

1 ***Phylo-Node: a molecular phylogenetic toolkit using Node.js***

2

3

4 Damien M. O'Halloran<sup>1,2\*</sup>

5

6 1. Department of Biological Sciences, The George Washington University, Science and  
7 Engineering Hall, Rm 6000, 800 22nd Street N.W., Washington DC 20052, USA.

8 2. Institute for Neuroscience, The George Washington University, 636 Ross Hall, 2300 I  
9 St. N.W. Washington DC 20052, USA.

10

11

12

13 **\*Corresponding author address:**

14 Damien O'Halloran, The George Washington University, 636 Ross Hall, 2300 I St. N.W.  
15 Washington DC 20052, USA.

16 Tel: 202-994-8955

17 Fax: 202-994-6100

18 EMAIL: [damienoh@gwu.edu](mailto:damienoh@gwu.edu)

19

20

21

22

23

24 **ABSTRACT**

25 **Background:** Node.js is an open-source and cross-platform environment that provides  
26 a JavaScript codebase for back-end server-side applications. JavaScript has been used  
27 to develop very fast, and user-friendly front-end tools for bioinformatic and phylogenetic  
28 analyses. However, no such toolkits are available using Node.js to conduct  
29 comprehensive molecular phylogenetic analysis.

30 **Results:** To address this problem, I have developed, *Phylo-Node*, which was developed  
31 using Node.js and provides a fast, stable, and scalable toolkit that allows the user to go  
32 from sequence retrieval to phylogeny reconstruction. Phylo-Node can execute the  
33 analysis and process the resulting outputs from sequence alignment software such as  
34 MUSCLE and Clustal Omega, the primer design software, Primer3, the best-fit modeling  
35 software, Protest3, and the phylogeny reconstruction software, PhyML; Phylo-Node  
36 also provides very easy-to-use sequence retrieval options from NCBI. Furthermore,  
37 Phylo-Node provides simple integration and interoperation with other Node modules to  
38 develop workflows across multiple components and languages using Node inheritance  
39 patterns and a customized piping module to support the production of diverse pipelines.

40 **Conclusions:** Phylo-Node is open-source and freely available to all users without sign-  
41 up or login requirements. All source code and user guidelines are openly available at  
42 the GitHub repository: <https://github.com/dohalloran/Phylo-Node>

43

44 **Keywords:** Node.js, JavaScript, phylogenetics

45

46

## 47 **BACKGROUND**

48 The cost of whole genome sequencing has plummeted over the last decade and as a  
49 consequence, the demand for genome sequencing technology has risen significantly  
50 [1]. This demand has meant that producing large complex datasets of DNA and RNA  
51 sequence information is common in small research labs, and in terms of human health  
52 this boom in sequence information and precipitous drop in sequencing costs has had a  
53 direct impact in the area of personalized medicine [2-5]. However, once the sequence  
54 information becomes available, perhaps the greater challenge is then processing,  
55 analyzing, and interpreting the data. To keep pace with this challenge, the development  
56 of new, fast, and scalable software solutions are required to visualize and interpret this  
57 information.

58 JavaScript is a lightweight programming language that uses a web browser as its  
59 host environment. JavaScript is cross-platform and supported by all modern browsers.  
60 Because JavaScript is client-side, it is very fast, as it doesn't have to communicate with  
61 a server and wait for a response in order to run some code. Web browsers are  
62 ubiquitous and require no dependencies to deploy and operate, and so JavaScript  
63 represents an obvious solution for visualizing sequence information. Front-end  
64 developments using JavaScript have proven to be extremely efficient in providing fast,  
65 easy-to-use, and embeddable solutions for data analysis [6-14]. A very active  
66 community of developers (<http://www.biojs.io/>) provides diverse components for parsing  
67 sequence data types, data visualization, and bioinformatics analysis in JavaScript [6, 7,  
68 15-19].

