**API Specification Doc**

**(*Numbers Game*)**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Author** |
| 1.0 | 11-Nov-2017 | Rajeev Varma |

Index

1. range – post and get

[Request](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.kd6tjl5w5wjo)

[Response](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.8b7ijrpuamb8)

2. guesses- post

[Request](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.38pypefkb82)

[Response](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.47awg09ehquu)

3. guess history- guesses get

[Request](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.xtszbus9yomc)

[Response](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.xuhh3y4ufz7)

[Conventions](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.uk42zio712fb)

[Status Codes](https://docs.google.com/document/d/15kfl1kVwul6_JWONgrhy3jD0-R_aEbI2Vg5o5iFXfR8/edit#heading=h.of6ke88xwee4)

Methods

**1. Range - POST**

**POST** the range of the numbers to be used and the correct value to guess later (minimum, maximum and correct value)

**Request**

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | /curl -i -H "Content-Type: application/json" -X POST -d '{"minimum":23,"maximum":27,"correct":24}' <http://localhost:5000/range> |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| POST  POST  POST | minimum  maximum  correct | Integer  Integer  Integer |

**Response:**

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | {  "correct": 24,  "maximum": 27,  "minimum": 23  } |
| 400 | Responses for POST:  {  "error": "Bad Request"  } |

**Range – GET**

Get the values of max, min range to guess the correct value in that range.

**Request**

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | **curl -i** [**http://localhost:5000/range**](http://localhost:5000/range) |

**Response**

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | **Response will be an object containing minimum and maximum values of the range**  {  "maximum": 27,  "minimum": 23  } |
| 404 | **Response for GET - If the range is not posted earlier**  {  "error": "Not found"  } |

**2. Guesses - POST**

Post the values of guesses to be tested against the true value.

**Request**

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | curl -i -H "Content-Type: application/json" -X POST -d '{"guess": 24}' http://localhost:5000/guesses |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| POST | guess | Integer |

**Response**

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | Guesses:  If the guessed value is lower than the correct value  {  "guess": 23,  "result": "lower than correct value"  }  If the guessed value is higher than the correct value  {  "guess": 25,  "result": "higher than correct value"  }  If the guessed value == correct value  {  "guess": 24,  "result": "correct"  }  If the guess value is outside the range  {  "guess": 28,  "result": "outside range"  } |
| 400 | **Response for POST - If the range is None or there is no Guess parameter in the request:**  {  "error": "Bad Request"  } |

**4. Guesses – GET**

Get guess history – If there is none then returns empty list, when we make a few guesses and can have the entire history of our guesses in one place.

**Request**

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | curl -i http://localhost:5000/guesses |

**Response**

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | Initially:  {  "guesses": []  }  After Guessing:  {  "guesses": [  {  "guess": 25,  "result": "higher than correct value"  },  {  "guess": 23,  "result": "lower than correct value"  },  {  "guess": 24,  "result": "correct"  },  {  "guess": 28,  "result": "outside range"  }  ]  } |

Glossary

**Conventions**

* **Client** - Client application.
* **Status** - HTTP status code of response.
* All the possible responses are listed under ‘Responses’ for each method. Only one of them is issued per request server.
* All response are in JSON format.

**Status Codes**

All status codes are standard HTTP status codes. The below ones are used in this API.

2XX - Success of some kind

4XX - Error occurred in client’s part

5XX - Error occurred in server’s part

|  |  |
| --- | --- |
| **Status Code** | **Description** |
| 200 | OK |
| 201 | Created |
| 400 | Bad request |
| 404 | Resource not found |