DISEÑO DE UN MANUAL DE PROGRAMACIÓN DE APLICACIONES ORIENTADAS A LA ENSEÑANZA MATEMÁTICA EN LENGUAJE LIVECODE

BUENAVENTURA SANABRIA PAREDES

UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA

FACULTAD SECCIONAL DUITAMA

ESCUELA DE MATEMÁTICAS Y ESTADÍSTICA

DUITAMA

2016

DISEÑO DE UN MANUAL DE PROGRAMACIÓN DE APLICACIONES ORIENTADAS A LA ENSEÑANZA MATEMÁTICA EN LENGUAJE LIVECODE

Autor: Buenaventura Sanabria Paredes

Proyecto de trabajo de grado tipo Monografía para optar al título de
LICENCIADO EN MATEMÁTICAS Y ESTADÍSTICA
Director: Mg. Jaime Alberto Reyes Triana

UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA
FACULTAD SECCIONAL DUITAMA
ESCUELA DE MATEMÁTICAS Y ESTADÍSTICA
DUITAMA
2016

NOTA DE ACEPTACION

_____

_____

_____

_____

_____

_____

_____

Firma del presidente del jurado

_____

Firma del jurado

_____

Firma del jurado

Ciudad y fecha (día, mes, año)

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

INTRODUCCIÓN

Durante los últimos años, la Educación Matemática ha hecho considerables esfuerzos por mejorar los desempeños docentes en el aula, esto contempla la implementación de nuevas estrategias y utilización de tecnologías, hecho que va de la mano con políticas educativas tanto internacionales como nacionales, quienes proponen diferentes programas para mejorar estos procesos; con este fin, el gobierno nacional dentro de su política computadores y tablet para educar ha dotado a las instituciones de equipos computacionales y/o móviles con el propósito de aumentar la cobertura de las tecnologías de la información y de la comunicación (TIC) en el país.

"El uso de las TIC en la educación depende de múltiples factores (infraestructuras, formación, actitudes, apoyo del equipo directivo, etc.), entre los cuales el más relevante es el interés y la formación por parte del profesorado, tanto a nivel instrumental como pedagógico" (Belloch, 2012, pág. 7). El Ministerio de Educación Nacional (MEN) para orientar y potenciar los procesos formativos con respecto al uso de las TIC ha planteado estrategias entre las que se puede destacar el desarrollo profesional docente, para incentivar el mejoramiento de las prácticas educativas que hacen uso de las TIC y fortalecer las competencias de los docentes y el fomento a la investigación, para fortalecer grupos de investigación y dinamizar el desarrollo de proyectos de investigación para el sistema educativo, con énfasis en innovación educativa con uso de TIC (MEN, 2013).

Con respecto a la implementación instrumental, es decir, de equipos computacionales y/o móviles y herramientas tecnológicas en educación, se observa que en su gran mayoría se está desaprovechando el uso de las mismas, por desconocimiento tanto del manejo de estas herramientas como del desarrollo de aplicaciones que puedan contribuir tanto en el aula como fuera de ella.

Para suplir de alguna manera esta dificultad, se propone el diseño de un manual que facilite la programación en lenguaje LiveCode, de aplicaciones orientadas a la enseñanza y a promover el desarrollo de habilidades en el área matemática, para innovar los procesos de enseñanza y aprendizaje de las mismas como estrategia didáctica mediante la utilización de dispositivos móviles y computadores, que facilite a los educadores su desempeño en el aula.

Esto articula con los propósitos de la licenciatura en cuanto a desarrollar competencias investigativas a la luz de teorías y modelos investigativos que permitan avanzar en el conocimiento matemático, estadístico y didáctico; y generar una cultura investigativa y de innovación para coadyuvar al desarrollo educativo.

Éste manual estará limitado a aplicaciones para contenidos matemáticos hasta el nivel de educación básica, ya que programar para contenidos matemáticos de orden superior, necesita de una ejercitación en programación que se ira adquiriendo con el desarrollo de aplicaciones como las que se proponen, e ir desarrollando destrezas para lanzarse a un nivel mayor como meta, pero teniendo siempre presente que el propósito final es que se pueda lograr esta última meta

JUSTIFICACIÓN

Gracias a la utilización continua y eficaz de las TIC en procesos educativos, los estudiantes tienen la oportunidad de adquirir capacidades importantes en el uso de éstas. El docente es la persona que desempeña el papel más importante en la tarea de ayudar a los estudiantes a adquirir esas capacidades, además es el responsable de diseñar tanto oportunidades de aprendizaje como el entorno propicio en el aula que facilite el uso de las TIC por parte de los estudiantes para aprender y comunicar. Por esto, es fundamental que todos los docentes estén preparados para ofrecer esas oportunidades a sus estudiantes (UNESCO, 2008, pág. 2). Dado que las instituciones están siendo dotadas de equipos e infraestructura para el mejoramiento continuo de las mismas.

Teniendo en cuenta lo anterior, en nuestro país el acceso a las nuevas tecnologías en los procesos educativos se ha incrementado gracias a las políticas que ha implementado el gobierno nacional para aumentar la cobertura de las TIC en todo el territorio, con programas como Educa Digital, Vive Digital, En TIC Confío, Redvolución, Computadores y Tabletas para Educar, entre otros, pero el hacer un uso efectivo y eficaz de estas herramientas ha sido la gran dificultad, debido al desconocimiento del manejo de las mismas, como la incapacidad de los docentes para desarrollar aplicaciones que puedan contribuir tanto en el aula como fuera de ella.

La consolidación de un sistema educativo de calidad requiere el desarrollo de nuevas competencias por parte de los protagonistas de los complejos procesos educativos y la evolución de las prácticas pedagógicas hacia la innovación, pues solo así convertiremos a las TIC en herramientas que favorezcan el aprendizaje y el conocimiento (MEN, 2013). Dada la importancia del uso de las nuevas tecnologías en los procesos educativos, del desarrollo de competencias y la implementación de estas tanto en las instituciones educativas como en los hogares y la dificultad de hacer uso adecuado de estas herramientas (por desconocimiento o por falta de manejo de las mismas) en los procesos de enseñanza-aprendizaje especialmente por parte de los educadores.

Este trabajo surge de la necesidad de diseñar un manual que facilite la programación de aplicaciones en lenguaje LiveCode, orientadas a la enseñanza y promover el desarrollo de habilidades en el área matemática, dirigido especialmente a los educadores para que mejoren sus experiencias de aula, pero también a todo aquel que desee convertirse en desarrollador de aplicaciones, aprovechando la facilidad del lenguaje comparado con los demás, que son

algorítmicos y difíciles de aprender, éste por su parte es dirigido a objetos, es decir, a un objeto se le asigna un comando y realiza una acción, además las ventajas de aplicabilidad en distintos dispositivos y la rapidez en la elaboración de las aplicaciones, que pueden ser utilizadas en dispositivos móviles como celulares, tabletas, IPad, IPhone y computadores portátiles y de escritorio sin importar su sistema operativo, porque esta herramienta funciona sin problemas en Windows, LINUX, MacOSX, UNIX (Solaris y BSDs) IOS y Android.

1. GENERALIDADES DEL PROYECTO

1.1. PROBLEMÁTICA

El reto de la educación actual será adaptarse al contexto y la tecnología, ya que los avances científicos hacen necesario pensar en una nueva manera de aprehender, para lo cual es indispensable hablar de innovación educativa, de manera que se puedan introducir novedades dentro de los objetivos institucionales y se modifiquen aún los modelos de enseñanza, esto para alcanzar al mundo desarrollado, puesto que la ciencia, la tecnología y las comunicaciones van más rápido que los sistemas educativos.

"Innovar significa pensar críticamente, abordar los problemas desde diferentes perspectivas, crear contextos participativos, disponer espacios diversos para las relaciones docente-estudiante y mejorar las condiciones de los ambientes de aprendizaje" (MEN, 2013, pág. 16). Para poder lograr esto, los profesores que deseen guiar los aprendizajes de sus alumnos, fomentando la interacción y el aprendizaje colaborativo, siguiendo los postulados del constructivismo social de Vygotsky, quien enfatiza la influencia de los contextos sociales y culturales en la apropiación del conocimiento y pone gran énfasis en el rol activo del maestro, mientras que las actividades mentales de los estudiantes se desarrollan "naturalmente" a través de varias rutas de descubrimientos: la construcción de significados, los instrumentos para el desarrollo cognitivo y la zona de desarrollo próximo (ZDP), o el aprendizaje por descubrimiento de Bruner (Valencia, 2015), que promueve que el alumno (aprendiente) adquiera los conocimientos por sí mismo, que los contenidos no se deben mostrar en su forma final, sino que han de ser descubiertos progresivamente por los alumnos y alumnas, estos docentes tienen en las TIC un fuerte aliado, fundamentalmente en los diferentes recursos y servicios que ofrece Internet (Belloch, 2012).

El desarrollo acelerado de la sociedad de la información está suponiendo retos, impensables hace unos años, para la educación y el aprendizaje. Tal vez lo más relevante sea que nos encontramos con una nueva generación de aprendices que no ha tenido que acceder a las nuevas tecnologías, sino que ha nacido con ellas y que se enfrenta al conocimiento desde postulados diferentes a los del pasado. Ello supone un desafío enorme para los profesores, la mayoría de ellos inmigrantes digitales, para las escuelas, para los responsables educativos y para los gestores

de las políticas públicas relacionadas con la innovación, la tecnología, la ciencia y la educación (Marchesi, 2009, pág. 120).

Con la llegada del internet, la interactividad y las comunicaciones en tiempo real se hace necesaria la implementación de pedagogías que interactúen con la globalización y que permita que los estudiantes descubran el conocimiento por medio de procesos, competencias y habilidades, para lo cual se necesita que quienes participan en la educación cambien los criterios de formación y enseñanza para adentrarse en el mundo fascinante de la creatividad e innovación de los niños y adolescentes (Cajiao, 2015, pág. 10).

Los docentes que pretendan desarrollar dichas competencias deben a su vez preocuparse por ser integradores de contenidos didácticos, pero con la visión de ser en mayor grado productor de contenido más que consumidor de los mismos, encontrando en su desarrollo oportunidades de complementar su saber profesional de manera transversal con contenido multidisciplinar.

En este momento la mayoría de educadores es inmigrante digital, debido a ello se encuentran dificultades al hacer uso adecuado de herramientas tecnológicas en los procesos de enseñanza-aprendizaje, pues presupone un reto familiarizarse con ellas y manipularlas al punto de no usarlas por temor a dañarlas, mucho más será la dificultad en la generación de contenidos y software como aplicaciones que puedan ejecutarse en cualquier dispositivo y dado que son necesarias en el entorno educativo actual, ¿El diseño de un manual de programación en lenguaje LiveCode facilitará a docentes la creación de aplicaciones multimedia para innovar los procesos de enseñanza y aprendizaje de las matemáticas como estrategia didáctica, para convertirlos en productores de contenido más que consumidores del mismo?.

1.2. OBJETIVOS

1.2.1. Objetivo General

Diseñar y elaborar un manual que describa las instrucciones básicas para programar aplicaciones didácticas orientadas a la enseñanza y a fortalecer habilidades matemáticas en educación básica, en lenguaje LiveCode, que pueda ejecutarse en dispositivos móviles como celulares, tabletas, iPhone, iPad y computadores portátiles y de escritorio sin importar su sistema operativo.

1.2.2. Objetivos Específicos

- Reconocer las implicaciones del uso de las TIC en la enseñanza.
- Estudiar y reconocer el software LiveCode y su lenguaje de programación.
- Diseñar y elaborar un manual básico de programación en lenguaje LiveCode.
- Elaborar una guía piloto.
- Elaborar un documento que contenga el estudio realizado.

1.3. MARCO METODOLÓGICO

El proyecto se desarrolló siguiendo la siguiente secuencia.

Investigación.

Para conocer los elementos y los conceptos de didáctica matemática que se usan en la realización de aplicaciones multimedia y su utilización en el aula.

Identificación de las competencias docentes requeridas por el MEN, para aportar en el desarrollo de las mismas.

Reconocimiento del lenguaje de programación con el que se trabajó en la realización de las aplicaciones didácticas.

Proyección.

Se tuvo presente la estructura del lenguaje y los comandos básicos requeridos para realizar aplicaciones y el planteamiento de una guía para realizar la validación del manual.

Ejecución.

Realización de aplicaciones como ejemplo y se recopiló en el informe final.

2. MARCO TEÓRICO

Para poner en contexto la elaboración del manual de programación, se debe argumentar algunos conceptos inmersos en la elaboración de instrumentos didácticos como aplicaciones multimedia, entre ellos el de innovación educativa y su empleo en el desarrollo docente, la incorporación de las TIC en la enseñanza y especialmente en el área matemática, comprendiendo las diferencias entre los recursos y materiales didácticos y a su vez como todo esto conlleva a desarrollar competencias, explorando las exigidas por el MEN, para saber en cuales profundiza quien desarrolla contenido digital y por último se trata en esta sección el programa LiveCode, que es el lenguaje de programación propuesto para elaborar las aplicaciones didácticas.

2.1. INNOVACIÓN EN EDUCACIÓN

Considérese el caso en que la innovación se mezcla con la educación y se observara que es en los salones donde menos se permite desarrollar un espíritu innovador, donde el protagonista es el profesor y los estudiantes repiten exámenes pero no analizan y mucho menos cuestionan, porque los errores se castigan en lugar de utilizarlos para potenciar la autonomía y habilidades de los estudiantes (Galviz, 2015, pág. 12).

Como el componente de innovación debe estar integrado de manera trasversal a toda actividad escolar e invitar a que el aprendizaje sea vivencial, donde se promueva el libre pensamiento, se hace necesario el esfuerzo docente por introducirse en el mundo de los programas educativos computacionales teniendo presente que:

[…] los docentes necesitan estar preparados para empoderar a los estudiantes con las ventajas que les aportan las TIC. Escuelas y aulas –ya sean presenciales o virtuales– deben contar con docentes que posean las competencias y los recursos necesarios en materia de TIC y que puedan enseñar de manera eficaz las asignaturas exigidas, integrando al mismo tiempo en su enseñanza conceptos y habilidades de estas. Las simulaciones interactivas, los recursos educativos digitales y abiertos (REA), los instrumentos sofisticados de recolección y análisis de datos son algunos de los muchos recursos que permiten a los docentes ofrecer a sus estudiantes posibilidades, antes inimaginables, para asimilar conceptos (UNESCO, 2008, pág. 2).

Todo ello con la intención de mejorar los procesos didácticos y curriculares que transformen las experiencias en el aula y que pueda desarrollar habilidades y competencias en los estudiantes, teniendo en cuenta estas como:

[…] el complejo proceso de formación y desarrollo de un ser *humano*, en permanente *actividad* y con *capacidades* para acceder a nueva información y apropiarse de nuevo *conocimiento*; para enfrentar con su *pensamiento* la incertidumbre y la complejidad de los problemas generados por la nueva sociedad del conocimiento, y para, desde *el trabajo, el lenguaje y el pensamiento*, contribuir a la transformación de la sociedad en la que históricamente se sitúa. (García Quiroga, Coronado, & Montealegre Quintana, 2011)(Énfasis en el original).

Desde esta perspectiva, la educación ejerce un papel importante dentro de la así llamada sociedad de los nativos digitales y es el fortalecimiento de la búsqueda del conocimiento en un mundo cada vez más globalizado donde la fortaleza sea el poder aprender y desaprender con facilidad para lograr competir. Los docentes también deben apropiarse de dicha fortaleza para trasformar el conocimiento propio y a su vez el de los estudiantes que participan de sus experiencias, de manera que con ese conocimiento se pueda innovar.

[…]Un ambiente innovador de aprendizaje es aquel en el cual los estudiantes desarrollan pensamiento crítico, autónomo y creativo mediante el trabajo en equipo y por supuesto, con la utilización de las nuevas tecnologías,… La creación de un ambiente innovador requiere la presencia de una serie de agentes (docentes, directivos, personal administrativo y de apoyo) que contribuyan a mantener un clima que esté articulado a través de lo académico y el entorno sociocultural. El docente es el encargado de construir ambientes innovadores seleccionando las estrategias y las TIC adecuadas para que entre los estudiantes se establezcan relaciones cooperativas, que se caracterizan por lograr que un miembro de la relación logre sus objetivos de aprendizaje, siempre y cuando los otros alcancen los suyos y entre todos construyen conocimiento aprendiendo unos de otros (MEN, 2013, pág. 19).

2.2. LAS TIC EN PEDAGOGÍA Y DIDÁCTICA DE LAS MATEMÁTICAS

Entender cómo se articulan las TIC con la enseñanza, especialmente con la didáctica y la tecnología, guía en la elaboración de un buen contenido digital y como integrarlo en el aula.

Las Tecnologías de la Información y Comunicación (TIC) han ido integrándose en los centros educativos de forma paulatina. A las primeras reflexiones teóricas que los profesionales de la educación realizaban sobre la adecuación o no de estas tecnologías para el aprendizaje, se ha continuado con el análisis sobre el uso de estas tecnologías y su vinculación a las teorías de aprendizaje, junto a propuestas metodológicas para su implementación (Belloch, 2012, pág. 6).

De nada vale al maestro/a de primaria o al profesor/a de secundaria saber muchas matemáticas si no sabe enseñarlas a sus alumnos/as. Tampoco son útiles las teorías didácticas o el conocimiento de herramientas didácticas si no conoce primero quien tiene que aprender, cuáles son sus intereses por el conocimiento, en qué condiciones puede estudiar en casa, cuál es su nivel de atención, en qué entorno cultural y social se desenvuelve o, en el caso que nos ocupa, las destrezas que pueda tener en el uso de las herramientas TIC. Las TIC pueden llegar a jugar un papel muy importante en el proceso de enseñanza y aprendizaje de las matemáticas, si se utilizan correctamente. Es más, si su uso no es el adecuado, pueden llegar a trazar un camino tortuoso pasando de ser una potente herramienta a una barrera que impida el proceso (Real Perez, 2012, pág. 3).

Se hace necesario reflexionar sobre el proceso de enseñanza y en especial en cada uno de los actores y su entorno, para poder proponer un modelo que no solo sea funcional sino que a su vez permita un uso crítico, tanto del modelo que se aplica como del contenido de enseñanza – aprendizaje.

La incorporación de las TIC a la educación exige pensar previamente cuáles son los objetivos y los retos de la educación y determinar posteriormente de qué manera y en qué condiciones la presencia de las TIC en las escuelas contribuye a ellos. Lo primero y más importante es determinar el sentido de las TIC en la educación y cuál es el modelo pedagógico con el que se puede contribuir de forma más directa a mejorar la calidad y la equidad educativa. Pero, como apunta Juan Carlos Tedesco, “estas promesas de las TIC en educación están lejos de ser realidad. No se trata de negar la potencialidad democratizadora o innovadora de las nuevas tecnologías sino de enfatizar que el ejercicio de esa potencialidad no depende de las tecnologías mismas sino de los modelos sociales y pedagógicos en los cuales se utilice” (Tedesco, 2005) citado por (Marchesi, 2009, pág. 122).

Además se debe tener presente que las TIC no son el objetivo, sino un medio. En muchas ocasiones se puede llegar al error de acabar enseñándole a un alumno o alumna el manejo de determinadas aplicaciones en lugar de el o los contenidos matemáticos que nos habíamos propuesto inicialmente, (…). Aun así,

debemos seguir insistiendo en que solamente es un recurso con el que podemos contar en el aula. Un recurso que forma parte del entorno en el que se mueve nuestro alumnado y que puede facilitar ese proceso de aprendizaje en el área de matemáticas. Un recurso al que no debemos temer, sino todo lo contrario, ya que nos puede facilitar mucho nuestra tarea (Real Perez, 2012, págs. 4-7).

Para ello es necesario tener presente cómo se logra la adquisición de conocimiento y para nuestro caso conocimiento matemático.

El conocimiento matemático es producto de las estructuraciones de la conciencia y de la experiencia del sujeto, entonces, dichas estructuraciones están dadas en la interacción dialéctica del sujeto con el objeto de conocimiento y con su entorno,(…) de cómo el sujeto logra interiorizar las acciones efectuadas sobre los objetos materiales o ideales, (…) esto desemboca en la necesidad de comunicar, discutir, demostrar, verificar y, por tanto, de simbolizar, proceso que entraña un progreso lógico-formal innegable hacia el desarrollo del pensamiento en la actividad matemática, El pensamiento matemático es, por tanto, un proceso mental sujeto a la necesidad de socializar, comunicar, que en matemáticas requiere de sistemas semióticos y se condiciona por la elección de un mediador simbólico o registros de representación (García Quiroga, Coronado, & Montealegre Quintana, 2011, pág. 170).

Estos mediadores pueden ser tomados como aplicaciones que se utilicen como recursos para facilitar el proceso de aprendizaje y por tanto la Pedagogía, al igual que otras disciplinas científicas, encuentra en las TIC nuevas actividades profesionales:

- Análisis y evaluación de los recursos tecnológicos y su uso educativo.
- Integración de los medios de comunicación para lograr el aprendizaje.
- Diseño de estrategias educativas para favorecer la integración de recursos tecnológicos en diferentes ambientes de aprendizaje.
- Diseño de materiales multimedia para favorecer el proceso de enseñanza/aprendizaje.
- Desarrollo de materiales digitales.
- Diseño y evaluación de software educativo.
- Diseño, desarrollo y evaluación de modelos de educación presencial y a distancia, entre otras (Belloch, 2012, pág. 8).

Dentro de estas actividades de diseño y desarrollo de materiales digitales multimedia se pretende avanzar por medio de LiveCode, el cual es un lenguaje de programación que permite a los docentes ingresar en el mundo de las aplicaciones móviles con muy poca exigencia de programación pero con muy buenos resultados, porque permite realizar aplicaciones en corto tiempo y de acuerdo al tema establecer los parámetros de la misma, con el propósito de fortalecer un punto específico de enseñanza; mediante esta herramienta el docente proyecta la aplicación y la realiza pensando en los procesos de pensamiento que quiere fortalecer, diferente de los software y aplicaciones en los que simplemente se obtienen resultados.

