

## ALEGEREA TEMEI

- **Am ales aceasta tema datorita interesului pentru muzica** al participantului, si datorita lipsei de optimizare a limbajelor de programare curente.
- Un astfel de limbaj de programare nu permite mai mult de un tip de alocare de memorie.

### - ANALIZA - Am efectuat studiul pieței:

- **JavaScript, Python:** Allocation is made by Java, but it is not customizable.
- **C++:** Allocation is made by the programmer, which, even if optimal (faster at runtime), is challenging.
- **C#:** Too many similar classes
- **Swift:** Slow Runtime
- **Definirea Problemei:** - imposibilitatea de a alege modul de alocare a memoriei ( de catre softist sau de catre soft ) folosind un limbaj de tip interpretor.

Lfo este un limbaj de programare general purpose bazat pe C++, care este focusat pe alocare avansata de memorie.

**Am realizat 2 fisiere sh, pentru platforma Linux.**

- **lfointerpreter** = interpretator de limaje high-level
- **lfocreator** = IDE pentru lfointerpreter + **4 demos**

### AVANTAJE

- **Sintaxa usor de utilizat**
- **Sintaxa faciliteaza imbunatatirea runtime**
- **Functioneaza fara instalare**
- **IDE Customizabil**
- **Extensie unica .lfop**
- **Functii predefinite implementate avansat**

**PUBLICUL ȚINTĂ** al aplicației sunt **programatorii**.

**Tehnologii** - Am utilizat framework-ul Qt folosit și la dezvoltarea Opera și Skype.

- Am utilizat LMMS si Audacity pentru editarea sunetului si Blender pentru grafica.
- Imaginile sunt originale, iar sunetele gratuite ([www.musicradar.com](http://www.musicradar.com));
- Licența: Apache License ([www.apache.org/licenses/LICENSE-2.0](http://www.apache.org/licenses/LICENSE-2.0))

**Qt Framework:** sistem inter-platformă de dezvoltare software în **C++**; include o bibliotecă cu elemente de control pentru crearea de programe cu interfață grafică și fără interfață grafică, de ex. servere

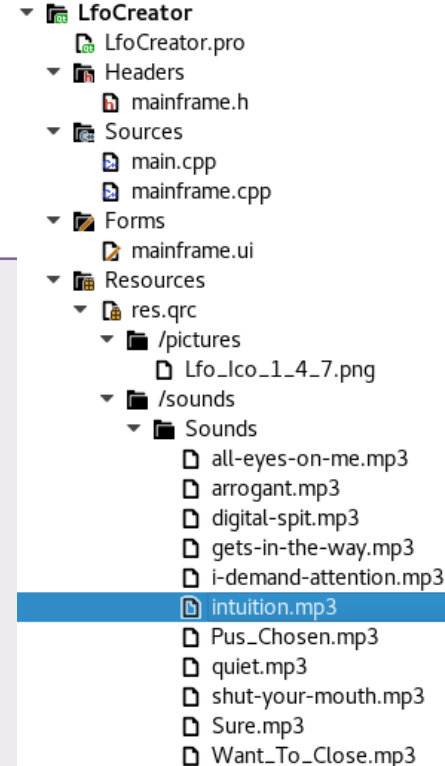
**Design Blender-** User Interface (UI)-3 ani experiență

- aplicații de animație grafică în Blender inclusiv pentru Concursul NASA-Space Settlement din 2017  
<http://wp-space-settlement.weebly.com/>



General Purpose Programming Language

# APLICAȚIA Lfo Creator



## ETAPELE DE DEZVOLTARE **Lfo**

- o Instalarea sistemului de operare Linux
- o Instalare Qt Framework C++
- o **Dupa instalările necesare, am realizat proiectul Lfo Creator (extensia .pro)**
- o **Am inclus clasele default, si clasele specifice muzicii (pentru UI).**
- o Am realizat interfața cu utilizatorul folosind butoane și comenzi C++
- o **Am realizat funcții custom si am folosit funcții Qt existente**
- o Se compilează folosind fișierul Makefile ce include *qmake* specific Qt
- o **S-au realizat doua fișiere shared library (sh) (lfointerpreter - Cpp, lfocreator - Qt)**
- o Rularea aplicației folosind fisierul *sh*, comanda: ***./Lfo Creator***

## Proiectarea APLICAȚIEI Lfo Creator

### ☐ PROGRAMARE ORIENTATĂ OBIECT (POO)

- o Tehnologia QtFramework C++
- o Extinderea proiectului se face ușor folosind obiectele existente în Qt și cele create de utilizator.

