Colegiul Naţional “Iancu de Hunedoara”

Building Escape

Profesor coordonator:
Stroe Andreea

Elev:
Lakatos Andreea

2019

Cuprins

Argument

Am ales să realizez acest proiect din dorinţa de învăţa lucruri noi şi din curiozitatea de a afla ce înseamnă defapt munca pentru un proiect real şi pentru a mă familiariza cu programarea propriu-zisă.

Mereu mi-au plăcut jocurile de tipul “escape room” şi am vrut să aflu dacă sunt capabilă să programez un astfel de joc. De asemenea am vrut să combin două dintre marile mele pasiuni: programarea şi matematica. Unul dintre modurile de a evada din camera este acela de a da răspunsul corect la un calcul matematic simplu; astfel uşile se vor deschide iar jucătorul va putea “evada” .

Astfel am reuşit, într-un singur proiect, să îmi îndeplinesc dorinţa de a “reliza un joc cu matematică” şi, în acelaşi timp, să îmi fixez şi bazele realizării jocurilor video pentru a putea continua cu programarea şi pentru a putea evolua.

Un alt motiv pentru care am ales realizarea acestui proiect a fost dorinţa de a a-mi testa cunoştinţele, de a le îmbunătăţii şi de a ştii sigur dacă aş putea face faţă unei facultăţi în acest domeniu.

Introducere

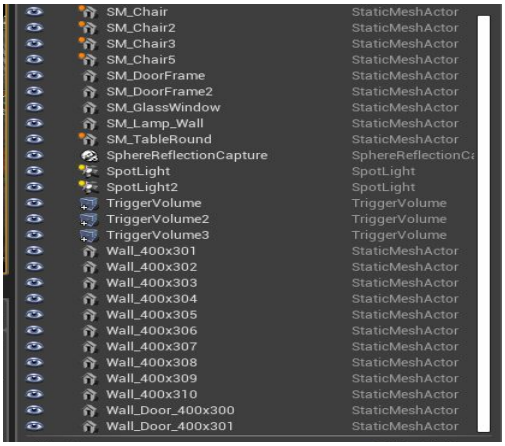
Aplicaţia Building Escape este un joc care are ca şi scop evadarea din cameră prin diferite mijloace.

Aceasta a fost realizată în Unreal Engine, iar partea de programare a fost scrisă în limbajul C++, într-un IDE(Integrated Development Environment) numit Visual Studio.

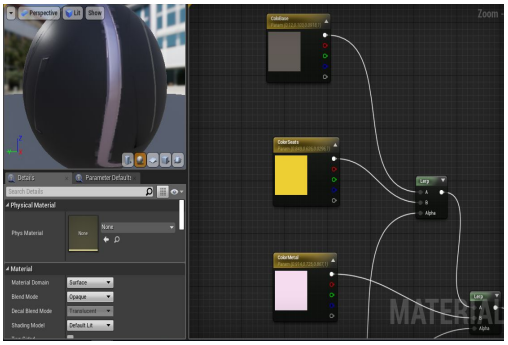
În Unreal Engine am realizat partea de grafica şi design, am aşezat pereţii astfel încât să nu existe niciun loc de scăpare, am introdus uşile şi componentele(obiecte de mobilier sau de iluminat) şi programarea vizuala de tip blueprints specifică pentru Unreal Engine 4, iar în Visual Studio am programat deschiderea uşilor, posibilitatea obiectelor de a fi mutate şi suprapuse.

Camera este formată din 12 pereţi, un tavan, o podea, dintre care doi au spaţiu pentru uşă, patru scaune, o masă şi componente de iluminat.

Textura pereţilor, a tavanului şi a podelei au fost alese astfel încât jucătorul sa-şi dorească să evadeze cât mai repede din cameră.



Culorile se pot schimba în editor în funcţie de preferinţe.

