Project-Maryo Rehber V1.0

[+] Project-Maryo Nedir ?

Project-Maryo, bildiğimiz Mario'nun basitleştirilmiş versiyonudur. Bu projenin amacı, giriş+ seviyesindeki hackerların severek yapabileceği, yaparken de bir çok konuda bilgi sahibi olabileceği bir proje ortaya koymaktır. Projeyi yaparken bir an önce bitirmeye odaklanılmamalıdır. Öğrenmeye, mantığı kavramaya odaklanılmalıdır. Okumakta olduğunuz rehber sadece yol göstericidir. Rehberin yanında internetten yapılacak araştırmalar ile konular hakkında daha derin bilgi edinilebilir. Rehber, Python, PyCharm ve PyGame üzerinden ilerlemektedir.

UYARI : Bu proje bir "kodlama ödevi" değildir. Lütfen bu projeyi gerçekleştirirken, mümkün olduğunca kendinizden bir şeyler katın. Burada söylenileni aynen yapmak zorunda değilsiniz, resim çizer gibi kod yazmaya çalışın. Yazdığınız kod, ona baktığınızda sizi mutlu etsin :)
bkz. Hackers & Painters

[+] Python, PyCharm ve PyGame Kurulumu

[1] Python kurulumu

Python 2 veya Python 3 kurabilirsiniz. Hangisini kuracağınız tamamen sizin isteğinize ve alışkanlığınıza bağlıdır. Kararsızsanız 3 kurabilirsiniz. Linux kullanıyorsanız Python büyük ihtimal yüklüdür. Windovz :(kullanıyorsanız,
https://www.python.org/
sitesinden indirip kurabilirsiniz.

[2] PyCharm geliştirme ortamının kurulumu

Sitesinden indirip kurabilirsiniz,
https://www.jetbrains.com/pycharm/

[3] PyGame kurulumu

Bu aşamada PyCharm'a girip projenin oluşturmuş olunması gerek. PyCharm'dan projeyi oluşturduktan sonra projeyi açıp "*To install a package*" kısmında yazan adımları izleyin,
https://www.jetbrains.com/help/pycharm/installing-uninstalling-and-upgrading-packages.html
Paket arama kısmına "*pygame*" yazarsanız pygame modülünü göreceksiniz. Tek tık ile kolay bir şekilde kurulum yapabilirsiniz.

[+] H{AC}K TI{M}E !

Proje aşama aşama gerçekleştirilecek ve her bir aşamada, projeye ana özelliklerinden biri eklenecek. Projenin sonunda, sağa sola gidebilen, zıplayabilen bir karakterimiz olacak. Bölüm tasarımında ise, karakterin üzerinde durabileceği zemin ve karakterin düşünce öleceği boşluk kısımlar olacak. Bu özellikler kullanılarak çok farklı bölümler, hatta oyunlar yapılabilir. Bonus olarak basit bir level editörünün nasıl yapılabileceği hakkında bilgi verilecek. Proje sonunda, geliştirilebilir yapıya sahip bir temel ve bu temeli kullanabilecek bilgiye sahip olacağız. Buradan sonra kendi oyununuzu yapmamak için hiç bir neden yok. Bir nevi projenin sonu, kendinize ait daha büyük bir projenin başlangıcı olacak :) Bonus 2 kısmında projeye ne tür geliştirmeler uygulanabileceğine hakkında fikirler olacak.

PyGame hakkında dökümantasyona buradan ulaşabilirsiniz : https://www.pygame.org/docs/

maryolib.py dosyasını, sizin kendi Python dosyanız ile aynı dizine koyup projenize “import”lamanız gerekmektedir. maryolib.py dosyasında projenin karışıklığını azaltmak için yazılmış bazı kodlar mevcuttur.

maryolib.py ve başlangıç için gerekli diğer dosyalara aşağıdaki linkten ulaşabilirsiniz. Dosyayı indirin ve bir dizine çıkartın. “starter-pack” adlı klasörün içinde gereli tüm dosyalar bulunmaktadır.
https://github.com/furkantokac/Project-Maryo/archive/master.zip

