Kyoro, a Pyoro 2 clone

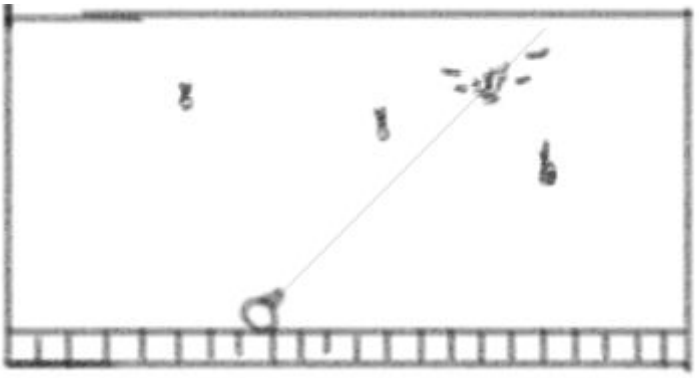
Game Design Document

Autor
Maximetinu Fucking Freaker (Miguel Medina Ballesteros)

Contacto
maximetinu@gmail.com

Versión
0.0.0

Fecha
21 de Enero de 2018



Resumen

El player debe proteger las baldosas del suelo que pisa de ser destruidas de las bombas que caen, disparándoles en un ángulo de 45º.

Índice

Visión general

Resumen (más largo)

El player anda sobre un suelo embaldosado de cuadrados unidad. Las baldosas son destruidas al ser golpeadas por unas bombas que caen del cielo desde posiciones aleatorias, unas más rápidas y otras más lentas. Hay que disparar a las bombas para destruirlas, el disparo no es un proyectil sino un rayo instantáneo que las atraviesa. Hay dos bombas especiales: unas regeneran una baldosa al ser destruidas y otras las regeneran todas y destruyen todas las que están cayendo en ese momento. Se regeneran siempre primero las más cercanas al player. Al ser destruidas suman puntos y si se destruyen varias alineadas es un combo y dan más puntos.

A más tiempo pasa, más rápido caen las bombas y más rápido se mueve el player. Llega un momento en que es imposible proteger todo el suelo, tienes que estar constantemente reparándolo. Las bombas reparadoras no aparecen desde el principio sino poco a poco.

El escenario tiene que ir cambiando conforme pasa el tiempo y la música volviéndose más rápida.

El objetivo es aguantar el máximo tiempo posible haciendo la máxima puntuación.

Sentimiento buscado en el player

Vicio puro. Un pique para superar su puntuación máxima y la curiosidad de ver cuál será el siguiente cambio en el escenario y/o la música.

Al principio debe sentirse calmado y luego luchando por su supervivencia.

Plataformas objetivo

Ordenadas por prioridad:

1. Android
2. iOS
3. WebPlayer
4. PC
5. Facebook game

Monetización

Al principio ninguna. Si gana público se estudiará si es posible monetizarlo con publicidad (teniendo en cuenta que es un clon de otro juego).

Según una búsqueda rápida en Google, los únicos aspectos sujetos a copyright en los videojuegos son la narrativa, el arte y la música, luego modificando estos aspectos pero manteniendo el gameplay se podría hacer: ¿antiaéreo y bombas?

Características online y multiplayer

- Solo para 1 player
- Tablas de highscore
    - Al menos en Android, WebGL y Facebook
- Savegame (del highscore únicamente) en la nube
    - Al menos en Android y Facebook

Motor gráfico del desarrollo

Unity, versión variable y por determinar.

Elementos de juego

- Player
- Baldosas del suelo (destruibles y cuadradas)
- Bombas
    - Normales
    - Regeneradoras de 1 baldosa
    - Regeneradoras de todo el suelo (y destructoras del resto)

Sistema de juego: acciones, reglas y mecánicas.

Un grupo de reglas es llamado sistema, y la interacción de dos o más reglas se denomina mecánica. Las reglas limitan la acción del player.

Acciones

Del player

- Moverse (para apuntar y esquivar)
- Disparar (para destruir las bombas)

No podrá ejecutar ambas acciones al mismo tiempo. Si pulsa en disparar, se detiene, por tanto predomina el disparo.

