EYE JS: A client-side reasoning engine supporting Notation3, RDF Surfaces and RDF Lingua

Wright, Jesse[1], Roo, Jos De[2] and Smessaert, Ieben[2]

[1]Computer Science Department, University of Oxford, UK
[2]IDLab, Department of Electronics and Information Systems, Ghent University - imec, Ghent, Belgium

Abstract

The Web is transitioning away from centralised services to a re-emergent decentralised platform. This movement generates demand for infrastructure that hides the complexities of decentralisation so that Web developers can easily create rich applications for the next generation of the internet.

This paper introduces EYE JS, an RDFJS-compliant TypeScript library that supports reasoning using Notation3 and RDF Surfaces from browsers and NodeJS.

By developing EYE JS, we fill a gap in existing research and infrastructure, creating a reasoning engine for the Resource Description Framework (RDF) that can reason over decentralised documents in a Web client.

Keywords

Notation3, Reasoner, Inference, RDFJS, RDF, JavaScript, TypeScript, N3, Notation3, RDF Surfaces, Web, Browser, WASM, WebAssembly, Solid

1. Introduction

Modern Web infrastructure is predominantly built in a centralised manner, hosted by a small number of influential organisations such as Meta, Amazon, Netflix and Google [1]. In contrast, Tim Berners-Lee's original vision for the Web was to create a democratic, distributed platform on which 'anyone can say anything about anything' [2]. Globally, there is a growing interest [3, 4] in data privacy and sovereignty on the Web, largely driven by problems around data exploitation [5, 6] and censorship [7] that arise within the centralised model.

Consequently, the Web ecosystem is moving back towards a decentralised model, as seen through the growing uptake of Linked Data [8] and the Solid protocol [9, 10]. These protocols build on the Semantic Web [11] technology stack.

Accordingly, researchers are increasingly seeking to develop ways for applications to interface with a decentralised Semantic Web. Front-end developers need tooling that is as *accessible* [12], *intuitive* and *efficient* to use as traditional APIs for accessing centralised resources [13, 14]. Until now, decentralisation research has focused on interoperable discovery and querying of Semantic Web data [15]. There existed a fundamental gap in the research of the enrichment of decentralised data on the Semantic Web, through the addition of implicit facts into the set of query results.

Posters, Demos, and Industry Tracks at ISWC 2024, November 13–15, 2024, Baltimore, USA
jesse.wright@cs.ox.ac.uk (W. Jesse); Jos.DeRoo@UGent.be (R. Jos De); ieben.smessaert@ugent.be (S. Ieben)
https://www.cs.ox.ac.uk/people/jesse.wright/ (W. Jesse); https://smessaert.be (S. Ieben)
0000-0002-5771-988X (W. Jesse); 0000-0001-8862-0666 (R. Jos De); 0009-0004-5281-0723 (S. Ieben)

In many centralised databases containing real-world Semantic Web data, implicit facts comprise a sizeable portion of the database. For instance, in the UniProt [16] knowledge graph of protein sequences, 46.1% of the 228 million facts within the database are implicit. Further, in the Classical Art Research Online Service (CLAROS) In OpenCyc [17], an ontology of general human knowledge, the effect is even more pronounced; 99.8% of the 1176 million ABox facts in the database are generated by rules [18].Given the importance of implicit data to centralised Semantic Web applications, there is an immense opportunity for decentralised applications to have similar reasoning functionality. To enable this, client-side Resource Description Framework (RDF) reasoning is an emerging area of research [19], which can unlock a wide range of IoT applications; ranging from user-empowered social media [20] to live health diagnoses [21]. However, the limited work on client-side RDF reasoning [19] has failed to produce a reasoner that meets the *performance*, *interoperability* and *usability* requirements for many of these applications.

2. Background

2.1. Notation3, RDF Surfaces and RDF Lingua

Notation3 (N3) [22] is a superset of RDF 1.1 Turtle [23] used to state assertions and logical rules. RDF Lingua (https://eyereasoner.github.io/lingua/) introduces a set of semantics to RDF 1.1 Trig [24] to allow the expression of any statement or rule expressible in Notation3. RDF Surfaces (https://w3c-cg.github.io/rdfsurfaces/) [25] is a Notation3 sublanguage supporting first-order-logic semantics.

2.2. EYE and EYE Lingua

EYE (https://eulersharp.sourceforge.net/) [26, 27] is a reasoning engine supporting the Semantic Web layers - implementing Notation3 and RDF Surfaces. EYE Lingua (https://github.com/eyereasoner/lingua) [28] is a reasoning engine implementing RDF Lingua. EYE and EYE Lingua are both written in Prolog [29].

3. Architecture

SWI-Prolog [30] is a Prolog interpreter used to execute programs written in Prolog.