2.3. RECURSOS Y MATERIALES DIDÁCTICOS EN MATEMÁTICAS

Saber que el manual permite la elaboración de recursos y materiales y a su vez saber utilizarlos garantizará que en la práctica se tenga control tanto del proceso de enseñanza como en el del aprendizaje.

Teniendo en cuenta que el ejercicio docente cuenta con múltiples ayudas, para enseñar y aprender y que hay diferencias entre ambos conceptos, “argumentando que para aprender hay que “hacer” y los materiales y recursos permiten que el alumno haga” (Flores, Lupiáñez, Berenguer, Marín, & Molina, 2011, pág. 5). Se debe definir lo que se considera un recurso o material didáctico, Alsina, Burgués y Fortuny (1988) afirman que “bajo la palabra <<material>> se agrupan todos aquellos objetos, aparatos o medios de comunicación que pueden ayudar a describir, entender y consolidar conceptos fundamentales en las diversas fases del aprendizaje” (p. 13) citados por (Gallardo, s.f).

Se pueden observar cuatro categorías, “los que pretenden el aprendizaje de conceptos y los que promueven la ejercitación de destrezas de manera lúdica” (Flores *et al.*, 2011,pág. 5). Y a su vez los “materiales didácticos manipulativos (policubos, bloques multibase, Mira,...) y los virtuales o no manipulables (software didáctico, materiales escritos, medios audiovisuales)” (Gallardo, s.f, pág. 2).

En estas categorías, se puede clasificar el desarrollo de software con LiveCode dentro de los virtuales y a su vez dentro de los de ejercitación y destreza, sin dejar de lado que se pueden elaborar también aplicaciones para aprender conceptos, todo depende del interés del docente, quién debe tener presente que “Los materiales y recursos permiten al profesor plantear tareas para que los alumnos utilicen los conceptos matemáticos” (Flores *et al.*, 2011, pág 10).

También se debe tener en cuenta que el uso de los materiales tiene objetivos que en su mayoría de veces se pueden clasificar así:

- *Para favorecer la adquisición de rutinas*. Existe un tipo de material didáctico que está diseñado para cumplir una función muy específica, principalmente de consolidación de conceptos o ejercitación de procedimientos.

- *Para modelizar ideas y conceptos matemáticos.*

- *Para plantear problemas*. Este uso es defendido y fomentado por la actual Reforma Educativa. El tangram, el plegado de papel o los policubos son ejemplos de recursos y materiales didácticos generadores de cuestiones, problemas abiertos y actividades de investigación (Flores et al., 2011, pág 10).

Con materiales multimedia se pude lograr el mismo objetivo y es lo que pretende el uso de las TIC, pues mediante la utilización de recursos didácticos como menciona Flores

[…] el profesor enseña para que el alumno aprenda. Para aprender, el alumno escucha, copia, resuelve, actúa, y finalmente memoriza. Además tiene que ponerle nombre y saber cuándo debe usar lo aprendido, para utilizarlo cuando la situación lo requiera. Si lo emplea para resolver problemas reales, el alumno será competente para emplear lo aprendido. Si sólo las emplea cuando el profesor le pregunta, estará desarrollando aprendizaje meramente escolar (Flores *et al.*, 2011, pág 6).

La utilización de materiales didácticos estará cuestionada y en la mayoría de las veces delimitada por dificultades que pueden surgir al aplicarlas en el aula, desde este punto de vista, Jesús Gallardo define los problemas y dificultades de la siguiente manera:

(a) *El profesor*: La formación científica y didáctica del profesor y sus concepciones sobre la matemática y su aprendizaje influyen notablemente a la hora de decidir la conveniencia de utilizar un determinado material didáctico con los alumnos.

(b) *El alumno*: El interés, la motivación, la disciplina o el nivel de los alumnos son factores que también influyen en la decisión de emplear recursos y materiales didácticos.

(c) *El Centro educativo*: La filosofía o cultura escolar del Centro y la infraestructura que ofrece son dos factores que pueden llegar a plantear dificultades importantes al profesor interesado en utilizar recursos y materiales didácticos en el aula

(d) *El conocimiento matemático a estudiar*, plantea al profesor una serie de cuestiones metodológicas que afectan también a la utilización de los recursos y materiales didácticos (Gallardo, s.f, págs. 4-5).

El docente enfrenta dichas dificultades en la elaboración de los materiales didácticos y debe poder dar solución satisfactoria antes de la aplicación de los mismos, además, tendrá presente como dice Gallardo que "en principio, la utilización de recursos y materiales didácticos no garantiza la comprensión de los sujetos" y que pueden causar adicción tanto a los estudiantes como al docente, "Los materiales didácticos deberían utilizarse exclusivamente para cubrir aquellos objetivos docentes en los que la aportación sea claramente efectiva. Sería un error tratar de ajustar contenidos matemáticos para poder utilizar nuestro material didáctico favorito" (Gallardo, s.f, pág. 7).

Todo lo anterior debe llevar a la enseñanza y aprendizaje de las matemáticas de modo que se suplan las necesidades de los estudiantes, teniendo presente que el aprendizaje matemático es muy complejo, tanto por la racionalidad del aprendiz como por la complejidad del conocimiento matemático. No es solo memorizar una serie de destrezas sino en tener ideas, comprender conceptos para saber en qué ocasiones y con qué problemas se utilizan (Flores *et al.*, 2011, pág 7).

Aun así, no se pueden desconocer las potencialidades de estos objetos al abrir la atractiva posibilidad de experimentar con las matemáticas, al permitir la reflexión y análisis de procedimientos y resultados, proporcionando oportunidades para que los estudiantes muestren sus concepciones erróneas acerca de determinados conceptos matemáticos, ya que desarrolla la motivación y potencia la capacidad creativa de los alumnos (Gallardo, s.f, pág. 6). Esto conlleva a que el docente desarrolle destrezas, habilidades y competencias necesarias para un mejoramiento continuo de su labor.

2.4. COMPETENCIAS DOCENTES EN TIC

Tener presentes las competencias que exige el MEN en los profesionales de la educación permite que se desarrollen habilidades para poder cumplir con las mismas y abordarlas en la elaboración de aplicaciones didácticas.

Las nuevas tecnologías (TIC) exigen que los docentes desempeñen nuevas funciones y también, requieren nuevas pedagogías y nuevos planteamientos en la formación docente. Lograr la integración de las TIC en el aula dependerá de la capacidad de los maestros para estructurar el ambiente de aprendizaje de forma no tradicional, fusionar las TIC con nuevas pedagogías y fomentar clases dinámicas en el plano social, estimulando la interacción cooperativa, el aprendizaje colaborativo y el trabajo en grupo. Esto exige adquirir un conjunto diferente de competencias para manejar la clase. En el futuro, las competencias fundamentales comprenderán la capacidad tanto para desarrollar métodos innovadores de utilización de TIC en el mejoramiento del entorno de aprendizaje, como para estimular la adquisición de nociones básicas en TIC, profundizar el conocimiento y generarlo (UNESCO, 2008, pág. 7).

La formación del docente en cuanto a ser investigativo e innovador hará la diferencia en la adquisición de nuevos retos, que son implementados por instituciones nacionales e internacionales, para que el desempeño docente cumpla con un objetivo globalizado de “preparar estudiantes, ciudadanos y trabajadores capaces de comprender las nuevas tecnologías tanto para apoyar el desarrollo social, como para mejorar la productividad económica” (UNESCO, 2008, pág. 8). El MEN propone guiar el proceso de desarrollo profesional docente para la innovación educativa pertinente con uso de TIC; definiendo cinco competencias a desarrollar, estas son:

2.4.1. Competencia Tecnológica. El propósito es la integración de TIC en la educación para mejorar los procesos de enseñanza y aprendizaje, así como la gestión escolar y se puede definir como la capacidad para seleccionar y utilizar de forma pertinente, responsable y eficiente una variedad de herramientas tecnológicas entendiendo los principios que las rigen, la forma de combinarlas y las licencias que las amparan.

2.4.2. Competencia Comunicativa. Facilitan la conexión entre estudiantes, docentes, investigadores, otros profesionales y miembros de la comunidad, incluso de manera anónima, y también permiten conectarse con datos, recursos, redes y

experiencias de aprendizaje, puede definirse como la capacidad para expresarse, establecer contacto y relacionarse en espacios virtuales y audiovisuales a través de diversos medios y con el manejo de múltiples lenguajes.

2.4.3. Competencia Pedagógica. Es el saber propio de los educadores adquirido a través de su práctica y se pude definir como la capacidad de utilizar las TIC para fortalecer los procesos de enseñanza y aprendizaje, reconociendo alcances y limitaciones de la incorporación de estas tecnologías en la formación integral de los estudiantes y en su propio desarrollo profesional.

2.4.4. Competencia de gestión. Se concentra en modular los factores asociados al proceso educativo, con el fin de imaginar de forma sistemática y sistémica lo que se quiere que suceda (planear); organizar los recursos para que suceda lo que se imagina (hacer); recoger las evidencias para reconocer lo que ha sucedido y, en consecuencia, medir qué tanto se ha logrado lo que se esperaba (evaluar) para finalmente realizar los ajustes necesarios (decidir), se puede definir como la capacidad para utilizar las TIC en la planeación, organización, administración y evaluación de manera efectiva de los procesos educativos; tanto a nivel de prácticas pedagógicas como de desarrollo institucional.

2.4.5. Competencia investigativa. Es la gestión del conocimiento y, en última instancia, la generación de nuevos conocimientos. La investigación puede ser reflexiva al indagar por sus mismas prácticas a través de la observación y el registro sistematizado de la experiencia para autoevaluarse y proponer nuevas estrategias, se define como la capacidad de utilizar las TIC para la transformación del saber y la generación de nuevos conocimientos (MEN, 2013, págs. 31-33).

Además cada competencia se desarrolla en tres momentos específicos que son: exploratorio, donde se hace un acercamiento a los conocimientos que permitirán una mayor elaboración conceptual, de integración, donde se plantea el uso de los conocimientos ya apropiados para la resolución de problemas en contextos diversos y un momento de innovación, donde se da mayor énfasis a los ejercicios de creación; lo que permite ir más allá del conocimiento aprendido e imaginar nuevas posibilidades de acción o explicación (MEN, 2013, pág. 34). Son estos los retos propuestos y en los que cada educador debe poner sus esfuerzos para mejorar el desempeño profesional, especialmente aplicando y reflexionando en los momentos de cada competencia anhelando ser innovador en cada una.

Teniendo en cuenta que debe guiar a los educandos para ser ciudadanos digitales, que a futuro puedan hacer contenidos creativos y que no se conviertan en meros consumidores digitales, sino que poniendo en práctica la creatividad sean productores de contenidos digitales, se plantea el aprendizaje de lenguajes de programación con el propósito de apropiar el uso de la tecnología en el quehacer diario como innovador, al integrar contenidos de las TIC en su desempeño laboral.

2.5. EL SOFTWARE LIVECODE

Es el lenguaje de programación propuesto para elaborar las aplicaciones didácticas, para lo cual debe hacerse un reconocimiento del mismo, empezando desde su historia, hasta realizar aplicaciones con él.

Figura 1. Logo del lenguaje de programación LiveCode

Fuente: Adaptado Hermano_Temblón recuperado de: http://www.hermanotemblon.com/livecode-una-nueva-forma-de-programar/

Livecode es una interesante herramienta de programación relativamente reciente, la cual promete algo hasta ahora inédito: Que un mismo programa hecho con esta herramienta funcione sin problemas en Windows, LINUX, MacOSX, UNIX (Solaris y BSDs) IOS y Android. Algo parecido a lo que hace Java, Python o ".Net" (Vacas, 2015). Pero con la dificultad que estos software no se pueden usar para programar para IPad e IPhone (Apple lo prohíbe).

2.5.1. Historia de LiveCode y Revolution (el lenguaje de programación de Livecode):

Livecode se inspira en un lenguaje de programación mucho más antiguo denominado "HyperTalk". Este lenguaje fue creado en 1987 por un programador Estadounidense llamado Dan Winkler

Figura 2. Ventana del desaparecido lenguaje HYPERTALK

Fuente: Adaptado Hermano_Temblón recuperado de: http://www.hermanotemblon.com/livecode-una-nueva-forma-de-programar/

LiveCode es el último de una larga serie de herramientas de desarrollo de “alto nivel” que pueden tener sus raíces en una herramienta de desarrollo muy influyente y popular, HyperCard, que apareció por primera vez en la década de 1980. Cuando HyperCard apareció en 1987, la mayoría de la gente no sabía qué hacer con él, porque nada parecido había sido creado para un público masivo. Bill Atkinson, el autor de HyperCard, lo calificó como “una herramienta de creación y un organizador de información (...) y una especie de reproductor para obtener información.” Danny Goodman, autor de varios libros “Guía” en HyperCard, lo llamo “entorno de programación autónomo para los no programadores”. Se le ha llamado también un “software Arma Todo”, ya que, es semejante al juego de construcción que tienes para Navidad, que tiene piezas que se pueden poner juntas y armar casi cualquier cosa (una analogía más moderna podría ser Legos, ya que se puede construir un montón de cosas con una colección de piezas estandarizadas) (Brigham Young University, 2005).

Fundamentalmente, lo que HyperCard hizo fue permitir a los no programadores montar las colecciones de diferentes tipos de información relacionada con texto, gráficos, sonidos, etc., y unirlos de diversas maneras. Esta técnica se llama "hiperenlaces" o "hipermedia", y es el modelo en que se basa la World Wide Web (Internet). De hecho, cuando Tim Berners-Lee escribió la propuesta original para la creación de la Web, citó HyperCard como un ejemplo de un sistema para vincular de forma rápida todo tipo de datos relacionados en un único entorno integrado.

La mayoría atribuyen la visión de hipermedia a Vannevar Bush, un matemático y científico influyente en la parte media del siglo XX. En un ensayo titulado "As We May Think" 1.945, imaginó un sistema de escritorio que permitiría a la gente asociar la información relacionada de varias fuentes, la creación de "caminos" que ayudarían mucho a la humanidad con la explosión de la información en el mundo moderno.

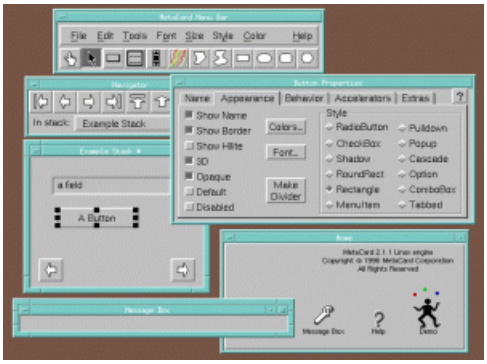
La parte del nombre hiper de HyperCard y los términos hipermedia y los hipervínculos se refiere a esta idea de la información asociada / vinculada. Por ejemplo, cuando se está leyendo algo en un libro y llega a un término que no entiende y quiere saber más acerca de él, usted puede ir a un diccionario o una enciclopedia. Una vez allí, es posible encontrar otro punto de vista, a continuación, buscar otra fuente y luego otra y así sucesivamente. Por supuesto, usted no está limitado a sólo libros, se podía ir a una biblioteca de videos, o hablar con un experto en el tema, o cualquier número de otros recursos. En un sistema hipermedia todo esto está disponible desde el ordenador.

HyperCard Avanzó rápido hasta el 2001. HyperCard había sido más o menos "huérfano" de Apple Computer, y varios contendientes habían aparecido como heredero del legado de HyperCard. Uno de los más prometedores de estos fue LiveCode, originalmente llamada Revolución (Brigham Young University, 2005).

Apple se interesó mucho en el lenguaje “HyperTalk” y lo compró para incorporarlo en uno de sus productos denominado HyperCard. HyperTalk era un lenguaje de programación extraordinariamente sencillo. Apple comercializó HyperCard desde 1987 hasta el año 2004, año en el cual Steve Jobs decidió dar por finalizado el producto y retirarlo del mercado (Vacas, 2015).

LiveCode está inspirado completamente en el lenguaje HyperTalk (Versión MetaCard) y la primera versión del lenguaje surgió en el año 2001. En aquella época se llamó “Revolution”. Revolution fue cambiado de nombre a “LiveCode” en el año 2010. La empresa que creó y desarrolla actualmente este lenguaje es Runtime Revolution Ltd, empresa escocesa con sede en Edimburgo.

Figura 3. Ventana panel de control de la primera versión de METACARD



Fuente: Adaptado Hermano_Temblon recuperado de: http://www.hermanotemblon.com/livecode-una-nueva-forma-de-programar/

Livecode ganó el premio de la revista Macworld (Macworld Annual Editor’s Choice Award) por “mejor desarrollo de software del año 2004”. A partir del año 2013 existen dos versiones de Livecode. Una versión de pago que cuesta unos 650€ y una versión libre (completamente pública), la cual es gratis pero pone como condición que si se comercializa un programa con la misma, es obligatorio publicar el código fuente (Licencia GPL).Se puede descargar LiveCode desde Http://livecode.com/ (Hay versiones de la herramienta de desarrollo para Windows, Linux y Mac) (RunRev, LiveCode7, 2000-2015).

Figura 4. Aplicación ejecutándose en un computador MAC

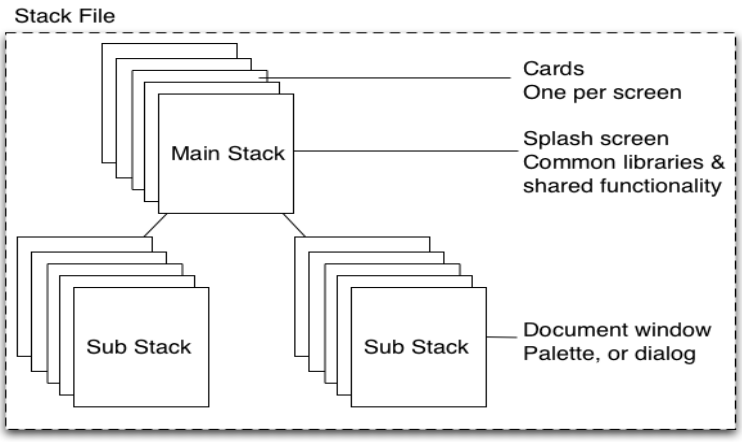
Fuente: Adaptado Hermano_Temblón recuperado de:
http://www.hermanotemblon.com/livecode-una-nueva-forma-de-programar/

2.5.2. ¿Cómo es LiveCode?

LiveCode es un lenguaje muy sencillo, tiene aproximadamente 1900 (un poco más) sentencias de programación y las mismas se pueden extender mediante "plugins" o programándolas directamente en lenguaje "C".

La filosofía de la herramienta de desarrollo se basa en una "baraja de cartas" denominada "Main Stack" o "Baraja principal" de la cual cuelgan "Cards" o "Cartas". (Vacas, 2015)

Figura 5. Estructura de una aplicación desarrollada en LiveCode



Fuente: Manual del usuario RunRev Ltd. (2010).
LIVECODE USER GUIDE. Copyright ©2010

Cuando se inicia la herramienta de desarrollo, se crea un nuevo proyecto. Este nuevo proyecto contendrá un “Main Stack” que será algo así como la “página” principal. De esta “página” colgaran las “pantallas” que van creando el programa. Cada una de éstas se llamará “Card” y tendrá un número único y exclusivo.

Por ejemplo: Para crear un pequeño software que se conecte a Internet y diga que tiempo hace en la calle, se hará lo siguiente:

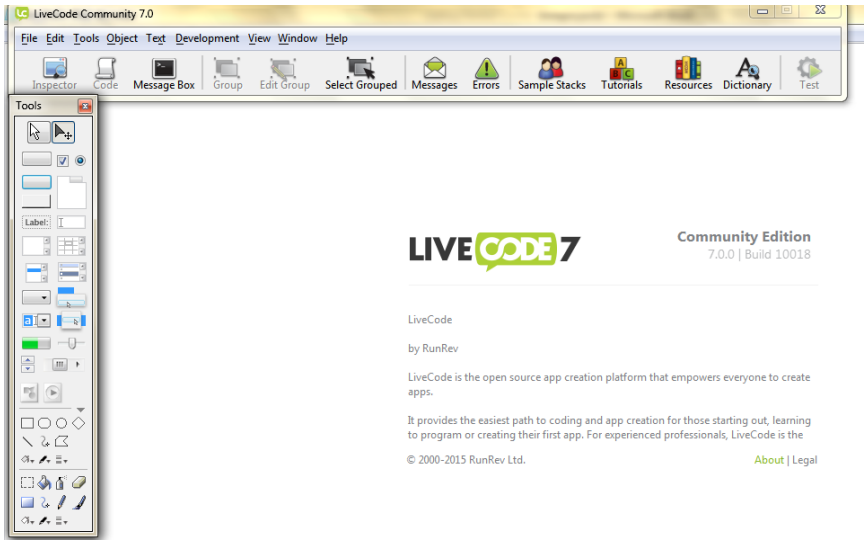
Primero: Crear un “Main Stack”

Segundo: Crear una nueva carta donde (por ejemplo) se invite a introducir la ciudad cuyo tiempo se quiere consultar.

Tercero: Crear una tercera carta donde aparecerá el resultado de la consulta. (Vacas, 2015).

La Herramienta es muy parecida al viejo “Visual Basic”, ya que se puede arrastrar a las diferentes cartas, botones, etiquetas, cuadros de texto, etc. Y programar cada objeto para que haga cualquier cosa. Es decir; técnicamente, LiveCode es un lenguaje de programación orientado a “eventos del ratón” (igual que Visual Basic).

Figura 6. Ventana del entorno del software LiveCode



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Figura 7. Página principal (Mainstack) de una aplicación realizada en LiveCode.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

3. CONTENIDO PRELIMINAR AL DESARROLLO DE APLICACIONES

Para poder abordar el lenguaje LiveCode se hace necesario aclarar conceptos de programación, con el fin de empezar a dar los primeros pasos en la elaboración de aplicaciones.

