

Applause from Nicholas Pinn, Al Graham, MS(Eng), and 5 others



Adrian D. Finlay

@thewiprogrammer. Writer @hackernoon. Code, LOTS of it. Mangos, LOVE THEM! Barbering. Health. Travel. Business. & more! Network w/ me @ adriandavid.me/network

Oct 17, 2017 · 13 min read

The Most Influential Person You've Probably Never Heard Of (& He's Everywhere...) 🐼

Dennis Ritchie
1941 - 2011



Dennis MacAlistair Ritchie [1]

Six years and five days ago from today, October 17 2017, the planet lost a legend. October 12, 2011.

Screw all the formalities. Dennis Ritchie is everywhere. You look at him everyday and you (probably) totally have no clue. You merely relish in his gifts with little to no appreciation of his significance and importance. Many of you before reading this article may probably have died not knowing or not fully understanding how important he was.

Ritchie was a Harvard educated American Computer Scientist whose primary gifts to us were the **C Programming Language** and the **UNIX Operating System**. These two technologies are arguably the most (or

among the most) influential technologies of the computing age. We should also thank **AT&T** as well for the Bell Research Lab wherein the research that led to C & UNIX took place as well as the several other engineers who worked on both of these technologies (notably, **Ken Thompson** & Brian Kernighan) in its inception. **Ken Thompson** was Ritchie's right hand man and was **the original principal author of the UNIX Operating System**. Like Ritchie, he is another individual that is largely taken for granted. Indubitably, major revolutions in software are rarely (if ever) the consequence of the work of one person and are rarely composed by majority of one individual's code for a long period of time.

The Linux Kernel, for example, created by Finnish engineer Linus Torvalds, is currently composed of 1% of code written by Linus himself. And Linux is just a OS kernel. Most popular "Linux Distributions" are operating systems that are the product of GNU tools, the Linux Kernel, & various other pieces of open source software. Consequently, a typical "Linux Distribution" would likely contain less than ~1% of code written by Torvalds. And guess what else? Linux is C-based and UNIX-like.

Ritchie's work is probably in your pocket or on your lap. You know, in that smartphone or tablet you love so dearly. It's incontrovertibly in your consumer laptop or PC. It might be in your nearby traffic light. It or its likeness is probably in your dishwasher, refrigerator, or car. Your cashier likely uses it when they check you out at the cash register. Ritchie's work is even found in the very platform we're having this discussion on—**Medium**. Medium's technology stack includes Node.js (which sits atop Chrome V8) and Go. [2] The relevance? Node.js is written in C, C++, & JavaScript (which is itself most commonly implemented by an interpreter written in C and/or C++). **Needless to say, C++ would not be here without C**. Chrome V8 is the JavaScript engine powering Google Chrome (and other browsers). Both tools are written in C++. The first web browser was written in Objective-C—a language that is a strict superset of C. All of the major web browsers are written in C or a C family language. Viewing this webpage from an Apple machine? That is a UNIX machine.

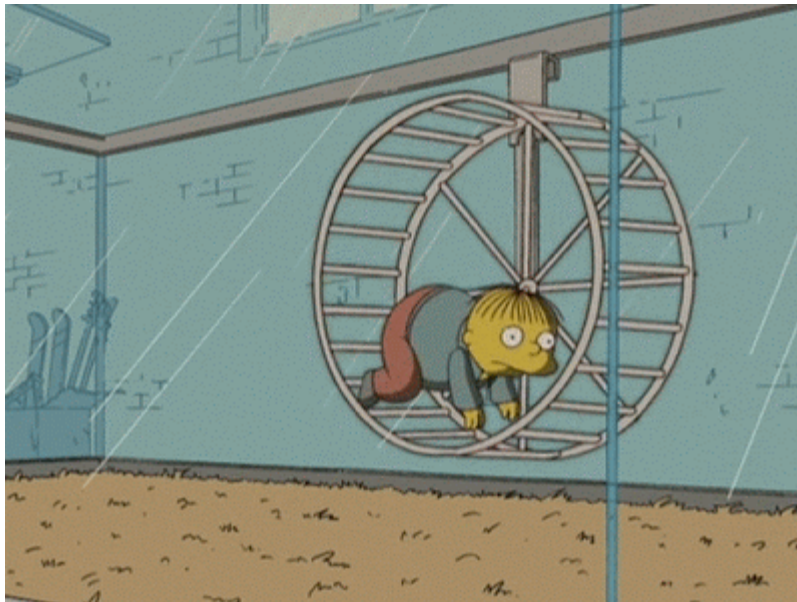
| *What about Go?*

Remember that Ken Thompson guy we briefly touched on? Well he went on to design the Go Programming Language. The language is (to no surprise) heavily influenced by C and the compilers for this language were written in C & C++ and up until only recently has it become self-hosting (written in itself).

