PlayCanvas

Wprowadzenie

Typy kont

Rozpoczęcie przygody z PlayCanvas wymaga założenia konta użytkownika. W zależności od potrzeb może to być konto w pełni darmowe, jednak posiadające pewne ograniczenia, lub płatne. Konta różnią się pod względem dostępnego miejsca na projekty, ilości projektów, ilości współpracowników itd.

FREE	PRO 5	PRO 10	PRO 20
Storage: 200 MB	**Storage: 500 MB**	**Storage: 2000 MB**	**Storage: 10000 MB**
Private Projects: 0	Private Projects: 5	Private Projects: 10	Private Projects: 20
Team: 2	Team: 5	Team: 10	Team: 20
Self-host Projects	Self-host Projects	Self-host Projects	Self-host Projects
	iOS Export	iOS Export	iOS Export
Price: $0 / mo	**Price: $15 / mo**	**Price: $30 / mo**	**Price: $60 / mo**

PlayCanvas vs. Unity

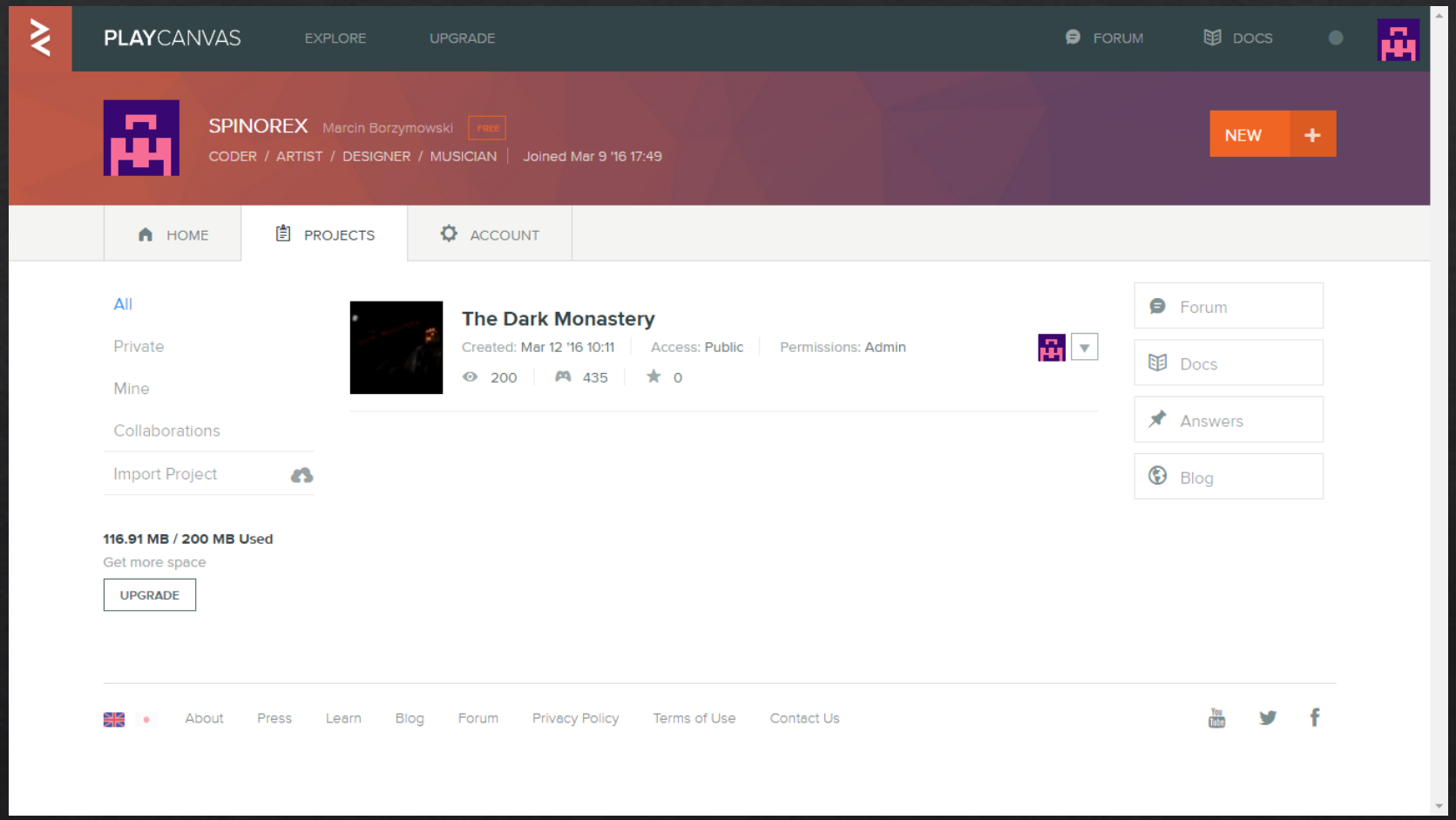
Wiele osób zaczynających przygodę z tworzeniem gier pyta, jaką platformę wybrać – PlayCanvas, czy Unity? Oto kilka zalet i wad obu z nich.

PlayCanvas	Unity
+ Niezależny od platformy systemowej + Skryptowanie w języku JavaScript + Prosty i intuicyjny edytor + Stale, dynamicznie rozwijany (nowa kompilacja czasami nawet kilka razy dziennie!)	+ Bardziej dojrzała platforma + Większe możliwości + Obsługa PhysX i Box2D
- Problemy z kompatybilnością w niektórych przeglądarkach (IE, Safari) - Brak obsługi silników fizycznych takich jak PhysX - Brak możliwości pracy w edytorze offline	- Skomplikowana dla początkujących - Kompilacja pod konkretną platformę - Konieczność poznania nowego języka UnityScript

Panel użytkownika

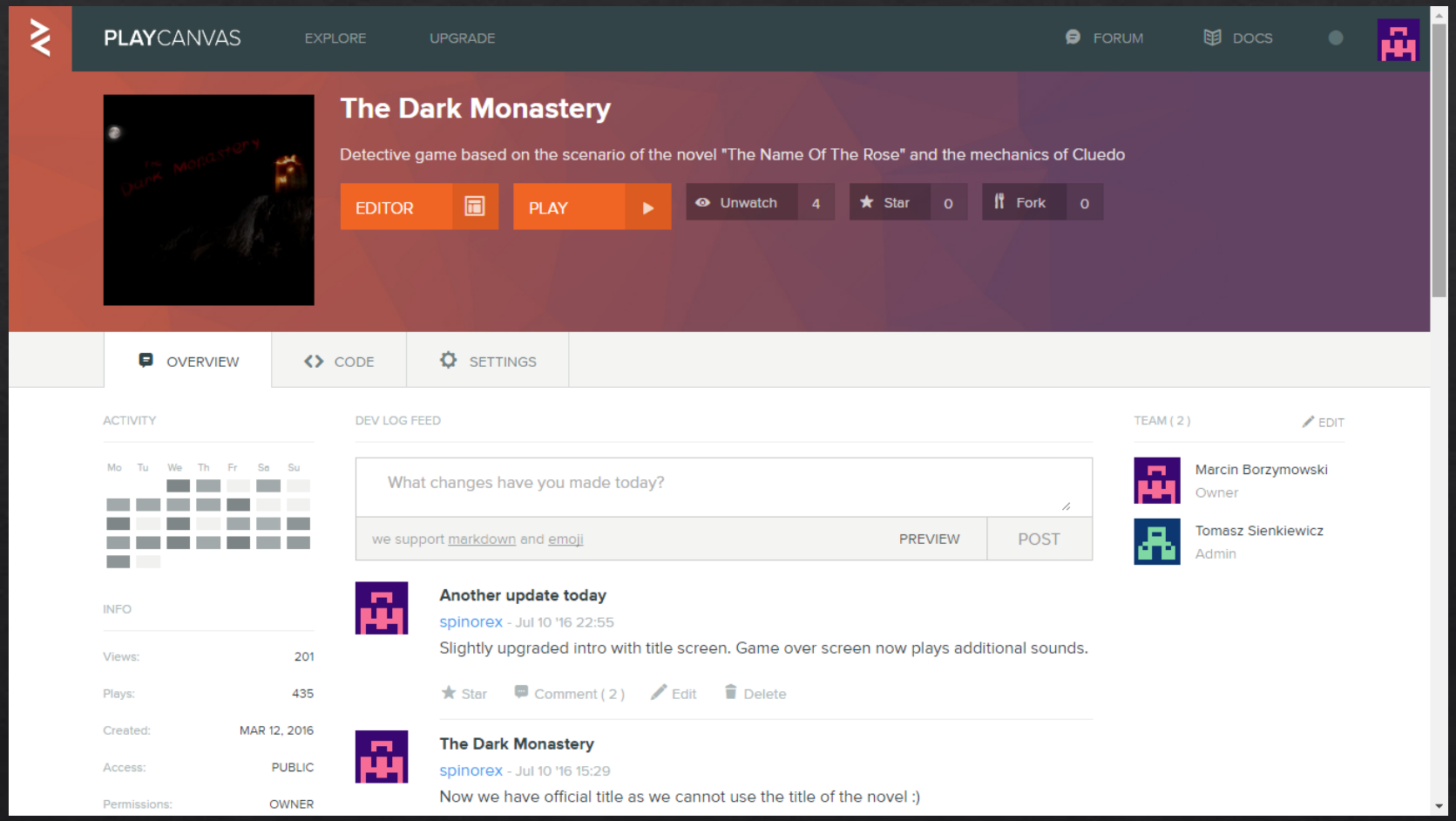
Konto użytkownika

Po zakończeniu rejestracji użytkownik otrzymuje dostęp do swojego konta. W tym miejscu możliwe jest zmienianie informacji o profilu takich jak nazwa, imię i nazwisko, adres e-mail, avatar i hasło.
Zakładka HOME zawiera ostatnie posty autorów z całego świata – odpowiednik Tablicy Facebooka.
W zakładce PROJECTS można przeglądać swoje projekty. Nowy projekt tworzy się przyciskiem + NEW



