



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
SUBDIRECCIÓN ACADÉMICA
Departamento de Formación Integral e Institucional
Comisión Académica de Trabajos Terminales



CDMX, 21 de mayo de 2019
DFII/CATT/DICT/II/2019-A053/2019

C. JIMENEZ MARURI PEDRO
C. VALENCIA RODRIGUEZ FERNANDO QUETZALCOATL
C. VILLEGAS DORANTES VANESSA

PRESENTES

Con base en los lineamientos establecidos en el Documento Rector de Operación y Evaluación para los Trabajos Terminales en la Escuela Superior de Cómputo, se comunica que la propuesta de Trabajo Terminal denominada **"Framework para la implementación de Interfaz Gráfica de Usuario: GUINOX"** con número de registro **2019-A053** ha sido dictaminada **APROBADA**, para realizarse en el ciclo escolar 2020-1 / 2020-2. En caso de existir observaciones al protocolo, favor de acudir con sus sinodales para atenderlas.

Sin otro particular, envío un cordial saludo.

ATENTAMENTE

"La Técnica al Servicio de la Patria"

M. EN E. MARIANA GÓMEZ TRESS

SECRETARIA EJECUTIVA DE LA

COMISIÓN ACADÉMICA DE TRABAJOS TERMINALES

MGT/jmmh

INSTITUTO POLITÉCNICO NACIONAL



ESCUELA SUPERIOR DE CÓMPUTO

COMISIÓN ACADÉMICA DE TRABAJO TERMINAL

C. A. T. T.

c.c.p. **MIRIAM PESCADOR ROJAS**.-Dirección del Trabajo Terminal

UKRANIO CORONILLA CONTRERAS.- Dirección del Trabajo Terminal

Framework para la implementación de Interfaz Gráfica de Usuario: GUINOX

Trabajo Terminal No. 2019-A053

Alumnos: Jiménez Maruri Pedro, Valencia Rodríguez Fernando Quetzalcoatl, Villegas Dorantes Vanessa*

Directores: Pescador Rojas Miriam, Coronilla Contreras Ukranio

Turno para la presentación del TT: Matutino e-

mail: guinox.ipn@gmail.com

Resumen – El propósito de este trabajo terminal es programar un framework de código abierto para C++, que permita implementar una Interfaz Gráfica de Usuario (GUI), utilizando HTML5 (“HyperText Markup Language”) y CSS3 (“Cascading Style Sheets”) para definición de la GUI. A partir de estos se creará un Modelo de Objetos de Documento (DOM), al cual se podrá acceder por medio de una interfaz en C++ 11, y renderizado por GPU (“Graphics Processing Unit”). Con esto se pretende dar una solución de mejor rendimiento en tiempo de procesador y memoria frente a otras herramientas actuales. Se tomarán como objetivos las plataformas Linux y Android.

Palabras clave – Framework, C++, Interfaz Gráfica de Usuario, Modelo de Objetos de Documento.

1. Introducción

En la actualidad, la mayoría de las aplicaciones requieren de un entorno para que los usuarios puedan interactuar con el programa en ejecución. El cual debe ser visualmente atractivo, sencillo de usar e interactivo. Para dar solución a estos requerimientos entran en escena las GUI.

En el lenguaje C++ es particularmente complicado realizar la implementación de una GUI. Esto debido principalmente a dos factores:

- Inicialmente fue creado para programas ejecutados en consola o en segundo plano. Por lo que carece de una forma nativa de integrar una GUI.
- Es un lenguaje que se compila. Esto quiere decir que el lenguaje es traducido a un lenguaje máquina que es dependiente de la arquitectura del dispositivo.

Las primeras herramientas que se hicieron para solventar esta problemática, se enfocan únicamente hacia sus propias plataformas. Haciendo que el programador tuviera que añadir varias bibliotecas para cada plataforma a la cual iba destinada su aplicación.

Con el tiempo surgieron proyectos que abarcaban la multiplataforma. Sin embargo, seguía siendo difícil escoger una herramienta, ya que no todas incluían todas la plataformas y algunas eran lentas, pesadas y difíciles de programar. Las herramientas más completas tenían licencias con costos bastantes elevados, que solo se podían permitir empresas bien establecidas. A pesar de todo esto, las herramientas cubrían la mayoría de las necesidades que tenían las aplicaciones de esa época.