As you can see—we are using Dennis Ritchie’s work **right now** to communicate on medium and over the web. And many **may never realize this**. The impregnable truth is that Ritchie’s work and work influenced by Ritchie is found **EVERYWHERE**.

E-V-E-R-Y-W-H-E-R-E.

Live footage of you trying to run away from UNIX and C...



Literary Carrie — Blogspot



Pinterest — Jack Sparrow Pin

Wait... what's C and what's UNIX???

I thought you would ask.

The **C Programming Language** is a general purpose programming language that was designed by Dennis Ritchie in 1969–1973 and published in 1973. It was designed to be a sort of **portable assembler**—providing language elements that very efficiently correspond with typical machine instructions. It grew out of the need to make the UNIX OS more portable; UNIX was given a fresh rewrite in (mostly) C. Instead of writing UNIX in various assembler codes for various architectures, one could instead write UNIX in C and the C compiler would spit out architecture specific code for that specific platform. The C Language is one of the most popular and widely used computer languages of all time.

C was designed to be fast, efficient, portable, lightweight, & so close to the hardware as to be considered a very light layer of abstraction (just a step) above the assembler. A consequence of the achievement of these goals and the work done to optimize C code over the years is that a C compiler is the most common compiler available for the majority of architectures. Often (or as it used to be) a platform, if it has support for nothing else, will have a C compiler. So widespread and so successful was C, that the majority of languages, if they provide an interface/bridge for nothing else, will all include a means to interface with C code. Because of its success in producing efficient machine code (among other things), C is generally considered to be an extremely successful systems language.

In my estimation, **C is probably the most influential language to modern computing**. It is that important. If languages have not used C for their own implementations, they have been influenced by C or other C descendant languages for their own language structures and features. Languages typically use C to implement their compilers because C has (over decades of work) been battle tested and made extremely efficient. C already maps efficiently to the hardware and has been ported to nearly every architecture, making it a perfect candidate for designing a language implementation.

C is a descendant of B, a language originally written by Ken Thompson for early UNIX with later contribution by Dennis Ritchie. B wound up being insufficient for the engineering needs of the UNIX project and

Ritchie (alongside Thompson, Kerninghan, & others) designed C based on B. **C borrows heavily from B** [5]. B itself is an ideological descendant of BCPL, which is an ideological descendant of CPL, which is an ideological descendant of the ALGOL languages that competed with Fortran in the late 1950s.

Because of C's widespread adoption and implementation on virtually every architecture and Operating System, C's presence has been felt in almost all of modern computing. If a language is not directly using C as part of their implementation, it may be using a language feature inspired or taken from C.

The net effect is that **most of the software that we regularly consume** is either written in C directly, built by another language that depends on C (as many do), or built by a language that is influenced by C or borrows C language features. A side-effect of this is the fact that **without Dennis Ritchie's work, a gigaton of software would not exist**. You can (probably) trace (in some way) a majority of software back to Ritchie's work.

I grant that software development may have continued its natural evolution without Ritchie and other solutions may have been created—but **that is not the present reality**. The reality is that we **live, breath, depend, & build upon the work of Dennis Ritchie. His work is ubiquitous**. It's been to the moon, it's helped save lives, it's helped solve crimes, and we probably see it or its likeness every day. C is the *lingua franca* of the programming universe.

The **Unix Operating System** is defined as follows: “Unix is a portable, multitasking, multiuser, time-sharing operating system (OS) originally developed in 1969 by a group of employees at AT&T. Unix was first programmed in assembly language but was reprogrammed in C in 1973.”[3]

The original AT&T Unix was an Operating System that arose from the abandonment of the MULTICS Operating System Project by Bell Lab. After a frustrated Bell Labs withdrew from MULTICS, the remaining engineers, led by Thompson & Ritchie, decided to create their own operating system.

Ritchie, while working with the MULTICS team at MIT, never actually wrote much code for MULTICS [3]. While he was a member of the

MULTICS team, his role primarily concerned Documentation. Ken Thompson, on the other hand was more intimately involved with MULTICS and CTSS [6].



Ken Thompson & Dennis Ritchie [5]

The original UNIX was more driven by ideas of Ken Thompson rather than Ritchie as Ken was most familiar with the MULTICS code base, admired the MULTICS hierarchical file-system, and had long desired to build a system of his own [4].

“...the fact that Ken had always wanted to write an operating system of his own was what led fairly directly to Unix” (Ritchie)[4]”.