3.1. ¿QUÉ ES LA PROGRAMACIÓN?

En un sentido básico, la programación implica la creación de un conjunto de instrucciones para completar alguna tarea específica. En este sentido, muchas de nuestras actividades diarias pueden ser descritas como programáticas, implican pasos específicos que a menudo siguen un orden establecido. (Brigham Young University, 2005)

Por ejemplo, el proceso de ir a la escuela implica levantarse temprano a una hora establecida, alistarse para salir, coger una ruta, llegar puntual para el inicio de clases y cada cierto tiempo cambiar de asignatura. Si las cosas se hacen fuera de orden, pueden resultar dificultades.

En éste sentido general, nuestras vidas están llenas de programas y de programación. Por ejemplo cuando se arregla la cama se siguen ciertos pasos en forma programática. Los pasos deben ser los correctos y deben estar en el orden correcto. Si se quiere hacer el pastel de manzana favorita de la abuela, y le pregunta cómo hacerlo, lo más probable es que le envíe un programa, una receta. Un programa, por tanto, también es útil para la replicación de un producto, incluso lejos del creador original.

En el contexto de la informática, la programación significa la creación de un conjunto de instrucciones para un ordenador, con el fin de realizar una tarea específica. Para ello se utiliza un conjunto de directivas, un lenguaje de programación conocido tanto por el programador como por el sistema operativo del ordenador. Por lo general, un conjunto de instrucciones o programas para un equipo, tiene la intención de completar una tarea que: (Brigham Young University, 2005)

- Es repetitiva, y por lo tanto excedería la paciencia humana o capacidad de atención a largo plazo de los detalles.
- Controla la maquinaria en condiciones inadecuadas para los seres humanos debido a limitaciones físicas, condiciones peligrosas, etc.
- Requiere un alto grado de precisión.
- Requiere de alta velocidad.

3.1.1. Conceptos básicos de programación. A pesar de que cada lenguaje de programación que se utiliza es único, hay ciertos conceptos comunes a todos, incluido el lenguaje de LiveCode. Veamos tres de los conceptos y estructuras más comunes utilizados en la programación.

3.1.1.1. Secuencia de comandos (Los comandos adecuados en el orden correcto.)

Es importante no sólo dar las órdenes correctas o pasos que deben darse, también la secuencia correcta. Podemos ver fácilmente la toma de decisiones en algunas de nuestras acciones cotidianas como hacer un sándwich, atar los zapatos, seguir una receta etc. el orden adecuado es esencial para nuestro éxito. Podríamos llamar a dicha *orden tarea* secuencia obvia, porque la secuencia correcta es dictada por la naturaleza de la tarea. (Brigham Young University, 2005)

Hay procedimientos en los que el orden de los pasos es importante. A menudo, en tales procedimientos, una *orden convencional* surge para evitar confusiones. Un excelente ejemplo son las cartas que se envían por correo. Todos sabemos que se hace en este orden:

Nombre y apellidos:

Dirección:

Ciudad y departamento:

País y Zona postal:

Ejemplo: Supongamos que desea borrar de la pantalla todos los botones y campos (limpiar), mostrar un campo con texto (orden o repuesta), esperar a que el usuario haga clic, a continuación, ocultar el campo y mostrar a los anteriores. Para

que funcione correctamente, no sólo todos los comandos tienen que estar allí, tienen que estar en el orden correcto.

3.1.1.2. Estructuras condicionales (hacer ciertas cosas sobre la base de verdadero o falso, la decisión sí o no.)

Estos prevén un resultado o una secuencia de eventos que se ejecutan si un enunciado es verdadero, y otro resultado o secuencia de eventos que se activará si la afirmación es falsa.

En la mayoría de los lenguajes de programación estas estructuras toman la forma **si. . . entonces. . .** **si no**… condiciones, **If**… **then**… **Else**…

Ejemplo:

Si existe una palabra en una lista, **entonces,** imprimirlo, **Si no** avisar al usuario de que no existe la palabra.

3.1.1.3. La estructura del bucle (una lista de instrucciones para hacer más de una vez.)

Se utiliza para hacer que el equipo repita una cierta orden o secuencia de comandos. El bucle puede funcionar durante un número predeterminado de veces, hasta que una cierta condición sea verdadera, o el tiempo que una determinada condición sigue siendo verdadera.

Aquí hay algunas maneras donde la estructura de bucle se podría usar:
Hacer las siguientes 20 horas.
Haga lo siguiente una vez para cada palabra en la lista
Repetir lo siguiente hasta que el usuario pulsa la tecla de opción
Repita lo siguiente, siempre y cuando la tecla de opción está pulsada (Brigham Young University, 2005).

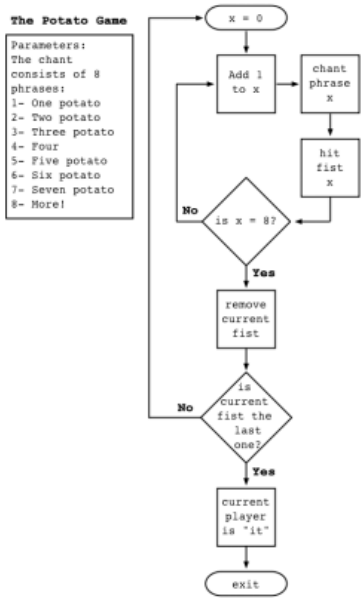
Otro ejemplo:
Dada una lista de estudiantes de un curso, asignar cada uno, a uno de tres grupos para los juegos que se plantearon para la clase.

3.2. ESTRATEGIAS DE PROGRAMACIÓN

La programación puede variar en complejidad desde la solución de pequeños problemas como el establecimiento de una hora de alarma en su reloj o teléfono, a aplicaciones de instrucción o de negocios muy sofisticados. Para las tareas más complejas, puede utilizar estas estrategias que le ayudaran a pensar a través de la lógica del programa antes de empezar a escribir código.

3.2.1. Diseño descendente o diagrama de flujo. El diseño de arriba hacia abajo o diagrama de flujo, es una forma de acercarse a una tarea compleja de programación por medio de gráficos y la identificación de los principales componentes que va a requerir. A continuación, el programador podría utilizar diagramas de flujo y declaraciones generales para representar el flujo lógico de su programa. Una vez que se identifican los principales componentes, el programador se centra en cada componente en mayor detalle, finalmente culmina en escribir el código del programa para la creación de cada componente.

Figura 8. Ejemplo de diagrama de flujo.



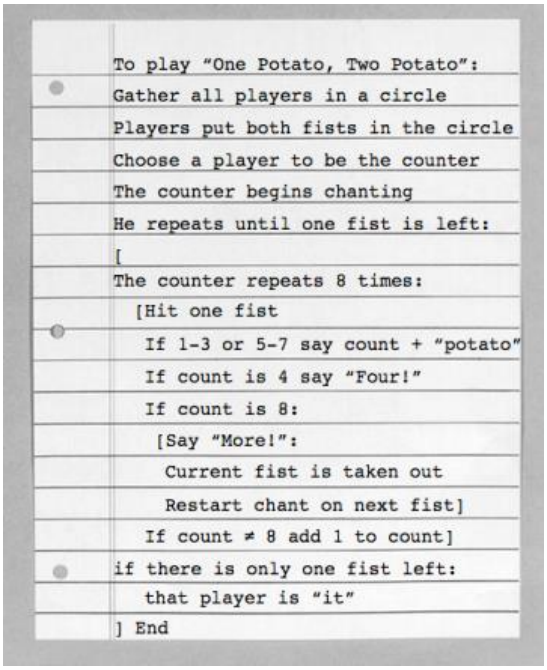
Fuente: Brigham Young University recuperado de: http://livecode.byu.edu/programmingconcepts/ControlStruct.php

En el ejemplo se ha representado un enfoque de arriba hacia abajo o diagrama de flujo. Cada forma en el diagrama representa un paso importante. Las

combinaciones de formas y flechas muestran puntos condicionales "si - entonces" decisiones, así como las estructuras de bucle en el que se repiten los segmentos del programa, quizás con ligeras variaciones en cada iteración.

3.2.2. Pseudocódigo. Este término, del pseudo-prefijo, "falso" y el código raíz de la palabra, hace referencia a “instrucciones de programación”, describe una forma de representar los pasos detallados que el programa debe realizar sin tener que preocuparse por el vocabulario o la sintaxis de un lenguaje de programación específico. Se utiliza el conocimiento de las estructuras de control básicas, el sentido común y la lógica para escribir declaraciones claras (en inglés), para explicar en detalle cómo va a lograr un paso principal (Brigham Young University, 2005). Todos los ejemplos que se muestran podrían ser considerados formas de pseudocódigo.

Figura 9. Ejemplo de seudocodigo.



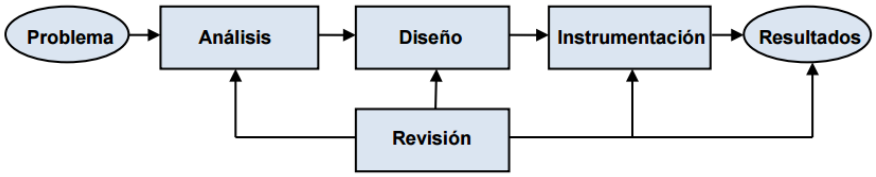
Fuente: Brigham Young University recuperado de: http://livecode.byu.edu/programmingconcepts/ControlStruct.php

3.3. UN MODELO PARA RESOLVER PROBLEMAS CON EL COMPUTADOR

El análisis y diseño de soluciones computacionales es una ciencia que facilita el uso eficiente del poder de las computadoras para resolver problemas.

Para facilitar el desarrollo de estas soluciones, es adecuado usar un lenguaje computacional simple, general y eficiente como el que ofrece LiveCode. El siguiente gráfico describe los pasos en la solución computacional de un problema (Rodríguez Ojeda, 2015)

Figura 10. Diagrama de resolución de problemas



Fuente: Phyton Programacion, Luis Rodriguez recuperado de: http://www.librearchivo.tk/2016/04/python-programacion-luis-rodriguez-pdf.html

Primero, es necesario asegurarnos que el problema que intentamos resolver está en nuestro ámbito de conocimiento. No es recomendable intentar resolver problemas si no tenemos el conocimiento y la práctica para enfrentar su solución. Rodríguez Ojeda plantea que:

En la etapa de Análisis se estudia el problema en forma detallada: sus características, las variables y los procesos que intervienen. Asimismo, se deben definir los datos que se requieren y cuál es el objetivo esperado. El resultado de esta etapa son las especificaciones detalladas de los requerimientos que en algunos casos se pueden expresar en forma matemática.

En la etapa de Diseño se procede a elaborar los procedimientos necesarios para cumplir con los requerimientos especificados en el análisis, incluyendo fórmulas, tablas, etc. El objeto resultante se denomina algoritmo.

En la etapa de Instrumentación, si el problema es simple, se puede obtener la solución interactuando directamente mediante instrumentos disponibles en el

entorno computacional. Si el problema es más complejo, deben construirse programas y definir el ingreso y la organización de los datos necesarios.

Al concluir la etapa de la instrumentación, se usan datos para realizar pruebas de los programas. Es necesario que se realice una revisión en cada etapa de este proceso y que se validen los resultados obtenidos antes de aceptarlos y continuar (Rodríguez Ojeda, 2015, págs. 10-11).

Posteriormente se efectúa la instalación y operación. Debe preverse la necesidad de mantenimiento y cambios en los programas para ajustarlos al entorno en el que se usarán.

Los instrumentos computacionales modernos tales como LiveCode disponen de facilidades para probar interactivamente instrucciones y programas a medida que son desarrollados, mejorando así la productividad. También ofrece librerías que se generan automáticamente y facilitan el desarrollo de los proyectos de programación.

3.4. HERRAMIENTAS TECNOLÓGICAS PREVIAS

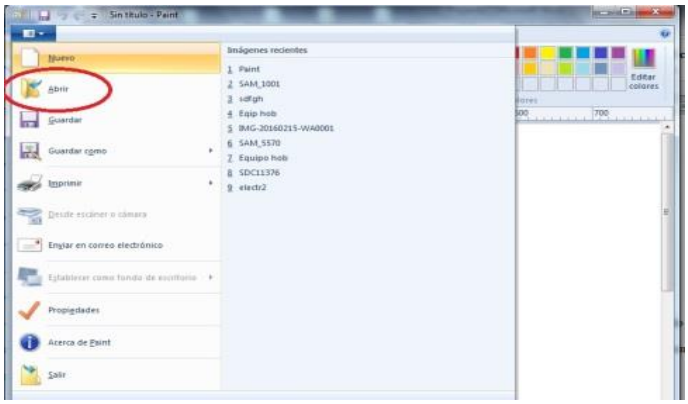
También se debe establecer unos contenidos mínimos de manejo de fotografía porque serán objetos importantes en la visualización de los programas, con este fin se presentará el manejo de Paint y Photoshop (el propósito no es enseñar manejo de fotografía) en las funciones de recortar imágenes y reducir pixeles que serán las más usadas, pero el lector podrá utilizar cualquier otro programa que se le facilite o que pueda ser más eficiente.

3.4.1. Paint

3.4.1.1. Recortar imágenes. Útil cuando se desea usar secciones de fotografías, las fotografías recortadas quedan dentro de un rectángulo de fondo blanco.

Para ello, abrir el programa y buscar el archivo que se desea recortar, entonces, ir a la parte superior izquierda y hacer clic en el botón azul que desplegará un menú con la opción ***abrir.***

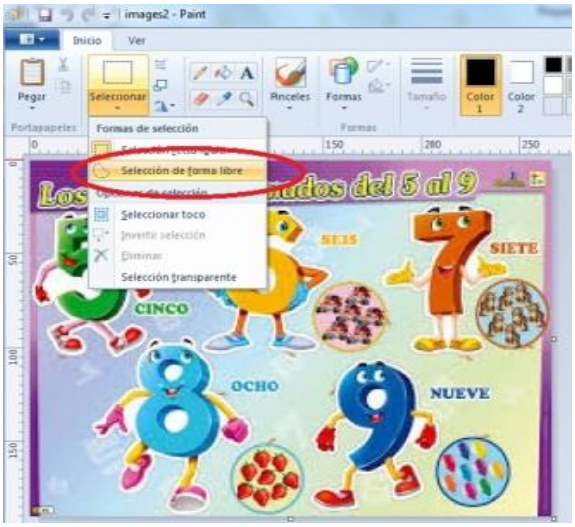
Figura 11. Menú de Paint (Abrir).



Fuente: Esta ilustración proviene del software **Paint**

Una vez abierto el archivo de imagen, hacer clic en la opción ***seleccionar*** del cual se despliega un menú, luego elegir ***selección de forma libre.***

Figura 12. Comando selección (Paint).



Fuente: Esta ilustración proviene del software **Paint**

Ubicar el puntero del ratón (mouse) en el dibujo que se desea recortar y manteniendo pulsado el botón izquierdo hacer el recorrido del contorno que se desea, dicho contorno se va delineando, cuando termine, suelte el botón del ratón, esto seleccionará la figura mediante un rectángulo en línea punteada.

Figura 13. Contorno usando la opción Selección de forma libre.

Fuente: Esta ilustración proviene del software **Paint**

Una vez seleccionada la figura se elige ***recortar*** en el menú ***Inicio***, se obtiene la figura recortada que se guarda como una nueva imagen.

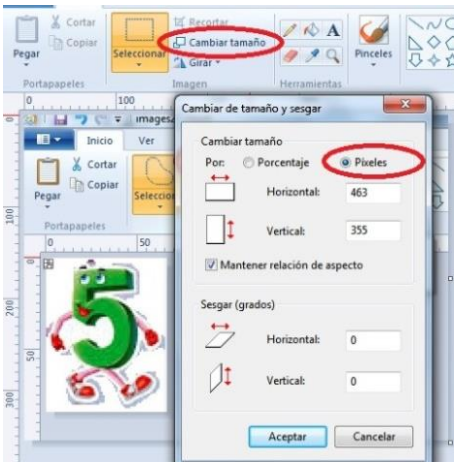
Figura 14. Opción Recortar (Paint).

Fuente: Esta ilustración proviene del software **Paint**

3.4.1.2. Reducir pixeles. Útil cuando se desea usar la fotografía completa pero con menor tamaño.

Abrir el archivo que se desea, en el menú ***Inicio*** buscar la opción seleccionar que por defecto es ***selección rectangular*** y seleccionar la fotografía, a continuación buscar la opción ***cambiar tamaño*** y hacer clic, saldrá una ventana emergente con dos opciones cambiar tamaño y sesgar (inclinar la fotografía), dentro de cambiar tamaño está la opción predeterminada en *porcentaje*, hacer clic en *pixeles* y escribir el número deseado para la medida vertical u horizontal y podrá finalizar, haciendo clic en *Aceptar*.

Figura 15. Cambiar tamaño y pixelado.



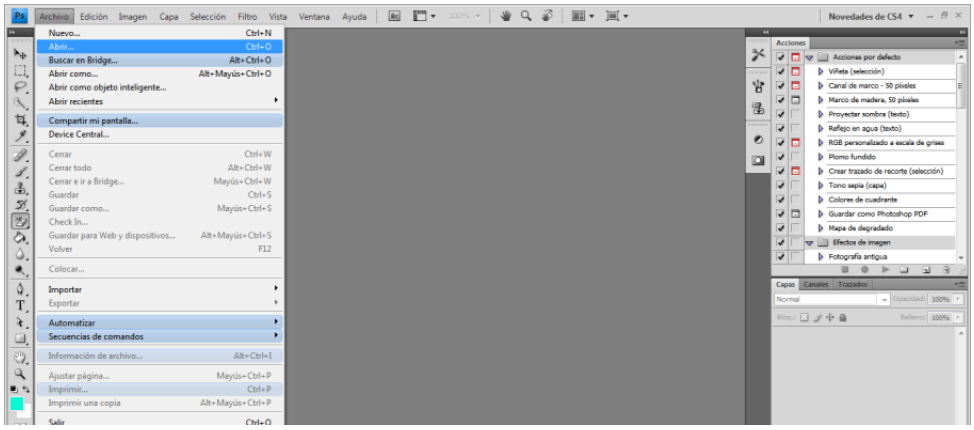
Fuente: Esta ilustración proviene del software **Paint**

3.4.2. Photoshop

3.4.2.1. Recortar imágenes. Útil cuando se desea editar fotografías para suprimir fondos y recortar con alta calidad de edición y fácil de usar.

Abrir el programa y buscar el archivo que desea recortar, para abrirlo ir a la parte superior izquierda y hacer clic en el menú ***Archivo*** y luego en la opción ***abrir*** o con las teclas ***Ctrl + O.***

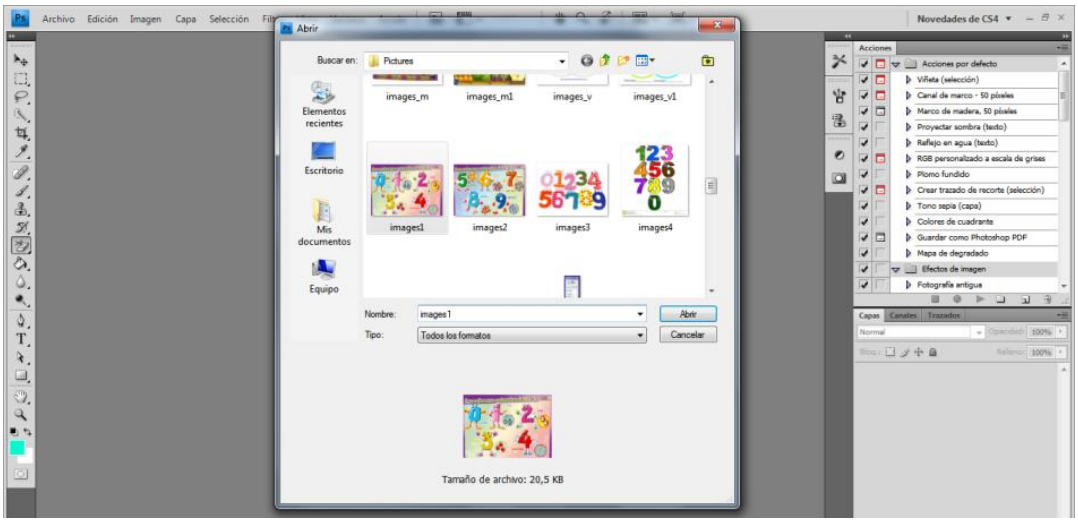
Figura 16. Menú archivo (Photoshop).



Fuente: Esta ilustración proviene del software **Photoshop CS4**

Se despliega una ventana que permite buscar el archivo deseado, luego de hallarlo hacer clic en la opción *abrir.*

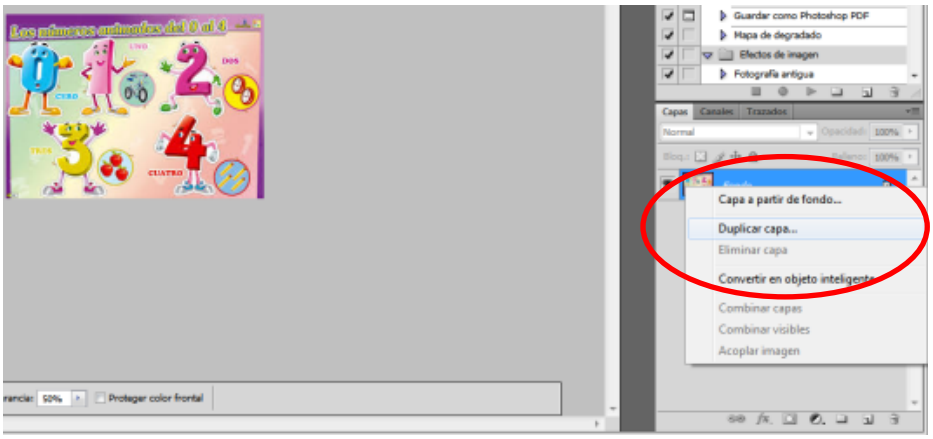
Figura 17. Abrir archivo.



