nedeljainformatike v6.0

# Kula od karata softverskog inženjerstva iliti How I learned to stop worrying and love the code

Lazar Mitrović

Matematička gimnazija

22. 04. 2021.

# Teri Dejvis

**Terry A. Davis**

Davis with his computer monitor, circa mid-1980s

| | |
|---|---|
| **Born** | Terrence Andrew Davis<br>December 15, 1969<br>West Allis, Wisconsin, U.S. |
| **Died** | August 11, 2018 (aged 48)<br>The Dalles, Oregon, U.S. |
| **Education** | Bachelor's degree in computer engineering (1992) and master's degree in electrical engineering (1994) from Arizona State University |
| **Occupation** | Computer programmer · video blogger |
| **Employer** | Ticketmaster (1990–1996) |
| **Known for** | TempleOS |
| **Website** | templeos.org |

# TempleOS

- x86_64 (AMD64)
- Sav programski kod u prva 4GB
- Nema virtuelne memorije, hipervizora, ASLR...
- Isključivo 640x480 rezolucija u 16 bita
- Sav kod se izvršava u Ring 0
- Sadrži svoj JIT kompajler za HolyC, self hosted je, koristi originalni RedSea FS, sadrži softverski 3D renderer
- Nema interop sa drugim jezicima, GC, Networking
- Custom 8-bit ekstenzija ASCII koja podržava custom čak i 3D grafiku u charset-u.

Po rečima autora, to je operativni sistem koji omogućava nešto slično C64 na PC.

Terry uses the following analogy:

- Linux is a semi-truck with 20 gears to operate.
- Windows is more like a car.
- TempleOS is a motorbike. If you lean over too far, you'll fall off. Don't do that.

# ???

- Kakve veze ovo sve ima sa naslovom predavanja?
- Linux Sucks by Bryan Lunduke

# ???

Šta ako Terijeve ideje nisu baš baš kompletno lude? *(mislimo naravno isključivo na ideje vezane za programiranje)*

# C64

- 1982-1994.
- 64 KB RAM (37 KB dostupno zbog BASIC interpretera)
- PEEK, POKE
- Commodore 1541, Commodore Datasette



197 PRINT "YOU DID ALL TH
1100 Q=INT(N*RND(1))+1
1110 IF U(Q)=1 THEN 100
The basics
of BASIC

## Programski jezik C


- Nastao 1972. kao zamena za pisanje asemblera za Unix OS
- AOT kompajliran
- K&R C (C78), ANSI C (~ C89), C99, C11, C17, C2x

## Programski jezik C++


- Nastao 1985. kao „C sa klasama“
- STL
- AOT kompajliran
- C++98, C++03, C++11, C++14, C++17, C++20

# A zašto ne C/C++?

## Bjarne Stroustrup > Quotes > Quotable Quote

"C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do it blows your whole leg off."

— Bjarne Stroustrup

# C++

From: Linus Torvalds <torvalds <at> linux-foundation.org>
Subject: Re: [RFC] Convert builin-mailinfo.c to use The Better String Library.
Newsgroups: gmane.comp.version-control.git
Date: 2007-09-06 17:50:28 GMT (2 years, 14 weeks, 16 hours and 36 minutes ago)

On Wed, 5 Sep 2007, Dmitry Kakurin wrote:
>
> When I first looked at Git source code two things struck me as odd:
> 1. Pure C as opposed to C++. No idea why. Please don't talk about portability,
> it's BS.

*YOU* are full of bullshit.

C++ is a horrible language. It's made more horrible by the fact that a lot of substandard programmers use it, to the point where it's much much easier to generate total and utter crap with it. Quite frankly, even if the choice of C were to do *nothing* but keep the C++ programmers out, that in itself would be a huge reason to use C.

In other words: the choice of C is the only sane choice. I know Miles Bader jokingly said "to piss you off", but it's actually true. I've come to the conclusion that any programmer that would prefer the project to be in C++ over C is likely a programmer that I really *would* prefer to piss off, so that he doesn't come and screw up any project I'm involved with.

C++ leads to really really bad design choices. You invariably start using the "nice" library features of the language like STL and Boost and other total and utter crap, that may "help" you program, but causes:

 - infinite amounts of pain when they don't work (and anybody who tells me that STL and especially Boost are stable and portable is just so full of BS that it's not even funny)

 - inefficient abstracted programming models where two years down the road you notice that some abstraction wasn't very efficient, but now all your code depends on all the nice object models around it, and you cannot fix it without rewriting your app.

From: Linus Torvalds
Subject: Re: Compiling C++ kernel module + Makefile
Date: Mon, 19 Jan 2004 22:46:23 -0800 (PST)

On Tue, 20 Jan 2004, Robin Rosenberg wrote:
>
> This is the "We've always used COBOL^H^H^H^H" argument.

In fact, in Linux we did try C++ once already, back in 1992.

It sucks. Trust me - writing kernel code in C++ is a BLOODY STUPID IDEA.

The fact is, C++ compilers are not trustworthy. They were even worse in 1992, but some fundamental facts haven't changed:

 - the whole C++ exception handling thing is fundamentally broken. It's _especially_ broken for kernels.
 - any compiler or language that likes to hide things like memory allocations behind your back just isn't a good choice for a kernel.
 - you can write object-oriented code (useful for filesystems etc) in C, _without_ the crap that is C++.

In general, I'd say that anybody who designs his kernel modules for C++ is either
 (a) looking for problems
 (b) a C++ bigot that can't see what he is writing is really just C anyway
 (c) was given an assignment in CS class to do so.

# C

```
GETS(3)                    Linux Programmer's Manual                    GETS(3)

NAME
       gets - get a string from standard input (DEPRECATED)

SYNOPSIS
       #include <stdio.h>

       char *gets(char *s);

DESCRIPTION
       Never use this function.

       gets()  reads  a  line from stdin into the buffer pointed to by s until
       either a terminating newline or EOF, which it replaces with a null byte
       ('\0').  No check for buffer overrun is performed (see BUGS below).
```

- Nije svaki AOT kompajler jednako koristan:

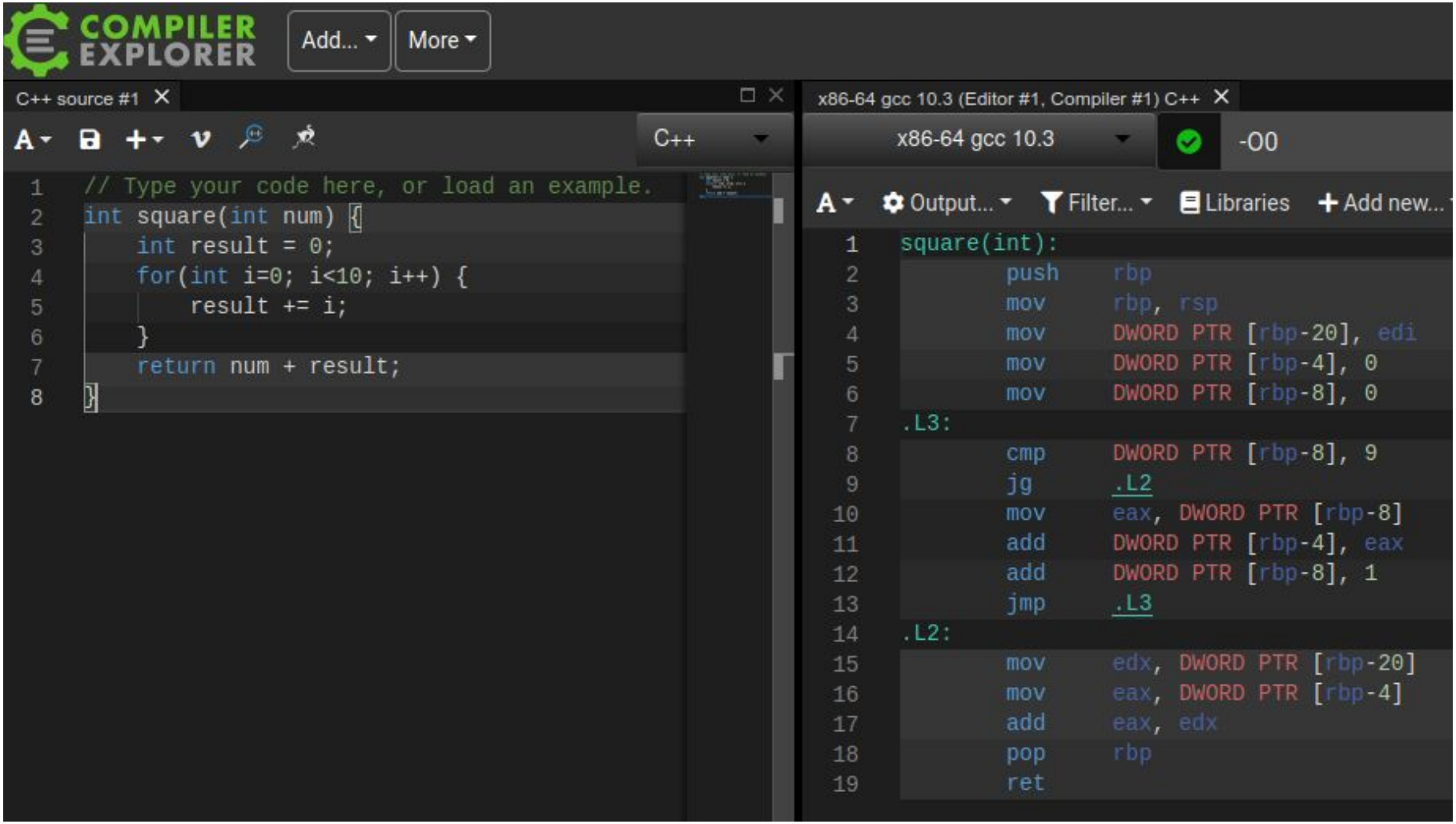
## More Of The Linux Kernel's x86 Assembly Code Gets Rewritten In C

Written by Michael Larabel in Linux Kernel on 18 June 2015 at 03:49 PM EDT. 43 Comments

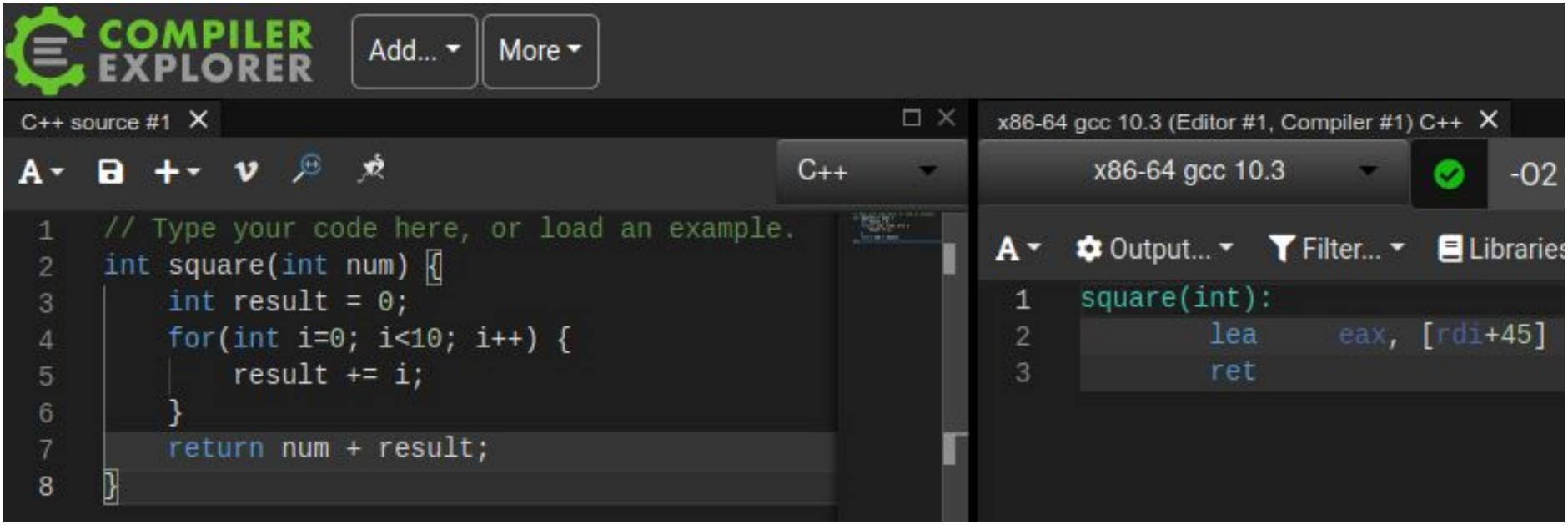
More of the Linux kernel's complicated and poorly maintained x86 Assembly code continues to be rewritten in modern and clean C.

With Linux 4.1 comes many x86 ASM code changes, "over 100 separate cleanups, restructuring changes, speedups and fixes in the x86 system call, irq, trap and other entry code, part of a heroic effort to deobfuscate a decade old spaghetti asm code and its C code dependencies."

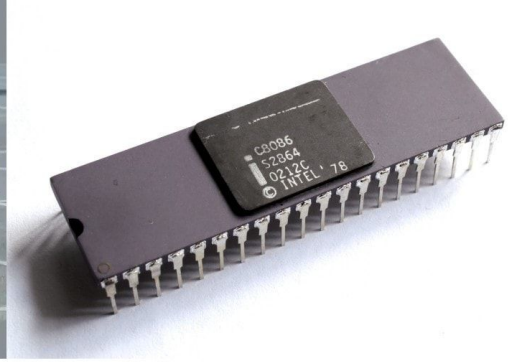
# Optimizacije



# Optimizacije



# x86




## YUGO 45

настао пре 40 година

није ваљо ни кад је изашао

троши превише и прегрева се често

има људи који се куну у њега из неког разлога

стартовао целу индустрију за производњу делова за буџење

нема серво

има преграду за рукавице

мораш да га гурнеш да крене, уз искрену молитву

домаће, комунистичко

може да се отвори кључем од стана

по дизајну има фазон 20 делова

## x86 архитектура

настала пре 40 година

није ваљала ни кад је изашла

троши превише и прегрева се често

цела индустрија се куне у њу из неког разлога

стартовала софтверску и хардверску индустрију да 40 година крпи лоше дизајнерске одлуке