No obstante la tecnología ha avanzado. Ahora los usuarios requieren que las GUIs tengan más funcionalidad y mejor diseño. Las nuevas herramientas tienen que cumplir con tres características principales:

1. La multiplataforma. Es necesario que con un solo código se puedan alcanzar todas o la mayoría de las plataformas que el cliente requiere.
2. Programación sencilla de la GUI. Las nuevas GUIs empiezan a ser más grandes y complejas, por lo que es necesario una mejor forma de organizar y programar. Con las viejas herramientas no hay orden ni estructura, mientras el proyecto crece el código empieza a ser complejo y desordenado.
3. Buena eficiencia. Si bien los procesadores y memorias son más rápidos y tienen más capacidad, al hacerse más complejas las GUIs se requieren más velocidad y memoria. Haciendo que las herramientas tengan que ser más eficientes.

Estas tres, son las mismas características a las que GUINOX intenta dar solución.

Herramientas similares que se han desarrollado Se analizaron las aplicaciones existentes que comparten la misma finalidad que el desarrollo propuesto en el presente documento. A continuación, se muestra la información que se ha recopilado. La tabla 1 contiene referencias a trabajos terminales realizados en el Instituto Politécnico Nacional y en la Escuela Nacional Autónoma de México relacionados con el desarrollo de GUIs.

Trabajo de Titulación	Características	Resumen	Desventajas
Herramienta de programación visual Xuitzil. [1] 1993-1997 TT0100	Reduce el tiempo de desarrollo, agarrando los objetos virtuales y soltandolos sobre otro objeto virtual para la implementación de interfaces gráficas.	Diseñar e implementar una herramienta CASE que permita desarrollar interfaces gráficas en modo texto o gráfico y entregue el código en C o C++ asociado a la misma.	- Falta de actualizaciones. - Aprender la sintaxis que te genera, para poder darle uso. - Diseño de la interfaz dentro de la programación.
Diseño de una interfaz de usuario gráfica (GUI), usando metodología orientada a objetos. [2] 1999	Utiliza la programación orientada a objetos para disminuir el tiempo de implementación.	El desarrollo de interfaces gráficas humano - computadora con el empleo de funciones gráficas de bajo nivel.	- Falta de actualizaciones. - Uso de bibliotecas externas. - Sintaxis propia de librería.

Tabla 1. Trabajos de titulación sobre Interfaz Gráfica de Usuario (GUI).

La tabla 2 enlista las herramientas existentes en el mercado.

Herramienta	Descripción	Licencias	Plataforma	Desventajas
SCITER. [3]	La principal tarea del motor es construir una interfaz web en aplicaciones de escritorio.	Free Enterprise ++ 6200 anuales.	Free Windows. Enterprise ++ Windows, Mac OS X y Linux.	- Minimización de backend en C++. - Utiliza los mismos recursos en todas las plataformas. - Utiliza eventos globales para adjuntar los widgets.

CEF (Chromium Embedded Framework). [4]	Permite a los desarrolladores agregar funcionalidad de navegación web a su aplicación, así como la capacidad de usar HTML , CSS y JavaScript para crear la interfaz de usuario de la aplicación.	Licencia BSD (Berkeley Software Distribution). [5]	Windows, Mac OS y Linux	-Requiere crear una versión propia para la funcionalidad web. -No acepta aplicaciones en tiempo real. -Retraso en comunicación de procesos.
Electron. [6]	Electron es un framework para crear aplicaciones nativas con tecnologías web como Javascript, HTML y CSS.	Licencia MIT. [7]	Windows, Mac OS y Linux.	- Utiliza dos motores JavaScript (nodejs y webkit) y conjuntos ddl. - Uso de CPU en modo inactivo. -No utiliza subprocesos, va creando procesos conforme los necesite.
Cegui. [8]	Está diseñado para la flexibilidad del usuario en apariencia y para ser adaptable a la elección del usuario en herramientas y sistemas operativos. Usa archivos XML.	Licencia MIT. [7]	Windows, Mac OS y Linux.	- Utiliza plantillas xml. - Diseñado para las necesidades de videojuegos. - Dependencia de bibliotecas externas.
WT (Web Toolkit). [9]	Biblioteca web GUI en C ++ para el desarrollo rápido de interfaz de usuario web interactivas con widgets [28], sin tener que escribir una sola línea de JavaScript.	Licencia Apache. [10]	Windows, Mac OS y Linux.	- Diseño de plantilla dentro del programa. -Para cada elemento de HTML hay un widget. -Uso de AJAX para la interactividad.

