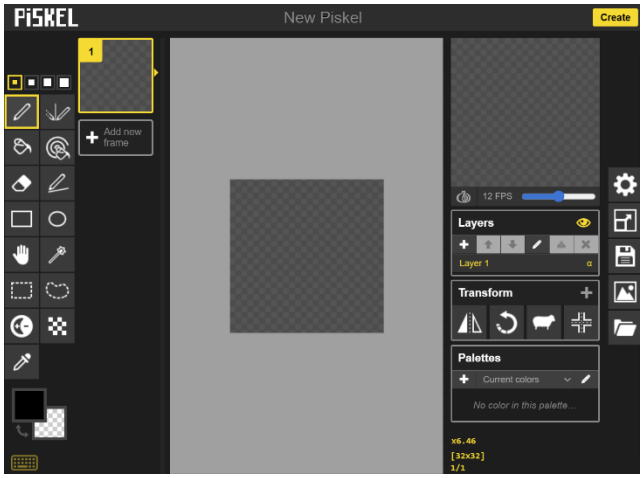
Pong Godotilla

Vaihe 1: Peliprojektin esineiden kuvien muodostaminen

Askel 1: Mene osoitteeseen: https://www.piskelapp.com/kids/

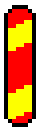


Askel 2: Luo kuvat pelaajien mailoille ja pelin pallolle

Mailan koko olisi 8*32 pikseliä ja pallon koko 16*16. (leveys*korkeus)

Valitse oikealta  ,muuta **Width** (leveys) ja **Height** (korkeus) esineen kokoja vastaaviksi ja valitse sen jälkeen **Resize**

Mallikuva pelaajan mailalle



Mallikuva pelin pallolle

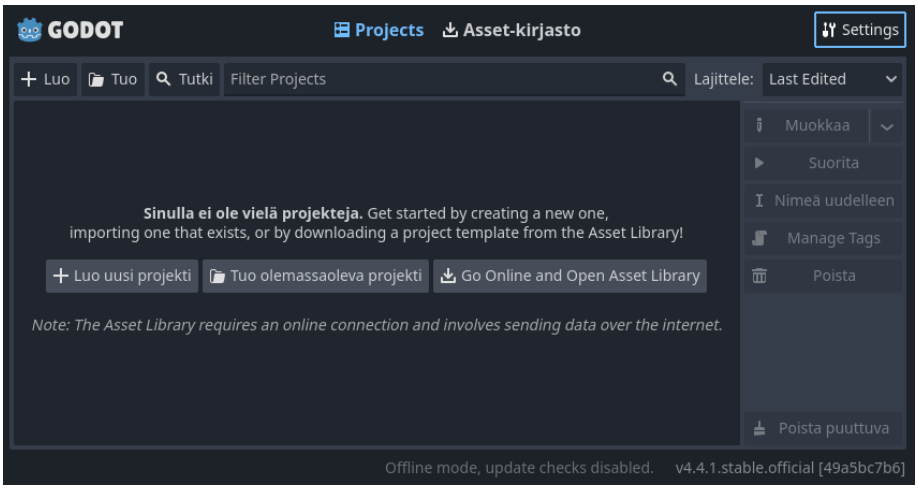


Tallenna kuvasi valitsemalla  ,anna kuvalle nimi kohtaan **Title** (mailan nimi voi olla Player ja pallon vaikka Ball) ja valitse lopuksi **Save as .piskel** tai **Save in Browser**

Lataa kuva koneelle valitsemalla  , sen jälkeen valitse **png** ja lopuksi **Spritesheet file export: Download**

