graphtransliterator-js

Release 0.5.1

A. Sean Pue

May 15, 2020

CONTENTS

A partial Javascript/Node implementation of Graph Transliterator, graph-based transliteration tool that lets you convert the symbols of one language or script to those of another using rules that you define.

- Free software: MIT license
- Documentation: https://graphtransliterator-js.readthedocs.io
- NPM: https://www.npmjs.com/package/graphtransliterator
- Repository: https://github.com/seanpue/graphtransliterator-js

CHAPTER

ONE

TRANSLITERATION... WHAT? WHY?

Moving text or data from one script or encoding to another is a common problem:

- Many languages are written in multiple scripts, and many people can only read one of them. Moving between them can be a complex but necessary task in order to make texts accessible.
- The identification of names and locations, as well as machine translation, benefit from transliteration.
- Library systems often require metadata be in particular forms of romanization in addition to the original script.
- Linguists need to move between different methods of phonetic transcription.
- Documents in legacy fonts must now be converted to contemporary Unicode ones.
- Complex-script languages are frequently approached in natural language processing and in digital humanities research through transliteration, as it provides disambiguating information about pronunciation, morphological boundaries, and unwritten elements not present in the original script.

Graph Transliterator abstracts transliteration, offering an "easy reading" method for developing transliterators that does not require writing a complex program. It also contains bundled transliterators that are rigorously tested. These can be expanded to handle many transliteration tasks.

Graph Transliterator Javascript provides **access to Graph Transliterator's bundled transliterators**, as well as **any JSON-dumped graph transliterator**.

CHAPTER

TWO

FEATURES

Graph Transliterator Javascript provides:

- a partial Javascript/Node implementation of Graph Transliterator (a Python library and CLI)
- bundled transliterators from Graph Transliterator
- processing of the JSON dump of a Graph Transliterator
- convenient client-side Javascript libraries

CHAPTER

THREE

INSTALLATION

Graph Transliterator Javascript is a Node.js module. It can be installed using `npm`:

```
$ npm install --save graphtransliterator
```

It can also be used independently as a client-side Javascript library.

CHAPTER

FOUR

CONTENTS

4.1 Usage

Graph Transliterator Javascript is accessed via the `graphtransliterator` module. It can load bundled transliterators or custom graph transliterators dumped as JSON.

4.1.1 Bundled Transliterators

Graph Transliterator Javascript contains a number of bundled transliterators that match up with those provided by Graph Transliterator. New contributions to the bundled transliterators are welcome. Please see the Graph Transliterator documentation on how to bundle a transliterator.

Server-side Usage

Graph Transliterator Javascript can be run server-side, and it includes all bundled transliterators:

```
var graphtransliterator = require("graphtransliterator");
// Includes bundled transliterators
var transliterator = graphtransliterator.transliterators.ITRANSDevanagariToUnicode;
transliterator.transliterate("namaskAr")
```

```
""
```

Client-side Usage

Graph Transliterator can also be accessed as a client-side Javascript library (`graphtransliterator.js`) that loads the Javascript library `graphtransliterator` and contains bundled transliterators in `graphtransliterator.transliterators`:

```
<script src="https://unpkg.com/graphtransliterator/dist/graphtransliterator.js"></
↪script>
<script>
    var gt = graphtransliterator.transliterators.ITRANSDevanagariToUnicode;
    console.log(
      gt.transliterate("namaste")
    );
</script>
```

```
""
```

Each bundled transliterator can be accessed as a stand-alone Javascript library (e.g., `graphtransliterator.Example.js`):

```
<script src="https://unpkg.com/graphtransliterator/dist/GraphTransliterator.
↪ITRANSDevanagariToUnicode.js"></script>
<script>
    console.log(ITRANSDevanagariToUnicode.transliterate("praNAm"));
</script>
```

```
""
```

4.1.2 JSON Graph Transliterators

Graph Transliterator, a Python library and CLI, supports configuration using an "easy-reading" YAML format for entering transliteration rules:

```
tokens:
  a: [vowel]                # type of token ("a") and its class (vowel)
  bb: [consonant, b_class] # type of token ("bb") and its classes (consonant, b_class)
  ' ': [wb]                 # type of token (" ") and its class ("wb", for wordbreak)
rules:
  a: A        # transliterate "a" to "A"
  bb: B       # transliterate "bb" to "B"
  a a: <2AS> # transliterate ("a", "a") to "<2AS>"
  ' ': ' '    # transliterate ' ' to ' '
whitespace:
  default: " "         # default whitespace token
  consolidate: false  # whitespace should not be consolidated
  token_class: wb      # whitespace token class
```

Graph Transliterator Javascript does not support YAML input. It can only read **JSON dumped settings**. See the CLI command graphtransliterator dump or the Python API's GraphTransliterator.dumps().