Tabla 2. Herramientas en el mercado relacionadas con Interfaces Gráficas de Usuario.

La tabla 3 muestra una comparativa de nuestra propuesta con las herramientas ya existentes en el mercado.











	GUINOX	UltraLight	SCITER	CEF	Electron	CEGUI	TNTNet	WT	GTK	NANA
Plataformas										
Renderizado por GPU	✓	✓	✓	✗	✗	✓	✓	✗	✗	✗
Dependencias	✗	✗	✗	✓	✓	✓	✓	✗	✓	✗
Independencia de archivos	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗
Licencia	—	LGPL	Free / Enterprise++	BSD	MIT	MIT	LGPL	Apache	LGPL	Boost
Uso de interpretes	✗	✗	✓	✗	✓	✗	✗	✓	✗	✗
Uso de HTML / CSS	✓	✓	✓	✓	✓	✗	✓	✗	✗	✗
Optimización de recursos	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗
Lenguaje	C++	C++	JavaScript	C++	C++ / JavaScript	C++	C++	C++ / JAVA	C / C++ / Python	C++

Tabla 3. Comparativa visual de GUINOX con herramientas similares.

En resumen, las herramientas que se encuentran activas en el mercado tienen una diversidad de características para su desarrollo, implementación y plataforma a la que van enfocadas, haciendo que todas resuelvan el problema de distintas formas. Así mismo su documentación suele estar incompleta o dispersa en diferentes sitios web, causando que los usuarios dediquen más tiempo en el aprendizaje de estas que en la implementación.

A diferencia de las herramientas que se encontraron, el framework propuesto ofrecerá al usuario información desplegada de manera intuitiva, para su fácil entendimiento; no se requerirá de estudios adicionales al lenguaje principal (C++ 11) así como de los conocimientos básicos que se tengan del Lenguaje de Marcado de Hipertexto (HTML) y el de Hojas de Estilo en Cascada (CSS).

Se usarán las versiones HTML5 y CSS3 que ya son estándares[11]. Esto por todo el desarrollo y respaldo que tienen, siendo versiones altamente aceptadas. También cuentan con nuevas formas de seleccionar elementos del DOM, nuevos elementos que hacen más clara y fácil la estructura de la información, así aumentando su usabilidad.

2. Objetivo

Crear un framework de código abierto para C++ que permita implementar una Interfaz Gráfica de Usuario (GUI) para las plataformas Android y Linux. Se usarán los estándares HTML5 y CSS3 para definir la estructura y diseño de la interfaz, lo anterior para dividir el diseño en módulos y simplificar la programación. Para dar funcionalidad a la GUI se usará C++ 11 con la finalidad de obtener mayor eficiencia en tiempo de ejecución. Además de aprovechar la Unidad de Procesamiento Gráfico (GPU) para la representación de los elementos en una ventana de aplicación.

2.1 Objetivos Particulares

Utilizar un lenguaje de marcado de hipertexto HTML5, para definir la estructura de la información en la GUI.

Utilizar hojas de estilo en cascada CSS3 que permitirá definir aspectos de presentación de la GUI.

Generar el DOM a partir de las definiciones en el HTML5 y CSS3. El DOM podrá ser modificado en tiempo de ejecución, permitiendo el cambio de atributos y estilos, así como, agregar o quitar nodos de este.

Uso de OpenGL para representar los elementos del DOM en una ventana de aplicación.

Hacer uso de eventos para interactuar con la GUI, por ejemplo: click, pulsación de teclas, click derecho, cierre de ventana.