Vaihe 2: Peliprojektin aloittaminen Godot -editorissa

Askel 1: Käynnistä Godot -editori



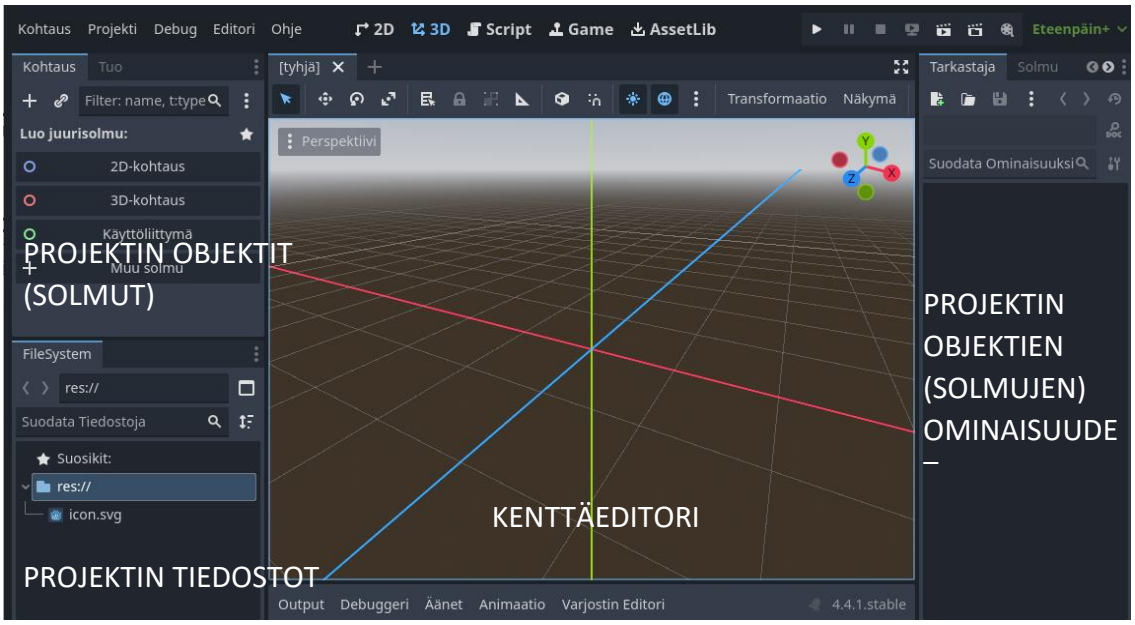
Askel 2: Luo uusi projekti

Anna projektille nimi. Esimerkiksi: Pong

Vaihda projektin poluksi oman OneDriven kansio

/Users/etunimi.sukunimi/OneDrive - Petajaveden kunta,Sivistystoimi/Tvt

Valitse **Luo**



Vaihe 3: Peliprojektin asetukset

Askel 1: Vaihdetaan asetukset sopivaksi projektia varten. Mene ylävalikosta kohtaan Projekti ja valitse Project Settings…

Mene vasemmasta palkista kohtaan **Display → Window**

Vaihda pelin ikkunan kooksi seuraavat
Viewport Width: 512 ja **Viewport Height:288**

Vaihda kohdasta **Size→Mode: Fullscreen** (kokoruutu-tila)

Vaihda kohdasta **Stretch→Mode: viewport** (peli skaalautuu oikein kokoruututilaan)

Mene vasemmasta palkista kohtaan **Rendering→Textures**

Vaihda kohdasta **Default Texture Filter: Nearest** (pikseligrafiikka näkyy oikein)

Mene vasemmasta palkista kohtaan **Rendering→Enviroment**

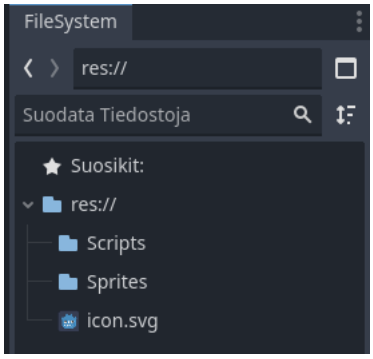
Vaihda kohtaan **Default Clear Color: musta**

Askel 2: Siirretään luodut kuvat Godotin tiedostojärjestelmään

Luodaan godotin käyttöön 2 uutta kansiota.

