UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Gºen Ingeniería en Informática

**Trabajo final del GºIng.Informática:**

**BlocklyToCFacil**

Presentado por Rosana Arnáiz Vicario
en febrero de 2018
Tutor Carlos Pardo Aguilar

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Gºen Ingeniería en Informática

D. Carlos Pardo Aguilar, profesor del departamento de Ingeniería Civil, área de Lenguajes y Sistemas Informáticos.

Expone:

Que el alumno D. Rosana Arnáiz Vicario, con DNI 71308098W, ha realizado el Trabajo final del GºIng.Informática titulado: BlocklyToChapinToCFacil.

y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual, Se autoriza su presentación y defensa.

En Burgos a 14 de febrero de 2018

Carlos Pardo Aguilar

# Índice de contenido



# Índice de ilustraciones

## Resumen

Se desea desarrollar una herramienta que ayude a iniciarse en la programación a estudiantes.

La aplicación Blockly es una herramienta muy utilizada en el inicio al aprendizaje de la programación, debido a que es muy gráfica e intuitiva. En la asignatura “Informática básica” del primer cuatrimestre del grado de Ingeniería Informática de la Universidad de Burgos se inicia a la programación con el lenguaje C_fácil, que es una adaptación del lenguaje C con una cabecera C_facil.h que facilita la lectura del lenguaje C.

En este trabajo se propone la creación de una herramienta que convierta el código creado al programar en Blockly a el lenguaje C_fácil.

**Descriptores**

Aprendizaje programación, algoritmo, conversor, C_fácil, Blockly

## *Abstract*

*It is desired to develop a tool that helps students to start programming.*

*The Blockly application is a very used tool in the beginning to learn programming, because it is very graphic and intuitive. In the subject "Basic Computer Science" of the first semester of the Computer Engineering degree at the University of Burgos, programming begins with C_fácil language, which is an adaptation of the C language with a C_facil.h header that facilitates the reading of the C language.*

*In this work we propose the creation of a tool that converts the code created from Blockly to C_fácil language.*

***Keywords***

*Learning programming, algorithm, converter, C_fácil, Blockly*

# I - *INTRODUCCIÓN*

Cada año nuevos alumnos empiezan el grado de “Ingeniería informática” en la universidad de Burgos, y muchos de ellos no tienen conocimientos de programación, por lo que les cuesta mucho empezar a programar con fluidez.

La asignatura “Informática básica” es la primera toma de contacto con la programación que los alumnos tienen en el grado y por ello, aquí es donde empiezan a entender cómo programar.

Por estas razones se utiliza en la asignatura el lenguaje C_fácil, que no es más que una cabecera que facilita la lectura del lenguaje C a los alumnos, y los prepara para asignaturas posteriores, como “Programación estructurada” del segundo semestre donde se realizan las prácticas programando en lenguaje C. También se utilizan herramientas como son Blockly, que al ser una herramienta gráfica y visual ayuda al inicio del aprendizaje de la programación, ya que convierte en código los programas creados gráficamente por la unión de bloques a los lenguajes JavaScript, Python, PHP, Lua y Dart, y además se ejecuta el programa creado en la propia aplicación.

Con este proyecto se desea desarrollar una herramienta que, una vez realizado un programa en Blockly, convierta ese programa en el lenguaje C_fácil, y de esta manera los alumnos puedan visualizar el mismo programa desde dos perspectivas diferentes, una perspectiva visual y gráfica, y otra de código fácil de leer.

# II - OBJETIVOS DEL PROYECTO

Los objetivos principales del proyecto son los siguientes:

- Ayudar a iniciarse en la programación.
- Crear un programa que lea un xml creado con Blockly en el que haya descritos bloques secuenciales y alternativos y gestión de funciones y genere un diagrama de Chapin de ese módulo y una función en C_fácil que lo use para lo que hay que extender el lenguaje C_fácil existente.

También existen los siguientes objetivos funcionales y técnicos:

- Utilizar Blockly tanto en modo local como en modo remoto con conexión a Internet.
- Utilizar la aplicación Blockly y el conversor en diferentes sistemas operativos, como son los sistemas Ubuntu y Windows.
- Desde un sistema Ubuntu, poder convertir el código Blockly en C_fácil, compilarlo y ejecutarlo.

