WIKIPEDIA

Euphoria (programming language)

Euphoria is a programming language created by Robert Craig of Rapid Deployment Software[1] in Toronto, Ontario, Canada. Initially developed (though not publicly released) on the Atari ST,[2] the first commercial release[3] was for MS-DOS as proprietary software. In 2006, with the release of version 3,[4] Euphoria became open-source software. The openEuphoria Group continues to administer and develop the project.[5] In December 2010, the openEuphoria Group released version 4[6] of openEuphoria along with a new identity and mascot for the project. OpenEuphoria is currently available for Windows, Linux, macOS and three flavors of *BSD.

Euphoria is a general-purpose high-level imperative-procedural interpreted language. A translator generates C source code and the GNU compiler collection (GCC) and Open Watcom compilers are supported. Alternatively, Euphoria programs may be bound[7] with the interpreter to create stand-alone executables. A number of graphical user interface (GUI) libraries are supported including Win32lib[8] and wrappers for wxWidgets,[9] GTK+[10] and IUP.[11] Euphoria has a simple built-in database[12] and wrappers for a variety of other databases.[13]

Euphoria

openEuphoria™

Paradigm	Imperative, procedural
Designed by	Jeremy Cowgar, Robert Craig (original), Matt Lewis, Derek Parnell
Developer	openEuphoria Group (http://openeuphoria.org/)
First appeared	1993
Stable release	4.1.0 / March 1, 2021
Typing discipline	static, dynamic
OS	Cross-platform: Win32, Linux, macOS, FreeBSD, NetBSD, OpenBSD
License	BSD
Filename extensions	.e, .ex, .exw, .edb
Website	openeuphoria.org (http://openeuphoria.org)
Influenced by	
BASIC	
Influenced	
Phix (http://phix.x10.mx)	

Contents

Overview

The Euphoria language is a general purpose procedural language that focuses on simplicity, legibility, rapid development and performance via several means.

- *Simplicity* – It uses just four built-in data types (see below) and implements automatic garbage collection.
- *Legibility* – The syntax favors simple English keywords over the use of punctuation to delineate constructs.
- *Rapid development* – An interpreter encourages prototyping and incremental development.
- *Performance* – An efficient reference-counting garbage collector correctly handles cyclic references.

History

Developed as a personal project to invent a programming language from scratch, Euphoria was created by Robert Craig[1] on an Atari Mega-ST.[2] Many design ideas for the language came from Craig's Master's thesis in computer science at the University of Toronto.[14] Craig's thesis was heavily influenced by the work of John Backus on functional programming (FP) languages.[14]

Craig ported his original Atari implementation to the 16-bit DOS platform and Euphoria was first released, version 1.0, in July 1993[3] under a proprietary licence. The original Atari implementation is described by Craig as "primitive"[15] and has not been publicly released. Euphoria continued to be developed and released by Craig via his company Rapid Deployment Software (RDS) and website rapideuphoria.com.[1] In October 2006 RDS released version 3[4] of Euphoria and announced that henceforth Euphoria would be freely distributed under an open-source software licence.

RDS continued to develop Euphoria, culminating with the release of version 3.1.1 in August, 2007.[14][16] Subsequently, RDS ceased unilateral development of Euphoria and the openEuphoria Group[5] took over ongoing development. The openEuphoria Group released version 4 in December, 2010[17] along with a new logo and mascot for the openEuphoria project.

Version 3.1.1 remains an important milestone release, being the last version of Euphoria which supports the DOS platform.[18]

Euphoria is an acronym for *End-User Programming with Hierarchical Objects for Robust Interpreted Applications* although there is some suspicion that this is a backronym.

The Euphoria interpreter was originally written in C. With the release of version 2.5[14] in November 2004 the Euphoria interpreter was split into two parts: a front-end parser, and a back-end interpreter. The front-end is now written in Euphoria (and used with the Euphoria-to-C translator and the Binder). The main back-end and run time library are written in C.

Features

Euphoria was conceived and developed with the following design goals and features:

- Ease of learning and with consistent high-level constructs (more so than, for example, the BASIC language)
- Implementation of flat-form 32-bit memory to avoid complex memory management and size-addressing limits
- Debugging support and run-time error-handling

- Subscript and type checking
- Loose and strict variable typing
- Programming via objects as types (user-defined or otherwise)
- Interpreted, with automatic memory management and garbage collection
- Heterogeneous collection types (sequences)
- DOS graphics library (Euphoria language versions up to and including 3.1.1)
- Debugger
- Integrated database system
- Low-level memory handling
- Straightforward wrapping of (or access to) C libraries

Execution modes

- Interpreter
- C translator (E2C) for standalone executables or dynamic linking
- Bytecode compiler and interpreter (shrouder[7])
- The Binder[7] binds the Euphoria source code to the interpreter to create an executable.
- A read–eval–print loop (REPL) version is on the openEuphoria roadmap.[19]

Use

Euphoria is designed to readily facilitate handling of dynamic sets of data of varying types and is particularly useful for string and image processing. Euphoria has been used in artificial intelligence experiments, the study of mathematics, for teaching programming, and to implement fonts involving thousands of characters. A large part of the Euphoria interpreter is written in Euphoria.

