

C language in our world

15.5. 2017 FI MUNI

Brno

@jurajmichalek

<http://georgik.rocks>

<https://www.ysofters.com>

Grab the source code

<https://github.com/ysoftdevs/cpp-examples>

<https://github.com/georgik/LampESP>

<https://github.com/ysoftiota>



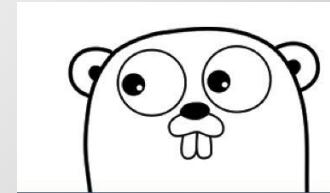
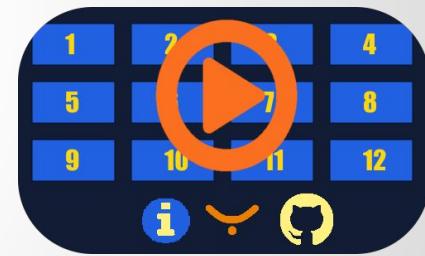
Who am I?



Blog: <http://georgik.rocks>



YSofters Blog: <https://www.ysofters.com>



C language today

Allegro5

NuGet

SDL2

Gradle

IDEs

IoT

Jenkins, Bamboo, TeamCity

Go language

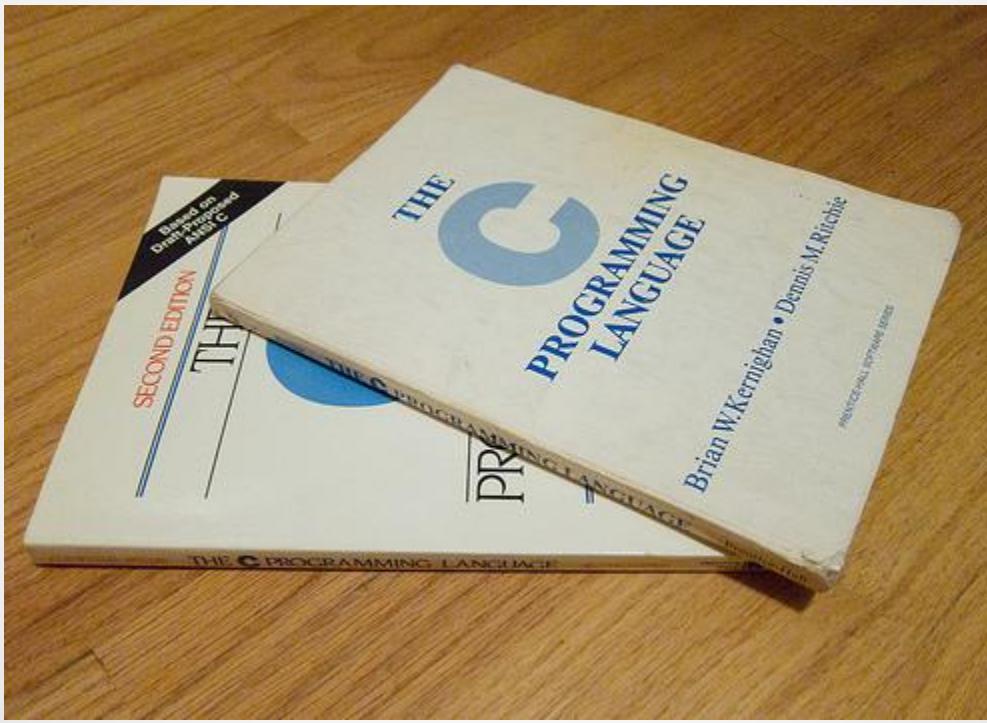
Once upon a time



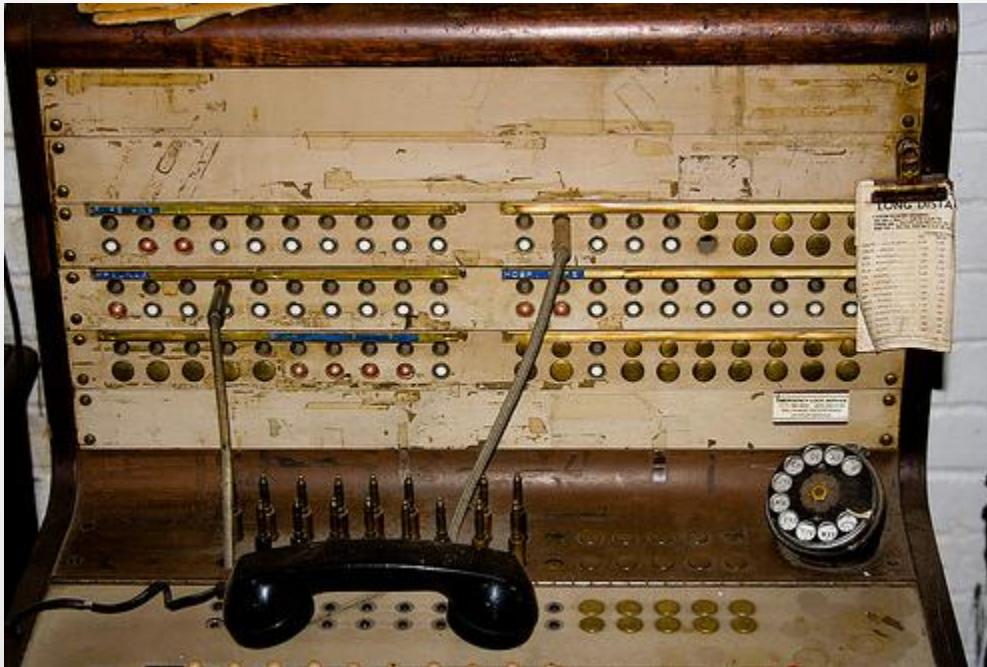
there lived a mighty king



His name was C



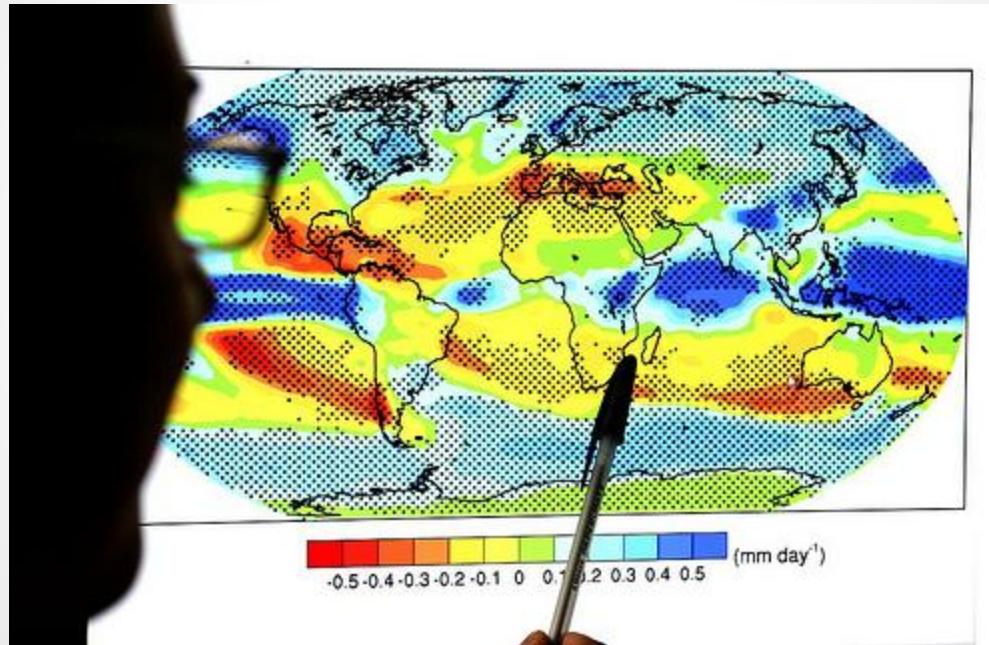
He ruled nearly everything telco, med, banks, games