3. Justificación

La primer pregunta a resolver es ¿Por qué C++? Existen tres lenguajes principales para aplicaciones de escritorio: C++, JAVA y Python. JAVA es el lenguaje más usado del mundo. Python fué el lenguaje más popular en el 2018. Si bien C++ no es ni el más usado del mundo, ni el más popular actualmente, siempre se ha mantenido entre los primeros 4 lugares en el top de los lenguajes [12]. Python es popular debido a que se puede desarrollar rápido y barato. No obstante, su eficiencia es muy baja, llegando incluso a ser 50 veces más lento que C++ [13]. Por esto, Python generalmente se usa para scripts cortos y entornos donde no se requiere una alta eficiencia, además de que no ha podido entrar completamente al mundo de los dispositivos móviles. JAVA por otra parte tiene un gran uso tanto en aplicaciones de escritorio como en móviles. Asimismo, ya cuenta con formas bastante populares y gratuitas de implementar GUIs. JAVA es más eficiente que Python, pero sigue teniendo poca eficiencia a comparación de C++[14]. Gracias a esta alta eficiencia, C++ se ha estado utilizando en aplicaciones donde el rendimiento es clave. Las aplicaciones más robustas y populares están hechas con C++ [15]. Con futuras actualizaciones y nuevos paradigmas que ya están en plan de ser integrados[16], este seguirá en los primeros lugares de los top de lenguajes. Aunado a lo anterior las empresas y desarrolladores, han estado adoptando cada vez más la ideología de tener una mayor eficiencia en sus programas, para poder ser más competitivos y brindar mejores soluciones. Por otra parte, C/C++ tienen una de las comunidades más grandes y activas en el mundo del software[17]. Con todo esto, podemos apostar que desarrollar en C++ hoy en día y en un futuro, es de las mejores opciones.

Como se había planteado anteriormente, es necesario que la solución sea multiplataforma. Para cubrir esta necesidad GUINOX, plantea dos puntos.

- Que sea de código abierto[18]. Al ser de código abierto, el programador podrá incluir fácilmente el framework a su proyecto y compilarlo a las arquitecturas que requiera, esto también ahorra espacio en la aplicación final. Otras ventajas, es que los mismos usuarios pueden aportar mejoras a GUINOX, mientras este se va acoplando mejor a las necesidades de los clientes. Se pueden detectar y corregir errores más rápidamente. La aceptación y uso del framework son mayores.

Otra ventaja que tiene el ser de código abierto es que los usuarios pueden crear mejoras o librerías para el framework, agregando valor a GUINOX. Todo esto bajo licencias, donde se debe referenciar el trabajo original y/o de ser el caso, se puede exigir que alguna modificación se cambie de nombre para que no se disipe la marca original[19].

- Enfocado a Linux y Android. Por cuestión de tiempo para el trabajo terminal, este solo se enfocara a estas dos plataformas. Aunque una vez finalizada la implementación para estas dos, debido a que la mayoría de nuestro código es independiente de la plataforma, solo se adaptaría una pequeña parte para Windows y IOS.

Una de las formas mejor desarrolladas para mostrar información e interactuar con el cliente, es la creación de páginas web[20]. Las páginas web se dividen en tres módulos: HTML[21], CSS[22] y JS[23]. Esto permite un mejor orden y reduce la complejidad en proyectos grandes. Es por esto que hemos preferido adoptar este esquema de trabajo. Se optó usar las versiones HTML5 y CSS3 por

