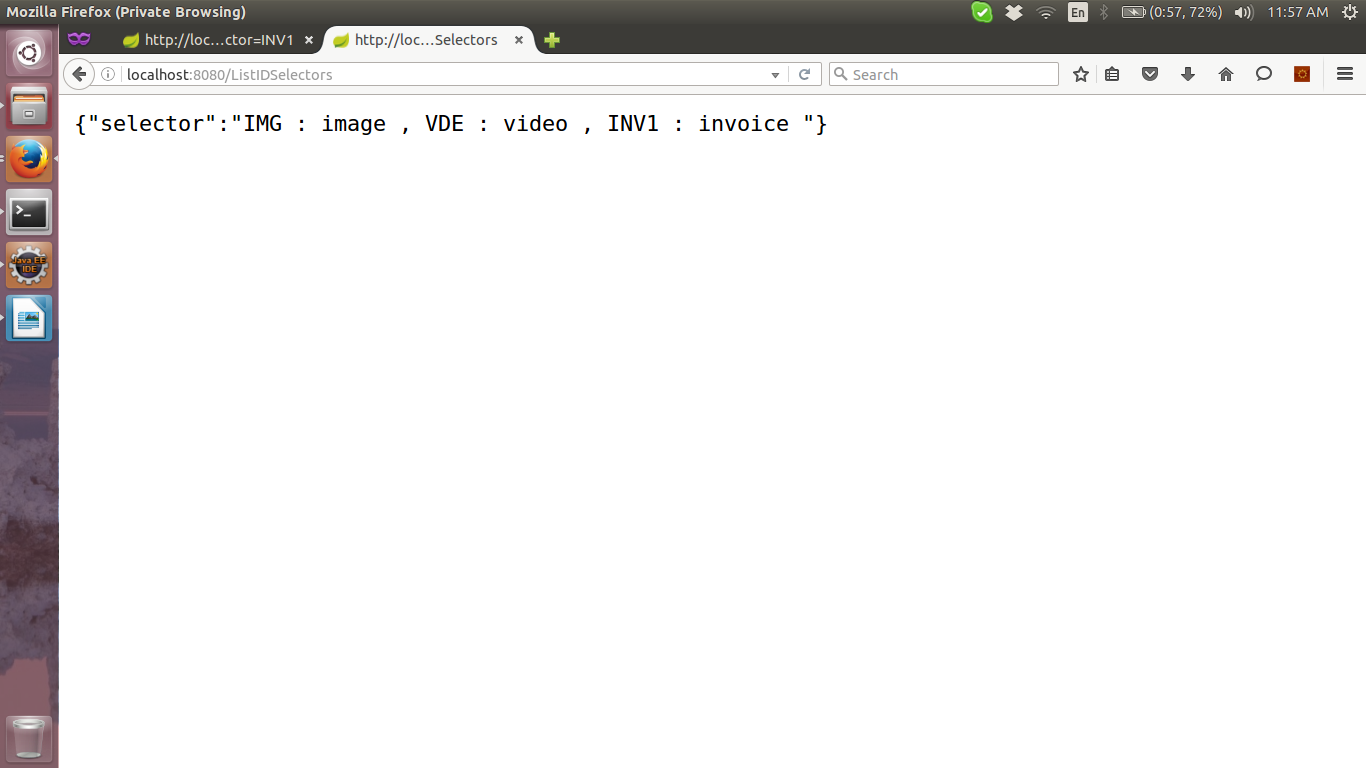


Note: Numbers are allowed with the selector value and selector values are case insensitive.

### 2) To get the listIDSelector we have the following url.

Request: http://localhost:8080/ListIDSelectorsResponse will be: {"selector":"IMG : image , IMG:01 : image , IMG:02 : image , IMG:03 : image "}.

**Screenshot:**



## Failure URL:

### 1) Error message for Special character.

Example1: When special character @ is used.

Request: <http://localhost:8080/getID/image?selector=IMG>!