69 Node.js provides server-side back-end JavaScript. Node.js is written in C, C++,  
70 and JavaScript and uses the Google Chrome V8 engine to offer a very fast cross-  
71 platform environment for developing server side Web applications. Node is a single-  
72 threaded environment, which means that only one line of code will be executed at any  
73 given time; however, Node employs non-blocking techniques for I/O tasks to provide an  
74 asynchronous ability, by using *callback* functions to permit the parallel running of code.  
75 Node holds much potential for the bioinformatic analysis of molecular data. A  
76 community of Node developers provides modules for bioinformatic sequence workflows  
77 (<http://www.bionode.io/>) which in time will likely parallel the BioJS community  
78 (<http://www.biojs.io/>) for the number of modules versus components. However, as of  
79 now there are no robust tools for phylogenetic analysis pipelines currently available  
80 using the Node.js codebase. To fill this void I have developed, *Phylo-Node*, which  
81 provides a Node.js toolkit that goes from sequence retrieval, to primer design, to  
82 alignment, to phylogeny reconstruction, all from a single toolkit (Figure 1). MolPhylo is  
83 fast, easy to use, and offers simple customization and portability options through  
84 various inheritance patterns. The Node package manager, *npm*  
85 (<https://www.npmjs.com/>), provides a very easy and efficient way to manage  
86 dependencies for any Node application. *Phylo-Node* is available at both GitHub  
87 (<https://github.com/dohalloran/Phylo-Node>) and npm  
88 (<https://www.npmjs.com/package/phylo-node>).

89

## 90 **IMPLEMENTATION**

91 *Phylo-Node* was developed using the Node.js codebase. The *Phylo-Node* core contains

92 methods for remote sequence retrieval, and phylogenetic analysis using a suite of  
93 popular software tools. A base wrapper object is used to prepare the arguments and  
94 directory prior to program execution; an individual software tool can be easily accessed  
95 and executed by importing the module for that tool so as to get access to the method  
96 properties on that object. These method properties are available to the user by using  
97 the 'module.exports' reference object. Inside a driver script file, the user can import the  
98 main module object properties and variables by using the 'require' keyword which is  
99 used to import a module in Node.js. The 'require' keyword is actually a global variable,  
100 and a script has access to its context because it is wrapped prior to execution inside the  
101 'runInThisContext' function (for more details, refer to the Node.js source code:  
102 <https://github.com/nodejs>). Once imported, the return value is assigned to a variable  
103 which is used to access the various method properties on that object. For example: a  
104 method property on the 'phym1' object is 'phym1.getphym1()', which invokes the  
105 'getphym1' method on the 'phym1' object to download and decompress the PhyML  
106 executable. For a complete list of all methods, refer to the 'README' file at the GitHub  
107 repository (<https://github.com/dohalloran/Phylo-Node/blob/master/README.md>). In  
108 order to correctly wrap and run each executable, new shells must be spawned so as to  
109 execute specific command formats for each executable. This was achieved by using  
110 'child.process.exec', which will launch an external shell and execute the command  
111 inside that shell while buffering any output by the process. Binary files and executables  
112 were downloaded and executed in this manner and the appropriate file and syntax  
113 selected by determining the user's operating system. Phylo-Node was validated on  
114 Microsoft Windows 7 Enterprise ver.6.1, MacOSX El Capitan ver.10.11.5, and Linux

115 Ubuntu 64-bit ver.14.04 LTS.

116

## 117 **RESULTS AND DISCUSSION**

118 Phylo-Node is a toolkit to interface with key applications necessary in building a  
119 phylogenetic pipeline (Figure 1). Firstly, Phylo-Node allows the user to remotely  
120 download sequences by building a unique URL and passing this string to the NCBI e-  
121 utilities API (<http://www.ncbi.nlm.nih.gov/books/NBK25501/>). Any number of genes can  
122 be supplied as command-line arguments to Phylo-Node by accessing the  
123 *fetch\_seqs.fasta* method on the *fetch\_seqs* object in order to retrieve sequence  
124 information in FASTA format. Phylo-Node also provides methods on specific objects to  
125 download various executable files using the 'download' module. Phylo-Node then  
126 provides a facility to execute the following programs: Primer3 [20-22] to facilitate primer  
127 design; Clustal Omega [23] and MUSCLE [24, 25] alignment executables for multiple  
128 sequence alignments; Prottest3 [31] to determine the best-fit model of evolution, and  
129 the PhyML executable [26, 27] for phylogeny reconstruction. Primer3 is the most  
130 popular software for primer design, and takes a very lengthy list of input variables to  
131 optimize primer selection. Clustal Omega and MUSCLE are very fast and accurate  
132 multiple sequence alignment tools that are commonly used to build robust DNA, RNA,  
133 or protein alignments. PhyML is a popular program for building phylogenies using  
134 maximum likelihood, and Prottest3 determines the best-fit model of evolution for protein  
135 sequences across 120 different potential models. Together, Phylo-Node provides a  
136 novel toolkit that allows the user to go from raw sequence to phylogeny using Node.  
137 Phylo-Node is highly scalable and customizable, and was inspired by projects