Del player

- Controlar al player
    - Moverse

    - Disparar
- Pausar el juego
    - Continuar
    - Salir del juego

De las bombas

- Caer (para destruir el suelo o al player)
    - A una velocidad base más un aleatorio
- Destruir el suelo con el contacto
- Matar al player con el contacto
- Ser destruida por un disparo

Reglas

Del player

- No se puede mover si no hay baldosa en esa dirección
- No puede salirse de la pantalla absolutamente nada
- Muere si es golpeado por una bomba Al morir:
    - Se guarda el highscore
    - Se reinicia el juego
- Tiene un cooldown entre disparo y disparo

De las bombas

- Al matar al player o destruir el suelo también se destruyen a sí mismas.
- Al ser destruidas por un disparo:
    - Suman una puntuación
    - Si son destruidas más de una al mismo tiempo, la puntuación de cada una se dobla
- Tipo especial regeneración de una baldosa
    - Regeneran la baldosa más cercana al player
- Tipo especial regeneración de todas las baldosas
    - Regeneran todas las baldosas del juego en orden de cercanía al player
    - Destruyen todas las bombas que están cayendo en ese momento
- Al ser destruidas por la bomba especial regeneradora de todas
    - No se destruyen a la vez, hay un pequeño delay y se destruyen primero las más bajas y luego las más altas.

Cómo spawnean

- No se pueden salir nada de la pantalla
- No pueden spawnear dos al mismo tiempo
- A más tiempo pasa, más probabilidad hay de spawnear las especiales
    - Primero la de regeneración de 1 baldosa

    - Después la de regeneración de todas

De las baldosas al ser regeneradas

- No se regeneran instantáneamente, hay un pequeño delay (una animación)
- Además, si se regeneran todas, no todas las regeneraciones se lanzan al mismo tiempo. Hay un pequeño delay entre ellas (al igual que en la destrucción de las bombas que caen).

Qué ocurre al pasar el tiempo

- Crece la velocidad base de las bombas
- El player se mueve más rápido
- Cambia el escenario
- Se acelera la música

Mecánicas emergentes

- El player intentará alinearse para destruir varias bombas a la vez y hacer combo. Esto dejará de preocuparle cuando ya lluevan demasiado rápido, por ello la puntuación solo se dobla y no se multiplica. Porque si se multiplicase, y teniendo en cuenta que será mucho más fácil darle a varias a la vez porque son más, sería un factor aleatorio en la suma de puntuación muy grande y, además, aleatorio e inintencionado por el player. Doblándolo solo se da ese incentivo, sobre todo al principio, pero no desbalancea por aleatoriedad.

- Cuando gran parte del suelo es destruido y el player queda en una isla, gran parte también de las bombas son spawneadas fuera de esa región, por lo que el player se centrará en esquivar las que tiene encima (con más dificultad) y en destruir las bombas recuperadoras a toda costa.

- Cuando ya es imposible proteger todo el suelo y destruir todas las que caen, la principal preocupación del player será identificar los patrones (aleatorios) de cómo caen las bombas para esquivarlas al mismo tiempo que busca desesperadamente la bomba regeneradora-destructora, que en ese momento spawneará cada mucho menos tiempo. Perder una de estas bombas significará muerte casi segura.

- Al cambiar el escenario, la música y la dificultad conforme pasa el tiempo, el player tendrá curiosidad por saber qué hay después. Es un incentivo más además del highscore.

- Si el player queda atrapado en una única baldosa o dos, en un espacio muy pequeño, lo único que podrá salvarle será la bomba regeneradora-destructora, pues desde ahí casi no podrá esquivar.

- La muerte no parecerá un factor aleatorio, pues el player, al morir, pensará cosas como: “si me hubiera centrado en proteger esa región de suelo… ” o “si me hubiera

colocado a disparar desde esta parte de la pantalla… ”, “... hubiera sobrevivido:”. El player querrá aprender y volver a intentarlo para no caer en los mismos errores.

Sistema de juego emergente

El player novato tendrá un minuto para hacerse con las mecánicas y otro para descubrir que no todo va a ser un paseo. No debería durar más de eso.

A partir de ahí descubrirá que cuando aguanta más tiempo, las cosas cambian, y le nacerá curiosidad y el reto por aguantar más.

Para el player medio, que supera la barrera del novato pero aún no es un veterano, querrá descubrir cómo cambia el escenario más allá al mismo tiempo que aprende de sus errores al morir y quiere volver y volver a intentarlo para hacer el principio perfecto y tener el suelo entero cuando las cosas se complican.