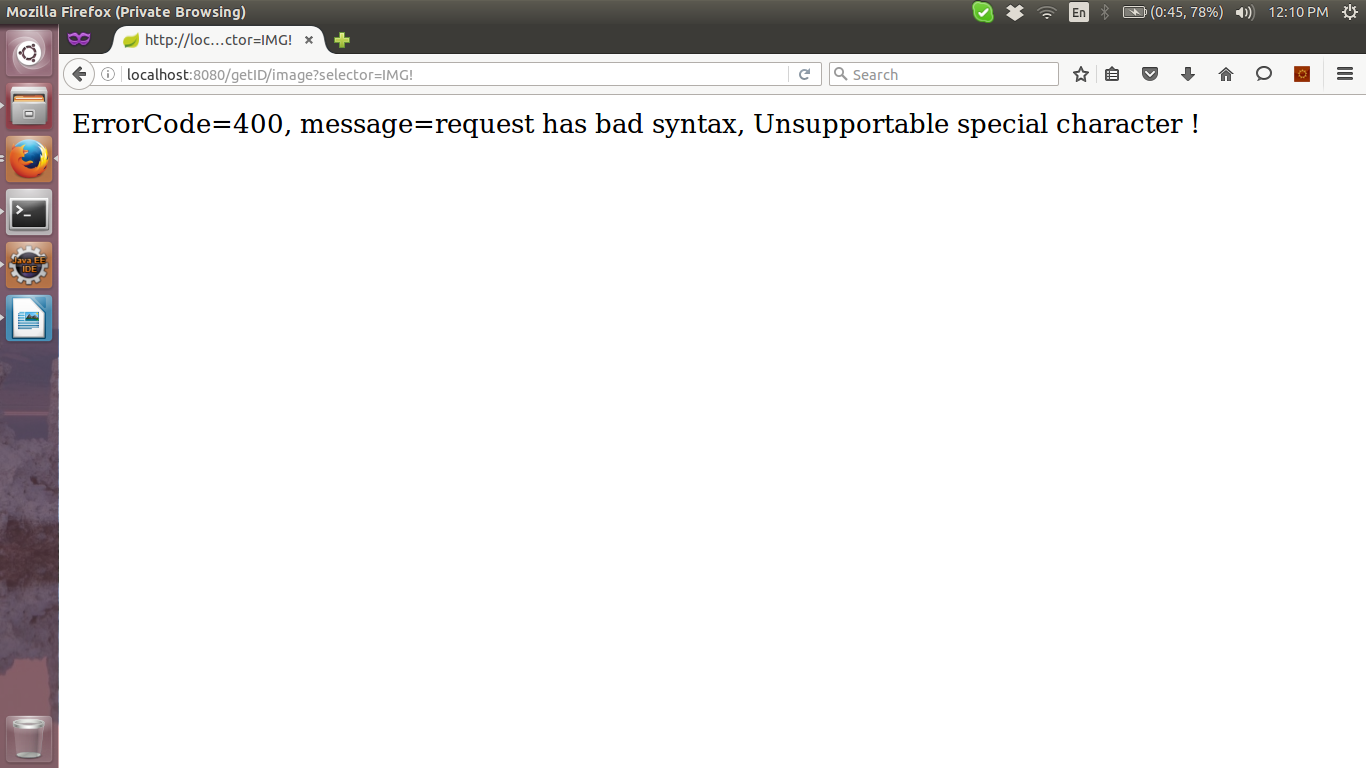
Response will be: ErrorCode=400, message=request has bad syntax, Unsupportable special character !

Example1: When special character @ is used.

Request: <http://localhost:8080/getID/image?selector=IMG@>

Response will be: ErrorCode=400, message=request has bad syntax, Unsupportable special character @

**Screenshot:**



### 2) If the File path is missing.

I) If the path of “IDGen\_Tracker.txt” is missing the error response will be.