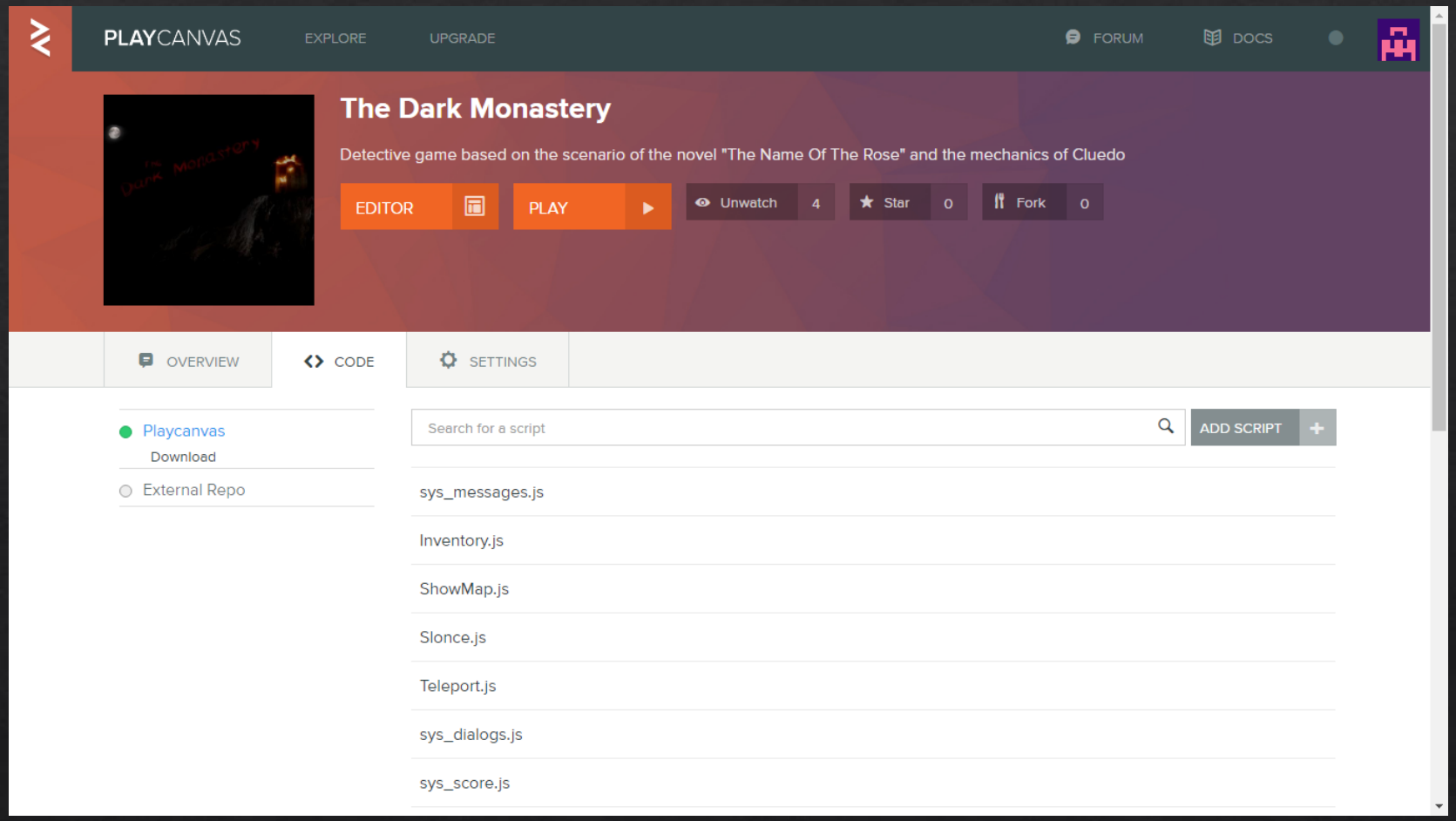
Strona projektu - OVERVIEW

Strona projektu również jest podzielona na 3 zakładki. W części wspólnej można m. in. przejść do edytora, uruchomić publiczną wersję projektu lub dodać projekt do obserwowanych.
Zakładka OVERVIEW zawiera kalendarz aktywności (ostatnie 30 dni), miejsce do publikowania aktualizacji oraz skład teamu.



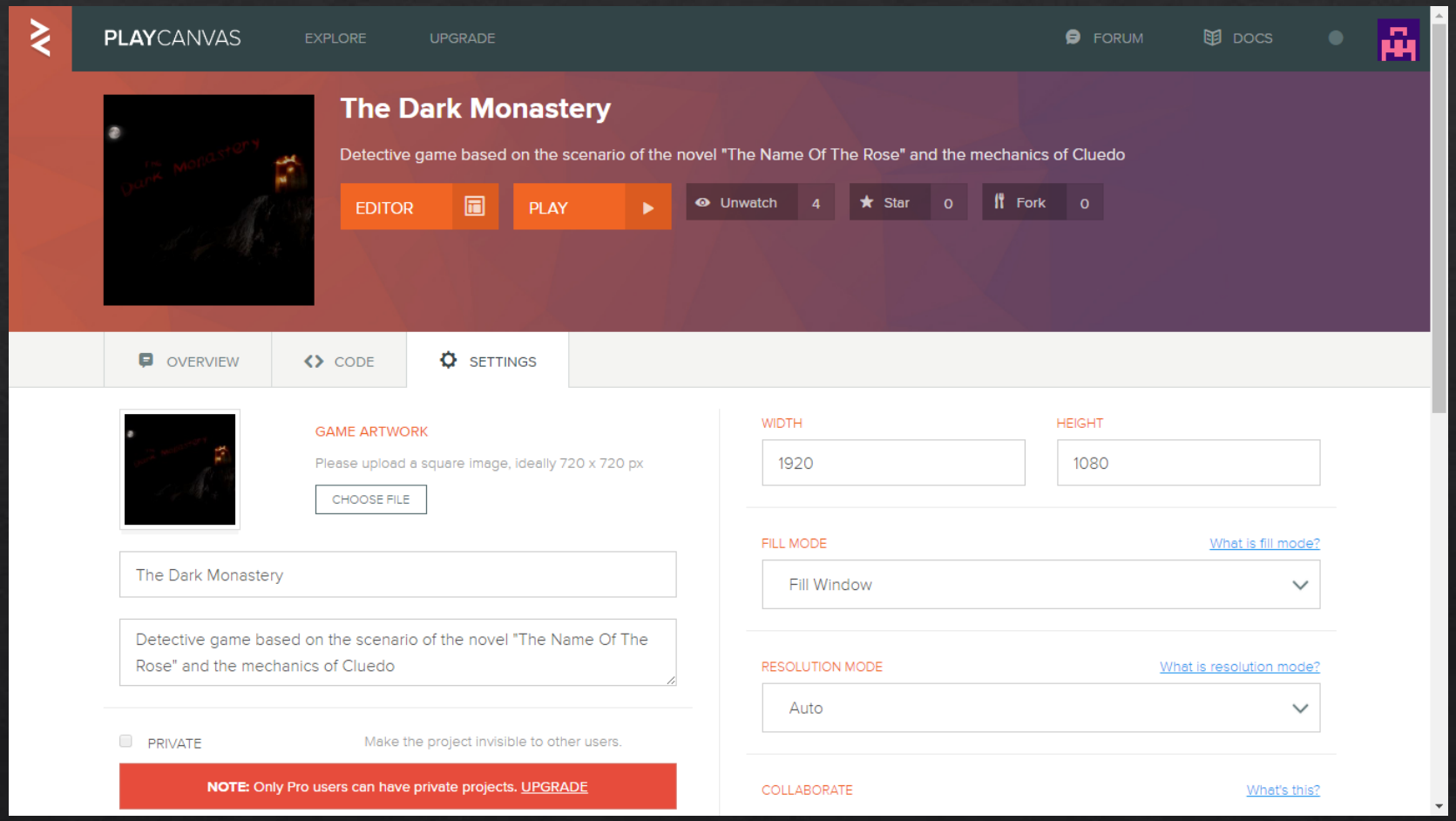
Strona projektu - CODE

W zakładce CODE znajdują się odnośniki do wszystkich skryptów używanych w projekcie. Można z tego miejsca je otwierać i edytować bez konieczności uruchamiania samego edytora. Kod można przechowywać także na zewnątrz – PlayCanvas obsługuje integrację z najpopularniejszymi repozytoriami, jak np. GitHub czy BitBucket.