Fuente: Esta ilustración proviene del software **Photoshop CS4**

Después de abierto el archivo, en la parte inferior derecha hacer clic derecho sobre ***capa*** y seleccionar la opción ***duplicar capa***, esto para trabajar en un duplicado del objeto

Figura 18. Duplicar capa.



Fuente: Esta ilustración proviene del software **Photoshop CS4**

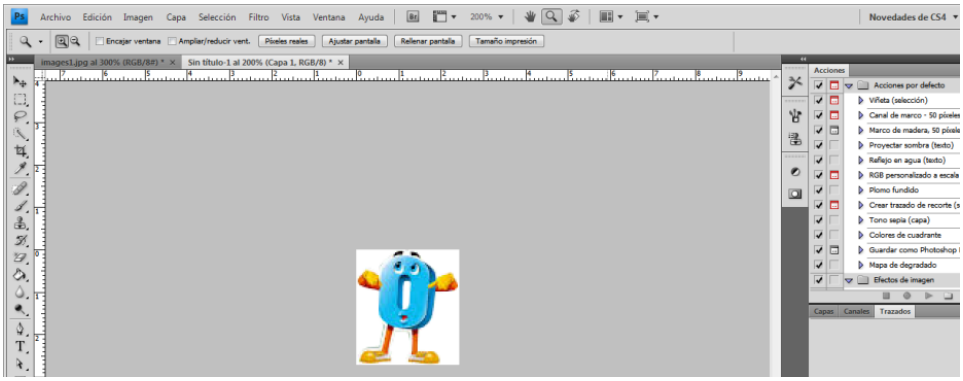
En el panel de herramientas, desplegado en la parte izquierda seleccionar la pluma, ésta permite ir bordeando la figura que deseamos recortar, iniciar haciendo clic en cualquier parte del borde y continuar siguiendo el borde del objeto, cuando es lineal se puede avanzar en mayor longitud y cuando encontramos curvas punteamos en espacios cortos de tal forma que se pueda hacer la curva lo más finita posible.

Figura 19. Bordear con pluma.

Fuente: Esta ilustración proviene del software **Photoshop CS4**

Cuando llegamos al punto de inicio el contorno se vuelve una línea continua sin puntos, entonces hacer clic derecho, pulsar en la ventana emergente en la opción *hacer selección*, crear un archivo nuevo y pegar el recorte. Como consejo guardar el archivo en formato PNG o JPG.

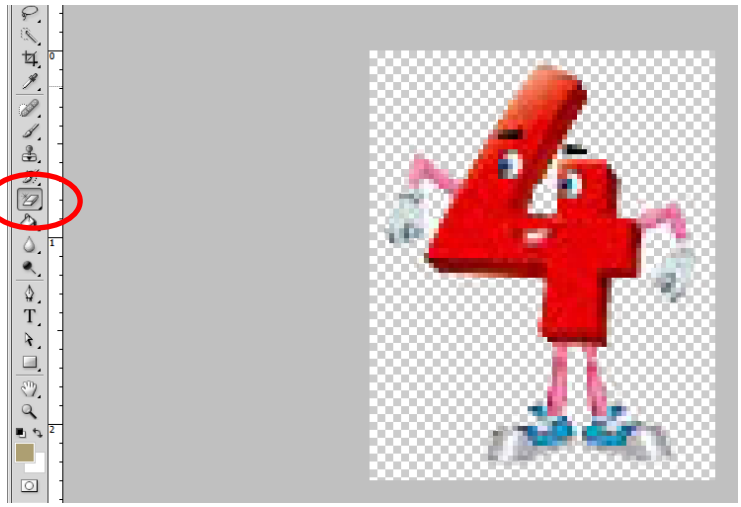
Figura 20. Objeto recortado en nueva ventana.



Fuente: Esta ilustración proviene del software **Photoshop CS4**

Otra forma, cuando las fotografías tienen un solo fondo utilizar la herramienta *borrador mágico* o *borrador de fondo,* pulsando clic derecho sobre el borrador, e ir haciendo clic sobre el fondo para ir borrando lo que se desea desaparecer, luego guardar el nuevo archivo.

Figura 21. Herramienta Borrador.



Fuente: Esta ilustración proviene del software **Photoshop CS4**

4. MANUAL DE APLICACIONES EN LIVECODE

4.1. DESCARGAR LIVECODE COMUNITY

Para descargar el programa de LiveCode gratuito y totalmente legal se puede hacer desde la página del autor en el link http://downloads.livecode.com/livecode/, en ella encontrará además todas las versiones desde los programas que salieron como prueba para los desarrolladores y que presentan errores hasta los ya estables para el público general, ahora a conocer el programa.

4.2. CONOCIENDO EL ENTORNO DE LIVECODE

El entorno de LiveCode lo podemos dividir en tres partes:

La barra de Menú Principal que contiene File (archivo), Edit (Edición), Objet (objetos), Text (texto), Development (desarrollo), View (vista), Windows (ventanas) y Help (ayuda) cada uno de ellos contiene sub-menús que nos serán útiles en la realización de los proyectos.

La paleta de herramientas que es visible por defecto y se ubica en la parte izquierda.

Las ventanas de los Stacks o las tarjetas que se van creando en el centro de la pantalla en forma de cascada.

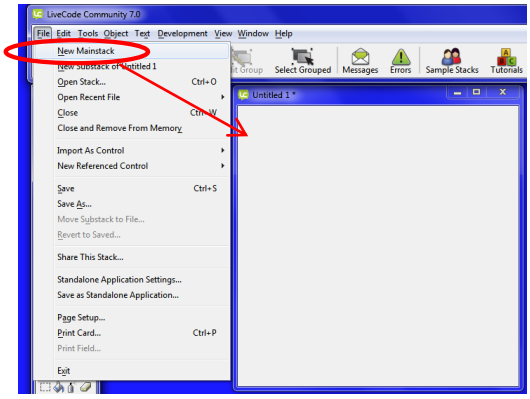
Figura 22. División principal del entorno LiveCode.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.1. Submenú Archivo. Contiene comandos para ejecutar una secuencia de órdenes o cerrar el editor de scripts, con o sin guardar los cambios en el guion. Se definirán los más usados:

Nueva pila de cartas o baraja principal (*New Mainstack*): abrirá la carta principal, es algo así como la portada del programa de la cual colgaran las demás cartas.

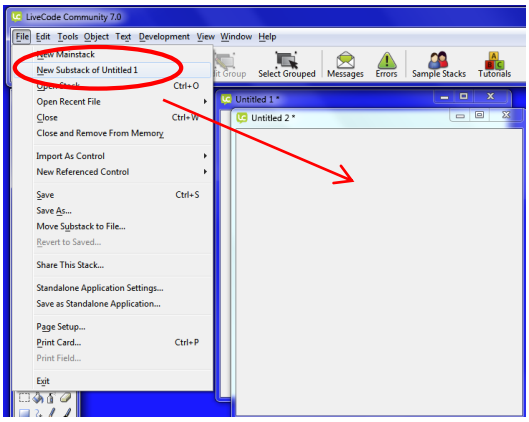
Figura 23. Creación de página principal (Mainstack).



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Nueva sub-pila o sub-baraja (*New Substack*): permite la creación de una nueva carta que se ira sumando al proyecto, esta además permite acomodar objetos en ella que serán programables.

Figura 24. Creación de substack.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

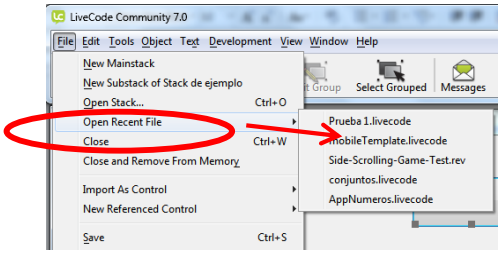
Abrir pila o baraja de cartas (*Open Stack*): Abre una ventana emergente que permitirá abrir una pila (aplicación) editada y guardada anteriormente.

Figura 25. Ventana emergente para abrir un Stak o proyecto.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Archivos Abiertos Recientemente (*Open Recent File*): Despliega una ventana con los últimos archivos editados.

Figura 26.Abrir proyecto desde el menú File.

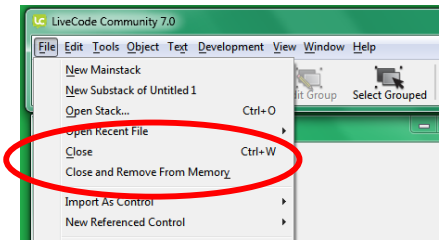


Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Cerrar (*Close*): Cierra el editor o Script.

Cerrar y borrar de la memoria (*Close and Remove From Memory*): Cerrar sin guardar cambios.

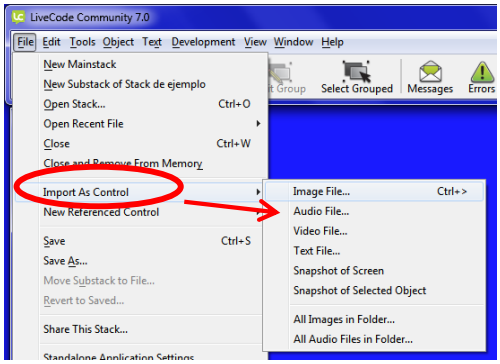
Figura 27. Opciones de cierre desde menú File.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.1.1. Control de Importaciones (Import As Control). Despliega una ventana con los comandos, Archivo de Imagen, Archivo de Audio, Archivo de Video, Archivo de Texto: Abre una ventana emergente para ubicar la carpeta donde están las imágenes, los audios, los videos, o los textos que se van a importar como objetos de las cartas o tarjetas.

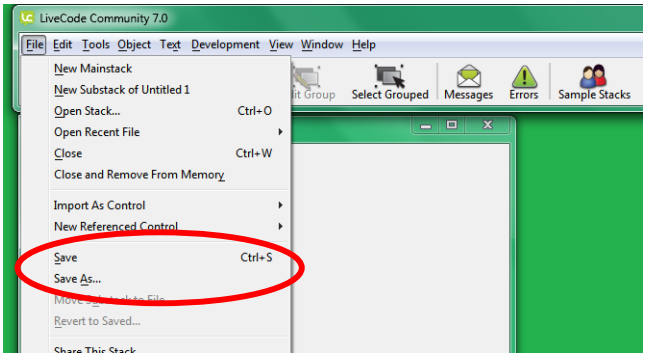
Figura 28. Importación de objetos a las cartas o tarjetas.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Guardar y Guardar como… (*Save y Save As*): Guardan el proyecto en la carpeta especificada.

Figura 29. Guardar y Guardar como…



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Compilar aplicación (*Standalone Application Setting*…) permite hacer la compilación de la aplicación móvil y la entrega en un paquete instalador.

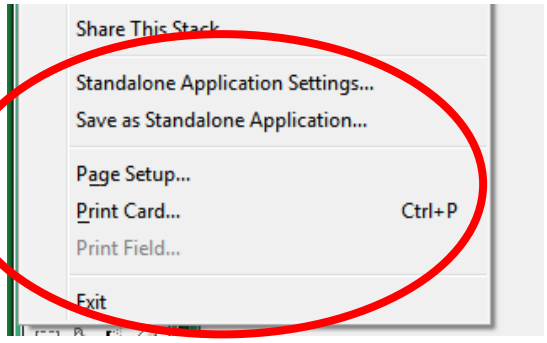
Configuración de Pagina (*Page Setup*): Permite configurar la página para imprimir.

Imprimir Carta o Tarjeta (*Print Card*).

Imprimir Campo (*Print Field*).

Salir (*Exit*): Salir de LiveCode, cierra el Programa.

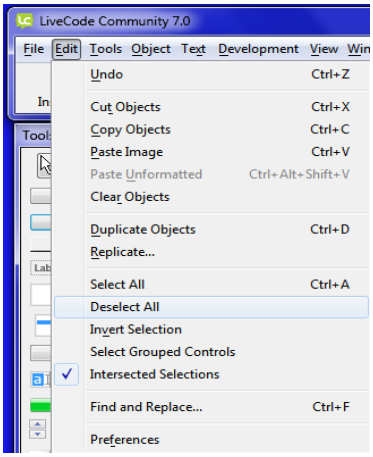
Figura 30. Configurar página, imprimir y salir.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.2. Sub menú Editar

Figura 31. Imagen del Sub-menú Editar



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Encontrará las siguientes opciones concernientes a la edición y modificación de los objetos en las tarjetas.

Deshacer *Undo* (ctrl z): Permite deshacer los últimos cambios que se han realizado.

Cortar (*Cut Objects*): Permite cortar los objetos seleccionados.

Copiar (*Copy Objects*): Permite copiar los objetos seleccionados.

Pegar (*Paste y Paste Unformatted*): Pega objetos en la misma tarjea o en la deseada por el programador conservando el formato (Paste) o sin formato (paste unformatted).

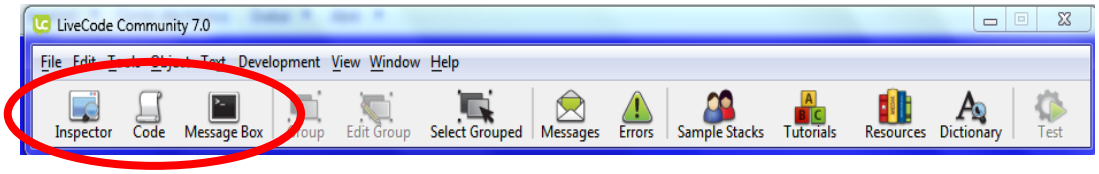
Borrar (*Clear Objects*): Elimina el objeto seleccionado.

Duplicar Objetos (*Duplicate Objects*): explicado en la pág. 66.

Replicar objeto (*Replicate*): explicado en la pág. 66.

4.2.3. Los botónes Inspector, Code y Message Box

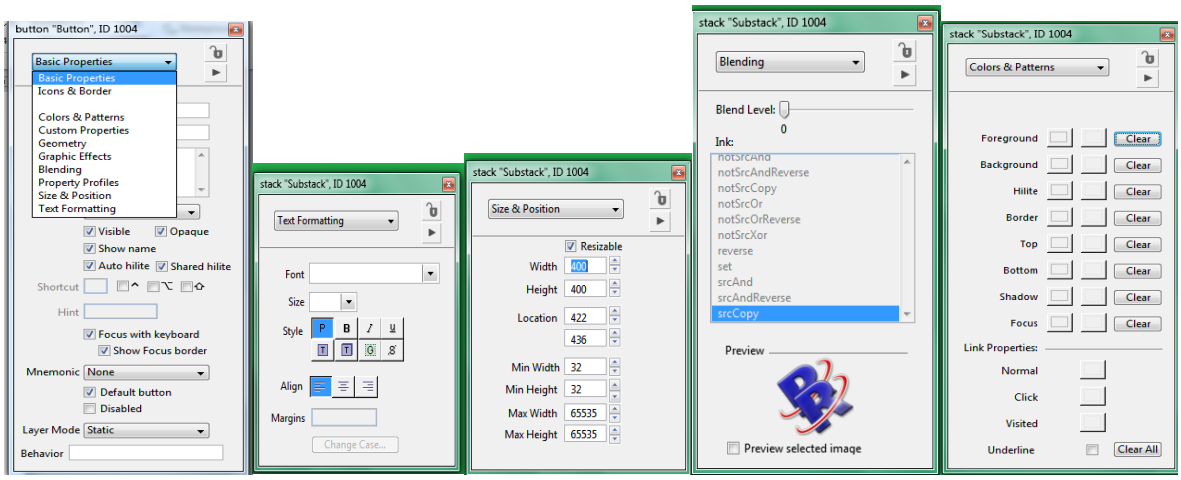
Figura 32. Botones Inspector, code y Message box en la barra de herramientas.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.3.1. Inspector. Despliega una ventana emergente en la cual se cambian los atributos de los objetos como nombre, colores, bordes, efectos gráficos, tamaño, formato del texto y muchas otras que se encuentran en el botón desplegable que dice por defecto propiedades básicas.

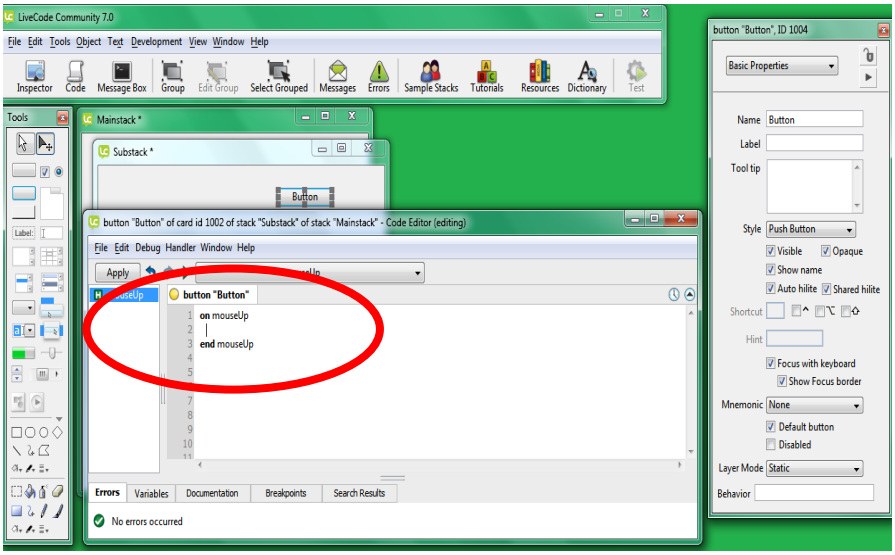
Figura 33. Opciones del botón inspector.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.3.2. Code. Es la ventana de programación donde se escriben los códigos para que los objetos hagan tareas, al pulsar el ícono, despliega una ventana emergente donde se escriben los comandos para controlar el objeto seleccionado, cuando esta seleccionado un botón en la tarjeta, sale escrito por defecto el comando On mouseUp, y en la parte superior aparece un circulo de color naranja, seguido del tipo de controlador y el nombre del mismo, como aparece en la figura 47.

Figura 34. Ventana de programación Code.

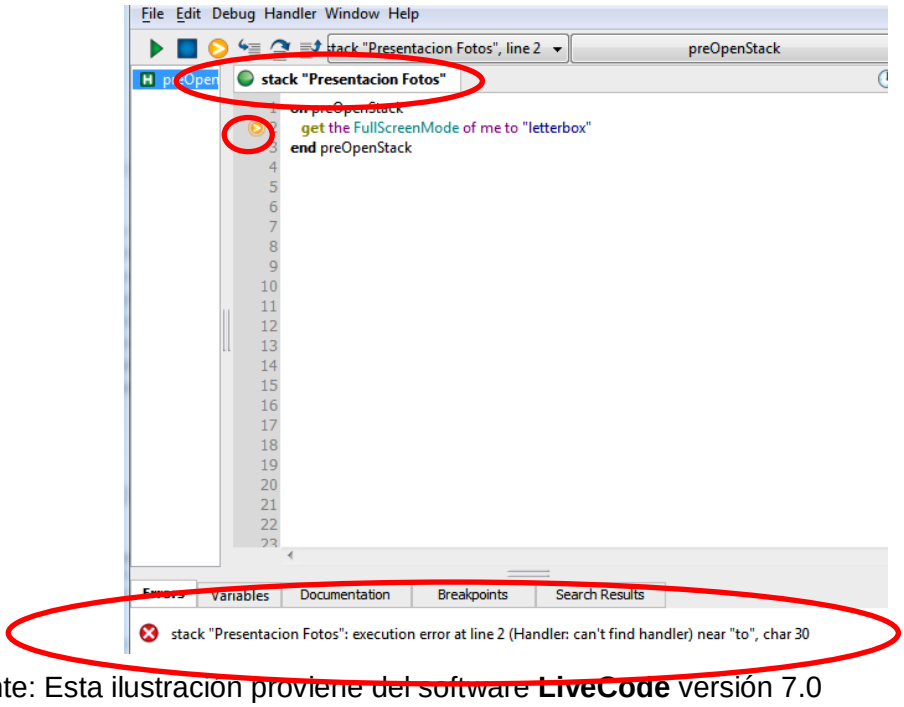


Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

La parte inferior de la ventana Code, presenta unas opciones muy útiles, estas son:

4.2.3.2.3. Error. Es una ventana que presenta las líneas de código donde hay errores, estos pueden ser de escritura, de sintaxis o de comando. Hace una pequeña descripción con la sección donde está el error, la línea de código y la posible falla, cuando no hay errores aparece un circulo verde con el mensaje *No errrors occurred*, cuando hay errores aparecerá como en la figura 48.

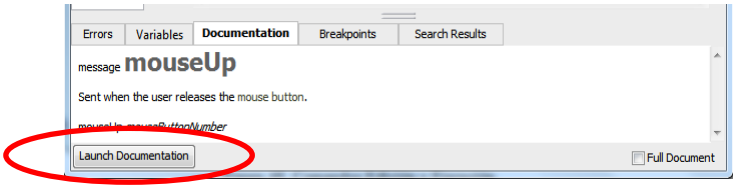
Figura 35. Ventana Code cuando hay error en la programación.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.3.2.2. Documentation. Presenta unas cuantas líneas que explican el ultimo código usado despliega un botón (Launch Documentation) que lleva al diccionario para obtener la documentación total de dicho código.