нема крштен меморијски модел

има гомилу инструкција за BCD

мора да пишеш бутлоудер који пролази кроз 4 мода, уз искрену молитву

америчко капиталистичко

евентуално уз крв једнорога

презакомпликована

## РАЗЛИКЕ

## Complete 8086 instruction set

Quick reference:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AAA | CMPSB | JAE | JNBE | JPO | MOV | RCR | SCASB |
| AAD | CMPSW | JB | JNC | JS | MOVSB | REP | SCASW |
| AAM | CWD | JBE | JNE | JZ | MOVSW | REPE | SHL |
| AAS | DAA | JC | JNG | LAHF | MUL | REPNE | SHR |
| ADC | DAS | JCXZ | JNGE | LDS | NEG | REPNZ | STC |
| ADD | DEC | JE | JNL | LEA | NOP | REPZ | STD |
| AND | DIV | JG | JNLE | LES | NOT | RET | STI |
| CALL | HLT | JGE | JNO | LODSB | OR | RETF | STOSB |
| CBW | IDIV | JL | JNP | LODSW | OUT | ROL | STOSW |
| CLC | IMUL | JLE | JNS | LOOP | POP | ROR | SUB |
| CLD | IN | JMP | JNZ | LOOPE | POPA | SAHF | TEST |
| CLI | INC | JNA | JO | LOOPNE | POPF | SAL | XCHG |
| CMC | INT | JNAE | JP | LOOPNZ | PUSH | SAR | XLATB |
| CMP | INTO | JNB | JPE | LOOPZ | PUSHA | SBB | XOR |
| | IRET | | | | PUSHF | | |
| | JA | | | | RCL | | |

| Instruction | Operands | Description |
|---|---|---|
| AAA | No operands | ASCII Adjust after Addition.<br>Corrects result in AH and AL after addition when working with BCD values.<br><br>It works according to the following Algorithm:<br><br>if low nibble of AL > 9 or AF = 1 then:<br>• AL = AL + 6<br>• AH = AH + 1<br>• AF = 1<br>• CF = 1<br><br>else<br>• AF = 0<br>• CF = 0<br><br>in both cases:<br>clear the high nibble of AL.<br><br>Example:<br>MOV AX, 15 ; AH = 00, AL = 0Fh<br>AAA ; AH = 01, AL = 05<br>RET<br><br>C Z S O P A<br>r ? ? ? ? r |
| AAD | No operands | ASCII Adjust before Division.<br>Prepares two BCD values for division.<br><br>Algorithm:<br>• AL = (AH * 10) + AL<br>• AH = 0<br><br>Example:<br>MOV AX, 0105h ; AH = 01, AL = 05<br>AAD ; AH = 00, AL = 0Fh (15)<br>RET<br><br>C Z S O P A<br>? r r ? r ? |

# x86

- Real mode (16 bit, 1 jezgro, 4K rama) -> Protected mode (32 bit) -> Long mode (64 bit)
- IA64 - AMD64
- Speculative execution - Meltdown, Spectre

The lack of seeing AVX512 for Alder Lake led Torvalds to comment:

> *I hope AVX512 dies a painful death, and that Intel starts fixing real problems instead of trying to create magic instructions to then create benchmarks that they can look good on.*
>
> *I hope Intel gets back to basics: gets their process working again, and concentrate more on regular code that isn't HPC or some other pointless special case.*
>
> *I've said this before, and I'll say it again: in the heyday of x86, when Intel was laughing all the way to the bank and killing all their competition, absolutely everybody else did better than Intel on FP loads. Intel's FP performance sucked (relatively speaking), and it matter not one iota.*
>
> *Because absolutely nobody cares outside of benchmarks.*

# Java

- Interpreter / JIT - Hotspot (warmup time - (C1,C2), deopt)
- JVM (200+ MB RAM-a)
- Maven Central, Sonatype (mvnrepository.com lista 20.3M artifakata)
- 1.0 - 1.4, 5 -> 8, BREAKING CHANGE,  9 -> 16 ...

- Refleksije, Bytecode generatori...
- Fragmentacija distribucija...

# C#

- JIT + dotNet runtime
- nuget (3.1 M verzija paketa)
- Do nedavno closed source .net  <->Mono
- WinOps (Windows, SQL Server, Azure...)

# JavaScript

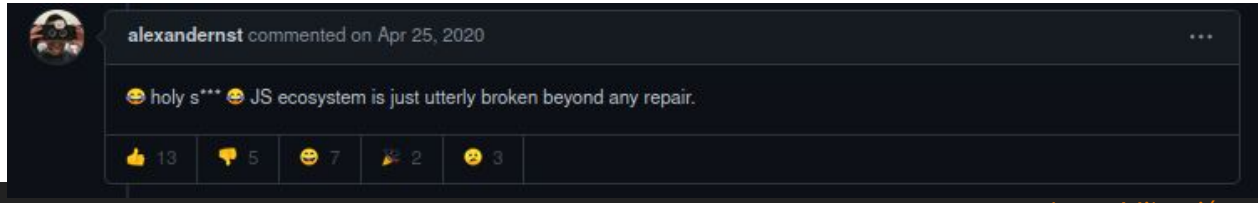
- Nema - ama baš - nikakve veze sa Javom.
- Osmišljen za 10 dana.
- JScript
- Slabo tipiziran -> TypeScript, CoffeeScript... (svi međusobno nekompatibilni)
- Dinamički -> interpreterski sve do v8
- npm (> 1.3 M paketa)
- Prvenstveno klijentski jezik - NodeJS doveo frontend-ovce u backend

## History [ edit ]

npm is written entirely in JavaScript and was developed by Isaac Z. Schlueter as a result of having "seen module packaging done terribly" and with inspiration from other similar projects such as PEAR (PHP) and CPAN (Perl).[5]

### Notable breakages [ edit ]

- In March 2016, npm attracted press attention[6] after a package called `left-pad`, historically used as an example which had become a dependency of many popular JavaScript packages, was unpublished as the result of a naming dispute between Azer Koculu, a self-taught software engineer, and Kik.[7] [8]Although the package was republished three hours later,[9] it caused widespread disruption, leading npm to change its policies regarding unpublishing to prevent a similar event in the future.[10]
- In February 2018, an issue was discovered in version 5.7.0 in which running `sudo npm` on Linux systems would change the ownership of system files, permanently breaking the operating system.[11]
- In July 2018, the npm credentials of a maintainer of the popular `eslint-scope` package were compromised resulting in a malicious release of `eslint-scope`, version 3.7.2. The malicious code copies the npm credentials of the machine running `eslint-scope` and uploads them to the attacker.[12]
- In November 2018, it was discovered that a malicious package had been added as a dependency to version 3.3.6 of the popular package `event-stream`. The malicious package, called `flatmap-stream`, contained an encrypted payload that steals bitcoins from certain applications. npm administrators responded by removing the offending package.[13][14]
- In April 2020, a small package called `is-promise` resulted in outage in serverless applications and deployments worldwide by virtue of being a dependency of many big and important applications.[15]

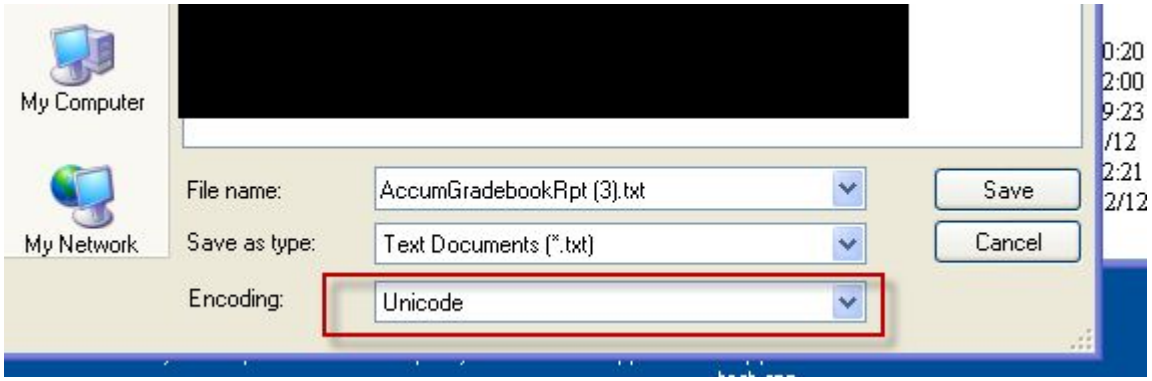


# Dependency hell

Okej, dakle sve zavisi od paketa koje neko održava.

