Godot Engine Tutorial
Creando un personaje de plataformas
Parte I

En este tutorial veremos cómo crear nuestro personaje del estilo plataformero (Donkey Kong, Mario Bros., Super Meat Boy) dentro del motor de videojuegos Godot.
En la primera parte creamos el esqueleto del proyecto.
En la segunda parte implementaremos el movimiento del personaje.
Aunque implementaremos el control de un personaje 2D el mismo principio se puede aplicar en un personaje 3D.

Comenzando

Mirando algunos personajes famosos como Mario Mario (de Mario Bros.) o Sonic the Hedgehog sabemos que se mueven en forma horizontal y vertical en la pantalla, que caminan sobre plataformas o el suelo de acuerdo a las reglas del juego. Estas reglas incluyen la gravedad, inercia, velocidad, etc. También sabemos que el personaje puede subir por escaleras, flotar sobre el agua y demás locuras.

¿Pero cómo es que se implementan estos controles tan complejos? Cada videojuego es un universo completamente diferente, y cada desarrollador puede elegir entre millones de maneras para manejar estos aspectos técnicos del juego.

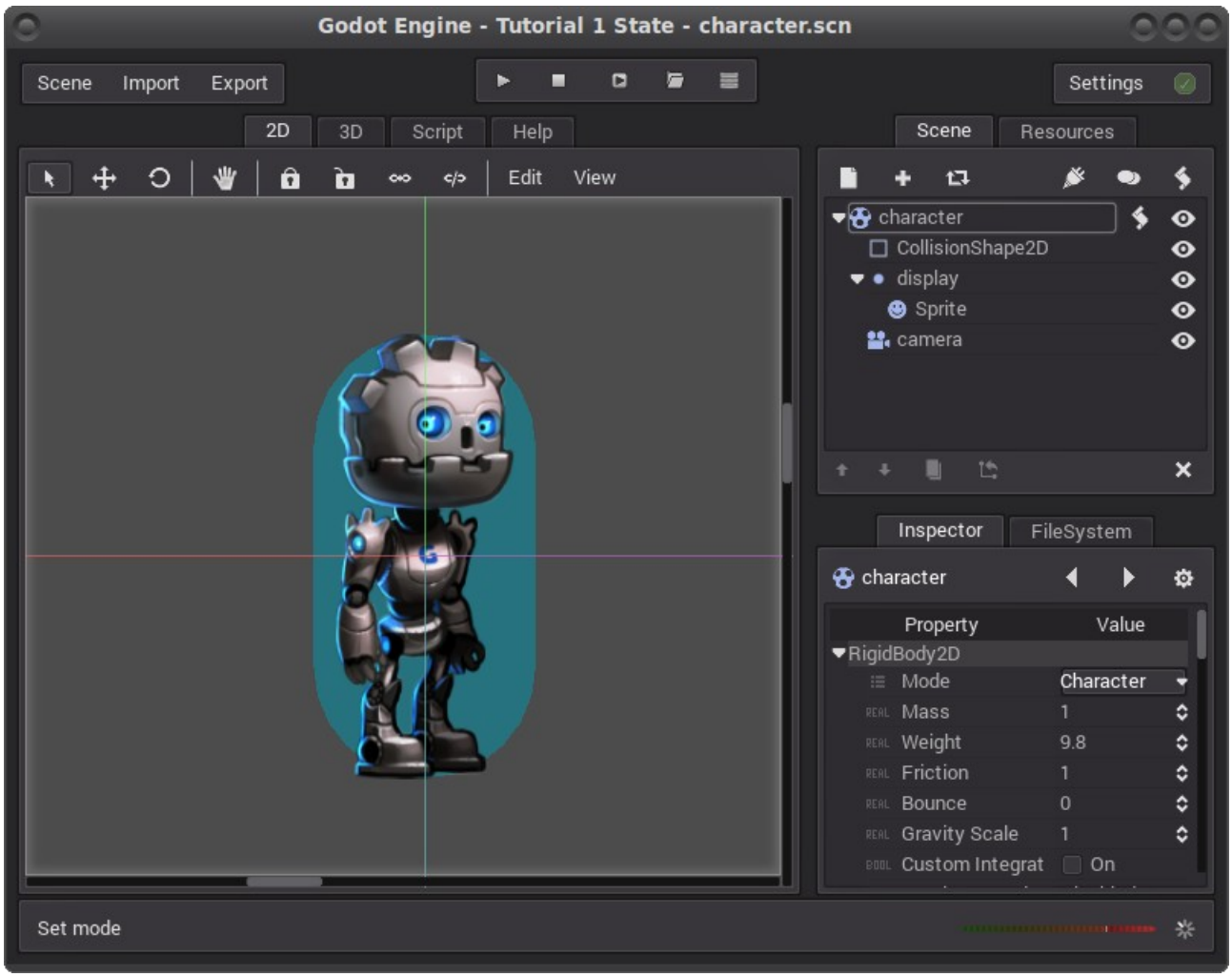
En nuestro caso utilizaremos el motor Godot, el cual nos da una gran ayuda al proporcionarnos un **motor de físicas** que nos servirá para representar nuestro personaje y el mundo que lo rodea. Nosotros ponemos nuestro personaje dentro del mundo físico y el motor se encarga de hacer las simulaciones: moverlo de acuerdo a las fuerzas físicas como la gravedad y manejar las colisiones con otros objetos y con las plataformas de nuestro nivel.