Klikkaa **res://** kohdassa **hiiren oikealla→Luo Uusi→Kansio…**

Nimeä ensimmäinen kansio nimellä **Sprites** (kuvat) ja toinen kansio nimellä **Scripts (**koodit)



Klikkaa **Sprites** kansiosta oikealla **hiiren klikkauksella → Avaa tiedostonhallinnassa**

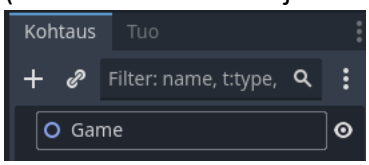
Siirrä vaiheessa 1 tekemäsi kuvat **Playe**r ja **Ball** avattuun Sprites kansioon

Vaihe 4: Peliprojektin esineet/objektit (solmut)

Askel 1: Juurisolmun luominen

Mene vasemmalle ja valitse niin kutsutuksi juurisolmuksi **2D-kohtaus**

nimeä muodostunut **Node2D** nimellä **Game**
(oikea hiiren klikkaus ja Nimeä uudelleen / kaksoisklikkaa solmua)



Askel 2: Luodaan keskiviivan solmut. Klikkaa + näppäintä

Luo keskiviiva käyttämällä seuraavanlainen solmu hakemalla: **ColorRect** (lopuksi valitse Luo)

Nimeä se vaikka nimellä **MiddleLine**

Luo pelialueen keskelle valkoinen viiva siirtämällä ja muokkaamalla pelialueelle tulleesta valkoisesta neliöstä

Tarkan keskikohdan voi löytää seuraavasti.

MiddleLine valittuna

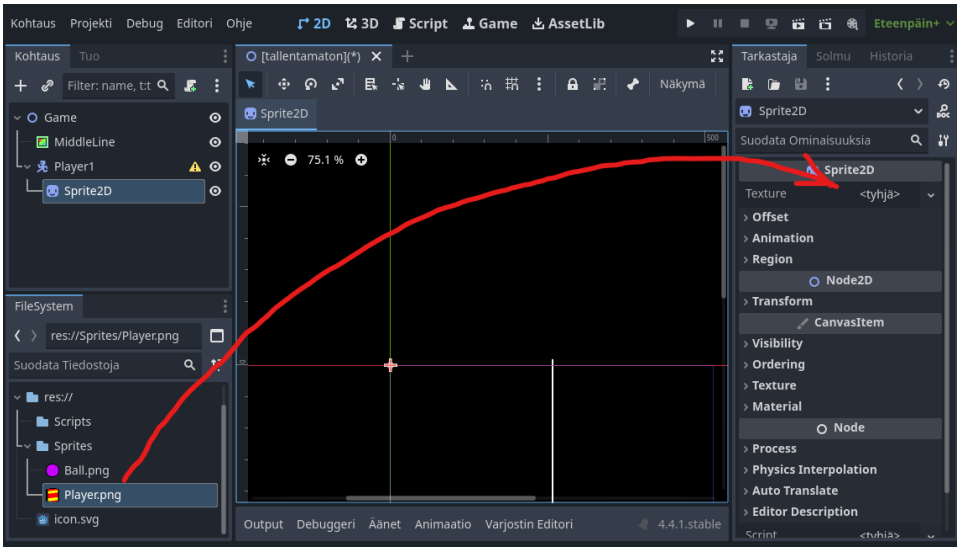
Oikealta palkista kohdasta
Layout→Transform→Position→X: 512/2 (enter)

Askel 3: Luodaan pelaaja 1 solmu. Valitse ensin Game-solmu ja sen jälkeen klikkaa samaa + näppäintä.

Hae: **CharacterBody2D**

Nimeä solmu nimellä: **Player1**

Luo tälle oma alisolmu. Pidä Player1 valittuna ja klikkaa + Hae **Sprite2D** ja solmun luomisen jälkeen rahaa Sprites kansiosta Player.png kohtaan oikealta **Texture**

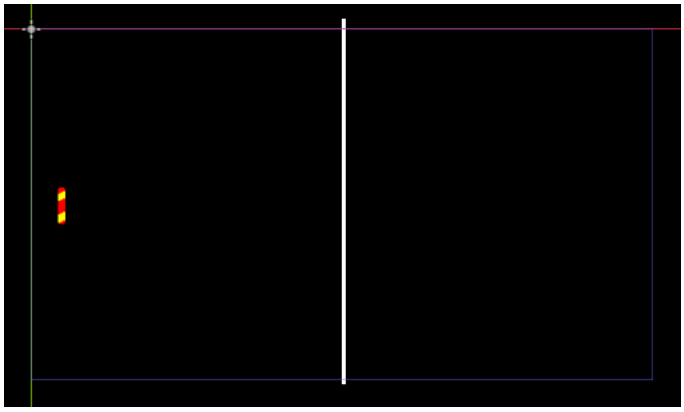


Siirrä pelaaja 1:n maila vasemmalle keskelle

Voit myös vaihtaa oikealta kohdasta **Transform→Position**

x: 25 (enter)