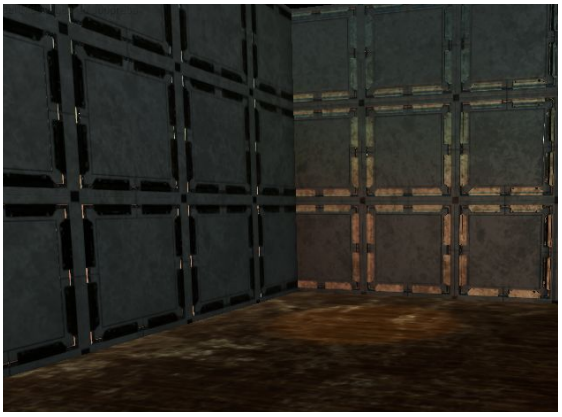



3. Aplicaţia
3.1 Reguli şi funcţionare

Jucătorul se poate mişca cu ajutorul săgeţilor pe care le are în componenţa tastaturii.

Astfel dacă jucătorul apasă pe săgeata din sus el va merge în faţă, dacă apasă pe săgeata din jos el va merge în spate, iar dacă apasă pe una dintre săgeţile din dreapta sau stânga el se va roti în dreapta sau în stânga, în funcţie de caz, iar apoi va putea merge în continuare cu săgeata din sus. Acest lucru se poate realiza şi cu mouse-ul.

Evadarea din cameră se poate realiza în două moduri. Unul dintre ele este acela ca jucătorul să ajungă în colţul din stânga cu privirea înspre uşi.

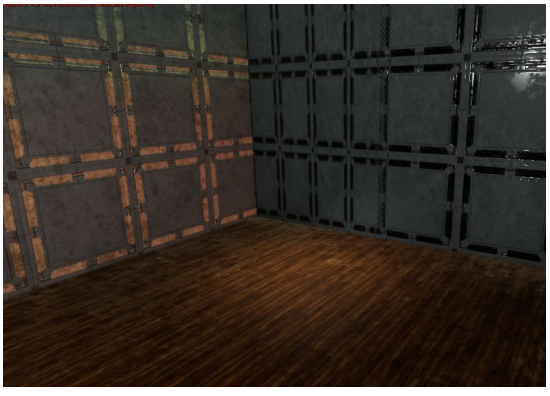


Dacă jucătorul ajunge în acest colţ uşile se vor deschide, dar când va încerca să meargă spre una dintre cele două uşi pentru a evada, ele se vor închide.

Astfel jucătorul va trebui să-şi dea seama cum să ţină această uşă deschisă.

Acest lucru este posibil doar dacă mută unele obiecte în respectivul colţ pentru ca acestea să aibă greutatea necesară pentru a putea ţine uşile deschise.

Dacă jucătorul va muta în acel colţ masa şi unul dintre scaune, uşa va rămâne deschisă.



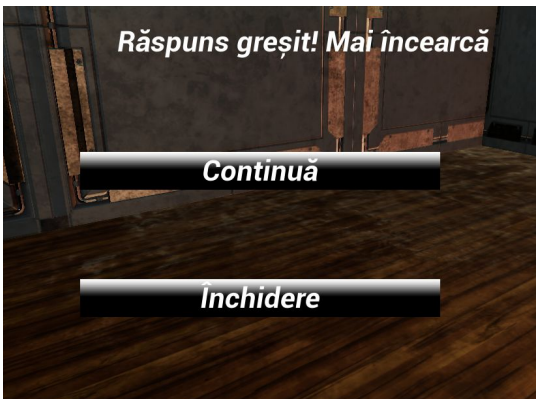
O altă modalitate de a evada este aceea de a merge în colţul din dreapta cu privirea înspre uşă.

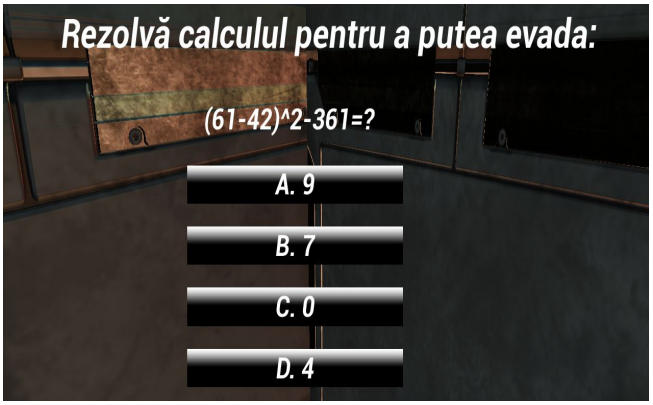
În momentul ajungerii în colţul respectiv, jucătorul va primi de rezolvat un calcul.