The authors of EYE JS, in collaboration with the authors of SWI-Prolog [30] developed the swipl-wasm npm package which uses Emscripten [31] to compile SWI-Prolog [30] into JavaScript and WebAssembly code. swipl-wasm also includes type declarations for the exported `SWIPL` functions. There is CI in place for swipl-wasm which cuts a new release whenever SWI-Prolog [30], Emscripten [31], Zlib or PCRE2 are updated, ensuring that a release of swipl-wasm compiled from the latest source codes is available.

EYE JS uses swipl-wasm to compile the EYE and EYE Lingua prolog code into PVM images as part of the build pipeline. These images are bundled with the EYE JS package as Base64 encoded string constants within JavaScript files ranging from 400-500kb in size. At run time, these images are used to initialise EYE and EYE Lingua in swipl-wasm with low memory and execution time overheads. There is CI in place for EYE JS which cuts a new release whenever

swipl-wasm, EYE or EYE Lingua are updated, ensuring that a release of EYE JS compiled from the latest source codes is available. At the time of writing, there are 577 releases.

4. Adoption

EYE JS (https://github.com/eyereasoner/eye-js/) [32] is currently available for all versions of NodeJS greater than or equal to 18.x and works in all major browsers. EYE JS has been well received by the community, and is being used in a range of research applications [33, 34, 35, 36, 37, 38, 39, 40, 41, 42], it also has 32 dependent GitHub repositories, 5 dependent GitHub packages and 33 stars at the time of writing.

EYE JS is distributed on npm, published on Zenodo, available as a GitHub release, offered as a browser build that can be used directly in HTML script tags, and served as a JavaScript bundle that can be dynamically imported into scripts. EYE JS implements the RDF JS data model specification [43] out of the box, enabling immediate interoperability with other applications and libraries that use this data model.

5. Usage

5.1. Live Deployments

Several websites have been deployed for interfacing with EYE JS. https://eyereasoner.github.io/eye-js/example/ is a minimal HTML page that showcases how EYE JS can be deployed in 28 lines of HTML using the browser builds. https://reasoner.smessie.com/ supports both in-browser reasoning with EYE JS and server-side reasoning with EYE. Users can reason over data directly entered into the GUI and remote resources referenced via a URL. In the latter case, users have the option of authenticating with a WebID, thereby enabling reasoning over private documents from a Solid pod. This demonstrates how EYE JS empowers users to securely reason over private data without relying on third-party remote services. The ability to switch between in-browser (EYE JS) and server-side (EYE) reasoning is made possible by EYE JS implementing an ad hoc interface standard for JavaScript reasoners. https://editor.notation3.org/ also offers the ability to use EYE JS as a reasoning engine, alongside server-side reasoning engines.

5.2. Development

The EYE JS README (https://github.com/eyereasoner/eye-js?tab=readme-ov-file) provides developer documentation for using EYE JS in applications.

6. Performance Results and Conclusion

Benchmark.js provides performance benchmarking for each commit on the main branch of EYE JS using. Benchmarks are executed on an Ubuntu GitHub runner with the default 7GB of RAM. These performance tests for each release of EYE JS are available at https://eyereasoner.github.io/eye-js/dev/bench/. We report the results for the most recent commit at the time of writing

Depth	NodeJS	Chrome	Firefox	eye (native)	cwm	hermit	jdrew	jena
10^1	0.082	0.204	0.104	0.022	0.160	0.055	0.130	0.047
10^2	0.402	0.762	0.672	0.028	1.05	1.04	0.200	0.422
10^3	3.22	7.32	5.21	0.066	65.9	3.58	0.870	9.30
10^4	28.9	69.5	49.5	0.484	7300	310	18.7	2600
10^5	303	691	518	4.62	732e3	OutOfMem	1860	OutOfMem

Figure 1: Time taken (seconds) to execute the Deep Taxonomy Benchmark. The results for NodeJS (v20.11.0), Chrome (126.0.6478.114) and Firefox (127.0.1) were obtained using v16.15.12 of EYE JS on a Dell XPS 9520 machine running Ubuntu 23.10 with 32GB of RAM. The files needed to reproduce these experiments can be found at https://github.com/eyereasoner/eye-js/tree/main/perf. The experiments for eye (native), cwm, hermit, jdrew and jena were performed on a machine with 4 2.40GHZ cores and 256 GB of memory (https://eulersharp.sourceforge.net/2003/03swap/dtb-note).

which is 2083053 where Node 22 was used to execute the benchmarks. It takes 66±4.27%ms to initialise the reasoner and 16±16.98%ms to execute the Socrates Query meaning that users will perceive reasoning over a handful of facts as instantaneous [44]. Applying RDFS inference to Tim Berners-Lee's profile card and the FOAF ontology (14 rules, 961 facts, 866 derivations) takes 893±1.31%ms which is the time frame within which a users flow of thought will remain uninterrupted [44].

