****

**Ing. De Software**

**Trabajo Practico N°1**

**Profesores:**  Miceli, Martin M.

Nonino, Julián

**Alumnos:** Ramírez, Nicolás Fabián

Tolay, Antonio Facundo

Vaira, Franco Gabriel

**Grupo:** Regedit. Exe

**Año Lectivo:** 2018

**Manejo de Configuraciones**

**Dirección y formas de acceso a la herramienta de control de versiones**



Para el control de versiones se usará **Git** que es una herramienta de control de versiones basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros.

Utiliza el concepto de revisión para guardar los cambios producidos en el repositorio. Entre dos revisiones sólo guarda el conjunto de modificaciones, optimizando así al máximo el uso de espacio en disco.

Git puede acceder al repositorio a través de redes como internet.

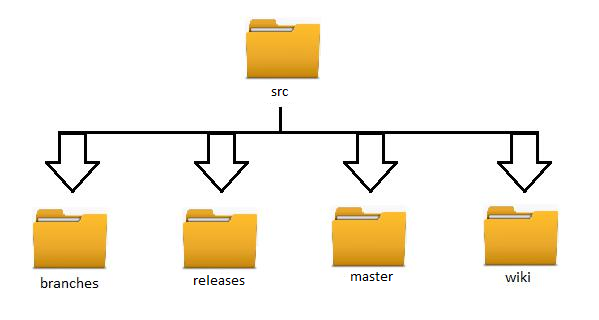


El repositorio estará físicamente en GitHub, el cual nos permite crear un repositorio gratuito bajo licencia de software libre.

Dirección del repositorio: <https://github.com/nick22xT/Simon_Gear.git>

Luego usaremos **Git** herramientas para administrar nuestro repositorio desde Windows. Con ésta se manejará el etiquetado de versiones.

**Esquema de directorios y propósito de cada uno**



*src:* Carpeta principal donde estarán alojadas todas las demás carpetas del repositorio.

*branches:* Usada para implementar funcionalidades ajenas al desarrollo troncal del programa.

*releases:* Aquí se pondrán las distintas versiones que se irán produciendo a medida que vayaevolucionando el software. Se seguirán las políticas de etiquetado de este proyecto.

*master:* Esta carpeta es el “hilo principal” del software que se va a desarrollar. En otras palabras, es donde irá evolucionando el programa para cumplir con los requerimientos del programa.

*wiki:* Aquí irá toda la documentación que esté asociada y sea necesario para nuestro proyectode software.

**Normas de etiquetado y de nombramiento de los archivos**

Para la **numeración de las versiones**, lo cual irá como nombre de las etiquetas que se irán dando a las distintas instancias del programa, se utilizará la notación de tres números separados por puntos y precedidos por la letra “V”. Ejemplo: V00.02.04

Su significado será:

* **El tercer dígito (build)** representa **correcciones** de bugs o errores encontrados.También se suelen incluir cambios no funcionales (correcciones ortográficas, agregado de comentarios, etc.).
* **El segundo dígito (release)** representa **modificaciones funcionales**, es decir se han añadido, eliminado o modificado funcionalidades al código.
* **El primer dígito (versión propiamente dicha)** representa **cambios mayores en el diseño del código.** Por ejemplo, algún cambio en cuanto a la arquitectura o patronesde diseño del software, es decir, al agregar nuevos enfoques al software. Éste primer dígito de numeración estará en 0 mientras se esté **en la etapa de desarrollo.**

El nombre de las ramas que se vayan a crear, tendrá de nombre la funcionalidad nueva que se intenta desarrollar.

**Plan del esquema de ramas a usar (y en uso)**

En base al esquema de directorios, en la carpeta “branches” irán las ramas que pueden llegar a surgir durante el desarrollo del programa bajo la siguiente condición.

