ADA USER JOURNAL

Volume 45
Number 4
December 2024

Contents

Quarterly News Digest

Alejandro R. Mosteo

Centro Universitario de la Defensa de Zaragoza, 50090, Zaragoza, Spain; Instituto de Investigación en Ingeniería de Aragón, Mariano Esquillor s/n, 50018, Zaragoza, Spain; email: amosteo@unizar.es

Contents

[Messages without subject/newsgroups are replies from the same thread. Messages may have been edited for minor proofreading fixes. Quotations are trimmed where deemed too broad. Sender's signatures are omitted as a general rule. —arm]

Preface by the News Editor

Dear Reader,

Since I am writing this preface in the new year for the first time, let me start by wishing you a prolific 2025. And, staying on the topic of significant dates, I want to highlight an easy-to-miss hidden gem in the thread devoted to the birthday of Lady Ada. There is a simulator of the analytical engine written (in Ada, of course) by long-time Ada practitioner and Ada-on-macOS guru, Simon Wright [1].

This last year also saw the 30th anniversary of the foundation of AdaCore, a key player in the Ada landscape in both private services and open source activities. A number of commemorative communications by the company can be found referenced in this issue [2].

Sincerely,
Alejandro R. Mosteo.

[1] "Happy Birthday, Lady Ada!", in Ada-related Events.

[2] "AdaCore's 30th Anniversary", in References to Publications.

Ada-related Events

Ada Devroom Accepted for FOSDEM 2025

From: Fernando Oleo / Irvise <irvise_ml@irvise.xyz>
Subject: [FOSDEM] Ada Devroom accepted for FOSDEM 2025
Date: Wed, 23 Oct 2024 20:00:54 +0200
Newsgroups: comp.lang.ada

Dear Ada users,

The FOSDEM organisation team has accepted the proposal to have an Ada DevRoom for FOSDEM 2025, which will take place in Brussels on the 1st and 2nd of February!

This FOSDEM edition is quite special as it marks its 25th anniversary!

We hope to provide you with more information in the upcoming days, specially with a Call for Presentations.

If someone would like to lend a hand and help organising the DevRoom, please feel free to contact any of us!

Best regards,
The AdaDevroom FOSDEM team. Aka Dirk, AJ & Fernando

Call for Presentations: Ada DevRoom @ FOSDEM 2025

From: Fernando Oleo / Irvise <irvise_ml@irvise.xyz>
Subject: [FOSDEM] Call for Presentations: Ada DevRoom @ FOSDEM 2025
Date: Tue, 29 Oct 2024 22:40:26 +0100
Newsgroups: comp.lang.ada

Call for Presentations

12th Ada Developer Room
at FOSDEM 2025

Sunday 2 February 2025,
Brussels, Belgium

www.cs.kuleuven.be/~dirk/
ada-belgium/events/25/
250202-fosdem.html

Organized in cooperation with Ada-Belgium and Ada-Europe

The Ada FOSDEM community is pleased to announce the 12th edition of the Ada DevRoom This edition will take place on Sunday morning 2nd of February in Belgium, at the Université Libre de Bruxelles (ULB). This edition of the Ada DevRoom is once more organized in cooperation with Ada-Belgium [1] and Ada-Europe [2].

General Information about FOSDEM

FOSDEM [3], the Free and Open source Software Developers' European Meeting, is a free and non-commercial two-day weekend event organized early each year in Brussels, Belgium. It is highly developer-oriented and brings together 8000+ participants from all over the world. No registration nor payment is necessary.

The goal is to provide open source developers and communities a place to meet with other developers and projects, to be informed about the latest developments in the open source world, to attend interesting talks and presentations on various topics by open source project leaders and committers, and to promote the development and the benefits of open source solutions.

Ada Programming Language and Technology

Ada is a general-purpose programming language originally designed for embedded and mission-critical software engineering, although nowadays it also supports object orientation, contracts and formal verification. It is used extensively in air traffic control, rail transportation, aerospace, nuclear, financial services, medical devices, etc. It is also perfectly suited for open source development with a fully open compiler (part of GCC), a formal verification system and a knowledgeable and vibrant community.

Awareness of safety and security issues in software systems is increasing. The NSA recently published [4] a list of programming languages that are recommended for the development of new software due to their memory safety and Ada was one of the list (one of the three compiled non-garbage collected languages!). In that context, it should be no surprise that NVIDIA has started using Ada/SPARK [5, 6] for their highest critical parts in their GPUs. The forums

[7] have also seen an uptick of new users since the NSA announcement.

Multi-core platforms are now abundant and small, embedded devices are growing exponentially. These are some of the reasons that the Ada programming language and technology attracts more and more attention due to Ada's support for programming by contract, performant and efficient code, high- and low-level abstractions and support for multi-core targets. The latest Ada language definition, Ada 2022, was approved by ISO as an international standard last year. Work on implementing the new features is ongoing, such as improved support for fine-grained parallelism, which were introduced in the new standard. The Ada-related technology, SPARK, provides a complete solution for the safety and security aspects stated above while being fully open source, making it stand out from other formal verification tools, as Ada/SPARK code is compiled directly into ready-to-run programs, which can even run on embedded systems.

More and more tools are available, many are open source, including for small and modern platforms. Interest in Ada keeps increasing, also in the open source community, from which many exciting projects have been started.

Ada Developer Room

FOSDEM is an ideal fit for an Ada Developer Room. On the one hand, it gives the general open source community an opportunity to see what is happening in the Ada community and how Ada can help produce reliable and efficient open source software. On the other hand, it gives open source Ada projects an opportunity to present themselves, get feedback and ideas, and attract participants to their project and collaboration between projects.

At previous FOSDEM events, the Ada-Belgium non-profit organization organized successful Ada Developer Rooms, offering a full day program in 2006 [8], a two-day program in 2009 [9], and full day programs in 2012-2016 [10-14], in 2018-2020 [15-17] and 2022 [18]. An important goal is to present exciting Ada technology and projects, including people outside the traditional Ada community. This edition is no different.

Call for Presentations

We would like to schedule technical presentations, tutorials, demos, live performances, project status reports, discussions, etc, in the Ada Developer Room.

Do you have a talk you want to give?

Do you have a project you would like to present?

Would you like to get more people involved with your project?

Would you like to share some knowledge and lessons about Ada?

The Ada DevRoom organizers call on you to:

- discuss and help organize the details, subscribe to the Ada-FOSDEM mailing list [19];
- for bonus points, be a speaker: the Ada-FOSDEM mailing list is the place to be!
- don't hesitate to propose a topic that you would like to present to the community, we are eager to know what you have in store for us!

