REST API

Note:

<https://restfulapi.net/resource-naming/>

<https://blog.dreamfactory.com/best-practices-for-naming-rest-api-endpoints/>

Service descriptions:

* species: retrieves the list of species

GET services

* Response: JSON. The list of species

[ “species1”, “species2”,…]

* Example:

{URL}/species

* samples: retrieves the samples by the species name

GET services

* Parameters:
  + species, list of species, separated by comma.

If it is not defined, return the list of all samples for all species.

* Response: JSON. Hash with list. The list of samples per species

{

“species1”: [s1, s2, sS],

“species2”: [ss1, ss2, ssS],

…

}

* Examples:

{URL}/samples

{URL}/samples?species={species1}

{URL}/samples?species={species1&species={species2}

* proteins: retrieves the proteins by the species name

GET services

* Parameters:
  + species, list of species, separated by comma.

If it is not defined, return the list of all proteins for all species.

* Response: JSON. Hash with list. The list of samples per species

{

“species1”: [p1, p2, pS],

“species2”: [pp1, pp2, ppS],

…

}

* Examples:

{URL}/proteins

{URL}/proteins?species={species1}

{URL}/samples?species={species1&species={species2}

* category: retrieves the categories by the species name

GET services

* Parameters:
  + species, list of species, separated by comma.

If it is not defined, return the list of all categories for all species.

* Response: JSON. Hash with list. The list of samples per species

{

“species1”: [s1, s2, sS],

“species2”: [ss1, ss2, ssS],

…

}

* Examples:

{URL}/categories

{URL}/categories?species={species1}

{URL}/ categories?species={species1,species2}

* complexes: retrieves the complexes by the species name

GET services

* Parameters:
  + species, list of species, separated by comma.

If it is not defined, return the list of all complexes for all species.

* Response: JSON. Hash with list. The list of complexes per species

{

“species1”: [s1, s2, sS],

“species2”: [ss1, ss2, ssS],

…

}

* Examples:

{URL}/complexes

{URL}/complexes?species={species1}

{URL}/complexes?species={species1,species2}

* get-psms (no me gusta el nombre): retrieves the values of PSMs for the table slice/proteins.

If the sample query is empty, then it retrieves the values for all samples from the given prot/category/complex

POST

* Parameters: Query with the attributes:
  + proteins, list of proteins,
  + categories, list of categories,
  + complexes, list of complexes.
  + samples, list of samples.

JSON

{

“proteins”: [p1, p2, p3, pP],

“categories”: [c1, c2, c3, …, cC],

“complexes”: [m1, m2, …mM],

“samples”: [s1, s2, …sS]

}

* Response: JSON de la misma forma que necesita la librería Plotly de JS. Por ejemplo:

var data = [ {

z: z,

x: x,

y: y,

}];

var data = [ {

z: [[1, null, 30, 50, 1], [20, 1, 60, 80, 30], [30, 60, 1, -10, 20]],

x: ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday'],

y: ['Morning', 'Afternoon', 'Evening'],

type: 'heatmap',

hoverongaps: false

}];

* get-escores (no me gusta el nombre): retrieves the values of e-scores for the table slice/proteins

If the sample query is empty, then it retrieves the values for all samples from the given prot/category/complex.

POST

* Parameters: Query with the attributes:
  + proteins, list of proteins,
  + categories, list of categories,
  + complexes, list of complexes.
  + samples, list of samples.

{

“proteins”: [p1, p2, p3, pP],

“categories”: [c1, c2, c3, …, cC],

“complexes”: [m1, m2, …mM],

“samples”: [s1, s2, …sS]

}

* Response: JSON de la misma forma que necesita la librería Plotly de JS. Por ejemplo:

var data = [ {

z: z,

x: x,

y: y,

}];

* get-corrs (no me gusta el nombre): retrieves the correlations for the table slice/proteins based on the given prot/category/complex

If the sample query is empty, then it retrieves the values for all samples from the given prot/category/complex.

POST

* Parameters: Query with the attributes:
  + p-value, p-value
  + method, Method of correlation: Spearman, Pearson, kendall. By default, Spearman.
  + proteins, list of proteins,
  + categories, list of categories,
  + complexes, list of complexes.
  + samples, list of samples.

{

“p-value”: 0.01,

“method”: “spearman”,

“proteins”: [p1, p2, p3, pP],

“categories”: [c1, c2, c3, …, cC],

“complexes”: [m1, m2, …mM],

“samples”: [s1, s2, …sS]

}

* Response: JSON de la misma forma que necesita la librería Plotly de JS. Por ejemplo:

var data = [ {

z: z,

x: x,

y: y,

}];

* get-upsets (no me gusta el nombre): retrieves the values for the creation of UpSet graphs.

Upset in JS

<https://medium.com/@sgratzl/upset-js-javascript-tutorial-1b84bfd6896d>