King was getting older and paunchy



World was changing



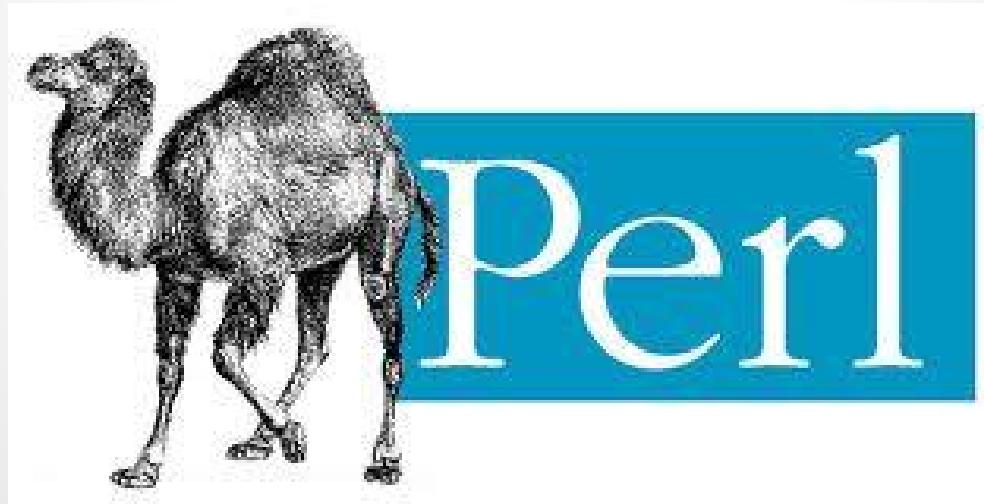
Changing so fast..



New rivals have arrived



Caravane with nomads from the land of Perl

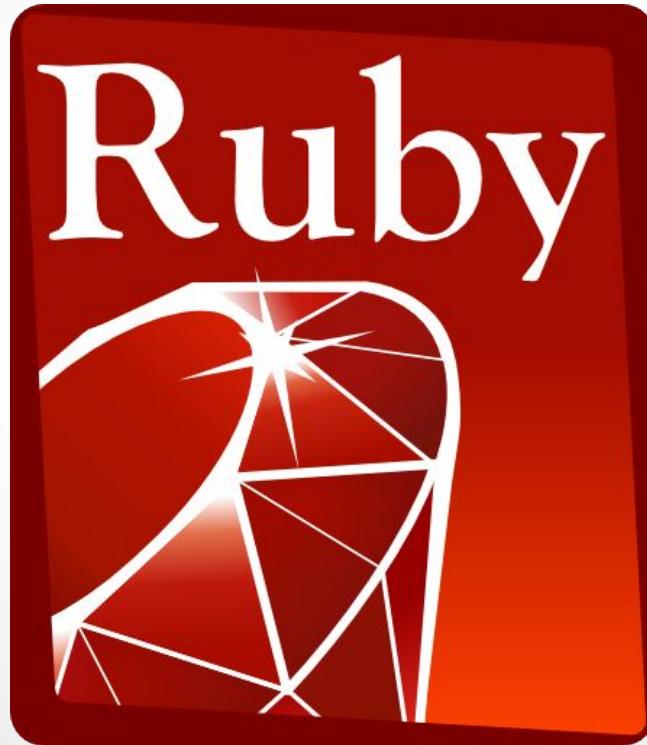


Lords of snakes

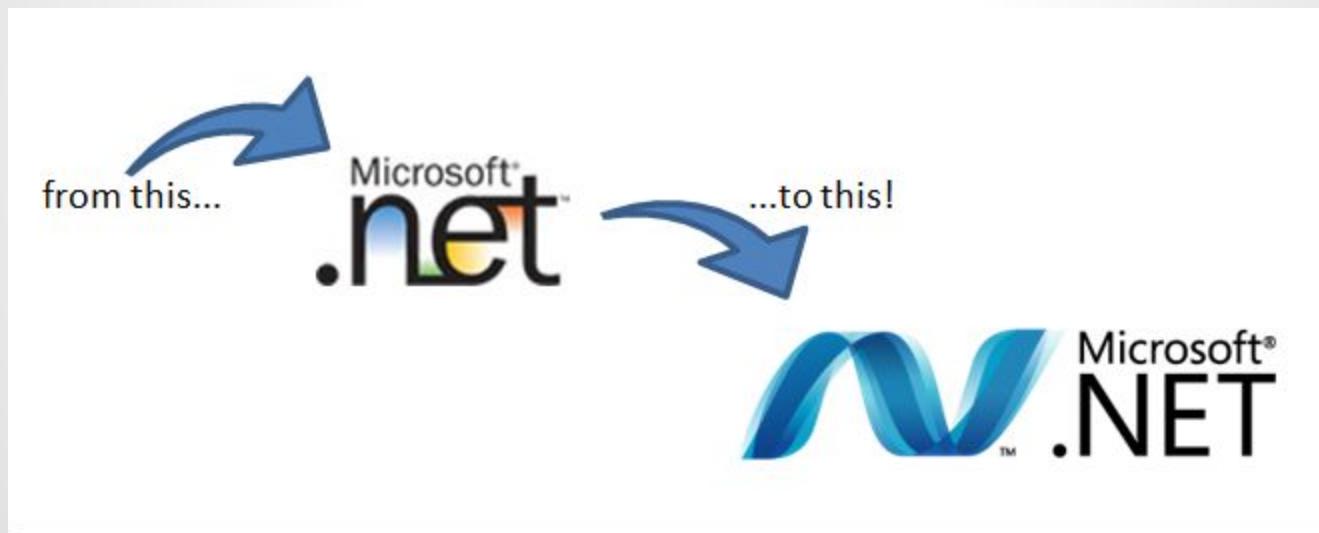
from the land of Python



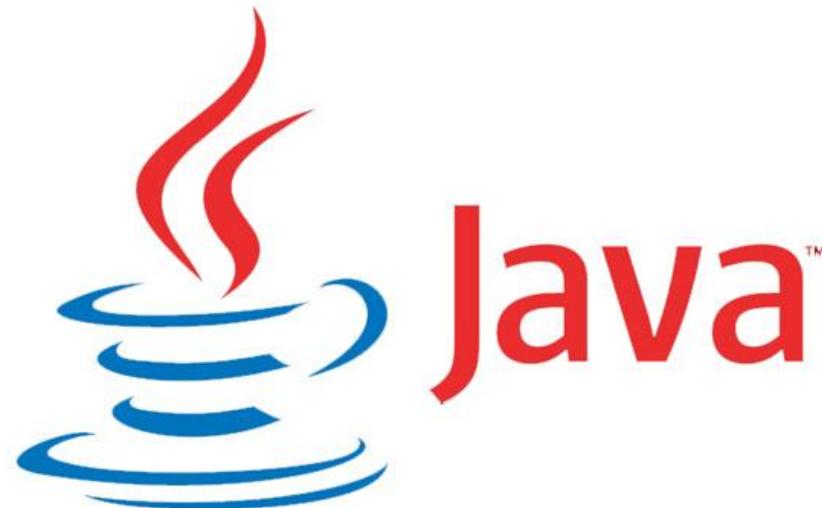
Jewelers from the the land of Ruby



Sharp warriors from the land of .Net



Coffee magnates from the island of Java



Cocoa drinkers from the land of Apple trees



Old kingdom of C fell into oblivion



**People were scared to enter
the realm of old C**



Beware SIGSEGV dragons!