El player veterano pasará a preocuparse más por el highscore que por descubrir escenarios. Aún así, los últimos cambios de escenario se situarán en las situaciones más extremas a las que lleguen estos playeres, de forma que no sientan que ya se lo han pasado aún. Esta preocupación por el highscore le llevará a hacer unas primeras fases impecables y a identificar muy rápido y bien los patrones para sobrevivir en fases avanzadas.

Estimaciones de dificultad a lo largo del tiempo

- Minuto 0: tutorial de lo fácil
- Minuto 1: acelera pero con tranquilidad
- Minuto 2: empieza lo chungo
- Minuto 3: hardcore players
- Minuto 4: infumable
- Minuto 5: no debería llegar nadie

Condiciones de victoria y derrota

El juego no tiene condiciones de victoria, solo la supervivencia y la satisfacción del highscore y los nuevos escenarios.

La única condición de derrota es la muerte.

Estado del juego

Los datos que se identifiquen aquí deben determinar el estado del juego hasta tal punto que, en el hipotético caso de cerrar el juego y después reanudarlo, fuese posible restaurar el punto anterior exacto sin provocar ningún fallo en situaciones especiales.

El estado del juego en cada momento viene determinado por:

- El tiempo de la partida (para las velocidades, spawns, escenarios, etc)

- Puntuación actual y máxima puntuación
- Baldosas: destruidas y cuales quedan
- Posición de las bombas en caída y velocidad de cada una
    - También: tiempo del último spawn
- Player:
    - Posición
    - Orientación
    - Vivo o muerto
    - También: tiempo desde el último disparo
- Estado del pause

Nótese que, elementos como la velocidad base de las bombas, el rating de spawn o la velocidad del player no son parte del estado inherente, ya que dependen del tiempo de la partida y esto sí que se recuerda. Serían, en cualquier caso, estados emergentes.

Los tiempos desde el último disparo y el último spawn serían para, en el hipotético caso de poder salvaguardar todo el estado del juego en tiempo real, evitar explotaciones del tipo: disparo, salgo de la partida y entro de nuevo y ahora puedo disparar sin esperar los 0,25s de cooldown (por ejemplo).

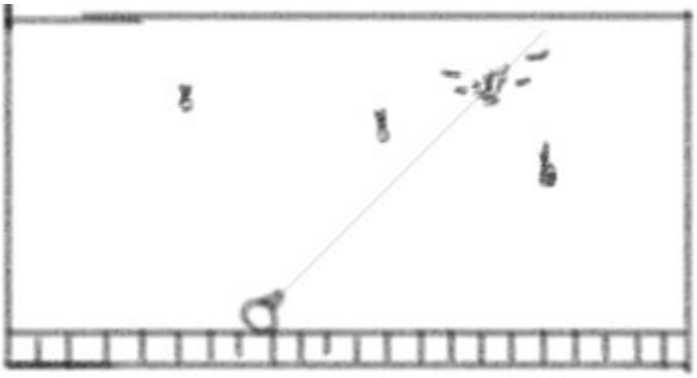
De cualquier modo, este nivel de detalle no es necesario, pues nuestro juego no requerirá esta precisión de estado. Aún así, siempre es útil identificar la completitud de nuestro estado de juego.

Por ejemplo, cabría mencionar que se han obviado el estado de una situación aún más especial y precisa: si una bomba regeneradora ha sido destruida pero el juego aún está en la transición de reparar el suelo y destruir las bombas, ya que no es instantáneo.

Persistencia (savegame)

El único estado que debe guardar nuestro juego es la puntuación en el momento de la muerte, que será recordada como highscore y persistirá entre sesiones.

Escenario



En primer plano, el escenario constará de 30 baldosas (las del boceto de arriba no son representativas), cuadradas, del tamaño del player.

Background. ¿Cómo cambia con el tiempo?

Tomar todo apartado estético, como este, como proposiciones.

Cambiará, a partir del minuto 2, aproximadamente cada minuto. De dos formas:

- Con un scroll parallax como si el horizonte estuviese rotando, aparecen nuevas cosas en el horizonte y desaparecen las cercanas.
- Apareciendo nuevos elementos de repente, con sus respectivas animaciones, etc.