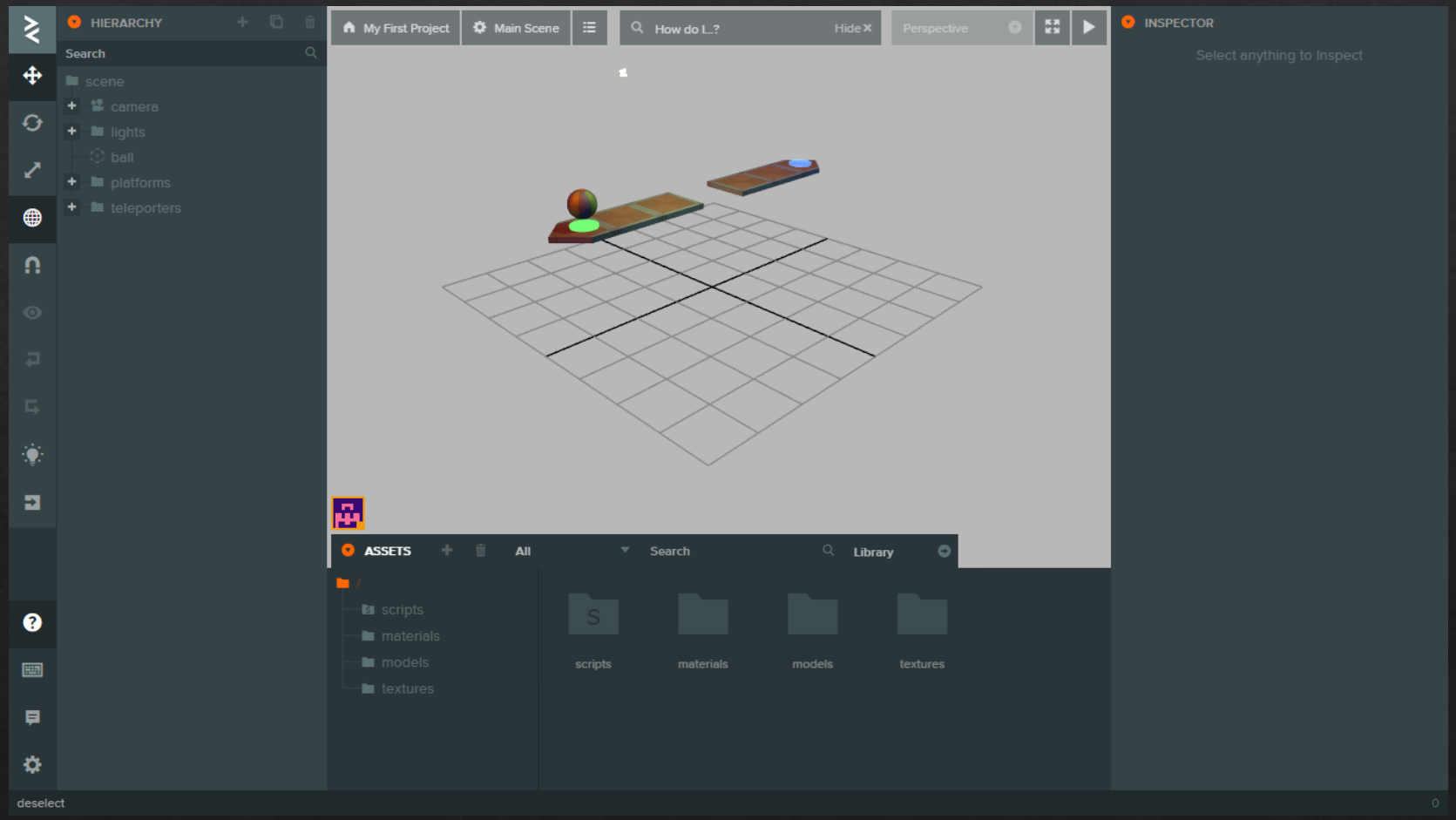
Strona projektu - SETTINGS

Zakładka SETTINGS pozwala na zmianę ustawień projektu, takich jak np. wyjściowa rozdzielczość, obsługę fizyki, tytuł, opis i miniaturkę projektu widoczną w katalogu PlayCanvas.
Można tutaj także zapraszać kolejne osoby do współpracy (na koncie darmowym tylko jedną) i zarządzać uprawnieniami współpracowników.



Pierwszy projekt - samouczek

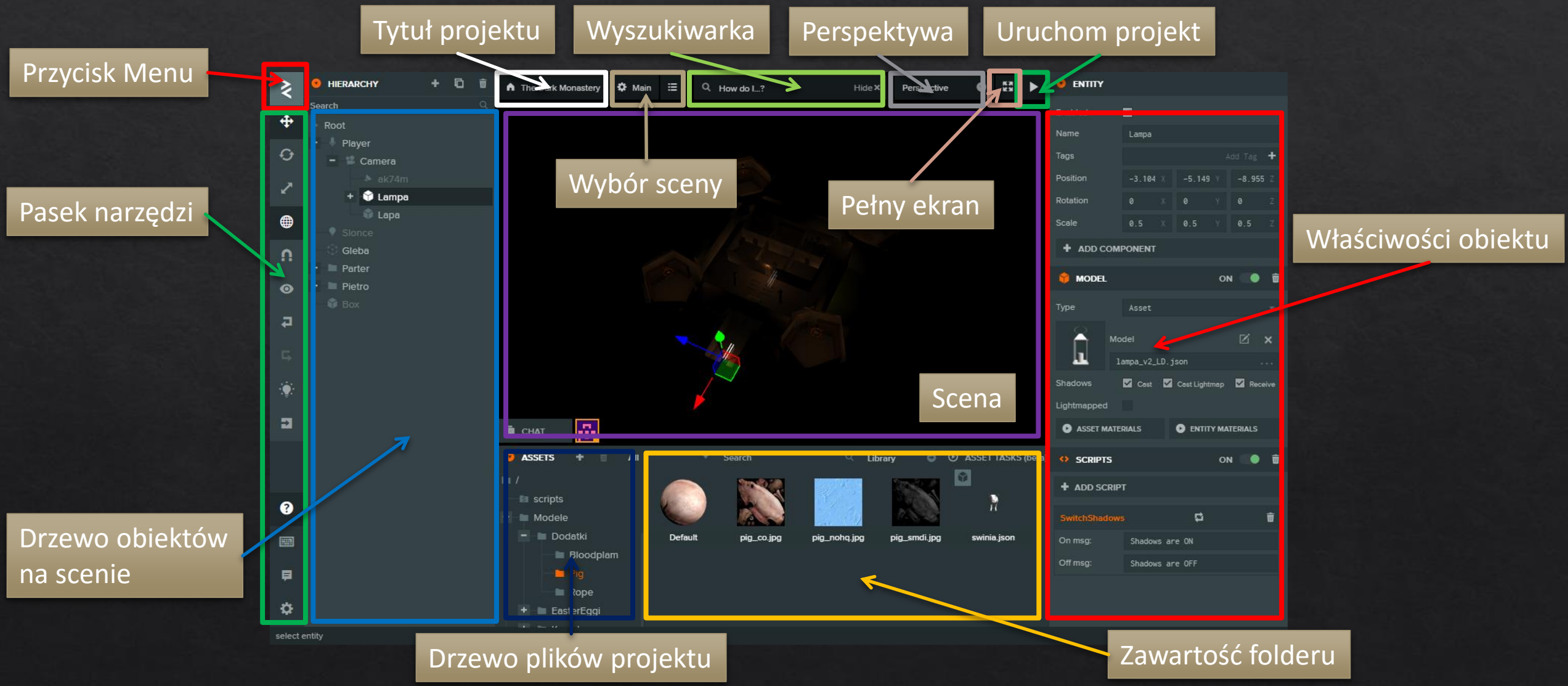
Po założeniu konta od razu utworzony na nim jest przykładowy projekt – gra Rolling Ball - zawierający samouczek. Warto się z nim zapoznać, gdyż prezentuje zasady poruszania się po edytorze.



Interfejs edytora

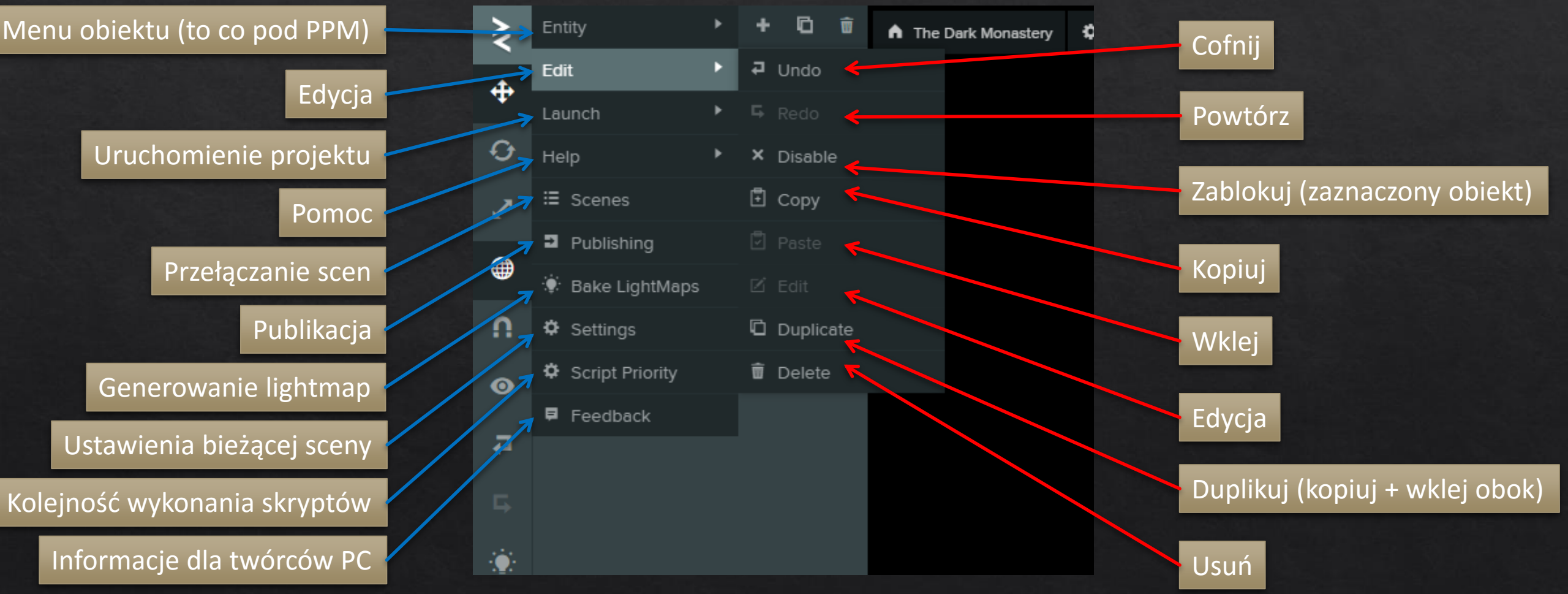
Interfejs edytora

Aby łatwo poruszać się po edytorze należy poznać komponenty jego interfejsu. Bardziej szczegółowo zostaną omówione na kolejnych slajdach, ich układ prezentuje się następująco:



Menu główne

Główne menu edytora zawiera zestaw podstawowych poleceń umożliwiających sprawne zarządzanie projektem. Menu wywołuje się klikając na logo PlayCanvas w lewym górnym rogu ekranu.



Pasek narzędzi

Umiejscowiony przy lewej krawędzi, zawiera najczęściej używane polecenia w formie dużych przycisków, aby były zawsze łatwo dostępne.