[00] PyGame temel yapısı ve ilk temas

Aşağıdaki kodun iyi incelenerek her satırının anlamaya çalışılması gerekiyor. “*print (event)*” kısmına dikkat edin ve event’in ne olduğunu kavramaya çalışın. Farklı şeyler denemekten, kodu değiştirmekten korkmayın! Örnek kodlar mümkün olduğunca basit yazılmaya çalışıldı. Kodu fonksiyonlara, hatta objelere bölerek çok daha güzel bir program elde edebilirsiniz.
https://github.com/furkantokac/Project-Maryo/blob/master/starter-pack/[00].py

[01] Grafik oluşturmak ve kullanmak

Oyunu yaparken ekrandaki her şey bu aşamada yaptığımız şekilde resimlerden oluşacak.
https://github.com/furkantokac/Project-Maryo/blob/master/starter-pack/[01].py

[02] Resmi hareket ettirme

Aşağıdaki kodda oluşturduğumuz “maryo”, sağ ok tuşu ile sağa hareket ediyor. Siz de sol tuşa basıldığında sola gidecek şekilde kodu tamamlayın.
https://github.com/furkantokac/Project-Maryo/blob/master/starter-pack/[02].py

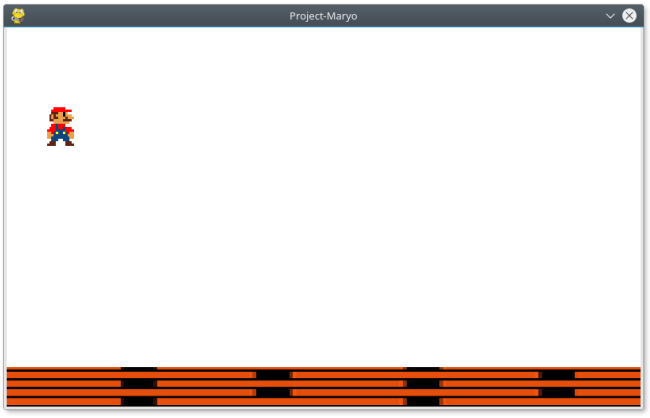
[03] Temas algılama (collision detection)

Mesela bir resmi sağa doğru hareket ettirirken önünüze duvar çıktı. Ne yapmanız gerek ? Durmanız gerek. Peki nasıl ? Temas algılayarak. Aşağıdaki kodda Maryo ve duvar oluşturuluyor. Maryo’muz sağa doğru hareket ediyor fakat duvara temas ettiğinde daha ilerleyemiyor.
https://github.com/furkantokac/Project-Maryo/blob/master/starter-pack/[03].py

[04] Checkpoint !

00-03 arasında öğrendiklerimiz aslında projenin temel kısmını oluşturuyor. Tüm projeyi bu kısımlarda öğrendiklerimizi farklı şekillerde kullanarak oluşturacağız. Yani asıl proje şimdi başlamakta. Yeni bir python dosyası oluşturup onun üzerinden devam edelim. Bu kısımdan sonra koddan ziyade daha çok mantık verilecek çünkü 00-03 arasında gerekli kod bilgisini aldık. Bundan sonra iş onları farklı şekillerde bir araya getirmeye bakıyor. Yani hacker resim çizmeye şimdi başlıyor :)

[05] Zemini ve karakteri yapma



Bu kısımın sonunda elimizde sağa sola gidebilen bir karakter, bu karakterin alt tarafında olacak bir zemin olması beklenmekte. Yani 2 grafik olacak ve bu grafiklerin boylarıyla ve konumlarıyla oynayarak yukarıdaki gibi bir görüntü oluşturacağız. Zeminin tamamını kaplamak için şöyle bir yöntem kullanabilirsiniz: “*DISPLAY_X*” değişkeni ekranın x düzlemdeki boyutunu belirtmektedir yani genişliğini. Ekranın genişliği kadar geniş, ve belli bir yükseklikte, mesela 50 piksel yükseklikte bir zemin oluşturabiliriz.

Konumlandırırken de “DISPLAY_Y” ile ekranın yüksekliğini alabiliriz ve ekranın yüksekliği sayesinde grafiği ekranın en altına yerleştirebiliriz.

İp ucu

floor = maryolib.MaryoImageObject(DISPLAY_X, 50, “wall.png")
floor.move_to(0, DISPLAY_Y-50)

[06] Yer çekimi uygulama

Şöyle bir şey uygulayabiliriz. Karakterimiz, zemin ile temas etmediği sürece, karakterimizi aşağı doğru hareket ettirebiliriz.