# III - Conceptos teóricos

Con el objetivo de comprender el fondo del proyecto, se van a explicar los conceptos más relevantes.

## 1. Herramientas de representación de algoritmos

Para poder realizar un programa es conveniente la previa definición del algoritmo.

Un algoritmo se define como “una secuencia de pasos a seguir, ordenados, no ambiguos y finitos que resuelven un problema.” [1]

Los algoritmos deben tener las siguientes características[2]:

- Deben ser precisos.
- Deben estar bien definidos.
- Deben tener un fin, formado por secuencias finitas de operaciones y terminar en un tiempo finito.
- Pueden tener cero o más elementos de entrada.
- Deben producir un resultado.

Entre dos algoritmos que produzcan el mismo resultado, es más preferible el más corto.

Para poder ejecutar un algoritmo desde un ordenador, debemos proporcionárselo en forma de programa, cada instrucción del programa se corresponde con un paso del algoritmo, y el programa debe especificar la secuencia de operaciones y su orden de ejecución.

Las etapas de creación de los programas son las siguientes[3]:

1. Análisis de problema, definición y delimitación.
2. Diseño y desarrollo del algoritmo.
3. Codificación en el lenguaje deseado.
4. Compilación e interpretación del programa.
5. Ejecución
6. Verificación y depuración.
7. Evaluación de resultados.

## 1.1. Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación. No existe una sintaxis estándar para el pseudocódigo, sino que se trata de una mezcla de palabras en un lenguaje y de símbolos[4].

Las principales características del pseudocódigo son:

- Es conciso.
- Es fácil de aprender y utilizar.
- Es independiente del lenguaje de programación.

Un ejemplo de pseudocódigo para el máximo de dos números es el siguiente:

```
InicioAlgoritmo maximo
        Declarar variables x, y como real
        Escribir "Introduzca dos números para buscar máximo de ellos"
        Leer x, y
        Si x>y  Entonces
                Escribir "El número mayor es " x
        Sino
                Escribir "El número mayor es " y
        Fin si
FinAlgoritmo maximo
```

**Ilustración 1: Ejemplo Pseudocódigo**

## 1.2. Diagrama de Nassi-Schneiderman

El diagrama de Nassi-Scheneiderman o diagrama de Chapin es un diagrama estructurado donde las acciones se escriben en rectángulos o en cajas sucesivas[6].

Sus principales características son:

- Favorece el diseño descendente.
- Favorece la partición de programas en módulos
- Dificultad de modificación.

Un ejemplo del diagrama de Chapin para el máximo de dos números es:

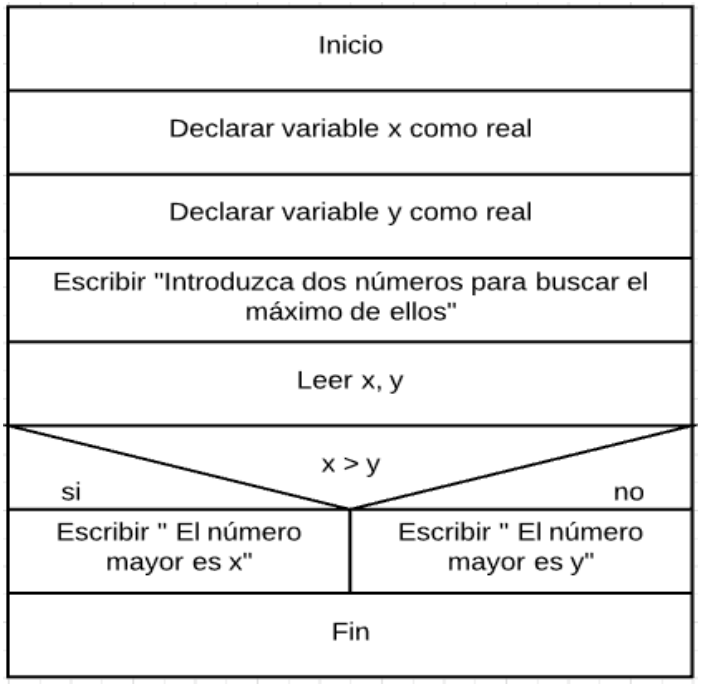


**Ilustración 2: Ejemplo Diagrama Nassi-Schneiderman**

## 1.3. Diagrama de flujo

El diagrama de flujo es una representación gráfica y visual de los algoritmos, compuesto por símbolos unidos por flechas[5].