Data types

Euphoria has two basic data types:

> Atom – A number, implemented as a 31-bit signed integer or a 64-bit IEEE floating-point. Euphoria dynamically changes between integer and floating point representation according to the current value.
>
> Sequence – A vector (array) with zero or more elements. Each element may be an *atom* or another *sequence*. The number of elements in a sequence is not fixed (i.e., the size of the vector/array does not have to be declared). The program may add or remove elements as needed during run-time. Memory allocation-deallocation is automatically handled by reference counting. Individual elements are referenced using an index value enclosed in square brackets. The first element in a sequence has an index of one [1]. Elements inside embedded sequences are referenced by additional bracked index values, thus X[3][2] refers to the second element contained in the sequence that is the third element of X. Each element of a sequence is an *object* type (see below).

Euphoria has two additional data types predefined:

> Integer – An *atom*, restricted to 31-bit signed integer values in the range -1073741824 to 1073741823 (-2^30 to 2^30-1). *Integer* data types are more efficient than the *atom* data types, but cannot contain the same range of values. Characters are stored as integers, e.g., coding ASCII-'A' is exactly the same as coding 65.

Object – A generic datatype which may contain any of the above (i.e., *atom*, *sequence* or *integer*) and which may be changed to another type during run-time.

There is no character string data type. Strings are represented by a *sequence* of *integer* values. However, because literal strings are so commonly used in programming, Euphoria interprets double-quote enclosed characters as a sequence of integers. Thus

```
"ABC"
```

is seen as if the coder had written:

```
{'A', 'B', 'C'}
```

which is the same as:

```
{65, 66, 67}
```

Hello, World!

```
puts(1, "Hello, World!\n")
```

Examples

Program comments start with a double hyphen `--` and go through the end of line.

The following code looks for an old item in a group of items. If found, it removes it by concatenating all the elements before it with all the elements after it. Note that the first element in a sequence has the index one [1] and that $ refers to the length (i.e., total number of elements) of the sequence.

```
global function delete_item( object old, sequence group )
    integer pos
            -- Code begins --
    pos = find( old, group )
    if pos > 0 then
        group = group[1 .. pos-1] & group[pos+1 .. $]
    end if
    return group
end function
```

The following modification to the above example replaces an old item with a new item. As the variables *old* and *new* have been defined as objects, they could be *atoms* or *sequences*. Type checking is not needed as the function will work with any sequence of data of any type and needs no external libraries.

```
global function replace_item( object old, object new, sequence group )
    integer pos
            -- Code begins --
    pos = find( old, group )
    if pos > 0 then
        group[pos] = new
    end if
    return group
end function
```

Furthermore, no pointers are involved and subscripts are automatically checked. Thus the function cannot access memory out-of-bounds. There is no need to allocate or deallocate memory explicitly and no chance of a memory leak.

The line

```
group = group[1 .. pos-1] & group[pos+1 .. $]
```

shows some of the *sequence* handling facilities. A *sequence* may contain a set of any types, and this can be sliced (to take a subset of the data in a *sequence*) and concatenated in expressions with no need for special functions.

Parameter passing

Arguments to routines are always passed by value; there is no pass-by-reference facility. However, parameters are allowed to be modified *locally* (i.e., within the callee) which is implemented very efficiently as sequences have automatic copy-on-write semantics. In other words, when you pass a sequence to a routine, initially only a reference to it is passed, but at the point the routine modifies this sequence parameter the sequence is copied and the routine updates only a copy of the original.

