

Saga General Functionality

This REST Handler contains:

- Endpoints that return Saga configuration and build information
- Endpoints to process text and return the interpretation graph

- [POST _saga/processText](#)
- [POST _saga/processBatch](#)
- [GET _saga/solution](#)
- [GET _saga/info](#)
- [GET _saga/stopServer](#)

POST _saga/processText

- Will return a Saga interpretation graph in the format specified
- By default, it returns the graph in HTML format which is used by the UI
- It also can return JSON or plain text

Parameters

- **q** (*type=string | required*) - This is the text to process
- **tags** (*type=string array | required*) - List of tags to use to process the text
- **tag** (*type=string | optional*) - Tag to use to process the text
- **processor** (*type=string | optional*) - Pipeline or pipeline stage to use to process the text
- **splitRegex** (*type=string | default=[\r\n]+ | optional*) - Specifies the regex to use to split text into TEXT_BLOCKS
- **combine** (*type=boolean | default=false | optional*) - If true, it will combine all the SEMANTIC_TAGS, which has same matching text and same confidence value
 - The combination includes the metadata, of every SEMANTIC_TAG
- **addComponents** (*type=boolean | default=true | optional*) - If true, it will also add the components from the highest route confidence tokens.
- **compFields** (*type=string array | optional*) - Applicable when addComponents is true. Indicates which components to add to the response
 - If missing all the components will be added
- **metadata** (*type=boolean | default=false | optional*) - If true, it will add the metadata of the entities found in every SEMANTIC_TAG
- **type** (*type=string | default=ux | optional*) - Specifies the format of the results. It could be one of these values: "ux", "text" or "json"
 - **ux** - Returns an output useful for Saga to show the graph on a web display.
 - **text** - Returns a "line" with the highest confidence route and a text representation of the graph.
 - **json** - Returns all the semantic tags on the graph.
- **pretty** (*type=boolean | default=false | optional*) - Used to pretty format the JSON result



The parameters tags, tag and processor are mutually exclusive, only one is required, and the priority order is:

1. tags

2. tag

Request Examples

3. processor

```
curl -X POST \
  http://localhost:8080/_saga/processText \
  -H 'Content-Type: application/json' \
  -d '{
    "q": "I like traveling with Air Paris",
    "tags": ["airline"],
    "combine": true,
    "addComponents": true,
    "compFields": "paris",
    "splitRegex": "[\r\n ]+",
    "type": "text",
    "pretty": true
  }'
```

Response

`$action.getHelper().renderConfluenceMacro("$codeS$body$codeE")`

POST _saga/processBatch

- Will return a Saga interpretation graph in the format specified
- By default, it returns the graph in HTML format which is used by the UI

- It also can return JSON or plain text

Parameters

- **batch** (*type=string array | required*) - List of texts to process
- **tags** (*type=string array | required*) - List of tags to use to process the text
- **tag** (*type=string | optional*) - Tag to use to process the text
- **processor** (*type=string | optional*) - Pipeline or pipeline stage to use to process the text
- **splitRegex** (*type=string | default=[\r\n]+ | optional*) - Specifies the regex to use to split text into TEXT_BLOCKS
- **combine** (*type=boolean | default=false | optional*) - If true, it will combine all the SEMANTIC_TAGS, which has same matching text and same confidence value
 - The combination includes the metadata, of every SEMANTIC_TAG
- **addComponents** (*type=boolean | default=true | optional*) - If true, it will also add the components from the highest route confidence tokens.
- **compFields** (*type=string array | optional*) - Applicable when addComponents is true. Indicates which components to add to the response
 - If missing all the components will be added
- **metadata** (*type=boolean | default=false | optional*) - If true, it will add the metadata of the entities found in every SEMANTIC_TAG



The parameters tags, tag and processor are mutually exclusive, only one is required, and the priority order is:

1. tags

Request Examples

2. tag

3. processor

```
curl -X POST \
  http://localhost:8080/_saga/processBatch \
  -H 'Content-Type: application/json' \
  -d '
{
  "batch": ["I like traveling with Air Paris", "I enjoy traveling with Delta"],
  "tags": ["airline"],
  "combine": true,
  "addComponents": true,
  "compFields": "paris"
}'
```

Response

`$action.getHelper().renderConfluenceMacro("$codeS$body$codeE")`

GET _saga/solution

- Returns the information of the solution currently being used.
- A solution is basically the set of Elasticsearch indices in use

Request Examples

```
curl --location --request GET 'http://localhost:8080/_saga/solution'
```

Response

`$action.getHelper().renderConfluenceMacro("$codeS$body$codeE")`

GET _saga/info

- Returns build-specific and configuration information about Saga
- Build information contains title, vendor, version, build revision and date
- Configuration information contains the solution (Elastic index) and providers configured at the moment

Request Examples

```
curl --location --request GET 'http://localhost:8080/_saga/info'
```

Response

`$action.getHelper().renderConfluenceMacro("$codeS$body$codeE")`

GET _saga/stopServer

- Stops the Saga Server



If you stop the server via HTTP request, it can only be restarted on-premise

Request Examples

```
curl --location --request GET 'http://localhost:8080/_saga/stopServer'
```

Response