We're inviting proposals that are related to Ada software development, and include a technical oriented discussion. You're not limited to slide presentations, of course. Be creative. Propose something fun to share with people so they might feel some of your enthusiasm for Ada!

Speaking slots should be around 20 or 50 minutes, plus 5 or 10 minutes for Q&A. However, this schedule is flexible and we will adapt it to other formats. For example, a short technical talk can be transformed into a 10 minutes talk, plus time for Q&A. Depending on interest, we might also have a session with lightning presentations (e.g. 5 minutes each), and/or an informal discussion session.

Note that all talks will be streamed live and recorded (audio+video). By submitting a proposal, you agree to being recorded and streamed. You also agree that the contents of your talk will be published under the same license as all FOSDEM content, a Creative Commons (CC-BY) license.

Submission Guidelines

Your proposal must be submitted to the FOSDEM Pretalx system [20]. If you already had an account from previous years, reuse it; if not, create a new account. If, for whatever reason, you cannot use Pretalx, you can also submit your proposal by messaging the Ada-FOSDEM mailing list [15]. If needed, feel free to contact us at the Ada-FOSDEM Mailing list or at <irvise (at) irvise.xyz> (without spaces).

Please, fill the information asked by the Pretalx system, which includes:

- your name, affiliation, contact info;
- the title of your talk (be descriptive and creative);
- a short descriptive and attractive abstract;
- preferred duration of your talk;
- pointers to more information if applicable;
- a short bio and photo.

See programs of previous Ada DevRooms (URLs below) for presentation examples, as well as for the kind of info we need.

Here is the slightly flexible schedule that we will follow:

- November 30, 2024: end of the submission period. Remember, we only need the information in the list above. You do not have to submit the entire talk by this date. Try to submit your proposal as early as possible. It is better to submit half of the details early than be late, so do not wait for the last minute.
- December 15, 2024: announcement of accepted talks.
- January 15, 2025: your slides should be uploaded to the Pretalx platform.
- February 2, 2025: Ada-DevRoom day!

We look forward to lots of feedback and proposals!

Regards,
The Ada-FOSDEM team

Main organiser: Fernando Oleo Blanco <irvise (at) irvise.xyz>

Second in command: Dirk Craeynest <Dirk.Craeynest (at) cs.kuleuven.be>

Third in command: A.J. Ianozi <aj (at) ianozi.com>

[1] https://www.cs.kuleuven.be/~dirk/ada-belgium

[2] https://www.ada-europe.org

[3] https://fosdem.org

[4] https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/3215760/nsa-releases-guidance-on-how-to-protect-against-software-memory-safety-issues/

[5] https://www.adacore.com/papers/nvidia-adoption-of-spark-new-era-in-security-critical-software-development

[6] https://blog.adacore.com/when-formal-verification-with-spark-is-the-strongest-link

[7] https://forum.ada-lang.io/

[8] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/06/060226-fosdem.html

[9] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/09/090207-fosdem.html

[10] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/12/120204-fosdem.html

[11] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/13/130203-fosdem.html

[12] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/14/140201-fosdem.html

[13] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/15/150131-fosdem.html

[14] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/16/160130-fosdem.html

[15] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/18/180203-fosdem.html

[16] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/19/190202-fosdem.html

[17] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/20/200201-fosdem.html

[18] https://www.cs.kuleuven.be/~dirk/ada-belgium/events/22/220206-fosdem.html

[19] http://listserv.cc.kuleuven.be/archives/adafosdem.html

[20] https://pretalx.fosdem.org/fosdem-2025/cfp

From: Fernando Oleo / Irvise <irvise_ml@irvise.xyz>
Date: Sun, 24 Nov 2024 20:14:52 +0100

As written by Dirk in https://forum.ada-lang.io/t/fosdem-call-for-presentations-ada-devroom-fosdem-2025/1386/12

"""
Proposals are very often submitted very late, which always makes the organisers nervous…;)

That said, we received some proposals by now, but certainly would like to get more. So all of you: if you can tell something about a tool, library, application, technique, case study, experience, etc., that is potentially interesting for FOSDEM, submit a proposal for the Ada DevRoom.

Note that proposals are just that: info about a potential presentation, NOT the presentation itself. There's time to prepare that after the submission deadline. For now, only a small amount of work is needed to submit.

You have until the end of November, less than one week from now. But don't delay: please submit sooner, rather than later.

Dirk
"""

Ada Monthly Meetup, 7th December 2024

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: Ada Monthly Meetup, 7th December 2024
Date: Sat, 16 Nov 2024 10:32:01 -0000
Newsgroups: comp.lang.ada

Original posted on Nov 4, 2024 8:48pm at https://forum.ada-lang.io/t/ada-monthly-meetup-7th-december-2024/ by Fernando aka Irvise <irvise_ml@irvise.xyz>.

Hello everybody!
I would like to announce the December (2024) Ada Monthly Meetup which will be taking place on the 7th of December at 14:00 UTC time (15:00 CET). As always the meetup will take place over at Jitsi. The Meetup will also be livestreamed/recorded to YouTube.

If someone would like to propose a talk or a topic, feel free to do so! We currently have no proposals.

Here are the connection details from previous posts: The meetup will take place over at Jitsi, a conferencing software that runs on any modern browser. The link is Jitsi Meet [1]. The room name is "AdaMonthlyMeetup" and in case it asks for a password, it will be set to "AdaRules". I do not want to set up a password, but in case it is needed, it will be the one above without the quotes. The room name is generally not needed as the link should take you directly there, but I want to write it down just in case someone needs it.

Also, this will be the last Monthly Meetup for a long while! There will be none in January 2025 nor one for February, as a large portion of the Ada community will be meeting in FOSDEM [2]!!

Best regards and see you soon!
Fer

P.S: you can see the October summary in "Ada Monthly Meeting October 2024" [3] or in YouTube [4].

[1] https://meet.jit.si/AdaMonthlyMeetup

[2] https://fosdem.org/2025/

[3] https://forum.ada-lang.io/t/ada-monthly-meetup-5th-october-2024/1209/4

[4] https://www.youtube.com/watch?v=hpbXvSAAu30

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Date: Mon, 9 Dec 2024 13:29:35 -0000

Original posted on Dec 7, 2024 5:12pm at https://forum.ada-lang.io/t/ada-monthly-meetup-7th-december-2024/1444/12 by Fernando aka Irvise <irvise_ml@irvise.xyz>.

