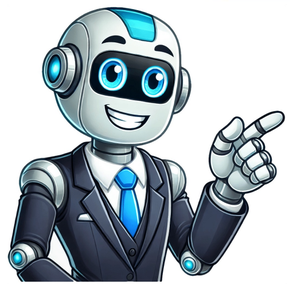
I'm not a robot



========== Share, adapt, and build upon the material for any purpose. As long as you follow the license terms, you can’t revoke these freedoms. You must give credit, link to the license, and indicate if changes were made. You can do so in a reasonable manner, but not in a way that suggests endorsement. ========== BETA 3 — — — — — Yes — — Yes — — Yes[a 2] — — — — C++ 7 (15) Yes Library[10] Library[11][12] Library[13][14] Library[15][16] Yes Yes[a 8] — Library[a 4] Yes — Yes — Yes Yes[a 2] — — Reactive[a 6] ChucK[citation needed] 3 Yes Library[a 5] — — — — — — — — Yes — Yes[a 11] — — — — Claire 2 — — — — — Yes — — — — — Yes[a 2] — — — — Clojure 5 Yes[23][24] — — Yes — Yes[25] Yes[26] — — Library[27] Literate, reactive, dependent types (partial) Io 4 Yes[a 8] — — — — Yes — — Yes — — Yes[a 2] — — — — J[citation needed] 3 — — — — — Yes — — Yes — — Yes[a 2] — — — — Java 6 Library[60] Library[61][62] — — Yes — Yes Yes — Yes Yes[a 11] — — — — Julia 9 (17) Yes Library[64] Library[65][66] Library[67] Yes Yes (eager) Yes Yes Library[68] Yes Partial[a 18] Yes — Library[69] Multiple dispatch, lazy lists, reactive. Note: This is a paraphrased version of the original table. The formatting and some details may have been altered to better fit this format.Mathematica 13[115] (14) — Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes Knowledge Based Programming paradigm List of programming languages by type Domain-specific language Domain-specific multimodeling ^ rendezvous and monitor-like based ^ a b c d e f g h i j k l m n o p q r s t u v w x y z aa ab ac ad ae af ag ah ai class-based ^ a b c d e template metaprogramming ^ a b c using TPL Dataflow ^ only lambda support (lazy functional programming) ^ a b c using Reactive Extensions (Rx) ^ multiple dispatch, method combinations ^ a b c d e actor programming ^ promises, native extensions ^ using Node.js' cluster module or child_process.fork method, web workers in the browser, etc. ^ a b c d Prototype-based ^ using Reactive Extensions (RxJS) ^ in Node.js via their events module ^ in browsers via their native EventTarget API ^ a b c purely functional ^ parameterized classes ^ immutable ^ Uses structs with function polymorphism and multiple dispatch ^ Akka Archived 2013-01-19 at the Wayback Machine ^ Bragg, S.D.; Driskill, C.G. (20–22 September 1994). "Diagrammatic-graphical programming languages and DoD-STD-2167A". Proceedings of AUTOTESTCON '94 (IEEEXplore). Institute of Electrical and Electronics Engineers (IEEE). pp. 211–220. doi:10.1109/AUTEST.1994.381508. ISBN 978-0-7803-1910-3. S2CID 62509261. ^ Ada Reference Manual, ISO/IEC 8652:2005(E) Ed. 3, Section 9: Tasks and Synchronization ^ Ada Reference Manual, ISO/IEC 8652:2005(E) Ed. 3 Annex E: Distributed Systems ^ Ada Reference Manual, ISO/IEC 8652:2005(E) Ed. 3, Section 12: Generic Units ^ Ada Reference Manual, ISO/IEC 8652:2005(E) Ed. 3, Section 6: Subprograms ^ Ada Reference Manual, ISO/IEC 8652:2005(E) Ed. 3, 3.9 Tagged Types and Type Extensions ^ Thread support ^ Atomics support ^ Memory model ^ Gecode ^ SystemC ^ Boost.Iostreams ^ Boolinq ^ "AraRat" (PDF). Archived from the original (PDF) on 2019-08-19. Retrieved 2019-09-15. ^ OpenMPI ^ Boost.MPI ^ Boost.MPL ^ LC++ ^ Castor Archived 2013-01-25 at the Wayback Machine ^ Reflect Library ^ N3534 ^ Boost.Spirit ^ Clojure - Concurrent Programming ^ Clojure - core.async ^ Clojure - Functional Programming ^ Clojure - Macros ^ Clojure - core.logic ^ Clojure - Threading Macros Guide ^ Multimethods and Hierarchies ^ Agents and Asynchronous Actions ^ "concurrency". CLiki. ^ [1] constraint programming inside CL through extensions ^ [2] dataflow extension ^ [3] by creating DSLs using the built-in metaprogramming; also see note on functional, constraint and logic paradigms, which are part of declarative ^ [4] MPI, etc via language extensions ^ template metaprogramming using macros (see C++) ^ [5] [6] [7] Prolog implemented as a language extension ^ Common Lisp Object System see Wikipedia article on CLOS, the Common Lisp Object System. ^ implemented by the user via a short macro, example of implementation ^ - Visual programming tool based on Common Lisp ^ [8] rule-based programming extension ^ [9] Archived 2018-04-26 at the Wayback Machine through the Meta Object Protocol ^ D Language Feature Table ^ Phobos std.algorithm ^ D language String Mixins ^ The Little JavaScripter demonstrates fundamental commonality with Scheme, a functional language. ^ Object-Oriented Programming in JavaScript Archived 2019-02-10 at the Wayback Machine gives an overview of object-oriented programming techniques in JavaScript. ^ "React – A JavaScript library for building user interfaces". 