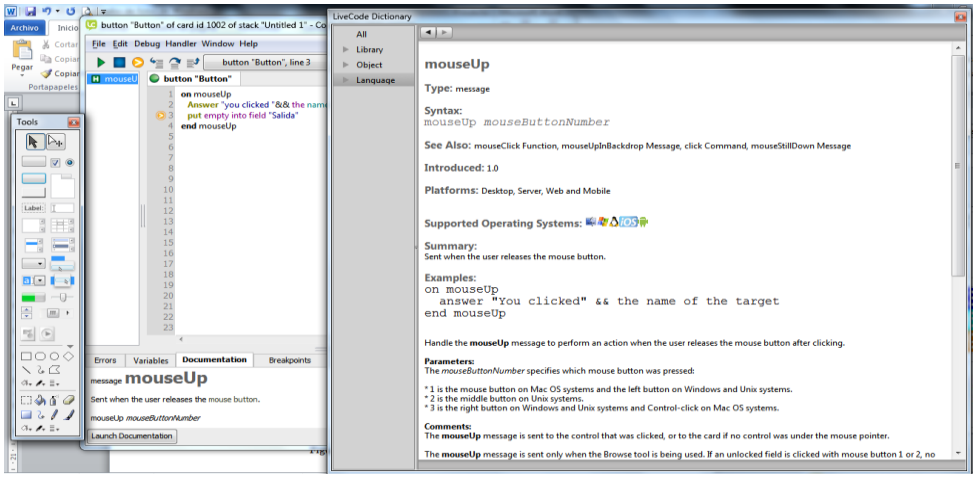
Figura 36. Despliegue de la opción de documentación



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

En la ventana emergente aparece el nombre del comando consultado, la sintaxis, ejemplos en los que se puede usar, los sistemas operativos en los que tiene soporte y una descripción de los parámetros que pueden ser utilizados así como algunos comentarios.

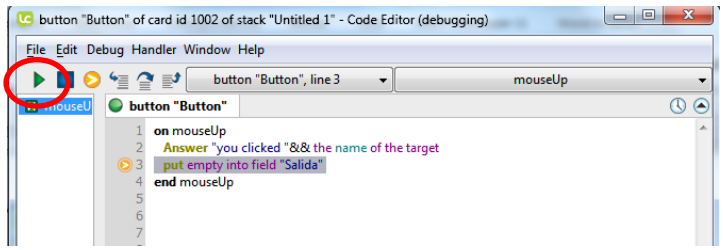
Figura 37.Despliegue del Diccionario al presionar Launch Documentation.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Cuando aparecen errores la ventana code se bloquea, por lo que no permite escribir, para desbloquearla hacer click en el triángulo verde de la parte superior izquierda de la misma ventana o presionar la tecla de funciones rápidas F5 en la parte superior del teclado, de esta forma podremos corregir el código.

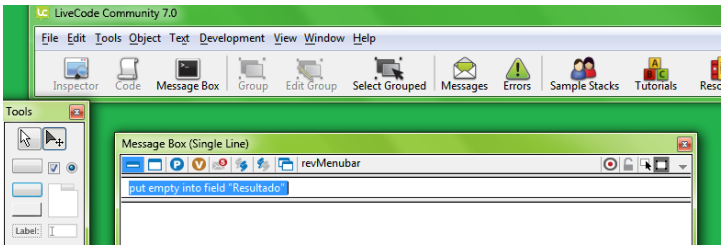
Figura 38. Botón para desbloquear Code.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.3.3. Massage Box. Permite probar comandos y sentencias de los que no se está seguro de su función.

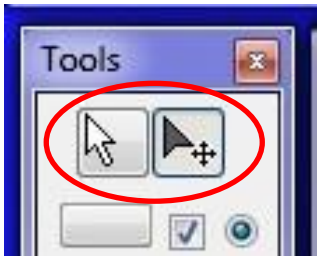
Figura 39. Ventana Massage Box.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.2.4. Modo de Edición y Modo Ejecución

Figura 40. Comandos Edición y Ejecución.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

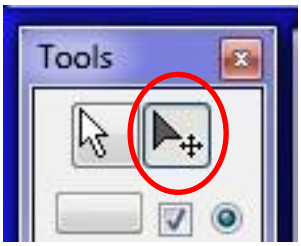
El botón que tiene la flecha es el comando Ejecución y lo utilizamos para correr las órdenes que hemos dado a los objetos (ejecutar el programa realizado) con el fin de observar errores o mirar cómo va el programa.

Para esto solo pulsar sobre el botón y el objeto programado ejecutará las órdenes dadas (correr la aplicación).

El botón que tiene una cabeza de flecha junto a un eje cartesiano, es el comando de edición, permite programar el objeto que este seleccionado, contrario al comando anterior al seleccionar un objeto le aparecerán las aristas y así permite programarlo.

Para saber en qué modo estamos basta mirar cual está de color negro.

Figura 41. Comandos Edición ejecutándose



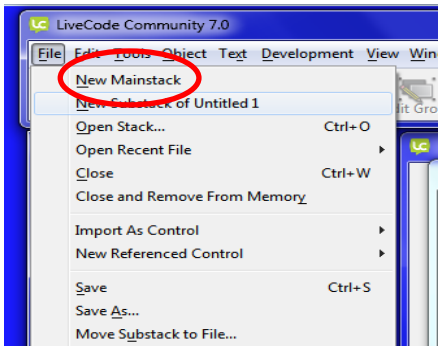
Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.3. CONOCIENDO LOS OBJETOS DE TRABAJO DE LIVECODE

Se puede comparar LiveCode a un juego de Lego, en el sentido de que se pueden construir cosas diferentes con él, mediante el uso de un suministro de piezas versátiles estándar. Con un juego de Lego tiene un suministro de ladrillos de diferentes tamaños, formas y colores. En LiveCode los "bloques de construcción básicos" que se utilizan para construir aplicaciones se denominan ***objetos***. Hay varios tipos de objetos, así como variaciones dentro de los tipos de objetos. Vamos a echar un vistazo a cada tipo de objeto LiveCode.

4.3.1. Pilas o Barajas. El bloque de construcción fundamental en LiveCode es la ***pila***. Ningún otro objeto puede existir independiente de una pila. Así que el primer paso para crear una aplicación es crear una pila. Esto se logra mediante la selección de **New Mainstack** desde el botón **archivo** de menú.

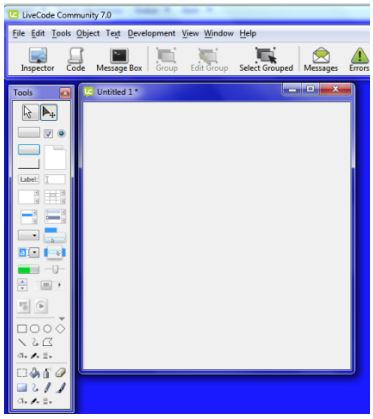
Figura 42. Creando un Mainstack o baraja principal en LiveCode.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Una vez creada la pila puede agregar otros objetos. Una vez que se oprime *new Mainstack* aparece la primera carta del proyecto titulada por defecto Untitled, después se debe abrir la ventana ***Inspector*** que tiene su icono en la parte superior izquierda allí se podrán cambiar los atributos del objeto que este seleccionado, como nombre, tamaño, color, etc. Esto se ejemplificara más adelante.

Figura 43. Primera cata o tarjeta el Mainstack.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

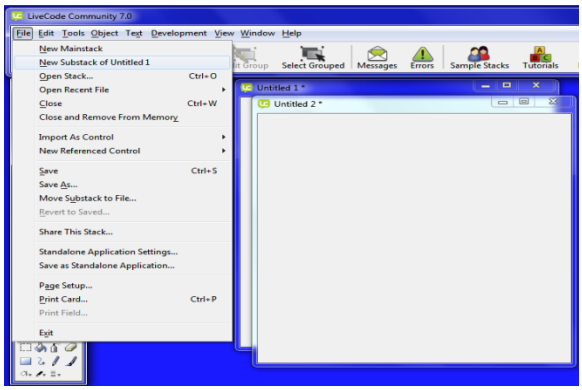
De hecho, todas las ventanas que se ven en el entorno de edición de LiveCode (incluyendo todas aquellas paletas de lujo) es una pila. Esto le da una idea de la flexibilidad del entorno LiveCode.

4.3.2. Cartas o Tarjetas. El siguiente bloque de construcción fundamental en LiveCode es la tarjeta. Cada pila muestra la información que contiene en una o más tarjetas. Por diseño cada pila tiene que tener al menos una tarjeta. Eso significa que cada vez que se crea una nueva pila, se crea también una tarjeta, para que nunca tenga que crear manualmente la primera tarjeta.

Como se mencionó anteriormente, las tarjetas son una pila donde se muestra la información que contiene. El usuario puede colocar diferentes tipos de información y objetos en diferentes tarjetas, de modo que la apariencia de las tarjetas comenzará a diferir una de otra. Una pila puede teóricamente tener un número ilimitado de tarjetas, pero en la práctica, cuando una pila llega a varios miles de tarjetas el rendimiento de LiveCode puede ser lento y difícil de manejar. A pesar de que una pila puede tener varias tarjetas, sólo una tarjeta junto con los objetos presentes en ella, es visible a la vez (Brigham Young University, 2005).

Para agregar una carta a la pila, simplemente se selecciona **New Card** del submenú **objet** en la barra de menú principal. La primera tarjeta que se crea aparece automáticamente con el título Untitled 2.

Figura 44. Segunda tarjeta en forma de cascada.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.3.3. Grupos. Los grupos son un tipo especial de objeto en LiveCode. Son algo así como tarjetas, ya que pueden contener otros tipos de objetos de control, creando lo que se denomina "controles agrupados". Pero los grupos, junto con los controles agrupados dentro de ellos, también se pueden colocar en las tarjetas. De esta manera los grupos se comportan como objetos de control. La naturaleza dual de los grupos tiene implicaciones importantes.

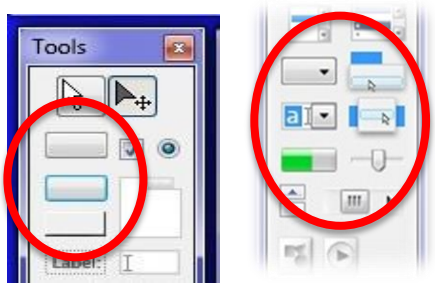
4.3.4. Objetos de control. Todos los objetos (conocidos como "controles") están contenidos y se muestran en las tarjetas de una manera u otra. Los principales objetos de control que se utilizan son los botones, campos, imágenes, gráficos y los players. Hay otros tipos de objetos de control, como las barras de desplazamiento y menús.

Los diferentes tipos de objetos de control que LiveCode proporciona son:

4.3.5. Botones. Uno de los tipos de control que puede existir en una tarjeta es el ***botón***. Un botón normalmente recibe un clic del ratón de un usuario y luego realiza una acción o serie de acciones específicas. Son los botones los que dan a una pila gran parte de su funcionalidad y ofrecen a los usuarios un grado de control sobre la pila y la información que ven.

Es posible crear un nuevo botón en una tarjeta haciendo clic y arrastrando una de las herramientas de botón en la paleta de herramientas hasta la tarjeta.

Figura 45. Herramientas de botones.

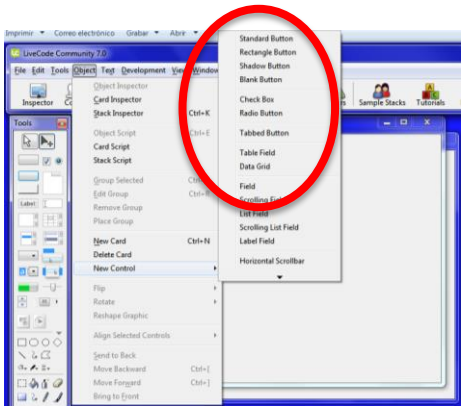


Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

También puede crear un botón eligiendo **New Control** del submenú **Objet** en la barra de menú principal. Sólo tiene que elegir el tipo de botón que desee, y un nuevo botón se creará automáticamente y se coloca en el centro de la tarjeta.

Se debe tener en cuenta que la apariencia de los botones y otros objetos de la paleta de herramientas se corresponde con la interface del sistema operativo que se está trabajando. Los objetos en una pila LiveCode abiertas en Windows asumirán el aspecto de Windows. Si se va a abrir la misma pila en Mac OS X los objetos asumirán el aspecto OS X; y si las pilas se abren en un sistema Linux se parecerán y se comportaran igual que la versión de Linux que estas utilizando. Esta característica hace que sea fácil crear aplicaciones multiplataforma en LiveCode sin tener que preocuparse por problemas de interfaz específicas para cada plataforma (Brigham Young University, 2005).

Figura 46. Creación de botones desde New Control



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Como se ha observado, hay varios tipos de botones disponibles para usar en las aplicaciones. Su apariencia y comportamiento son diferentes de acuerdo a su función y propósito. La paleta de herramientas proporciona acceso a todos los botones, incluyendo todas las formas de botones, botones estándar con pestañas, botones de casillas de verificación, botones de opción y botones de menú.

4.3.6. Campos

Figura 47. Campos en la paleta de herramientas de LiveCode

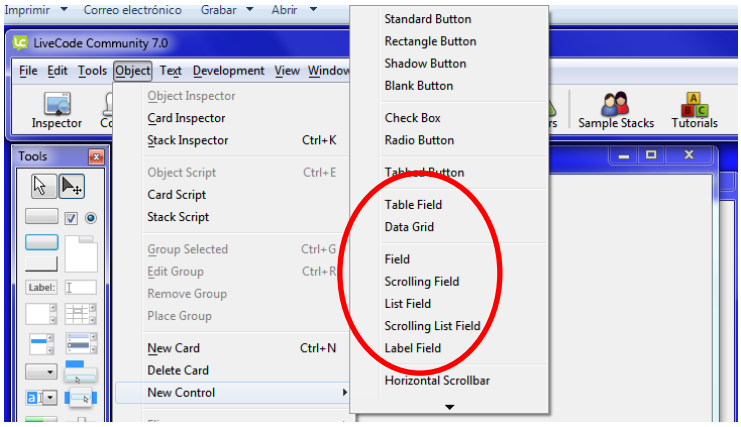
Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Un ***campo*** en LiveCode es un recipiente para contener y mostrar el texto. Los campos se pueden utilizar para presentar texto donde el usuario lea, mostrar etiquetas en otros objetos de la tarjeta, o para proporcionar un lugar donde el usuario pueda escribir, por ejemplo, completar la información, responder a una pregunta de examen, etc.

Al igual que con los botones, puede crear un campo en una tarjeta haciendo clic y arrastrando una de las herramientas de campo en la paleta de herramientas hasta la carta o tarjeta.

También puede crear un campo por la elección de un tipo de campo bajo ***New control*** en el submenú ***Objet***. Esto crea y coloca un nuevo campo en el centro de la tarjeta. Al igual que con los botones, hay varios tipos de campos.

Figura 48. Creación de un campo desde New Control.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

En un campo de texto desbloqueado el usuario tiene todas las capacidades de edición de texto básico.

En los campos LiveCode puede observar y modificar estas características:

- Tipo de texto, incluyendo caracteres no latinos
- Apagar o encender envoltura automática de palabras
- Insertar / borrar, cortar, copiar, pegar texto
- Seleccionar (arrastrar seleccionar o haga doble clic para seleccionar una palabra)
- Establecer estilos comunes, como negrita, cursiva y subrayado
- Seleccionar el tamaño, la fuente y el color de los caracteres en el campo
- Establecer sangrías, relleno, alineación y márgenes por separado para cada párrafo
- Las teclas de flecha de movimiento del cursor por defecto en el texto, pero este comportamiento se puede apagar (Brigham Young University, 2005).

Aun así, los campos LiveCode no tienen las capacidades de un procesador de palabra con todas las funciones:

- No tiene ninguna característica de lujo como la corrección ortográfica, notas al pie, etc.

4.3.7. Gráficos. Otro tipo de objeto de control en LiveCode es un ***gráfico***. Un objeto gráfico es una forma vectorial de tamaño variable que aparece en una tarjeta. Dado que es una forma definida por una fórmula matemática (en lugar de datos '" de mapa de bits", como con imágenes), puede ser modificada y se transforma sin distorsionarse (pixelada), como puede suceder con las imágenes.

Los gráficos pueden ser creados mediante la selección de la herramienta deseada en la paleta de herramientas. Por defecto, las herramientas de gráficos se muestran en la parte inferior de la dicha paleta.

Figura 49. Herramientas para creación de gráficos.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

También puede crear un gráfico por la elección de un tipo de gráfico en el submenu ***New control*** en el menú ***Objet***. Esto crea y coloca un nuevo gráfico en el centro de la tarjeta. Al igual que con otros objetos de control, hay varios tipos de gráficos.

4.3.8. Imágenes. Otro tipo de objeto de control que puede aparecer en las tarjetas es una ***imagen***. Los objetos de imagen se muestran *en mapa de bits* de datos, con más frecuencia fotografías o imágenes de alta calidad producidos en aplicaciones como Photoshop. LiveCode puede mostrar varios de los formatos de imagen más comunes, como JPEG, PNG, GIF y BMP.

Las herramientas de pintura son básicas y se muestran en la parte inferior de la paleta de herramientas, se pueden utilizar para modificar imágenes existentes o crear nuevas imágenes simples.

Figura 50. Herramientas para edición de imágenes.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Las herramientas de pintura son bastante rudimentarias, por lo que cualquier modificación en imágenes complejas se hace mejor en un programa de edición de imágenes como los que se observaron al comienzo.

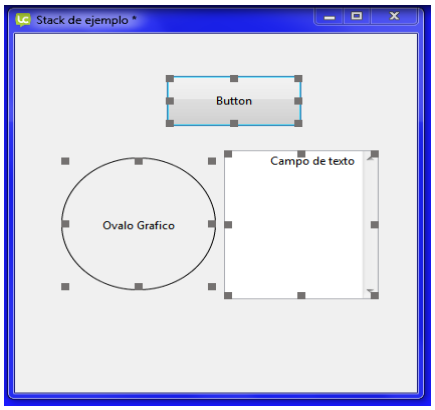
4.3.9. Players. *Los Players* son objetos de control en LiveCode que permiten reproducir en la tarjeta archivos de audio y vídeo compatibles con QuickTime. El *Players* del controlador incorporado puede dar al usuario el control sobre la reproducción de los medios de comunicación.

Figura 51. Herramientas para elementos multimedia (players.)

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.3.10. Trabajando con objetos de control

Figura 52. Identificadores de objetos.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

En la experimentación con objetos de control, se darán cuenta que cada vez que se crea un nuevo control, estará rodeado por unos ocho pequeños cuadros de color gris oscuro. Son las llamadas "aristas" o "asas". Cuando las aristas de un objeto son visibles, significa que el objeto ha sido "seleccionado" con el mouse y puede ser editado. Puede hacer clic y arrastrar cualquiera de estos puntos de control para cambiar el tamaño del objeto.

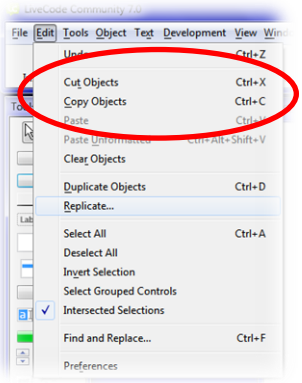
Cuando se selecciona un objeto también se puede cambiar la posición haciendo clic dentro del objeto, y arrastrándolo a una nueva ubicación en la tarjeta. Usted puede "empujar" o deslizar un objeto seleccionado utilizando las teclas de flecha en el teclado.

Puede eliminar un objeto seleccionado pulsando la tecla Retroceso o Supr, o seleccionando el menú ***Edición***> *Borrar.*

Los objetos también se pueden duplicar de varias maneras:

1. Después de seleccionar un objeto, puede elegir Copiar objeto del menú Editar, y luego pegar objetos, aparece un nuevo botón (Nota: Al copiar / pegar un botón, se crea en la misma posición pero en la parte superior del original. Esto puede causar confusión y la creación de más " capas " al no darse cuenta que ya está. Por supuesto, puede utilizar copiar y pegar con atajos de teclado (*Ctrl. + c* y *Ctrl. + v*) para hacer copias de los objetos seleccionados.

Figura 53. Copiar y pegar objetos desde el menú Editar.



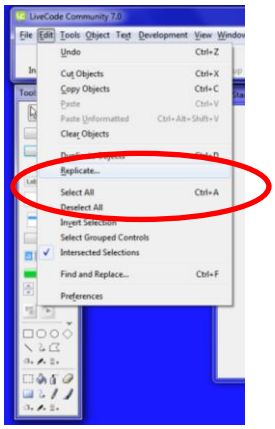
Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

2. *Arrastre Copiar*. Puede copiar un objeto manteniendo pulsada la tecla Control (Windows / Linux) o la tecla Opción (Mac) y haciendo clic y arrastrando un objeto seleccionado.

3. *Duplicado* de comandos en el menú Edición. Duplicar crea una copia, ligeramente desplazada desde la posición del original.

4. *Replicar...* comando en el menú Editar. Replicar le permite crear muchas copias al mismo tiempo, y especificar el desplazamiento (espaciado) de la original.