el gran respaldo y desarrollo que tienen, además de contar con nuevos elementos y métodos que facilitan el diseño de interfaces[24]. Donde podemos usar HTML5 para definir la estructura de la información en la GUI, esto también evita que tengamos que definir nuestro propio vocabulario, ya que el HTML5 es ampliamente conocido entre los programadores. Asociado al HTML5, el estilo y diseño de la información recae en CSS3. Sin embargo estos solo se tomarán como base, lo cual significa que no estarán incorporadas todas la funcionalidades, refiriéndonos principalmente a las etiquetas multimedia (video, sonido, canvas), más adelante en la metodología se definen qué elementos y características son los que abarcaran este proyecto. Por último, la funcionalidad de la GUI en las herramientas más populares es implementado con intérpretes para poder hacer uso de JS y así tener el set completo de las páginas web. Esto tiene sus ventajas, como alcanzar fácilmente la multiplataforma y utilizar un lenguaje más sencillo. No obstante, el uso de intérpretes afecta notablemente el rendimiento de la aplicación, eliminando las principales ventajas que tiene C++. Aquí es donde GUINOX (ver figura 1.) difiere de estas herramientas, ya que este permitirá programar la funcionalidad de la GUI con el mismo C++, manteniendo así una alta eficiencia.

Una vez que el usuario haya definido el HTML y el CSS, el framework los leerá y creará un DOM que contendrá todos los elementos con sus atributos y estilos definidos anteriormente. GUINOX permitirá acceder y modificar el DOM por medio de una interfaz en C++11. Ya que se adoptarán la mayoría de atributos y métodos de HTML5, la forma de acceder a estos será muy similar a como se hace en JS. Pero esto solo en semántica, ya que todo esto será en C++11 y este tiene una sintaxis diferente a JS. Se usará C++11 por las características que nos proporciona para hacer una mejor codificación y diseño de nuestro framework, como son los punteros inteligentes, el uso de funciones lambdas, nuevas sobrecargas de operadores y nuevos algoritmos incluidos.

El DOM será renderizable por medio de OpenGL[25]. Esta API está incluida en la mayoría de las plataformas actuales, haciéndola multiplataforma, se acopla fácilmente a C++ y permite una programación de bajo nivel, lo que permite optimizar el renderizado por GPU. El DOM se renderiza en un ventana de aplicación[26], en la cual se pueden realizar distintos eventos para poder interactuar con la GUI. Se podrá dar funcionalidad a estos eventos con la misma interfaz en C++ con la que se accede al DOM, haciendo uso de un esquema similar al que tiene JS con los eventos.

Los eventos son la forma en la que el usuario final se comunica con la interfaz. Cuando un usuario hace click, mueve el mouse, o escribe algo en la interfaz, esto genera varios eventos a los cuales se les puede programar un comportamiento para que le entregue un resultado al usuario. GUINOX ayudará a definir qué eventos son los que el usuario podrá realizar, cómo activar y responder a esos eventos. Los eventos nos ayudan a que el programa reaccione a estímulos y no tenga que estar buscando cambios en la GUI.

Ya que Android es un sistema operativo para dispositivos móviles, donde los recursos son muy limitados, este cuenta con una versión ligera de OpenGL, la cual se llama OpenGL ES[27]. Aunque no es muy diferente la forma de trabajar con esta versión, el framework tendrá que adaptar la funcionalidad para los métodos que difieran, asimismo con los datos y eventos de entrada como el touch.