The meetup is now over. Here are the minutes of the meetup:

* A strong reminder of the Ada Crate of the Year competition [1]. The deadline is approaching quickly! There are three prices, one for Ada, another for SPARK and finally another for embedded system crates! Here is the forum thread covering the topic [2].
* A reminder that Advent of Code 2024 [3] is now live.
  - AdaCore, like last year, is going to donate money [4] based on the amount of solutions and submission that are being done in Ada or SPARK. The submissions need to be done in this forum thread [5]
* The Learn.AdaCore.com website has received several improvements in these past few months [6]. Some changes are detailed in this blog post [7]. Improvements include controlled and limited types and discriminants.
* The Call for Presentations for the FOSDEM [8] conference has now ended. We did get a nice bunch of submissions which we hope to publish in short time.
* An Ada program was recently showcased in HackerNews [9]. Prunt [10] is a motion controller for 3D printers written in Ada. It is open source, so you can go ahead and take a look at the code.
* There was recently a question in the forum [11] about the use of SPARKlib [12]. The library is a set of nice utilities, algorithms and data structures which have been formally verified. I recommend people to check it out!
* For the people who use AWS [13] in Fedora systems, it is recommended that you read this email thread [14] by Björn Persson [15]. It discusses a security vulnerability disclosed by AdaCore.
* For those interested, charlie5 (Rod Kay) is porting the Linux e1000e network driver to Ironclad. Here is the repo with the progress [16]. Feel free to help and lend a hand. For more info, join Ironclad's Matrix chat room [17].
* The WG9, the ISO Work Group behind the Ada standard, had a meeting a few days ago. Some information [18] was shared with regards to the Ada Users Society. The public list of WG9 documents can be found here [19].

A huge thanks to @AJ-Ianozi [20] for showing us his MMO game whose backend is written using Ada and is based on AWS.

There will be no meetup in January as most people are unavailable on those dates. February will also not have a meetup as we will be in FOSDEM!

Best regards to you all!
Fer

P.S: sorry for the technical issues during the meetup!

[1] https://blog.adacore.com/announcing-the-2024-ada-spark-crate-of-the-year-award

[2] https://forum.ada-lang.io/t/2024-crate-of-the-year-awards/923

[3] https://adventofcode.com/

[4] https://blog.adacore.com/announcing-advent-of-ada-2024-coding-for-a-cause

[5] https://forum.ada-lang.io/t/advent-of-code-2024/1500

[6] https://learn.adacore.com/courses/advanced-ada/changelog.html

[7] https://blog.adacore.com/learn-advanced-ada-2024-09

[8] https://fosdem.org/2025/

[9] https://news.ycombinator.com/item?id=42314905

[10] https://600f3559.prunt-docs.pages.dev/

[11] https://forum.ada-lang.io/t/where-is-sparklib/218/4

[12] https://docs.adacore.com/spark2014-docs/html/ug/en/source/spark_libraries.html#

[13] https://github.com/AdaCore/aws

[14] https://lists.fedoraproject.org/archives/list/ada@lists.fedoraproject.org/thread/IYXTYY2SCZQ32U76MVO5GHK52RXVNQJ6/?noscript

[15] https://lists.fedoraproject.org/archives/users/34271505be0745b085f0fd955fb2e3ef/

[16] https://codeberg.org/charlie5/ironclad_intel_e1000e_driver_port.git

[17] https://matrix.to/#/#ironclad:matrix.org" rel="noopener nofollow ugc

[18] https://www.open-std.org/jtc1/sc22/wg9/n654_WG_9_Future_Plan.pdf

[19] https://www.open-std.org/jtc1/sc22/wg9/documents.htm

[20] https://forum.ada-lang.io/u/aj-ianozi

Happy Birthday, Lady Ada!

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: Happy birthday, Lady Ada!
Date: Tue, 10 Dec 2024 16:19:30 -0000
Newsgroups: comp.lang.ada

2024/12/10: birthday of Lady Ada Lovelace, born in 1815, namesake of the #AdaProgramming language.

Happy Programmers' Day!

#AdaEurope #AdaBelgium

https://en.m.wikipedia.org/wiki/Ada_Lovelace

From: Lawrence D'Oliveiro <ldo@nz.invalid>
Date: Tue, 10 Dec 2024 20:59:06 -0000

> 2024/12/10: birthday of Lady Ada Lovelace, born in 1815, namesake of the #AdaProgramming language.

Came across this comic strip <https://sydneypadua.com/2dgoggles/> some years ago: a contrafactual imagining of the adventures of Lovelace and Babbage, getting together to fight crime.

("Crime" being, in Babbage's case, music, and in Lovelace's case, poetry.)

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Date: Wed, 11 Dec 2024 18:18:57 -0000

> Came across this comic strip [...]

Some of the CLA regulars might remember that comic strip. It was mentioned on several occasions here: my CLA archive has postings in 2009, 2011, 2014, and 2023.

Recommended! (I have the book...)
Dirk

From: Simon Wright <simon@pushface.org>
Date: Wed, 11 Dec 2024 19:36:26 +0000

> Recommended! (I have the book...)

I have the first edition... Really good.

For interest, https://github.com/simonjwright/analytical-engine

12th Ada Developer Room at FOSDEM 2025 - Sun 2 Feb - Brussels

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: 12th Ada Developer Room at FOSDEM 2025 - Sun 2 Feb - Brussels
Date: Thu, 19 Dec 2024 17:19:05 -0000
Newsgroups: comp.lang.ada, fr.comp.lang.ada

[CfPart is included in the Forthcoming Events Section —arm]

Ada-Europe Conference - 2nd Call for Contributions - AEiC 2025

From: Dirk Craeynest <dirk@orka.cs.kuleuven.be>
Subject: Ada-Europe Conference - 2nd Call for Contributions - AEiC 2025
Date: Tue, 24 Dec 2024 15:33:00 -0000
Newsgroups: comp.lang.ada,fr.comp.lang.ada,comp.lang.misc

[CfP is included in the Forthcoming Events Section —arm]

Ada-related Resources

[Delta counts are from November 13th to January 29th. —arm]

Ada on Social Media

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Ada on Social Media
Date: 29 Jan 2025 19:03 CET
To: Ada User Journal readership

Ada groups on various social media:

- Reddit: _992 (+124) members [1]
- LinkedIn: 3_580 (+31) members [2]
- Stack Overflow: 2_435 (+9) questions [3]
- Ada-lang.io: 321 (+34) users [4]
- Gitter: 277 (+6) people [5]
- Telegram: 214 (+6) users [6]
- Libera.Chat: 78 (+9) concurrent users [7]

[1] https://old.reddit.com/r/ada/

[2] https://www.linkedin.com/groups/114211/

[3] https://stackoverflow.com/questions/tagged/ada

[4] https://forum.ada-lang.io/u

[5] https://app.gitter.im/#/room/#ada-lang_Lobby:gitter.im

[6] https://t.me/ada_lang

[7] https://netsplit.de/channels/details.php?room=%23ada&net=Libera.Chat

Repositories of Open Source Software

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Repositories of Open Source software
Date: 29 Jan 2025 19:03 CET
To: Ada User Journal readership

GitHub: >1_000* (=) developers [1]
Rosetta Code: 1_008 (+3) examples [2]
42 (=) developers [3]
Alire: 521 (+39) crates [4]
1_315 (+47) releases [5]
Sourceforge: 242 (-9) projects [6]
Open Hub: 214 (=) projects [7]
Codelabs: 61 (+1) repositories [8]
Bitbucket: 37 (=) repositories [9]

*This number is a lower bound due to GitHub search limitations.

