C++ in our world

8.12. 2014 FI MUNI
Brno
@jurajmichalek
http://www.ysofters.com
Grab the source code

https://github.com/ysoftdevs/cpp-examples
Who am I?

Blog: http://georgik.sinusgear.com
C++ today
NuGet
REST communication
Gradle & C++
Jenkins
IDEs
Go language
Programming languages we know strongly influence the way we think about programming.

- JS Conf 2014 - Jenna Zeigen
Breeze of fresh ideas starts blowing from NodeJS, AngularJS and others
Old rust is falling apart

New shiny tools and libraries

Visual Studio 2013

C++

C++ REST SDK

gradle
NuGet - http://www.nuget.org
C++ REST SDK
The C++ REST SDK is a cross-platform, modern, a...

C++ REST SDK Redist
Redistributable components for for package 'cpprestsdk'

TheMovieDb

librets-dotnet
A RETS client library, originally writing in cross platform C++ this contains the .NET...

TVACodeLibrary
This TVA Code Library is a collection of class libraries that extends and expands th...

Created by: casablanacore
Id: cpprestsdk
Version: 1.3.1
Last Published: 14.11.2013
Downloads: 1430
License
View License
Project Information
Report Abuse

Description:
This library is a Microsoft effort to support cloud-based client-server communication in native code using a modern asynchronous C++ API design. The C++ REST SDK (codename Casablanca) is a project to start exploring how to best support C++ developers who want to take advantage of the radical shift in software
C++ REST SDK (codename "Casablanca")

http://casablanca.codeplex.com/

Windows

Linux
C++ Rest SDK

Talk: http://youtu.be/mvDDHxBrwU8

Slides: https://www.codeplex.com/Download?ProjectName=casablanca&DownloadId=683527

Example: rest-client
http_client client("http://feeds.feedburner.com/PythonInsider?format=xml");
http_request request(methods::GET);
std::wcout << "Initializing client" << std::endl;
client.request(request).then([](http_response response)
{
    // Perform actions here to inspect the HTTP response...
    if (response.status_code() == status_codes::OK)
    {
        std::wcout << "Ok."
    } else {
        std::wcout << "Error."
    }

    std::wcout << " Result code : " << response.status_code() << std::endl;
    std::wcout << "Response, reading 512 bytes" << std::endl;
    istream bodyStream = response.body();
    container_buffer<std::string> inStringBuffer;

    return bodyStream.read(inStringBuffer, 512).then([&inStringBuffer](size_t bytesRead){
        const std::string &text = inStringBuffer.collection();
        std::wcout << text.c_str();
    });
}).wait();
Illustrator plugin

http://www.adobe.com/devnet/illustrator/sdk.html

Demo: LiveDropShadow
Fix for VS2013: http://bit.ly/1w0lxZ8
Photoshop plugin

http://www.adobe.com/devnet/photoshop/sdk.html
L10N - verify your translations

http://www.microsoft.com/Language
Conemu Maximus 5

Powerful terminal for Windows use with PowerShell, Python, Ruby…

https://code.google.com/p/conemu-maximus5/
Yum/Apt-like installation of Win packages

https://chocolatey.org
Gradle Native Builds
C/C++, Objective-C

http://www.gradle.org/docs/current/userguide/nativeBinaries.html
Build tool

Extendible by plugins

Power of Domain Specific Language
Plugin system

Motivation

- focused functionality is added by plugins
- reuse patterns and practices
- avoiding boilerplate build code

Tons of plugins: http://plugins.gradle.org/
Project structure

Convention over configuration

Decrease number of decisions that developers need to make

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

```groovy
apply plugin: 'cpp'

executables {
    main
}
```

Recent tasks

- 01-hello-muni [mainExecutable]

All tasks

- 01-hello-muni
  - assemble
  - clean
  - compileMainExecutableMainCpp
  - InstallMainExecutable
  - linkMainExecutable
  - mainExecutable
Gradle command line & GUI