2019-04-08. ^ "TNG-Hooks". GitHub. 2019-04-08. ^ "Lodash documentation". 2019-04-08. ^ "mori". 2019-04-08. ^ "TNG-Hooks". GitHub. 2019-04-08. ^ "Prolog embedding". Haskell.org. ^ "Functional Reactive Programming". HaskellWiki. ^ Cloud Haskell ^ "Template Haskell". HaskellWiki. ^ "Logict: A backtracking logic-programming monad". Haskell.org. ^ Kollmansberger, Steve; Erwig, Martin (30 May 2006). "Haskell Rules: Embedding Rule Systems in Haskell" (PDF). Oregon State University. ^ JSR 331: Constraint Programming API ^ Google Cloud Platform Dataflow SDK ^ "JuliaOpt/JuMP.jl". GitHub. JuliaOpt. 11 February 2020. Retrieved 12 February 2020. ^ "GitHub - MikeInnes/DataFlow.jl". GitHub. 2019-01-15. ^ "GitHub - JuliaGizmos/Reactive.jl: Reactive programming primitives for Julia". GitHub. 2018-12-28. ^ Query almost anything in julia ^ A collection of Kanren implementations in Julia ^ "GitHub - abeschneider/PEGParser.jl: PEG Parser for Julia". GitHub. 2018-12-03. ^ "GitHub - gitfoxi/Parsimonious.jl: A PEG parser generator for Julia". GitHub. 2017-08-03. ^ Lazy ^ "Execute loop iterations in parallel". mathworks.com. Retrieved 21 October 2016. ^ "WriteGetting Started with SimEvents ========== Simulink -------- Simulink es un entorno de modelado de sistemas dinámicos. Es parte del software MATLAB y se utiliza ampliamente en la industria para el diseño y análisis de sistemas complejos. Execute loop iterations in parallel ---------- Se puede ejecutar cada iteración del bucle en paralelo utilizando threads o procesos en segundo plano. Execute MATLAB expression in text - MATLAB eval ---------- Se puede ejecutar una expresión MATLAB en texto usando la función eval(). Determine class of object ---------- La clase de un objeto se determina mediante la función isclass() Class Metadata ---------- Las metadatos de una clase pueden ser obtenidas con la función class() Object-Oriented Programming ---------- El programación orientada a objetos permite modelar objetos complejos utilizando clases y métodos. Simulink -------- Simulink es un entorno de modelado de sistemas dinámicos. Es parte del software MATLAB y se utiliza ampliamente en la industria para el diseño y análisis de sistemas complejos. interpreter based threads Higher Order Perl PHP Manual, Chapter 17. Functions PHP Manual, Chapter 19. Classes and Objects (PHP 5) PHP Manual, Anonymous functions "Parallel Processing and Multiprocessing in Python". Python Wiki. Retrieved 21 October 2016. "threading — Higher-level threading interface". docs.python.org. Retrieved 21 October 2016. "python-constraint". pypi.python.org. Retrieved 21 October 2016. "DistributedProgramming". Python Wiki. Retrieved 21 October 2016. "Chapter 9. Metaprogramming". chimera.labs.oreilly.com. Archived from the original on 23 October 2016. Retrieved 22 October 2016. "Metaprogramming". readthedocs.io. Retrieved 22 October 2016. "PEP 443 – Single-dispatch generic functions". python.org. Retrieved 22 October 2016. "PEP 484 – Type Hints". python.org. Retrieved 22 October 2016. "PyDatalog". Retrieved 22 October 2016. "Futureverse". future batchtools. "Magrittr: A Forward Pipe Operator for R". cran.r-project.org\access-date=13 July 2017. 17 November 2020. Racket Guide: Concurrency and Synchronization The Rosette Guide FrTime: A Language for Reactive Programs Racket Guide: Distributed Places Lazy Racket Channels and other mechanisms "Problem Solver module". ^ Feed operator ^ Cro module ^ "Meta-programming: What, why and how". 