- Šta se dešava kada neki paket zavisi od drugog paketa koji ne postoji?
- (ili od druge verzije drugog paketa koja nije dostupna)
- Šta se dešava kada autor paketa ne zakrpi sigurnosni propust na vreme?
- (ili kada u lancu zavisnosti neko ne bump-uje verziju kritične zavisnosti)
- Šta se dešava kada autor prestane da apdejtuje paket?
- (ili kada dođe do fragmentacije forkova)

# .txt




[The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)]

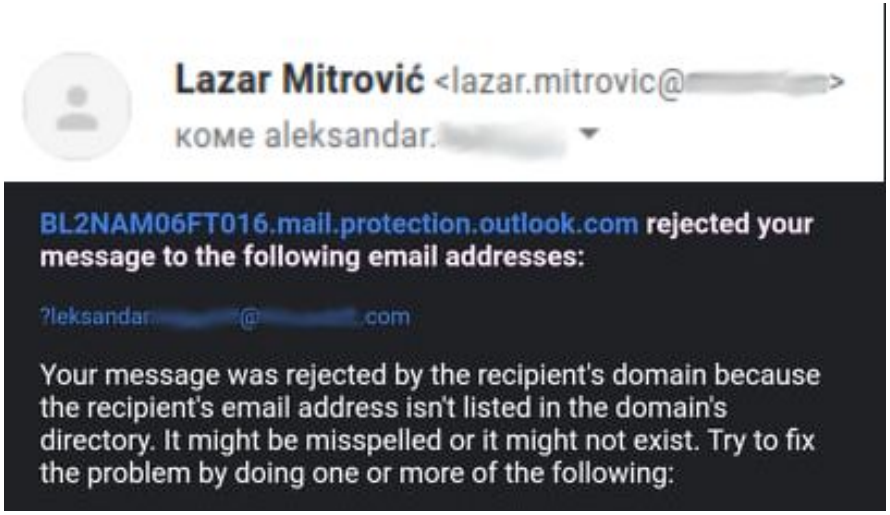
- U početku behu EBCDIC i ostali slični sistemi.
- Onda nastade ASCII i sve je bilo super, i svi su živeli srećno do kraja života...

- Ne.

- OEM character set-ovi...
- Unicode! (Hajde da izlistamo sve tipografske code point-ove na svetu u jednu veeeeliku tabelu)

# Enkodiranje

- UTF-16... (Byte order mark FE FF,  memorija...)
- UTF-8!!! (Hajde da ostavimo ASCII kakav jeste i da u prazan bit dodamo informacije za ne-engleske karaktere).
- strlen() je težak, ‘A’ != ’***A***’, renderovanje ume da ubije iOS ponekad

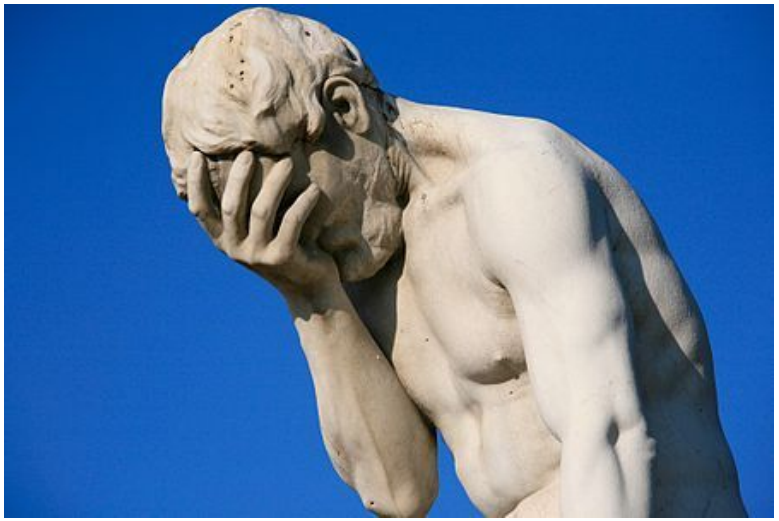



Utf-8 je nastao 1993. godine.

# Žurka sa enkodiranjem (1)

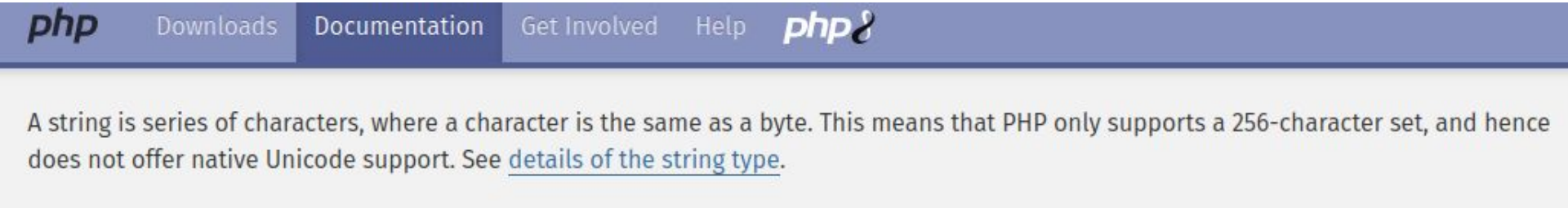
- Benevolent dictator for life Python programskog jezika - Guido van Rossum je 2000. uveo podršku za unicode kao 16 ili 32-bitni tip.
- Već 2006. na mejling listi javila se ideja za breaking promene i string postaje unicode po default-u.
- Python 3.0 je nekompatibilan sa Python 2.x. EOL period je trajao 12 GODINA (zaključno sa 1.1.2020.)

- 20.02.2021. Kodi Media Center izbacuje verziju 19.0 kojom se uklanja podrška za Python 2.x, čime automatski umire većina third party plugin-ova.



# Žurka sa enkodiranjem (2)

- PHP ni 2021. nema podršku za unicode.




- Windows kroz API i interno koristi UTF-16.
- Linux i MacOS koriste UTF-8 (MacOS garantuje UTF-8 svuda)

A šta je sa Web Browser-ima?

# Žurka sa web browser-ima (1)

# Žurka sa web browser-ima (2)

## User-agent lepota :)

| | | |
|---|---|---|
| NCSA Mosaic |  | *NCSA_Mosaic/2.0 (Windows 3.1)* Slike |
| Mozilla (Mosaic Killer) - Netscape |   | *Mozilla/1.0 (Win3.1)* Frejmovi |
| Internet Explorer |  | *Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)* |
| Gecko (Firefox) |  | *Mozilla/5.0 (Windows; U; Windows NT 5.1; sv-SE; rv:1.7.5) Gecko/20041108 Firefox/1.0* |
| Konquerer (KHTML) |  | *Mozilla/5.0 (compatible; Konqueror/3.2; FreeBSD) (KHTML, like Gecko)* |
| Opera |  | *Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; en) Opera 9.51*<br><br>*Mozilla/5.0 (Windows NT 6.0; U; en; rv:1.8.1) Gecko/20061208 Firefox/2.0.0 Opera 9.51*<br><br>*Opera/9.51 (Windows NT 5.1; U; en)* |

# Žurka sa web browser-ima (3)

| | | |
|---|---|---|
| Safari (KHTML -> Webkit) |  | *Mozilla/5.0 (Macintosh; U; PPC Mac OS X; de-de) AppleWebKit/85.7 (KHTML, like Gecko) Safari/85.5* |
| Internet Explorer 8 |  | *Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0)* |
| Chrome (WebKit) |  | *Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/525.13 (KHTML, like Gecko) Chrome/0.2.149.27 Safari/525.13* |

Najnoviji Chrome UA: **Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.72 Safari/537.36**

A šta je ovo? Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.72 Safari/537.36 Edg/90.0.818.41

# Žurka sa web browser-ima (4)

**<html>**

**<head>**

**<meta http-equiv="Content-Type" content="text/html; charset=utf-8">**

**….**

A šta ako nema charset tag-a?