Adaptación de pantalla en dispositivos móviles

En PC, WebGL y Facebook será fácil adaptar la pantalla y el juego no tendrá por qué ocupar la completitud del monitor, por lo que nos centraremos en los tamaños de dispositivos móviles, que son los más difíciles de adaptar, y a partir de ahí extenderemos al tamaño monitor.

El juego se jugará siempre en con el móvil en orientación apaisada.

A la hora de adaptarlo tenemos dos opciones: extender a lo ancho o a lo alto. La decisión no es banal porque podría modificar sustancialmente el gameplay, hay que evaluar qué opción lo modifica menos conservando estética y quedarnos con esta.

Conservar altura VS conservar anchura

Pongamos el caso de dos dispositivos apaisados, uno más ancho que el otro. En el dispositivo más ancho, tenemos la opción de añadir baldosas al suelo hasta completar la pantalla (respecto al dispositivo pequeño) o de hacer que el suelo se vea más de cerca hasta que se ajuste a la pantalla, pero reduciendo así la altura a la que se ven caer las bombas.

- Si añadimos baldosas a la pantalla más grande, tiene más espacio para esquivar por lo que facilitamos su gameplay.
- Si ampliamos la imagen hasta ajustar el suelo, tiene el mismo espacio para esquivar que la pantalla pequeña pero ve menos altitud, por lo que tiene menos tiempo para esquivar las bombas y, por tanto, dificultamos su gameplay.

Es muy difícil determinar cual de los dos casos arriba mencionados modifica en más cantidad la dificultad del juego. En mi opinión subjetiva, a ojo, creo que importa más el espacio por el que puedas esquivar y la posibilidad de que spawneen las bombas en un espacio mayor que la altura a la que puedas verlas para anticiparte a ellas y esquivarlas, por lo que apostaría por adaptarlo en altura y conservar la anchura.

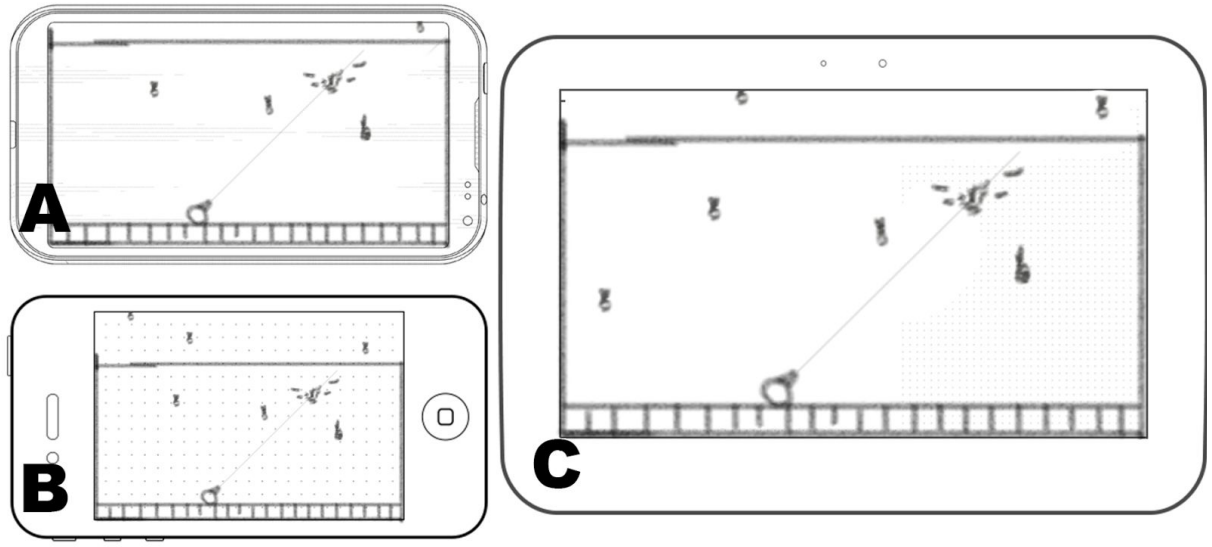
Si intercambiamos los papeles y tomamos como base que el original es la pantalla grande, tenemos el mismo caso pero al contrario. Si reducimos baldosas para adaptar, tendría menos espacio para esquivar, por lo que más dificultad. Y en mi caso preferido, si conservamos baldosas pero modificamos altura, le estaríamos facilitando el juego a las pantallas pequeñas.