The above example would be dumped using a simple compression as follows:

```
{"graphtransliterator_version":"1.2.0","compressed_settings":[["b_class","consonant",
↪"vowel","wb"],[" ","a","bb"],[[3],[2],[0,1]],[["<2AS>",0,0,[1,1],0,0,-2],["A",0,0,
↪[1],0,0,-1],["B",0,0,[2],0,0,-1],[" ",0,0,[0],0,0,-1]],[" ","wb",0],0,{},null]}
```

Server-Side Loading from JSON

To load from the server, create a new `GraphTransliterator`:

```
{ GraphTransliterator } = require("graphtransliterator");
// The dumped settings are the output of ``graphtransliterator dump -f bundled␣
↪Example``
var gt = GraphTransliterator(
  {"graphtransliterator_version":"1.2.0","compressed_settings":[["b_class","consonant
↪","vowel","wb"],[" ","a","bb"],[[3],[2],[0,1]],[["<2AS>",0,0,[1,1],0,0,-2],["A",0,0,
↪[1],0,0,-1],["B",0,0,[2],0,0,-1],[" ",0,0,[0],0,0,-1]],[" ","wb",0],0,{},null]});
);
gt.transliterate("a");
```

Client-Side Loading from JSON

The Graph Transliterator class (`graphTransliterator.GraphTransliterator`), without bundled transliterators, is available from the main library (`graphtransliterator.js`).

The class, without the bundled transliterators, is distributed as `graphtransliterator.GraphTransliterator.js`:

```
<script src="https://unpkg.com/graphtransliterator/dist/graphtransliterator.
↪Graphtransliterator.js"></script>
<script>
    // The dumped settings are the output of ``graphtransliterator dump -f bundled␣
↪Example``
    var settings = {"graphtransliterator_version":"1.2.0","compressed_settings":[[
↪"consonant","vowel","whitespace"],[" ","a","b"],[[2],[1],[0]],[["!B!",[0],[1],[2],
↪[1],[0],-5],["A",0,0,[1],0,0,-1],["B",0,0,[2],0,0,-1],[" ",0,0,[0],0,0,-1]],[" ",
↪"whitespace",0],[[[1],[1],","]],{"name":"example","version":"1.0.0","description":
↪"An Example Bundled Transliterator","url":"https://github.com/seanpue/
↪graphtransliterator/tree/master/transliterator/sample","author":"Author McAuthorson
↪","author_email":"author_mcauthorson@msu.edu","license":"MIT License","keywords":[
↪"example"],"project_urls":{"Documentation":"https://github.com/seanpue/
↪graphtransliterator/tree/master/graphtransliterator/transliterators/example","Source
↪":"https://github.com/seanpue/graphtransliterator/tree/graphtransliterator/
↪transliterators/example","Tracker":"https://github.com/seanpue/graphtransliterator/
↪issues"}},null]};
    var gt = graphtransliterator.GraphTransliterator(settings);
    console.log(
       gt.transliterate("a")
    );
</script>
```

4.2 API Reference

A list of the full API reference of all public classes and functions is below.

4.2.1 Core Classes

class GraphTransliterator(*settings*)

Create a GraphTransliterator.

Arguments

- **settings** (*Object*) –

GraphTransliterator.**isWhitespace**(*token*)

Check if a token is whitespace.

Returns **boolean** –

GraphTransliterator.**lastMatchedRuleTokens**

Get the last tokens matched.

GraphTransliterator.**lastMatchedRules**

Get the last rules matched.

GraphTransliterator.**matchAllAt**(*tokenIdx*, *tokens*)

Match all tokens at a particular index.