Dacă jucătorul dă răspunsul corect, se vor deschide uşile iar el va putea evada, în caz contrar, el va mai putea încerca să dea

răspunsul corect dacă ajunge din nou în colţul respectiv.

Răspunsul corect este C. 0, răspuns pentru care se vor deschide uşile, în timp ce pentru orice alt răspuns, va apărea pe ecran un mesaj şi posibilitatea de a juca în continuare sau de a închide jocul.





În imaginea din dreapta este ceea ce se afişează în momentul în care jucătorul ajunge în colţul respectiv, adică calculul şi rezultatele, iar în imaginea din dreapta ce se afişează dacă jucătorul alege răspunsul greşit. Are posibilitatea de a continua sau de a renunţa la joc.

În joc există şi posibilitatea de a pune pauză. Dacă jucătorul se plictiseşte sau trebuie să oprească pentru câteva momente jocul, acesta poate face acest lucru doar apăsând pe tasta “P” de la tastatură.

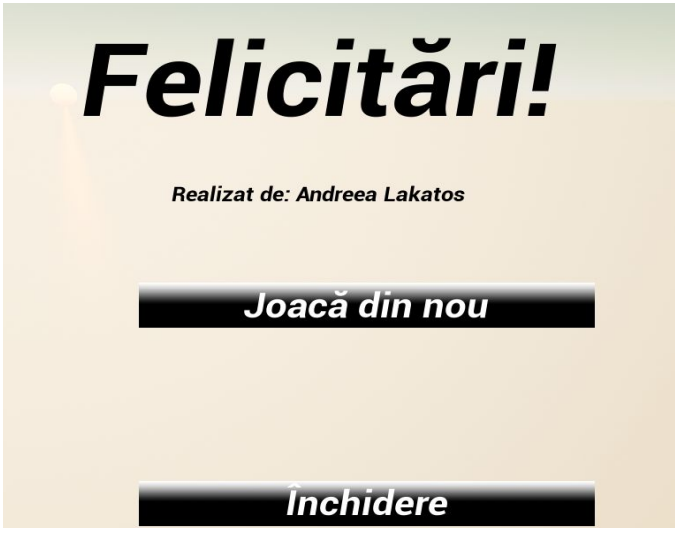
După apăsarea acestei taste, jucătorul are posibilitatea de a a continua jocul apăsând butonul “Continuă”, de a se întoarce la meniu apăsând butonul “Meniu” sau de a încheia jocul apăsând butonul “Închidere”.

Acest buton poate fi apăsat în orice moment al jocului şi de oricâte ori doreşte jucătorul.

.

Jocul se încheie în momentul în care jucătorul iese pe una dintre cele două uşi, indiferent de metoda pe care o alege să evadeze. După “evadare”, jucătorului i

se va afișa pe ecran mesajul “Felicitări” urmat de două butoane: unul pentru a începe un joc nou numit “Joc Nou”, iar altul pentru a închide jocul numit “Închidere”.

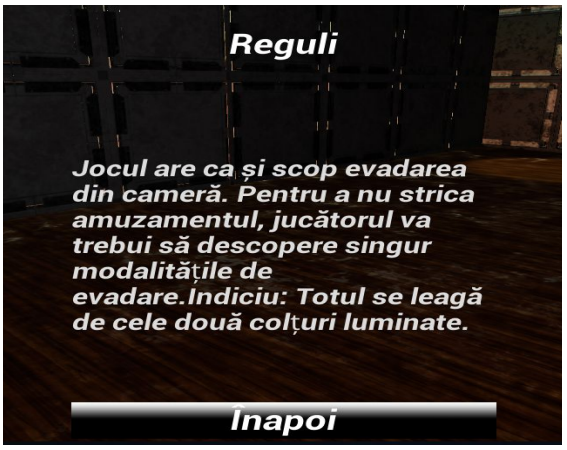


3.2 Meniuri

Aplicaţia conţine diferite meniuri realizate cu programarea vizuala de tip blueprints specifica pentru Unreal Engine 4.