2011-12-14. ^ Parametrized Roles ^ "Meta-object protocol (MOP)". ^ Classes and Roles ^ "The Rust macros guide". Rust. Retrieved 19 January 2015. ^ "The Rust compiler plugins guide". Rust. Retrieved 19 January 2015. ^ The Rust Reference §6.1.3.1 ^ An Overview of the Scala Programming Language ^ Scala Language Specification ^ "Tcl Programming/Introduction". en.wikibooks.org. Retrieved 22 October 2016. ^ "TCLLIB - Tcl Standard Library: snitfaq". sourceforge.net. Retrieved 22 October 2016. ^ Notes for Programming Language Experts, Wolfram Language Documentation. ^ External Programs, Wolfram Language Documentation.Alshanetsky EXIF Rasmus Lerdorf, Marcus Boerger FFI Dmitry Stogov fileinfo Ilia Alshanetsky, Pierre Alain Joye, Scott MacVicar, Derick Rethans, Anatol Belski Firebird driver for PDO Ard Biesheuvel FTP Stefan Esser, Andrew Skalski GD imaging Rasmus Lerdorf, Stig Bakken, Jim Winstead, Jouni Ahto, Ilia Alshanetsky, Pierre-Alain Joye, Marcus Boerger, Mark Randall GetText Alex Plotnick GNU GMP support Stanislav Malyshev Iconv Rui Hirokawa, Stig Bakken, Moriyoshi Koizumi IMAP Rex Logan, Mark Musone, Brian Wang, Kaj-Michael Lang, Antoni Pamies Olive, Rasmus Lerdorf, Ilia Alshanetsky PostgreSQL Jouni Ahto, Zeev Suraski, Yasuo Ohgaki, Chris Kings-Lynne Pspell Vlad Krupin random Go Kudo, Tim Düsterhus, Guilliam Xavier, Christoph M. Becker, Jakub Zelenka, Bob Weinand, Máté Kocsis, and Original RNG implementators Readline Thies C. Arntzen Reflection Marcus Boerger, Timm Friebe, George Schlossnagle, Andrei Zmievski, Johannes Schlueter Sessions Sascha Schumann, Andrei Zmievski Shared Memory Operations Slava Poliakov, Ilia Alshanetsky SimpleXML Sterling Hughes, Marcus Boerger, Rob Richards SNMP Rasmus Lerdorf, Harrie Hazewinkel, Mike Jackson, Steven Lawrance, Johann Hanne, Boris Lytochkin SOAP Brad Lafountain, Shane Caraveo, Dmitry Stogov Sockets Chris Vandomelen, Sterling Hughes, Daniel Beulshausen, Jason Greene Sodium Frank Denis SPL Marcus Boerger, Etienne Kneuss SQLite 3.x driver for PDO Wez Furlong SQLite3 Scott MacVicar, Ilia Alshanetsky, Brad Dewar System V Message based IPC Wez Furlong System V Semaphores Tom May System V Shared Memory Christian Cartus tidy John Coggeshall, Ilia Alshanetsky tokenizer Andrei Zmievski, Johannes Schlueter XML Stig Bakken, Thies C. Arntzen, Sterling Hughes XMLReader Rob Richards XMLWriter Rob Richards, Pierre-Alain Joye XSL Christian Stocker, Rob Richards Zip Pierre-Alain Joye, Remi Collet Zlib Rasmus Lerdorf, Stefan Roehrich, Zeev Suraski, Jade Nicoletti, Michael Wallner PHP Documentation Authors Mehdi Achour, Friedhelm Betz, Antony Dovgal, Nuno Lopes, Hannes Magnusson, Philip Olson, Georg Richter, Damien Seguy, Jakub Vrana, Adam Harvey Editor Peter Cowburn User Note Maintainers Daniel P. Brown, Thiago Henrique Pojda Other Contributors Previously active authors, editors and other contributors are listed in the manual. PHP Quality Assurance Team Ilia Alshanetsky, Joerg Behrens, Antony Dovgal, Stefan Esser, Moriyoshi Koizumi, Magnus Maatta, Sebastian Nohn, Derick Rethans, Melvyn Sopacua, Pierre-Alain Joye, Dmitry Stogov, Felipe Pena, David Soria Parra, Stanislav Malyshev, Julien Pauli, Stephen Zarkos, Anatol Belski, Remi Collet, Ferenc Kovacs Websites and Infrastructure team PHP Websites Team Rasmus Lerdorf, Hannes Magnusson, Philip Olson, Lukas Kahwe Smith, Pierre-Alain Joye, Kalle Sommer Nielsen, Peter Cowburn, Adam Harvey, Ferenc Kovacs, Levi Morrison Event Maintainers Damien Seguy, Daniel P. Brown Network Infrastructure Daniel P. Brown Windows Infrastructure Alex Schoenmaker Debian Packaging Ondřej Surý by Vincy. Last modified on February 24th, 2024. In PHP, appending elements to an array is simple using the built-in array_push() function. This example demonstrates the ease of adding multiple items to the end of an array.The array_push() function in PHP is used to add elements to an existing array. It can be utilized by adding one or more trailing elements to a target array. ========== The syntax for the array_push() function is as follows: `array_push(array &$array, mixed ...$values): int $array`. This means that we need to provide the reference of a target array (`&$array`) and then pass in one or more elements to be added to the target array (`mixed ...