Tal vez la solución ideal pasaría por una solución intermedia que parametrizara elementos del gameplay adaptándolos a la pantalla. **Ejemplo de soluciones ideales:**

- Modificamos baldosas y conservamos altura, pero modificamos el ratio de spawn de las bombas para que haya más o menos posibilidad de que caigan ahí en la anchura que queda.
- Modificamos altura y conservamos baldosas, pero modificamos también la velocidad de caída de las bombas en función de la altura, de forma que en pantallas con menos altura caigan más lento y el tiempo para esquivarlas sea el mismo.

La segunda solución ideal es la más fácil de implementar, pues bastaría con modificar el concepto "velocidad de la bomba" por "tiempo que tarda la bomba en cruzar la pantalla", de forma que adapte su velocidad para tardar lo mismo en cualquier pantalla.

En la imagen del boceto de debajo, podemos ver una comparación de cómo se vería en los distintos dispositivos en el caso de conservar la anchura.



Aunque todos tienen el mismo terreno que proteger y por el que esquivar. es fácilmente apreciable que el dispositivo B tiene más tiempo para anticiparse a las bombas que los otros dos. De hecho, puede anticiparse a mayor cantidad de bombas que el A y el C. Fíjense en la diferencia del A, de poder anticiparse solo a una, al B, a poder anticiparse a 3 más. Es una diferencia sustancial que se agrava en casos especiales, como con el juego avanzado cuando hay más bombas y vamos buscando las especiales.

La solución pasaría por adaptar la velocidad en función de la pantalla, pero, ¿es esta la decisión correcta? No estamos teniendo en cuenta factores como que, en una pantalla más

pequeña pero más alta en proporción como la de B, al ser más pequeña nuestros ojos reciben menos información y reaccionan menos rápido. Y si para colmo las bombas son más rápidas...

Conclusiones

Es difícil encontrar una solución ideal. La mejor solución parece ser adaptar en altura y conservar la anchura y suplir la diferencia de dificultad modificando la velocidad de las bombas en función de la altura.

Como no estoy seguro de si se sentirá justa la diferencia de velocidad en las bombas, en principio esta decisión queda pendiente de testear empíricamente.

En principio el desarrollo se hará adaptando a lo alto sin más, teniendo en cuenta la posibilidad de ajustar la velocidad y programando anticipándonos a este posible cambio.

Interfaz

Pantallas

- Pantalla de inicio (menú principal)
- Pantalla de opciones
- Pantalla de créditos
- Posible pantalla de highscores
- Pantalla de juego

Todas en vista apaisada.

Descripciones de pantallas

- Pantalla de inicio (menú principal): a la izquierda menú con las opciones -Jugar, -Opciones, -Créditos y -Highscores. A la derecha una imagen del juego, cayendo bombas.
- Pantalla de opciones. A la izquierda las opciones, a la derecha otra animación (o la misma que la del menú principal).
- Pantalla de highscores. A falta de estudiar las funciones que ofrece los Highscores de Google Play en Android, los de Facebook, simplemente una lista con los highscores. Evaluar más adelante si permitir filtrar por semana, mes, por amigos (en facebook), país, etc. → ¿por contactos en Android?
- Créditos. A falta de saber cuáles serán los créditos, se queda sin definir. Lo ideal sería, si somos pocas personas una pantalla para cada uno y que scrollee lentamente. En caso de ser varios unos créditos tradicionales. Valorar si incluir créditos al Pyoro original. Ver si es necesario incluir créditos a los programas utilizados (Unity, etc).

Transiciones entre pantallas

- Entre menú principal y opciones, que las letras de las opciones se desplacen hacia la derecha y desaparezcan y que por la izquierda vengan las opciones. Con las opciones al revés, que se vayan hacia la izquierda. Considerar

In-game HUD

- Current score, arriba a la izquierda
- Highscore: arriba a la derecha
- Símbolo de pause || → arriba a la izquierda
- Destroying score: puntos ganados con la destrucción de una bomba, aparecen durante un momento cada vez que una bomba es destruida, en la posición en la que ha sido destruida.
- Press / Touch for restart cuando el player muere
- Indicador de pause al pausar el juego e indicador de resume y de salir.

A lo largo de este apartado falta bocetado, prototipado, testing, pero sobre todo elegir un estilo gráfico para el juego, lo cual se determina en secciones posteriores. En dichas secciones habrá una dedicada al estilo de la interfaz.