- Internet Explorer pokušava da izvuče histogram pojavljivanja karaktera iz opsega 128 do 255 i da ga uporedi sa histogramima svih podržanih jezika…
- Ovo zapravo najčešće radi, ali kada ne radi može vaš tekst na bugarskom da renderuje kao korejski…

Danas sa popularnošću UTF-8 ovo predstavlja manji problem, ali sličan kod i dalje postoji u Gugl Hromu:

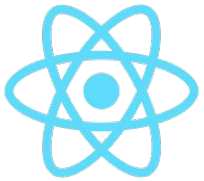
```
bool DetectTextEncoding(const char* data,
                        uint32_t length,
                        const char* hint_encoding_name,
                        const KURL& hint_url,
                        const char* hint_user_language,
                        WTF::TextEncoding* detected_encoding) {
  *detected_encoding = WTF::TextEncoding();
  // In general, do not use language hint. This helps get more
  // deterministic encoding detection results across devices. Note that local
  // file resources can still benefit from the hint.
  Language language = UNKNOWN_LANGUAGE;
  if (hint_url.Protocol() == "file")
    LanguageFromCode(hint_user_language, &language);
  int consumed_bytes;
  bool is_reliable;
  Encoding encoding = CompactEncDet::DetectEncoding(
      data, length, hint_url.GetString().Ascii().c_str(), nullptr, nullptr,
      EncodingNameAliasToEncoding(hint_encoding_name), language,
      CompactEncDet::WEB_CORPUS,
      false,  // Include 7-bit encodings to detect ISO-2022-JP
      &consumed_bytes, &is_reliable);

  if (encoding == UNKNOWN_ENCODING)
    *detected_encoding = WTF::UnknownEncoding();
  else
    *detected_encoding = WTF::TextEncoding(MimeEncodingName(encoding));

  // Should return false if the detected encoding is UTF8. This helps prevent
  // modern web sites from neglecting proper encoding labelling and simply
  // relying on browser-side encoding detection. Encoding detection is supposed
  // to work for web sites with legacy encoding only (so this doesn't have to
  // be applied to local file resources).
  // Detection failure leads |TextResourceDecoder| to use its default encoding
  // determined from system locale or TLD.
  return !(encoding == UNKNOWN_ENCODING ||
           (hint_url.Protocol() != "file" && encoding == UTF8));
}
```

# HTML 5

- 2008-2014. W3C
- Apple, Google, Mozilla, Microsoft WHATWG
- Adobe Flash - iPhone problem (2011. Flash discontinued)
- W3C - WHATWG conflict
- CSS3 (1999-2018+)
- SEO -> Client Rendering -> Server Rendering
- InLiNe CsS






# http://motherfuckingwebsite.com/

## This is a motherfucking website.

And it's fucking perfect.

### Seriously, what the fuck else do you want?

You probably build websites and think your shit is special. You think your 13 megabyte parallax-ative home page is going to get you some fucking Awwward banner you can glue to the top corner of your site. You think your 40-pound jQuery file and 83 polyfills give IE7 a boner because it finally has box-shadow. Wrong, motherfucker. Let me describe your perfect-ass website:

- Shit's lightweight and loads fast
- Fits on all your shitty screens
- Looks the same in all your shitty browsers
- The motherfucker's accessible to every asshole that visits your site
- Shit's legible and gets your fucking point across (if you had one instead of just 5mb pics of hipsters drinking coffee)

**Well guess what, motherfucker:**

You. Are. Over-designing. Look at this shit. It's a motherfucking website. Why the fuck do you need to animate a fucking trendy-ass banner flag when I hover over that useless piece of shit? You spent hours on it and added 80 kilobytes to your fucking site, and some motherfucker jabbing at it on their iPad with fat sausage fingers will never see that shit. Not to mention blind people will never see that shit, but they don't see any of your shitty shit.

You never knew it, but this is your perfect website. Here's why.

### It's fucking lightweight

This entire page weighs less than the gradient-meshed facebook logo on your fucking Wordpress site. Did you seriously load 100kb of jQuery UI just so you could animate the fucking background color of a div? You loaded all 7 fontfaces of a shitty webfont just so you could say "Hi." at 100px height at the beginning of your site? You piece of shit.

### It's responsive

You dumbass. You thought you needed media queries to be responsive, but no. Responsive means that it responds to whatever motherfucking screensize it's viewed on. This site doesn't care if you're on an iMac or a motherfucking Tamagotchi.

### It fucking works

Look at this shit. You can read it ... that is, if you can read, motherfucker. It makes sense. It has motherfucking hierarchy. It's using HTML5 tags so you and your bitch-ass browser know what the fuck's in this fucking site. That's semantics, motherfucker.

It has content on the fucking screen. Your site has three bylines and link to your dribbble account, but you spread it over 7 full screens and make me click some bobbing button to show me how cool the jQuery ScrollTo plugin is.

Cross-browser compatibility? Load this motherfucker in IE6. I fucking dare you.

# http://vanilla-js.com/


*Vanilla JS* is a fast, lightweight, cross-platform framework
for building incredible, powerful JavaScript applications.

## Introduction

The *Vanilla JS* team maintains every byte of code in the framework and works hard each day to make sure it is small and intuitive. Who's using *Vanilla JS*? Glad you asked! Here are a few:

Facebook Google YouTube Yahoo Wikipedia Windows Live Twitter Amazon LinkedIn MSN

eBay Microsoft Tumblr Apple Pinterest PayPal Reddit Netflix Stack Overflow

In fact, *Vanilla JS* is already used on more websites than jQuery, Prototype JS, MooTools, YUI, and Google Web Toolkit - *combined*.

## Download

Ready to try *Vanilla JS*? Choose exactly what you need!

- [x] Core Functionality
- [x] DOM (Traversal / Selectors)
- [x] Prototype-based Object System
- [x] AJAX
- [x] Animations
- [x] Event System
- [x] Regular Expressions
- [x] Functions as first-class objects
- [x] Closures
- [x] Math Library
- [x] Array Library
- [x] String Library

### Options

- [x] Minify Source Code
- [x] Produce UTF8 Output
- [x] Use "CRLF" line breaks (Windows)