The dissolution of the Bell Lab’s team from the MULTICS project, the desire for an interactive, portable MULTICS, and Thompson’s desire to create his own system all lead to the emergence of the UNIX operating system [5].

The project was named UNIX as a pun on MULTICS by either Brian Kerninghan or Peter Nuemann. It wasn’t until 1972 that UNIX was rewritten in the C programming language[7]. Unix tools were designed to do one thing and do it well. This notion is part of what is known as the UNIX philosophy.

UNIX is important for near innumerable reasons. One is that it popularized many existing techniques used in the development of operating systems (such as the hierarchical file system and shell found in MULTICS) and innovated many others. Another, and arguably the strongest reason for it’s ubiquitous availability was the success of the C programming language (in tandem with UNIX). UNIX is an interactive,

highly modular OS that popularized file system communication and that was **very highly portable** because of the C programming language without compromising performance. This (among other things) led to the widespread deployment of UNIX.

A consequence of UNIX's successes is its influence on virtually all of modern computing.

Unfortunately, due to legal and licensing complications, not everyone could afford or acquire UNIX or BSD (Berkley UNIX). UNIX-like OSs like MINIX, too, had restrictive license stipulations. Consequently, efforts such as GNU+Linux were born. Many consider the emergence of GNU and Linux to be the birth and revolution periods of Free & Open Source Software (FOSS), for which the Harvard educated engineer and FOSS advocate Richard Stallman is given a lot of credit. Almost wholly in the likeness of UNIX, GNU/Linux provided a free as in free cheese environment similar to UNIX. Over time, and because of the global collaborative efforts of the FOSS community, GNU+Linux gradually began to replace areas where UNIX had once been heavily used. Had UNIX been open source, however, Linux may very well have not existed.

OK... What is the impact of C and UNIX?

If one were to keep a long list of all the pieces of technology that are direct or indirect derivatives of C and UNIX one would be writing for several lifetimes.

The short answer: **There impact is felt almost everywhere.**



Spongebob — "Everywhere" [8]

Where is C felt...

- The vast majority of **modern programming languages** either A) used C directly or indirectly or B) were influenced by C. This includes but its not limited to: **Ratfor, C Shell, C++ AMPL, Objective-C, C*, Perl, Java, S-Lang, SAC, Alef, Limbo, PHP, JavaScript/ECMAScript, C—, C#, Ch, D, eC, Cyclone, LSL, Squirrel, Go, OpenCL C, C0, Swift, AWK, BitC, LPC, Pike, Seed7, Processing, Split-C, Unified Parallel C, Clik, Chapel, Fortress, Agora, Falcon, Nim, Nermle, ApeScript, Amiga E, Lite-C, NEwsqueak, NXC, NQC, Oak, PROMAL, Handel-C, Dart, CINT, Cg, nesC, R, Hack, Charm, Claire, Noop, Neko, Axum, Umple, TOM, Telescript, Fantom, etc and their derivatives [9]; Python, D, Go, Julia, Verilog, Rust, Vala, and so on.....All but two of the TIOBE Index's most popular programming languages are C-related.**
- The overwhelming majority of **modern operating systems** use C or C++ extensively. This includes: **Windows, macOS/Mac OSX, Unix, BSD, & GNU/Linux, Solaris.** C and C++ are the *lingua franca* of systems development. There isn't a single operating system in existence with more than 1% market share in the PC market that isn't written in C or a C-family language. This includes mobile phone operating systems such as **Android, iOS, J2ME** etc

—all C based with the exception of J2ME which is Java based, which in turn is implemented by a compiler written in C or C ++ (for example HotSpot).

- Owing to his efficiency and ubiquitous availability, **C has been used in the development of programming language** compiler, libraries, interpreters, and other tools.
- Applications Development, Device Drivers, Networking Software, Databases, Systems Utilities, Misc. Systems Software, Embedded Software. Traffic Lights. Refrigerators. **Everywhere.**

The Takeaway: Even if C isn't directly behind your favourite application or service, it's behind or related to the tools use to provide that service. According to the most recent TIOBE index as of the posting of this article, C is the second most popular programming language.

Where hasn't C touched?: Given the sheer vastness of the computing domain, lots of places. But it far less common that something hasn't been touched by C than that it has. FORTRAN, Lisp, COBOL, & Pascal are examples of prominent Non-C related languages.



"Ken Thompson (sitting) and Dennis Ritchie working together at a PDP-11" [10]

Where is UNIX felt...