Figura 54. Duplicar y replicar objetos.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.4. Comandos más usados en la programación con LiveCode

En la mayoría de los casos se trata de manipular los objetos creados, lo que comúnmente se denomina eventos del ratón, y los comandos son usados para escribir las líneas de código, una línea de código es cada renglón de programación dentro de la pantalla ***Code***, que se encuentra en la parte superior izquierda junto al comando ***Inspector***, al abrir ***Code*** cuando esta seleccionado un botón, por defecto sale el comando más usado:

On mouseUp finaliza con **End mouseUp**

Traducido es cuando suelte el botón del mouse haga esto o mientras se pulsa y suelta el objeto en aparatos touch haga esto, Después de la línea **On MouseUp** se escriben los controladores, condicionales, bucles, etc. ejemplo:

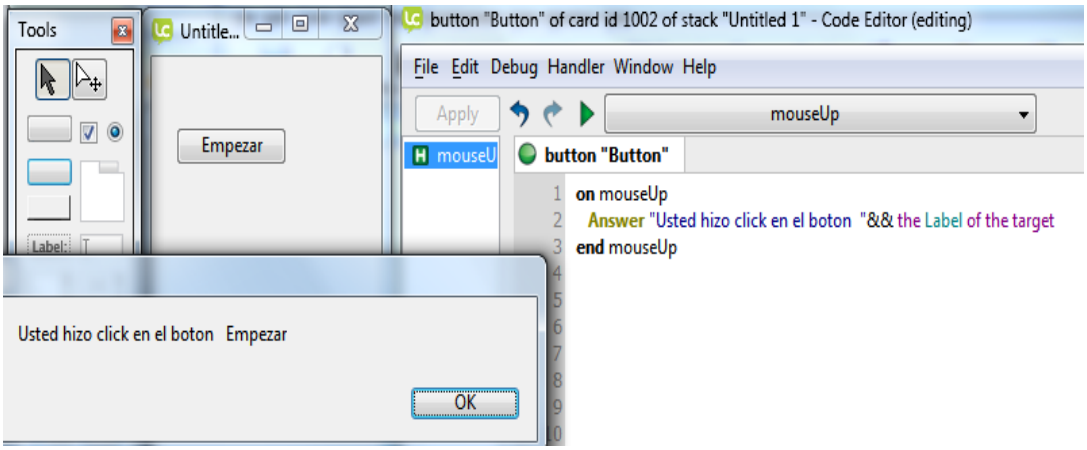
```
on mouseUp
  Answer "Usted hizo click en  "&& the name of the target
end mouseUp
```

Answe*r*: comando de respuesta, genera un cuadro de dialogo en el que se puede escribir una respuesta o seleccionar de las propuestas, utilizado para obtener información o una confirmación del usuario antes de continuar. El usuario debe hacer clic en uno de los botones para cerrar el cuadro de diálogo, ejemplo:

```
on mouseUp
Answer "Deceas ingresar " with "Log In" or "Cancel" titled "Login"
end mouseUp
```

Estructurado así: Después de *Answer* (comando) y entre comillas lo que se desea preguntar, with (con) las opciones que se van a plantear, máximo siete separadas por la palabra or (o) para esta caso dos opciones "Log In" or "Cancel" y opcional el título que va a tener la caja de dialogo para este caso titled "Login"

Figura 55.Comando On mouseUp y Answer.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

On mouseDown finaliza con **End mouseDown**

Igual que en el comando anterior se refiere a eventos que realizaran los objetos, traducido es, mientras este presionado el botón del mouse haga esto o mientras este pulsado el objeto en aparatos touch haga esto, es por ejemplo para desplazar objetos de un lugar a otro de la pantalla manteniéndolos presionados.

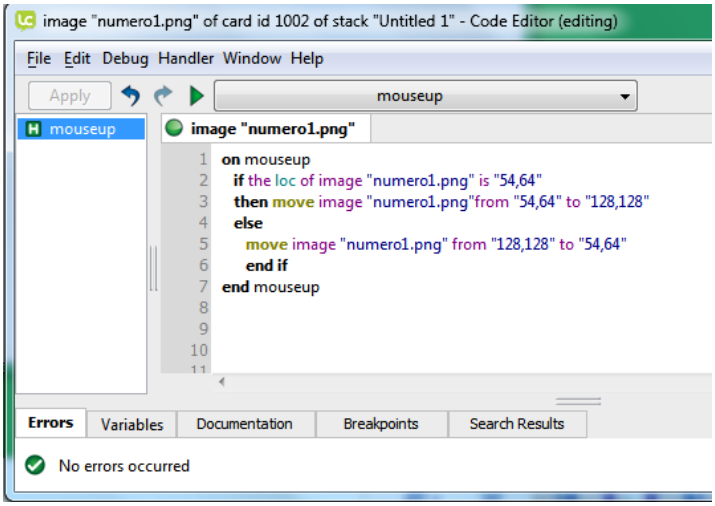
Son importantes de igual forma los siguientes:

Move: Mover o desplazar el objeto desde un lugar hasta otro. Ejemplo: importamos una imagen en una tarjeta y en Code programamos mientras este pulsado el objeto desplazarlo o moverlo de un punto a otro, para ello debemos preguntar dónde se encuentra el objeto.

```
on mouseup
  if the loc of image "numero1.png" is "54,64"
  then move image "numero1.png"from "54,64" to "128,128"
  else
    move image "numero1.png" from "128,128" to "54,64"
    end if
end mouseup
```

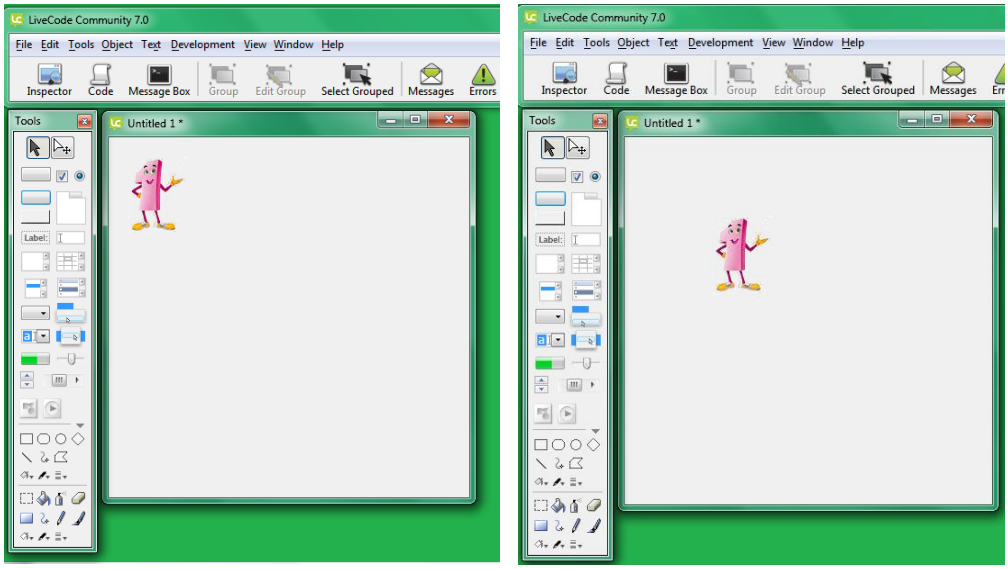
Loc: localización de un objeto, se da por medio de coordenadas ej. (80,120), para saber la posición de un objeto hacer clic en el botón ***Inspector*** y buscar ***Size & Position*** dentro del menú desplegable, se encontraran las coordenadas en ***location***, si se selecciona y mueve el objeto las coordenadas cambian, así se pueden saber las coordenadas deseadas.

Figura 56. Programando movimiento con comandos move y loc.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Figura 57. Cambio de posición al ejecutar el comando.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Put: Es un comando que permite al programador transferir la respuesta de los usuarios (it) en una variable significativa, es por tanto para colocar por ejemplo texto dentro de un campo.

Sintaxis:

put *value* [{before | into | after} *container*]

put value into URL destinationURL

Utilice el comando **put** para cambiar el valor de una variable, poner texto en un campo, poner datos en un archivo, visualizar texto en el cuadro de mensaje, o cargar un archivo en un servidor.

value : Es cualquier expresión que se evalúa o toma cualquier valor.

Container: Especifica el tipo de recipiente en el que un valor se va a colocar. El recipiente es uno de los siguientes:

Campo (field)

Botón (Button)

Imagen (image)

Variable (variable)

Caja de mensaje (message box)

Dirección de internet (URL)

Por ejemplo:

```
on mouseUp
  Ask "Escribe tu nombre  "
  put it into field "Salida"
end mouseUp
```

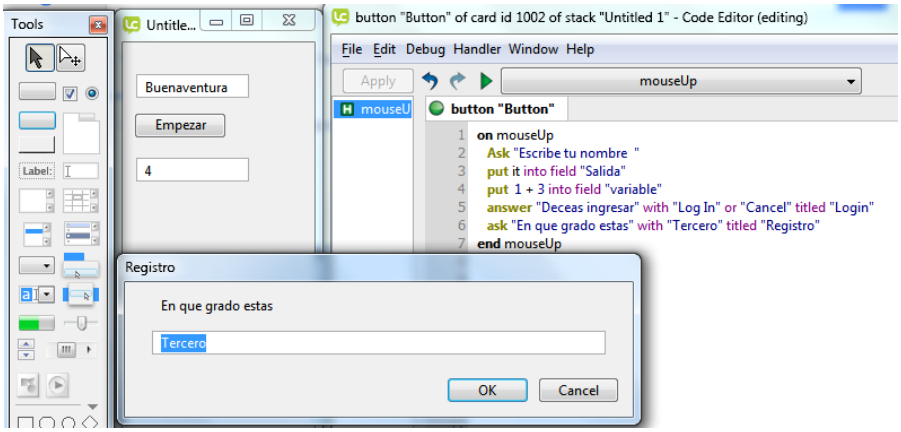
Ask: Utilice el comando preguntar cuando un controlador necesita obtener información del usuario antes de continuar.

Sintaxis:

Ask "En qué grado estas" with "Tercero" titled "Registro"

Estructurado así: Similar al comando Answer, después de Ask (comando), entre comillas lo que se desea preguntar, para el caso “En qué grado estas?”, with (con) opción que se va a plantear aparece por defecto, ésta aparecerá como texto editable para el ejemplo se asigna “Tercero” y opcional el título que va a tener la caja de dialogo para este caso titled es "Registro". La línea escrita para mostrar la sintaxis muestra una salida como lo indica la figura.

Figura 58. Cuadro de dialogo para el comando Ask.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Hide: Ocultar o esconder, un botón, un campo una imagen o cualquier objeto dentro de la tarjeta. Ejemplo: oculte el botón numero 14

hide button 14

hide field "New" with visual effect dissolve

Show: Mostrar, es por lo tanto lo contrario a ***Hide***, muestre una fotografía un objeto o un resultado

show stack "project1"

show image "picture" with visual effect barn door open

Go: Ir a, la mayor de las veces para desplazamientos entre tarjetas, ir a la tarjeta “#” o de nombre “X”, cuando por ejemplo se abre el menú con las opciones jugar y salir, al presionar jugar va a otra tarjeta que tiene otro menú, así que hay que presionar botones para ir a otra tarjeta, por ejemplo:

On mouseUp

Go to next card

End mouseUp

Una forma de obtener la repetición de una o más líneas de código, es mediante el uso de un bucle. Los dos tipos principales de bucle son: un bucle fijo y un bucle condicional.

El bucle fijo es **Repeat with loop** puede utilizarse para repetir la acción de una línea de código tantas veces como el usuario establece. En el siguiente ejemplo, el bucle se fija en la repetición del mensaje "Introduzca un número" para solicitar 4 números.

Repeat with loop = 1 a 4

Put "Introduzca un número"

End Repeat

En bucle condicional es **Repeat until loop** puede utilizarse para repetir una línea de código hasta que cierta condición se cumple. En el siguiente ejemplo, el bucle no terminará hasta que el usuario entra un número válido entre 0 y 100. Esta es la condición.

Answer "Por favor, introduzca un número entre 0 y 100."

Put it into Número

Repeat until Número> = 0 and Número <= 100

Answer "Número no válido. Por favor, vuelva a introducir un número entre 0 y 100."

Put it into Número

End Repeat

Idle: Siempre (haga algo) por ejemplo siempre coloque la hora en un campo.

On Idle

Put the time into filed “Hora”

End Idle

4.4.1. Variables Globales: Una variable global es una a la que se puede acceder y alterar desde cualquier parte del programa, incluso desde otro script / evento siempre que sea declarada en el comienzo mismo.

Las variables globales se deben utilizar siempre con cuidado ya que sus valores pueden cambiar accidentalmente si el programador olvida que ya los ha utilizado en otra subrutina. El siguiente ejemplo muestra la creación de una serie de variables globales en LiveCode:

//Configuración de las variables globales que se utilizan en este evento

Global Largo_habitación, Ancho_habitación, Altura_habitación, Volumen_habitación

En el fragmento de código anterior cuatro variables globales se han creado. Estas variables se pueden utilizar en cualquier subrutina y en cualquier caso.

4.4.2. Variables locales: Una variable local es aquella que sólo existe dentro de una subrutina, función o procedimiento.

Las variables locales se crean cuando la subrutina se llama (ejecución) y luego se destruyen cuando termina la subrutina. No se puede acceder o asignar un valor a excepción dentro de esa subrutina.

On Obtener_Nombre_Usuario

Local Presionar_tecla

REPEAT until Presionar_tecla = "S" or Presionar_tecla = "s"

Ask "Por favor escribe tu nombre"

Put it into Nombre_de_Persona

```
Ask "Está bien escrito tu nombre? (S o s si está correcto)"
IF the result = "Cancel" THEN exit to top
Put it into Presionar_tecla
END REPEAT
End Obtener_Nombre_Usuario
```

En la subrutina Get_Users_Name, se crea la variable local KEY_PRESSED. Esta variable local es única para esta subrutina y no se puede utilizar en cualquier otra subrutina. La ventaja de utilizar variables locales es que evita que sean utilizadas en otras partes del programa y, posiblemente, cambiar accidentalmente su contenido.

Substrac: restar, Resta un número de un recipiente y coloca el valor resultante en el recipiente.

```
subtract 5 from 10
put "10,10" into tPoint
subtract 5 from item 2 of tPoint -- subtracts 5 from the second item
# RESULT
# tPoint = 10,5
```

Add: sumar, Agrega un número a un recipiente y se coloca el valor resultante en el recipiente. Utilice el mandato add para añadir un número a un contenedor o una parte de un recipiente

```
add 7 to field 1
add field "New" to tSummaryOfInventory
add (qty * price) to last line of tOrder
```

Beep: sonido

Set: Cambiar atributos (alto, nombre, posición, etc.) Asigna un valor a una propiedad.

Utilice el comando set para cambiar la configuración de una propiedad o propiedad personalizada .

Sintaxis.

```
set the textFont of button "next" to "Arial"
set the cursor to watch
set the backgroundcolor of graphic "background" to green
```

4.5. APRENDIENDO A PROGRAMAR APLICACIONES BÁSICAS PARA LA ENSEÑANZA Y DESARROLLO DE HABILIDADES Y DESTREZAS MATEMÁTICAS

Es necesario familiarizar las órdenes más usadas, por consiguiente se empezará con programas sencillos y se irá aumentando la complejidad a medida que se avance.

Dentro de los ejercicios propuestos no se encuentra ninguno para la enseñanza sistemática de un concepto o elemento matemático, pero se pretende poner las bases para que con la puesta en práctica de estas actividades, se pueda mesclar los conocimientos adquiridos en cada ejemplo y se logre desarrollar una aplicación a la medida de la necesidad particular de cada docente y cada curso.

4.5.1 Actividad 1. Para tomar confianza con la interface de LiveCode se debe explorar con los objetos, por tanto, Abrir LiveCode, crear un nuevo Mainstack, agregar una imagen, un botón y un gráfico, cambiar los nombres, tamaños y colores.

4.5.2 Actividad 2. Crear un nuevo proyecto (Mainstack), construir cuatro cartas (tarjetas), hacer una portada (importar imagen), colocar tres botones (jugar, opciones y salir).programar el botón jugar para que cuando se pulse vaya a otra carta donde encuentre el menú de juegos (sumar, restar, multiplicar y dividir), y que por ultimo al pulsar sobre cualquier juego vaya a la tarjeta donde encuentre los números de 0 a 9.

El desarrollo de esta actividad es como sigue: abrir LiveCode y crear la primera tarjeta, para esto se hace click en file >new Mainstack, esto crea la primera carta, se pude modificar el tamaño tomando la carta de las aristas, luego hacemos click en File > Import As control > Image file esto permite buscar la foto de portada, acomodarla a el tamaño de la carta y pulsar en Inspector >menú desplegable > Size &Positions >pulsar el cuadro lock size and positions, esto para que el tamaño quede estático, luego tomar lo botones de la paleta de herramientas y arrastrarlos hasta la carta y darles nombre en Inspector> Name y buena ubicación. En este momento debe estar así:

Figura 59. Portada de la actividad.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Para crea la segunda carta hacer click en *object* > *new card* y seguir nuevamente los pasos anteriores pero creando cinco botones. Para las otras dos cartas se siguen nuevamente los pasos anteriores. Ademas se introducirán las imágenes de los números uno por uno en la tercera carta desde file> importas control> image file, una vez creados en la tarjeta tres, se pueden copiar a las demás tarjetas, de tal manera que las tarjetas queden similar a las fotografías siguientes**.**

Figura 60. Menú de opciones para jugar.

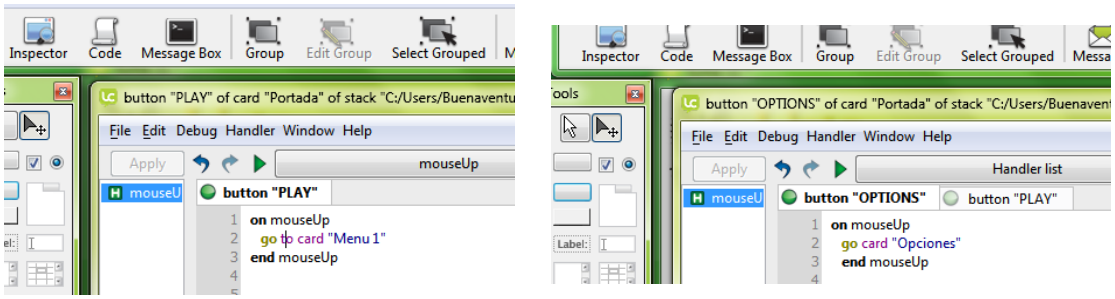
Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Figura 61. Portada del juego suma.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

El proceso de programación será simple, para programar los botones basta seleccionarlos y pulsar el botón Code, en el escribir el comando Go to card (ir a la carta) y colocar el nombre de la tarjeta (entre comillas y exactamente igual al de la tarjeta) a la que ira, así:

Figura 62. Programación de botones.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

4.5.3 Creación de calculadora. (RunRev, 2010) El siguiente ejercicio se realiza con el ánimo de mostrar las agrupaciones de objetos en la creación de aplicaciones más que para la enseñanza, es un ejercicio de práctica de programación, se ejercitan los comandos y se aplican propiedades de agrupación.

Primero, crear un nuevo proyecto, hacer una portada e ingresar los botones para los números y para las operaciones, ingresar un campo para las respuestas y adornar para que se vea vistoso, aunque como es un ejercicio que persigue se observen propiedades de agrupación puede dejarse simple.

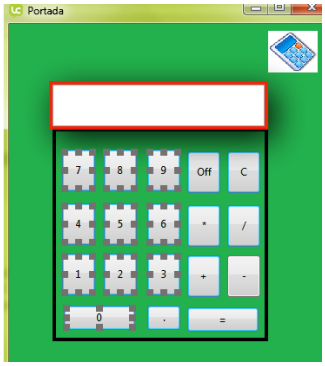
Al crear el tablero, después de colocar los nombres a los botones, puede quedar así.

Figura 63. Portada de calculadora.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

La finalidad es que se programe una sola vez para varios objetos, para este caso para los 10 números y el punto decimal se realiza una sola programación. Para esto seleccionar los números, presionando la tecla mayúscula seleccionamos cada uno de los botones, se les observará las aristas a todos los botones.

Figura 64. Selección de botones para agrupar.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

En la barra de herramientas hacer click en el botón ***select grouped*** esto formará un grupo, en el inspector cambiar el nombre a Números, hacer lo mismo con los operadores (+-*/), quedarán dos grupos, cuando pulsamos sobre cualquier elemento se observan las aristas del grupo.

Figura 65. Dos grupos seleccionados.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

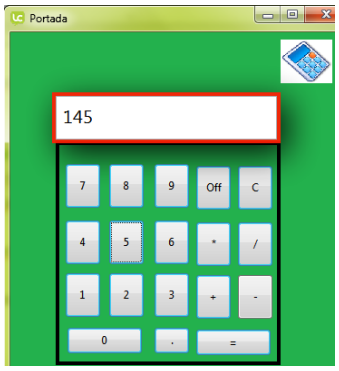
Seleccionar el grupo números y hacer click en el botón code de la barra de herramientas en la ventana emergente escribir.

```
    //nombrar las variable que se van a usar separadas por comas.
global esta_tecla, total_acumulado, total_acumulado_actual, este_operador, hacer_evaluacion


//cuando se pulse una tecla de este grupo.
On MouseUp


//ponga la etiqueta de esta tarjeta en esta variable, ponga una variable en otra variable
  put the label of target into esta_tecla
  put the total_acumulado of this card into total_acumulado_actual


//si la última variable es vacía o es “=”
  if total_acumulado_actual is empty or the last char of total_acumulado_actual is "="
  then


//si la tecla pulsada es punto decimal
    if esta_tecla is "." then put "0." into esta_tecla
    put esta_tecla into field "Resultado"
    set the total_acumulado of this card to esta_tecla
  else


//si el último carácter es un operador
    if last char of total_acumulado_actual is "+" or "-" or "*" or "/"
      then


//ponga el valor en el campo de respuesta
      put esta_tecla into field "Resultado"
    else
```

```
        put esta_tecla into field "Resultado"
      end if

//cambie el total y guárdelo en otra variable
      set the total_acumulado of this card to total_acumulado_actual&esta_tecla
  put total_acumulado_actual&esta_tecla  into field "Resultado"
      end if
end MouseUp
```

Esto permite al pulsar el botón de modo ejecutar, escribir los números así.

Figura 66. Escribiendo números de prueba.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Seleccionar ahora el grupo operadores, pulsar en code y escribir.

```
//nombre las mismas variables
global esta_tecla, total_acumulado, total_acumulado_actual, este_operador, hacer_evaluacion

//cuando se pulse cualquier botón del grupo
On MouseUp
  put the label of the target into esta_tecla
  put the total_acumulado of this card into total_acumulado_actual
```

```
//si la primera tecla pulsada es un operador reporte error
  if  total_acumulado_actual is empty or last char of total_acumulado_actual  is "+" or "-" or "*" or "/" then reporte_error
  repeat for each char este_operador in "+-*/"

//si hay numeros y operadores, hacer la evaluacion
    if  total_acumulado_actual  contains  este_operador  then  put  true  into hacer_evaluacion
    end repeat
  if hacer_evaluacion then

//ponga el valor en el campo de respuesta
    put the value of total_acumulado_actual into field "Resultado"
    set the total_acumulado of this card to the value of total_acumulado_actual & "="
 else  set the total_acumulado of this card to total_acumulado_actual&esta_tecla
end MouseUp
```

Esto permite con el comando ejecutar, hacer operaciones así.