138 such as BioPerl [28] which provides Perl modules for many bioinformatic tasks and also  
139 provides parsers and wrappers for diverse sequence formats and applications. BioPerl's  
140 open source structure and architecture allows users to plug new modules into BioPerl  
141 pipelines to design new applications. Node.js implements prototypal inheritance as per  
142 JavaScript but also provides access to the 'module.exports' object which permits easy  
143 portability between the Phylo-Node toolkit and any other modules, and also  
144 interoperation between different languages by using the 'child.process.exec' process.  
145 Therefore, Phylo-Node can be integrated with existing Node.js bioinformatics tools [29,  
146 30] or software written in other languages. For example, Protttest3 requires a Java  
147 runtime environment ([http://www.oracle.com/technetwork/java/javase/downloads/jre8-](http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html)  
148 [downloads-2133155.html](http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html)), and by using 'require' to import the 'protttest' module, the  
149 user can execute the analysis of Protttest3. The 'protttest' module and driver script  
150 (index.js) are contained within the 'Protttest3' directory and sample input is provided in  
151 the 'COX2\_PF0016' sub-directory of the 'input\_examples' folder.

152 To further facilitate the ease of interoperation between various applications and  
153 components, the Phylo-Node package also contains a module called 'phylo-node\_pipes'  
154 inside the 'Pipes' directory. The 'phylo-node\_pipes' module allows the user to easily  
155 pipe data between different applications by requiring the 'child\_process' module which  
156 provides the ability to spawn child processes. Through 'phylo-node\_pipes' the user can  
157 chain commands together that will be executed in sequence to build consistent, and  
158 extensive pipelines (see the README.md file on GitHub for examples).

159

160

161 **CONCLUSIONS**

162 In conclusion, Phylo-Node is a novel package that leverages the speed of Node.js to  
163 provide a robust and efficient toolkit for researchers conducting molecular  
164 phylogenetics. Phylo-Node can be easily employed to develop complex but consistent  
165 workflows, and integrated with existing bioinformatics tools using the Node.js codebase.

166

167

168

169 **ACKNOWLEDGMENTS**

170 I thank members of the O'Halloran lab for critical reading of the manuscript, and would  
171 like to thank The George Washington University (GWU) Columbian College of Arts and  
172 Sciences, GWU Office of the Vice-President for Research, and the GWU Department of  
173 Biological Sciences for Funding.

174

175

176 **AVAILABILITY AND REQUIREMENTS**

- 177 • Project name: Phylo-Node
- 178 • Project home page: <https://github.com/dohalloran/phylo-node>
- 179 • Operating system(s): Platform independent
- 180 • Programming language: Node.js
- 181 • Other requirements: none
- 182 • License: MIT
- 183 • Any restrictions to use by non-academics: no restrictions or login requirements