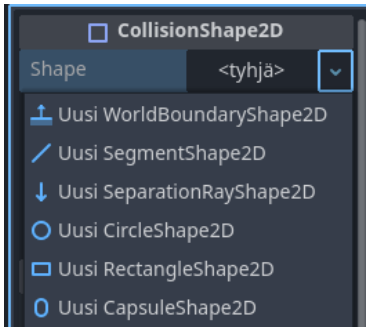
y**: 288/2** (enter)



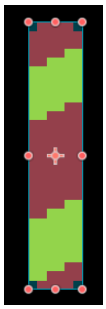
Lisätään vielä törmäystunnistin. Valitse Player1 ja lisää alisolmu +:lla

Hae: **Collissionshape2D**

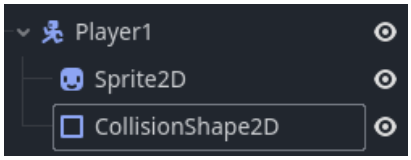
Collissionshape2D valittuna, mene oikealta kohtaan **Shape** ja valitse siihen **Uusi RectangleShape2D**



Muokkaa törmäystunnistin vastaamaan pelaajan mailan muotoa



HUOM: Solmurakenteen Player1:lle pitää näyttää seuraavalta



Askel 4: Luo pelaaja 2 nimeltä Player2

Toista pelaaja 1:n ohjeet pelaaja 2:lle

Siirrä Player2 oikealle kohdasta **Transform→Position**

x: 512-25 (enter)

y: 288/2 (enter)

HUOM: Solmurakenteen Player2: pitää näyttää seuraavalta



Askel 5: Luo pelin pallo nimellä Ball

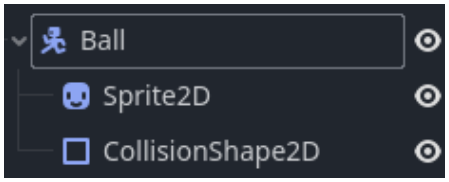
Toista edelliset ohjeet

siirrä Ball pelikentän keskelle

x: 512/2 (enter)

y: 288/2 (enter)

HUOM: Solmurakenteen Player2:n pitää näyttää seuraavalta



Askel 6: Luodaan seinät pelikentän sivuille

Luo **Game-solmulle** alisolmu + näppäimen kautta ja hae: **StaticBody2D**

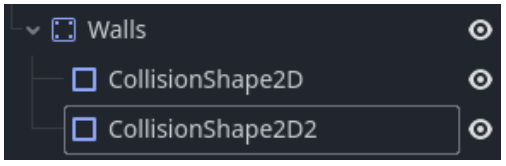
nimeä solmu nimellä **Walls**

Lisää tälle kaksi uutta **CollisionShape2D** alisolmua

CollissionShape2D solmuille lisää oikealta **Shape** kohdasta **Uusi SegmentShape2D**

Laita toinen peliruudun yläosan reunaksi ja toinen alaosan reunaksi

HUOM: Solmurakenteen Walls-solmulle näyttää seuraavalta

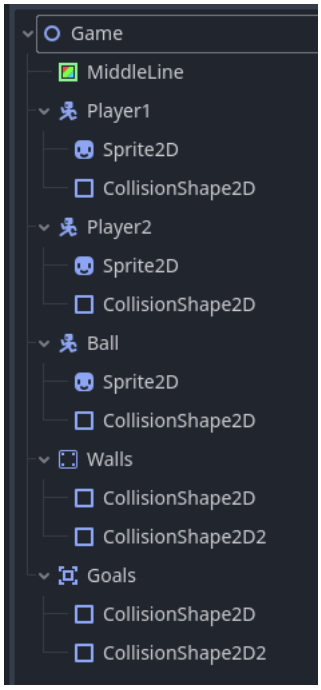


Askel 7: Luodaan maalit

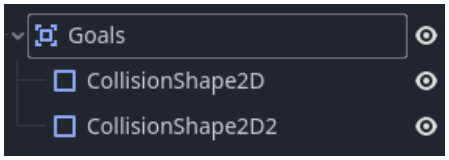
Luo Game-solmulle alisolmu + näppäimen kautta. Hae: **Area2D.** Nimeä solmu nimellä **Goals**