El objetivo de este tutorial no es explicar a fondo el motor de físicas en Godot Engine, asi que si tienes problemas entendiendo esta parte del motor te recomiendo mirar algunos tutoriales en YouTube sobre el tema o visitar la wiki official en https://github.com/okamstudio/godot/wiki.

Nuestro proyecto se compone de tres escenas principales:

- *character.scn* el personaje
- *platform.scn* las plataformas donde el personaje camina
- *level.scn* nuestra escena principal que se ejecutara al iniciar el juego

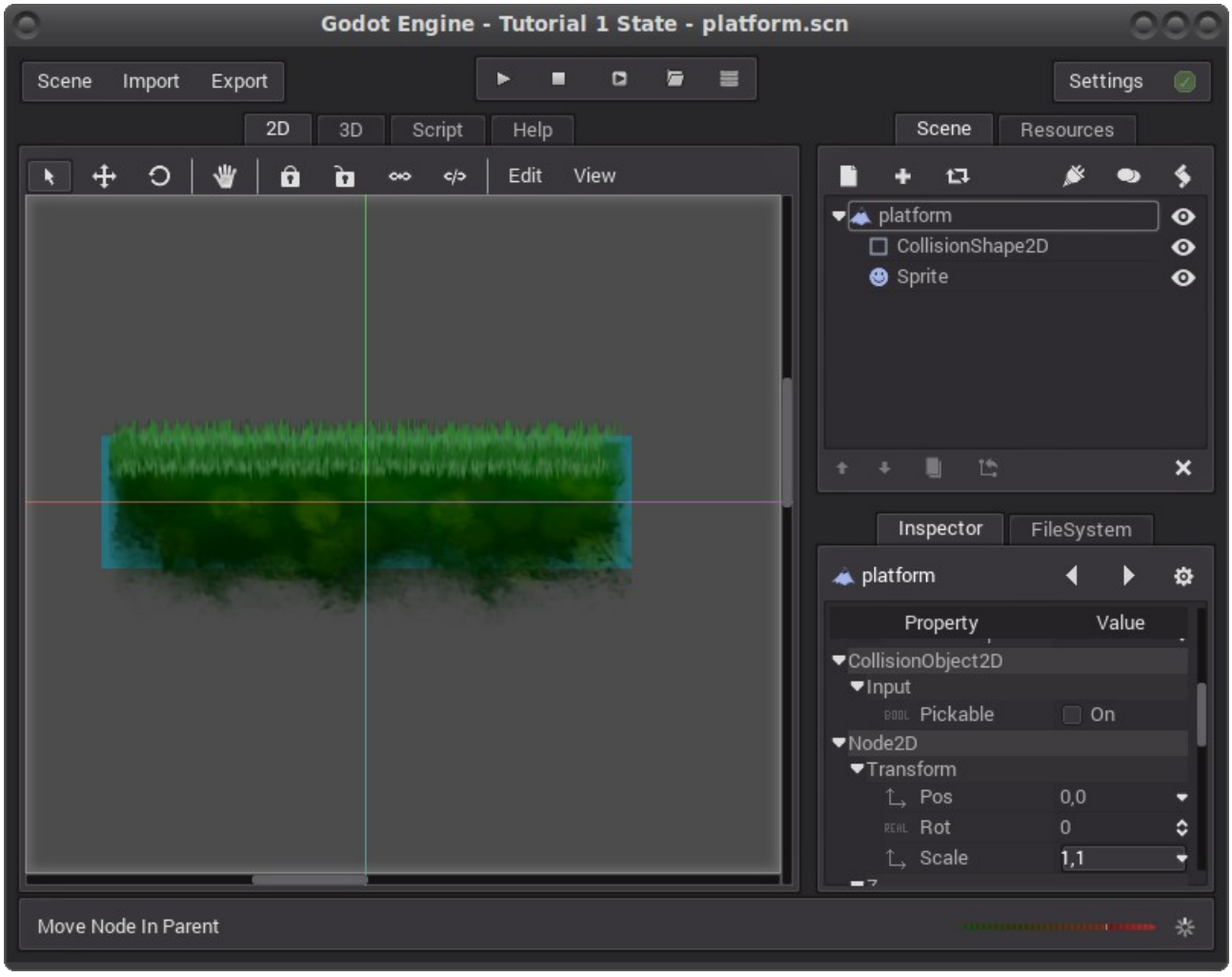
Nuestro Personaje: character.scn



Tenemos una escena llamada *character.scn* compuesta por:

- Un cuerpo rígido (RigidBody2D) como raíz de la escena. Notar que seteamos el atributo Mode a Character, esto hace que el personaje se mantenga erguido y no rote como lo harían otros cuerpos.
- Una cápsula de 32x64 como colisión para el cuerpo.
- Un nodo (Node2D) usado para agrupar todo los nodos “visuales” (sprites)
- Un sprite (Sprite2D) donde cargamos la imagen de nuestro personaje (diseñado por Andreas Esau para su serie de tutoriales en YouTube)
- Una cárama (Camera2D) usada como cámara principal para ver nuestro juego