As shown in Figure 1, EYE JS outperforms cwm, hermit, jdrew and jena on the Deep Taxonomgy Benchmark in all of the environments in which it was run - even though the NodeJS environment in which it was fastest is single-threaded and limited to using 512MB of memory whilst the other reasoners had access to 4 cores and 215GB of memory.

These results show that even though EYE JS is slower than EYE due to resource constraints in browser and NodeJS environments, EYE JS is still performant enough to be deployed in the browser for real-world use cases. The adoption of EYE JS across a range of applications evidences this.

Acknowledgements

Jesse Wright is funded by the Department of Computer Science, University of Oxford. Jos De Roo and Ieben Smessaert are funded by SolidLab Vlaanderen (Flemish Government, EWI and RRF project VV023/10).

References

[1] P. De Filippi, S. McCarthy, Cloud computing: Centralization and data sovereignty, European Journal of Law and Technology 3 (2012).

[2] T. Berners-Lee, Long live the web, Scientific American 303 (2010) 80–85.

[3] P. Voigt, A. Von dem Bussche, The eu general data protection regulation (gdpr), A Practical Guide, 1st Ed., Cham: Springer International Publishing 10 (2017) 10–5555.

[4] 2018 reform of eu data protection rules, 2018. URL: https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pdf.

[5] J. Hinds, E. J. Williams, A. N. Joinson, "it wouldn't happen to me": Privacy concerns and perspectives following the cambridge analytica scandal, International Journal of Human-Computer Studies 143 (2020) 102498.

[6] I. Chaston, et al., Internet marketing and big data exploitation, Springer, 2015.

[7] F. Stjernfelt, A. M. Lauritzen, Facebook and google as offices of censorship, in: Your Post has been Removed, Springer, 2020, pp. 139–172.

[8] C. Bizer, T. Heath, K. Idehen, T. Berners-Lee, Linked data on the web (ldow2008), in: Proceedings of the 17th international conference on World Wide Web, 2008, pp. 1265–1266.

[9] S. Capadisli, T. Berners-Lee, R. Verborgh, K. Kjernsmo, Solid protocol, URL https://solid-project.org/TR/2021/protocol-20211217 (2021).

[10] E. Mansour, A. V. Sambra, S. Hawke, M. Zereba, S. Capadisli, A. Ghanem, A. Aboulnaga, T. Berners-Lee, A demonstration of the solid platform for social web applications, in: Proceedings of the 25th international conference companion on world wide web, 2016, pp. 223–226.

[11] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Scientific american 284 (2001) 34–43.

[12] R. Verborgh, R. Taelman, Ldflex: A read/write linked data abstraction for front-end web developers, in: International Semantic Web Conference, Springer, 2020, pp. 193–211.

[13] R. Verborgh, Reflections of knowledge, https://ruben.verborgh.org/blog/2021/12/23/reflections-of-knowledge/, 2021. [Accessed 23-06-2024].

[14] C. Pautasso, E. Wilde, Why is the web loosely coupled? a multi-faceted metric for service design, in: Proceedings of the 18th international conference on World wide web, 2009, pp. 911–920.

[15] R. Taelman, J. Van Herwegen, M. Vander Sande, R. Verborgh, Comunica: a modular sparql query engine for the web, in: Proceedings of the 17th International Semantic Web Conference, 2018. URL: https://comunica.github.io/Article-ISWC2018-Resource/.

[16] U. Consortium, Uniprot: a hub for protein information, Nucleic acids research 43 (2015) D204–D212.

[17] D. Foxvog, Cyc, Theory and applications of ontology: Computer applications (2010) 259–278.

[18] B. Motik, Y. Nenov, R. Piro, I. Horrocks, Handling owl:sameas via rewriting, Proceedings of the AAAI Conference on Artificial Intelligence 29 (2015). URL: https://ojs.aaai.org/index.php/AAAI/article/view/9187. doi:10.1609/aaai.v29i1.9187.

[19] M. Terdjimi, L. Médini, M. Mrissa, Hylar: Hybrid location-agnostic reasoning, in: ESWC Developers Workshop 2015, 2015, p. 1.