- The vast majority of **modern operating systems** are influenced by UNIX, are UNIX-like, or are UNIX derivatives. This includes Linux, which is the closest thing non-UNIX to UNIX, macOS, which is SUS-UNIX certified owing to its hybrid kernel of BSD and Mach, Windows, whose Windows-NT kernel is indirectly influenced by UNIX, and many, many more.
- **Academia & Research.** UNIX and the Bell Labs where it was invented is responsible for
- **Resource constrained system.** The design of truly tiny systems is influenced by UNIX.
- **Almost every operating system you've come across.** Even the tiny ones in your set-top boxes. Almost. Not quite all, though. Windows, for example, is not generally considered to be in the UNIX tradition though Windows NT borrows UNIX and the Windows OS borrows significantly from concepts that UNIX popularizes such as the hierarchical file system and the shell, which are considered ubiquitous and universal.

The systems you use and the stuff running on the systems you use are highly likely to be the descendants of C or Unix. **You can't walk a single mile in a densely populated urban US area, without running into Ritchie's work.** If you can, and you can prove it, challenge me in the comments.

A Brief Note on Ken Thompson—A Legend In His Own Right

Throughout all this talk of Dennis Ritchie, a few points need to be realized.

- While Thompson & Ritchie were the Batman and Robin of UNIX, **Thompson was clearly Batman. At least in the beginning.** Without him, UNIX may have never existed or existed in the way that he did. Thompson was the MULTICS & CTSS developer & Thompson had the burning desire to create his own system per Ritchie's own words[4]. Per Ritchie's own word, most of the initial ideas on UNIX were Thompson's [4]. Ritchie's desire came from wanting an interactive system like MULTICS and joining Thompson's efforts while sharing most of his sentiments.

Thompson is just as unsung as Ritchie, if not more unsung in light of the creation of his later very important works, like UTF-8, which allows you the luxury of viewing the emojis on your devices and text in various languages. You probably didn't even know Ken Thompson helped you with your emojis. More on that later.

- While C was an invention of Dennis Ritchie, it borrowed **heavily** from B, a language designed by Ken Thompson. **There would be no C without B**, or if there would it would incontrovertibly not had the semblance of its current form. **Ritchie took an existing language engineered by Thompson** and made several modifications to it. Had Thompson's B not been word oriented, we may have never seen a C. Maybe I'm wrong, but that is food for thought.
- We talked about how you probably don't know about Dennis Ritchie. But he is indubitably more popular than Ken Thompson. **Thompson, is perhaps, an even greater unsung hero**, who's name may be forgotten by all but the curious, which most of us are not. We will reap from his intellectual gifts forever, and most of us will likely never thank him let alone know who he is.

All in all. Tell your friends. Tell your family. Tell the people across the street. Tell the world the hidden truth behind their technology.

And tell me **WHY** media coverage on Dennis Ritchie and Ken Thompson is so scant. It's shameful. It's pathetic. On the day of the sixth anniversary of Ritchie's passing, I saw **little to no media coverage of Ritchie by the major news networks** here in the United States. **Tell me why that is the case in the comments below. Compare his coverage to the coverage of Steve Jobs, who died exactly one week earlier. Tell me your thoughts below.**

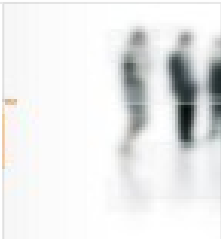
You now know that the likelihood that technology that the media companies use on a daily basis is the brain child of Ritchie/Thompson is almost certain. You'd bet your life on it. **Do you really think Ritchie & Thompson are revered, celebrated, and remembered in proportion to their impact?**

He gave us C, he gave us UNIX, we see him every day and we seemed to have forgotten, we seemed to have taken him for granted. How quickly we forget.



We Will Remember Them

Interested in Java? Join my Java group on Facebook:

<p>Join My Java Facebook Group</p> <p>Interested in Java? Check out my Facebook Group: Java Software Development Group!</p> <p>medium.com</p>	
--	--

Like my Content? Subscribe to my mailing list:

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

[Learn More about Medium's DNT policy](#)

Don't forget to give it a.... ;)



!Emoji.com

Works Cited

[1]—Chippewa Valley Technical College's MSDN Portal

[2]—Medium Engineering: The Stack That Helped Medium Drive 2.6 Millennia of Reading Time. Pupius, Dan

[3]—Techopedia UNIX Definition

[4]—Princeton Interview with Dennis Ritchie

[5]—Photo(1): Ken Thompson & Dennis Ritchie

[6]—Princeton Interview with Ken Thompson

[7]—UNIX, Wikipedia

[8]—The Science of Not Science Tumblr

[9]—C Family Programming Languages, Wikipedia

[10]—Photo(2): Ken Thompson & Dennis Ritchie