[1] https://github.com/search?q=language%3AAda&type=Users

[2] https://rosettacode.org/wiki/Category:Ada

[3] https://rosettacode.org/wiki/Category:Ada_User

[4] https://alire.ada.dev/crates.html

[5] `alr search --list --full`

[6] https://sourceforge.net/directory/language:ada/

[7] https://www.openhub.net/tags?names=ada

[8] https://git.codelabs.ch/?a=project_index

[9] https://bitbucket.org/repo/all?name=ada&language=ada

Language Popularity Rankings

From: Alejandro R. Mosteo <amosteo@unizar.es>
Subject: Ada in language popularity rankings
Date: 29 Jan 2025 19:09 CET
To: Ada User Journal readership

[Positive ranking changes mean to go up in the ranking. —arm]

- PYPL Index: 15 (=) 1.21% (+0.08%) [1]

- TIOBE Index: 26 (-1) 0.65% (-0.06%) [2]

- Stack Overflow Survey: 40 (=) 0.9% (=) [3]

- IEEE Spectrum (trending):46 (=) Score: 0.0022 (=) [4]

- IEEE Spectrum (general): 50 (=) Score: 0.0014 (=) [4]

- IEEE Spectrum (jobs): 55 (=) Score: 0.0 (=) [4]

- Languish Trends: 141 (+12) 0.01% (=) [5]

[1] http://pypl.github.io/PYPL.html

[2] https://www.tiobe.com/tiobe-index/

[3] https://survey.stackoverflow.co/2024/

[4] https://spectrum.ieee.org/top-programming-languages/

[5] https://tjpalmer.github.io/languish/

Re: Adaic.org; Is There a Problem?

[Cont'd from AUJ 45-3, July 2024, this thread discussed the temporary unavailability of the adaic.org website. —arm]

From: Randy Brukardt <randy@rrsoftware.com>
Subject: Re: adaic.org; is there a problem?
Date: Fri, 1 Nov 2024 22:39:35 -0500
Newsgroups: comp.lang.ada

In case anyone is interested:

AdaIC.org was failing because a PHP update borked the Wordpress "theme" that it uses. We had to pay the hosting company to figure out the problem; they rolled back the PHP and all seems well. This is the bane of using "standard" tools for web hosting; they make it easier to create posts, but also are much more attackable and thus get updated and break a lot more often. (I preferred the old mostly Ada+HTML version, but that wasn't my choice.)

Ada-Auth.Org was down because a water main broke near my office building and flooded the entire lower level with 14" of muddy water. (This of course shorted out most of the electrical things; luckily all of the computers themselves were high enough to escape damage.) All of the tenants had to move out immediately (the building had to be gutted to eliminate mold and rot problems). I ended up moving all of the computers to my living room, and had to get the business internet and phone moved as well. (That was a major mess, still not quite cleaned up.) Everything appears to be working properly now.

Today is the first time I've managed to find time to look at comp.lang.ada since mid-September. Someday I might catch up again, but probably not fully until next year.

From: Lawrence D'Oliveiro <ldo@nz.invalid>
Date: Sat, 2 Nov 2024 05:34:36 -0000

> they rolled back the PHP and all seems well.

That's not "figuring out" the problem though, is it? The next time you retry the PHP update, won't it just just happen again?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Wed, 20 Nov 2024 19:35:07 -0600

My understanding is that the tech tried to fix the problem but couldn't figure it out and gave up and rolled back instead. I agree that isn't a real fix, but that's definitely not something I can do anything about. (I have no interest in figuring out PHP!)

Ada-related Tools

GCC 14.2.0-3 (aarch64, MacOS)

From: Simon Wright <simon@pushface.org>
Subject: [ANN] GCC 14.2.0-3 (aarch64, macOS)
Date: Sun, 17 Nov 2024 14:41:33 +0000
Newsgroups: comp.lang.ada

Available at [1]. Included tools at v25.0.0.

[1] https://github.com/simonjwright/distributing-gcc/releases/tag/gcc-14.2.0-3-aarch64

From: Bill Findlay <findlaybill@blueyonder.co.uk>
Date: Thu, 21 Nov 2024 15:30:22 +0000

Hi Simon, many thanks for that.

I note that 14.2.0-3 has the same issue as all previous 14.2 versions: a spurious error message on a simple string concatenation, which appears only when link-time optimization is called for:

```
gnatmake -aI../Source -aO../Build -funwind-tables -gnatl12j96 -gnatw.e -gnatwD -gnatwH -gnatwP -gnatwT -gnatw.W -gnatw.B -gnatwC -gnatw.u -gnatyO -gnatw.Y -gnatw.N -fdata-sections -ffunction-sections -O3 -flto askance -bargs -static -Sin -largs -Wl,-dead_strip -Wl,-dead_strip -largs -flto

...

gnatbind -aI../Source -aO../Build -static -Sin -x askance.ali
gnatlink askance.ali -funwind-tables -fdata-sections -ffunction-sections -O3 -flto -Wl,-dead_strip -Wl,-dead_strip -flto
lto-wrapper: warning: using serial compilation of 4 LTRANS jobs
lto-wrapper: note: see the '-flto' option documentation for more information
/Users/wf/KDF9/emulation/Source/posix.adb: In function 'posix__output_line':
/Users/wf/KDF9/emulation/Source/posix.adb:277:49: warning: '__builtin_memcpy' writing between 1 and 2147483647 bytes into a region of size 0 [-Wstringop-overflow=]
277 | message_line : constant String :=
    |
message & NL;
          ^
```

This is in the routine:

```
procedure output_line (message: in String)
is
  message_line : String := message & NL;
begin
  output(message_line);
end output_line;
```

where:

```
NL : constant String := OS_specifics.EOL;
```

in the package spec., and:

```
package body OS_specifics is
function EOL
return String
is (1 =>  Character'Val(16#0A#));
```

...

Changing the complained-of line to:

```
message_line : String :=
message &  (NL &  "");
```

makes the warning go away.

From: Moi <findlaybill@blueyonder.co.uk>
Date: Thu, 21 Nov 2024 15:45:53 +0000

I forgot to say that I'm running up-to-date Sequoia.