Sus principales características son:

- Fácil de entender y leer, se lee de arriba a abajo, de izquierda a derecha.
- Los símbolos estandarizados por ANSI e ISO.
- Los símbolos representan acciones o funciones en el programa.
- Las flechas representan el orden de las acciones o funciones.
- Cada símbolo tendrá al menos una flecha de entrada y otra de salida, excepto terminadores y conectores.
- Dificultad de mantenimiento y actualización.

Un ejemplo de diagrama de flujo para el máximo de dos números es:

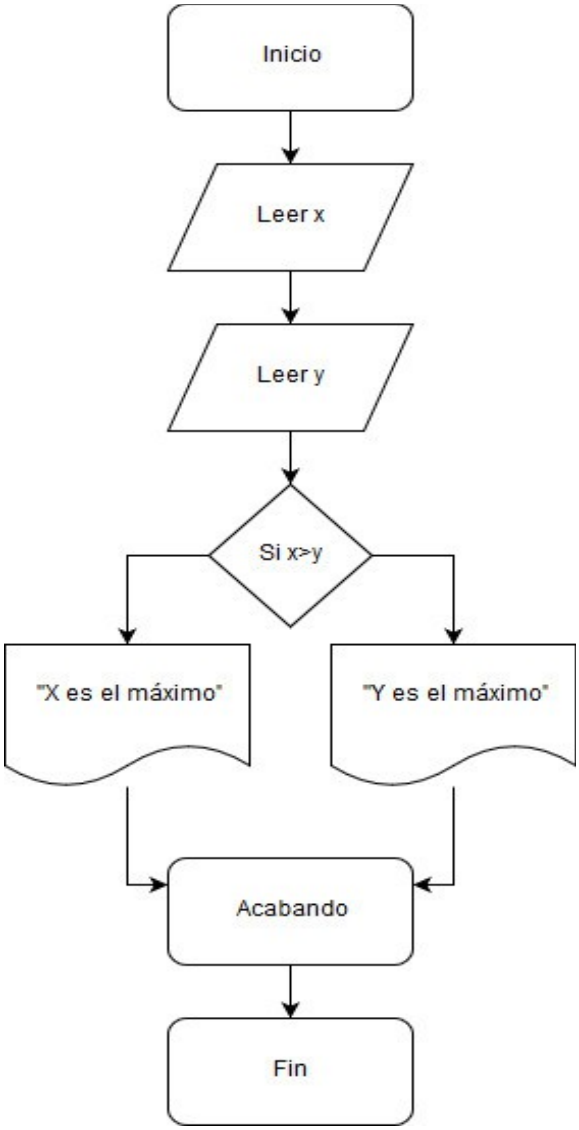


**Ilustración 3: Ejemplo Diagrama de flujo**

## 1.4. Blockly

Blockly es la herramienta que se utiliza en el proyecto para representar algoritmos mediante la unión de bloques formando un programa.[7]

Un ejemplo en Blockly del máximo de dos números es:

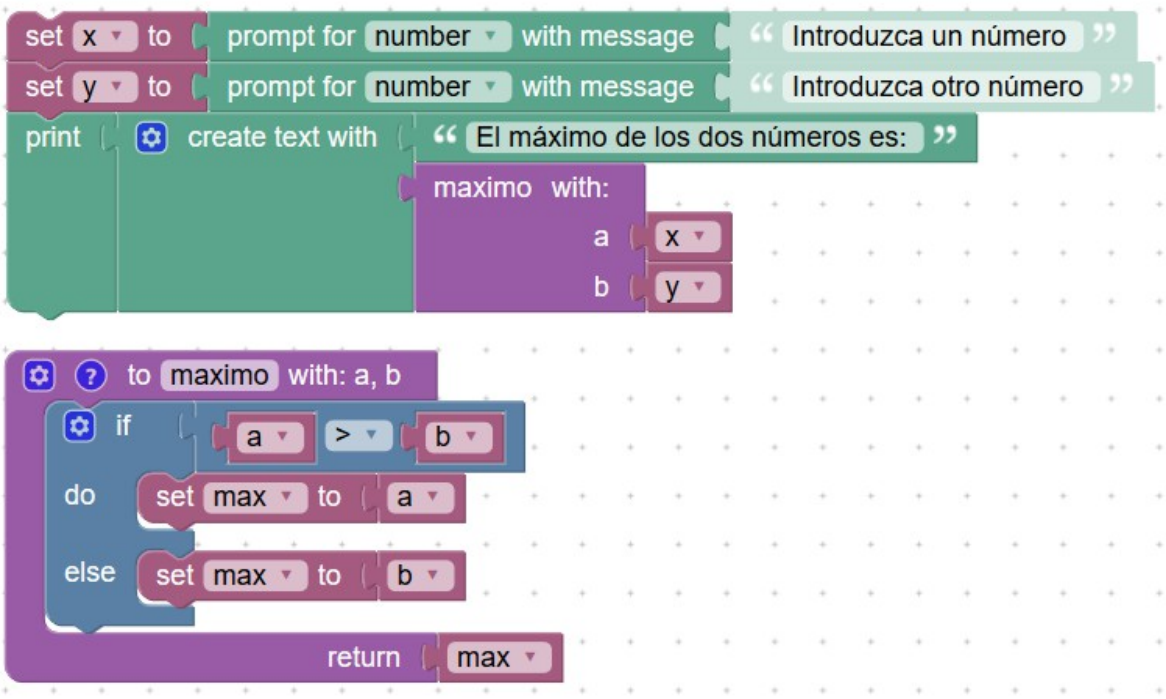


**Ilustración 4: Ejemplo Blockly**

# IV - Técnicas y herramientas

## 1. Metodología usada

La metodología que se ha decidido utilizar para el desarrollo del proyecto ha sido una metodología ágil de trabajo. En concreto se ha decidido utilizar una metodología Scrum[8] porque:

- Esta metodología fue estudiada en la asignatura de "Gestión de proyectos" y se hicieron prácticas simulando el desarrollo de un proyecto con ella.
- Se trata de una metodología que divide el desarrollo del proyecto en sprints, que ayuda a la planificación del desarrollo en un tiempo, de manera iterativa e incremental.
- A lo largo del desarrollo se van planteando unas tareas que hay que ir resolviendo para cada sprint.

## 2. Herramientas software

### 2.1. GitHub

GitHub[9] es un repositorio web que permite el almacenamiento de proyectos.

Se ha decidido utilizar esta herramienta porque:

- Se conocía su uso, porque ha sido utilizada en las prácticas de la asignatura "Gestión de proyectos"
- Su versión gratuita permite almacenar el código de un proyecto de forma pública.
- Facilidad de uso del software de sincronización de los archivos locales con el repositorio web.

#### 2.1.A. Alternativas

Otras alternativas de repositorio web para el almacenamiento de proyectos son las herramientas Bitbucket[10] y GitLab[11], pero se ha descartado su uso debido a que permiten el almacenamiento de proyectos gratuitamente de manera privada y se tenía más conocimientos de la herramienta GitHub utilizada anteriormente.

### 2.2. Javacc

Javacc[12] es un analizador léxico y sintáctico basado en Java, que ha sido utilizado para realizar el conversor de código.

Se ha decidido utilizar Javacc porque:

- Se conocía su uso porque ha sido utilizado en las prácticas de la asignatura "Procesadores del lenguaje".

- Su programación es sencilla e intuitiva.
- Requiere una sola herramienta para el análisis completo.

### *2.2.A. Alternativas*

Otras alternativas estudiadas de analizadores léxicos son Flex[13] y Lex[14], y de analizadores sintácticos con Bison[15] y Yacc[16].