Figura 67. Operaciones y resultados.

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Están incluidos casi todos los botones falta el de borrar, seleccionarlo y pulsar en code y escribir.

```
//cuando se presione
on mouseUp

//cambie el valor de la variable  a vacio
  set the total_acumulado of this card to empty

//ponga esto en el campo de respuesta
  put empty into field "Resultado"
end mouseUp.
```

Esta terminado el ejercicio de agrupación, presionar en el botón de ejecutar y poner a correr la aplicación para probarla.

En la mayoría de aplicaciones que se realicen será necesario agrupar botones para minimizar líneas de programación, esto sirve también cuando los botones no están en la misma tarjeta, por ejemplo, en el menú de varios tableros el botón regresar o salir, o cuando se tiene una cantidad de elementos que hacen una misma función.

4.5.4. Volumen de una habitación. (White) Esta aplicación presenta las opciones de interactuar con ella al realizar preguntas, y permitir que se ingresen datos, tiene como fin mostrar las entradas de datos en aplicaciones mas complejas.

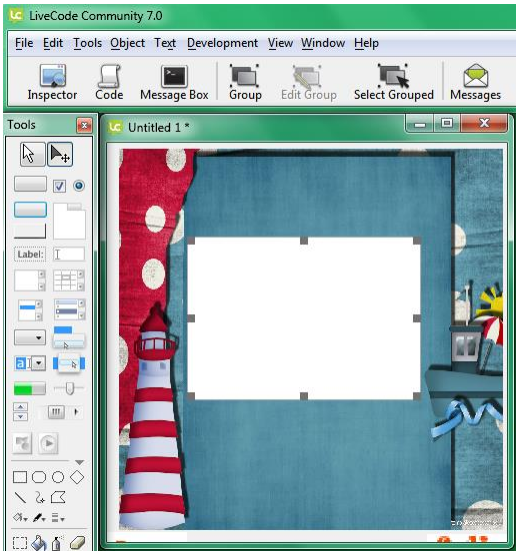
Para iniciar crear la interface del juego, lo primero será crear la portada del ejercicio, es importante que sea llamativa al estudiante. Ir a file > new Mainstack esto creara la primera carta, luego ir a file > Import As Control > Image file.

Figura 68. Imagen Portada de Volumen de una Habitación

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

También es necesario crear un campo donde se colocará la respuesta. Ir a la paleta de herramientas y seleccionar el gráfico del campo, manteniéndolo pulsado se arrastra hasta la portada, se toma de los vértices para darle las dimensiones consideradas apropiadas.

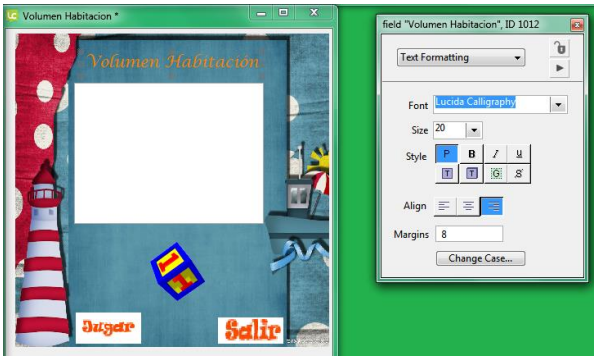
Figura 69. Imagen de la creación del campo de respuesta



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Ahora se crean los botones de iniciar, de limpiar y de salir, puede ser tomándolo arrastrado desde la paleta de herramientas, ponerle el nombre (limpiar) o puede ser una foto, que será el caso de jugar y salir, además la portada puede ser decorada con un objeto vistoso o llamativo, en este caso un Gif de un cubo. También se colocará el nombre al ejercicio, arrastrando el campo label hasta el lugar que tentativamente ocupe en la portada, haciendo clic en *Inspector > menú desplegable Basic Properties > Contents* borrar label y escribir el nombre, para el caso Volumen Habitación. Posteriormente se pueden cambiar las propiedades del texto en menu desplegable > *Text Formating*, aquí se cambia lo que se desee como el tipo de fuente, el tamaño, el estilo (Negrilla, cursiva, etc), la alineacion (derecha, centrado, izquierda) y el margen.

Figura 70. Cambio de propiedades en un texto



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

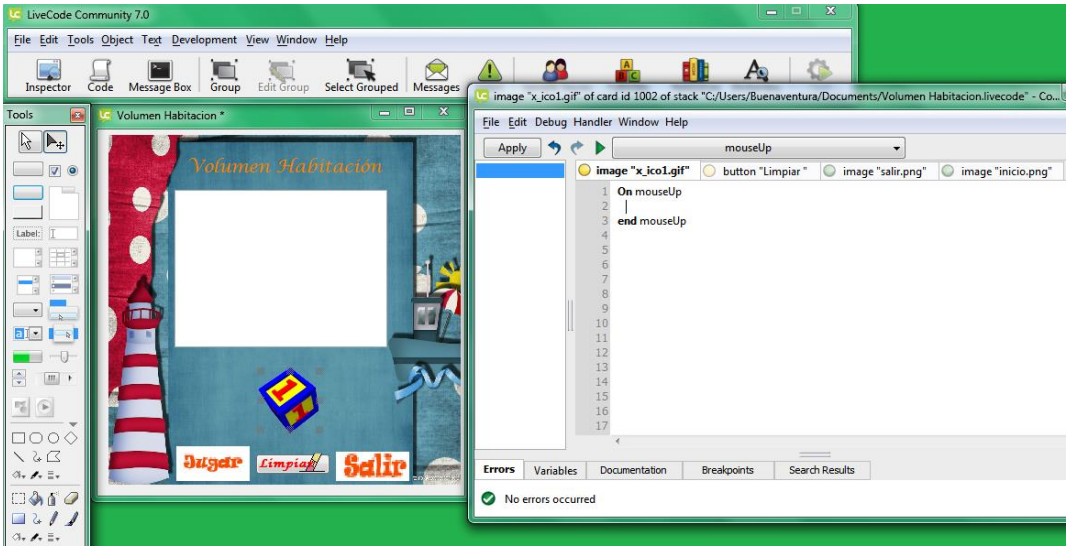
Terminando de adecuar la portada puede quedar así:

Figura 71. Portada final para Volumen Habitación

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Ahora se procede a programar los objetos que contiene el stack, pulsando el botón Jugar para seleccionarlo, luego hacer click en el botón superior Code y aparece una ventana emergente con el comando por defecto On mouseUp y con color naranja el círculo del objeto (botón, campo o imagen) que se está programando, además de los que ya se han programado o hemos seleccionado para programar así:

Figura 72. Pantalla (Code) para programar una imagen.



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Estando seleccionado el botón Jugar se escribe en las líneas de la pantalla *Code*:

//Configuración de las variables globales que se utilizan en este evento

Global Largo_habitación, Ancho_habitación, Altura_habitación, Volumen_habitación

On MouseUp

Setup_Variables

Get_Medidas_habitación

Calculate_Volumen_habitación

Display_Volumen_habitación

End MouseUp

On Setup_Variables

// Configuracion de las variables a 0.00 (tipos de datos reales)

Put 0.00 into Largo_habitación

Put 0.00 into Ancho_habitación

Put 0.00 into Altura_habitación

Put 0.00 into Volumen_habitación

End Setup_Variables

Hasta aquí solo se ha dado nombre a las variables que se necesitan para calcular cualquier volumen, asegurando que empiezan en cero.

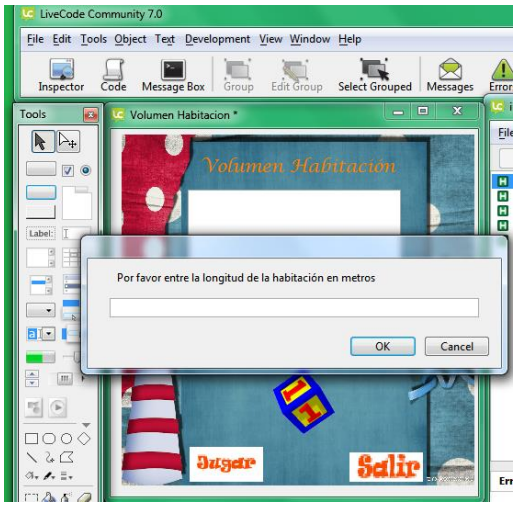
Ahora empezarán las preguntas para que el usuario responda entrando datos, serán estos (datos) los que además hagan operaciones y generen una respuesta.

On Get_Medidas_habitación

// Obtener las medidas de la habitacion del usuario

Ask "Por favor entre la longitud de la habitación en metros"

Figura 73. Pantalla emergente para la ejecución del comando Ask

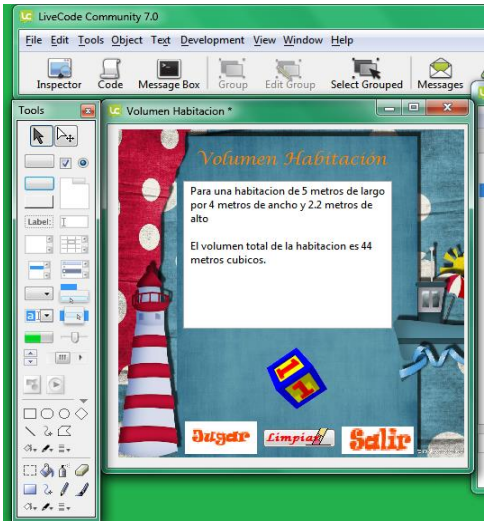


Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Y hará lo mismo para todo comando ***Ask*** en el programa

```
Put it into Largo_habitación
Ask "Por favor entre el ancho de la habitación en metros"
Put it into Ancho_habitación
Ask "Por favor entre la altura de la habitación en metros"
Put it into Altura_habitación
End Get_Medidas_habitación
On Calculate_Volumen_habitación

// Calcular el volumen de la habitación
    Put (Largo_habitación * Ancho_habitación * Altura_habitación) into Volumen_habitación

//Publicando respuesta
    Put "Para una habitacion de " &Largo_habitación &" metros de largo por " &Ancho_habitación & " metros de ancho y " &Altura_habitación & " metros de alto" into Line 1 of field "Salida"
End Calculate_Volumen_habitación
On Display_Volumen_habitación
```

Figura 74. Salida de datos en el campo indicado



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

```
// Indicar el volume de la habitacion usando el resultado de la variable Volumen_habitacion

Put "El volumen total de la habitacion es " & Volumen_habitación & " metros cubicos." into  line 3 of field "Salida"
End Display_Volumen_habitación
```

Estando seleccionado el botón Salir se escribe en las líneas de la pantalla *Code*:

```
//cuando se presione es botón preguntar

on mouseUp
  answer " Seguro quieres Salir" with "Si" and "No"
  if it is "Si"
  then

// con respuesta Si entonces salir del juego

    quit
    end if
    if it is "No"
    then

//Con respuesta negativa entonces vuelva a la portada
      go card "Portada"
      End If
end mouseUp
```

Se ha finalizado la programación ahora se hace click en la flecha de la paleta de herramientas para correr la aplicación, si hay errores aparecen inmediatamente en la casilla de abajo y se mostrarán los demás indicadores de error, corregir y a disfrutar del juego.

4.5.5. Tres Adiciones. (White) En esta App, hay mucho ejercicio mental por parte del estudiante ya que solo se presentarán datos y el estudiante calculara resultados, se necesita la entrada de datos para recoger la respuesta del estudiante, a la vez que se valora su desempeño y se anima a que termine el ejercicio planteado, será útil para observar los comandos de pregunta, colocar datos en variables y mostrar los resultados.

Los pasos de creación de portada e niterface son los mismos del ejercicio anterior.

Ir a file > new Mainstack esto creara la primera carta, luego ir a file > Import As Control > Image file, crear un campo donde se colocará la respuesta. Ir a la paleta de herramientas y seleccionar el gráfico del campo, ahora se crean los botones de iniciar, de limpiar y de salir.

Estando seleccionado el botón Jugar se escribe en las líneas de la pantalla *Code*:

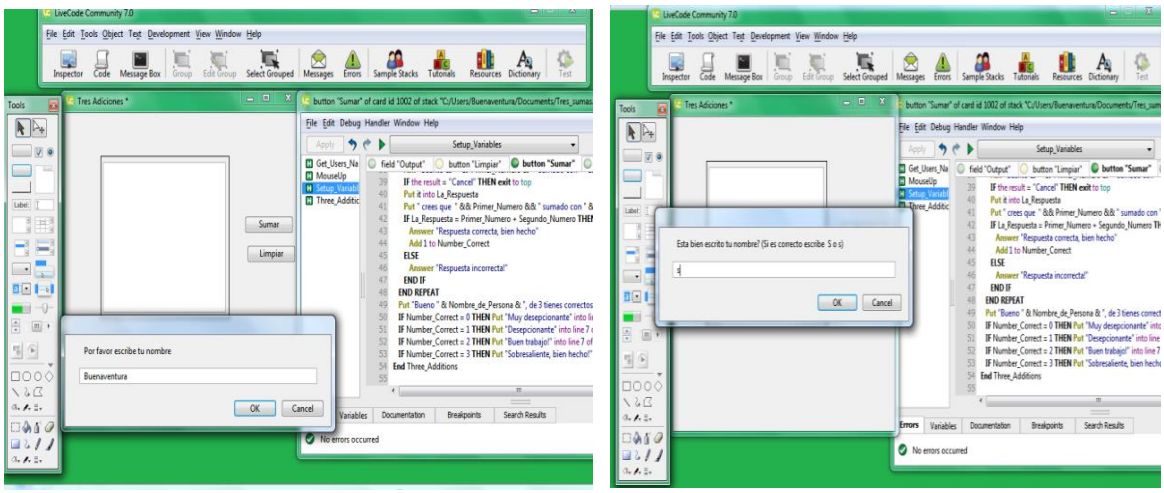
```
// Configuración de las variables globales que se utilizan en este evento

Global Nombre_de_Persona,Primer_Numero, Segundo_Numero, La_Respuesta

//(cuando se presione y se suelte el botón)

On MouseUp
Configurar_Variables
Obtener_Nombre_Usuario
Operaciones_Resultados
End MouseUp

// Configuración de todas las variables en NULO o 0
```

```
On Configurar_Variables
Put "" into Nombre_de_Persona
Put 0 into Primer_Numero
Put 0 into Segundo_Numero
Put 0 into La_Respuesta
End Configurar_Variables
```

Se ha implementado un aparte para solicitar el nombre del jugador para escribirlo en el cuadro de respuesta junto con la evaluación y a su vez para poder observar los comandos de bucle y de variable local.

```
On Obtener_Nombre_Usuario

// Configuración de una variable local para comprobar si una tecla ha sido presionada

Local Presionar_tecla

// Un bucle es utilizado para preguntar al usuario si esta correcto el nombre que introducen (repita hasta que)
// ingreso de S o s indica que esta correcto y se sale del bucle.

REPEAT until Presionar_tecla = "S" or  Presionar_tecla = "s"
Ask "Por favor escribe tu nombre"
Put it into Nombre_de_Persona
Ask "Está bien escrito tu nombre? (S o s si está correcto)"
IF the result = "Cancel" THEN exit to top
Put it into Presionar_tecla
END REPEAT
End Obtener_Nombre_Usuario
```

Figura 75. Cuadro de dialogo para el comando Ask preguntando por el nombre



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

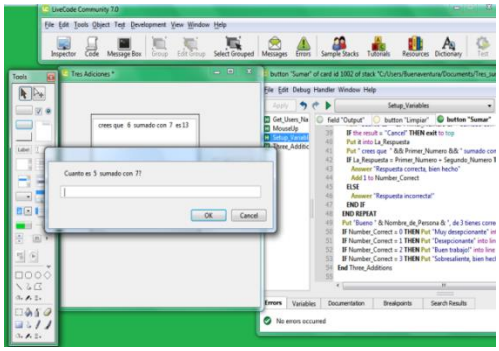
```
On Operaciones_Resultados

// Configuración de una variable local para realizar un seguimiento del número de sumas correctas

Local Numero_Correcto
Put 0 into Numero_Correcto

// Este bucle fijo hace al usuario tres preguntas aritméticas básicas

REPEAT with loop = 1 to 3
// Los números son generados al  azar con la variable random el número dentro del paréntesis indica el límite que proponemos para este caso entre 1 y 20

Put Random (20) into Primer_Numero
Put Random (20) into Segundo_Numero
Ask "Cuanto es  " & Primer_Numero & " sumado con " & Segundo_Numero & "?"
```

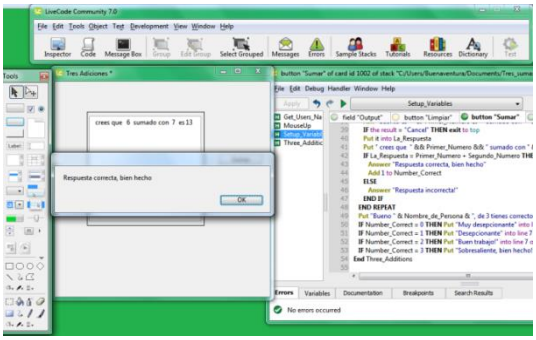
```
IF the result = "Cancel" THEN exit to top
Put it into La_Respuesta
```

Se escriben los números propuestos en el campo respuesta para que la evaluación sea más crítica y se pueda certificar como verdadera o falsa y el estudiante pueda hacer nuevamente el cálculo aunque no pueda volver a responder.

```
Put " crees que  " & Primer_Numero & " sumado con " & Segundo_Numero & " es " & La_Respuesta &  into line loop of  the field "Salida" // En la misma línea del bucle
IF La_Respuesta = Primer_Numero + Segundo_Numero THEN
Answer "Respuesta correcta, bien hecho"
Add 1 to Numero_Correcto
ELSE
Answer "Respuesta incorrecta!"
END IF
END REPEAT
```

Figura 76. Cuadro de dialogo planteando sumas y dando una valoración





Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Ahora se plantean las salidas (mostrar) de las valoraciones

Put "Bueno " & Nombre_de_Persona & ", de 3 repuestas contestadas tienes correctos " & Numero_Correcto & " Tu desempeño es " into line 5 of field " Salida"

// En la misma línea

IF Numero_Correcto = 0 **THEN** **Put** "Insuficiente" into line 7 of field " Salida "

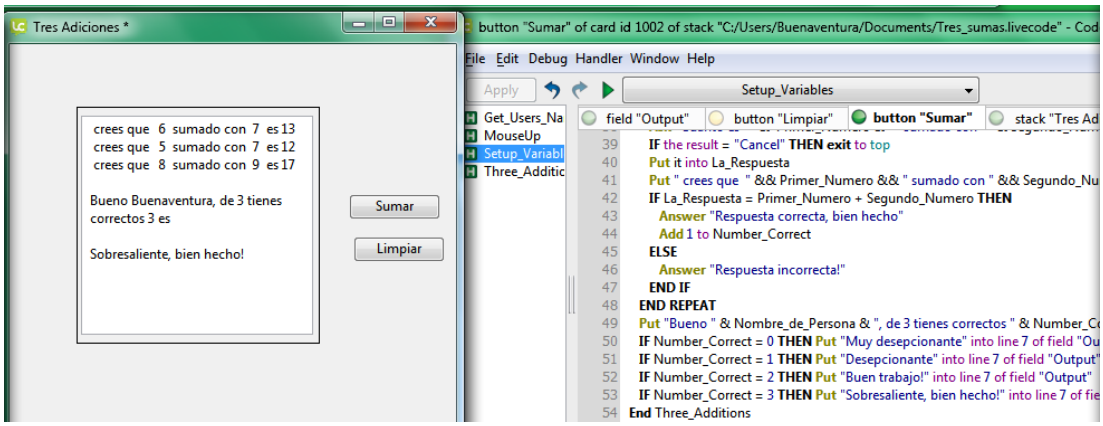
IF Numero_Correcto = 1 **THEN** **Put** "Regular" into line 7 of field " Salida "

IF Numero_Correcto = 2 **THEN** **Put** "Buen trabajo!" into line 7 of field " Salida "

IF Numero_Correcto = 3 **THEN** **Put** "Sobresaliente, bien hecho!" into line 7 of field " Salida "

End Operaciones_Resultados

Figura 77. Presentación y valoración final



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Se finaliza con una valoración cualitativa.

Programación del botón Limpiar. Se selecciona y en la ventana *Cod*e se escribe:

// cuando se presione el botón pregunte si está seguro de salir

on mouseUp

Answer "Seguro quieres borrar" with "Si " or "No "

```
    If  it is "Si "   then

  // Si la respuesta es afirmativa entonces borre las líneas # del lugar de respuestas

     delete line 1 to 7 of field "Salida"
    end if
  end mouseUp
```

Está terminada la programación ahora a ejecutar la aplicación.

4.5.6. Números Enteros. Hasta aquí se han presentado aplicaciones estáticas de preguntas y respuestas, ahora se observará una app dinámica donde los objetos requieren desplazamientos, tendrá también un solo Mainstack y contendrá números (0 a 9) y símbolos (+-*/) que se desplazan para formar las operaciones.

La creación de portada sigue los pasos de los ejercicios anteriores, quedara similar a:

Figura 78. Portada Números Enteros

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Todos los objetos de esta aplicación se programan igual excepto el igual que contendrá las operaciones, por lo que basta con programar uno y copiar el código a los otros elementos. Hay que crear un campo de texto igual al tablero, pero ponerlo en invisible en las propiedades de Inspector. De la misma manera crear el campo orden y el campo equivalentes.