From: Simon Wright <simon@pushface.org>
Date: Thu, 21 Nov 2024 20:40:18 +0000

I can't reproduce.

The only change made to the 14.2.0 compiler release are to do with fixincludes vs SDK16.

-1: gcc-14.2-darwin-r1
-2, -3: gcc-14.2-darwin-r2

From: Moi <findlaybill@blueyonder.co.uk>
Date: Fri, 22 Nov 2024 01:18:36 +0000

> I can't reproduce.

I'm not surprised, it seems to be very dependent on the exact context. I could not reproduce it in a simple example. It has been present in all 14* compilers, but it is absent from 13.2.0.

From: Björn Persson <bjorn@xn--rombobjrn-67a.se>
Date: Fri, 22 Nov 2024 11:02:23 +0100

> warning: '__builtin_memcpy' writing between 1 and 2147483647 bytes into a region of size 0 [-Wstringop-overflow=]

I've seen many occurrences of that bogus warning in Fedora, so it's not unique to Simon's build. Thus the place to report it would be https://gcc.gnu.org/bugzilla/.

From: Moi <findlaybill@blueyonder.co.uk>
Date: Sun, 24 Nov 2024 00:19:06 +0000

Do you get it only in 14.2.0-*, Björn?

From: Simon Wright <simon@pushface.org>
Date: Sun, 24 Nov 2024 11:17:07 +0000

It happens with GCC version 15.0.0 20241102 (experimental) :-(

And with an x86_64-apple-darwin GCC 4.2.0 build (on aarch64, under Rosetta) I noticed

```
In function
'disassembly__data_access_name',
inlined from
'disassembly__the_full_name_of' at
/Users/simon/tmp/emulation/Source/
disassembly.adb:483:44:
/Users/simon/tmp/emulation/Source/disas
sembly.adb:452:48: warning:
'__builtin_memcpy' writing between 1
and 2147483647 bytes into a region of
size 0 overflows the destination
[-Wstringop-overflow=]
452 | modifier  : constant String  :=
      |
M_suffix & Q_suffix;
           ^
/Users/simon/tmp/emulation/Source/
disassembly.adb:452:48: note: destination
object 'S1471b.315' of size 0
452 | modifier  : constant String  :=
      |
M_suffix & Q_suffix;
           ^
/Users/simon/tmp/emulation/Source/disas
sembly.adb:452:48: warning:
'__builtin_memcpy' writing between 1
and 2147483647 bytes into a region of
size 0 overflows the destination [-
Wstringop-overflow=]
452 | modifier  : constant String  :=
      |
M_suffix & Q_suffix;
           ^
/Users/simon/tmp/emulation/Source/disas
sembly.adb:452:48: note: destination
object 'S1471b.315' of size 0
452 | modifier  : constant String  :=
      |
M_suffix & Q_suffix;
           ^
```

From: Björn Persson <bjorn@xn--rombobjrn-67a.se>
Date: Tue, 26 Nov 2024 16:48:31 +0100

> Do you get it only in 14.2.0-*, Björn?

I haven't taken notes of the GCC version. I just concluded that the warning was obviously wrong, and moved on. I can let you know if I see it again.

From: Björn Persson <bjorn@xn--rombobjrn-67a.se>
Date: Fri, 29 Nov 2024 20:08:53 +0100

> Do you get it only in 14.2.0-*, Björn?

It happened today in GCC 14.2.1 (as packaged in Fedora 41), so no, not only in 14.2.0.

From: Simon Wright <simon@pushface.org>
Date: Fri, 29 Nov 2024 20:52:08 +0000

> It happened today in GCC 14.2.1 (as packaged in Fedora 41), so no, not only in 14.2.0.

There's no official FSF 14.2.1 release - it may just be like Alire, which only handles 3 levels, so they call the first packaging of 14.2.0 14.2.1. If you say 'gcc -v' it'll probably say 14.2.0.

Of course I could be completely wrong and Fedora have added lots of value!

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Fri, 29 Nov 2024 13:21:30 -0800

> There's no official FSF 14.2.1 release - it may just be like Alire, which only handles 3 levels, so they call the first packaging of 14.2.0 14.2.1

I see some potential for confusion, since there almost certainly will be an official GCC 14.2.1 release in the near future. That official release will include code that's not included in what Fedora calls gcc 14.2.1. It's a point release, so I wouldn't expect substantial changes, but still, I think Fedora should use a different naming scheme.

From: Simon Wright <simon@pushface.org>
Date: Fri, 29 Nov 2024 22:25:22 +0000

> I see some potential for confusion, since there almost certainly will be an official GCC 14.2.1 release in the near future.

I have this vague memory that 14.2.1 is a draft for 14.3.0, and that GCC never has .1 _releases_. In the same way that 15.0.0 migrates to 15.0.1 before the 15.1.0 release.

Yes, see [1], section "Version Numbering Scheme for GCC 5 and Up".

Also, in gcc-mirror on github, gcc/BASE-VER in the releases/gcc-14 branch [2] holds 14.2.1.

[1] https://gcc.gnu.org/develop.html

[2] https://github.com/gcc-mirror/gcc/blob/releases/gcc-14/gcc/BASE-VER

From: Björn Persson <bjorn@xn--rombobjrn-67a.se>
Date: Fri, 29 Nov 2024 23:52:17 +0100

> If you say 'gcc -v' it'll probably say 14.2.0.

```
$ LANG=en GCC -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexe
c/gcc/x86_64-redhat-linux/14/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-
none:amdgcn-amdhsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-redhat-linux
Configured with: ../configure --enable-
bootstrap --enable-
languages=c,c++,fortran,objc,obj-
c++,Ada,go,d,m2,lto --prefix=/usr --
mandir=/usr/share/man --
infodir=/usr/share/info --with-
bugurl=http://bugzilla.redhat.com/bugzilla
--enable-shared --enable-threads=posix --
enable-checking=release --enable-multilib
--with-system-zlib --enable-__cxa_atexit -
-disable-libunwind-exceptions --enable-
gnu-unique-object --enable-linker-build-
id --with-gcc-major-version-only --
enable-libstdcxx-backtrace --with-
libstdcxx-zoneinfo=/usr/share/zoneinfo --
with-linker-hash-style=gnu --enable-
plugin --enable-initfini-array --with-
isl=/builddir/build/BUILD/gcc-14.2.1-
build/gcc-14.2.1-20240912/obj-x86_64-
redhat-linux/isl-install --enable-offload-
targets=nvptx-none,amdgcn-amdhsa --
enable-offload-defaulted --without-cuda-
driver --enable-gnu-indirect-function --
enable-cet --with-tune=generic --with-
arch_32=i686 --build=x86_64-redhat-
linux --with-build-config=bootstrap-lto --
enable-link-serialization=1
Thread model: posix
Supported LTO compression algorithms:
zlib zstd
gcc version 14.2.1 20240912 (Red Hat
14.2.1-3) (GCC)
```

> I see some potential for confusion, since there almost certainly will be an official GCC 14.2.1 release in the near future. That official release will include code that's not included in what Fedora calls gcc 14.2.1. It's a point release, so I wouldn't expect substantial changes, but still, I think Fedora should use a different naming scheme.