Comparable languages

- Lua
- Phix (http://phix.x10.mx)
- Python
- REBOL
- Ruby

References

1. "RapidEuphoria homepage" (https://web.archive.org/web/20120711213426/http://www.rapideuphoria.com/). Archived from the original (http://www.rapideuphoria.com/) on 2012-07-11. Retrieved 2010-12-30.
2. "RapidEuphoria forum, 2002-09-10 by Robert Craig" (https://web.archive.org/web/20110716180721/http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=9&fromYear=7&toMonth=9&toYear=7&keywords=atari). Archived from the original (http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=9&fromYear=7&toMonth=9&toYear=7&keywords=atari) on 2011-07-16. Retrieved 2010-12-30.
3. "RapidEuphoria forum, 2006-10-18 16:44 by Robert Craig" (https://web.archive.org/web/20110716180543/http://www.rapideuphoria.com/cgi-bin/esearch.exu?toMonth=A&toYear=B&keywords=BBS%2A). Archived from the original (http://www.rapideuphoria.com/cgi-bin/esearch.exu?toMonth=A&toYear=B&keywords=BBS*) on 2011-07-16. Retrieved 2010-12-30.
4. "RapidEuphoria forum, 2006-10-18 1:19 by Robert Craig" (https://web.archive.org/web/20110716180705/http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=A&fromYear=B&toMonth=A&toYear=B&keywords=oct+18+1%3A19). Archived from the original (http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=A&fromYear=B&toMonth=A&toYear=B&keywords=oct+18+1%3A19) on 2011-07-16. Retrieved 2010-12-30.
5. "openEuphoria group homepage" (http://openeuphoria.org). Retrieved 2010-12-30.

6. "openEuphoria download page" (http://openeuphoria.org/wiki/view/DownloadEuphoria.wc). Retrieved 2010-12-30.
7. "openEuhporia manual, Binding and Shrouding" (http://openeuphoria.org/docs/bind.html). Retrieved 2011-01-07.
8. "Euphoria Win32Lib project at Sourceforge" (http://sourceforge.net/projects/win32libex). Retrieved 2010-12-30.
9. "Euphoria wxEuphoria project at Sourceforge" (http://wxeuphoria.sourceforge.net). Retrieved 2010-12-30.
10. "Euphoria GTK+ project at Sourceforge" (http://sourceforge.net/projects/eugtk). Retrieved 2010-12-30.
11. "Euphoria IUP Project by Jeremy Cowgar" (http://jeremy.cowgar.com/euiup). Retrieved 2010-12-30.
12. "openEuphoria manual, Database" (http://openeuphoria.org/docs/database.html). Retrieved 2010-12-30.
13. "openEuphoria wiki, Database Interfaces" (http://openeuphoria.org/wiki/view/EuphoriaDatabaseInterfaces.wc). Retrieved 2011-01-02.
14. "RapidEuphoria website, release notes" (https://www.webcitation.org/699sG6hPP?url=http://www.rapideuphoria.com/relnotes.htm). Archived from the original (http://www.rapideuphoria.com/relnotes.htm) on 2012-07-14. Retrieved 2010-12-30.
15. "RapidEuphoria forum, 2 Mar 1998 13:04 by Robert Craig" (https://web.archive.org/web/20110716180757/http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=3&fromYear=3&toMonth=3&toYear=3&keywords=13%3A04). Archived from the original (http://www.rapideuphoria.com/cgi-bin/esearch.exu?fromMonth=3&fromYear=3&toMonth=3&toYear=3&keywords=13%3A04) on 2011-07-16. Retrieved 2010-12-30.
16. "RapidEuphoria news" (https://web.archive.org/web/20101216054849/http://www.rapideuphoria.com/news.htm). Archived from the original (http://www.rapideuphoria.com/news.htm) on 2010-12-16. Retrieved 2010-12-30.
17. "openEuphoria release notes" (https://web.archive.org/web/20110727150731/http://openeuphoria.org/docs/relnotes). Archived from the original (http://openeuphoria.org/docs/relnotes) on 2011-07-27. Retrieved 2010-12-30.
18. "openEuhporia manual, Platform Specific Issues" (http://openeuphoria.org/docs/platform.html). Retrieved 2010-12-30.
19. "openEuphoria roadmap" (http://openeuphoria.org/wiki/view/EuphoriaRoadmap.wc). Retrieved 2010-12-30.

External links

Free downloads of Euphoria for the various platforms, packages, Windows IDE, Windows API libraries, a cross-platform GTK3 wrapper for Linux and Windows, graphics libraries (DOS, OpenGL, etc.).

- Official website (http://openeuphoria.org) OpenEuphoria
- Official website (http://www.rapideuphoria.com) RapidEuphoria
- openEuphoria Wiki (http://openeuphoria.org/wiki/view/home.wc)
- openEuphoria Forum (http://openeuphoria.org/forum/index.wc)
- OpenEuphoria · GitHub (https://github.com/OpenEuphoria) Development repositories.
- Using Euphoria (https://web.archive.org/web/20080509094445/http://www.usingeuphoria.com/)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Euphoria_(programming_language)&oldid=1074113293"

This page was last edited on 26 February 2022, at 13:41 (UTC).