DataGator Specification: Data Model*

author: LIU Yu <liuyu@opencps.net>
revision: 1.3

Background

This document specifies the conceptual data model of DataGator, namely, the identification, format, and operation of core elements including repository, data set, data item, etc. Targeted readers of this document are developers experienced in data processing, especially, manipulating JSON objects in accordance with a formal schema.

Repository and Data Set

Identifier

DataGator uses <Owner>/<Name> as the primary identifier of DataSet's, where <Owner> is the registered name of a user of DataGator, and <Name> is a *variable name*[1] chosen for the DataSet. In the context of data management, <Owner> is conceptually a repository, or Repo, collecting all DataSet's of the same user.

Internally, each DataSet will be assigned a version 4 UUID. For example, if the DataSet was submitted by user Pardee and named IGOs, and if DataGator has assigned fc4d4da3-d998-4d55-a8f5-fd36cce0f643 as its id, then the following two URL's[2] are equivalent for accessing this dataset.

```
Pardee/IGOs
fc4d4da3-d998-4d55-a8f5-fd36cce0f643
```

Exchange Format

When accessing datasets through the RESTful API (i.e. programmatic interface) of DataGator, the responses are JSON objects conforming to the following schema[3],

http://www.data-gator.com/api/v1/schema

[1] regular expression pattern "[A-Za-z][A-Za-z0-9_]{0,29}"

[2] Unless otherwise specified, URL's are relative to either http(s)://www.data-gator.com/api/v1/ (for programmatic access), or http(s)://www.data-gator.com/ (for browser access).

For the exemplified `Pardee/IGOs` dataset, the response from the programmatic interface of `DataGator` should look like,

```
{
    "kind": "datagator#DataSet",
    "name": "IGOs",
    "repo": {
        "kind": "datagator#Repo",
        "name": "Pardee"
    },
    "id": "fc4d4da3-d998-4d55-a8f5-fd36cce0f643",
    "rev": 3,
    "created": "2014-06-03T18:00:23Z",
    "items": [
    {
        "kind": "datagator#Matrix",
        "name": "AAAID"
    },
    {
        "kind": "datagator#Matrix",
        "name": "AAEA"
    },
    ...
    {
        "kind": "datagator#Matrix",
        "name": "WTOURO"
    }],
    "itemsCount": 402
}
```

Version Control

Note from the above JSON snippet that the `DataSet` contains (i) a `rev` attribute that equals `3`, and (ii) a `created` attribute that equals `2014-06-03T18:00:23Z`. The former is the number of *revisions* that have been made to the `DataSet`, and the latter is the time of **commit** of the respective revision in ISO 8601 format.

`DataGator` stores all historical revisions of a `DataSet`. When accessing a `DataSet` through `<Owner>/<Name>` or `<UUID>`, as we demonstrated in previous section, the `HEAD` (or latest *revision*) of the `DataSet` will be returned. Alternatively, one can specify a `.<rev>` suffix to the URL's for accessing a particular *revision* of the `DataSet`. For example, one can access the `DataSet` with the following URL's

```
Pardee/IGOs.2
fc4d4da3-d998-4d55-a8f5-fd36cce0f643.2
```

And the 2nd revision of the `DataSet` will be returned, in which `DataItem`'s `ISESCO` thru `WTOURO` are not present. Intuitively, this means that these items were introduced to the `DataSet` in later revisions.

[3] Draft 4 JSON schema as defined by http://json-schema.org/

```
{
    "kind": "datagator#DataSet",
    "name": "IGOs",
    "repo": {
        "kind": "datagator#Repo",
        "name": "Pardee"
    },
    "id": "fc4d4da3-d998-4d55-a8f5-fd36cce0f643",
    "rev": 2,
    "created": "2014-06-03T17:55:32Z",
    "items": [
    {
        "kind": "datagator#Matrix",
        "name": "AAAID"
    },
    {
        "kind": "datagator#Matrix",
        "name": "AAEA"
    },
    ...
    {
        "kind": "datagator#Matrix",
        "name": "ISB"
    }],
    "itemsCount": 274
}
```

Data Item

Identifier and Exchange Format

`Matrix` is the primary form of `DataItem` in a `DataSet`. Conceptually, a `Matrix` is a 2D array with possibly heterogeneous data values. A `Matrix` can be accessed by its `<Key>`[4] from the container `DataSet`. For example, the `Matrix` labeled `WTO` from `Pardee/IGOs` can be accessed via the following URL's,

```
Pardee/IGOs/WTO
fc4d4da3-d998-4d55-a8f5-fd36cce0f643/WTO
```

And the (partial) response from `DataGator` should look like

```
{
    "kind": "datagator#Matrix",
    "columnHeaders": 1,
    "rowHeaders": 1,
    "rows": [
        [null, 1816, 1817, 1818, ... ],
        ["Abkhazia", null, null, null, ... ],
```