Anyone who needs to know the exact state of the code packaged in Fedora 41 can find the Git revision and all the patches here: https://src.fedoraproject.org/rpms/gcc/blob/f41/f/gcc.spec

If someone has a problem with the version numbering, they could always try

reporting it as a bug, but they should first know that the main admin of the Fedora package is also frequently seen in the revision history at gcc.gnu.org. One who is that deeply involved in the development of GCC probably has a better idea of the differences between versions than most of us do.

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Sat, 30 Nov 2024 13:15:24 -0800

After I wrote the above, I built GCC from its git repo, using the tip of the releases/gcc-14 branch (not a release tag). Presumably the Fedora folks did something similar. The resulting GCC reports its own version as:

$ gcc –version
gcc (GCC) 14.2.1 20241129

whereas a gcc built from the releases/gcc-14.2.0 tag reports:

$ gcc –version
gcc (GCC) 14.2.0

with no date.

I still see some potential for confusion, but not as much as I initially thought. (The fact that GCC doesn't do *.*.1 releases is fairly obscure, and I wouldn't expect most people to know about it.)

The PragmAda Reusable Components

From: Pragmada Software Engineering <pragmada@pragmada.x10hosting.com>
Subject: [Reminder] The PragmAda Reusable Components
Date: Sun, 1 Dec 2024 11:21:34 +0100
Newsgroups: comp.lang.ada

The PragmARCs are a library of (mostly) useful Ada reusable components provided as source code under the GMGPL or BSD 3-Clause license at https://github.com/jrcarter/PragmARC.

This reminder will be posted about every six months so that newcomers become aware of the PragmARCs. I presume that those who want notification when the PragmARCs are updated have used Github's notification mechanism to receive them, so I no longer post update announcements. Anyone who wants to receive notifications without using Github's mechanism should contact me directly.

Vulnerability in AWS.Client

From: Björn Persson <bjorn@xn--rombobjrn-67a.se>
Subject: Anyone using AWS.Client in Fedora? You need Rawhide.
Date: Fri, 6 Dec 2024 19:45:39 +0100
Newsgroups: comp.lang.ada

Anyone who uses the client-side HTTPS functionality of the Ada Web Server library needs to know about CVE-2024-37015. HTTPS requests made with AWS.Client are vulnerable to monster-in-the-middle attacks.

Here's the announcement from AdaCore: https://docs.adacore.com/corp/security-advisories/SEC.AWS-0031-v2.pdf

Although the vulnerability was disclosed in August, version 25.0.0 is the only public release that includes the fix. It is now finally available in Fedora, but only in Rawhide, the development version that will become Fedora 42.

The fix comes with API changes that make it difficult to backport to older versions. That also means that programs using AWS will probably need to be adapted to use version 25. Furthermore, AWS 25 needs Gnatcoll 25, and as usual each new library version has a new soname. If we would push AWS 25 and Gnatcoll 25 as updates to Fedora 40 and 41, then any programs using Gnatcoll would stop working when users install the update, even if they have nothing to do with AWS. That would be bad.

Thus, AWS.Client in Fedora 40 and 41 should not be used except on isolated networks where everything on the network is fully trusted. Only in Rawhide is AWS.Client suitable for use on the Internet.

If you run programs in Fedora that use AWS.Client on the Internet, these are your options:

1: Install Rawhide and follow the development version, accepting the instability and the higher maintenance burden, until Fedora 42 is released. Adapt your programs to the API changes in AWS 25. Recompile more or less all of your own programs. Expect further recompilations before the release date, such as when the soname of Libgnat will change sometime in January.

2: Download the source RPM packages of AWS 25 and Gnatcoll 25 from Rawhide, and compile them yourself on Fedora 41. Adapt your programs to the API changes, and also recompile anything that uses Gnatcoll.

This situation is not how I wish it were, but there are limits to what packagers can do when the upstream developers don't make clean bugfix releases.

From: Niocláiſín Cóilín De Ġloſtéir <master_fontaine_is_dishonest@strand_in_london.gov.uk>
Date: Fri, 6 Dec 2024 20:01:49 +0100

"[. . .]

Furthermore, AWS 25 needs Gnatcoll 25, and as usual each new library version has a new soname. If we would push AWS 25 and Gnatcoll 25 as updates to Fedora 40 and 41, then any programs using Gnatcoll would stop working when users install the update, even if they have nothing to do with AWS. That would be bad."

An electronic engineer who also used to be a GNU/Linux administrator had already perceived this problem of shared libraries in the Year 2006.

From: Lawrence D'Oliveiro <ldo@nz.invalid>
Date: Fri, 6 Dec 2024 20:15:29 -0000

> An electronic engineer who also used to be a GNU/Linux administrator had already perceived this problem of shared libraries in the Year 2006.

Note the soname part. That only increments when there are backward-incompatible changes to the ABI. Otherwise you could install the new version, and existing compiled code would run against it just fine.

Consider the transition in the Linux world from libc5 to GNU libc6. That happened over 20 years ago. Isn’t time for libc7? No, because they have been able to keep all the changes backward-ABI-compatible so far.

References to Publications

AdaCore’s 30th Anniversary

From: Alejandro R. Mosteo
Subject: AdaCore’s 30th anniversary
Date: Wed, 29 Jan 2025 21:06 +0100
To: Ada User Journal readership

AdaCore, one of the more prominent companies in the Ada world, has celebrated its 30th anniversary in 2024. A number of blog entries have been published on its website; since these are rather long, I include here the references and a brief excerpt. Interested readers can find the complete entries at the company’s website by following the provided links.

July 15, 2024: Celebrating 30 years of AdaCore [1]:

This press release revisits AdaCore's 30-year legacy of tools for reliable, safe, and secure software, highlighting its pioneering work with the Ada language, open-source commitment, and expansion into languages like Rust and technologies like SPARK/Ada.

October 8, 2024: AdaCore Memories: the stories behind the first 30 years of AdaCore [2]:

This blog entry presents an interview with co-founders Ed Schonberg and Richard Kenner, in which they reminisce about the beginnings of the company.

December 11, 2024: AdaCore's 2024; Highlights from our 30th year in Business [3]:

This blog post provides an overview of AdaCore's activities in 2024, during its 30th anniversary, including some key projects and the release of the GNAT Pro 25 toolsuite.