**Am folosit POO pentru a oferi facilități cum sunt:**

### ☐ FACILITĂȚI

- o **Cross-platform compatibility (Linux, Windows, etc.)**
- o **UI customizabil in IDE**
- o **RGB Selector → Send Colors to Elements**
- o **Bine Documentat**
- o

### ☐ ELEMENTE DISTINCTE - Aplicația se distinge prin

- originalitatea UI-ului
- compatibilitatea cu diverse sisteme de operare.

### ☐ PLANUL DE DEZVOLTARE este inclus in fisierul To Do de pe Github.

- o Site
- o Deb installer
- o Implementare sunete custom
- o Data Types
- o Io for Data types
- o **Eliminate user terminal interaction**
- o **Optimize code (both raw Cpp and Qt code)**
- o **Deploy for MacOS**



## General Purpose Programming Language

Am folosit funcții și tipuri de date Qt de ex. pentru:

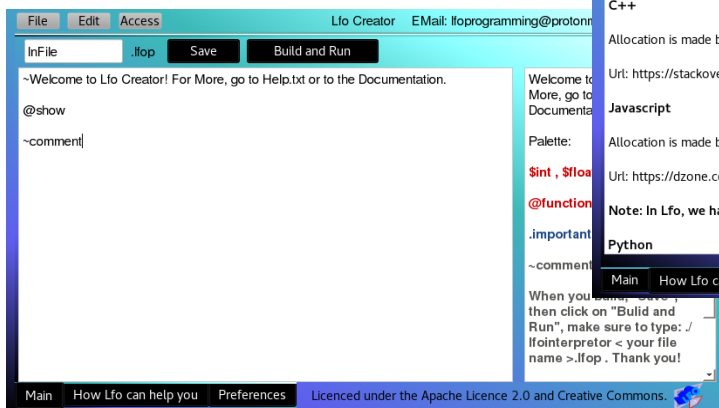
Pentru Fisiere:

```
QFile data(ui->MainLineNameIn->text()+".lfop");
if (data.open(QFile::WriteOnly | QFile::Truncate)) {
    QTextStream out(&data);
    QTextStream stream(&data);
    stream << ui->InText->toPlainText();
    data.flush();
    data.close();
    out << "Result: " << qSetFieldWidth(10) << left << 3.14 << 2.7;
}
```

Cod preluat si adaptat./ Metoda de realizare de fisiere noi Qt

```
ui->Site->setText("www.lfoprogramming.dx.am");
ui->SitesLabel->setText("<a style='color:#534496;' href='\"http://www.lfoprogramming.dx.am/\">Go! \>\></a>");
ui->SitesLabel->setTextFormat(Qt::RichText);
ui->SitesLabel->setTextInteractionFlags(Qt::TextBrowserInteraction);
ui->SitesLabel->setOpenExternalLinks(true);
```