**Final size:** 0 bytes uncompressed, 25 bytes gzipped. [ ] Show human-readable sizes

Download

# Mobilni telefoni

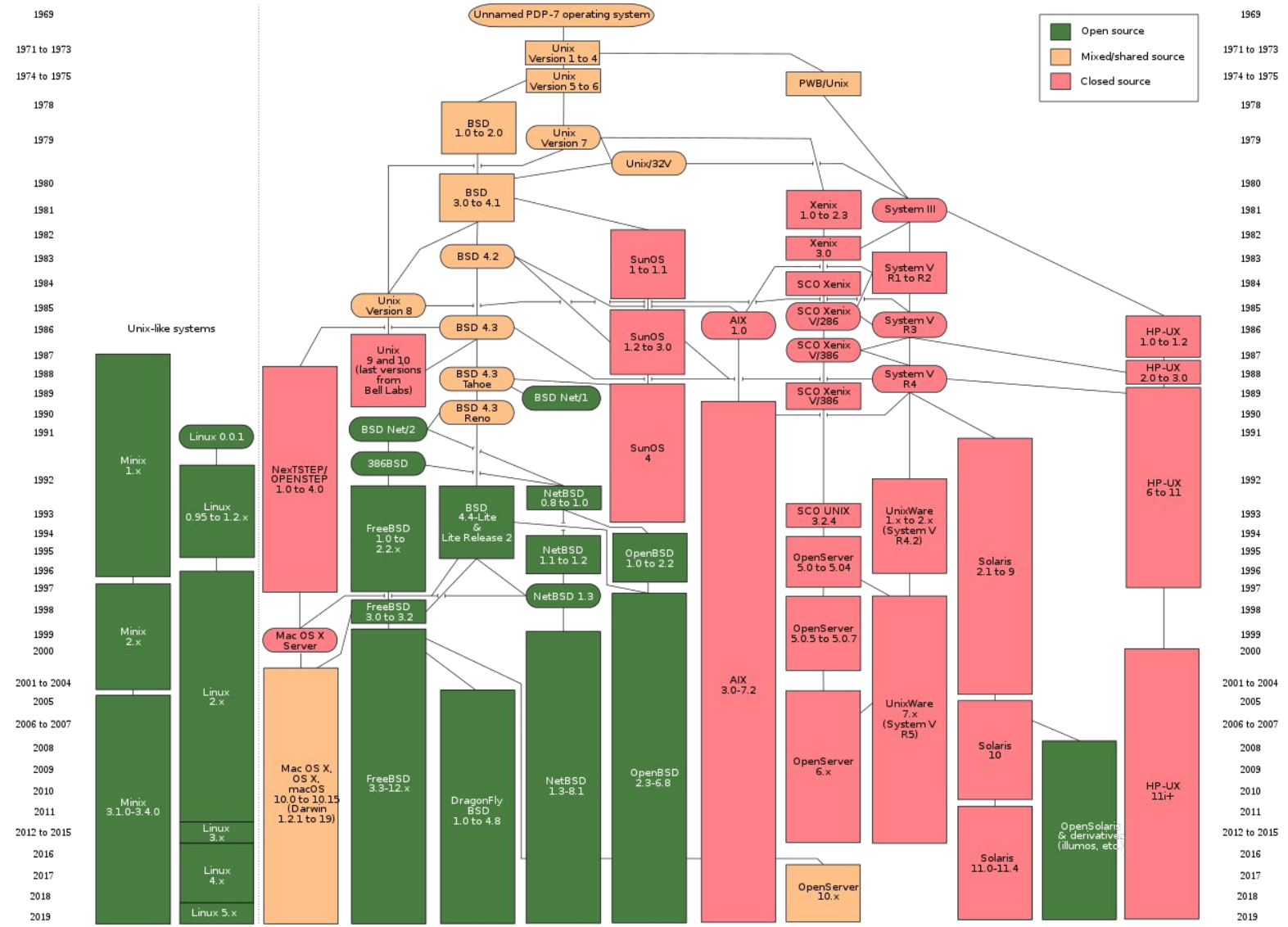
Apple

- 2007. Stiv Džobs izbacuje iPhone bez SDK (svi treba da pišu HTML5 PWA).
- 2008. App Store + iPhone SDK (Objective-C 1988. NeXT)
- 2014. Swift (Objective-C without C)

Google

- 2009. Apache Harmony Java (Dalvik VM)
- Dalvik -> ART
- 2017. Kotlin uveden, 2019. default
- 2017. Flutter inicijalni, 2021. Flutter svuda (pa i na Web-u)