[1] https://www.adacore.com/press/celebrating-30-years-of-adacore

[2] https://blog.adacore.com/adacore-memories-the-stories-behind-the-first-30-years-of-adacore

[3] https://blog.adacore.com/adacores-2024-highlights-from-our-30th-year-in-business

Ada Practice

'Reduce from Right to Left

From: b.mcguinness747@gmail.com (Brian9000)
Subject: How can I make Reduce run from right to left?
Date: Tue, 10 Dec 2024 17:20:53 +0000
Newsgroups: comp.lang.ada

I am trying to find a way to make Reduce operate from right to left like APL reduction, but this doesn't seem to work.

I wrote a test program:

```
pragma Ada_2022;
pragma Extensions_Allowed (On);

with Ada.Text_IO;

procedure Reverse_Reduction is
  type Real is digits 15;
  type Vector is array(Natural range <>)
  of Real;

  data : constant Vector := [for i in 0 .. 11 =>
  Real (i + 1)];
begin
  for i in data'Range loop
     Ada.Text_IO.Put_Line (i'Image & ": " &
     data(i)'Image);
  end loop;
  Ada.Text_IO.New_Line;
  Ada.Text_IO.Put_Line (Real'Image
   ([ for i in 1 .. data'Last => data(i)
   ]'Reduce("-", data(0))));
  Ada.Text_IO.Put_Line (Real'Image
  ([ for i in reverse 0 .. data'Last -
   1 => data(i) ]'Reduce
   ("-", data(data'Last))));
end Reverse_Reduction;
```

but the compiler won't accept "reverse" in the next-to-last line, even though you can use it in a normal loop. The other lines compile and run without any problems.

```
$ gnatmake reverse_reduction.adb
x86_64-linux-gnu-gcc-12 -c
reverse_reduction.adb
reverse_reduction.adb:18:47: error: missing
operand
x86_64-linux-gnu-gnatmake-12:
"reverse_reduction.adb" compilation error
```

Of course I can just write a normal loop to do this, but I wanted to see if I could do it with Reduce.

From: J-P. Rosen <rosen@adalog.fr>
Date: Tue, 10 Dec 2024 18:54:27 +0100

ARM 2022/4.5.10(6/5):
The iterated_element_association of a value_sequence shall not have a key_expression, nor shall it have a loop_parameter_specification that has the reserved word reverse.

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Date: Tue, 10 Dec 2024 22:04:02 +0100

> Ada.Text_IO.Put_Line (Real'Image ([for i in reverse 0 .. data'Last - 1 => data(i)]'Reduce("-", data(data'Last))));

Presumably you could do

```
[for I in 0 .. Data'Last - 1 =>
Data (Data'Last - 1 - I)]'Reduce
```

Or write a Reverse function

```
Reverse (Data (0 .. Data'Last - 1) )'Reduce
```

From: b.mcguinness747@gmail.com (Brian9000)
Date: Wed, 11 Dec 2024 01:38:59 +0000

I finally got an APL-like reduction using Ada Reduce:

```
pragma Ada_2022;
pragma Extensions_Allowed (On);

with Ada.Text_IO;

procedure Reverse_Reduction is
  type Real is digits 15;
  type Vector is array(Natural range <>)
  of Real;

  data : constant Vector := [for i in 0 .. 11 =>
  Real (i + 1)];

  function Minus (Accumulator : Real;
   Value : Real) return Real is
   (Value –  Accumulator);
begin
  for i in data'Range loop
     Ada.Text_IO.Put_Line (i'Image & ": " &
     data(i)'Image);
  end loop;
  Ada.Text_IO.New_Line;
  Ada.Text_IO.Put_Line (Real'Image
   ([ for i in 0 .. data'Last - 1 =>
   data(data'Last - 1 - i) ]'Reduce(Minus,
   data(data'Last))));
end Reverse_Reduction;
```

```
$ ./reverse_reduction
0:  1.00000000000000E+00
1:  2.00000000000000E+00
2:  3.00000000000000E+00
3:  4.00000000000000E+00
4:  5.00000000000000E+00
5:  6.00000000000000E+00
6:  7.00000000000000E+00
7:  8.00000000000000E+00
8:  9.00000000000000E+00
9:  1.00000000000000E+01
10:  1.10000000000000E+01
11:  1.20000000000000E+01

-6.00000000000000E+00
```

In GNU APL:

```
      ⍳12
1 2 3 4 5 6 7 8 9 10 11 12
      -/⍳12
_6
```

Re: Ichbiah 2022 Compiler Mode

[Cont'd from AUJ 45-3, July 2024, this thread discussed hypothetical enhancements Jean Ichbiah might have wanted to see on modern Ada revisions. —arm]

From: Lioneldraghi <lionel.draghi@free.fr>
Subject: Re: Ichbiah 2022 compiler mode
Date: Sat, 21 Dec 2024 00:26:06 +0100
Newsgroups: comp.lang.ada

I am sensible to the complexity vs use balance, and I would easily cope with the negative consequences of most of the simplifications you propose (at least those I understand!) but obviously it's hard to be fully aware of the consequences.

> (3) A number of syntax options are eliminated. Matching identifiers are required at the end of subprograms and packages. Initializers are always required (<> can be used if default initialization is needed). Keyword "variable" is needed to declare variables (we do not want the worst option to be the easiest to write, as it is in Ada).

Why are you considering variables worse than constants?

I don't want the "worst" option to be the easiest to write, but neither do I want to put one more keyword in the most common case.

Note that:

1. I have no statistics, but it seems to me that there are more variables than constants in my code.
2. I say "Useless" from my coder point of view, I don't know if it simplifies the work for compiler or tools implementers.

From: Lioneldraghi <lionel.draghi@free.fr>
Date: Sat, 21 Dec 2024 01:52:43 +0100

> (10) Variable-returning functions are introduced. They're pretty similar to the semantics of anonymous access returns (or the aliased function returns suggested by Tucker). This means that a variable can easily be treated as a

> function (and indeed, a variable declaration is just syntactic sugar for such a function).

I suppose that to allows the compiler to discriminate this nonsense code

> Square (2) := 3;

from the legitimate

> List.Index (3) := Item;

you will have to introduce some specific syntax, like Tucker's "aliased function".

I see the huge benefit from a user point of view, but I'm not aware of compiler internals: doesn't the introduction of a second function type increase complexity?

From: Randy Brukardt <randy@rrsoftware.com>
Date: Sat, 21 Dec 2024 02:14:00 -0600

> Why are you considering variables worse than constants? I don't want the "worst" option to be the easiest to write, but neither do I want to put one more keyword in the most common case.