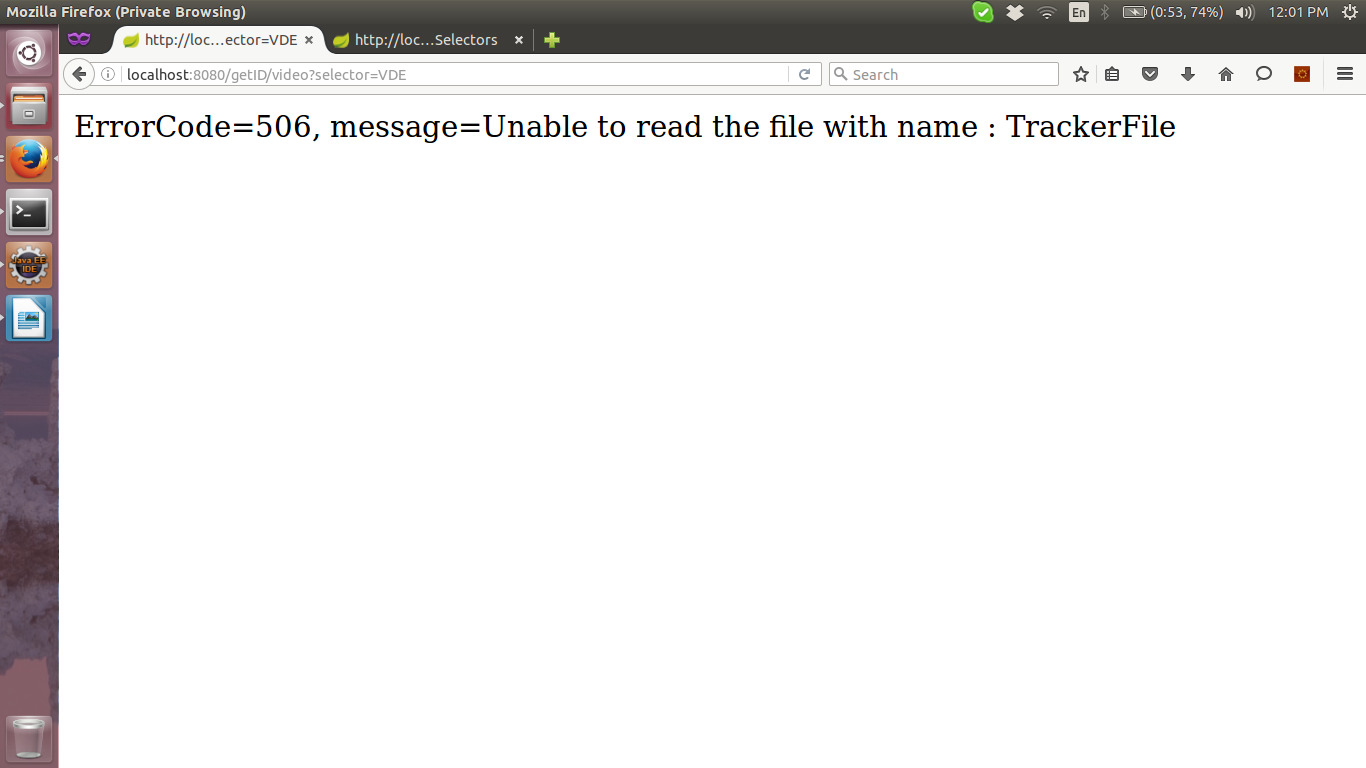
**Request:**

When request for “getID” comes and if the path of “IDGen\_Tracker.txt” is missing or there is no file in the mentioned path then

**Response:**

ErrorCode=506, message=Unable to read to file with name: TrackerFile

**Screenshot:**



II) If the path of “IDGen\_Selector.txt” is missing the error response will be.