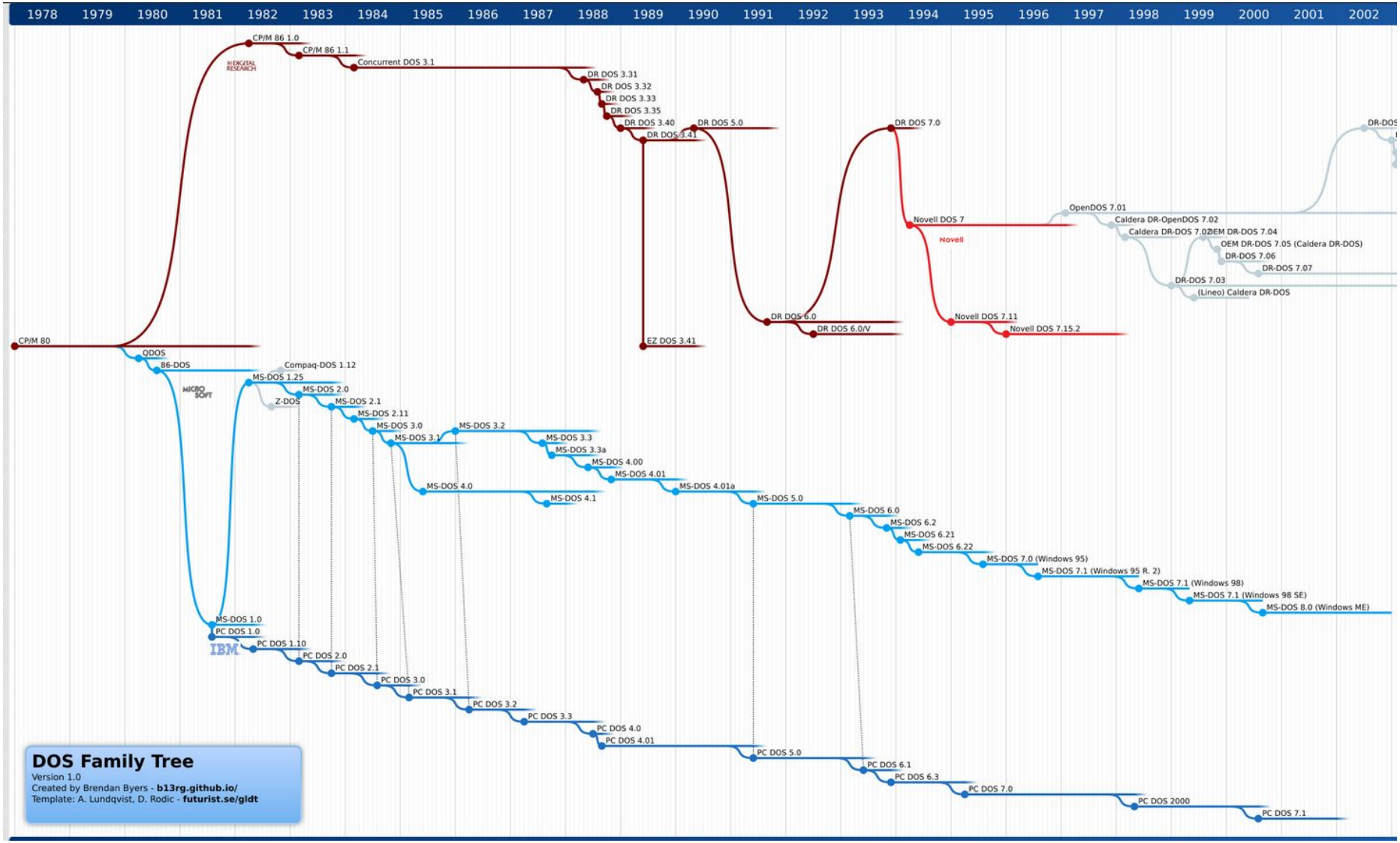
# Unix


Unnamed PDP-7 operating system
Unix Version 1 to 4
Unix Version 5 to 6
PWB/Unix
BSD 1.0 to 2.0
Unix Version 7
Unix/32V
BSD 3.0 to 4.1
Xenix 1.0 to 2.3
System III
BSD 4.2
SunOS 1 to 1.1
Xenix 3.0
System V R1 to R2
SCO Xenix
Unix Version 8
BSD 4.3
SunOS 1.2 to 3.0
AIX 1.0
SCO Xenix V/286
System V R3
SCO Xenix V/386
HP-UX 1.0 to 1.2
HP-UX 2.0 to 3.0
Unix 9 and 10 (last versions from Bell Labs)
BSD 4.3 Tahoe
BSD Net/1
System V R4
BSD 4.3 Reno
BSD Net/2
SunOS 4
386BSD
NetBSD 0.8 to 1.0
BSD 4.4-Lite & Lite Release 2
SCO UNIX 3.2.4
UnixWare 1.x to 2.x (System V R4.2)
HP-UX 6 to 11
OpenServer 5.0 to 5.04
Solaris 2.1 to 9
NetBSD 1.1 to 1.2
OpenBSD 1.0 to 2.2
NetBSD 1.3
FreeBSD 3.0 to 3.2
OpenServer 5.0.5 to 5.0.7
Mac OS X Server
AIX 3.0-7.2
UnixWare 7.x (System V R5)
OpenServer 6.x
Solaris 10
HP-UX 11i+
OpenSolaris & derivatives (illumos, etc)
Solaris 11.0-11.4
OpenServer 10.x
Open source
Mixed/shared source
Closed source
Unix-like systems
Minix 1.x
Minix 2.x
Minix 3.1.0-3.4.0
Linux 0.0.1
Linux 0.95 to 1.2.x
Linux 2.x
Linux 3.x
Linux 4.x
Linux 5.x
NexTSTEP/ OPENSTEP 1.0 to 4.0
Mac OS X, OS X, macOS 10.0 to 10.15 (Darwin 1.2.1 to 19)
FreeBSD 1.0 to 2.2.x
FreeBSD 3.3-12.x
DragonFly BSD 1.0 to 4.8
NetBSD 1.3-8.1
OpenBSD 2.3-6.8
1969
1971 to 1973
1974 to 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 to 2004
2005
2006 to 2007
2008
2009
2010
2011
2012 to 2015
2016
2017
2018
2019

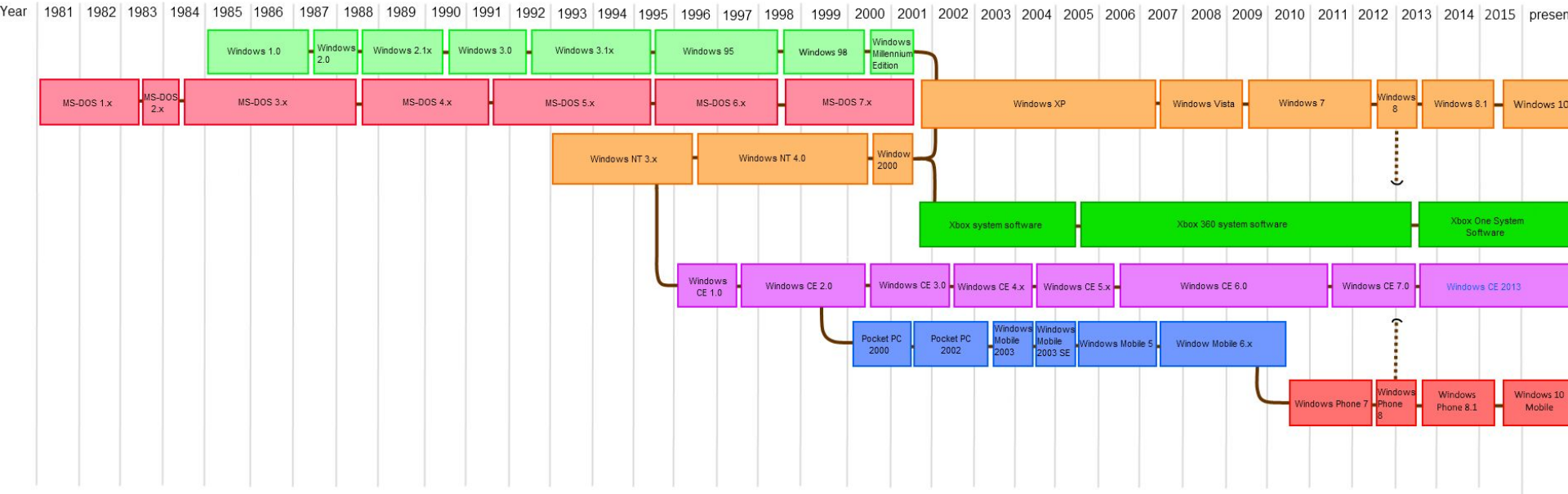
# Linux

- Najpopularniji Unix-oliki kernel - Linux, nastao je kao lični projekat jednog čoveka
- Monolitni kernel - Linux / Mikrokernel - Minix (videti Davidovo predavanje sa NI 2.0)
- Preko 28 miliona linija koda, uključujući sve drajvere za sve uređaje i sve podržane arhitekture (koliko prostora za greške?)
- Male optimizacije + nedostatak intenzivnog testiranja -> gubitak podataka (4.19.1 - 2018. godine, 5.12 - 2021. godine)
- Greška u grafičkom steku -> korupcija fajl sistema.

# CP/M, DOS, Windows, OS/2



# CP/M, DOS, Windows, OS/2



# CP/M, DOS, Windows, OS/2

- Ne mogu da nazovem folder CON !!!
  (ni PRN, AUX, NUL, COM0-9, LPT0-9)

- *Security through obscurity* ne radi (niti je ikada radio).
- Samo u oktobru 2020. pečovano je 87 ranjivosti, od kojih je najgora RCE pomoću ICMPv6 Router Advertisement paketa (CVE skor 9.8).

Kako saznajem verziju trenutnu verziju operativnog sistema?

# Ime OS-a

Linux / MacOS:

```
@Override
protected String osNameValue() {
    Utsname.utsname name = StackValue.get(Utsname.utsname.class);
    if (Utsname.uname(name) >= 0) {
        return CTypeConversion.toJavaString(name.sysname());
    }
    return "Unknown";
}
```