184

185

186 **AUTHOR CONTRIBUTIONS**

187 D.O'H. conceived the idea for *Phylo-Node*, wrote and tested the code, and wrote the  
188 manuscript.

189

190 **COMPETING INTERESTS**

191 The author declares no competing interests.

192

193 **CONSENT FOR PUBLICATION**

194 Not applicable.

195

196 **ETHICS APPROVAL AND CONSENT TO PARTICIPATE**

197 Not applicable.

198

199

200

201

202

203

204

205

206

207

## References

- 208 1. Shaffer C: **Next-generation sequencing outpaces expectations.** Nat Biotechnol  
209 2007, **25**(2):149.
- 210 2. Wade N: **The quest for the \$1,000 human genome: DNA sequencing in the**  
211 **doctor's office? At birth? It may be coming closer.** N Y Times Web 2006, :F1, F3.
- 212 3. Mardis ER: **Anticipating the 1,000 dollar genome.** Genome Biol 2006, **7**(7):112.
- 213 4. Service RF: **Gene sequencing. The race for the \$1000 genome.** Science 2006,  
214 **311**(5767):1544-1546.
- 215 5. Hayden EC: **The \$1,000 genome.** Nature 2014, **507**(7492):294-295.
- 216 6. Yachdav G, Goldberg T, Wilzbach S, Dao D, Shih I, Choudhary S, Crouch S, Franz  
217 M, Garcia A, Garcia LJ, Gruning BA, Inupakutika D, Sillitoe I, Thanki AS, Vieira B,  
218 Villaveces JM, Schneider MV, Lewis S, Pettifer S, Rost B, Corpas M: **Anatomy of**  
219 **BioJS, an open source community for the life sciences.** Elife 2015,  
220 **4**:10.7554/eLife.07009.
- 221 7. Gomez J, Garcia LJ, Salazar GA, Villaveces J, Gore S, Garcia A, Martin MJ, Launay  
222 G, Alcantara R, Del-Toro N, Dumousseau M, Orchard S, Velankar S, Hermjakob H,  
223 Zong C, Ping P, Corpas M, Jimenez RC: **BioJS: an open source JavaScript**  
224 **framework for biological data visualization.** Bioinformatics 2013, **29**(8):1103-1104.

- 225 8. Salazar GA, Meintjes A, Mulder N: **PPI layouts: BioJS components for the display**  
226 **of Protein-Protein Interactions**. F1000Res 2014, **3**:50-50.v1. eCollection 2014.
- 227 9. Gomez J, Jimenez R: **Sequence, a BioJS component for visualising sequences**.  
228 F1000Res 2014, **3**:52-52.v1. eCollection 2014.
- 229 10. Cui Y, Chen X, Luo H, Fan Z, Luo J, He S, Yue H, Zhang P, Chen R: **BioCircos.js:**  
230 **an interactive Circos JavaScript library for biological data visualization on web**  
231 **applications**. Bioinformatics 2016, **32**(11):1740-1742.
- 232 11. Buels R, Yao E, Diesh CM, Hayes RD, Munoz-Torres M, Helt G, Goodstein DM,  
233 Elsik CG, Lewis SE, Stein L, Holmes IH: **JBrowse: a dynamic web platform for**  
234 **genome visualization and analysis**. Genome Biol 2016, **17**(1):66-016-0924-1.
- 235 12. Salavert F, Garcia-Alonso L, Sanchez R, Alonso R, Bleda M, Medina I, Dopazo J:  
236 **Web-based network analysis and visualization using CellMaps**. Bioinformatics  
237 2016, .
- 238 13. Franz M, Lopes CT, Huck G, Dong Y, Sumer O, Bader GD: **Cytoscape.js: a graph**  
239 **theory library for visualisation and analysis**. Bioinformatics 2016, **32**(2):309-311.
- 240 14. Vanderkam D, Aksoy BA, Hodes I, Perrone J, Hammerbacher J: **pileup.js: a**  
241 **JavaScript library for interactive and in-browser visualization of genomic data**.  
242 Bioinformatics 2016, .

- 243 15. Garcia L, Yachdav G, Martin MJ: **FeatureViewer, a BioJS component for**  
244 **visualization of position-based annotations in protein sequences.** F1000Res 2014,  
245 **3:47-47.v2.** eCollection 2014.
- 246 16. Kalderimis A, Stepan R, Sullivan J, Lyne R, Lyne M, Micklem G: **BioJS**  
247 **DAGViewer: A reusable JavaScript component for displaying directed graphs.**  
248 F1000Res 2014, **3:51-51.v1.** eCollection 2014.
- 249 17. Villaveces JM, Jimenez RC, Habermann BH: **KEGGViewer, a BioJS component**  
250 **to visualize KEGG Pathways.** F1000Res 2014, **3:43-43.v1.** eCollection 2014.
- 251 18. Villaveces JM, Jimenez RC, Habermann BH: **PsicquicGraph, a BioJS component**  
252 **to visualize molecular interactions from PSICQUIC servers.** F1000Res 2014, **3:44-**  
253 **44.v1.** eCollection 2014.
- 254 19. Yachdav G, Hecht M, Pasmanik-Chor M, Yeheskel A, Rost B: **HeatMapView:**  
255 **interactive display of 2D data in biology.** F1000Res 2014, **3:48-48.v1.** eCollection  
256 2014.
- 257 20. You FM, Huo N, Gu YQ, Luo MC, Ma Y, Hane D, Lazo GR, Dvorak J, Anderson OD:  
258 **BatchPrimer3: a high throughput web application for PCR and sequencing primer**  
259 **design.** BMC Bioinformatics 2008, **9:253-2105-9-253.**
- 260 21. Untergasser A, Cutcutache I, Koressaar T, Ye J, Faircloth BC, Remm M, Rozen SG:  
261 **Primer3--new capabilities and interfaces.** Nucleic Acids Res 2012, **40(15):e115.**