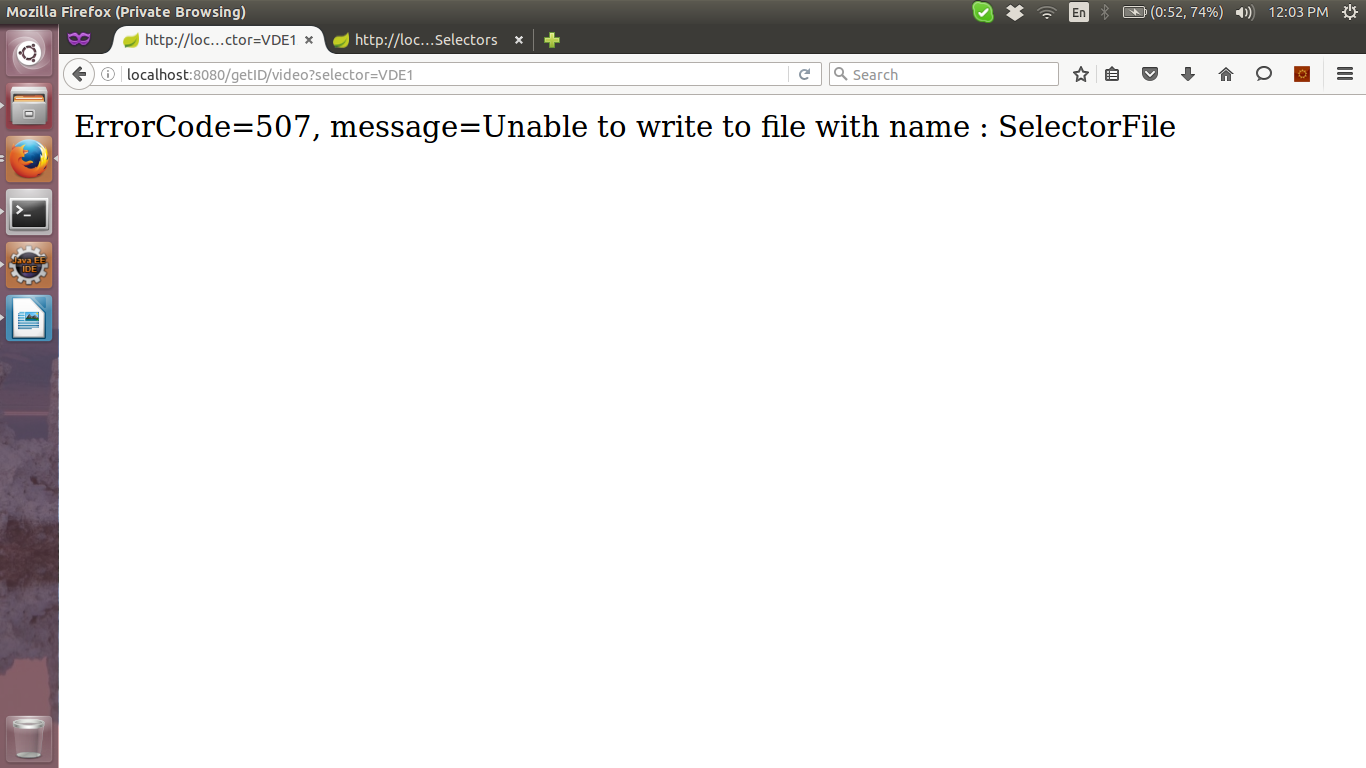
**Request:**

When request for “getID” comes and if the path of “IDGen\_Selector.txt” is missing or there is no file in the mentioned path then

**Response:**

ErrorCode=507, message=Unable to Write to file with name : SelectorFile.

**Screenshot:**

****

**Request:**

When request for “ListIDSelector” comes and if the path of “IDGen\_Selector.txt” is missing or there is no file in the mentioned path then

**Response:**

ErrorCode=506, message=Unable to Read to file with name: SelectorFile.

**Screenshot:**

****

III) If the path of “IDGen\_Log.txt” is missing the error response will be.