Opciones

En la pantalla de opciones se permitirán modificar las siguientes opciones:

- Volumen de la música
- Volumen de los efectos de sonido
- En PC: controles, Key Mappings

Tomar esto como propuesta ideal. Es posible que no sea necesario permitir modificar el volumen de ambas cosas por separado, o modificarlo del todo incluso. También es posible que no sea necesario permitir modificar los controles del PC, ya que estos son extremadamente sencillos.

Controles

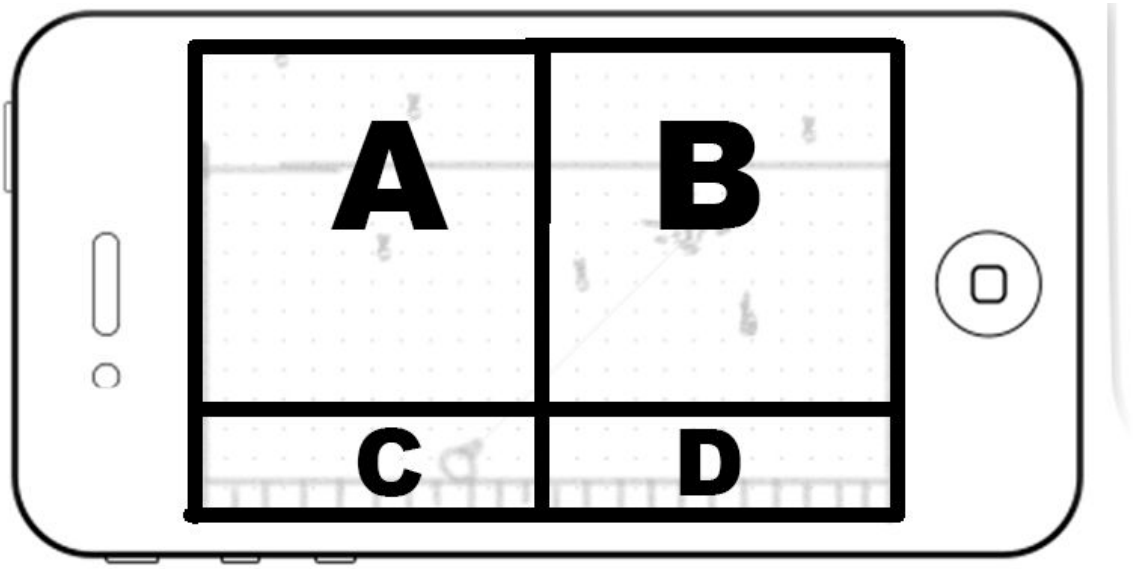
- Moverse a izquierda/derecha
- Disparar/seleccionar opción del menú
- Pause/Salir de pause

En dispositivos móviles

Los controles en el menú principal y en el menú de pausa serán táctiles sencillos, tocar con el dedo sobre el menú para pulsar el botón.

Durante el gameplay

Se controla al player tocando la pantalla en determinadas regiones, que son las siguientes:



- C-D: moverse hacia la derecha o la izquierda.
- A-B: disparar hacia arriba a la derecha o arriba a la izquierda

Si se identifica el input de A o B, C y D no tendrían efecto (predomina el disparar al moverse)

¿Cómo pulsar pause? Pulsando en el símbolo de pausa de arriba a la izquierda, no representado en el boceto superior.

En PC

Hay que considerar si permitir su modificación. O, en caso de que no, habría que considerar si incluir controles duplicados para diferentes preferencias (por ejemplo, A y D y ← y → para el movimiento, ambos a la vez.

- Durante el gameplay
    - A y D, y ← y → para moverse (respectivamente)
    - W, [UP ARROW] y [SPACE] para disparar
    - [ESC] para pausa.
        - [ESC] para salir de pausa
        - W y S, [UP ARROW] y [DOWN ARROW] para cambiar entre las opciones del menú de pausa (continuar y salir)
        - [ENTER] para seleccionar la opción

        - También mouse y click izquierdo sobre el botón para seleccionarlo
    - *<Experimental a falta de testing>* Con el mouse:
        - Izquierda y derecha para mover al player
        - Click para disparar (en la dirección de su último movimiento)
- Durante los menús
    - [UP ARROW] y [DOWN ARROW] para moverse por los menús
    - [ENTER] para seleccionar
    - ← y → para variar las opciones de slide, como el volumen
    - He olvidado a propósito WASD porque son poco intuitivos durante los menús
    - Con mouse:
        - Click izquierdo sobre el botón para seleccionarlo
        - Click y arrastrar sobre las opciones de slide para modificarlas, como el volumen.