Primul meniu creat este cel cu care şi debutează aplicaţia, cel în care jucătorul poate începe, afla regulile sau închide jocul.

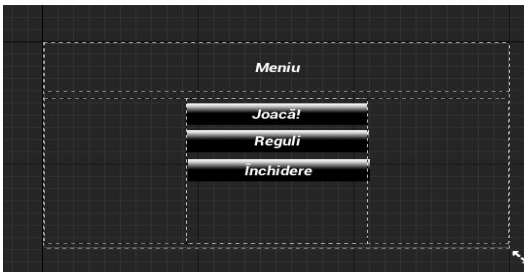
În imaginea din stânga se poate observa începutul jocului cu cele trei butoane. Fiecare buton este programat astfel încât să facă ce trebuie, adică exact ce spune şi denumirea acestuia.



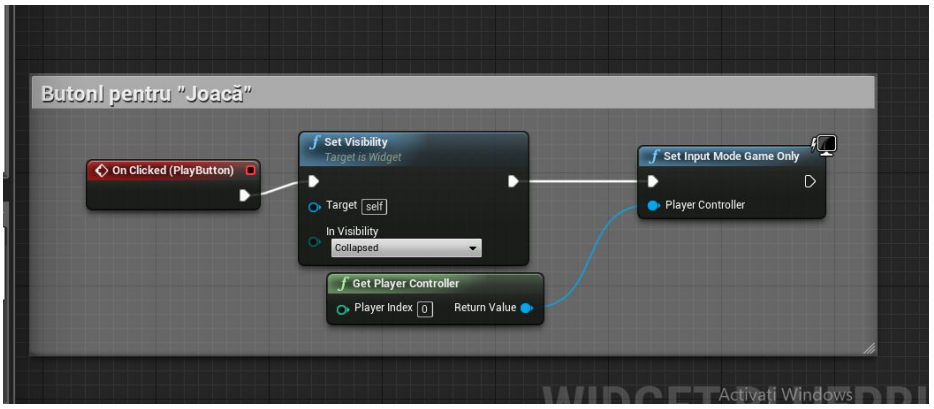
Dacă jucătorul apasă pe butonul de reguli el va afla câteva lucruri despre jocul pe care urmează să-l joace, cum se vede şi în imaginea alăturată.

De asemenea, el se poate întoarce la joc apăsând butonul “Înapoi”.

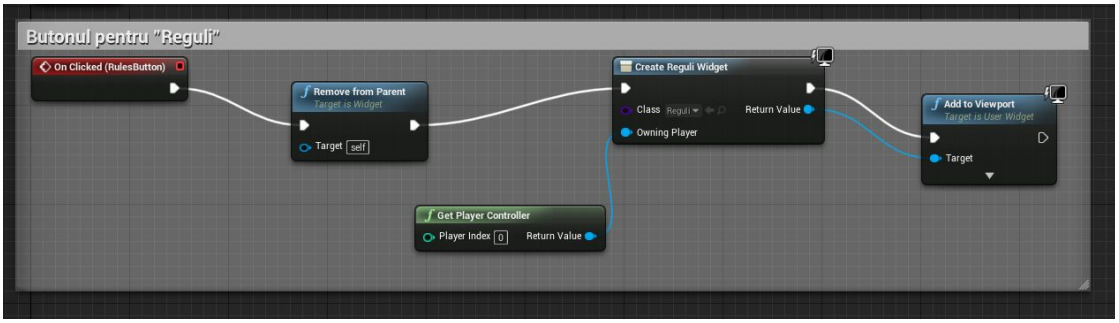
Mai jos voi ataşa programarea butoanelor pentru primul meniu:

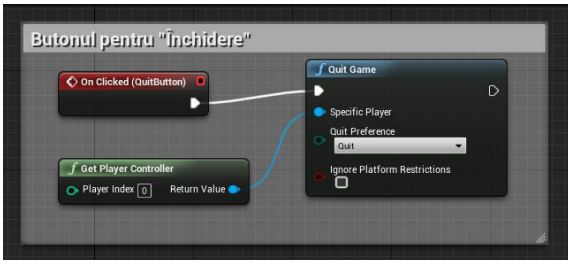


În imaginea din stânga se poate vedea cum a fost proiectat meniul, fiecare buton în parte urmând sa fie programat.