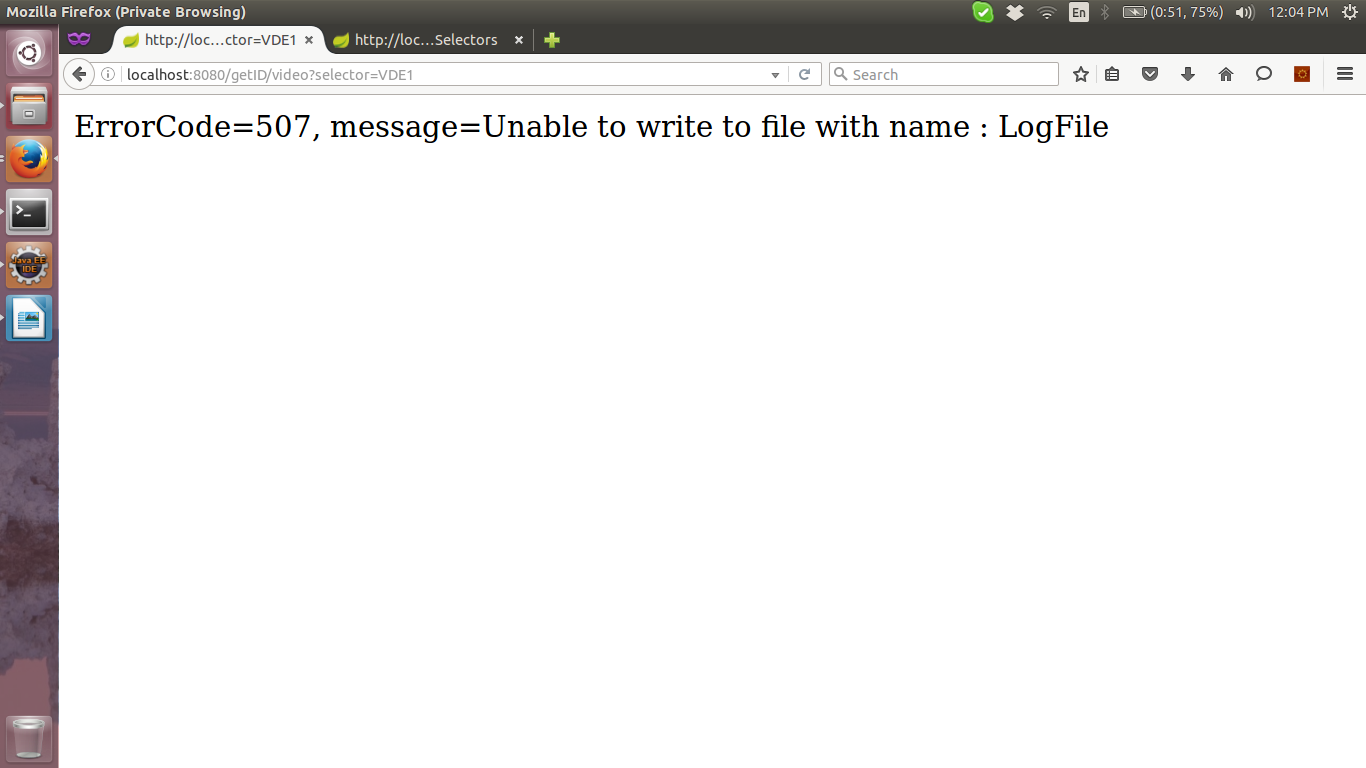
**Request:**

When request for “getID” comes and if the path of “IDGen\_Log.txt” is missing or there is no file in the mentioned path then