Lisää kaksi **CollissionShape2D** ja laita molemille **Shape** kohtaan **SegmentShape2D**

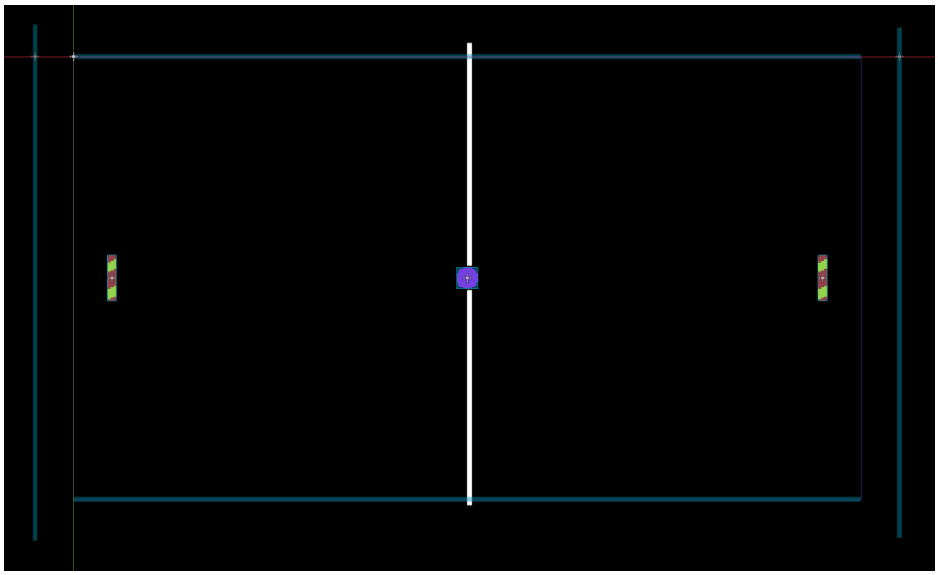
Laita toinen maaliksi vasemmalle puolelle ja toinen maaliksi oikealle puolelle



HUOM: Solmurakenteen Goals-solmulle näyttää seuraavalta



Askel 8: Valmiin pelikentän pitäisi näyttää seuraavalta



Vaihe 5: Peliprojektin näppäinasetukset

Askel 1: Lisätään asetuksiin peliprojektin näppäinasetukset. Mene ylävalikosta kohtaan Projekti ja valitse ylhäältä Syötekartta

Kirjoita kohtaan Lisää uusi toiminto: **P1_up**

hyväksy painamalla Enter

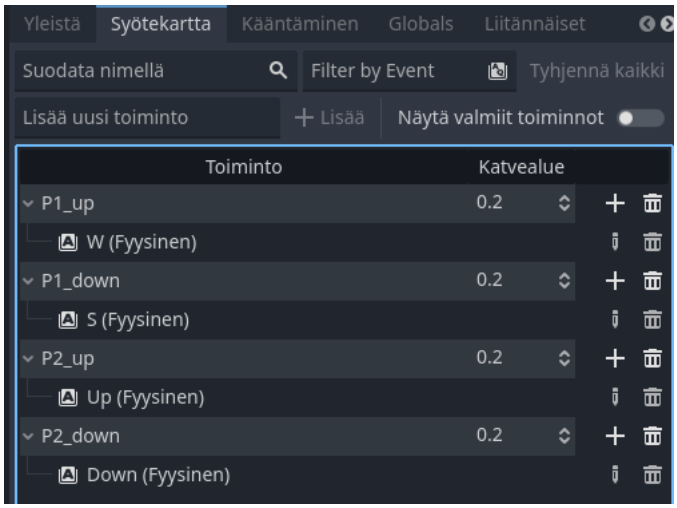
Paina muodostuneesta rivistä oikealta +

Paina näppäimistöltä **W** ja klikkaa valintaa OK

Muodosta sama **P1_down** ja laita tälle näppäimeksi **S**

Muodosta vielä **P2_up ja P2_down** ja laita nämä **ylös ja alas nuolinäppäimiin**