İp ucu

if not pygame.sprite.collide_rect(karakterimiz, zemin) :
karakterimiz.move_down(1)

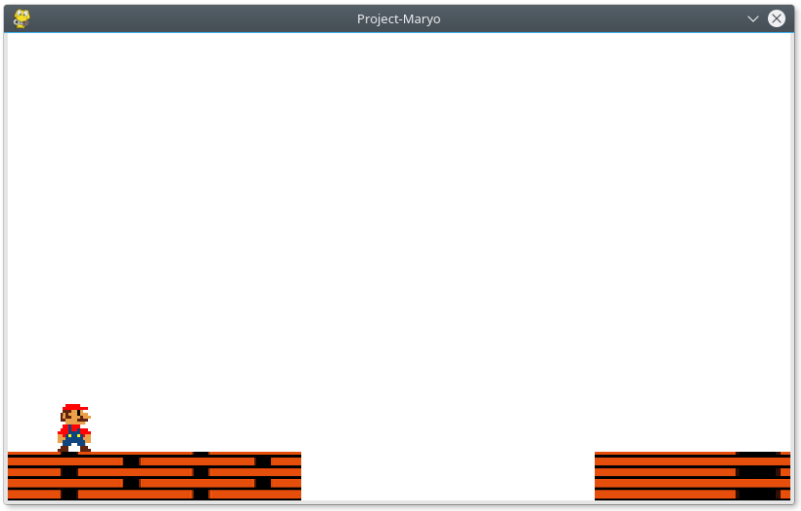
[07] Zıplamak

Zıplamak için biraz fizik uygulamamız gerekiyor. Gerçek hayatta nasıl oluyor ? Zıpladığımızda, ilk anki hızımız en fazla ve bu hız sıfır olana kadar git gide yavaşlıyor. Sıfır olduğunda en tepe noktadayız demek ve şimdi aşağı doğru hızlanmaya başlıyoruz. Biz bunu koda nasıl uygulayabiliriz ? İlk olarak zıplamayı algılayabilmek için programa bir tuş atamalıyız. Bu “space” tuşu olabilir. “Space” tuşuna basılması demek zıplamak demek. Şimdi bir değişken oluşturabiliriz, bu değişkene “down_speed” diyelim. Zıpladığımızda, down_speed’i eksili bir değer yaparsak, aşağı gitme hızımız eksi olur ve bunun anlamı yukarı gitmek demek aslında. down_speed değişkenimizin ilk değerini -20 yapalım ve karakterimize “down_speed kadar aşağı git” diyelim yani maryo.move_down(down_speed). Bu kod satırından sonra down_speed += 5 kodunu eklersek her bir döngüde ne olacak ? down_speed’in değeri, her bir döngüde şu şekilde olacak : -20, -15, -10, -5, 0, 5, 10, ….
Yani bu sayılar ne demek oluyor ? Karakterimiz ilk başta yukarı doğru 20 hızında gidecek. Daha sonra bu hız 15’e düşecek. Böyle 0’a kadar gidecek ve sonrasında aşağı doğru hızlanmaya başlayacağız. Tam da istediğimiz gibi :) Peki aşağı doğru hızlanma ne zamana kadar devam edecek ? Tabiki karakterimiz tekrardan zemine temas edene kadar.

İp ucu (kodun tamamı değildir. Mantığı anlayarak siz uygulamaya çalışın. Uygularken karşılaştığınız sorunların nedenini anlayarak düzeltmeye çalışın)

if pressed SPACE: maryo.down_speed = -20
maryo.move_down(self.down_speed)
maryo.down_speed += 5
if pygame.sprite.collide_rect(karakterimiz, zemin) : self.down_speed = 0

[08] Boşluktan düşünce oyunun bitmesi



Buradaki amaç anlayabileceğiniz üzerine zıplayabilen, sağa sola gidebilen karakterimizin aşağı düştüğünde oyunu kaybetmesidir. İlk önce grafikleri 2 zemin, 1 karakter olacak şekilde oluşturup yukarıdaki resimdeki gibi yerleştirelim. Bunun yanında ekranın en altına ekstradan oyuncu tarafından çok belli olmayacak yeni bir grafik oluşturalım. Bu grafik DISPLAY_X genişliğinde ve 5 piksel yüksekliğinde olsun ve ekranın en altına koyalım. Bundan sonra oyunumuzun ana döngüsünde, karakterimiz en alta oluşturduğumuz grafik ile temas etmiş mi etmemiş mi diye kontrol edelim. Ettiyse “print(‘GAME OVER’)”