Memory leak swamps!



Zombies of legacy code!



Evil MACROmancers



#define true false

Insidious multi-threaded bugs.



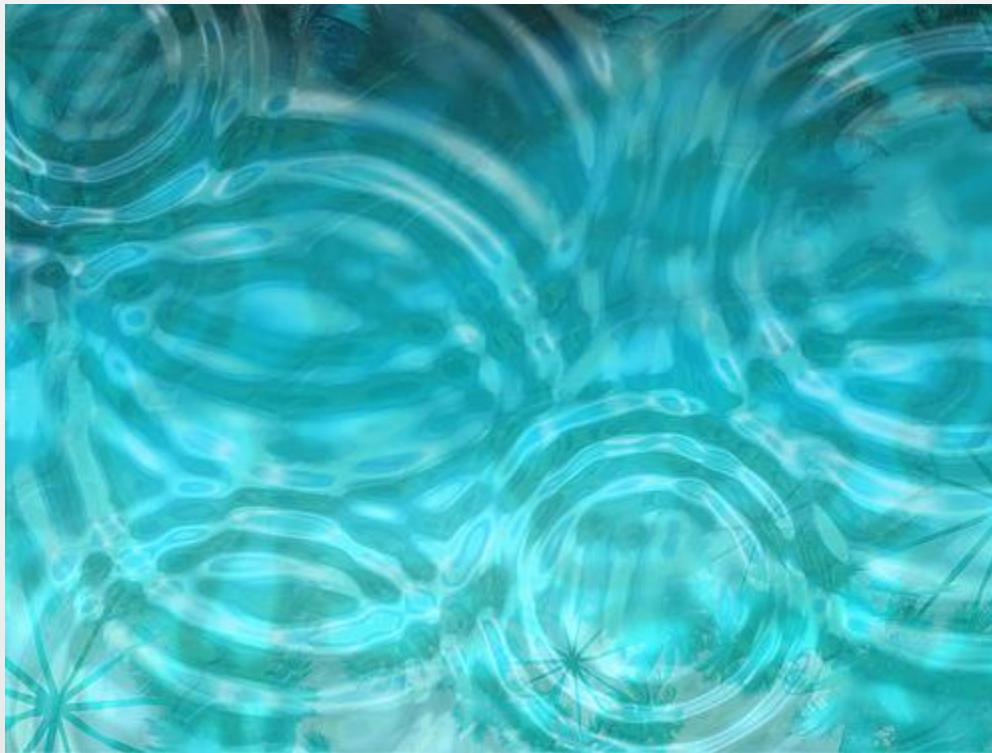
Scary place for life



The era of Cloud has emerged



Technologies influencing each other



Programming languages we know
strongly influence the way we think
about programming.

- JS Conf 2014 - Jenna Zeigen

Breeze of fresh ideas starts blowing from other technologies...



Bridge between technology

C - C#

C - Java

C - Python

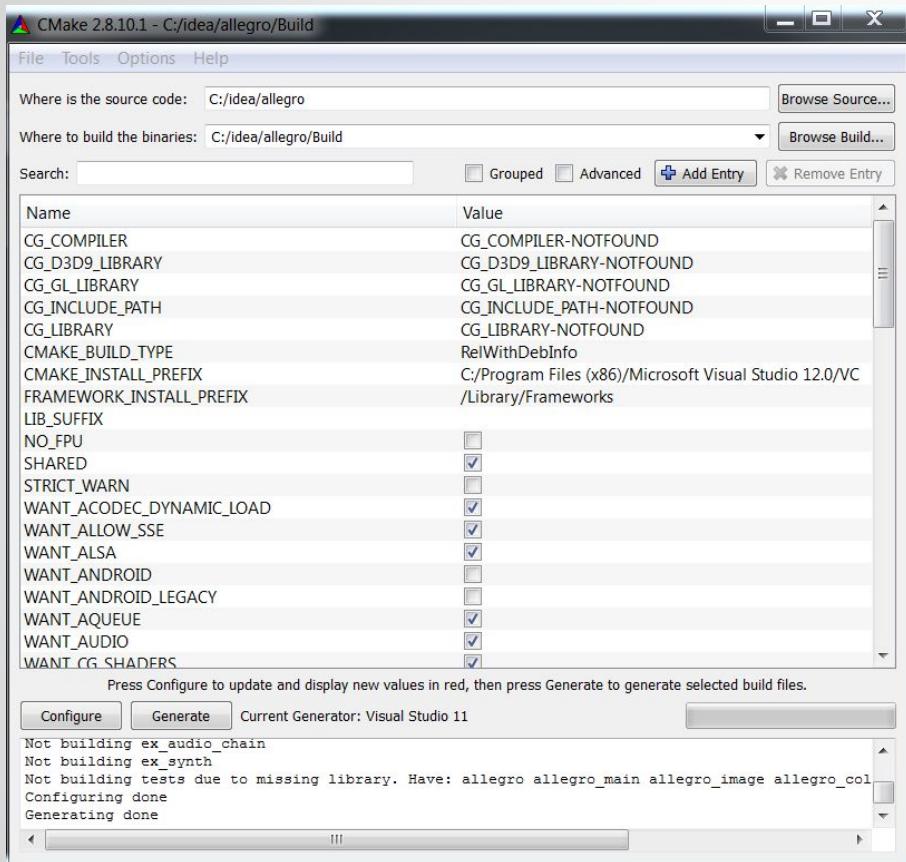
C - Android, iOS, UWP



CMake

Allegro





Allegro 5.1

Win, Lin, Mac

iOS, Android

<http://alleg.sourceforge.net/a5docs/refman/>

Initialization

al_init();

Graphic environment

```
al_create_display(int w, int h)
```

Conemu Maximus 5

Powerful terminal for Windows

use with PowerShell, Python, Ruby...



<https://code.google.com/p/conemu-maximus5/>



CMake 3.0.2

By: scaftw

CMake is a family of tools designed to build, test and package software. CMake is used to control the software compilation process using simple platform and compiler independent configuration files. CMake generates native makefiles and workspaces that can be used in the compiler environment of your choice.

