Pygame importieren	`import pygame`	Ohne das Modul pygame zu importieren, können wir die Methoden, um das Spiel zu machen, nicht benutzen.
Einen Bildschirm erstellen	`screen = pygame.display.set_mode(size)`	Das Spielfeld wird mit der Größe aus dem Tuple size gemacht.
size	`size = width, height = 640, 480`	Das Tuple size, das für die Auflösung des Fensters gebraucht wird.
Tuple	`foo = (a, b) oder foo = a, b`	Eine Liste, aber unveränderbar und speicherplatzsparender.
Spriteklasse erstellen	`class Foo(pygame.sprite.Sprite):` `  [...]`	Erstellt eine neue Spriteklasse (zum Beispiel ein Tier).
RenderClear (Erstellung)	`foos = pygame.sprite.RenderClear()`	Eine Gruppe von Sprites einer Art (z.B. Tiere, Blätter).
Sprite zu s.o. hinzufügen	`foos.add(foo1)`	Fügt einen Sprite zu einem RenderClear hinzu.
Sprite (Erstellung)	`foo1 = Foo(arg1, arg2)`	Ein Objekt, das PyGame behandeln und testen kann.
Dictionary (Erstellung)	`{ "foo":bar, x:bar2, 1234:"hallo" }`	Eine Liste, in der Einträge mit einem "Key" aufgerufen werden, anstatt mit einem Index.
Hauptschleife	`while True:` `  [...]`	Wiederholt das Hauptprogramm, das alles abhandelt, damit es nicht nur einmal ausgeführt wird.

Event (Abfrage)	`event = pygame.event.poll()`
	`if event.type == pygame.KEYDOWN:`
	`  [...]`
		Events treten z.B. bei Tastendruck auf

Die RenderClear-Methoden clear, update, draw benutzen	`clear(bildschirm, hintergrund)`
	`update(*args)`
	`draw(bildschirm)`
		clear übermalt ein Sprite mit dem Teil des Hintergrunds, auf dem es sich gerade befindet. update fuehrt die update-Methode aller Sprites in der Gruppe aus. draw malt alle Sprites in der Gruppe an ihrer Stelle auf den Bildschirm.

Display updaten	`pygame.display.update()`	Aktualisiert den ganzen Bildschirm (am Ende der Schleife).
Kollisionsabfragen (spritecollide)	`pygame.sprite.spritecollide(sprite, renderclear, kill)`	Testet die Kollision zwischen einem Sprite und einer Spritegruppe und löscht alle kollidierenden Sprites der Gruppe, wenn kill gleich True ist.
Kollisionsabfragen (groupcollide)	`pygame.sprite.groupcollide(renderclear1, renderclear2, kill1, kill2)`	Das Gleiche wie spritecollide, nur testet groupcollide die Kollision zwischen zwei RenderClears.

Sound	`pygame.mixer.init()`
	`foosound = Sound("foo.wav")`
	`foosound.play()`
		Macht einen Sound und spielt ihn ab.

Nicht vergessen die jeweiligen Module zu importieren, wenn ihr zum Beispiel spritecollide, groupcollide, oder mixer benutzt!