El uso de estas herramientas para un análisis completo requiere el uso de ambos analizadores, bien el conjunto Flex/Bison o Lex/Yacc. Además de esto se han descartado porque aunque Flex y Bison se utilizaron en las prácticas de la asignatura "Procesadores del lenguaje", se utilizaron por separado, por lo que no se han utilizado anteriormente los conjuntos de herramientas Flex/Bison ni Lex/Yacc.

## 2.3. Sublime Text

Sublime Text[17] es un editor de texto que se ha utilizado para la programación del conversor.

Se ha decidido utilizar esta herramienta porque:

- Permite la descarga y utilización gratuita.
- Fácil de utilizar y con numerosas ventajas a la hora de trabajar con código.
- Se ha utilizado en diversas ocasiones a lo largo del grado.

### *2.3.A. Alternativas*

Otras alternativas estudiadas de editores de texto son Vim[18] y Notepad++[19], que aunque se han utilizado a lo largo del grado no ofrecen tantas ventajas como Sublime Text.

El editor Vim se ha utilizado en el sistema Ubuntu durante el uso de la aplicación, debido a que ha sido el editor de texto utilizado en durante el grado en el sistema Ubuntu.

## 2.4. OpenOffice Writer

Writer[20] es un procesador de texto del paquete de ofimática OpenOffice.

Se ha decidido utilizar esta herramienta porque:

- Existe un modelo de plantilla del proyecto adaptado en el formato de Writer.
- Se aprendió a utilizar esta herramienta en la asignatura "Informática básica" y se ha continuado utilizándola en diferentes asignaturas.
- Su descarga es gratuita.

### *2.4.A. Alternativas*

Otras alternativas estudiadas son LaTex[21] y Word[22]. Estas alternativas se han descartado porque, aunque de LaTex existe un modelo de plantilla del proyecto, se trata de un editor de texto basado en la inclusión de etiquetas de texto que definen sus características, no se ha utilizado en ninguna asignatura y requiere de un aprendizaje para su utilización. Word se ha descartado porque no existe un modelo de la plantilla del proyecto adaptado.

## 2.5. Make

Make[23] es un gestor de tareas que permite automatizar y agilizar la tarea de generar y compilar código fuente a ejecutable, y luego ejecutarlo.

Para ello se crea el archivo make (makefile), archivo de texto que contiene la sintaxis utilizada por make para la gestión de la compilación de los archivos de código.

## 2.6. Indent

Indent[24] es una herramienta que permite automatizar el nivel de esquema de los archivos generados para que la presentación de los archivos sea siempre común.

# V - Aspectos relevantes del desarrollo del proyecto

En este apartado se va a recoger la experiencia del desarrollo del proyecto, describiendo sus aspectos relevantes.

# VI - Inicio del proyecto

Comprobando la lista de proyectos propuestos se decidió elegir este proyecto debido a que era un proyecto que iba a tener utilidad, que no sólo iba a ser un desarrollo de proyecto que después quedaría olvidado en una estantería.

El proyecto que se iba a desarrollar iba a ayudar a los alumnos que acababan de acceder al grado a entender y a aprender a programar, por lo que el desarrollo del proyecto se ha planteado para el uso de la aplicación por parte de los alumnos.

## *1. Metodología*

Desde el primer momento se planteó utilizar una metodología ágil de tipo Scrum, pero por motivos personales no se pudo seguir la planificación temporal como se muestra en los anexos.

## *2. Desarrollo*

Como se muestra en los anexos, al principio del desarrollo se decidió desarrollar la aplicación con diferentes herramientas hasta la elección de la herramienta final.

Una vez elegida esta herramienta, se empezó con el desarrollo de la aplicación.

Debido a que este proyecto está especialmente dirigido a alumnos del grado, se decidió crear una página web donde publicar el manual de usuario y varios ejemplo de uso para que pueda ser utilizado por los alumnos en cualquier momento.

## *3. Final proyecto*

Esta etapa del proyecto fue la más dura debido a que aparecieron diversos problemas en la ejecución del proyecto, que previamente no habían aparecido, y que hubo que solucionar.