3,884 downloads | Tags make build test package

```
C:\> choco install cmake
```

Yum/Apt-like installation of Win packages
<https://chocolatey.org>

SDL

Simple Directmedia Layer



Made with SDL

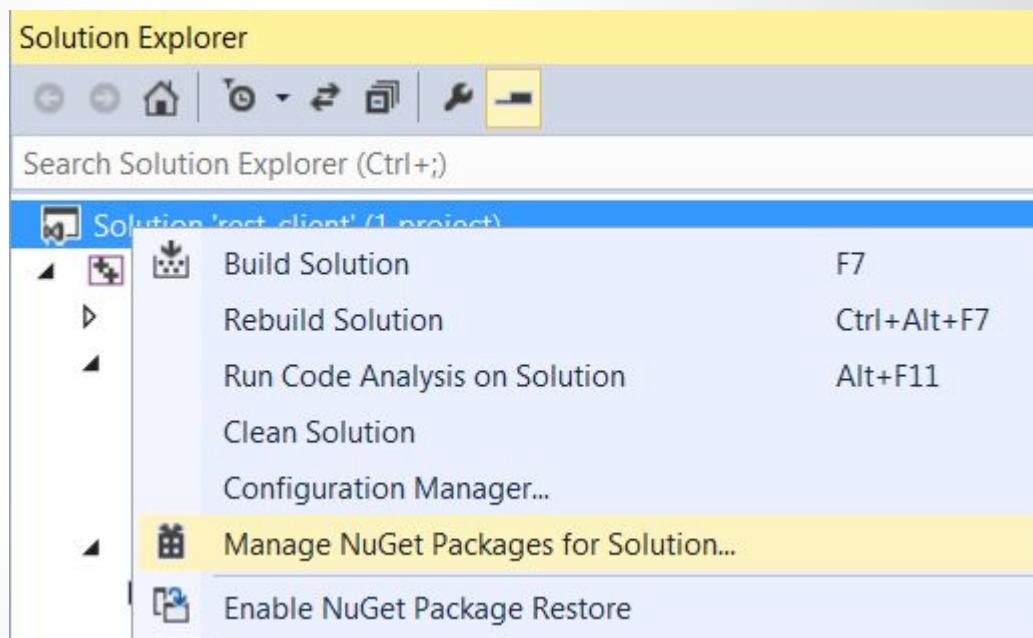


Made with SDL



NuGet - <http://www.nuget.org>





▶ Installed packages

Stable Only ▾ Sort by: Relevance ▾

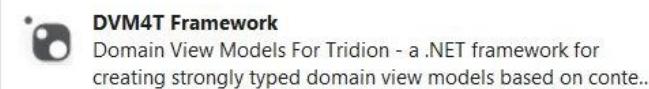
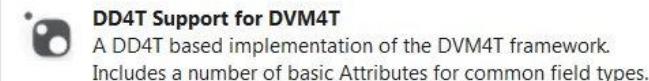
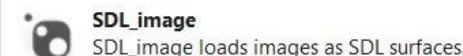
sdl ▾

◀ Online

[All](#)
[nuget.org](#)
[Microsoft and .NET](#)
[Search Results](#)

▶ Updates

Each package is licensed to you by its owner. Microsoft is not responsible for, nor does it grant any licenses to, third-party packages.



Created by: Sam Lantinga, SDL contributors

Id: SDL

Version: 1.2.15.15

Last Published: 5.7.2013

Downloads: 1398

License

[View License](#)

LGPL-2.1

[Project Information](#)

[Report Abuse](#)

Description:

Simple DirectMedia Layer is a cross-platform multimedia library designed to provide low level access to audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer.
Homepage: <http://www.libsdl.org/>

Tags: sdl native CoApp nativepackage

Dependencies:

SDL.redist (\geq 1.2.15.15)

Each item above may have sub-dependencies subject to additional license agreements.

Multiplatform

SDL officially supports
Windows, Mac OS X, Linux, iOS, and Android.

Support for other platforms may be found in the
source code.

SDL versions

1.2 stable - rock solid

2.x development - new features

SDL_init(flags)

SDL_INIT_TIMER - The timer subsystem

SDL_INIT_AUDIO - The audio subsystem

SDL_INIT_VIDEO - The video subsystem

SDL_INIT_CDROM - The cdrom subsystem

SDL_INIT_JOYSTICK - The joystick subsystem

SDL_INIT_EVERYTHING - All of the above

SDL_INIT_NOPARACHUTE - Prevents SDL from catching fatal signals

SDL_INIT_EVENTTHREAD - Runs the event manager in a separate thread

Quit application

`SDL_quit()`

Window

```
SDL_CreateWindow("Hello World!", 100, 100,  
640, 480, SDL_WINDOW_SHOWN);
```

Load bitmap

```
SDL_Surface *bmp = nullptr;  
bmp = SDL_LoadBMP("smajlik.bmp");
```

Visual data

`SDL_Renderer`

`SDL_Texture`

Keyboard

```
SDL_PollEvent(SDL_Event *event)
```

```
event.key.keysym.sym
```

Timer

```
SDL_TimerID SDL_AddTimer(  
    Uint32           interval,  
    SDL_TimerCallback callback,  
    void*            param)
```

Mouse

```
SDL_GetMouseState(*x, *y);
```

Text

Not implemented



Extensions

extension for many languages:

C++, Java, PHP, Python, Ruby

PyGame

Power of C and Power of Python

<http://www.pygame.org>



Kivy.org



kivy

iOS

Android

Windows Desktop

Windows Phone

Raspberry Pi

Cross-platform development of smartphone application with the Kivy framework
- Master thesis - Mgr. Ondřej Chrastina: http://is.muni.cz/th/430596/fi_m/