4. Productos o Resultados esperados

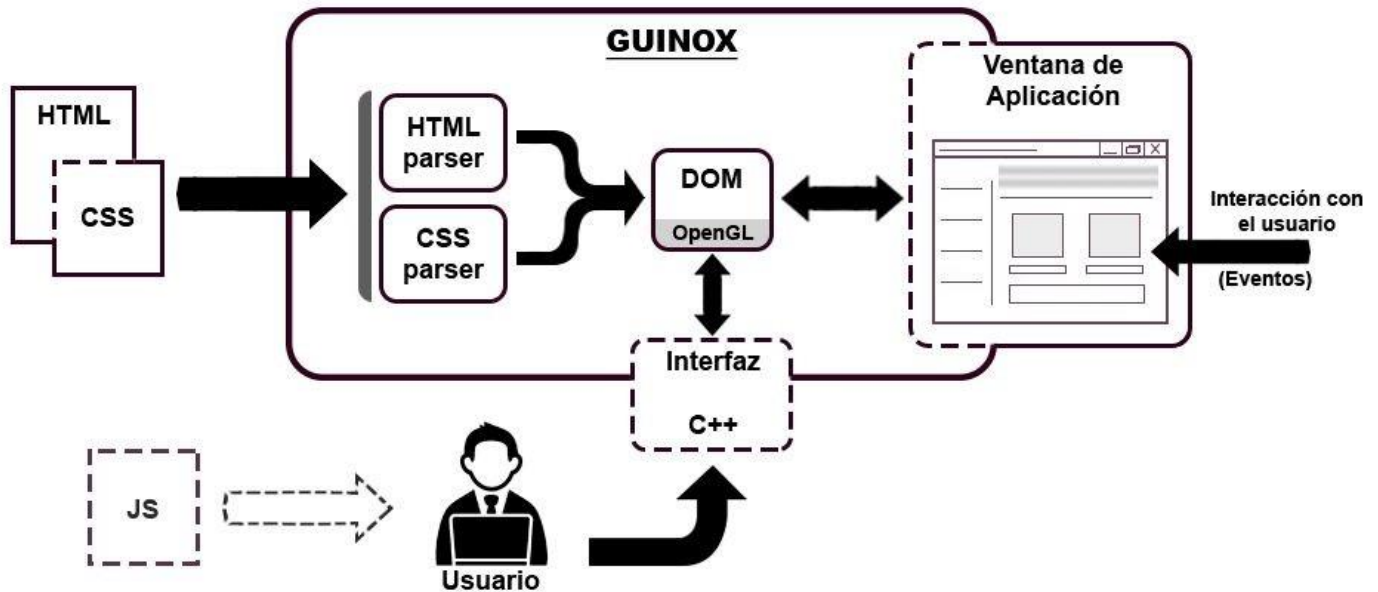


Figura 1. Diagrama a bloques del sistema.

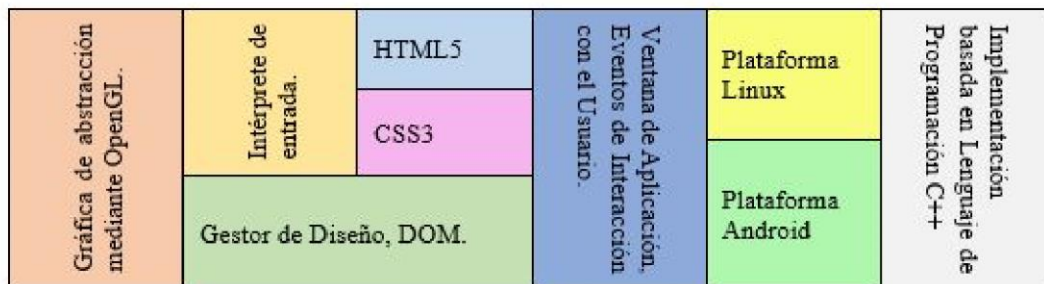


Figura 2. Módulo de herramienta

- ❑ **Gráficos de abstracción:** Implementación de gráficos que aísla los módulos de los detalles de la plataforma en particular.
 - Implementación Back-End de OpenGL (Linux, Android).
- ❑ **CSS** - Analizador de CSS que definirá aspectos de presentación de la GUI.
- ❑ **HTML** - Analizador HTML para definir la estructura de la información en la GUI.
- ❑ **DOM:** Generar el árbol DOM a partir de las definiciones en el HTML5 y CSS3. El DOM podrá ser modificado en tiempo de ejecución, permitiendo el cambio de atributos y estilos, así como, agregar o quitar nodos del mismo.

Los productos esperados al finalizar el trabajo terminal son:

1. El código.
2. La documentación técnica del sistema.
3. Página web.
 - a. Guía de uso
4. Pruebas de eficiencia

5. Comparativas contra otras herramientas.

5. Metodología

El proceso de desarrollo de software debe de ser de tipo iterativo. Este propone una comprensión incremental del problema a través de refinamientos sucesivos y un crecimiento incremental que de una solución efectiva a través de varios ciclos[28].

Cada incremento será dividido en 4 etapas, aunque hacemos mención que no todos los incrementos requerirán todas y cada una:

+Análisis, +Diseño, +Codificación, +Pruebas

Las actividades que adoptaremos de este modelo procederán de la siguiente forma:

Incremento 1: Preparar el entorno de trabajo.