“*Las ramas se harán siempre que se intente experimentar/implementar alguna nueva* *funcionalidad sobre el programa principal y que no pertenezca a los requerimientos del programa (explícitos o implícitos) pero que sume valor al software.”*

**Políticas de fusión de archivos (o sea merge) y de etiquetado de acuerdo al progreso de calidad en los entregables**

La política de fusión de las ramas con el hilo principal del programa (en la carpeta master), se harán cumpliendo los siguientes pasos:

1. El desarrollador que intente hacer merge, deberá primero contar con la última versión del programa en el master del repositorio en su workspace para hacer un “merge” con su versión ramificada, verificando mediante la ventana de comparación que nos permite el plugin de EGit en Eclipse, que no haya conflictos con la versión principal actual en desarrollo.
2. Una vez verificado esto se deberá pedir la aprobación del grupo de desarrollo para agregar la nueva funcionalidad.
3. Si se obtiene la aprobación de la totalidad del grupo, se realiza el merge.

**Forma de entrega de los “releases”, instrucciones mínimas de instalación y formato de entrega**

El software que se desarrollará para los clientes se distribuirá mediante Ejecutables de tipo jar.

Las instrucciones de ejecución serán:

1. Tener la última versión JVM.
2. Tener espacio mínimo disponible en el ordenador.

**Listado y forma de contacto de los integrantes del equipo, así como sus roles en la CCB. También incluir periodicidad de las reuniones y miembros obligatorios**

Los integrantes del grupo son:

|  |  |  |
| --- | --- | --- |
| Integrantes | Forma de Contacto | Roles en la CCB |
| Ramírez, Nicolás Fabián | -Redes Sociales: Facebook | **Colaborador General** |
|  | -Contacto Telefónico: Celular | Master |
|  | -Reuniones Periódicas | Arquitecto |
|  | -Correo Electrónico | Analista |
|  |  | Gestor de Proyectos |
|  |  | Programador |
| Tolay, Antonio Facundo |  | **Colaborador General** |
|  | -Contacto Telefónico: Celular | Master |
|  | -Reuniones Periódicas | Arquitecto |
|  | -Correo Electrónico | Analista |
|  |  |
|  |  | Diseñador grafico |
|  |  | Programador |
| Vaira, Franco Gabriel | -Redes Sociales: Facebook | **Colaborador General** |
|  | -Contacto Telefónico: Celular | Master |
|  | -Reuniones Periódicas | Calidad (Tester) |
|  | -Correo Electrónico | Analista |
|  |  |
|  |  | Arquitecto |
|  |  | Programador |

* **El Diseñador (gráfico):** Encargado de la disposición general de una aplicación. Estopuede ir desde el diseño completo de la interfaz de usuario, hasta el definir sólo algunas directrices de interfaz de usuario que los desarrolladores deban cumplir.
* **Master:** Vigila el cumplimiento de la metodología. Define prioridades, roles, proyectos.Orquesta. Facilita cosas.
* **Gestor de Proyectos**: reparte de tareas entre miembros del equipo. Maximizaregularidad del equipo y reparto coherente de tareas.
* **Arquitecto:** El papel del arquitecto de software es traducir los requisitos, tal como sedefine por los analistas, en una solución técnica. Él puede crear un diseño técnico, o simplemente algunos bocetos a mano alzada, de cómo el sistema va a estar estructurado. En cualquier caso, es la responsabilidad del arquitecto a pensar en el sistema antes de que se desarrolle. El arquitecto de una aplicación es el que decide qué camino tomar, en base a la arquitectura global que se ha elegido. Cuando el desarrollo se ha iniciado, es responsabilidad del arquitecto realizar un seguimiento del desarrollo, para ver si todavía se mantiene en consonancia con el diseño general.
* **Calidad (Tester):** Construir los Units-Test en base a los requerimientos del software yrequerimientos de arquitectura.
* **Analista:** El analista es alguien que es responsable de entender las necesidades delcliente, y asegurarse de que la solución que está siendo desarrollada se ajusta a esas necesidades. Participa en la definición de requerimientos.
* **Programador:** Aquellos que se encargarán de la programación propiamente dicha.
* **Colaborador General:** Por ser un grupo reducido y tratarse de un proyecto de softwarepequeño, todos los integrantes del grupo podremos participar en todo el proceso y todas las etapas de desarrollo del software.