En la ventana *Code* se escribe:

```
//Mientras este presionado el botón desplace una imagen del número hasta el campo invisible del tablero

On MouseDown
  global elementos
  send mouseup to the button "Ok"
  put the loc of me into Origen

//ponga la localización de los objetos en el origen y cree un botón (el que se desplaza)

  create button
  add 1 to elementos

//enumere los elementos y póngalos en una variable Nombre

  put "Objeto_"&elementos into Nombre
  set the name of last button to Nombre
  put the short name of me into Forma
  set the icon of button Nombre to Forma
  set the showname of button Nombre to false
  set the rect of button Nombre to the rect of me
  set the opaque of button Nombre to false
  set the showborder of button Nombre to false
```

```
//Cambie las propiedades de icono nombre y forma y displace mientras este presionado

    repeat until the mouse is up
      set the loc of button Nombre to the mouseloc
    end repeat
    put the rectangle of btn "tablero" into Tablero

//si la localización no está dentro del tablero devuélvalo al origen y borre el botón creado

    if the loc of button Nombre is not within Tablero
    then
      move button Nombre to Origen
      delete button Nombre
    end if
  end MouseDown
```

Lo anterior permite desplazar la imagen de un objeto (número) hasta el tablero sin que el número real se desplace

Figura 79. Desplazamiento de objetos

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

En el boton igual el codigo es:

```
On MouseUp
  send mouseup to button "Análisis"

//envíe el mouse al botón análisis y muestre la ecuación y su resultado

  Answer fld "Ecuación"&"="&the value of field "Ecuación"
end MouseUp
```

En el botón análisis se escribe:

```
on mouseUp
  global elementos

//el campo ecuación  en vacío

 put empty into fld "Ecuación"
  repeat with i = 1 to elementos

//en una variable temporal

     put "Objeto_"&i into Temporal
     put the icon of button Temporal into item 1 of line i of fld "Orden"

//ponga los objetos en orden

     put the icon of button Temporal into valor
      set the itemdel to "@"

//utilice el valor de los objetos
      repeat with j = 1 to the number of lines of fld "equivalentes"
        if item 2 of line j of fld "equivalentes" is valor
```

```
            then
              put item 1 of line j of fld "equivalentes" into resultado
              end if
            end repeat
            set the itemdel to ","

    //Cambie el valor si tiene coma, haga operaciones y entregue el valor en el campo resultado

        put resultado into item 3 of line i of field "Orden"
        put the item 1 of the loc of button Temporal into item 2 of line i of fld "Orden"
      end repeat
      sort lines of fld "Orden" by item 2 of each
      repeat with i = 1 to the number of lines of fld "Orden"
        put fld "Ecuación"&item 3 of line i of fld "Orden" into fld "Ecuación"
      end repeat
    end mouseUp
```

Terminado el código, probar la aplicación.

Figura 80. Proponer operaciones

Figura 81. Entrega de resultados

Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Con estas directrices básicas se puede empezar en el mundo de la programación de aplicaciones para dispositivos móviles y depende de la creatividad de cada docente (Programador, Innovador) y de la necesidad de incorporar nuevos objetos de control en las aplicaciones que realice, de acuerdo a las situaciones que plantea reforzar o enseñar, en los que por ejemplo, coloque tiempo cronometrado de respuesta, que la aplicación genere las operaciones, que el estudiante las solucione y entregue respuestas y hasta aplicaciones que se van desarrollando por niveles de aprendizaje, en el que se desbloquee cada nivel a medida que desarrolla actividades y aumenta la complejidad de los mismos. En este caso simplemente se condiciona cada nivel al desarrollo de todas las actividades propuestas para que se pueda avanzar. En general todo depende de la situación, de la disponibilidad de tiempo y de los recursos pero que esto no desanime en el camino integrador - innovador…

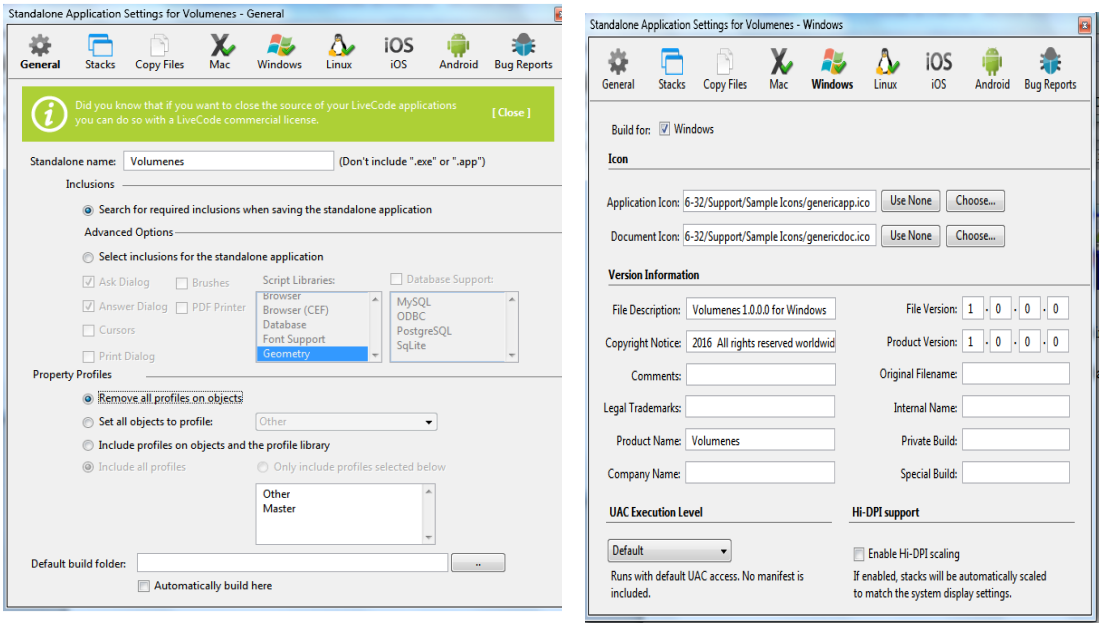
4.6. CREACIÓN DE APLICACIÓN INDEPENDIENTE

Cuando la aplicación está terminada y lista para ser compilada, para que se ejecute en los diferentes equipos, se debemos seguir los siguientes pasos:

Ingresar para configurar los parámetros con los que se guardará la aplicación en File > Standalone Application Setting… Click

Esto envía a una ventana emergente que coloca parámetros por defecto, entre los cuales está el nombre de la aplicación, que toma el nombre del Stack , además aparece en flecha verde los sistemas operativos para los que se generará el instalador de la aplicación, por defecto será para Mac, Windows y Linux, esta ventana es:

Figura 82. Ventanas para crear aplicación independiente



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

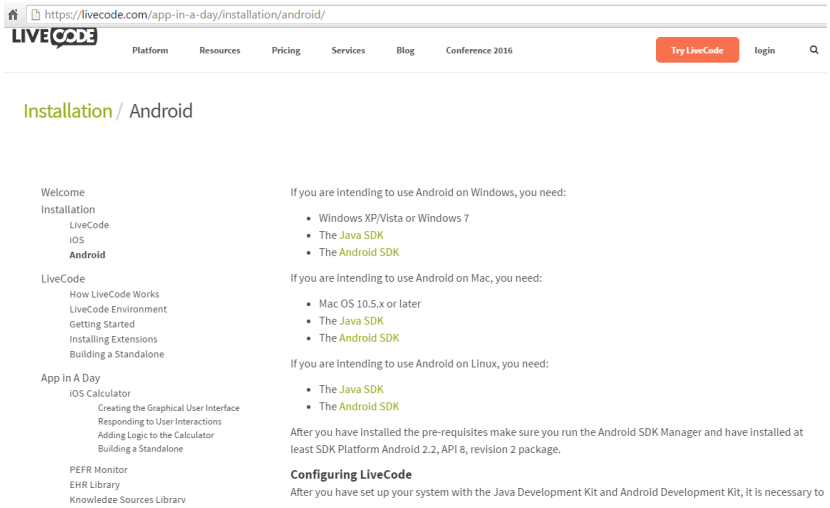
Después de completar la configuración hacer clik en File>Save as Standalone Aplication… ubicar la carpeta de destino, aceptar y se crean los archivos ejecutables para los diferentes sistemas operativos. Para generar los instaladores para Android e iOS se debe tener en cuenta unos pasos adicionales que incluyen la instalación de programas de complemento, esto se describe a continuación.

4.6.1. Para Android.

Se deben instalar dos programas auxiliares que se llaman SDK de Android y JDK de Java que se pueden descargar de la página de LiveCode:

https://livecode.com/app-in-a-day/installation/android/

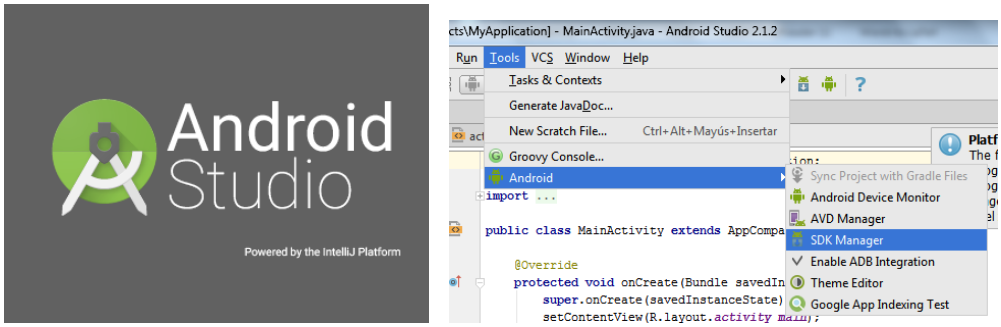
Figura 83. Página de descarga del SDK y el JDK



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Después de instalados abrir el Android Studio y en herramientas (Tools) buscar el SDK Manager de Android pulsar sobre él y descargar los paquetes de las versiones de Android que se van a usar (obligatorio descargar Froyo, conviene descargar del 4.0 en adelante).

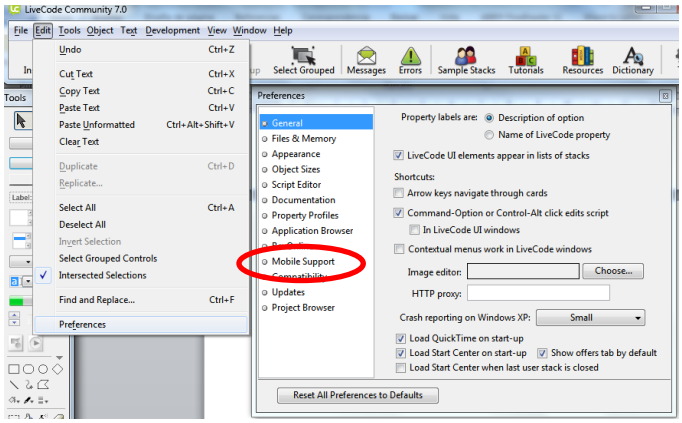
Figura 84. Ubicación del sdk manager



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Posteriormente se debe ir a LiveCode, hacer click en Edit > preferences, esto abre una ventana emergente así:

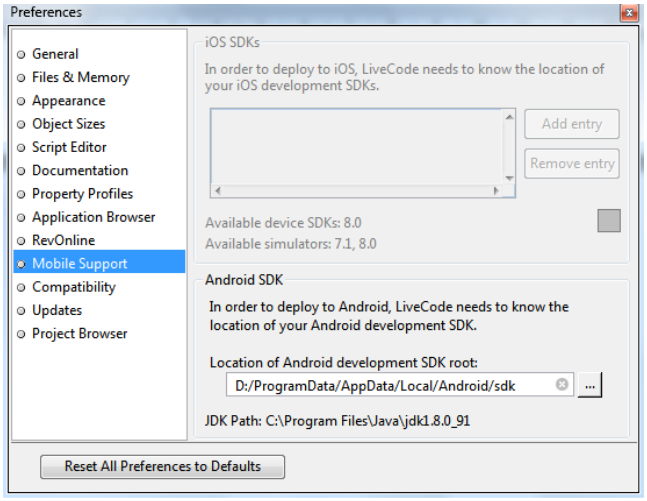
Figura 85. Configurar preferencias LiveCode (Movile Support)



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Aquí se busca en la izquierda la opción Movile Support y hacemos click, y en la parte inferior derecha de la nueva ventana hay un botón para buscar la ubicación del SDK de Android, en la mayoría de las veces a ubicado en este directorio C:/ProgramData/AppData /Local/ Android/sdk, cuando es ubicado, automáticamente el ubica el JDK de Java, ahora sí se puede grabar la aplicación como independiente para dispositivos Android.

Figura 86. Confirmación de instalación del sdk



Fuente: Esta ilustración proviene del software **LiveCode** versión 7.0

Hay que tener presente que estos últimos programas pesan alrededor de 10 Gb, espacio que debe estar disponible en el disco duro de la computadora, más otro poco para que pueda funcionar correctamente.

4.6.2. Para iPhone.

Hay una restricción para hacer aplicaciones independientes para Apple y es que se deben compilar desde aparatos con sistema operativo MacOs es decir con computadores de la casa matriz Apple, que traen incorporado el sdk iOS que aparece en letras translucidas en la parte superior cuando damos la dirección del de Android en LiveCode > Preferences >Movile Support, esto con el fin de llenar registros como desarrollador y poderse inscribir para subir programas a la tienda de Apple Store (Esta inscripción tiene un costo anual).

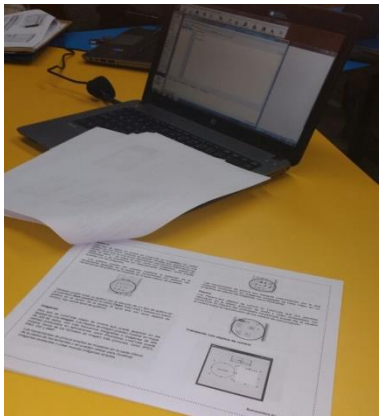
Se siguen los mismos pasos descritos para android, al crear el archivo ejecutable.

4.7. APLICACIÓN DE GUÍA (Evidencias de socialización y sugerencias)

Para la realización se planteó la comparación de distintas fuentes de información que permitieron hacer un acercamiento a la problemática y de esta manera encontrar una propuesta viable.

Se diseñó y elaboró un manual de programación con algunos comandos y algunas actividades básicas y se ejecutó como actividad de socialización con el grupo de investigación EDUMAES

Figura 87. Guía socializada.

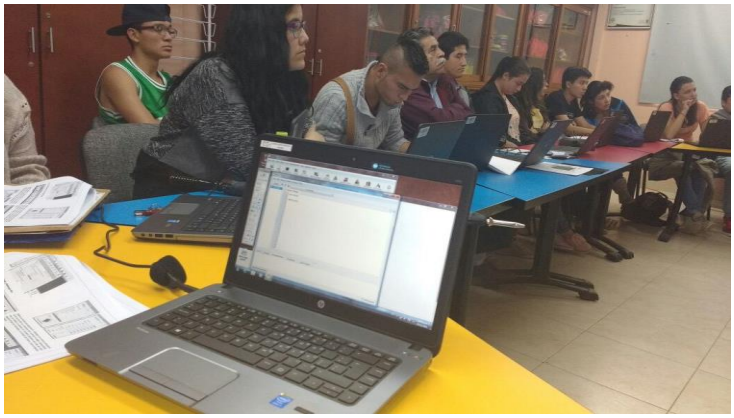


Se propuso guiar en el desarrollo de competencias TIC en los futuros docentes, con la elaboración de aplicaciones móviles en el lenguaje LiveCode

Figura 88. Trabajando objetos de control

Se propusieron actividades de aprendizaje del lenguaje de programación como iniciativa para adquirir conocimiento interdisciplinar para complementar el desarrollo profesional auto-investigativo.

Figura 89. Grupo EDUMAES reconociendo LiveCode



Se obtuvieron las siguientes sugerencias y conclusiones de la realización de esta guía piloto.

El uso de la tecnología no garantiza la adquisición de conocimiento, ya que las actividades propuestas no generan adquisición de contenidos matemáticos y parece que sirven para que se desarrollen habilidades de destreza.

El proceso de aprender el lenguaje puede ser tedioso y ocupar más tiempo del que dispone el docente para preparar su material didáctico.

Para la realización de aplicaciones tener en cuenta plantear ejercicios donde se empleen y fortalezca el uso de algoritmos.

La realización de aplicaciones debe estar bien planeada para poder medir la eficacia bien sea en fortalecer los contenidos o los procesos de pensamiento.

Se observa que con la realización del proyecto efectivamente se desarrollan competencias y habilidades en TIC y que se ve beneficiado quien aprende la programación, por el esfuerzo de aplicar elementos y contenidos de matemática aplicada, especialmente de sistemas y programación.

5. CONCLUSIONES

- En la realización de aplicaciones con este lenguaje, se han puesto en práctica conocimientos y procesos de pensamiento lógico matemáticos y de regularidades que hicieron interesante el desafío, al encontrarse con conocimiento interdisciplinar que no se ha adquirido durante su formación profesional (lenguajes de programación) pero que tiene inmersa la matemática, como algebra booleana, conjunciones lógicas, condicionales, bucles repetitivos y hasta pensamiento aleatorio.

- El desarrollo del proyecto fortaleció mi desarrollo como educador en la medida que permite integrar saberes, para desarrollar estrategias que permitan cuestionar el desempeño profesional desde diversos puntos, especialmente en el de innovación tecnológica y permite ubicarme dentro de los productores de contenido digital con proyección investigativa en TIC.

- El docente que ponga en práctica el manual se convertirá en un innovador, que rompa paradigmas, dispuesto a convertirse en un productor de contenido digital y que a su vez sepa integrar dichos contenidos en el aula de clase.

- La realización de aplicaciones requiere que el docente haga esfuerzos por desarrollar destrezas en la competencia tecnológica y de investigación para que la implementación de las mismas no se convierta en una dificultad, esto va de la mano con un proceso de actualización y adquisición de nuevas estrategias.

- El aprendizaje y familiarización del lenguaje de programación requiere de paciencia aunque es sencillo, no se facilita para todos los docentes, por lo que requiere cierto grado de interés y compromiso, de lo contrario será infructuoso, pero enriquecerá ya que permite aprender una disciplina para la que no se formó.

- El desarrollo de aplicaciones y su implementación en el aula se facilitará en la medida que todos los estudiantes puedan tener el recurso físico que permite su funcionalidad, por lo que en el momento puede ser una limitante.

- Las aplicaciones que más se usarán serán las de refuerzo de contenidos, ya que las de enseñanza requieren de una planeación de mayor cantidad de tiempo y del avance de los estudiantes, su evaluación también se torna un poco tediosa.

6. REFERENCIAS BIBLIOGRÁFICAS

Belloch, C. (2012). *Las Tecnologias de la Informacion y Comunicacion en el Aprendizaje.* España: Universidad de Valencia.

Brigham Young University. (31 de Mayo de 2005). *BYU - Brigham Young University*. Recuperado el 02 de Marzo de 2016, de BYU - Brigham Young University: http://livecode.byu.edu/

Cajiao, F. (Marzo de 2015). Innovación Educativa: La evolución del aprendizaje. (EL TIEMPO, Ed.) *Huella Social*(19), 46.

Flores, P., Lupiáñez, J. L., Berenguer, L., Marín, A., & Molina, M. (2011). *Materiales y recursos en el aula de matematicas.* Granada, España: Universidad de Granada.

Gallardo, R. J. (s.f). *Recursos y Materiales Didacticos Para La Enseñanza De Las Matemáticas: Algunas Consideraciones.* Málaga: Universidad de Málaga.

Galviz, L. S. (Marzo de 2015). ¿Educar para innovar o innovar para educar? (EL TIEMPO, Ed.) *Huella Social*(19), 46.

García Quiroga, B., Coronado, A., & Montealegre Quintana, L. (enero_abril de 2011). Formación y desarrollo de competencias matemáticas: Una perspectiva teórica en la didáctica de las matemáticas. (Universidad de Antioquia, Ed.) *Revista Educacíon y Pedagogía, 23*(59), 175.

Marchesi, Á. (2009). Las metas Educativas 2021. Un proyecto iberoamericano para transformar la educacioó en la década de los bicentenarios. *Revista CTS, 4*(12).

MEN. (2013). *Competencias TIC Para el Desarrollo Profesional Docente.* (Ministerio de Educacion Nacional, Ed.) Bogota: Imprenta Nacional.

Real Perez, M. (2012). *Las TIC en el proceso de enseñanza y aprendizaje.* Recuperado el 15 de octubre de 2015, de Las TIC en el proceso de

enseñanza y aprendizaje: http://personal.us.es/suarez/ficheros/tic_matematicas.pdf

Rodríguez Ojeda, L. (2015). *Phyton Programación.* Guayaquil, Ecuador: Escuela Superior Politécnica del Litoral.

RunRev, L. (2000-2015). *LiveCode7*. Obtenido de liveCode7: http://www.RunRev.com

RunRev, L. (2000-2015). *LiveCode7*. Recuperado el 4 de septiembre de 2015, de LiveCode7: https://livecode.com/

Tedesco, J. C. (2005). las TIC y la desigualdad educativa en América Latina. *Tercer Seminario sobre Las Tecnologías de Información y Comunicación y los Desafíos del Aprendizaje en la Sociedad del Conocimiento*.

UNESCO. (8 de Enero de 2008). Estandares de Competencia en TIC para Docentes. (I. C. Organizacion de las Naciones Unidas Para la Educacion, Ed.) 28.

Vacas, P. (2015). *HERMANO_TEMBLÓN*. Recuperado el 18 de Agosto de 2015, de HERMANO_TEMBLÓN: http://www.hermanotemblon.com/livecode-una-nueva-forma-de-programar/

Valencia, U. I. (09 de marzo de 2015). *Universidad Internacional de Valencia* . Recuperado el 14 de octubre de 2015, de Universidad Internacional de Valencia : http://www.viu.es/blog/el-aprendizaje-por-descubrimiento-de-bruner/

White, S. (s.f.). *LiveCode for Education*. Recuperado el 18 de Marzo de 2016, de LiveCode for Education: https://livecode.com/products/livecode-platform/livecode-in-education/

RunRev Ltd. (2010). LIVECODE USER GUIDE. Copyright ©2010

https:// www.runrev.com.