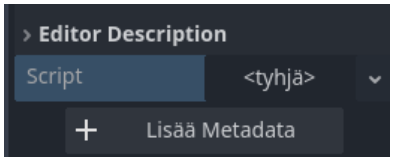
Syötekartan pitäisi nyt näyttää seuraavalta



Vaihe 6: Peliprojektin ohjelmointi

Askel 1: Player1:n ohjelmakoodi

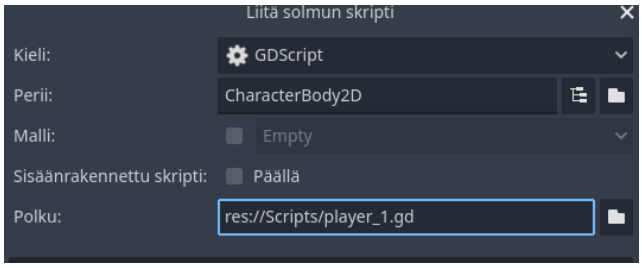
Valitse Player1 ja mene oikealta kohtaan **Script** ja luo **Uusi skripti…**



Poista valinta kohdasta **Malli**

Siirrä Polku: Scripts kansioon

Lopuksi paina Luo



Kirjoita seuraava ohjelmakoodi

```
1   extends CharacterBody2D
2
3   const SPEED = 120
4
5   func _physics_process(delta):
6       var direction = Input.get_axis("P1_up", "P1_down")
7       if direction:
8           velocity.y = direction * SPEED
9       else:
10          velocity.y = move_toward(velocity.x, 0, SPEED)
11      
12      move_and_slide()
13
```

Muutama seloste koodille

const: tarkoittaa muuttujaa, jota ei voi muokata muodostamisen jälkeen. On vakio.

SPEED: Pelaajan mailan nopeus

direction: Pelaajan mailan liikkumisen suunta. Katsotaan näppäimistä **INPUT**-komennon kautta.

velocity.y: pelaajan mailan nopeus pystysuunnassa

move_and_slide(): laittaa mailan liikkumaan velocity:ssä olevien lukuarvojen mukaisesti

Askel 2: Player 2:n ohjelmakoodi

Seuraa Player1 -ohjetta. Muista muuttaa koodissa näppäimet pelaaja 2:lle sopivaksi

Askel 3: Ball:n ohjelmakoodi

Seuraa Player1:n ohjetta. Kirjoita seuraava **Ball**:n ohjelmakoodiksi

```
1   extends CharacterBody2D
2
3   const DIRECTION = [-1,1]
4   var speed = Vector2(40*DIRECTION.pick_random(), 60*DIRECTION.pick_random())
5
6   func _physics_process(delta):
7       var collission = move_and_collide(speed*delta)
8       if collission:
9           speed = speed.bounce(collission.get_normal())
10          speed = 1.05*speed
11
```

Muutama seloste koodille

DIRECTION.pick_random(): määrätään pallon kulkusuunta satunnaisesti
-1 negatiivinen suunta akselilla ja 1 positiivinen suunta akselilla

speed: pallon nopeus x-akselilla ja y-akselilla

move_and_collide: liikuttaa palloa ja katsoo, tuleeko törmäystä. Tallennetaan törmäystieto muuttujaan **collission**

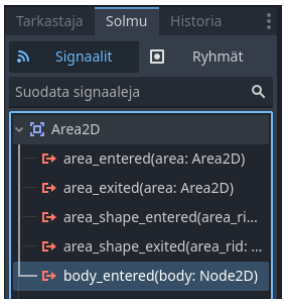
speed.bounce(): pallon kimpoaminen törmäyksessä