Las reuniones entre los integrantes del grupo se realizarán principalmente 1 vez por semana para definir los lineamientos a seguir en el desarrollo del software. En algunos casos las reuniones tendrán solo fines informativos a fin de que cada integrante comunique el trabajo que ha realizado y se encuentra realizando.

Los miembros obligatorios de estas reuniones son todos los integrantes del grupo sin excepciones. Se deberá pactar un horario y lugar al cual puedan asistir todos.

**Herramienta de seguimiento de bugs usado para reportar los defectos descubiertos y su estado**

Se utilizará la sección “Issue” que nos ofrece GitHub para que tanto los desarrolladores como los usuarios colaboradores que lo deseen, nos informen si detectan algún error, falla, o mala implementación en alguna parte del código.

**Requerimientos funcionales**

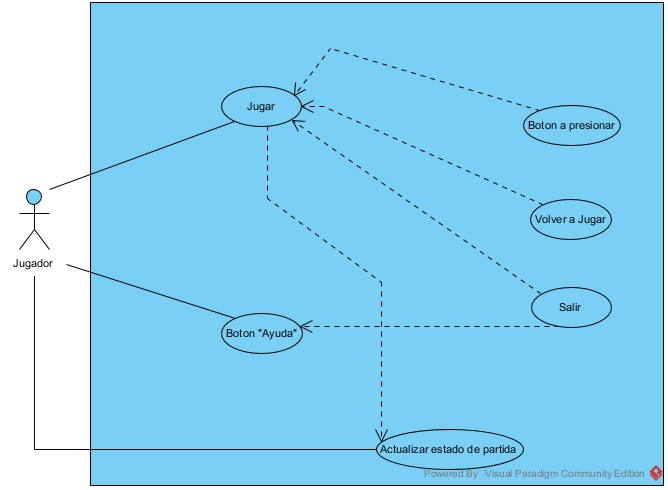
* El usuario solo podrá jugar partidas individuales contra la maquina.
* El usuario no necesita una cuenta, ni proporcionar un nickname.
* El usuario tiene 4 botones en pantalla que debe presionar dada una secuencia.
* El usuario dispone del botón de “Atrás” del dispositivo para abandonar la partida.
* La interfaz grafica debe mostrar en el centro de la pantalla un circulo en el cual se visualiza el puntaje.
* La interfaz grafica debe mostrar los botones de colores (rojo, verde, azul y amarillo). Para esto la pantalla sera dividida en 4 secciones rectangulares iguales, ocupando la totalidad de la pantalla.
* Al completar o terminar la partida debe presentarse una opción de jugar de nuevo y otra de salir.
* El ganador es aquel que logra completar la secuencia sin equivocarse.
* Cuando el jugador se equivoca en la secuencia, la partida termina automáticamente.
* En el centro de la pantalla se ubica un circulo, el cual muestra el puntaje.
* Al abrir la aplicación, se reproducirá una canción.
* La pantalla principal tendrá 3 botones: “Jugar”, “Ayuda” y “Salir”.
* El botón ayuda tendrá un texto explicativo de como jugar correctamente.
* Cada botón de color deberá parecer iluminado cuando se presiona.
* Al presionar el botón correcto de la secuencia, se reproduce la/s nota/s correspondiente a la canción.

**Requerimientos no funcionales**

* La aplicación debe ser gratuita (freeware).
* La aplicación deberá estar destinada para todo tipo de usuario de todas las edades.
* Con una interacción menor a 5 minutos, el usuario deberá ser capaz de utilizar la aplicación de forma completa.
* La respuesta de los botones deberá ser casi inmediata.
* La forma en la que se genera la secuencia es totalmente aleatoria.
* La secuencia a seguir será generada apenas se pulsa el botón “Jugar”.