[4] Formal regex pattern of `<Key>` is not yet specified, the baseline requirement is that the `<Key>` may not contain URL-special characters, such as slash (`"/"`), question mark (`"?"`), hash (`"#"`), etc.

```
            ...
            ["Zimbabwe", null, null, null, ... ]
        ],
        "rowsCount": 337,
        "columnsCount": 198
    }
```

Structural Layout

In a `Matrix`, data values are arranged as an *array* of `#rowsCount` of `rows`, each containing an *array* of `#columnsCount` *primitive values* including, (i) `NULL` values, (ii) numeric values (integer or real), (iii) string literals (unicode), and (iv) `datetime` as strings in ISO 8601 format. Depending on the annotation during import, a `Matrix` may contain two optional counters `#columnHeaders` and `#rowHeaders`.

Intuitively, the `Matrix` model defines a four-block layout for tabular data, where (i) the first (one or few) *rows* contain descriptive information for each column, and are collectively named the *headers of columns* (or `columnHeaders`), (ii) the first (one or few) *columns* contain likewise descriptive information for each row of the table, and are collectively named the *headers of rows* (or `rowHeaders`), (iii) the south-east block defines the `body` of the table, which typically contains the majority of numerical data, and (iv) the north-west block defines the `preamble` of the table, which is the intersecting area of `columnHeaders` and `rowHeaders`.

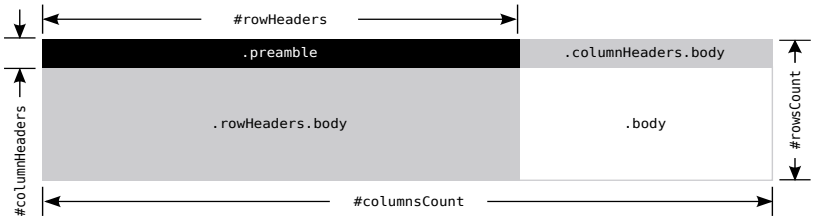

Figure 1: Illustration of annotated `Matrix` layout

Matrix Blocking

Given a `Matrix` M with `#columnHeaders` = `#rowHeaders` = 1.

$$M = \left[\begin{array}{c|ccc} A & B & C & D \\ \hline x & 1 & 2 & 3 \\ y & 4 & 5 & 6 \\ z & 7 & 8 & 9 \end{array}\right]$$

The blocks, or *sub-matrices*, of M are defined as follows,

`.preamble`:

$$M.\texttt{preamble} = [A]$$

`.columnHeaders`:

$$M.\texttt{columnHeaders} = \left[\begin{array}{c|ccc} A & B & C & D \end{array}\right]$$

`.rowHeaders`:

$$M.\texttt{rowHeaders} = \left[\begin{array}{c} A \\ \hline x \\ y \\ z \end{array}\right]$$

`.body`:

$$M.\texttt{body} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

The four-block layout of `Matrix` exhibits certain degree of self-similarity. Namely, if we view the `columnHeaders` as a sub-`Matrix`, then the `preamble` of the full `Matrix` becomes the `rowHeaders` of the sub-`Matrix`, i.e.,

$$M.\texttt{columnHeaders.rowHeaders} = M.\texttt{preamble} = [A]$$

Likewise for the `rowHeaders`, the `preamble` of the full `Matrix` can also be viewed as the `columnHeaders` of the sub-`Matrix`, i.e.,

$$M.\texttt{rowHeaders.columnHeaders} = M.\texttt{preamble} = [A]$$

Following this manner, the north-east block of the full `Matrix` is the `body` of the `columnHeaders` (or `columnHeaders.body`); and the south-west block of the full `Matrix` is the `body` of the `rowHeaders` (or `rowHeaders.body`), i.e.,

$$M.\texttt{columnHeaders.body} = \begin{bmatrix} B & C & D \end{bmatrix}$$

$$M.\texttt{rowHeaders.body} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Matrix Striding

Striding is the iterative traversal of *row vectors* from a `Matrix`. Data processing functions and arithmetic operators can be applied to the `.rows` of a `Matrix`.

`.rows`:

$$M.\texttt{rows} = \left[\begin{array}{cccc} x & 1 & 2 & 3 \\ \hline y & 4 & 5 & 6 \\ \hline z & 7 & 8 & 9 \end{array}\right]$$

To access data in a `Matrix` on *column* basis, one should first obtain the *transpose*, i.e., `.T`, of the `Matrix`, then access it's `.rows`, i.e.,

`.T`:

$$M.\texttt{T.rows} = \left[\begin{array}{cccc} B & 1 & 4 & 7 \\ \hline C & 2 & 5 & 8 \\ \hline D & 3 & 6 & 9 \end{array}\right]$$