Set Link to go to Site

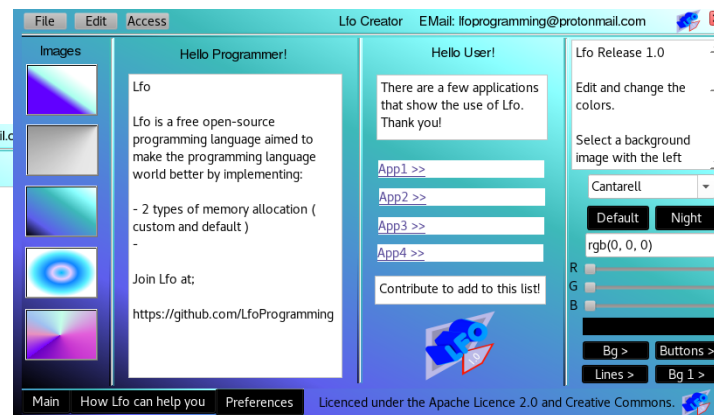


FUNCȚII Qt standard sunt utilizate si sunt implementate noi funcții C++ specific pentru Lfo Creator .  
(Specifice QMediaPlayer, QTextStream.)

**Am realizat de ex.:**

- functii de meniu (cu butoane hidden)

```
void MainFrame::on_DemoButton_clicked()
{
    on_HiddenClearButton_clicked();
    if (isDemo==0){
        ui->Demo->show();
        isDemo=1;
    } else {
        ui->Demo->hide();
        isDemo=0;
    }
}
```



```
#include "mainframe.h" #include "ui_mainframe.h"
#include <fstream> #include <string.h>
```



# General Purpose Programming Language

## Lfo Libraries:

lfoio.h  
lfomath.h

+

## Functii neincluse in clase:

\$void @dealloc \$int . . .

## Aplicatii:

- Ode to Joy
- Distanța între două puncte

## Interpretare:

Citeste fisierul si se ocupa de  
apelarea functiilor si alocarea  
de memorie

## Caracteristici ale Limbajului:

Variabile si functii, biblioteci.

## Elemente Metaprogramare:

Lfo foloseste sub anumite  
aspecte metaprogramare.

~distanța între 2  
~puncte

```
#include lfoio  
#include lfomath
```

```
$int xa  
$int xb  
$int ya  
$int yb
```

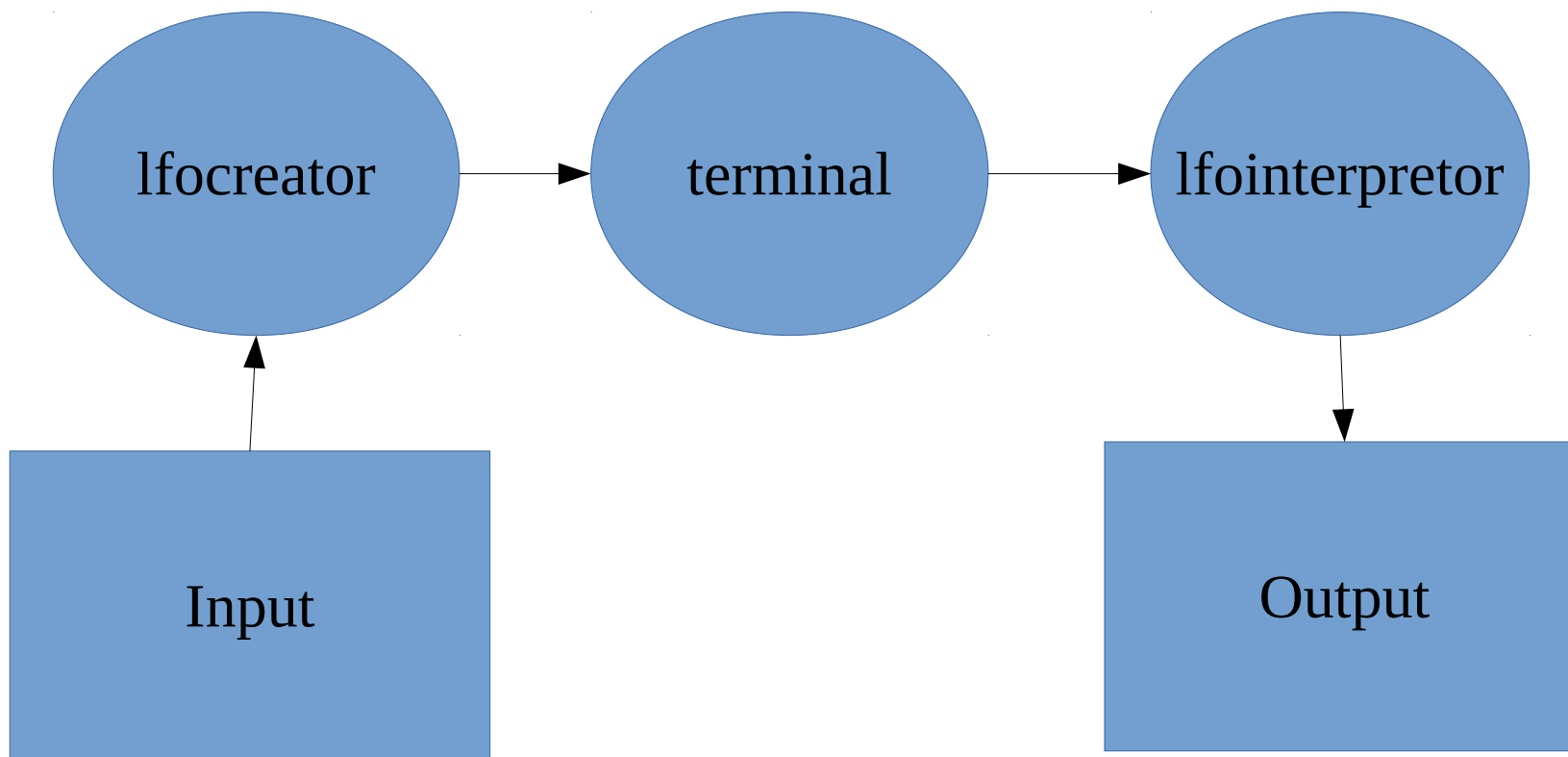
```
xa = @read  
xb = @read  
ya = @read  
yb = @read
```

```
xa = @minus $xa $xb  
ya = @minus $ya $yb
```

```
xa = @multiply $xa $xb  
ya = @multiply $ya $yb
```

```
xa = @sum $xa $ya  
xa = @sqrt $xa
```

```
@print $xa
```