**Descripción de casos de uso**

Un jugador abre la aplicación que le permite jugar SimonGear. Al presionar “Jugar” sonara una canción que se usara como base para el juego, al terminar la canción o al fracasar se desplegara una pantalla donde indicará el puntaje obtenido y dará la opción de jugar de nuevo o salir.



**Casos de prueba de sistema**

|  |  |
| --- | --- |
| **Descripción** | Secuencia de botones a presionar |
| **Ejecución del test (pasos)** | 1. Abrir app 2. Presionar jugar 3. Ver la secuencia de botones 4. Imitar la secuencia |
| **Resultado esperado** | Repetir la secuencia anterior con un elemento mas. |

|  |  |
| --- | --- |
| **Descripción** | Abandonar la partida |
| **Ejecución del test (pasos)** | 1. Abrir la app 2. Presionar jugar 3. Presionar el botón “Atrás” |
| **Resultado esperado** | Volver al menú principal |

|  |  |
| --- | --- |
| **Descripción** | Puntaje del jugador |
| **Ejecución del test (pasos)** | 1. Abrir la app 2. Presionar jugar |
| **Resultado esperado** | Ver que la puntuación comienza en cero. |

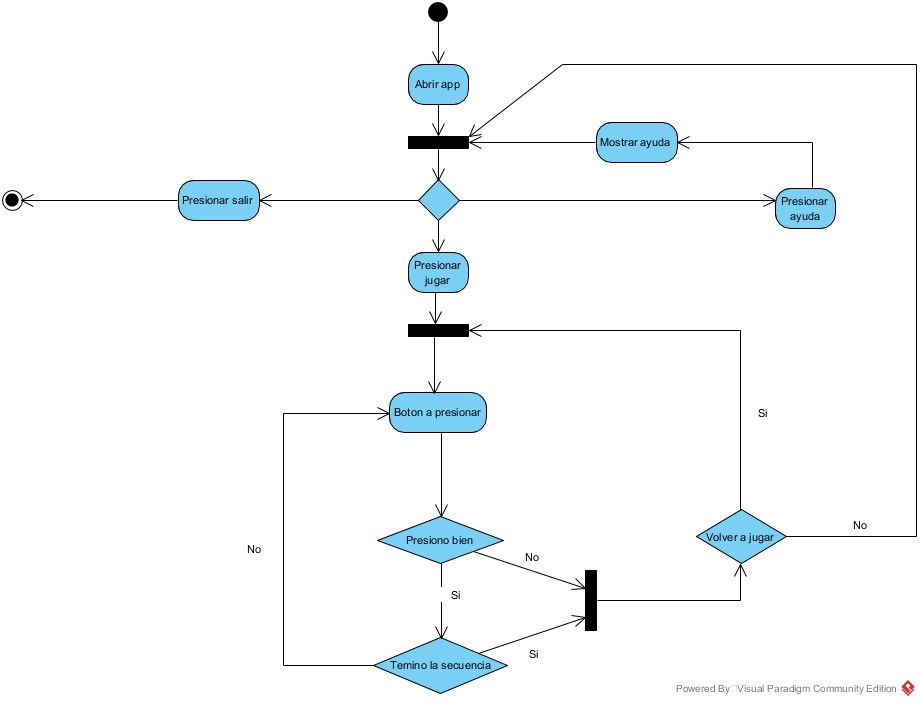
|  |  |
| --- | --- |
| **Descripción** | Partida completada |
| **Ejecución del test (pasos)** | 1. Abrir la app 2. Presionar jugar 3. Estar en modo juego 4. Haber completado toda la secuencia |
| **Resultado esperado** | Visualizar menú con la opción de jugar de nuevo o salir |