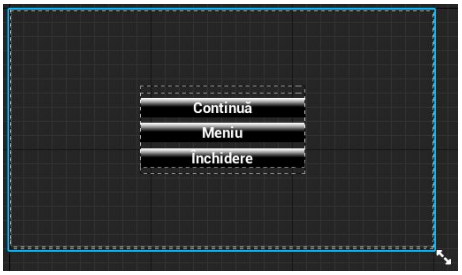
În imaginea alăturată se poate observa tot ce ține de programarea primului buton, cel pentru a începe jocul, denumit “Joacă!”. După începerea jocului meniul afişat iniţial va dispărea deoarece vizibilitatea a fost setată ca şi “Collapsed”.

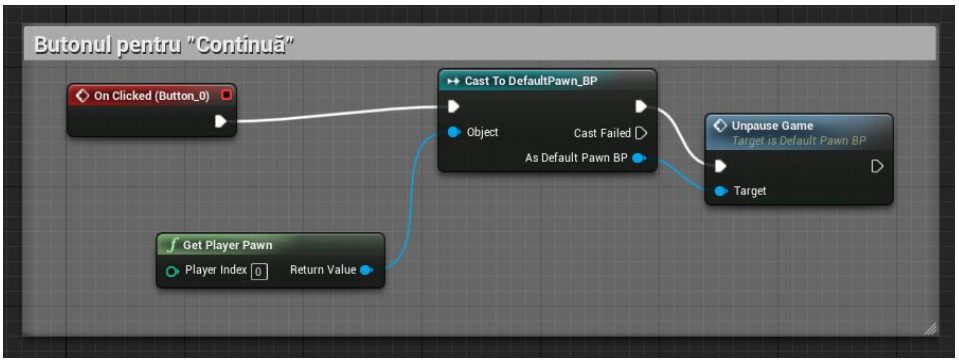




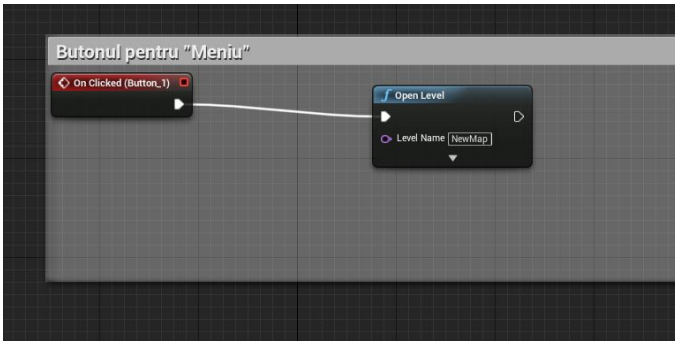
În imaginea de mai sus se poate observa programarea meniului “Reguli”, iar în cea din stânga programarea meniului “Închidere”.

Meniul pentru pauză cuprinde trei butoane care oferă posibilitate de a continua “Continuă”, de a se întoarce la meniul iniţial pentru a începe un joc nou sau pentru a consulta regulile “Meniu”, dar şi unul pentru a închide jocul “Închidere”.



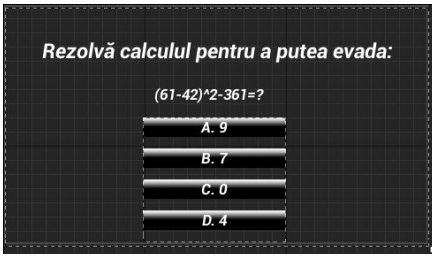


În imaginea alăturata avem programarea butonului “Continuă”.