Gradle Native Builds

C/C++, Objective-C

<http://gradle.org/getting-started-native/>



Build tool

Extensible by plugins

Power of Domain Specific Language

<http://plugins.gradle.org>

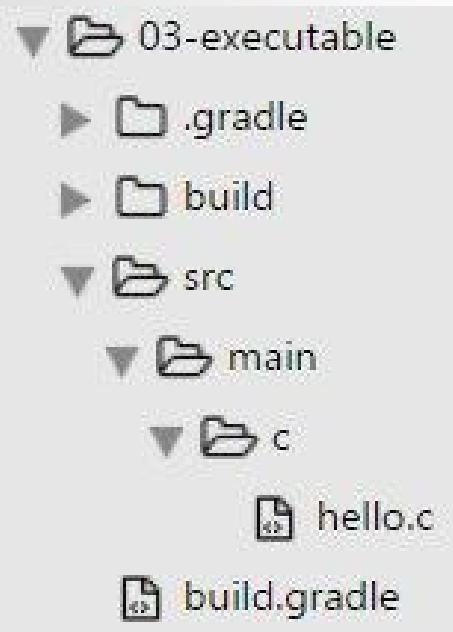


Search Gradle Plugins



search by tag or keyword

Project structure



Convention over configuration

Decrease number of decisions that developers need to make

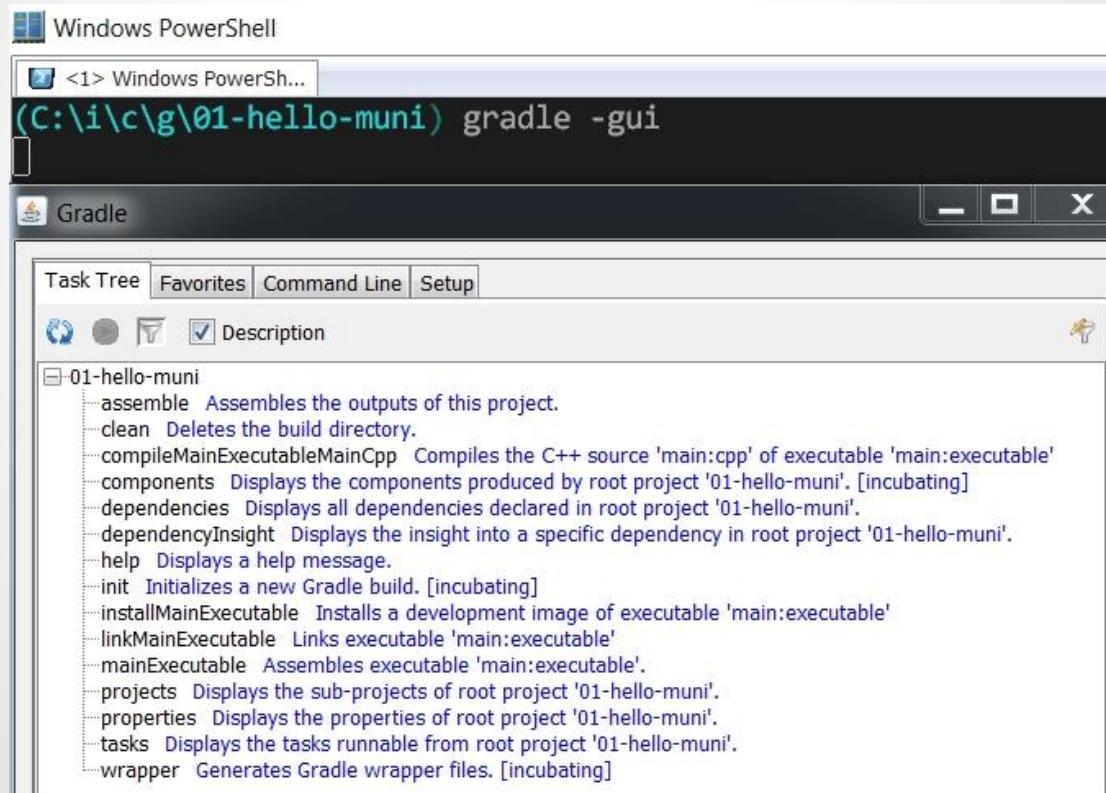
http://en.wikipedia.org/wiki/Convention_over_configuration

C plugin

```
▶ build.gradle

1 apply plugin: 'c'
2
3 model {
4     components {
5         main(NativeExecutableSpec) {
6
7             }
8         }
9     }
```

Gradle command line & GUI



gradle components

```
(C:\i\c\g\03-executable) gradle components
:components

-----
Root project

-----
Native executable 'main'
-----

Source sets
    C source 'main:c'
        src\main\c

Binaries
    Executable 'main:executable'
        build using task: :mainExecutable
        install using task: :installMainExecutable
        platform: windows_x86
        build type: debug
        flavor: default
        tool chain: Tool chain 'visualCpp' (Visual Studio)
        executable file: build\binaries\mainExecutable\main.exe
```

Gradle build Linux package

Netflix Nebula OS Package plugin:

<http://plugins.gradle.org/plugin/nebula.os-package>



```
1  plugins {
2      id "nebula.os-package" version "2.0.3"
3  }
4
5  apply plugin: 'c'
6
7 ▼ model {
8    ▼ components {
9        hello(NativeExecutableSpec) {
10            }
11        }
12    }
13 }
14
15 ▼ ospackage {
16     packageName = "hello"
17     version = "1.0"
18     release = 1
19     os = LINUX
20     packageDescription = "Linux Gradle hello package"
21     summary = "contains binary with hello world example"
22
23     from("build/binaries/helloExecutable") {
24         into "/usr/bin/"
25     }
26 }
27
28 buildDeb {
29     requires("libc6")
30 }
31
32 buildRpm {
33     requires("libc6")
34 }
```

Build package

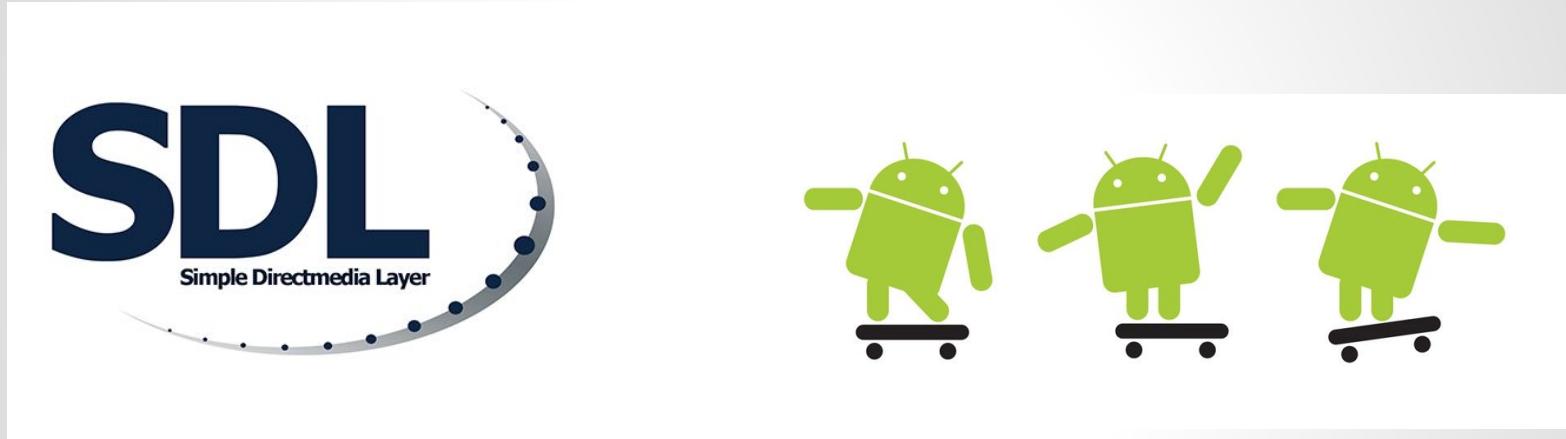
```
[georgik@pidi:pts/5]—(...-plugin/04-hello-linux-package)
[17:33:%)— gradle hE bD —(Sat, Dec 06)
:compileHelloExecutableHelloCpp
:linkHelloExecutable
:helloExecutable
:buildDeb

BUILD SUCCESSFUL

Total time: 7.12 secs
[georgik@pidi:pts/5]—(...-plugin/04-hello-linux-package)
[17:33:%)— dpkg -c build/distributions/hello_1.0-1_all.deb
drwxr-xr-x georgik/0          0 2014-12-06 17:33 ./usr/
drwxr-xr-x georgik/0          0 2014-12-06 17:33 ./usr/bin/
-rwxr-xr-x georgik/0        6367 2014-12-06 17:33 ./usr/bin/hello
```