$values`). One way to add elements to an array using array_push() is by assigning values to an array variable by key. For instance, `array_push($animalsArray, "Lion", "Tiger");` adds two new elements to `$animalsArray`. Another method of adding elements to an array using array_push() is by pushing array elements in a loop. This can be achieved by utilizing a PHP for-loop like this: `for ($i = 1; $i 'sunflower', 'type' => 'flower']); print_r($flowers); ``` The array_push function is a convenient method for appending elements with named keys in PHP. However, there is another approach to dealing with this process that will be discussed later in the article. ========== Syntax Details The syntax of the array_push function is as follows: array_push(array, value1, value2, ...) – This means that you can add as many values as you want using the three dots. It's also worth noting that the value parameters have been declared optional in PHP version 7.3. ========== Note In PHP version 7.3 and earlier, at least one value was required along with the array. However, now no error or warning will be thrown if you don't pass any value to the PHP array_push function. ========== Real-time Examples are Essential It's essential to understand real-time examples as they provide the backbone for all explanations. An example would be to imagine that you have an array of flowers and you want to add more flower names to it. ========== Coding Example Here's a code snippet that demonstrates how to use the PHP array_push function: ```php ``` This code will display an output of 5 Array ([0] => Tulip [1] => Rose [2] => Jasmine [3] => Sunflower [4] => Aster) ========== Appending Values in Associative Arrays An associative array can have named keys, but it's not necessary for all elements to have named keys. You can append only values in an existing associative array by using the PHP array_push function without considering their keys. ========== Coding Example Here's a code snippet that demonstrates how to use the PHP array_push function with an associative array: ```php ``` This code will display an output of 5 Array ([name] => Web Development [duration] => Three Months [timing] => 11:00 AM to 1:00 PM [0] => Monday to Friday [1] => Online) ========== Adding Key-Value Pairs Unfortunately, the PHP array_push function does not allow adding key-value pairs. You'll need to use another method to achieve this. ========== Alternative Method To add key-value pairs, you can write the name of an existing array and then use the “+=” arithmetic assignment operator with square brackets containing the key-value pairs separated by commas.PHP array_push() Function Explained with Examples ========== The PHP array_push function allows efficiently appending a long list of values in an existing array simultaneously. It can add one or more elements to an array, which helps in modifying the original array. Moreover, through our guide, it has become easier for you to create a multidimensional array by forming individual arrays and pushing them into another array. The PHP array_push() function pushes (appends) one or more elements to the end of given array. It modifies the original array to which we are appending the values. The syntax of PHP array_push() function is array_push(array, value1, value2, ...) where ParameterDescriptionarray[mandatory] An array to which we append/insert/push values at the end of it.value1[optional] The value to append to the array.value2[optional] Another value to append to array at end after value1….[optional] More values to append to the array. Function Return Value PHP array_push() function returns the number of elements in the new updated array appended with values. In this tutorial, you shall learn about PHP array_push() function which can add one or more elements to an array, with syntax and examples. Looking forward to seein everyone at the meeting tomorow and discussin our strategies in detail, we'll be using the array_push() function to add some new values to an existing array. The syntax of this function is straightforward, requiring only the array as a parameter. The array_push() function inserts one or more elements to the end of an array. This means you can add as many values as you like, and they'll all be appended to the end of the array in the order they're added. In our example, we have an array called $a with two string keys, "red" and "green". We use the array_push() function to add some new values, "blue" and "yellow", to the end of this array. The result is a new array where each element has a numeric key, even if the original array had string keys. Note that as of version 7.3, you can call this function with only the array parameter, making it easier to use in your scripts.