**Response:**

ErrorCode=507, message=Unable to write to file with name: LogFile

**Screenshot:**

****

### 3) If ID Reaches maximum i.e[9999999999L].

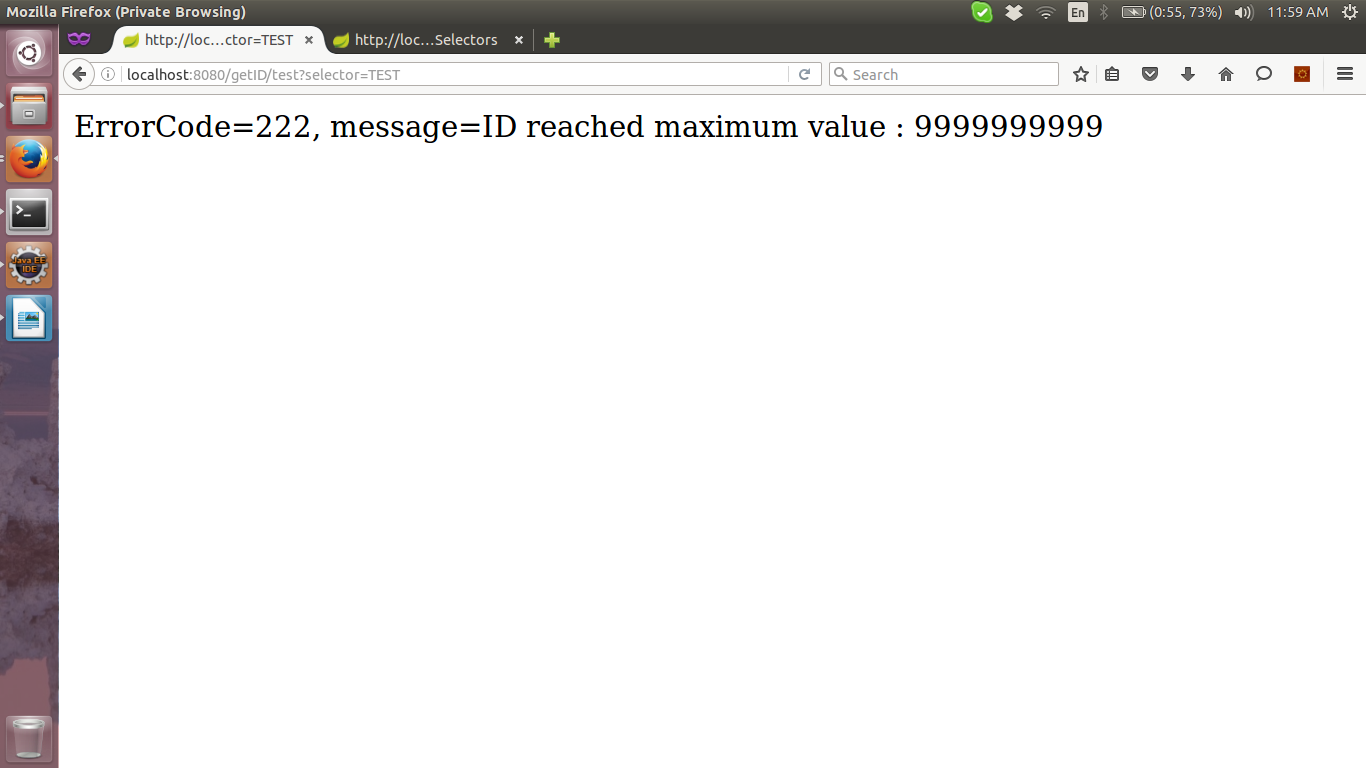
**Request:**

When request for “getID” comes and the selector value reaches it maximum number then

**Response:**

ErrorCode=222, message=ID reached maximum value 9999999999.

**Screenshot:**



# File Contents

## Content of Tracker File:

IMG 1

VDE 1

INV 1

## Content of Selector File:

IMG image

VDE video

INV invoice

## Content of Log File:

2016.04.15 at 01:12:05 PM IST Fri image IMG IMG0000000001

2016.04.15 at 01:12:14 PM IST Fri video VDE VDE0000000001

2016.04.15 at 01:12:27 PM IST Fri invoice INV INV0000000001