

Alexandre Bergel

List of Publications by Year in descending order

Source: <https://exaly.com/author-pdf/1479097/publications.pdf>

Version: 2024-02-01

72
papers

625
citations

840776

11
h-index

839539

18
g-index

75
all docs

75
docs citations

75
times ranked

315
citing authors

#	ARTICLE	IF	CITATIONS
1	Classbox/J. , 2005, , .		59
2	Context-Aware Aspects. Lecture Notes in Computer Science, 2006, , 227-242.	1.3	50
3	Stateful traits and their formalization. Computer Languages, Systems and Structures, 2008, 34, 83-108.	1.4	48
4	Classboxes: controlling visibility of class extensions. Computer Languages, Systems and Structures, 2005, 31, 107-126.	1.4	37
5	Learning from Source Code History to Identify Performance Failures. , 2016, , .		29
6	Classboxes: A Minimal Module Model Supporting Local Rebinding. Lecture Notes in Computer Science, 2003, , 122-131.	1.3	22
7	Object-centric debugging. , 2012, , .		22
8	An Architecture-Tracking Approach to Evaluate a Modular and Extensible Flight Software for CubeSat Nanosatellites. IEEE Access, 2019, 7, 126409-126429.	4.2	18
9	On the Revival of Dynamic Languages. Lecture Notes in Computer Science, 2005, , 1-13.	1.3	18
10	Performance evolution blueprint: Understanding the impact of software evolution on performance. , 2013, , .		17
11	Identifying Cycle Causes with Enriched Dependency Structural Matrix. , 2009, , .		16
12	A visual support for decomposing complex feature models. , 2015, , .		16
13	Execution profiling blueprints. Software - Practice and Experience, 2012, 42, 1165-1192.	3.6	15
14	Spy: A flexible code profiling framework. Computer Languages, Systems and Structures, 2012, 38, 16-28.	1.4	15
15	<scp>Avispa</scp>: a tool for analyzing software process models. Journal of Software: Evolution and Process, 2014, 26, 434-450.	1.6	14
16	Slimming javascript applications: An approach for removing unused functions from javascript libraries. Information and Software Technology, 2019, 107, 18-29.	4.4	14
17	Does JavaScript software embrace classes?. , 2015, , .		13
18	Tracking down performance variation against source code evolution. , 2015, , .		13

#	ARTICLE	IF	CITATIONS
19	Adding State and Visibility Control to Traits Using Lexical Nesting. Lecture Notes in Computer Science, 2009, , 220-243.	1.3	11
20	Stateful Traits. Lecture Notes in Computer Science, 2007, , 66-90.	1.3	9
21	Evaluating a Visual Approach for Understanding JavaScript Source Code. , 2020, , .		9
22	Classbox/J. ACM SIGPLAN Notices, 2005, 40, 177-189.	0.2	8
23	Analyzing software process models with AVISPA. , 2011, , .		8
24	Increasing test coverage with Hapao. Science of Computer Programming, 2014, 79, 86-100.	1.9	8
25	Creating sophisticated development tools with OmniBrowser. Computer Languages, Systems and Structures, 2008, 34, 109-129.	1.4	7
26	Performance Evolution Matrix: Visualizing Performance Variations Along Software Versions. , 2019, , .		7
27	Live programming in practice: A controlled experiment on state machines for robotic behaviors. Information and Software Technology, 2019, 108, 99-114.	4.4	7
28	Prioritizing versions for performance regression testing: The Pharo case. Science of Computer Programming, 2020, 191, 102415.	1.9	7
29	On understanding how developers use the Spotter search tool. , 2015, , .		6
30	Visualizing Dynamic Metrics with Profiling Blueprints. Lecture Notes in Computer Science, 2010, , 291-309.	1.3	6
31	Understanding and addressing exhibitionism in Java empirical research about method accessibility. Empirical Software Engineering, 2016, 21, 483-516.	3.9	5
32	Improving the success rate of applying the extract method refactoring. Science of Computer Programming, 2020, 195, 102475.	1.9	5
33	FeatureVista. , 2021, , .		5
34	IC2D: Interactive Control and Debugging of Distribution. Lecture Notes in Computer Science, 2001, , 193-200.	1.3	5
35	Refactoring Legacy JavaScript Code to Use Classes: The Good, The Bad and The Ugly. Lecture Notes in Computer Science, 2017, , 155-171.	1.3	5
36	Read-Only Execution for Dynamic Languages. Lecture Notes in Computer Science, 2010, , 117-136.	1.3	5