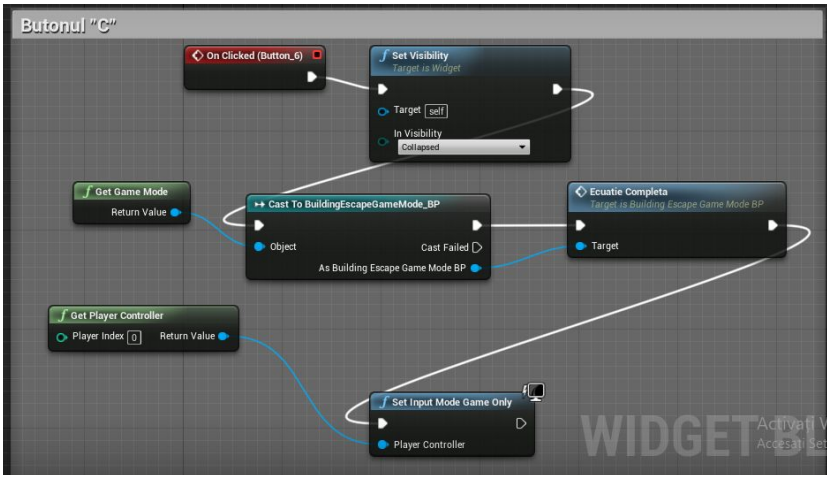


Această imagine este pentru butonul de întoarcere la Meniu, iar butonul pentru Închidere este acelaşi ca şi la meniul precedent.

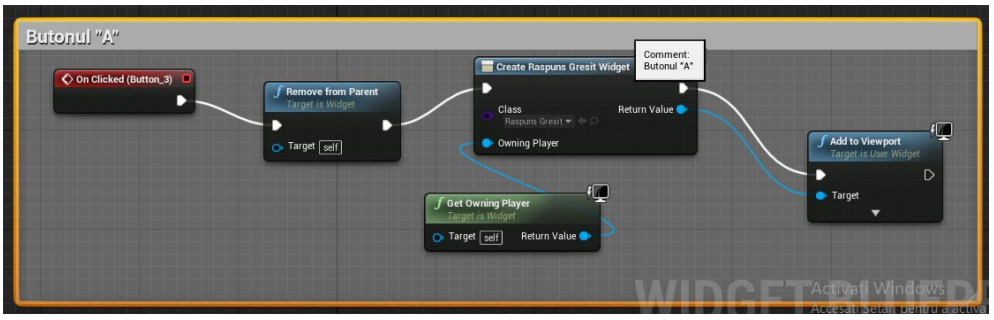
Deoarece una dintre metodele de a evadata este aceea de a da răspunsul corect la un calcul matematic, alt meniu este cel pentru calcul.



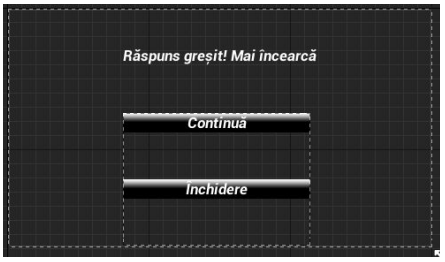
Acesta cuprinde calcului urmat de patru butoane cu răspunsuri. Fiecare buton în parte este programat să facă ceva în funcţie de răspunsul pe care îl dă jucătorul. După cum am văzut mai sus, răspunsul corect este C. 0, iar pentru acest răspuns uşile sunt programate să se deschidă, iar pentru orice alt răspuns se va afişa un mesaj şi butonul cu posibilitatea de a continua.



Astfel a fost programat acest buton, în care, vom vedea mai târziu, au fost şi uşile programate să se deschidă.



Acesta este felul în care au fost toate celelalte butoane programate.
Am ataşat un alt meniu care crează posibilitatea întoarcerii la joc după un răspuns greşit sau posibilitatea închiderii acestuia.



Nu este nimic nou în acest meniu, butoanele Continuă şi Închidere fiind identice cu cele de mai sus.

Ultimul meniu este chiar cel cu care se încheie jocul.

Butonul Joacă din nou este identic cu butonul “Meniu” de mai sus, iar cel de “Închidere”, de asemenea.
Acest meniu se afişează în momentul în care jucătorul este detectat de un trigger volume ataşat chiar în spatele uşilor.

3.3 Deschiderea uşilor

După cum am văzut, cele două uşi existente în cameră se pot deschide în două moduri bine prezentate mai sus.

Pentru primul mod, cel în care trebuie mutate obiecte în colţul din stânga a fost programat atât în C++, cât şi în blueprints. Deschiderea efectivă a uşii a fost programată în C++, iar sunetul care se aude când se închide uşa a fost ataşat în blueprints.