Arguments

- **tokenIdx** (*number*) –
- **tokens** (*Array*) –

Returns **undefined|Array** – List of rule indexes

GraphTransliterator.**matchAt**(*tokenIdx*, *tokens*, *matchAll=false*)

Match best (least costly) transliteration rule at a given index in the input tokens and return the index to that rule. Optionally, return all rules that match.

Arguments

- **tokenIdx** (*number*) – Location in *tokens* at which to begin
- **tokens** (*Array*) – List of strings of tokens
- **matchAll** (*boolean*) – If true, return the index of all rules matching at the given index. The default is false.

Returns (undefined|number|Array) - Index of rule matched or list of rules matched

GraphTransliterator.**tokenize**(*input*)

Tokenize input string.

Arguments

- **input** (*string*) – Input string

Returns **Array** – - match details

GraphTransliterator.**transliterate**(*input*)

Transliterate an input string into an output string.

Whitespace will be temporarily appended to start and end of input string.

Arguments

- **input** (*string*) –

Throws ***UnrecognizableInputTokenError*** –

Returns **string** – Transliterated input string.

GraphTransliterator.**fromDict**(*dictSettings*)

Create a GraphTransliterator from settings. (From Python implementation, can be removed.)

Arguments

- **dictSettings** (*object*) – Compressed on decompressed settings.

Returns **GraphTransliterator** –

class DirectedGraph(*edge*, *node*, *edge_list*)

DirectedGraph

Graph data structure used in Graph Transliterator.

Arguments

- **edge** (*object*) – Mapping from head to tail of edge, holding edge data
- **node** (*Array*) – Array of node attributes
- **edge_list** (*Array*) – Array of head and tail of each edge

DirectedGraph.**addEdge**(*head*, *tail*, *edgeData*)

Add new edge.

Arguments

- **head** (*number*) – Index of head of edge
- **tail** (*number*) – Index of tail of edge
- **edgeData** (*Object*) – Attributes of edge

Returns **Object** – - Reference to new edge

DirectedGraph.**addNode**(*nodeData*)

Arguments

- **nodeData** (*object*) – Attributes for node

Returns **Array.<number, number>** – - Index of new node

4.2.2 Bundled Transliterators

class Example()
Example transliterator

class ITRANSDevanagariToUnicode()
ITRANSDevanagariToUnicode transliterator

4.2.3 Errors

class GraphTransliteratorError()
Base Graph Transliterator error.

class NoMatchingTransliterationRuleError()
Graph Transliterator no matching transliteration rule error.

class UnrecognizableInputTokenError()
Graph Transliterator unrecognizable token error.

4.3 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

4.3.1 Contributor Code of Conduct

Please note that this project is released with a Contributor Code of Conduct. By participating in this project you agree to abide by its terms.

4.3.2 Types of Contributions

You can contribute in many ways:

Report Bugs

Report bugs at https://github.com/seanpue/graphtransliterator-js/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

Graph-based Transliterator could always use more documentation, whether as part of the official Graph Transliterator docs, in docstrings, or even on the web in blog posts, articles, and such. Documentation is generated using sphinx-js.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/seanpue/graphtransliterator-js/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Add Transliterators

We welcome new transliterators to be added to the bundled transliterators!

However, these should be added to Graph Transliterator, not Graph Transliterator Javscript. See the Graph Transliterator documentation on how to add a transliterator.

4.3.3 Get Started!

Ready to contribute? Here's how to set up `graphtransliterator-js` for local development.

1. Fork the `graphtransliterator-js` repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/graphtransliterator-js.git
```

3. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. Run the tests:

```
$ npm run test
```

   This will automatically generate coverage. Check that your changes are covered:

```
$ make coverage
```

5. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

4.3.4 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Node 12, 13, and 14. Check https://travis-ci.org/seanpue/graphtransliterator-js/pull_requests and make sure that the tests pass for all supported Python versions.

4.3.5 Deploying

Here is a reminder for the maintainers on how to deploy a new version:

```
$ npm run update-transliterators
$ npm version major
$ git push --follow-tags
$ npm publish
```

4.4 Credits

4.4.1 Development Lead

- A. Sean Pue @seanpue <pue@msu.edu>

4.5 Acknowledgements

Software development was supported by an Andrew W. Mellon Foundation New Directions Fellowship (Grant Number 11600613) and by matching funds provided by the College of Arts and Letters, Michigan State University.

4.6 Change Log

4.6.1 [Unreleased - Maybe]

- Add flag for logging full errors or just descriptive messages
- Add multiple JS core versions (?)
- Add compression functions
- remove stripEmpty calls from compress.js
- Add pre- and post-transliteration hooks
- move build transliterators to git submodule
- implement jupyter_sphinx(already included in conf.py) with node kernel in docs
- consider whether or not to instantiate bundled transliterators

4.6.2 [Unreleased - To Do]

- Make sure if works in Vue, etc.
- Finish documentation
- Fix serialization discrepencies whereby bundled transliterator JSON is not exact matching

4.6.3 0.6.2 (05-15-2020)

- Fixed documentation
- Shifted to production in dist

4.6.4 0.6.1 (05-14-2020)

- added dist to files in package.json
- added build script to package.json

4.6.5 0.6.0 (05-14-2020)

- rebuilt from scratch
- updated updatedTransliterators.js
- added settings and tests to bundled transliterators
- confirmed webpack for individual transliterators

4.6.6 0.5.1 (04-29-2020)

- updated *update_transliterators.js* with jsdoc strings generated for transliterators
- Removed "git add" from package.json for lint-staged
- added .eslintconfig as eslint was getting stuck
- added *scripts/update_transliterators.js* script to generate transliterators/index.js and docs/transliterators.inc to sync bundled transliterators with *graphtransliterator* using its CLI
- removed bundled transliterators' index.js and surfacing from transliterators/index.js
- Got js-sphinx working
- Experiments with jsdoc and js-sphinx, following some issues with bundled transliterators and jsdoc namespace.
- Added basic documentation to javascript.

4.6.7 0.5.0 (04-21-2020)

- disabled stripEmpty() in compress.js; will likely remove
- added "jest–coverage &&" to .travis.yml after_script to provide coverage info to coveralls
- removed transliterator directories with differently cased names remaining on github
- rewrote transliterators/index.js with some struggle due to file name errors on travis due
- wrote scripts/updateTransliterators.js and changed bundled transliterator naming format
- Added decompressSettings in compress.js
- Updated bundled transliterators with faster (less to download and quicker to load than expanded JSON) compressed versions

4.6.8 0.4.2 (01-11-2020)

- removed lib/__tests__ from dist

4.6.9 0.4.1 (01-10-2020)

- adjusted node engine requirement in package.json
- fixed files setting in package.json to include lib

4.6.10 0.4.0 (01-10-2020)

- Removed esm support due to difficulty configuring to work with jest
- Added support for compressed graphs
- Added graph creation as fromGraph(), as well as onmatchRulesLookupOf(), tokensByClassOf()
- Added esm for ecmascript management

4.6.11 0.3.0 (12-13-2019)

- Adjusted webpack.config.js to generate transliterators with babel
- Added update script to copy graphtransliterator transliterators into transliterators
- Added webpack yielding dist/GraphTranliterator.node.js and dist/GraphTransliterator.
- Added babel to client config converting from ES6 to CoreJS 3.0

4.6.12 0.2.0 (12-10-2019)

- Added matchAllAt() to GraphTransliterator
- Added console logging of error message or throwing of errors if ignoreErrors is false.
- Added NoMatchingTransliterationRuleError, UnrecognizableInputError, GraphTransliteratorError
- Changed from Python version by switching to details from tokenize(), including position in string, unrecognizable characters, whitespace
- Fixed capitalization in index.js for Travis CI
- Added tests for coverage of all transliteration so far
- Implemented basic transliteration functionality from detailed JSON
- Added lib/__tests__/graphtransliterator.test.js
- Added lib/__tests__/test_config.json and test_config.yaml
- Restored afterscript to travis.yml and removed script from package.json
- Added GraphTransliterator.js
- Began constructor for GraphTransliterator

4.6.13 0.1.0 (12-03-2019)

- Added coveralls script to package.json
- Moved afterscript to script in travis.yml following coveralls docs
- Added .coveralls.yml (locally)
- Added travis status badge
- Restricted to node >= 12.0.0 in package.json
- Removed pre-12 versions of node from .travis.yml
- Added HISTORY.md
- Initialized using yeoman node [https://github.com/yeoman/generator-node]

CHAPTER

FIVE

BACK MATTER

- genindex
- search

INDEX