|  |  |
| --- | --- |
| **Descripción** | Menú principal |
| **Ejecución del test (pasos)** | 1. Abrir la app |
| **Resultado esperado** | Visualizar menú con las opciones “jugar”, “ayuda” y “ salir” |

|  |  |
| --- | --- |
| **Descripción** | Ayuda |
| **Ejecución del test (pasos)** | 1. Abrir la app 2. Presionar el botón ayuda |
| **Resultado esperado** | Visualizar las instrucciones de como jugar correctamente. |

|  |  |
| --- | --- |
| **Descripción** | Botón a presionar y presionado |
| **Ejecución del test (pasos)** | 1. Abrir la app 2. Presionar jugar 3. Ver la secuencia de botones |
| **Resultado esperado** | Ver que los botones se iluminen con la secuencia |

**Diagrama de actividades**

****

**Matriz de Trazabilidad**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Requerimientos del sistema | Casos de uso | | | | |
| Jugar | Ayuda | Salir | Volver a jugar | Botón a presionar |
| El usuario solo podrá jugar partidas individuales contra la maquina. | ✓ |  |  |  |  |
| El usuario tiene 4 botones en pantalla que debe presionar dada una secuencia. | ✓ |  |  |  | ✓ |
| El usuario dispone del botón de “Atrás” del dispositivo para abandonar la partida. |  |  | ✓ |  |  |
| En el centro de la pantalla se ubica un circulo, el cual muestra el puntaje. | ✓ |  |  |  |  |
| Al completar o terminar la partida debe presentarse una opción de jugar de nuevo y otra de salir. |  |  | ✓ | ✓ |  |
| La pantalla principal tendrá 3 botones: “Jugar”, “Ayuda” y “Salir”. | ✓ | ✓ | ✓ |  |  |
| El botón ayuda tendrá un texto explicativo de como jugar correctamente |  | ✓ |  |  |  |
| Cada botón de color deberá parecer iluminado cuando se presiona. |  |  |  |  | ✓ |
| La secuencia a seguir sera generada apenas se pulsa el botón “Jugar”. | ✓ |  |  | ✓ |  |
| Al presionar el botón correcto de la secuencia, se reproduce la/s nota/s correspondiente a la canción. | ✓ |  |  |  | ✓ |

**Matriz de Trazabilidad entre requerimientos y pruebas**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Requerimientos | Secuencia de botones a presionar | Abandonar la partida | Puntaje del jugador | Partida completada | Menú principal | Ayuda | Botón a presionar y presionado |
| El usuario solo podrá jugar partidas individuales contra la máquina. | ✓ |  | ✓ | ✓ |  |  | ✓ |
| El usuario tiene 4 botones en pantalla que debe presionar dada una secuencia. | ✓ |  |  |  |  |  | ✓ |
| El usuario dispone del botón de “Atrás” del dispositivo para abandonar la partida. |  | ✓ |  |  | ✓ |  |  |
| En el centro de la pantalla se ubica un circulo, el cual muestra el puntaje. |  |  | ✓ | ✓ |  |  |  |
| Al completar o terminar la partida debe presentarse una opción de jugar de nuevo y otra de salir |  | ✓ |  | ✓ |  |  |  |
| La pantalla principal tendrá 3 botones: “Jugar”, “Ayuda” y “Salir”. |  |  |  |  | ✓ | ✓ |  |
| El botón ayuda tendrá un texto explicativo de como jugar correctamente |  |  |  |  |  | ✓ |  |
| Cada botón de color deberá parecer iluminado cuando se presiona | ✓ |  |  |  |  |  | ✓ |
| La secuencia a seguir sera generada apenas se pulsa el botón “Jugar”. | ✓ |  |  |  |  |  | ✓ |
| Al presionar el botón correcto de la secuencia, se reproduce la/s nota/s correspondiente a la canción. | ✓ |  |  |  |  |  | ✓ |