Incremento 2: Definición a nivel conceptual de la herramienta, a partir de este punto se podrá definir a detalle los siguientes incrementos. Los módulos considerados son:

- HTML Parser
- Gestor DOM
- CSS Parser
- Graficador OpenGL
- Integrador de interfaz C++

Incremento 3: HTML Parser

- Básicos HTML
- Formularios e Inputs
- Imágenes. Únicamente etiqueta
- Listas , ,
- Estilo y Semántica

Incremento 4: CSS Parser 1 (Modelo de caja)

- Bordes
- Márgenes
- Padding
- Background
- Altura/Anchura
- Modelo de caja

Incremento 5: CSS Parser 2 (Estilos)

- Colores
- Texto excepto decoración, sombra, transformación, unicode-bidi
- Fuentes de Texto
- Íconos

Incremento 6: CSS Parser 3 (Posición)

- Tablas
- Listas
- Display
- Posición

→ Flexbox

Incremento 7: Creación Árbol DOM

Incremento 8: Graficador OpenGL

- Renderización de elementos
- Crear Sistema Render
- Generar Manejador de Textura
- Cargar Textura (Imágenes)
- Asignación de Textura
- Posible Optimización con Buffers

Incremento 9: Integrador de interfaz C++

- Personalizar Ventana de Aplicación
- Generar Sistema de Eventos (dependiente del SO)
- Obtener Traza de Elementos del Evento

Incremento 10: Tutorial

- Aplicación de ejemplo
- Página Web

6. Cronogramas

Nombre del alumno: Jiménez Maruri Pedro

Actividad	AGO	SEP	OCT	NOV	DIC	ENE	FEB	MAR	ABR	MAY	JUN
Incremento 1.											
Incremento 2.											
Incremento 3.											
Evaluación de TT1.											
Incremento 6.											
Incremento 9.											
Incremento 10.											
Documentación Técnica											
Evaluación de TT2.											

Nombre del alumno: Valencia Rodríguez Fernando Quetzalcoatl

Actividad	AGO	SEP	OCT	NOV	DIC	ENE	FEB	MAR	ABR	MAY	JUN
Incremento 1.											
Incremento 2.											
Incremento 4.											
Evaluación de TT1.											
Incremento 7.											
Incremento 9.											
Incremento 10.											
Documentación Técnica											
Evaluación de TT2.											

Nombre del alumna: Villegas Dorantes Vanessa

Actividad	AGO	SEP	OCT	NOV	DIC	ENE	FEB	MAR	ABR	MAY	JUN
Incremento 1.											
Incremento 2.											
Incremento 5.											
Evaluación de TT1.											
Incremento 8.											
Incremento 9.											
Incremento 10.											
Documentación Técnica											
Evaluación de TT2.											