- Tryb przesuwania obiektu
- Tryb obracania obiektu
- Tryb skalowania obiektu
- Punkt odniesienia (globalne/lokalne)
- Przyciąganie do siatki
- Dopasowanie widoku do zaznaczenia
- Cofnij ostatnią akcję
- Powtórz („cofnij cofnięcie”)
- Auto. Generowanie lightmap
- Publikacja
- Wyszukiwanie w pomocy
- Sterowanie w edytorze
- Informacje zwrotne dla twórców PC
- Ustawienia bieżącej sceny

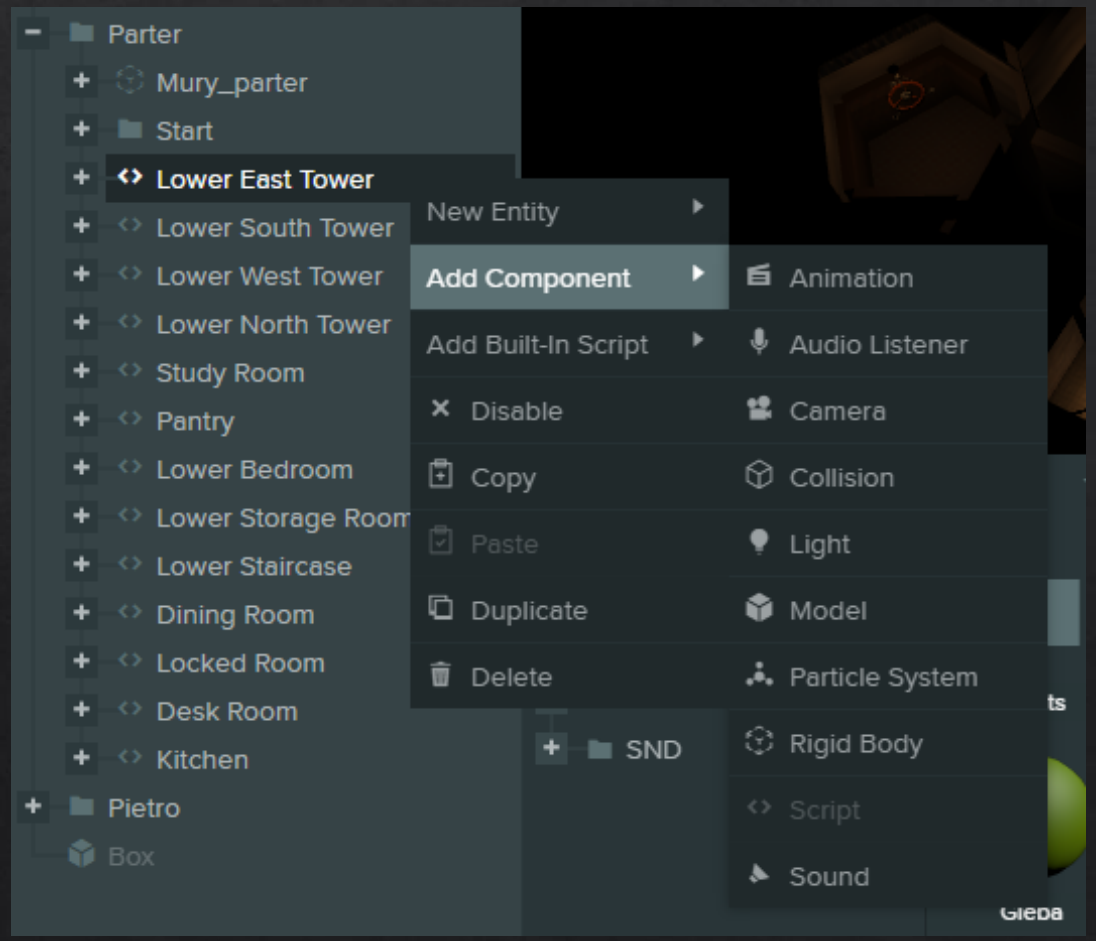
Hierarchia (drzewo obiektów w scenie)

Ważnym elementem jest hierarchia. Obiekty znajdujące się w scenie prezentowane są w formie drzewka. Służy to organizacji obiektów i ułatwieniu manipulacji na nich. Np. jeśli mamy obiekt będący pokojem, a w nim obiekty krzesła, to możemy przestawiać każde krzesło z osobna, ale przesuwając pokój przesuniemy go z całą zawartością w niezmienionym położeniu (względem pokoju).

Logiczne uporządkowanie obiektów znacznie ułatwia pracę – zwłaszcza, gdy jest ich bardzo dużo

Hierarchia – menu kontekstowe

Menu kontekstowe to menu wyświetlane po kliknięciu prawym przyciskiem myszy na obiekcie w drzewie. Zawiera listę akcji możliwych do wykonania na zaznaczonym obiekcie (lub kilku – zaznaczenie jak w Windows, z shiftem lub controlem).

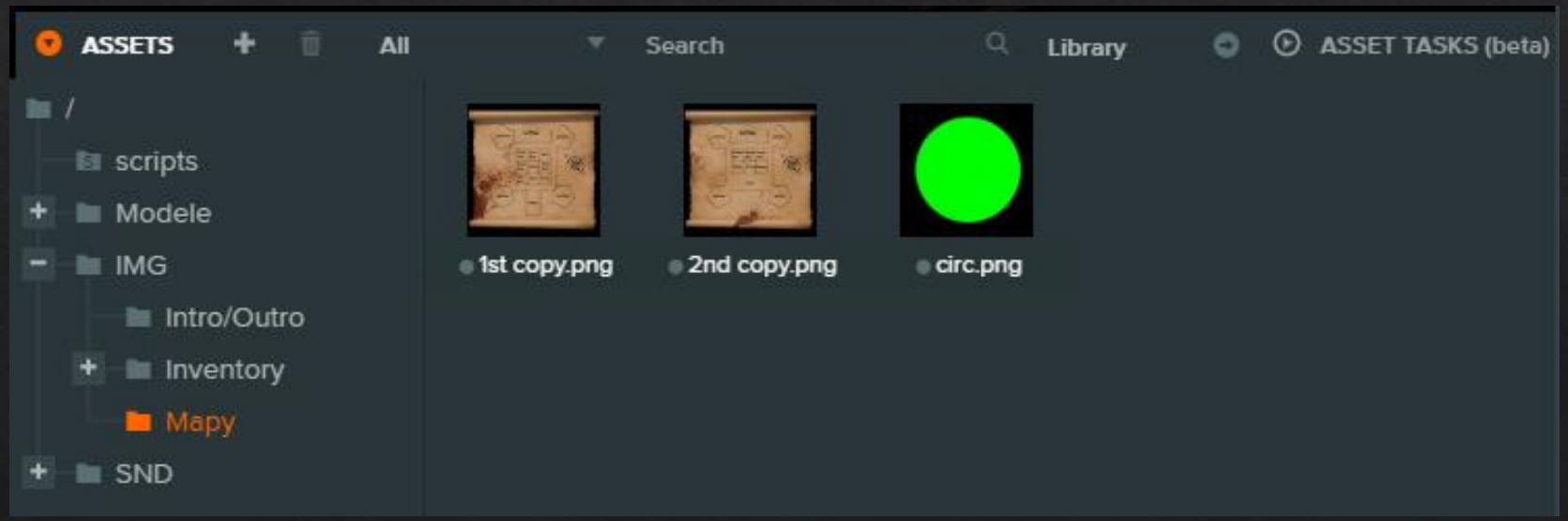


Entity, to inaczej właśnie obiekt na scenie – czyli np. model 3D, źródło światła, ale też folder, dźwięk, kamera itd.

Istniejące już obiekty mogą zawierać także komponenty, którymi również mogą być modele, dźwięki, światła…
Zalecamy podłączanie wymaganych elementów jako komponentów do obiektów w miarę możliwości – znacznie łatwiej później odwoływać się do nich w kodzie.

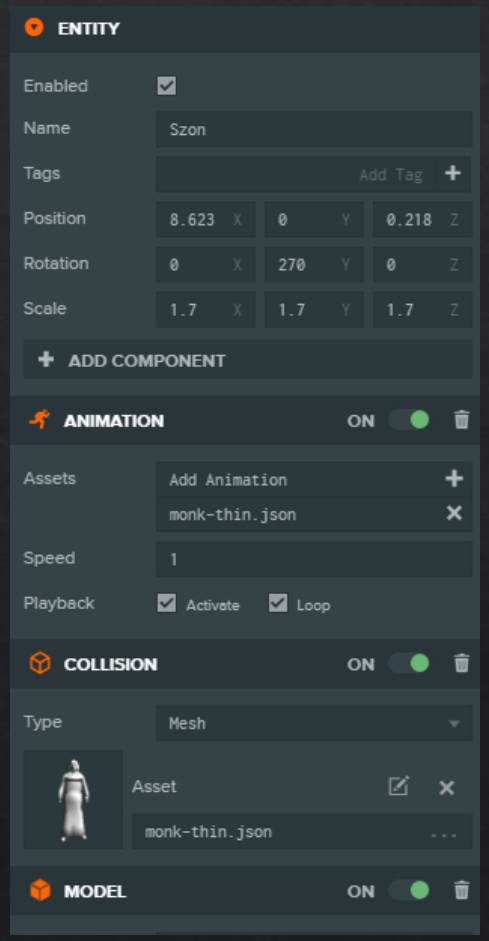
Drzewo plików

To miejsce reprezentuje folder przydzielony na projekt na serwerach PlayCanvas. Znajdują się tutaj wszystkie pliki zaimportowane z zewnątrz do PlayCanvas – obrazy, tekstury, dźwięki, modele itd. Całość prezentowana jest w doskonale znanej formie z komputerów osobistych. Po lewej stronie znajduje się drzewo folderów, które można dowolnie zagnieżdżać. Prawa, większa część to podgląd plików znajdujących się w wybranym aktualnie folderze.
Wyjątek w układzie folderów stanowi systemowy folder scripts – tutaj zapisane są wszystkie skrypty. Nie można zapisać skryptu w innym folderze ani budować drzewa folderów wewnątrz folderu scripts.



Właściwości obiektu

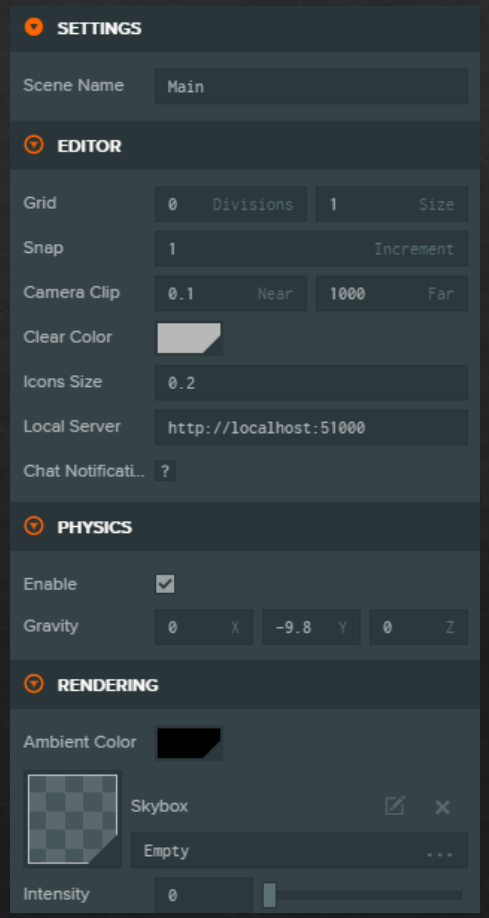
Po zaznaczeniu wybranego obiektu, czy to z hierarchii czy z drzewa plików można uzyskać dostęp do jego właściwości. Wszystkie opcje pogrupowane są według kategorii i rozmieszczone z prawej strony ekranu.