İp ucu

if pygame.sprite.collide_rect(karakterimiz, en_alttaki_zemin)

[09] Checkpoint ! Level sistemi

Ana projenin temelini bitirmiş bulunmaktayız. Bundan sonra projeye eklenmesi gereken en önemli özellik level sistemidir. Bunun için genellikle ekranın bir kopyası 2 boyutlu dizi olarak (python’da liste) tutulur. Bu dizide level tasarımı bulunur. Daha sonra ekrana grafik çizdirilirken ilk önce dizideki bilgiler (0:1 konumunda duvar var yani level[0][1] = “duvar”, 0:2 konumunda çiçek var yani level[0][2] = “cicek” gibi bilgiler tutulacak) ekrana çizdirilir ve sonrasında karakter çizdirilir. Mesela elimizde 100x50’lük bir ekran var diyelim. Bu ekranı dizi ile temsil etmek istediğimizde oranlama yapmamız gerekiyor çünkü pikseller üzerinde çalışmak gereksiz efor sarf etmemize neden olur. Diyoruz ki, biz bu ekranı 10x5’lik bir dizi ile temsil edelim.

level = [
“0 0 0 0 0 0 0 0 0 0”,
“0 0 0 0 0 0 0 0 0 0”,
“0 0 0 0 0 0 0 0 0 0”,
“0 0 0 0 0 1 1 0 0 0”,
“1 1 0 0 1 1 1 1 1 1”]

Burada görülen 1'ler duvar oluyor, 0'lar ise boşluk. Şimdi oranlamamızı yapalım
100 / **10** -> 10
50 / **10** -> 5
Burada altın sayı 10 sayısı. Şimdi ben duvarları wall.png grafiği oluşturarak yapmak istiyorum. Her oluşturduğum duvarın eni ve boyutu 10 olmalı ki tam olarak ekrana oturabilsin. Burada anlatılan mantık kısmen şu programda geçrekleştirilmiştir : https://pythonspot.com/maze-in-pygame/

Buraya kadar her şey normal (umarım :)) Ama level bu kadar kısa olmuyor. Bunun için nasıl bir yöntem geliştirilebilir ? Buradan sonrası artık size kalmış. Biraz zaman geçtikten sonra https://github.com/furkantokac/Project-Maryo adresinde kendi yaptığımız programı paylaşacağız ve yapan tüm arkadaşların kodlarını hesapta paylaşacağız. İsterseniz oradan bizim kullandığımız yöntemi kontrol edebilirsiniz.

[Bonus 1] Level editörü
Topluluğumuzdan Furkan Ahmet Kara'nın Arduino ve Led Matrix kullanarak yaptığı Maryo projesi için yaptığı level editörü editlenerek kullanılabilir. Ayrıntılı bilgi README kısmında mevcut.
https://github.com/furkanahmetk/maryo

[Bonus 2] Geliştirmeye devam
- Level için dönen bir sopa yapılabilir. Normal Mario oyununda ateşten dönen sopalar vardı ya onun gibi. Ona deyince Maryo'muz ölebilir.
- Bölümü zorlaştırmak için üzerinde düşünülmüş bir bölüm yapılabilir.
- Level sistemi yapılabilir. Birinci level bitirildiğinde 2. si başlar.
- Menü yapılabilir. Menüde "Oyna", "Ayarlar", "Çıkış" gibi 3 seçenek olabilir.
- Mario oyunundakine benzer mantar sistemi yapılabilir. Gri renkteki dikdörtgen alındığında karakterimizin boyu uzar. Eğer zaten boyu uzunsa ateş edebilir.
- Puan sistemi yapılabilir. Sarı renkli dikdörtgenler oluşturulur, bu noktalar ile temas edildiğinde puan kazanılabilir.
- Düşman eklenebilir. Düşmanın tepe kısmına temas edildiğinde düşman yok olur ve puan kazanılır, sağ, sol veya alt kısmına temas edildiğinde maryomuz yok olur.
- Prenses yapılabilir. Bölüm sonunda prensese ulaşıldığında bölüm bitmiş sayılır.
- Çok farklı şeyler yapılabilir ! Hayal gücünüze kalmış.