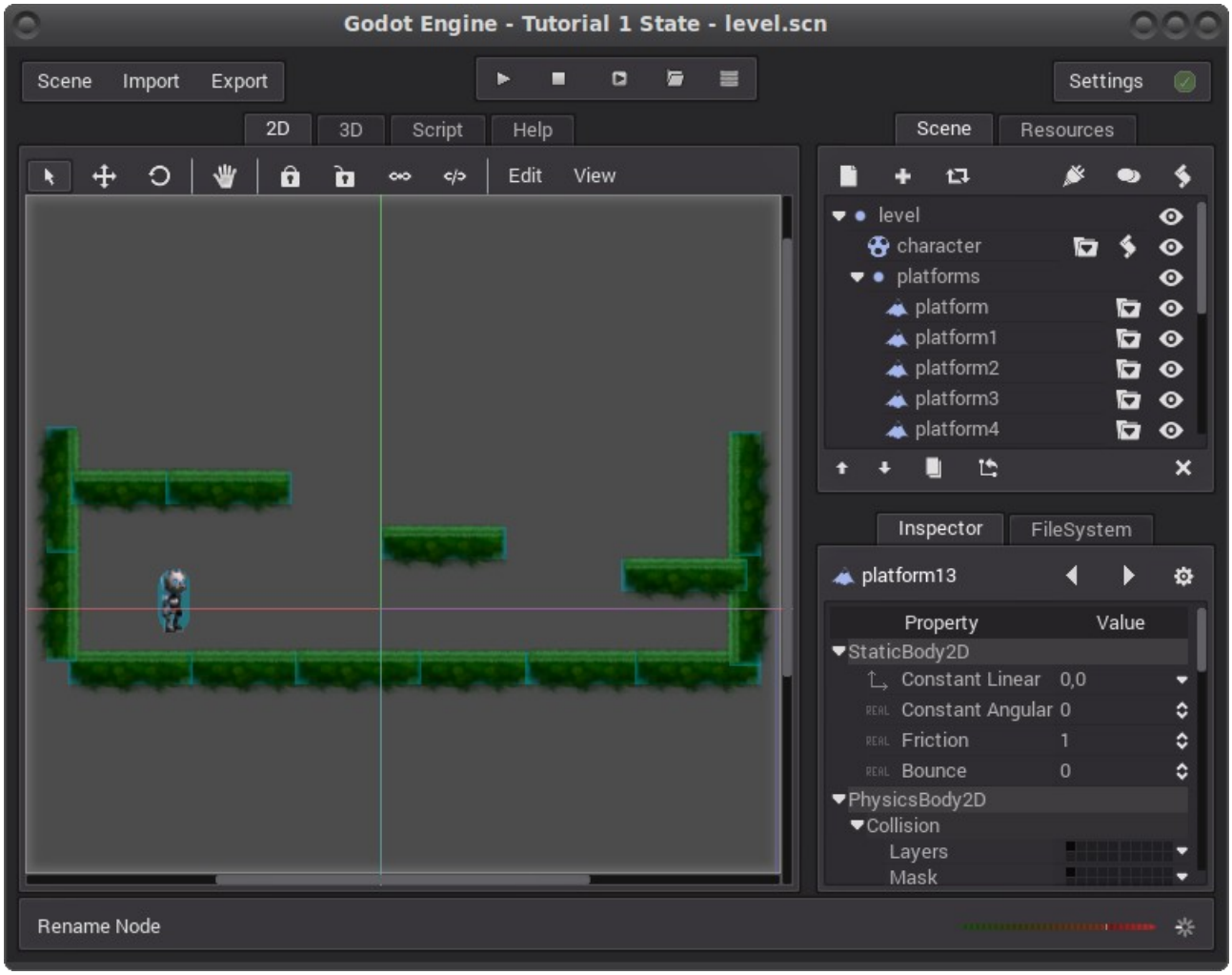
Nuestra Plataforma: platform.scn



Tenemos la escena *platform.scn* para representar nuestras plataformas compuestas por:

- Un cuerpo estático (StaticBody2D) como raíz de la escena
- Un rectángulo de 128x32 como colisión para el cuerpo
- Un sprite (Sprite2D) donde cargamos la imagen de una plataforma

Nuestro Nivel: level.scn

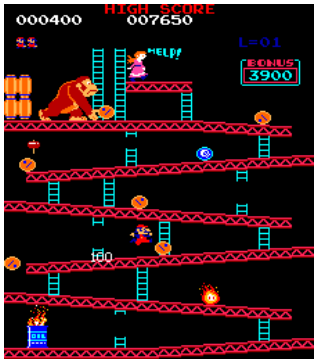


Por último la escena *level.scn* contiene:

- un nodo raíz (Node2D)
- nuestro nodo personaje *character.scn*
- un nodo (Node2D) para agrupar las plataformas
- un conjunto de plataformas *platform.scn*

Notar que para que la simulación se viera mejor tuve que modificar el valor normal de la gravedad por 980. ***Scene > Project Settings > Physics 2d > default_gravity = 980***

¿Moviendo el personaje?



Si ejecutamos nuestro juego veremos que el personaje cae por acción de la gravedad colisionando con las plataformas y continúa en estado de reposo hasta la eternidad.
Para cambiar esto haremos que el personaje reaccione a los eventos de teclado moviéndose de acuerdo a las teclas presionadas.

Para realizarlo editamos la escena *character.scn* y le agregamos un script al nodo raíz "character". Editamos el script y agregamos la sentencia "***set_process_input(true)***" dentro de la función de inicialización ***_ready()***.
Esto le dice al motor que nuestro nodo reaccionara a eventos de entrada del usuario utilizando la función **_input(event)**. Dentro de esta función podríamos ver si el parámetro de entrada ***event*** se corresponde al evento de "tecla izquierda presionada" o "tecla derecha presionada" y mover el personaje de acuerdo a estos eventos.