Z tego miejsca można ustawić położenie obiektu (względem środka wirtualnego świata lub względem środka obiektu nadrzędnego, zależy od stanu przycisku Global/Local), oraz dodawać komponenty takie jak animacje, modele, dźwięki, skrypty i ustawiać ich parametry.

Ustawienia sceny

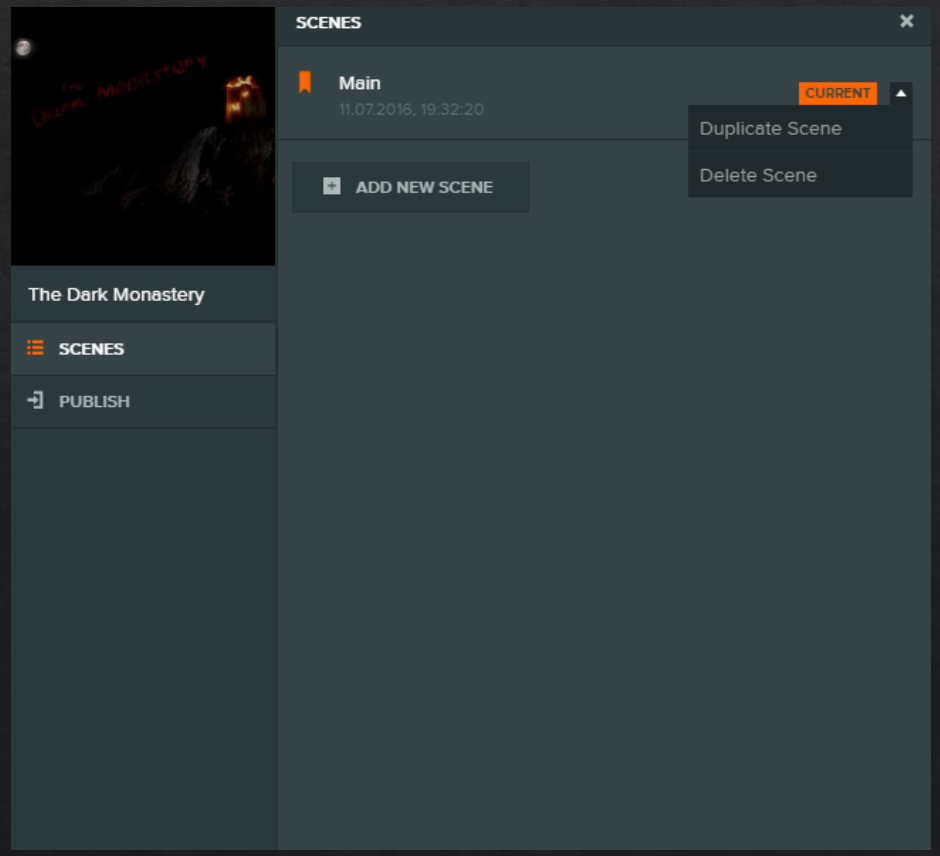
W menu ustawień można zmienić parametry bieżącej sceny, zaczynając od nazwy, kończąc na sposobie renderowania.
Menu ustawień wyświetlane jest z prawej strony ekranu, w miejscu właściwości obiektu.



Zmiana parametrów jest intuicyjna, a po najechaniu kursorem myszy na nazwę parametru wyświetlany jest opis.
Prostota obsługi jest jedną z wielu zalet PlayCanvas. Jak widać, aby włączyć grawitację w projekcie, wystarczy zaznaczyć odpowiedni checkbox.

Zarządzanie scenami

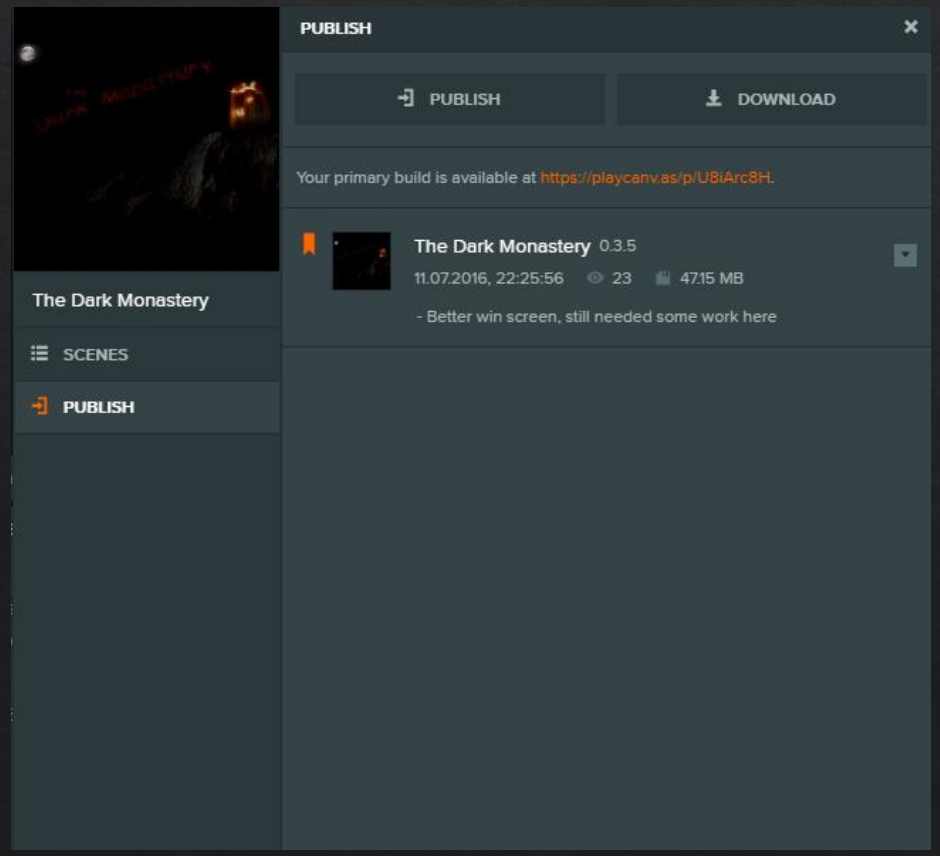
W projekcie można korzystać z wielu scen. Każda scena posiada własną hierarchię obiektów, podczas gdy drzewo plików jest wspólne dla wszystkich scen (można korzystać z tych samych plików i skryptów).
Logicznym posunięciem przy tworzeniu gry wielopoziomowej jest umieszczenie każdego poziomu w osobnej scenie.



W oknie zarządzania scenami można wykonywać jedynie podstawowe akcje na scenach:
- dodać kolejne sceny
- skopiować scenę (przydatne, jeśli część elementów w nowej scenie ma być taka jak w kopiowanej)
- usunąć sceny. Uwaga! Usunięcia sceny nie można cofnąć.

Publikacja

Można przechowywać kilka wersji gry (swego rodzaju snapshoty), na tyle na ile pozwala na to dostępne miejsce. Jedną wersję można oznaczyć jako główną. Będzie ona dostępna publicznie pod stałym adresem, inne wersje mają adresy generowane na bieżąco.
Jest to przydatne, jeśli chcemy udostępnić publicznie stabilną wersję i pracować nad kolejnym wydaniem w czasie, gdy gracze mogą korzystać ze stabilnej wersji.



Z tego miejsca można wykonać dwie operacje:
- opublikować nową wersję i oznaczyć ją jako główną
- pobrać wybraną wersję

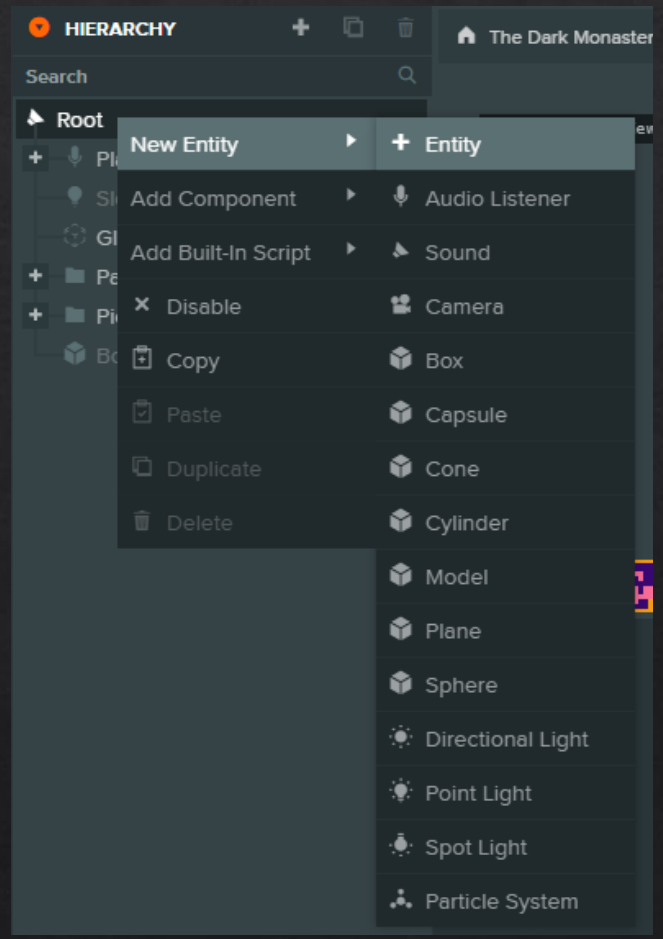
Niestety nie można pobranej wersji wgrać później ponownie do edytora (jest to możliwe na płatnych kontach). Pobrany projekt można umieścić na własnym serwerze i z niego uruchamiać grę.

Wersja oznaczona jako główna posiada przy miniaturce pomarańczowy znacznik. Aby oznaczyć jakąś wersję jako główną, wystarczy kliknąć przy niej taki znacznik (dla pozostałych wersji ma on kolor szary).