Lore

Aún por determinar

Estilo gráfico

Aún por determinar

Gama de colores

Aún por determinar

Gráficos y animaciones en menú principal y créditos

Aún por determinar

De momento, idea: menú principal a la izquierda. Al pulsar opciones se sale de la pantalla hacia la derecha y por la izquierda vienen las opciones que se colocan a la derecha de la pantalla. Al volver al menú principal se invierte la animación (las opciones se van hacia la izquierda y el menú principal viene de la derecha).

Estilo sonoro

Aún por determinar

Efectos de sonido

Aún por determinar

Música

Aún por determinar

Assets gráficos

Aún por determinar

- Player
    - Caminar
    - Disparar
    - Morir
    - Idle
- Bomba
    - Caída (caso de la semilla original)
    - Explosión de destrucción
        - Dirección 45º en el lado opuesto del disparo
    - Explosión de destrucción del suelo
    - Hojas restantes de la destrucción (caso de la semilla original)
        - Hojas pequeñas
            - animación de soplido (al dispararle, se mueven)
            - animación de caída
        - Hojas grandes (caen más lento)
            - animación de soplido
            - animación de caída
- Bomba regeneración 1 (igual que bomba pero con cambios que enfaticen)
- Bomba regeneración - destrucción - todos (igual que bomba pero con cambios que enfaticen aún más)
- Baldosas
    - Destrucción
    - Regeneración
- Background, animación, cambios y elementos del background

Assets de sonido

Efectos de sonido

Aún por determinar

- Disparo
- Impacto bomba-suelo
- Bomba destruida
    - ¿Diferente para bombas destruidas en combo? Para enfatizar

- Suelo regenerado
    - Diferente pitch para regeneraciones en cadena por la bomba regeneración todo
- Highscore superado
- Muerte

¿Sonidos del escenario?

Música

Aún por determinar

- Música del menú principal
- Música de los créditos
- Música del juego
    - Un solo track lo suficientemente largo como para que nadie llegue a terminarlo jamás y cada vez más acelerado, con algún cambio entre medias (no simplemente acelerado y ya). **En sintonía con el background**.

Detalle de las entidades de juego: atributos

A pesar de estar de las últimas, el detalle de esta sección es el más importante para la programación A partir de aquí los conceptos definidos serán casi traducibles a código.

Inicializar los valores del Player, Cooldown, rates, etc analizando con exactitud el gameplay del juego original

Dispositivo

- Screen height
- Screen width

Player

- Speed
- Shot Cooldown
- Additional Speed by time (inicialmente 0)

Player State

- Current Score
- Highscore

Bomba / Regenerar 1 / Regenerar destruir todos

- Base Speed
- Additional Speed by randomness → *Checkear si los especiales también la tienen*
- Additional Speed by time (inicialmente 0)

De momento, obviado la adaptación a la altura de la pantalla. En caso de implementarla habrá dos opciones, o modificando las por tiempo en cruzar la pantalla, o puede que mejor por más intuitivo: haciendo una regla de 3 desde un tamaño estándar de pantalla y aplicando ese modificador a todas las velocidades.

Bomb spawner

Entidad encargada de spawnear las bombas

- Time between spawns
- Regenerar 1 rate
- Regenerar todos rate
- Score
- Combo score

Gamemode

Entidad encargada de acelerar el juego y hacerlo más difícil con el tiempo

- Curva Player.Additional Speed - Time
- Curva Bomb.Additional Speed - Time
- Curva Bomb spawner.Regenerar 1 rate - Time
- Curva Bomb spawner.Regenerar todos rate - Time

Background

Al momento de escribir esto, aún faltan por determinar los gráficos del juego, por lo que no se puede detallar más que esto:

- Velocidad de cambio del background
- Lista de cosas que spawnear - momento en que se spawnean. Cada cosa entonces tendrá sus propio atributos también, aún por determinar.

Programación

Aún por determinar

Marketing

Aún por determinar

Organización temporal del proyecto

Aún por determinar