En lugar de esto nosotros usaremos **Input Actions** (acciones de input). Esta es una manera más sencilla que proporciona el motor para el procesamiento de entradas de usuario.
Para esto abrimos el panel **Project Settings (Scene > Project Settings)** y vamos a la pestaña **Input Map**. Esta pestaña guarda todos los input actions utilizados en el juego. Veremos que ya existen algunos por defecto en nuestro proyecto, pero nosotros crearemos los nuestros (***btn_left***, ***btn_right***, ***btn_jump***, botón izquierda, botón derecha y botón saltar respectivamente).

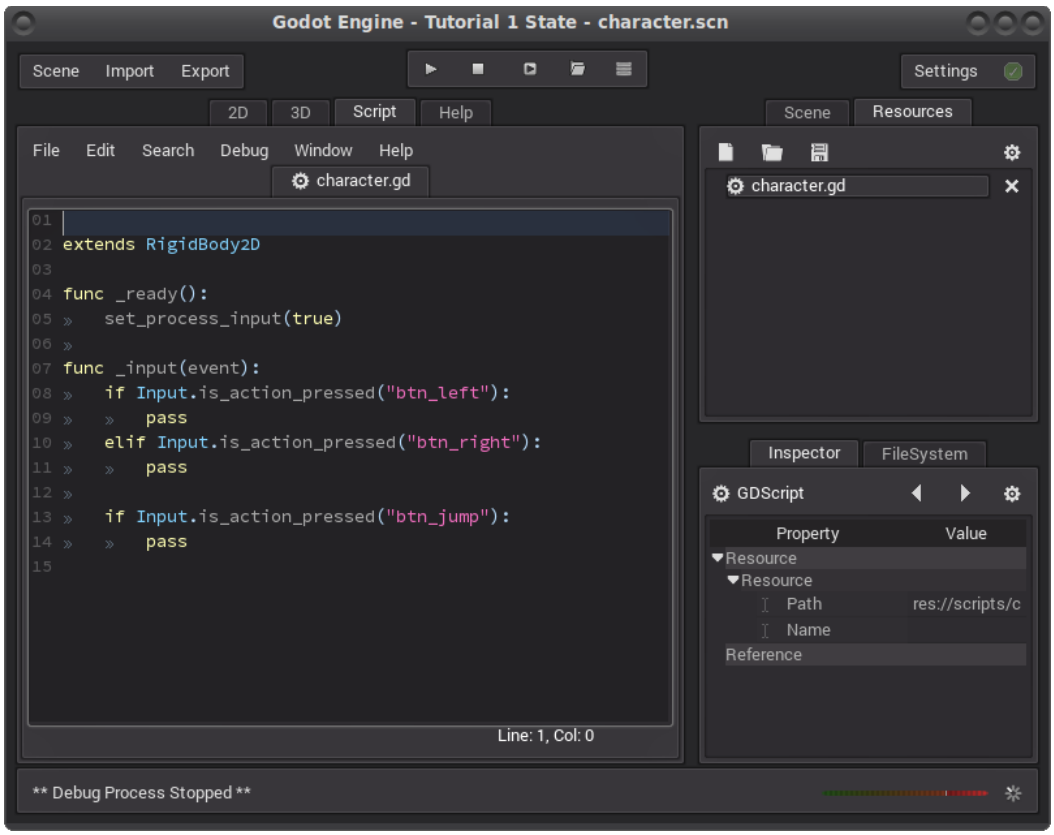
Para crear las acciones escribimos el nombre de la acción y apretamos el botón **Add**.
Vemos que se agrega nuestra acción a la lista. Lo próximo a hacer es asociar nuestra acción a un evento de entrada presionando en el simbolo **+** a la derecha de nuestro acción.

Notar que se pueden crear acciones para entradas de mouse, teclado y gamepads (nosotros solo utilizaremos entradas de teclado).