Dodawanie obiektów do sceny

Nowy obiekt w hierarchii

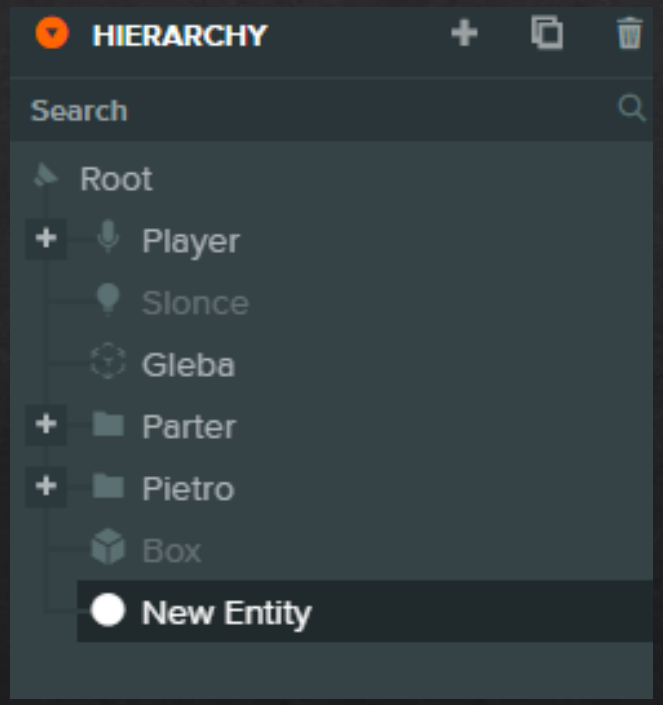
Jeśli obiekt, który chcemy dodać jest np. modelem zaimportowanym z komputera, możemy dodać go do sceny po prostu przeciągając z drzewa plików. Jednak nie polecamy tego z jednego powodu – nie mamy wpływu na to, gdzie w hierarchii umieści go edytor. Lepiej kliknąć PPM w miejscu w hierarchii, którym chcemy go dodać i wybieramy New entity → Entity.

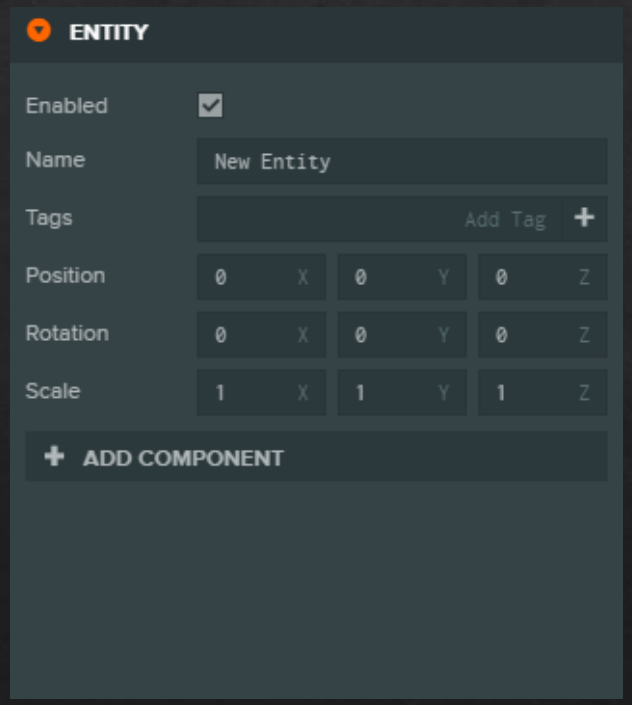


Możemy od razu wybrać takie typy obiektów jak np. kamera, czy model. Jeśli chcemy zacząć od prymitywów, również możemy wybrać je z listy. Puste Entity daje jednak możliwość dodania później wszystkich potrzebnych elementów jako komponenty.

Nowy obiekt w hierarchii

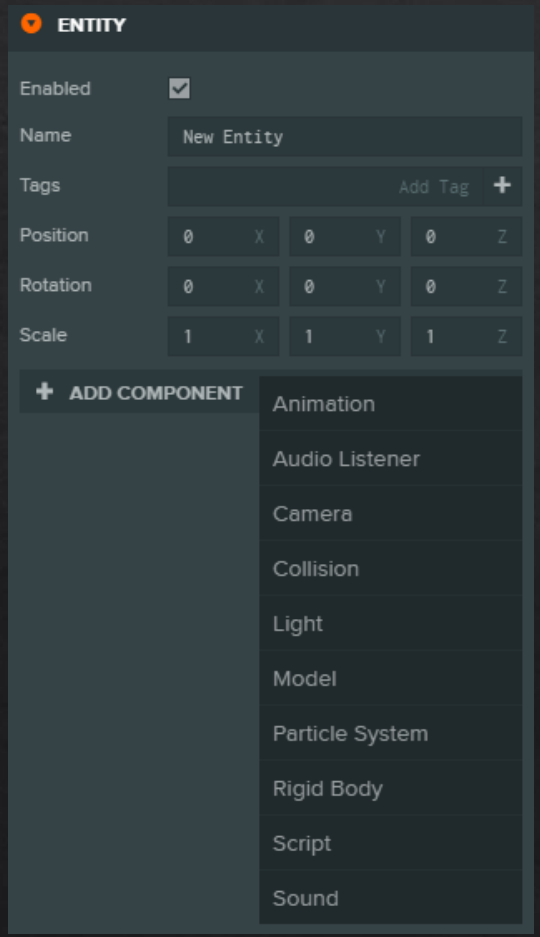
Po dodaniu obiektu i wybraniu go, z prawej strony załadują się jego właściwości. Można od razu zmienić jego położenie, wielkość i obrót w każdej z trzech osi wpisując wartości w pola przy danym parametrze (możliwe jest także wykonanie tego za pomocą myszy, włączając wcześniej odpowiedni tryb w pasku narzędzi).





Nowy obiekt w hierarchii

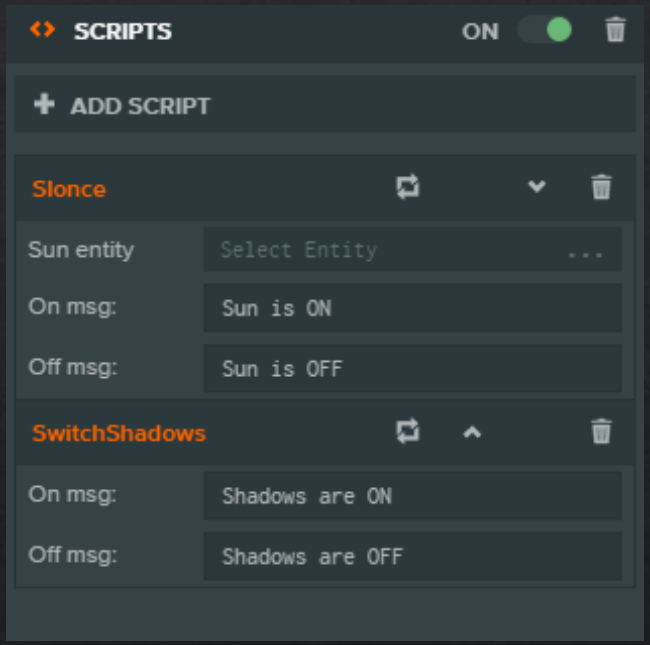
Do obiektu możemy dodawać komponenty przy użyciu przycisku + ADD COMPONENT. Należy jednak pamiętać, że do danego obiektu można dodać tylko po jednym komponencie danego typu (ale niektóre komponenty posiadają tzw. sloty, w których można umieszczać wiele elementów składających się na komponent – np. dźwięki i skrypty).

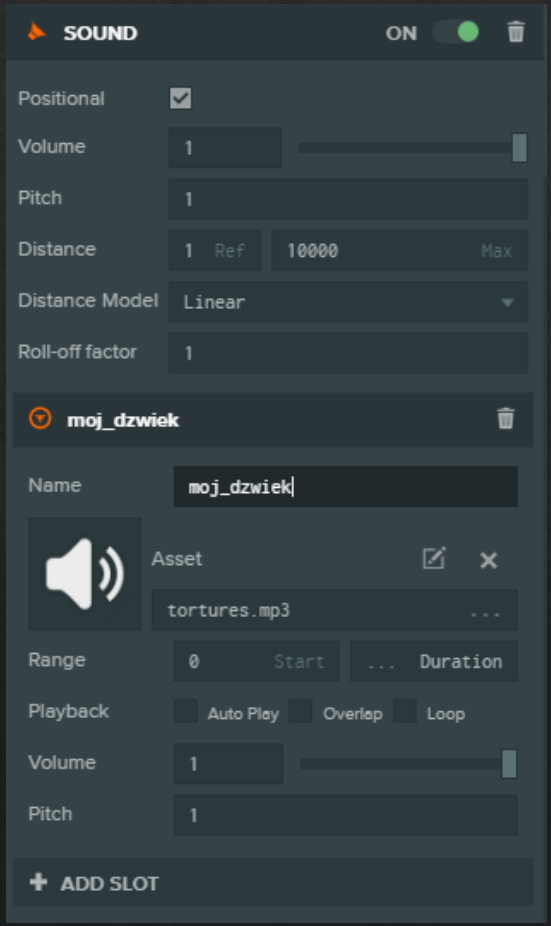


Animation – plik z animacjami do modelu. Bez dodanego modelu komponent nie jest wykorzystywany.
Audio listener – służy do odbierania dźwięków. Np. jeśli tworzymy grę typu FPS, gdzie kamera porusza się wraz z graczem, to należy dodać ten komponent do niej, aby gracz słyszał dźwięki z gry. Ten komponent nie posiada ustawień.
Camera – na ekranie gry można wyświetlać widok z kamery. W scenie musi być co najmniej jedna kamera.
Collision – komponent pozwalający na obsługę kolizji (eliminuje „przenikanie się” obiektów)
Light – źródło światła
Model – model 3D
Particle system – komponent do generowania efektów cząsteczek, np. dym, deszcz, ogień
Rigid Body – dodaje obsługę fizyki do obiektu (grawitacja, tarcie)
Script – włącza obsługę skryptów. Można dodawać wiele skryptów w ramach komponentu.
Sound – włącza emisję dźwięku. Można dodawać wiele dźwięków w ramach komponentu.

Nowy obiekt w hierarchii

Sound i Script to jedyne komponenty posiadające możliwość dodawania slotów. Polega to na tym, że komponent tylko włącza obsługę dźwięku czy skryptów i stanowi kontener obejmujący sloty. Dzięki temu można szybko włączyć/wyłączyć całą obsługę przełącznikiem w nagłówku komponentu.