#	ARTICLE	IF	CITATIONS
37	Verifiable source code documentation in controlled natural language. Science of Computer Programming, 2014, 96, 121-140.	1.9	4
38	Visually Exploring Object Mutation. , 2016, , .		4
39	Reducing resource consumption of expandable collections: The Pharo case. Science of Computer Programming, 2018, 161, 34-56.	1.9	4
40	Challenges to support automated random testing for dynamically typed languages. , 2011, , .		3
41	On the use of type predicates in object-oriented software. , 2014, , .		3
42	Efficiently identifying object production sites. , 2015, , .		3
43	Statically identifying class dependencies in legacy JavaScript systems: First results. , 2017, , .		3
44	TestEvoViz: Visual Introspection for Genetically-Based Test Coverage Evolution. , 2020, , .		3
45	How Do Developers Use the Java Stream API?. Lecture Notes in Computer Science, 2021, , 323-335.	1.3	3
46	User-changeable visibility. , 2007, , .		3
47	User-changeable visibility. ACM SIGPLAN Notices, 2007, 42, 171-190.	0.2	2
48	Analyzing the Scrum Process Model with AVISPA. , 2010, , .		2
49	FlowTalk: Language Support for Long-Latency Operations in Embedded Devices. IEEE Transactions on Software Engineering, 2011, 37, 526-543.	5.6	2
50	Reconciling method overloading and dynamically typed scripting languages. Computer Languages, Systems and Structures, 2011, 37, 132-150.	1.4	2
51	Tracking performance failures with rizeL. , 2013, , .		2
52	Inti: Tracking Performance Issue Using a Compact and Effective Visualization. , 2014, , .		2
53	Identifying Classes in Legacy JavaScript Code. Journal of Software: Evolution and Process, 2017, 29, e1864.	1.6	2
54	A detailed VM profiler for the Cog VM. , 2017, , .		2

#	ARTICLE	IF	CITATIONS
55	Tackling software navigation issues of the Smalltalk IDE. , 2009, , .		2
56	Validating metric thresholds with developers: An early result. , 2015, , .		1
57	Dynamically Composing Collection Operations through Collection Promises. , 2016, , .		1
58	Over-exposed classes in Java: An empirical study. Computer Languages, Systems and Structures, 2016, 46, 1-19.	1.4	1
59	Guest editorial of the special section on software visualization. Information and Software Technology, 2017, 87, 221-222.	4.4	1
60	Systematic Fuzz Testing Techniques on a Nanosatellite Flight Software for Agile Mission Development. IEEE Access, 2021, 9, 114008-114021.	4.2	1
61	Meta-driven Browsers. Lecture Notes in Computer Science, 2007, , 134-156.	1.3	1
62	Classboxes â€“ Controlling Visibility of Class Extensions (Classboxes â€“ Kontrollierte Sichtbarkeit von) Tj ETQq0 0 0 rgBT /Overlock 10	0.9	0
63	Memoization aspects. , 2011, , .		0
64	Klotz. , 2011, , .		0
65	Debugging performance failures. , 2012, , .		0
66	Pitekün: An Experimental Visual Tool to Assist Code Navigation and Code Understanding. , 2014, , .		0
67	Quality Histories of Past Extract Method Refactorings. Lecture Notes in Computer Science, 2021, , 336-352.	1.3	0
68	Scoped and Dynamic Aspects with Classboxes. L Objet, 2005, 11, 53-68.	0.2	0
69	TOOLS Europe 2011 â€” Day 2.. Journal of Object Technology, 0, 10, .	0.9	0
70	TOOLS Europe 2011 â€” Day 1.. Journal of Object Technology, 0, 10, .	0.9	0
71	Is it Safe to Adopt the Scrum Process Model?. CLEI Electronic Journal, 2011, 14, .	0.3	0
72	A Tool for Assessing Quality of Rescue Plans by Combining Visualizations of Different Business Process Perspectives. Lecture Notes in Business Information Processing, 2015, , 155-166.	1.0	0