7. Referencias

- [1] M. A. Norzagaray Cosío, “Herramienta de programación visual Xuitzil”, Instituto Politécnico Nacional, México D.F., 1997.
- [2] O. Toxtle Hernández, “Diseño de una interface usuario gráfica (GUI), usando metodología orientada a objetos”, Universidad Nacional Autónoma de México, México D.F., 1999.
- [3] Sciter, 04/03/2019, <https://sciter.com/>
- [4] Wikipedia, 04/03/2019, https://en.wikipedia.org/wiki/Chromium_Embedded_Framework
- [5] Wikipedia, 04/03/2019, https://es.wikipedia.org/wiki/Licencia_BSD
- [6] Electronjs, 04/03/2019, <https://electronjs.org/>
- [7] Wikipedia, 04/03/2019, https://en.wikipedia.org/wiki/MIT_License
- [8] Cegui, 04/03/2019, <http://cegui.org.uk/>
- [9] Webtoolkit, 04/03/2019, <https://www.webtoolkit.eu/wt/>
- [10] Wikipedia, 04/03/2019, https://es.wikipedia.org/wiki/Apache_License
- [11] Hostinger, 11/04/2019, <https://www.hostinger.com/tutorials/difference-between-html-and-html5>
- [12] TIOBE, 03/03/2019, <https://www.tiobe.com/tiobe-index/>
- [13] Benchmarksgame, 03/03/2019, <https://benchmarksgame-team.pages.debian.net/benchmarksgame/faster/gpp-python3.html>
- [14] Benchmarksgame, 03/03/2019, <https://benchmarksgame-team.pages.debian.net/benchmarksgame/faster/gpp-java.html>
- [15] Mentofacturing, 03/03/2019, <https://www.mentofacturing.com/vincent/implementations.html>
- [16] ISO CPP, 03/03/2019, <https://isocpp.org/std/status>
- [17] ISO CPP, 03/03/2019, <https://isocpp.org/blog/2017/07/a-cpp-review-community>
- [18] Open Source, 03/03/2019, <https://opensource.org/about>
- [19] Genbeta, 16/04/2019, <https://www.genbeta.com/a-fondo/cual-es-la-diferencia-entre-el-software-libre-y-el-open-source>
- [20] Stack Overflow, 03/03/2019, <https://insights.stackoverflow.com/survey/2018/#technology-programming-scripting-and-markup-languages>
- [21] Mozilla Developer, 03/03/2019, <https://developer.mozilla.org/es/docs/Web/HTML>
- [22] Mozilla Developer, 03/03/2019, <https://developer.mozilla.org/es/docs/Web/CSS>
- [23] Mozilla Developer, 03/03/2019, <https://developer.mozilla.org/es/docs/Web/JavaScript>
- [24] BBVA Open4u, 16/04/2019, <https://bbvaopen4u.com/es/actualidad/por-que-desarrollar-en-html5>
- [25] Wikipedia, 03/03/2019, <https://es.wikipedia.org/wiki/OpenGL>
- [26] Wikipedia, 03/03/2019, [https://es.wikipedia.org/wiki/Ventana_\(inform%C3%A1tica\)](https://es.wikipedia.org/wiki/Ventana_(inform%C3%A1tica))
- [27] Wikipedia, 03/03/2019, https://es.wikipedia.org/wiki/OpenGL_ES
- [28] José Salvador Sánchez Garreta. (2003). Ingeniería de Proyectos Informáticos: Actividades y Procedimientos. Castellón, España: Universitat.

8. Alumnos y Directores


Jiménez Maruri Pedro.- Alumno de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2013600931, Tel. 5567765150, email randy162009@hotmail.com.

Firma: 

Valencia Rodríguez Fernando Quetzalcoatl.- Alumno de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2014630612, Tel. 5574883827, email quetzalfir@hotmail.com.

Firma: 

Villegas Dorantes Vanessa.- Alumna de la carrera de Ing. en Sistemas Computacionales en ESCOM, Especialidad Sistemas, Boleta: 2014630527 , Tel. 5560443933 , email srta.vd@gmail.com.

Firma: 

Pescador Rojas Miriam.- Candidata a Dr. en C. en Computación del CINVESTAV/IPN 2019, M. en C. en Computación del CINVESTAV/IPN en 2010, Ing. en Sistemas Computacionales de ESCOM/IPN en 2008, Profesor de ESCOM/IPN (Departamento de Ciencias e Ingeniería de la Computación) desde 2010, Áreas de Interés: Computación Evolutiva, Graficación y modelado, Procesamiento digital de imágenes. Tel. 5729600 Ext 52022, email: miriam.pescador@gmail.com.

Firma: 
Miriam Pescador Rojas

Coronilla Contreras Ukranio.- Ing. Físico UAM-Azcapotzalco en 1997, M. en C. de la Computación UAM-Azcapotzalco en 2002, Profesor de ESCOM/IPN (Departamento de Programación y desarrollo de sistemas) desde 2001, Áreas de Interés: Sistemas Distribuidos, Inteligencia Artificial. Ext. 50233, email ukraniocc@yahoo.mx.

Firma: 
Ukranio Coronilla Contreras

CARÁCTER: Confidencial
FUNDAMENTO LEGAL: Art. 3, fracc. II, Art. 18, fracc. II y Art. 21, lineamiento 32, fracc. XVII de la L.F.T.A.I.P.G.
PARTES CONFIDENCIALES: No. de boleta y Teléfono.