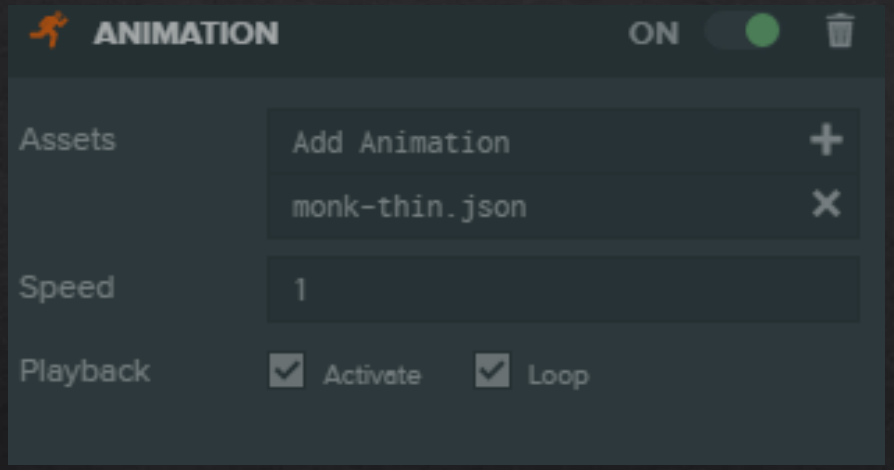




Komponenty

Animacja

Animacja zawiera zestaw ruchów dotyczących danego modelu. Najczęściej zawarta jest w pliku eksportowanym z programu do modelowania 3D, jednak przy importowaniu do PlayCanvas zostaje wydzielona do osobnego pliku (zarówno model jak i animacja są konwertowane do plików .json, jednak PlayCanvas potrafi je rozróżnić).



Assets – lista plików zawierających animacje. Możliwe jest dodawanie wielu plików.
Speed – prędkość animacji (względem wyjściowej, zapisanej w pliku)
Playback Activate – automatyczne odtwarzanie (przy odznaczonym checkboksie trzeba uruchamiać animację za pomocą skryptu)
Playback Loop – określa, czy animacja ma się odtwarzać w pętli, czy zatrzymać po jednokrotnym odtworzeniu

Kamera

Jest komponentem za pomocą którego możliwe jest oglądanie świata w grze. W scenie musi być przynajmniej jedna kamera – może być podczepiona do ruchomego obiektu reprezentującego gracza w grach typu FPS lub zawieszona w przestrzeni. Nie ma limitu kamer w scenie, pomiędzy nimi można przełączać się korzystając ze skryptów.

Clear buffers – czyszczenie buforów w każdej klatce
Clear Color – kolor używany przy czyszczeniu
Projection – typ projekcji. Perspektywa lub ortho.
Frustum Culling – niewidoczne powierzchnie nie będą renderowanie (polepsza wydajność)
Field of view – kąt widzenia kamery
Clip – odległość cięcia, czyli odległości, dla których nie będą renderowane powierzchnie
Priority – służy do ustalania kolejności renderowania kamer
Viewport – miejsce, w którym widoczny jest obraz z kamery. X i Y określają położenie lewego górnego rogu, H oraz W wysokość i szerokość ekranu.

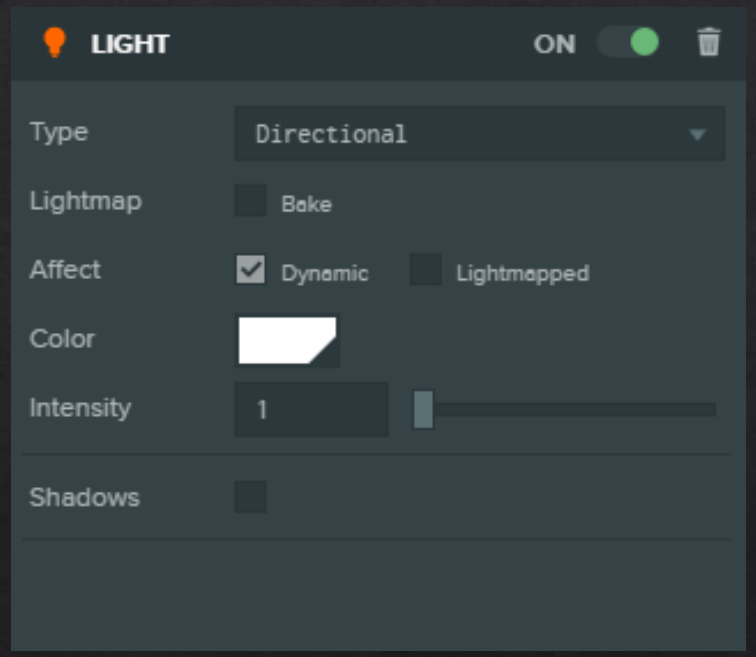
Kolizje

Kolizje pozwalają na kontrolę zachowania obiektu przy zderzeniu z innym. Akcje występujące w przypadku kolizji można oprogramować w skryptach. Sam komponent pozwala tylko na włączenie lub wyłączenie wykrywania kolizji, wybór typu modelu użytego do sprawdzania kolizji i jego rozmiaru.

Dla prymitywów możemy wybrać proste typy wykrywania kolizji, w formie prostopadłościanu, kuli, stożka lub walca.
W przypadku zaawansowanych modeli, jak np. postacie ludzi warto jednak ustawić typ na Mesh i wskazać model, który ma być użyty do obliczania kolizji (w większości przypadków będzie to ten sam, co model 3D obiektu).

Światło

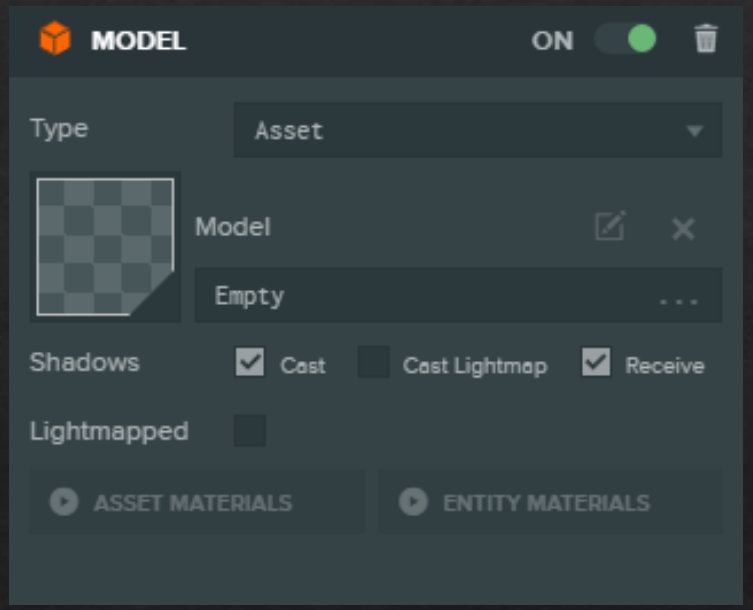
Jeśli chcemy, by obiekt emitował światło, należy skorzystać z komponentu Light.
W momencie pisania tego poradnika maksymalna liczba źródeł światła w scenie wynosi 16, ze względów na wydajność, jednak w miarę rozwoju platformy ograniczenie to z czasem powinno zniknąć.



Type – typ światła (kierunkowe, punktowe lub stożkowe – jak z latarki)
Lightmap Bake – zaznaczony checkbox spowoduje wykorzystania tego światła przy generowaniu lightmap
Affect Dynamic – światło dynamiczne, przeliczane w każdej klatce
Affect Lightmapped – określa wpływ światła na lightmapy
Color – kolor światła
Intensity – intensywność emisji
Shadows – obiekty, na które pada światło z tego źródła mogą rzucać cienie

Model

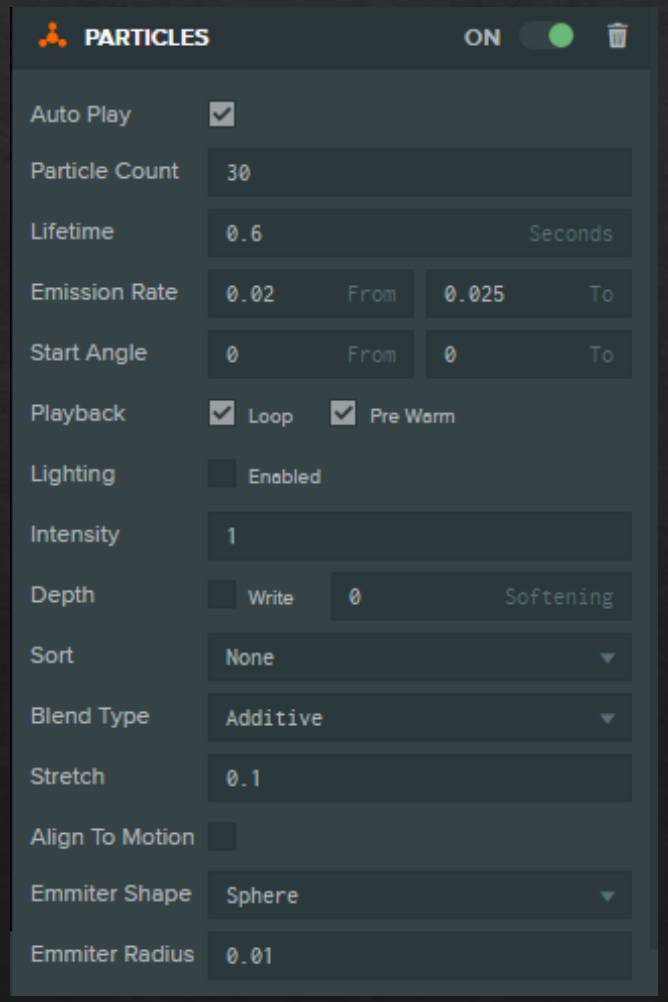
Określa to, jak obiekt będzie wyglądać. Może to być jeden z prymitywów lub model załadowany z pliku. PlayCanvas obsługuje wszystkie popularne typu plików, m. in. .3ds, .obj, .fbx, .stl. Podczas importowania zostają przekonwertowane do pliku .json.