# Lfo Creator

## General Purpose Programming Language

Ode to Joy

```
#include lfoio      @sleep $timeb      @sleep $timeb
#include lfomath    @beep $fa $time  @beep $mi $time
                  @sleep $timeb    @sleep $timeb
$int do = 261      @beep $mi $time  @beep $re $time
$int dod = 277    @sleep $timeb    @sleep $timeb
$int re = 293     @beep $re $time  @sleep $timeb
$int red = 311    @sleep $timeb    @beep $do $time
$int mi = 329     @beep $do $time  @sleep $timeb
$int fa = 349     @sleep $timeb    @beep $do $time
$int fad = 370    @beep $do $time  @sleep $timeb
$int sol = 392    @sleep $timeb    @beep $re $time
$int sold = 415   @sleep $timeb    @sleep $timeb
$int la = 440     @beep $re $time  @beep $mi $time
$int lad = 446    @sleep $timeb    @sleep $timeb
$int si = 493     @beep $mi $time  @beep $re $time
$int dos = 523    @sleep $timeb    @sleep $time
~notes            @beep $mi $time  @beep $do $time
                  @sleep $timec    @sleep $timed
                  @beep $re $time  @beep $do $time
                  @sleep $timed    @sleep $time
$int time = 400   @sleep $timed    @sleep $time
$int timeb = 200 @beep $re $timec
$int timec = 300 @sleep $timec
$int timed = 100 @sleep $timec
                  @beep $mi $time
                  @sleep $timeb
@beep $mi $time  @beep $mi $time
@sleep $timeb   @sleep $timeb
@beep $mi $time @beep $fa $time
@sleep $timeb   @sleep $timeb
@beep $fa $time @beep $sol $time
@sleep $timeb   @sleep $timeb
@beep $sol $time@beep $sol $time
@sleep $timeb   @sleep $timeb
@beep $sol $time@beep $fa $time
```



# Demo