Note: Gradle supports abbreviation. You can write hE instead of helloExecutable

SDL2 and Android



Android Studio + NDK + Gradle

IDE & Text editors

c9.io



The screenshot shows the Cloud9 IDE interface. The top navigation bar includes back, forward, refresh, file operations, and preview/run buttons. The left sidebar has sections for Workspace (with main.c selected), Navigate (visualize, bower_components, cloud-test, main.c, runme), and Commands. The main workspace shows a terminal window with the output of a C program running. The code editor displays the following C code:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Greetings from cloud!\n");
5     return 0;
6 }
```

The terminal window shows the command "cloud-test/main.c" and the output "Greetings from cloud!". The status bar at the bottom indicates "6:2 C and C++ Spaces: 4".



A hackable text editor
for the 21st Century

<https://atom.io/>

Sublime Text

Demonstration - Sublime Text 2

File Edit Selection Find View Goto Tools Project Preferences Help

base64.cc

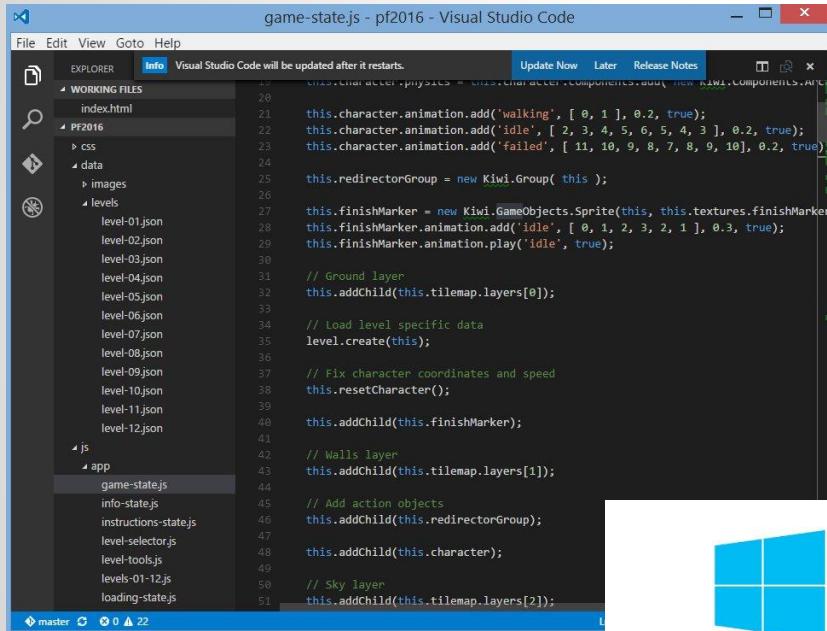
```
31 void base64_encode(const uint8_t * data, size_t leng, char * dst)
32 {
33     size_t src_idx = 0;
34     size_t dst_idx = 0;
35     for (; (src_idx + 2) < leng; src_idx += 3, dst_idx += 4)
36     {
37         uint8_t s0 = data[src_idx];
38         uint8_t s1 = data[src_idx + 1];
39         uint8_t s2 = data[src_idx + 2];
40
41         dst[dst_idx + 0] = charset[(s0 & 0xfc) >> 2];
42         dst[dst_idx + 1] = charset[((s0 & 0x03) << 4) | ((s1 & 0xf0) >> 4)];
43         dst[dst_idx + 2] = charset[((s1 & 0x0f) << 2) | (s2 & 0xc0) >> 6];
44         dst[dst_idx + 3] = charset[(s2 & 0x3f)];
45     }
46
47     if (src_idx < leng)
48     {
49         uint8_t s0 = data[src_idx];
50         uint8_t s1 = (src_idx + 1 < leng) ? data[src_idx + 1] : 0;
51
52         dst[dst_idx++] = charset[(s0 & 0xfc) >> 2];
53         dst[dst_idx++] = charset[((s0 & 0x03) << 4) | ((s1 & 0xf0) >> 4)];
54         if (src_idx + 1 < leng)
55             dst[dst_idx++] = charset[((s1 & 0x0f) << 2)];
56     }
57 }
```

5 selection regions

Spaces: 4

C++

Visual Studio Code



A screenshot of the Visual Studio Code interface. The title bar says "game-state.js - pf2016 - Visual Studio Code". The menu bar includes File, Edit, View, Goto, Help. The status bar at the bottom shows "master" and "22". The Explorer sidebar on the left shows a tree view with "WORKING FILES" expanded, showing files like index.html, PF2016 (with subfolders css, data, images, levels containing level-01.json through level-12.json), and js (with subfolders app containing game-state.js, info-state.js, instructions-state.js, level-selector.js, level-tools.js, levels-01-12.js, and loading-state.js). The main editor area displays the content of game-state.js, which is a JavaScript file defining a character's physics and animation logic for a game.

```
class CharacterPhysics = require('better-components').duel(new Kiwi.ComponentData({  
    physics: {  
        mass: 1,  
        friction: 0.1,  
        density: 1  
    }  
});  
  
this.character.animation.add('walking', [ 0, 1 ], 0.2, true);  
this.character.animation.add('idle', [ 2, 3, 4, 5, 6, 5, 4, 3 ], 0.2, true);  
this.character.animation.add('failed', [ 11, 10, 9, 8, 7, 8, 9, 10 ], 0.2, true);  
  
this.redirectorGroup = new Kiwi.Group( this );  
  
this.finishMarker = new Kiwi.GameObjects.Sprite(this, this.textures.finishMarker);  
this.finishMarker.animation.add('idle', [ 0, 1, 2, 3, 2, 1 ], 0.3, true);  
this.finishMarker.animation.play('idle', true);  
  
// Ground layer  
this.addChild(this.tilemap.layers[0]);  
  
// Load level specific data  
level.create(this);  
  
// Fix character coordinates and speed  
this.resetCharacter();  
  
this.addChild(this.finishMarker);  
  
// Walls layer  
this.addChild(this.tilemap.layers[1]);  
  
// Add action objects  
this.addChild(this.redirectorGroup);  
  
this.addChild(this.character);  
  
// Sky layer  
this.addChild(this.tilemap.layers[2]);  
  
});
```

Code editing. Redefined.
- <https://code.visualstudio.com/>



 Windows

Windows 7, 8, 10

 .deb

Debian, Ubuntu

 .rpm

Red Hat, Fedora, CentOS

 OS X

OS X Yosemite, El Capitan



CLion

Toolchain detection



CLion

Customize CLion

UI Themes → Toolchains → Default plugins → Featured plugins

Configure MinGW/Cygwin, CMake and GDB

Environment:

Use MinGW home: C:\MinGW
MinGW version: 3.20

Use Cygwin home: C:\cygwin
Cygwin version: 1.7.15

CMake executable:

Use bundled CMake 3.2.2

Use specified: []

GDB executable:

Use bundled GDB 7.8

Use specified: []

⌘ Environment
⌘ CMake
⌘ make
⌘ C Compiler
⌘ C++ Compiler
⌘ GDB

Toolchains can be changed later in Settings | Build, Execution, Deployment | Toolchains

[Back to UI Themes](#) [Next: Default plugins](#)

Edit project



CLion

The screenshot shows the CLion IDE interface. The title bar reads "hellofi - [C:\Users\georgik\ClionProjects\hellofi] - ...\\main.c - CLion 1.0.1". The menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The toolbars include standard icons for file operations and build status. The left sidebar displays the project structure under "hellofi": CMakeLists.txt, main.c (selected), and External Libraries. The main editor window shows the code for main.c:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hello Brno!\n");
5     return 0;
6 }
```

The bottom panel shows the terminal output of the "Build All" command:

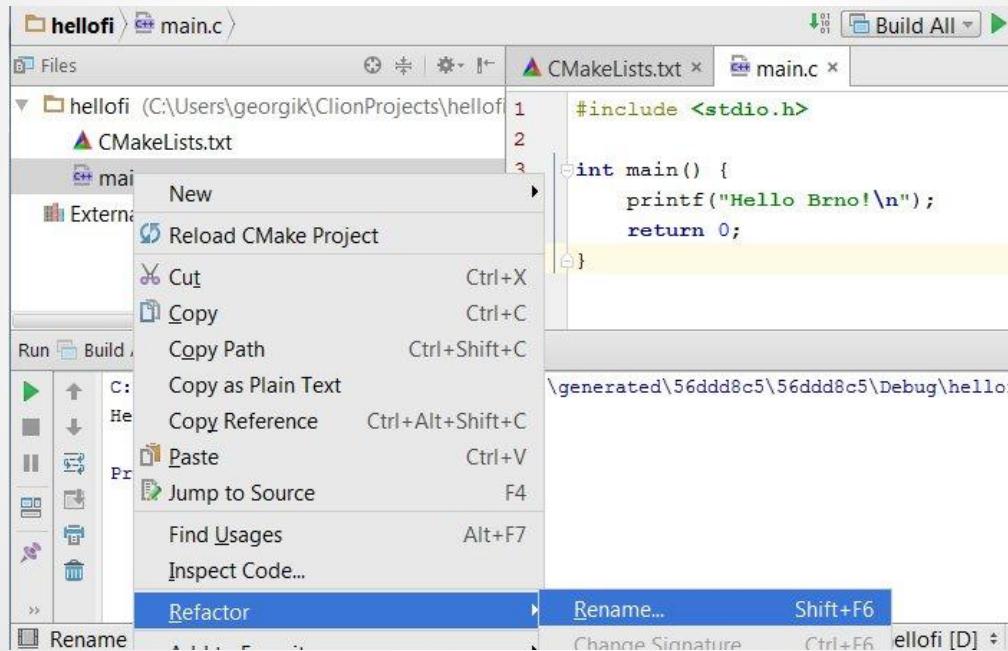
```
Run Build All
C:\Users\georgik\.clion10\system\cmake\generated\56ddd8c5\56ddd8c5\Debug\hellofi.exe
Hello Brno!
Process finished with exit code 0
```

At the bottom, a status bar indicates "Build finished in 2s 145ms (4 minutes ago)" and "5:1 LF windows-1250 Context: hellofi [D]".

Leverage Refactor



CLion



Use Debugger



A screenshot of the CLion IDE interface. The title bar shows "hellofi - [C:\Users\georgik\ClionProjects\hellofi] - ...\\main.c - CLion 1.0.1". The menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The toolbar has icons for Build All, Run, Stop, and Debug. The left sidebar shows a project tree with "hellofi" containing "CMakeLists.txt" and "main.c", and an External Libraries section. The main editor window displays the following C code:

```
#include <stdio.h>
int main() {
    printf("Pay a tribute\n");
    int answer = 42; answer: 42
    printf("header set X-Clacks-Overhead:");
    printf("GNU Terry Pratchett\n");
    printf("%i\n", answer);
    return 0;
}
```

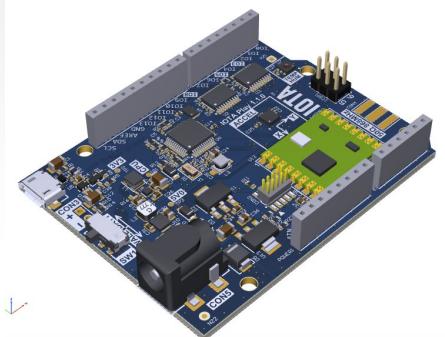
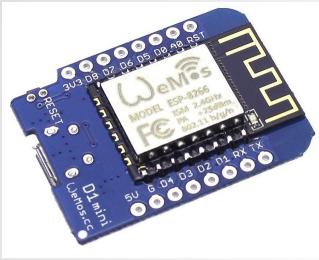
The line "answer = 42" is highlighted with a red arrow icon, indicating it is the current line of execution. The bottom panel contains a "Debug" tab, a "Console" tab showing the output "Pay a tribute header set X-Clacks-Overhead:GNU Terry Pratchett", and a "Variables" tab showing "answer = {int} 42". The status bar at the bottom says "Build finished in 774ms (a minute ago)".

Fine tune



The image shows two overlapping screenshots of the CLion settings interface. The top screenshot displays the 'Editor > General' section under the 'Mouse' heading. It contains three checkboxes: one checked ('Honor "CamelHumps" words settings when selecting double click'), one unchecked ('Change font size (Zoom) with Ctrl+Mouse Wheel'), and one checked ('Enable Drag'n'Drop functionality in editor'). The bottom screenshot shows the 'Editor > General > Appearance' section. It lists several appearance-related options with checkboxes: 'Use anti-aliased font' (checked), 'Caret blinking (ms): 500' (checked), 'Use block caret' (unchecked), 'Show right margin (configured in Code Style options)' (checked), 'Show line numbers' (unchecked), 'Show method separators' (unchecked), 'Show whitespaces' (unchecked), 'Leading' (checked), and 'Inner' (checked). The left sidebar of both screenshots includes sections for Keymap, Editor (with General selected), Live Templates, and Images.

Disabled by default for all JetBrains tools :-(



C in embedded and IoT world



<https://github.com/ysoftiota>

Arduino + Platform IO



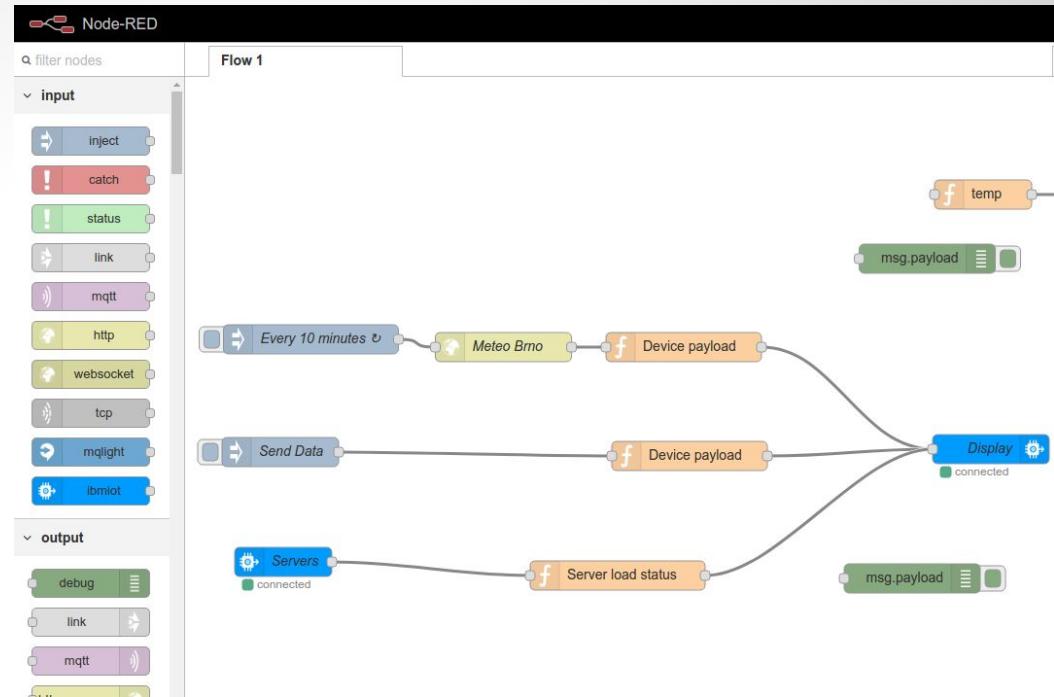
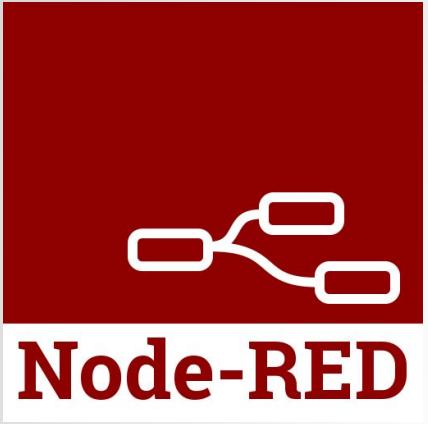
<http://platformio.org/>