1.05*speed: Lisätään vauhtia kertoimen mukaan jokaisessa törmäyksessä

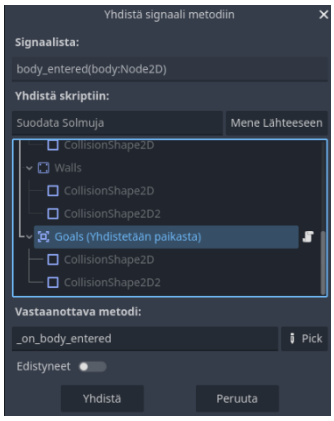
Askel 4: Maalien ohjelmakoodi

Valitse **Goals** ja luo sille skripti tiedosto kohtaan **Scripts**

Mene oikealta ylhäältä kohtaan **Solmu** ja kaksoisklikkaa **body_entered**(body: Node2D)



yhdistä skriptiin Goals



Kirjoita seuraava koodi

```
1   extends Area2D
2
3   func _on_body_entered(body):
4       get_tree().reload_current_scene()
```

Muutama seloste koodille:

Ohjelmakoodin tarkoituksena on nollata peli alkuun, kun jompikumpi pelaaja saa maalin.

Askel 5: Ohjelma koodi Game-solmulle

Luo ohjelmakoodi **Game**-solmulle ja kirjoita sinne seuraava koodi

```
1   extends Node2D
2
3   func _process(delta):
4       if Input.is_action_just_pressed("ui_cancel"):
5           get_tree().quit()
```

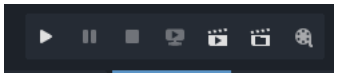
Muutama seloste koodille:

Ohjelmakoodi tarkastaa, tapahtuuko toiminto ”ui_cancel”, eli tässä tapauksessa onko Esc-näppäintä painettu. Jos näin tapahtuu, peli sammutetaan.

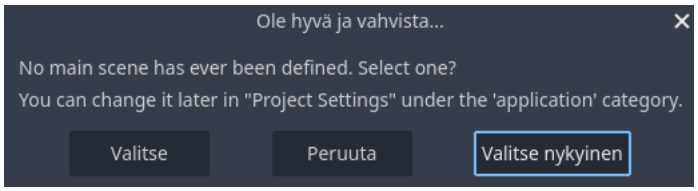
Vaihe 7: Pelin testaaminen

Askel 1: Peliä voi testata ylhäältä nuolipainikkeesta

HUOM: Aina testatessa peliprojekti tallentuu automaattisesti



Askel 2: Ensimmäisellä testauskerralla peli kysyy pääkohtausta. Valitaan siihen nykyinen



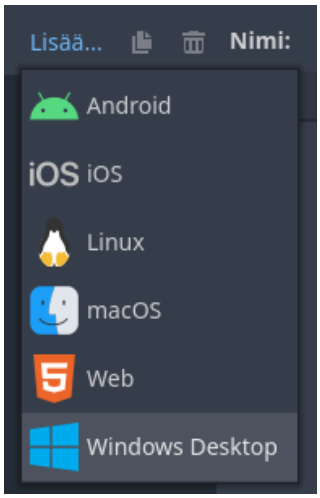
Klikkaa avautuvasta ruudusta Tallenna

Vaihe 8: Pelin luonti pelattavaksi tiedostoksi

Askel 1: Jotta pelistä saa muillekin pelattavan version, täytyy peliprojektista luoda vielä pelattava tiedosto

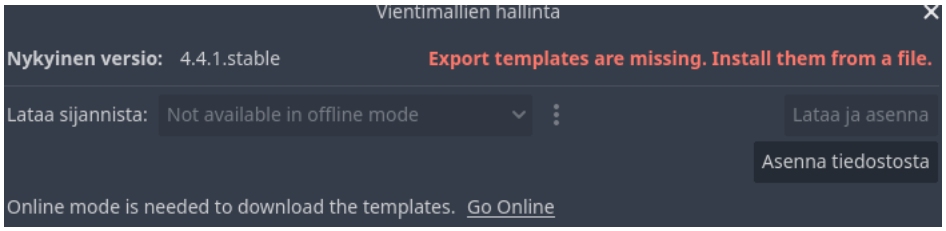
Valitse ylävalikosta **Projekti→Export…**

Lisää **Windows Desktop**



Ensimmäistä kertaa pelitiedostoa luotaessa täytyy vientimalli hakea. Valitse: **Hallinnoi vientimalleja**

Paina asenna tiedostosta. Etsi vientimallit Godotin asennuskansiosta.



Sulje ja mene uudestaan kohtaan **Projekti→Export…**

Valitse **Vie Projekti**

Voit vielä tarvittaessa vaihtaa projektin nimeä

Tallenna