A lot of "variables" in code actually are only written once. In Ada, those are better modeled as constants. A constant tells the reader that the value doesn't change during the life of the object, which is easier for analysis (both human and machine).

Secondly, I am assuming that automation is helping to write a lot of code. "One more keyword" is irrelevant in terms of creating code, the only question is whether it hurts readability. I prefer to have most things written explicitly (but not to the point of noise, of course). That seems especially true if the code is being written by a program and mostly you are trying to figure out why it doesn't work!

> Note that:

> 1. I have no statistics, but it seems to me that there is more variables than constants in my code.

But how many of them *have* to be variables vs. the number that just are because it is easier? I know I have a number of the latter.

> 2. I say "Useless" from my coder point of view, I don't know if it simplifies the work for compiler or tools implementers.

Constants do help the compiler generate better code, although a lot of the benefits can be gained also by working harder. (That's what C compilers do, after all.)

From: Randy Brukardt <randy@rrsoftware.com>
Date: Sat, 21 Dec 2024 02:19:09 -0600

>> (10) Variable-returning functions are introduced. [...]

> I suppose that to allows the compiler to discriminate this nonsense code

> Square (2) := 3;

> from the legitimate

> List.Index (3) := Item;

> you will have to introduce some specific syntax, like Tucker's "aliased function".

> I see the huge benefit from a user point of view, but I'm not aware of compiler internals: doesn't the introduction of a second function type increase complexity?

Yes, but Ada already has a bunch of different mechanisms for dealing with objects/functions/exceptions/packages/types. My intent is to collapse those all into one (somewhat more complex) mechanism. The basic idea is that everything resolves a single way, meaning that everything can be overloaded, and there no longer is a semantic difference between:

```
A : constant T := ...;
```

and

```
function A return T is (...);
```

Whether that really helps remains to be seen, of course. But the goal is to reduce the number of disjoint mechanisms both in the language description and in the implementation. The hope is then to be able to introduce additional capabilities on top of a simpler and stronger foundation.

From: Jeffrey R.Carter <spam.jrcarter.not@spam.acm.org.not>
Date: Sat, 21 Dec 2024 10:50:41 +0100

> "One more keyword" is irrelevant in terms of creating code, the only question is whether it hurts readability.

More keywords = fewer words that can be used as identifiers so more keywords make writing code a little bit harder than fewer keywords.

> But how many of them *have* to be variables vs. the number that just are because it is easier? I know I have a number of the latter.

I put a lot of effort into making sure that all constants are so declared, because I have the rule that (with certain exceptions) no non-local variables may be referenced from subprograms, but constants may be referenced from anywhere. However, I sometimes have constants that cannot be initialized with a single expression, resulting in

```
C : T; -- Constant after initialization
```

Once C has been initialized, I treat it as a constant. Would your approach allow the compiler to know that C is really a constant?

From: G.B. <bauhaus@notmyhomepage.invalid>
Date: Sat, 21 Dec 2024 18:19:12 +0100

> Constants do help the compiler generate better code, although a lot of the benefits can be gained also by working harder. (That's what C compilers do, after all.)

What are some compilers offering today? That is, can they find declarations of variables that could be constants, if so instructed?

I am seeing some warnings about non-initialized variables for a meaningless mock-up, but not much else. Ada, C++, Java.

(Maybe there are options that I have missed. Or an analysis of a whole program yields more.)

```
function testc (b : Boolean) return Integer is

package P is
  x : Integer;
end;

begin
  if b then
          P.x := 42;
  end if;
  return P.x;
end testc;
```

```
int testc(bool b) {
   struct {
     int x;
  } P;
  if (b) {
    P.x = 42;
  }
  return P.x;
}

class testc {
  class P {
    int x;
  }
  P P;
  int $(boolean b) {
     if (b) {
       P.x = 42;
      }
      return P.x;
  }
}
```

From: Chris Townley <news@cct-net.co.uk>
Date: Sat, 21 Dec 2024 17:35:30 +0000

> [Code sample from previous post. – arm]

My understanding is that many compilers will optimise these, and if trivial numbers will 'optimise' out the variable. Confusing in the debugger!

From: Keith Thompson <keith.s.thompson+u@gmail.com>
Date: Sat, 21 Dec 2024 13:26:06 -0800

> A lot of "variables" in code actually are only written once. In Ada, those are better modeled as constants. A constant tells the reader that the value doesn't

> change during the life of the object, which is easier for analysis (both human and machine).

Agreed. But my understanding is that compilers typically do this kind of analysis anyway, at least when optimization is enabled. For example, if I write:

```
N : Integer := 42;
```

and later refer to the value of N, if the compiler is able to prove to itself that N is never modified after its initialization, it can replace a reference to N with the constant 42 (and possibly fold it into other constant expressions).

Using "constant" for something that isn't going to be modified is good practice, but I'd say it's for the benefit of the human reader.

From: Pascal Obry <pascal@obry.net>
Date: Sun, 22 Dec 2024 10:32:20 +0100

> Using "constant" for something that isn't going to be modified is good practice, but I'd say it's for the benefit of the human reader.

Exactly the point of Randy. And as Jean-Pierre would say "a program is written once and read many times" (or something like that).

So having "constant" is a very important point when reading the code for maintenance. Actually a developer doesn't care about the compiler, what is important is the "visible" semantic to the human reader.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Mon, 23 Dec 2024 19:00:32 -0600

> I put a lot of effort into making sure that all constants are so declared, because I have the rule that (with certain exceptions) no non-local variables may be referenced from subprograms, but constants may be referenced from anywhere.

Precisely. The idea is to encourage use of constants by eliminating the unnatural advantage to writing uninitialized variables. If everything is equally easy/hard to write, then one is more likely to make the best choice for the program.

> However, I sometimes have constants that cannot be initialized with a single expression, resulting in

> C : T; -- Constant after initialization

> Once C has been initialized, I treat it as a constant. Would your approach allow the compiler to know that C is really a constant?

Not with the approach I was envisioning. Of course, Ada 2022 and beyond already make it possible to initialize a lot more objects (especially with the introduction of container aggregates), so hopefully it will be less necessary to write things like your example.

From: Randy Brukardt <randy@rrsoftware.com>
Date: Mon, 23 Dec 2024 19:03:53 -0600

> What are some compilers offering today? That is, can they find declarations of variables that could be constants, if so instructed?

I was thinking from a code generation perspective, as opposed to static analysis. These are really the same process, but in one case the output is only for a computer, and in the other, the output is for a human. The needs of those outputs are quite different. I think a lot of compilers do optimizations of constants by discovery, but few compilers do much beyond rudimentary static analysis. I expect that to change, but I may be an optimist on that...