(C:\i\c\g\01-hello-muni) gradle -gui

- assemble Assembles the outputs of this project.
- clean Deletes the build directory.
- compileMainExecutableMainCpp Compiles the C++ source 'main.cpp' of executable 'main:executable'
- components Displays the components produced by root project '01-hello-muni'. [incubating]
- dependencies Displays all dependencies declared in root project '01-hello-muni'.
- dependency:insight Displays the insight into a specific dependency in root project '01-hello-muni'.
- help Displays a help message.
- init Initializes a new Gradle build. [incubating]
- installMainExecutable Installs a development image of executable 'main:executable'
- linkMainExecutable Links executable 'main:executable'
- mainExecutable Assembles executable 'main:executable'.
- projects Displays the sub-projects of root project '01-hello-muni'.
- properties Displays the properties of root project '01-hello-muni'.
- tasks Displays the tasks runnable from root project '01-hello-muni'.
- wrapper Generates Gradle wrapper files. [incubating]
gradle components

(C:\i\c\g\01-hello-muni) gradle components
:components

Root project

Native executable 'main'

Source sets
  C++ source 'main:cpp'
    src\main\cpp

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

Download and prepare infrastructure
Gradle - compile with debug

```groovy
apply plugin: 'cpp-exe'

// Based on http://www.gradle.org/docs/current/userguide/nativeBinaries.html

binaries.all {
    if (toolChain in Gcc && buildType == buildTypes.debug) {
        cppCompiler.args "-g"
    }
    if (toolChain in VisualCpp && buildType == buildTypes.debug) {
        cppCompiler.args '/Zi'
        cppCompiler.define 'DEBUG'
        linker.args '/DEBUG'
    }
}
```
Gradle build Linux package

Netflix Nebula OS Package plugin:
http://plugins.gradle.org/plugin/nebula.os-package
plugins {
    id "nebula.os-package" version "2.0.3"
}

apply plugin: 'cpp'

executables {
    hello {}
}

ospackage {
    packageName = "hello"
    version = "1.0"
    release = 1
    os = LINUX
    packageDescription = "Linux Gradle hello package"
    summary = "contains binary with hello world example"

    from("build/binaries/helloExecutable") {
        into "/usr/bin/"
    }
}

buildDeb {
    requires("libc6")
}

buildRpm {
    requires("libc6")
}
Build package

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

Jenkins

Atlassian
Bamboo

TC
TeamCity
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
IDE & Text editors
```cpp
#include <iostream>
using namespace std;

int main() {
    // your code goes here
    cout << "Hello" << endl;
    return 0;
}
```
# Sublime Text

```c
void base64_encode(const uint8_t *data, size_t len, char *dst)
{
    size_t src_idx = 0;
    size_t dst_idx = 0;
    for (; (src_idx + 2) < len; src_idx += 3, dst_idx += 4)
    {
        uint8_t s0 = data[src_idx];
        uint8_t s1 = data[src_idx + 1];
        uint8_t s2 = data[src_idx + 2];

        dst[dst_idx + 0] = charset[(s0 & 0xfc) >> 2];
        dst[dst_idx + 1] = charset[(s0 & 0x03) << 4 | (s1 & 0xf0) >> 4];
        dst[dst_idx + 2] = charset[(s1 & 0x0f) << 2 | (s2 & 0xc0) >> 6];
        dst[dst_idx + 3] = charset[(s2 & 0x3f)];
    }

    if (src_idx < len)
    {
        uint8_t s0 = data[src_idx];
        uint8_t s1 = (src_idx + 1 < len) ? data[src_idx + 1] : 0;

        dst[dst_idx++] = charset[(s0 & 0xfc) >> 2];
        dst[dst_idx++] = charset[(s0 & 0x03) << 4 | (s1 & 0xf0) >> 4];
        if (src_idx + 1 < len)
            dst[dst_idx++] = charset[(s1 & 0x0f) << 2];
    }
}
```
```cpp
#include <iostream>

using namespace std;

int main() {
    cout << "Hello, World!" << endl;
    return 0;
}
```
Qt Creator

```cpp
#include <QApplication>
#include "mainwindow.h"

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);
    app.setOrganizationName("QtProject");
    app.setApplicationName("Application Example");
    QMainWindow mainWin;
    mainWin.show();
    return app.exec();
}
```
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 Krennmaier
http://synflood.at/tmp/golang-slides/mrmcd2012.html#1

Steve Francia
YSofters

Twitter: @ysoftdevs
GitHub: github.com/ysoftdevs
Blog: www.ysofters.com
Technology Hour: www.meetup.com/ysoft-th