# VII - Trabajos relacionados

## 1. BlocklyToChapinToCFacil

Existe un desarrollo anterior de este proyecto con el mismo nombre, “BlocklyToChapinToCFacil”, para la asignatura de “Trabajo Fin de Grado” del grado “Ingeniería Informática” de la universidad de Burgos, que fue presentado el día 19/03/2015.

Este proyecto se encuentra como recurso en el departamento de informática de la universidad.

La principal diferencia entre los dos proyectos, es que en el primero se convertía el código de Blockly a C_facil con todas las variables como variables globales, mientras que en el actual proyecto los variables declaradas en el código convertido a C_facil son variables locales.

## 2. Herramientas generadoras de código

Existen varias herramientas que generan código desde una perspectiva visual. Un ejemplo de herramientas son:

- Generador código CSS[25]

  La página “CSS Layout Generator” permite crear código CSS para el diseño web.

- Generador código formulario[26]

  La página pForm es un generador gratuito de formularios HTML

# VIII - Conclusiones y líneas de trabajo futuras

## 1. Conclusiones

Durante el desarrollo del proyecto se ha llegado a las siguientes conclusiones:

- Se ha decidido no desarrollar el conversor al diagrama de Chapin debido a que habría que mostrarlo en otra ventana aparte y dificultaría el uso seguido de la aplicación.
- Los objetivos del proyecto se han cumplido satisfactoriamente, se ha logrado desarrollar un conversor para los módulos de la cabecera c_facil.h
- Se han aplicado conocimientos adquiridos a lo largo del grado.

## 2. Lineas de trabajo futuras

Como líneas de trabajo futuras se puede plantear:

- Que el conversor pueda utilizar más bloques de Blockly, pero para ello primero habría que ampliar los módulos de la cabecera c_facil.h e implementar nuevas funciones.
- Crear, desde Blockly en modo local, una nueva pestaña en la aplicación llamada C_facil e implementar el conversor en la aplicación.
- Añadir el diagrama de Chapin y el diagrama de estructura a el conversor de manera que no entorpezca el uso de la herramienta.

# IX - REFERENCIAS

## Bibliografía

[1] Alberto Prieto Espinosa, Beatriz Prieto Campos: «CONCEPTOS DE INFORMÁTICA»
[2] : «Herramientas representación algoritmos»,
[3] : «http://ing.unne.edu.ar/pub/informatica/Alg_diag.pdf»,
[4] :
«http://formacion.educalab.es/pluginfile.php/43820/mod_imscp/content/8/representacin_de_un_algoritmo.html»,
[6] : «https://es.wikipedia.org/wiki/Diagrama_Nassi-Shneiderman»,
[5] : «http://enriquebarrueto0.tripod.com/algoritmos/sesion03algoritmos.htm»,
[7] : «https://www.genbetadev.com/herramientas/google-blockly-un-lenguaje-visual-para-aprender-a-programar»,
[8] D. Y Soportado Por, K. Schwaber, and J. Sutherland: «La Guía de Scrum La GuíaDefinitiva de Scrum: Las Reglas del Juego»
[9] : «https://github.com/»
[10] : «https://bitbucket.org/»
[11] : «https://about.gitlab.com/»
[12] : «https://javacc.org/»
[13] : «http://es.tldp.org/Manuales-LuCAS/FLEX/flex-es-2.5.html»
[14] : «http://www.lcc.uma.es/~galvez/ftp/tci/TutorialLex.pdf»
[15] : «https://es.wikipedia.org/wiki/GNU_Bison»
[16] : «https://es.wikipedia.org/wiki/Yacc»
[17] : «https://www.sublimetext.com/»
[18] : «https://sourceforge.net/error-404.html»
[19] : «https://notepad-plus-plus.org/download/v7.5.4.html»
[20] : «https://www.openoffice.org/es/producto/writer.html»
[21] : «https://es.wikipedia.org/wiki/LaTeX»
[22] : «https://products.office.com/es-es/word»
[23] : «https://www.gnu.org/software/make/manual/make.html»
[24] : «https://www.gnu.org/software/indent/»
[25] : «http://www.cssportal.com/»
[26] : «http://www.phpform.org/»

Impreso en Burgos el miércoles, 14 de febrero de 2018