[20] J. Werbrouck, P. Pauwels, J. Beetz, L. van Berlo, Towards a decentralised common data environment using linked building data and the solid ecosystem, in: 36th CIB W78 2019 Conference, 2019, pp. 113–123.
[21] T. Sondes, H. B. Elhadj, L. Chaari, An ontology-based healthcare monitoring system in the internet of things, in: 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), IEEE, 2019, pp. 319–324.
[22] T. Berners-Lee, Notation3, http://www.w3.org/DesignIssues/Notation3.html (1998).
[23] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, G. Carothers, Rdf 1.1 turtle, World Wide Web Consortium (2014).
[24] G. Carothers, A. Seaborne, C. Bizer, R. Cyganiak, Rdf 1.1 trig, World Wide Web Consortium (2014).
[25] P. Hochstenbach, J. De Roo, R. Verborgh, Rdf surfaces: Computer says no, arXiv preprint arXiv:2305.08476 (2023).
[26] R. Verborgh, J. De Roo, Drawing conclusions from linked data on the web: The eye reasoner, IEEE Software 32 (2015) 23–27.
[27] J. D. Roo, P. Hochstenbach, J. Wright, B. D. Vloed, J. Wielemaker, R. Verborgh, P. Moura, hongsun502, eyereasoner/eye: v10.19.7, 2024. URL: https://doi.org/10.5281/zenodo.13369975. doi:10.5281/zenodo.13369975.
[28] J. D. Roo, J. Wright, eyereasoner/lingua: v1.2.3, 2024. URL: https://doi.org/10.5281/zenodo.13370911. doi:10.5281/zenodo.13370911.
[29] A. Colmerauer, An introduction to prolog iii, Communications of the ACM 33 (1990) 69–90.
[30] J. Wielemaker, T. Schrijvers, M. Triska, T. Lager, Swi-prolog, Theory and Practice of Logic Programming 12 (2012) 67–96.
[31] A. Zakai, Emscripten: an llvm-to-javascript compiler, in: Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion, 2011, pp. 301–312.
[32] J. Wright, J. D. Roo, I. Smessaert, P. Hochstenbach, W. Slabbinck, eyereasoner/eye-js: v16.15.11, 2024. URL: https://doi.org/10.5281/zenodo.12211024. doi:10.5281/zenodo.12211024.
[33] I. Smessaert, P. Hochstenbach, B. De Meester, R. Taelman, R. Verborgh, Dfdp: A declarative form description pipeline for decentralizing web forms, in: The Second International Workshop on Semantics in Dataspaces, co-located with the Extended Semantic Web Conference, 2024.
[34] R. Zhao, J. Zhao, Perennial semantic data terms of use for decentralized web, in: Proceedings of the ACM Web Conference 2024, WWW '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 2238–2249. URL: https://doi.org/10.1145/3589334.3645631. doi:10.1145/3589334.3645631.
[35] W. Van Woensell, S. Abidi, S. S. R. Abidi, Check for updates decentralized web-based clinical decision support using semantic glean workflows, in: Artificial Intelligence in Medicine: 21st International Conference on Artificial Intelligence in Medicine, AIME 2023, Portorož, Slovenia, June 12–15, 2023, Proceedings, volume 13897, Springer Nature, 2023, p. 362.
[36] W. Van Woensel, S. Abidi, S. S. R. Abidi, Decentralized web-based clinical decision support

using semantic glean workflows, in: International Conference on Artificial Intelligence in Medicine, Springer, 2023, pp. 362–367.
[37] W. Van Woensel, F. Scioscia, G. Loseto, O. Seneviratne, E. Patton, S. Abidi, Explanations of symbolic reasoning to effect patient persuasion and education, in: J. M. Juarez, C. Fernandez-Llatas, C. Bielza, O. Johnson, P. Kocbek, P. Larrañaga, N. Martin, J. Munoz-Gama, G. Štiglic, M. Sepulveda, A. Vellido (Eds.), Explainable Artificial Intelligence and Process Mining Applications for Healthcare, Springer Nature Switzerland, Cham, 2024, pp. 62–71.
[38] J.-P. Baert, Designing consensus-based semantic web applications with n3-reasoning and solid personal data vaults, 2023. URL: https://thesis.jan-pieter.be/.
[39] I. Smessaert, Solidlabresearch/formrenderer: v1.0.0, 2023. URL: https://doi.org/10.5281/zenodo.10285210. doi:10.5281/zenodo.10285210.
[40] I. Smessaert, ember tomster, Solidlabresearch/formgenerator: v1.0.0, 2023. URL: https://doi.org/10.5281/zenodo.10285192. doi:10.5281/zenodo.10285192.
[41] I. Smessaert, Solidlabresearch/formcli: v1.0.0, 2023. URL: https://doi.org/10.5281/zenodo.10285224. doi:10.5281/zenodo.10285224.
[42] W. Slabbinck, R. Dedecker, J. A. Rojas Meléndez, R. Verborgh, A rule-based software agent on top of personal data stores, in: Proceedings of the 22nd International Semantic Web Conference: Posters, Demos, and Industry Tracks, 2023.
[43] T. Bergwinkl, M. Luggen, elf Pavlik, B. Regalia, P. Savastano, R. Verborgh, RRDF/JS: Data model specification, W3C Community Group Draft Report, W3C, 2023. http://rdf.js.org/data-model-spec/.
[44] J. Nielsen, Response times: The three important limits, Usability Engineering (1993). URL: https://cir.nii.ac.jp/crid/1571698600977853312.