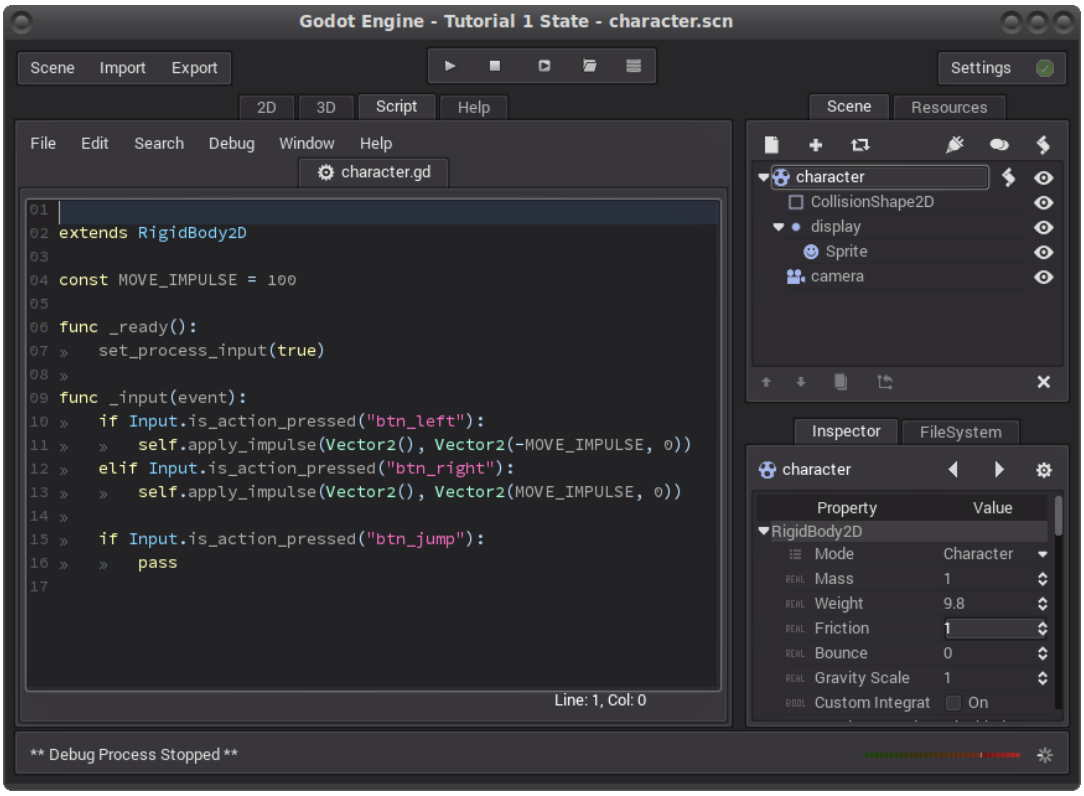
Las acciones nuestras serán:

btn_left	evento Key (tecla izquierda)
btn_right	evento Key (tecla derecha)
btn_jump	evento Key (tecla barra espaciadora)

Una vez creadas las acciones volvemos al script para nuestro personaje. Creamos la función ***_input(event)*** que reaccionará a eventos de entrada y escribimos un poco de código para manejar nuestro personaje.



En este momento podríamos aplicar un impulso a nuestro personaje de acuerdo a la tecla presionada utilizando el método *apply_impulse(posicion, impulso)*.



Si lo hacemos vemos que el personaje se mueve de manera “extraña”. Esto es debido a:

- El personaje acelera sin límite. Debemos decirle al cuerpo que deje de

acelerar al alcanzar cierta velocidad
- El personaje tarda en comenzar a moverse. Esto se debe a que el cuerpo está generando una fricción contra el suelo.

Nosotros utilizaremos otra técnica para obtener más control sobre el movimiento del personaje.

Fin de la Primera Parte (podes tomarte un descanso)

Descargar Parte I: zip con el esqueleto del proyecto

Contacto:

Federico Pacheco
@fede0d
http://fede0d.github.io/