Am făcut o clasă pentru deschiderea uşii numită “OpenDoor” căreia i-am ataşat un header.

Mai voi ataşa codul sursă al clasei şi al header-ului, lucruri care pot fi accesate şi de pe DVD-ul ataşat.

Open.cpp:

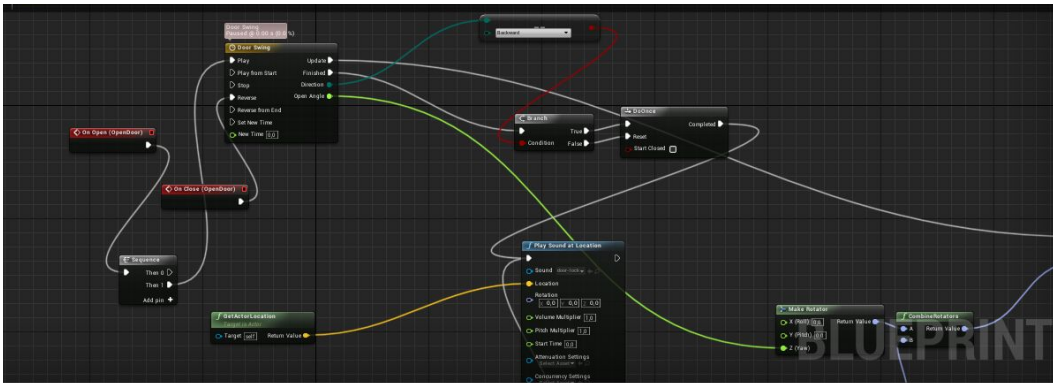
```
#include "OpenDoor.h"

#define OUT

// Sets default values for this component's properties
UOpenDoor::UOpenDoor()
{
   // Set this component to be initialized when the game starts, and to be ticked every frame.  You can turn these features
   // off to improve performance if you don't need them.
   PrimaryComponentTick.bCanEverTick = true;

   // ...
}


// Called when the game starts
void UOpenDoor::BeginPlay()
{
   Super::BeginPlay();

   Owner = GetOwner();
   if (!PressurePlate)
   {
         UE_LOG(LogTemp, Error, TEXT("%s missing pressure plate"), *GetOwner()->GetName())
```

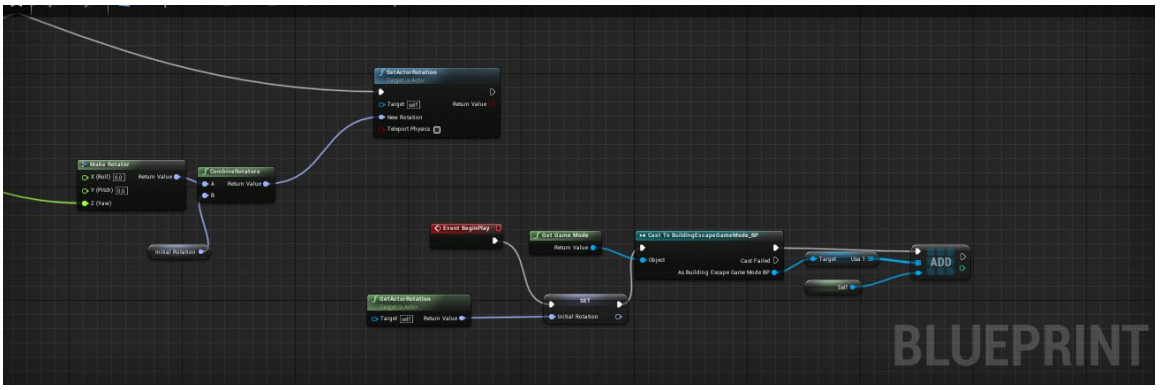
```
    }
}

// Called every frame
void UOpenDoor::TickComponent(float DeltaTime, ELevelTick TickType,
FActorComponentTickFunction* ThisTickFunction)
{
    Super::TickComponent(DeltaTime, TickType, ThisTickFunction);

    // Poll the Trigger Volume
    if (GetTotalMassOfActorsOnPlate() > TriggerMass)
        {
        OnOpen.Broadcast();
        }
    else
        {
        OnClose.Broadcast();
        }
}

float UOpenDoor::GetTotalMassOfActorsOnPlate()
{
    float TotalMass = 0.f;

    // Find all the overlapping actors
    TArray<AActor*> OverlappingActors;
    if (!PressurePlate) { return TotalMass; }
    PressurePlate->GetOverlappingActors(OUT OverlappingActors);

    // Iterate through then adding their nasses
    for (const auto& Actor : OverlappingActors)
    {

TotalMass+=Actor->FindComponentByClass<UPrimitiveComponent>()->GetMass();
            UE_LOG(LogTemp, Warning, TEXT("%s on pressure plate"),
*Actor->GetName())
    }

    return TotalMass;
}
OpenDoor.h:
#pragma once
```