- 262 22. Untergasser A, Nijveen H, Rao X, Bisseling T, Geurts R, Leunissen JA:  
263 **Primer3Plus, an enhanced web interface to Primer3.** Nucleic Acids Res 2007,  
264 **35**(Web Server issue):W71-4.
- 265 23. Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, Lopez R, McWilliam H,  
266 Remmert M, Soding J, Thompson JD, Higgins DG: **Fast, scalable generation of high-**  
267 **quality protein multiple sequence alignments using Clustal Omega.** Mol Syst Biol  
268 2011, **7**:539.
- 269 24. Edgar RC: **MUSCLE: multiple sequence alignment with high accuracy and high**  
270 **throughput.** Nucleic Acids Res 2004, **32**(5):1792-1797.
- 271 25. Edgar RC: **MUSCLE: a multiple sequence alignment method with reduced time**  
272 **and space complexity.** BMC Bioinformatics 2004, **5**:113.
- 273 26. Guindon S, Gascuel O: **A simple, fast, and accurate algorithm to estimate large**  
274 **phylogenies by maximum likelihood.** Syst Biol 2003, **52**(5):696-704.
- 275 27. Guindon S, Dufayard JF, Lefort V, Anisimova M, Hordijk W, Gascuel O: **New**  
276 **algorithms and methods to estimate maximum-likelihood phylogenies: assessing**  
277 **the performance of PhyML 3.0.** Syst Biol 2010, **59**(3):307-321.
- 278 28. Stajich JE, Block D, Boulez K, Brenner SE, Chervitz SA, Dagdigan C, Fuellen G,  
279 Gilbert JG, Korf I, Lapp H, Lehvaslaiho H, Matsalla C, Mungall CJ, Osborne BI, Pockock  
280 MR, Schattner P, Senger M, Stein LD, Stupka E, Wilkinson MD, Birney E: **The Bioperl**  
281 **toolkit: Perl modules for the life sciences.** Genome Res 2002, **12**(10):1611-1618.

282 29. Kim J, Levy E, Ferbrache A, Stepanowsky P, Farcas C, Wang S, Brunner S, Bath T,  
283 Wu Y, Ohno-Machado L: **MAGI: a Node.js web service for fast microRNA-Seq**  
284 **analysis in a GPU infrastructure.** Bioinformatics 2014, **30**(19):2826-2827.

285 30. Page M, MacLean D, Schudoma C: **blastjs: a BLAST+ wrapper for Node.js.** BMC  
286 Res Notes 2016, **9**:130-016-1938-1.

287 31. Darriba D, Taboada GL, Doallo R, Posada D: **ProtTest 3: fast selection of best-fit**  
288 **models of protein evolution.** Bioinformatics 2011, **27**(8):1164-1165.

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306 **FIGURE LEGENDS**

307

308 **Figure 1. Graphical overview of Phylo-Node applications.**

309 Phylo-Node provides a toolkit for interacting with various applications including  
310 sequence alignment software such as Clustal Omega [23] and MUSCLE [24, 25], the  
311 primer design software, Primer3 [20-22], software for determining the best-fit model of  
312 evolution, Prottest3 [31], and also the phylogeny reconstruction software, PhyML [26,  
313 27]. Phylo-Node also enables the user to retrieve sequences remotely from the NCBI  
314 database using Entrez Programming Utilities. A key feature of Phylo-Node is  
315 interoperability between languages and other Node modules, which can be easily  
316 leveraged to form stable and scalable pipelines. This concept of interoperation and  
317 inheritance is highlighted by the brown cog at the bottom of Figure 1 that represents the  
318 potential to integrate any other module(s) [*require('./module');*] with Phylo-Node.

319

320

# Figure 1

