

C++ and OpenFrameworks WorkShop



Documentación intensiva/avanzada C++ y Openframeworks.

Nota: Se requiere de OpenframeWorks instalado.

OF07 - Manual C++ y Openframeworks creado por [Carles Gutierrez](#) se encuentra bajo una Licencia [Creative Commons Atribución-NoComercial-CompartirIgual 3.0 Unported](#).
Basada en una obra en [docs.google.com](#).



1

Definición C++ y OpenFrameworks:

- C++ : <http://es.wikipedia.org/wiki/C%2B%2B>
- OpenFrameworks: www.openframeworks.cc

5 reglas para construir buen código:

<http://yanpritzker.com/2009/09/29/five-rules-for-writing-good-code/>

Introducción a C++

- Leer Doc **of_simple_c++** (Openframeworks Tutorial for artists)
- Leer Doc **C++ QUICK REFERENCEC++**

- Concepto Programación de Clases y Objetos C++

Estructura típica de una clase:

- * Constructor/Destructor
- * Getters
- * Setters

Las variables, estructuras, metodos, funciones, etc de dentro de una clase son mas o menos accesibles segun sus permisos:

* **Private**: Todo aquello que use nuestra clase y que no tenga razon de ser usado o modificado desde fuera de esta clase.

* **Public**: Todo aquello lo que se necesite consultar desde otras clases

→ Ejemplo: [Primer sistema de particulas](#), tutorial de Patricio Gonzalez Vivo. Ver hasta el paso 3.

2

Punteros en programación

Concepto y usos, ver: http://es.wikipedia.org/wiki/Puntero_%28inform%C3%A1tica%29

Nomeclatura:

&x // Direccion de x

p // Contenido de la direccion p (&x igual a x)

Pasos por **referencia** en funciones.

<http://codigomaldito.blogspot.com/2005/11/paso-por-referencia.html>

Cuando se usan ? Para optimizar la memoria y el performance del programa...

→ Cuando una clase o estructura tiene que “viajar” entre varias clases o funciones, sino es un puntero o una referencia a puntero, crearía una copia “inútil” en memoria cada vez que es utilizada (en un draw o update por ejemplo >30 veces por segundo), evitar todas estas copias en memoria con un puntero o una referencia al objeto es lo mas adecuado.

→ Breve muestra del principal uso de punteros y [referencias](#) dentro de OpenFrameworks (→ ofPath, → ofImage)

Operadores c++

http://es.wikipedia.org/wiki/Operadores_de_C_y_C%2B%2B

→ Breve muestra del principal uso de los operadores dentro de OpenFrameworks (→ ofxCvColorImage)

Herencias en c++:

- Herencia en clases

Herencia simple:

http://es.wikipedia.org/wiki/C%2B%2B#Herencia_simple

Herencia multiple:

Concepto (http://es.wikipedia.org/wiki/Herencia_m%C3%BAltiple)

Las Herencias de clases sirven para absorber métodos y funciones de otras clases compatibles con ellas.

→ 8º paso del [Primer sistema de particulas](#), tutorial de Patricio Gonzalez Vivo: Variedad (herencia y polimorfismo)

Librerías

- Concepto librerías:

ANSI o estándar:

<http://www.aprendiendoaprogramar.netii.net/cmasmas/librerias.html>

No estándar

En OF: Comentar librerías que incluyen openframeworks y ver cuales se han ido introduciendo progresivamente

- En el log de OF estan las novedades y mejoras:

<http://www.openframeworks.cc/download/>
<https://github.com/openframeworks/openFrameworks/blob/develop/changes.txt>

- Concepto Wrapper de código

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

http://es.wikipedia.org/wiki/Adapter_%28patr%C3%B3n_de_dise%C3%B1o%29

- ofxOpencv
- [ofxKinect](#) an openFrameworks wrapper for the xbox kinect.
- [ofxSQLite](#) SQLite addon and wrapper for openFrameworks
- [ofxASIFT](#) A wrapper for the Affine-independent SIFT feature matching library for openFrameworks
- [ofxTesseract](#) tesseract-ocr wrapper for openFrameworks
- [ofxZXing](#) openFrameworks wrapper of ZXing for detecting and decoding QR Codes in real time
- [ofxClutter](#) A wrapper for Clutter, an open source (LGPL 2.1) software library for creating fast, compelling, portable, and dynamic graphical user interface
- ofxOpenSteer [Animation](#) an openframeworks wrapper for openSteer

3

Introducción a Openframeworks

Código de Openframeworks:

OF code style: <https://github.com/openframeworks/openFrameworks/wiki/openFrameworks-Coding-style-guidelines>

Addons OpenframeWorks

Como usar y programar Addons para OpenFrameworks

<http://ofxaddons.com/>

++ Buscar SDKs (tipo Canon) o Librerías (tipo xxx) que los alumnos les gustaria tener, hacer una investigación de de viabilidad (licencia libre, y que esté en c/c++) y determinar que necesitaria dicho Addon.

- Muestra de como construir un addon para OF, buscando un standart:
<http://ofxaddons.com/howto>

- Customizarse clases en C++ para trabajo en grupo con OF

Hacer un código limpio y sobretodo modular.

- TesApp limpio de lineas de código, solo llamadas a las funciones setup, update y draw de las clases principales.
- Para ello debe de haber clases Manager que encapsulen el codigo.

Psudocodigo TesApp:

```
→ Setup1()
→ Setup2()

→ update1()
→ update2()

→ draw1()
→ draw2()
```

4

- Estructuras de datos:

Struct, Pilas, Pilas Ordenadas, Arrays, etc.

<http://c.conclase.net/edd/>

++ Ejemplo de uso de los pixeles de una imagen. unsigned char *
(ventajas de usar unsigned char *) vs una textura.

Tablas HASH.

Similar usar STL map: <http://www.cplusplus.com/reference/stl/map/>

++ STL Map. <http://forum.openframeworks.cc/index.php/topic,5751.0.html>

- Vectores, uso y optimización.

Vectores STL

<http://arturocastro.net/blog/2011/10/28/stl::vector/>

Lectura de ficheros (XML)

- ofxXmlSettings: Ver ejemplo en Openframeworks

Timers: Concepto y uso.

Hacer contadores independientes de la velocidad del procesador.
Visualizaciones de movimientos, Maquinas de estado.

Maquinas de estados.

- http://es.wikipedia.org/wiki/M%C3%A1quina_de_estados

Se suele programación de un flujo de estados diseñado para manejar los estados de un programa o de un objeto que tiene mas de dos estados.

→ *Ejemplos* [ofxTimer](#), [ofxPocoEventTimer](#)

GUI

Graphical User Interface

→ [ofxSimpleGuiToo](#) and [others OF Gui](#)