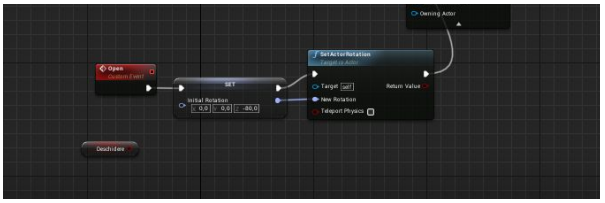
```
#include "CoreMinimal.h"
#include "Components/ActorComponent.h"
#include "Engine/TriggerVolume.h"
#include "OpenDoor.generated.h"

DECLARE_DYNAMIC_MULTICAST_DELEGATE(FDoorEvent);

UCLASS( ClassGroup=(Custom), meta=(BlueprintSpawnableComponent) )
class BUILDINGESCAPE_API UOpenDoor : public UActorComponent
{
    GENERATED_BODY()

public:
    // Sets default values for this component's properties
    UOpenDoor();

protected:
    // Called when the game starts
    virtual void BeginPlay() override;

public:
    // Called every frame
        virtual void TickComponent(float DeltaTime, ELevelTick TickType,
FActorComponentTickFunction* ThisTickFunction) override;

    UPROPERTY(BlueprintAssignable)
    FDoorEvent OnOpen;

    UPROPERTY(BlueprintAssignable)
    FDoorEvent OnClose;

private:
    UPROPERTY(EditAnywhere)
    ATriggerVolume* PressurePlate = nullptr;

    UPROPERTY(EditAnywhere)
    float TriggerMass = 30.f;

    // The Owning door
    AActor* Owner = nullptr;
```

```
    // Returns total mass in kg
    float GetTotalMassOfActorsOnPlate();
};
```

Pentru sunetul de pe fundal la închiderea uşii, cât şi cea de-a doua metodă de evadare setările şi programarea au fost realizate în blueprints.







3.4 Mobilitatea obiectelor

În aplicaţie, după cum am aflat şi în cele spuse mai sus, obiecte se pot muta, acest fapt constituind chiar o metodă de evadare. Ele se pot ridica prin double-right-click şi se pot muta ţinând right-click-ul apăsat prin intermediul săgeţilor de mişcare.

Obiectele pot fi mutate şi suprapuse.

Aceste lucruri au fost realizate cu ajutorul a doua noi componente: un grabber şi un physics handle. Toate obiectele sunt de tip movable, lucru care le permite să fie mutate în timpul jocului. Codul necesar a fost scris în C++, într-o clasă numită “Grabber.cpp” căreia i s-a ataşat un header numit “Grabbar.h”.

Acest cod poate fi accesat pe DVD-ul ataşat.

4. Concluzie

În crearea aplicaţiei au fost folosite două limbaje de programare, C++ şi blueprints, combinând astfel programarea vizuală cu programarea text.

Este un joc uşor de înţeles, dar, în acelaşi timp şi un exerciţiu de gândire deoarece jucătorii trebuie să găsească singuri modalităţile de evadare.

Instalarea software-urilor necesare se poate realiza individual sau prin rularea executabilului UE4PrereqSetup_x86 găsit în interiorul directorului WindowsNoEditor\Engine\Extras\Redist\en-us.

Bibliografie

1. https://docs.unrealengine.com/en-us/