# Ime OS-a

- Windows:

```
public Pair<String, String> getOsNameAndVersion() {
    /*
     * Reimplementation of code from java_props_md.c
     */
    SysinfoAPI.OSVERSIONINFOEXA ver = StackValue.get(SysinfoAPI.OSVERSIONINFOEXA.class);
    ver.dwOSVersionInfoSize(SizeOf.get(SysinfoAPI.OSVERSIONINFOEXA.class));
    SysinfoAPI.GetVersionExA(ver);

    boolean is64bit = ImageSingletons.lookup(Platform.class).getArchitecture().endsWith("64");
    boolean isWorkstation = ver.wProductType() == VER_NT_WORKSTATION;
    int platformId = ver.dwPlatformId();

    int majorVersion = ver.dwMajorVersion();
    int minorVersion = ver.dwMinorVersion();
    int buildNumber = ver.dwBuildNumber();
    do {
        /* Get the full path to \Windows\System32\kernel32.dll ... */
        LibC.WCharPointer kernel32Path = StackValue.get(WinBase.MAX_PATH, LibC.WCharPointer.class);
        LibC.WCharPointer kernel32Dll = NonmovableArrays.addressOf(NonmovableArrays.fromImageHeap(KERNEL32_DLL), 0);
        int len = WinBase.MAX_PATH - (int) LibC.wcslen(kernel32Dll).rawValue() - 1;
        int ret = SysinfoAPI.GetSystemDirectoryW(kernel32Path, len);
        if (ret == 0 || ret > len) {
            break;
        }
        LibC.wcsncat(kernel32Path, kernel32Dll, WordFactory.unsigned(WinBase.MAX_PATH - ret));

        /* ... and use that for determining what version of Windows we're running on. */
        int versionSize = WinVer.GetFileVersionInfoSizeW(kernel32Path, WordFactory.nullPointer());
        if (versionSize == 0) {
            break;
        }

        VoidPointer versionInfo = LibC.malloc(WordFactory.unsigned(versionSize));
        if (versionInfo.isNull()) {
            break;
        }

        if (WinVer.GetFileVersionInfoW(kernel32Path, 0, versionSize, versionInfo) == 0) {
            LibC.free(versionInfo);
            break;
        }

        LibC.WCharPointer rootPath = NonmovableArrays.addressOf(NonmovableArrays.fromImageHeap(ROOT_PATH), 0);
        WordPointer fileInfoPointer = StackValue.get(WordPointer.class);
        CIntPointer lengthPointer = StackValue.get(CIntPointer.class);
        if (WinVer.VerQueryValueW(versionInfo, rootPath, fileInfoPointer, lengthPointer) == 0) {
            LibC.free(versionInfo);
            break;
        }

        VerRsrc.VS_FIXEDFILEINFO fileInfo = fileInfoPointer.read();
        majorVersion = (short) (fileInfo.dwProductVersionMS() >> 16); // HIWORD
        minorVersion = (short) fileInfo.dwProductVersionMS(); // LOWORD
        buildNumber = (short) (fileInfo.dwProductVersionLS() >> 16); // HIWORD
        LibC.free(versionInfo);
    } while (false);

    String osVersion = majorVersion + "." + minorVersion;
    String osName;

    switch (platformId) {
        case VER_PLATFORM_WIN32_WINDOWS:
            if (majorVersion == 4) {
                switch (minorVersion) {
                    case 0:
                        osName = "Windows 95";
                        break;
                    case 10:
                        osName = "Windows 98";
                        break;
                    case 90:
                        osName = "Windows Me";
                        break;
                    default:
                        osName = "Windows 9X (unknown)";
                        break;
                }
            } else {
                osName = "Windows 9X (unknown)";
            }
            break;
        case VER_PLATFORM_WIN32_NT:
            if (majorVersion <= 4) {
                osName = "Windows NT";
            } else if (majorVersion == 5) {
                switch (minorVersion) {
                    case 0:
                        osName = "Windows 2000";
                        break;
                    case 1:
                        osName = "Windows XP";
                        break;
                    case 2:
                        if (isWorkstation && is64bit) {
                            osName = "Windows XP"; /* 64 bit */
                        } else {
                            osName = "Windows 2003";
                        }
                        break;
                    default:
                        osName = "Windows NT (unknown)";
                        break;
                }
            } else if (majorVersion == 6) {
                if (isWorkstation) {
                    switch (minorVersion) {
                        case 0:
                            osName = "Windows Vista";
                            break;
                        case 1:
                            osName = "Windows 7";
                            break;
                        case 2:
                            osName = "Windows 8";
                            break;
                        case 3:
                            osName = "Windows 8.1";
                            break;
                        default:
                            osName = "Windows NT (unknown)";
                    }
                } else {
                    switch (minorVersion) {
                        case 0:
                            osName = "Windows Server 2008";
                            break;
                        case 1:
                            osName = "Windows Server 2008 R2";
                            break;
                        case 2:
                            osName = "Windows Server 2012";
                            break;
                        case 3:
                            osName = "Windows Server 2012 R2";
                            break;
                        default:
                            osName = "Windows NT (unknown)";
                    }
                }
            } else if (majorVersion == 10) {
                if (isWorkstation) {
                    switch (minorVersion) {
                        case 0:
                            osName = "Windows 10";
                            break;
                        default:
                            osName = "Windows NT (unknown)";
                    }
                } else {
                    switch (minorVersion) {
                        case 0:
                            if (buildNumber > 17762) {
                                osName = "Windows Server 2019";
                            } else {
                                osName = "Windows Server 2016";
                            }
                            break;
                        default:
                            osName = "Windows NT (unknown)";
                    }
                }
            } else {
                osName = "Windows NT (unknown)";
            }
            break;
        default:
            osName = "Windows (unknown)";
            break;
    }
    return Pair.create(osName, osVersion);
}
```

# CLOUD

ClOuD
sErVerLesS
lAmBdA
CoNtAiNeRs

https://killedbygoogle.com/

## Killed by Google

Search

All (227)

June 2022

### Google Chrome Apps

Off to the glue factory in about 1 year, Google Chrome Apps were hosted or packaged web applications that ran on the Google Chrome browser. It will be over 11 years old.

January 2022

### Android Things

Done for in 9 months, Android Things was an Android-based embedded operating system (originally named Brillo) aimed to run on Internet of Things (IoT) devices. It will be over 6 years old.

December 2021

### AngularJS

Scheduled to be killed in 8 months, AngularJS was a JavaScript open-source front-end web framework based on MVC pattern using a dependency injection technique. It will be about 11 years old.

June 2021

### Google Hangouts

Turning to ashes in 2 months, Google Hangouts was a communication platform which included messages, video chat, and VOIP features. Execution scheduled "first half 2021". It will be over 8 years old.

June 2021

### Expeditions

Running out of juice in 2 months, Expeditions is a program for providing virtual reality experiences to school classrooms through Google Cardboard viewers, allowing educators to take their students on virtual field trips. It will be almost 6 years old.

June 2021

### Tour Creator

Another one bites the dust in 2 months, Tour Creator allowed users to build immersive, 360° guided tours that could be viewed with VR devices. It will be about 3 years old.

June 2021

### Poly

Done for in 2 months, Poly was a distribution platform for creators to share 3D objects. It will be over 3 years old.

2010 - 2021

### Google Go Links

Killed 23 days ago, (also known as Google Short Links) was a URL shortening service. It also supported custom domain for customers of Google Workspace (formerly G Suite (formerly Google Apps)). It was about 11 years old.

2014 - 2021

### Google Cardboard

Killed about 2 months ago, Google Cardboard was a low-cost, virtual reality (VR) platform named after its folded cardboard viewer into which a smartphone was inserted. It was over 6 years old.

# !!!!


SKINII GA S GOLA, niko nek ne brani!

# STAN'DE BOLAN!

# (de)Motivacija


# Always look on the bright side of code (1)

C/C++

- Bolji tool-ovi za detekciju potencijalnih race condition-a, memory error-a itd.
- Rust, tj oksidacija ??? (i dalje su podeljena mišljenja)
- Go?
- clang i gcc su izuzetno napredni danas (i biće još napredniji)

X86

- ARM (Apple M1, Microsoft...)
- RISC V (nadajmo se): SiFive Tapes Out Their First 5nm RISC-V Processor Core

Java

- GraalVM Native Image (AOT kompajler za Javu)
- Biblioteke koje niko ne održava već XY godina, (te ih i niko ne prevede na noviju Javu) verovatno i ne želite da koristite
- *(ili ih sami apdejtujte)*
- Aktivna zajednica i mnogo velikih firmi

C#

- Xamarin (pokreći sve svuda)
- .net 5.0 promoviše AOT svuda
- .net open source

# Always look on the bright side of code (2)

JavaScript
- Možda wasm?

Dependency hell
- Flatpack, snap
- Više ulaganja u maintainer-e
- Dependabot

Enkodiranje:
- Python 3 je već rešio (sada je do zajednice)
- PHP jednostavno ne koristiti :)

Web:
- Onaj framework koji najredovnije održava neka jaaako velika firma i koji ima najbolju podršku i interop sa neophodnim bibliotekama.
- Ne izmišljati rupu na saksiji (ma koliko primamljivo zvučalo).

# Always look on the bright side of code (3)

Mobilni telefoni:

- Ili slušati šta Apple i Google izmisle sutra...
- ... ili se prikačiti nekoj velikoj firmi i koristiti njihove cross platform framework-e (React Native, Flutter, Xamarin)
- ... ili živeti rizično (CodeName One, Apache Cordova)
- Tražiti balans -> dev time, UX

OS:

- Redovno ažurirati sistem...
- ... isključivo stabilnim verzijama kernela
- I velike kompanije imaju iste probleme i rade na otklanjanju istih (niste sami)
- Ukoliko koristite open source OS, možete i vi da se bacite u popravljanje :)
- ... u suprotnom žalim slučaj

Backwards kompatibilnost, koliko unosi probleme tolko vam omogućava da vaš kod radi i za N godina (osim ako je softver Guglov, tad ga već otpišite kao Deprecated)

Kad inženjerski hakovi postanu i nepodnošljivi, uvek je dobra ideja prošetati malo po suncu :)

# Hvala na pažnji!

Pitanja?