**Start screen, timer, level progression, finish**

**Phase 7#, Diagonal player collision:**

* Zaznaczone komentarzem w ktorym miejscu w movePlayer() ma byc

**Phase 6#, Level progression:**

* Znajdz przeciwny rog do rogu w ktorym pojawia sie gracz
* Skanuj w sposob pokazany ponizej, by znalezc pierwsze wolne miejsce

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* + A~~lbo y sie zmienia przy kazdej linii lub x,~~ znajdujac wiecej niz jeden punkt mozna sobie darowac bias
  + DeltaY -1 (jezeli startPosY=1) lub +1 przy kazdym okrazeniu while
  + Pomniejsze deltay odwrotne od duzego
  + Pomniejsze deltax +1 gdy startPosX =1 i -1 gdy startPosX po drugiej stronie
* Ewentualnie znajdz wiecej oddalonych punktow i wybierz losowo (lub z warunkiem ze na koncu korytarza) jeden z nich
* Znajdz wiecej pustych miejsc, lecz zrob zeby wyjscie bylo z dojsciem oddalonym od kierunku z ktorego przychodzi gracz (jezeli gracz zaczyna w prawym dolnym rogu, nie powinno byc dojsc z prawej i od gory)
* Generacja nowego poziomu przy dojsciu
* Skanowanie wiecej niz polowy lub niekwadratowych leveli jest zle, zrobic system niediagonalny dla niekwadratowych

**Phase 5#, Walls optimalization - DONE:**

* Utworzy Array zawierajacy wszystkie wall’e obecnie na ekranie
  + Size = (screenSize/tileSize)+1
* Uzyc deltaOffset by zobaczyc, ktore wall’e removeChild, a ktore addChild
  + Znalezc stary i nowy offset
  + Znalezc mapcoordinates wedlug offsetu
  + Jezeli koordynaty sa takie same – continue
  + Przerobic koordynaty na pozycje w arrayach
  + removeChild scian ktore nie sa na ekranie
  + przesunac array by sciany ktore nadal sa na ekranie byly w odpowiednim miejscu
  + wypelnic puste miejsca scianami ktore pojawily sie na ekranie
    - najpierw dodac wszystkie nowe sciany do arrayu
    - potem przeszukac array i dodac sciany ktore nie sa w wallsLayer do wallsLayer if(!contains)

**Phase 4#, Screen movement - DONE:**

* Po detekcji kolizji sprawdz czy gracz jest blisko granicy mapy
* Przesun gracza zamiast «walls» jezeli blisko granicy mapy, lock in place walls
* Przesun walls zamiast gracza jezeli dalej od granicy mapy, lock player in place

**Fase 3#, Generating - DONE:**

* Wybierz punkt poczatkowy
* Wybierz dlugosc linii od 2 do 5(?) liczac od pierwszego edytowanego tile’a (tzn nie od punktu poczatkowego linii)
  + Sprawdz czy linia nie wychodzi za granice mapy
* Utworz linie jezeli wszystkie punkty poza pierwszym i ostatnim sa wallami

|  |  |  |  |
| --- | --- | --- | --- |
| Mx (whatever), length = 3 | Mx+dx (whatever) | Mx+2\*dx (must be wall) | Mx2 (whatever) |

* Sprobuj utworzyc linie 5 razy, inaczej koniec

**Fase #2 - DONE:**

* Legg til «2» som betegnelsen på hvor spilleren begynner, endre navn på tabllen «walls» til «map»
  + Lag et symbol for spilleren
  + Senter skjermposisjonen til spilleren
* Kollisjonsdeteksjon i funksjonen «loop»
  + Lag en funksjon som returnerer koordinater wx og wy til spilleren henholdsvis til «map» tabellen
  + Simuler den neste posisjonen til spilleren og sjekk for kollisjonen, vertikal og horisontal bevegelse for seg

**Fase #1 -** **DONE:**

* Plasser hele koden i klasser, begynn med klasse Main der alt har utgangspunkt
* Lag et enkelt symbol som representerer en vegg, bildet er rektangulært. Bestem størrelsen, alle framtidige grafikker er avhengige av det. Lag en lett tilgjengelig konstant «tileSize» i koden.
* 2d tabell med vegger
  + Først fyll tabellen med tall (1 for vegg, 0 for tom plass)
  + Plasser veggene på skjermen (i en movie clip «walls») ved å tolke tabellen
* Kan bevege seg rundt veggene (i koden er det egentlig veggene som beveger seg på skjermen)
  + EventListener for opp, ned, venstre og høyre
  + Beveg hele «walls»-movie clipen