LampESP example

<https://github.com/georgik/LampESP>

- OTA
- WifiManager
- Web Server
- MQTT Client (works also with Bluemix)
- TaskScheduler (async style)

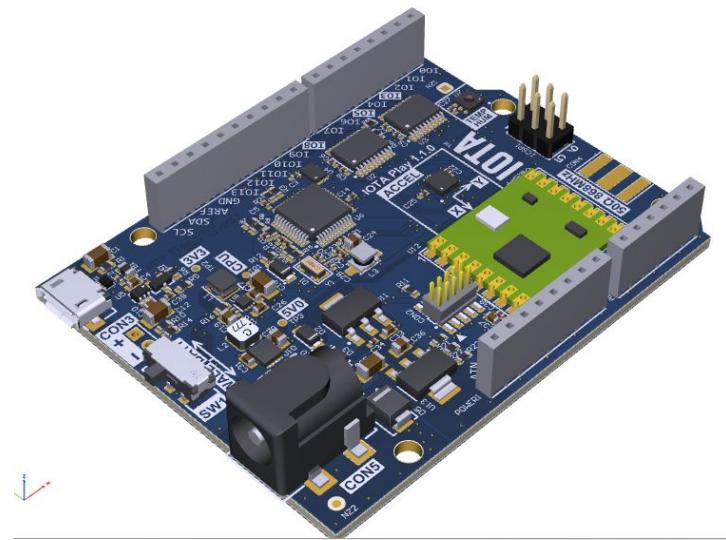
More info: <http://georgik.rocks/category/iot/>



<https://nodered.org/>

YSoft Iota Play

<https://github.com/ysoftiota/yi-play>



Když firmy pořizují vývojářům elektroniku „na hraní“:

<https://www.lupa.cz/clanky/senzory-martina-maleho-kdyz-firmy-porizuji-vyvojarum-elektroniku-na-hrani/>

Monkey C

Garmin Connect IQ

- <https://developer.garmin.com/connect-iq>



iot-inc - podcast



<http://www.iot-inc.com/category/mediatype/podcasts/>

Bastlíři SH

<http://macgyver.sh.cvut.cz/>

OpenAlt 2016 - video

https://openalt.cz/2016/program_detail.php#event_3135

Continuous integration



Jenkins

Atlassian
 Bamboo  TeamCity

Jenkins

The screenshot shows the Jenkins dashboard with the following interface elements:

- Left Sidebar:** Includes links for People, Build History, Project Relationship, Check File Fingerprint, Disk usage, and We Need Beer.
- Build Queue:** A panel stating "No builds in the queue."
- Build Executor Status:** A panel for the "master" executor showing one build in progress: "infra_backend_pull_request greeter" (#2050), with a progress bar at approximately 50% completion.
- Top Navigation:** Buttons for Jenkins, All, All Disabled, All Failed, All Unstable, Infrastructure, and Jenkins core.
- Build List:** A table displaying ten builds with columns for Status (S), Weather icon, and Name. The builds listed are:
 - config-provider-model (Yellow Sun)
 - core_selenium-test (Red Cloud)
 - fix-git-configuration-on-remote-slave-8 (Yellow Sun)
 - infra_accountapp (Yellow Sun)
 - infra_backend-confluence-spam-remover (Yellow Sun)
 - infra_backend-merge-all-repo (Red Cloud)
 - infra_backend-plugin-report-card (Yellow Sun)
 - infra_backend-war-size-tracker (Red Cloud)
 - infra_backend_jenkins_ci_cloubess_com_filler (Red Cloud)

Hit for Windows users: Do not install Jenkins into path with special characters and blank space.
E.g: Wrong: C:\Program Files (x86)\Jenkins. Correct: Use C:\projects\jenkins

From desktop to cloud

Software is slow

Software is hard to write

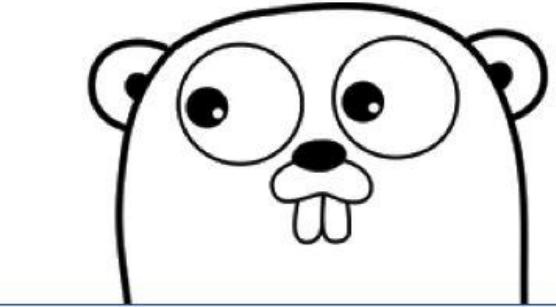
Software is hard to scale

Go

<http://golang.org>

Authors:

- Ken Thompson - known for Unix
- Rob Pike - known for UTF-8
- Robert Griesemer



Main features of language

syntax patterns from dynamic languages

performance of C

blazing fast compilation

output one binary

concurrency

libraries from internet (e.g. Github)

works on: Mac, Linux, Windows and more...

Materials

Andreas Krennmair

<http://synflood.at/tmp/golang-slides/mrmcd2012.html#1>

Steve Francia

<http://spf13.com/presentation/first-go-app/>

Thanks to artists

images used in this presentation were published under creative commons license. Links to originals:

<http://www.flickr.com/photos/fatboyke/3405148748/>

<http://www.flickr.com/photos/teveve/6301993588/>

<http://www.flickr.com/photos/stevewilhelm/6242822362/>

<http://en.wikipedia.org/wiki/Chess>

<http://www.flickr.com/photos/akosma/9486807123/>

<http://www.flickr.com/photos/charlestilford/6362884553/>

<http://www.flickr.com/photos/anieto2k/4455227465/>

<http://www.geograph.ie/photo/1113036>

[http://commons.wikimedia.org/wiki/File:Dark_Sky_\(3274525313\).jpg](http://commons.wikimedia.org/wiki/File:Dark_Sky_(3274525313).jpg)

<http://www.elfwood.com/~arknott/Red-Dragon.2539297.html>

<http://commons.wikimedia.org/wiki/File:Wolf-River-swamp-North-Mississippi.jpg>

<http://pako0007.deviantart.com/art/Zombie-Imp-2-267822507>

<http://www.flickr.com/photos/bogenfreund/367091428/>

<http://www.flickr.com/photos/infinite-magic/4016608841/>

<http://www.flickr.com/photos/lennysan/4403695597/>

<http://www.flickr.com/photos/avaverino/4870587458/>

<http://www.flickr.com/photos/ciat/6917871707/>

L10N - verify your translations



<http://www.microsoft.com/Language>



Swiss knife tool for web <https://curl.haxx.se/>

Generate source code:

```
curl http://www.ysoft.com -o index.html --libcurl download.c
```

What's next?

GOTO 2016

The Future of Software Engineering

- Mary Poppendieck

<https://youtu.be/6K4ljFZWgW8>



**The future is already
here — it's just not
very evenly
distributed.**



Twitter: [@ysoftdevs](#)

GitHub: [github.com/ysoftdevs](#)

Blog: [www.ysofters.com](#)

Technology Hour: [www.meetup.com/ysoft-th](#)

Thesis: [Andryi.Stetsko@ysoft.com](#)