Type – typ modelu, prymityw lub plik (Asset)
Model – jeśli wybrany został typ Asset, to tutaj dodaje się wybrany plik
Shadows Cast – model po oświetleniu będzie rzucał cienie
Shadows Cast Lightmap – j.w. w trybie lightmapy
Shadows Receive – model będzie przyjmował cienie
Lightmapped – określa, czy model korzysta z lightmap
ASSET MATERIALS – zbiór wszystkich materiałów modelu
ENTITY MATERIALS – lista dostowsowanych materiałów dla danego obiektu

System cząstek

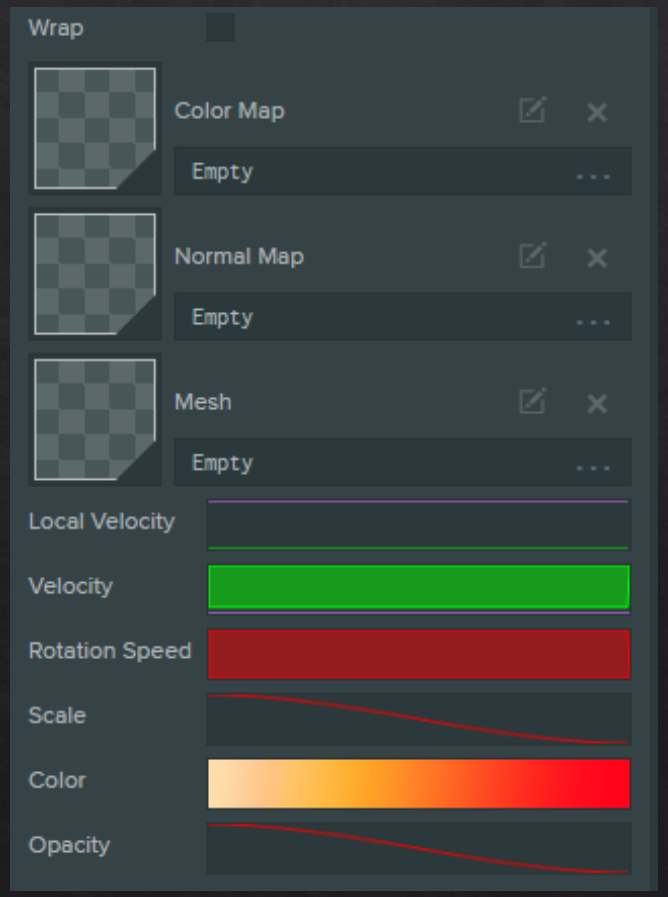
Komponent ten pozwala na generowanie dynamicznych efektów pozwalających uzyskać realistycznie wyglądające płomienie, deszcz oraz wiele innych elementów świata.
Jest to najbardziej rozbudowany pod względem konfiguracji komponent dostępny w PlayCanvas.



Auto Play – efekt będzie odtwarzany automatycznie
Particle Count – ilość cząsteczek w animacji
Lifetime – czas od pojawienia się każdej cząstki do jej zaniku
Emission rate – przedział dla losowo wybieranej częstotliwości kolejnych emisji
Start Angle – przedział dla losowo wybieranego kątu obrotu kolejnej cząsteczki
Playback Loop – jeśli zaznaczone, animacja będzie powtarzana w pętli
Playback Pre Warm – animacja jest inicjalizowana po jednym pełnym przebiegu. Działa tylko przy znaznaczonym Playback Loop.
Lighting – umożliwia odbijanie przez cząsteczki padającego na nie światła
Intensity – intensywność świecenia cząstek
Depth – efekt będzie posiadać głębię
Sort – służy do określenia kolejności renderowania cząstek
Blend Type – określa w jaki sposób cząstki będą dodawane do bufora (dodawanie, mnożenie, kanał alpha)
Stretch – rozciągnięcie cząsteczek
Align To Motion – ruch cząsteczek będzie dopasowany do ruchu obiektu zawierającego komponent
Emmiter Shape – kształty generowanych cząstek
Emmiter Radius – wielkość generowanych cząstek

System cząstek

Komponent ten pozwala na generowanie dynamicznych efektów pozwalających uzyskać realistycznie wyglądające płomienie, deszcz oraz wiele innych elementów świata.
Jest to najbardziej rozbudowany pod względem konfiguracji komponent dostępny w PlayCanvas.



Wrap – włącza granice zawijania
Color Map – plik z mapą kolorów
Normal Map – plik z mapą normalnych (do określenia faktury)
Mesh – plik z modelem 3D na podstawie którego mają być generowane cząstki
Local Velocity – krzywa określająca zmianę prędkości cząstek w czasie względem systemu cząstek
Velocity – krzywa okeślająca zmianę prędkości cząstek w czasie względem sceny
Rotation Speed – krzywa określająca obrót cząstek w czasie
Scale – krzywa określająca wielkość cząstek w czasie
Color – krzywa określająca zmianę koloru cząstek w czasie
Opacity – krzywa określająca zmianę widoczności cząstek w czasie

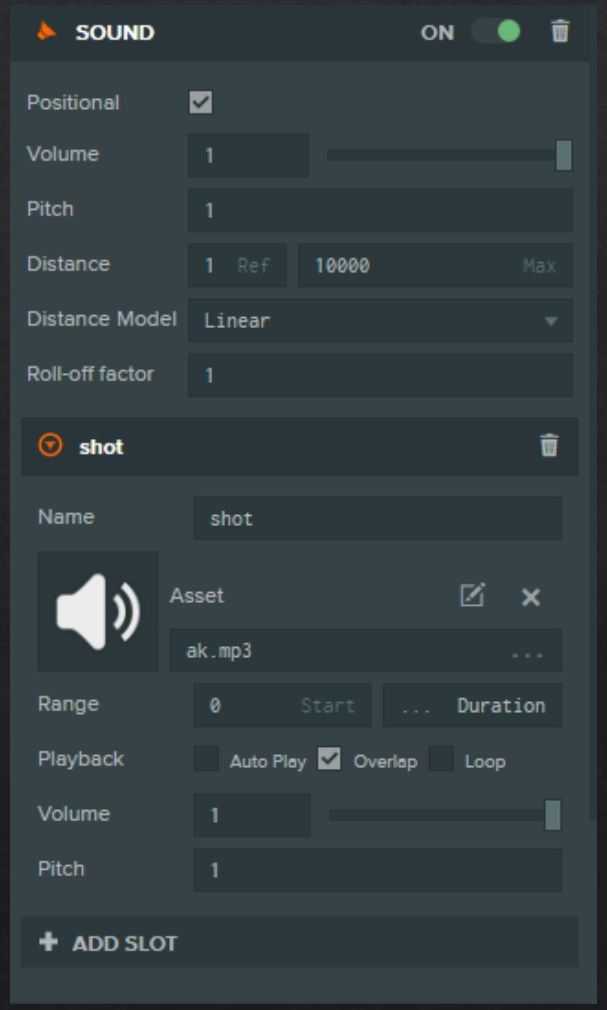
Rigid body

Za pomocą komponentu Rigid body można łatwo uzyskać naturalne zachowanie obiektu zgodnie z prawami fizyki. Aby komponent mógł przynieść efekty, należy włączyć obsługę fizyki w ustawieniach projektu.

Type – typ oddziaływania (statyczny dla statycznych obiektów, dynamiczny dla poruszających się)
Mass – wirtualny ciężar obiektu
Damping – współczynnik utraty prędkości liniowej i kątowej
Linear Factor – liniowy współczynnik ruchu obiektu w każdej z trzech osi
Angular Factor – kątowy współczynnik obrotu obiektu w każdej z trzech osi
Friction – współczynnik tarcia obiektu
Restitution – współczynnik odbicia obiektu

Dźwięk

Komponent za pomocą którego obiekt może emitować dźwięk. Na dodawanie wielu dźwięków pozwalają sloty. Do slotów można odwoływać się w skryptach za pomocą nazwy i w ten sposób sterować odtwarzaniem dźwięków przez obiekt.



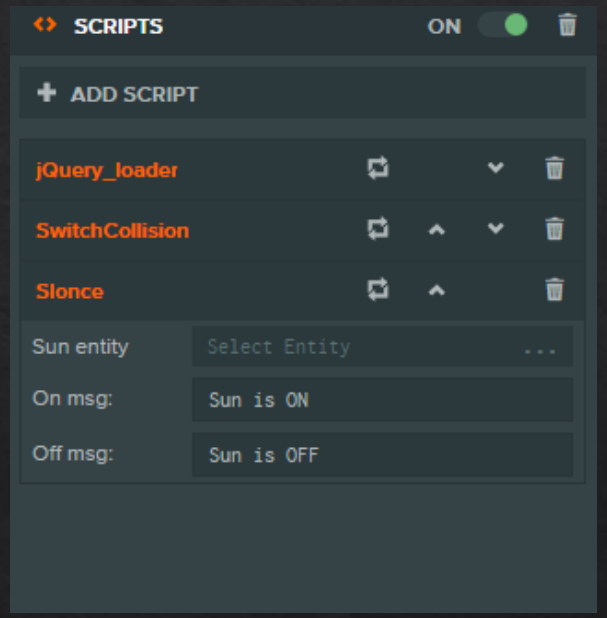
Komponent:
Positional – określa, czy dźwięk ma być emitowany z pozycji w której znajduje się obiekt
Volume – głośność odtwarzania dźwięków, globalnie dla komponentu
Distance – określa maksymalną odległosć słyszalności dźwięku oraz współczynnik zaniku wraz z odległością
Distance model – sposób obliczania odległości (liniowy lub wykładniczy)
Roll-off factor – Współczynnik odcięcia

Sloty:
Name – nazwa, po niej odwołujemy się w skryptach
Asset – plik z dźwiękiem
Range – zakres w pliku, który będzie odtwarzany
Playback AutoPlay – automatyczne odtwarzanie
Playback Overlap – nakładanie się dźwięków. Jeśli odznaczone, poprzedni dźwięk będzie zatrzymany
Playback Loop – odtwarzanie w pętli
Volume – głośność, lokalnie dla slotu
Pitch – ton dźwięku

ADD SLOT – dodanie kolejnego slotu

Skrypty

Komponent umożliwiający obsługę skryptów na bieżącym obiekcie. Bez skryptów nie da się stworzyć gry, to dzięki nim możliwe jest zaprogramowanie logiki rozgrywki. Podobnie jak dźwięk, także tutaj można dodawać kolejne skrypty w slotach.



Sam komponent nie posiada dodatkowych ustawień, umożliwia jedynie włączenie lub wyłączenie wykonywania skryptów na obiekcie.

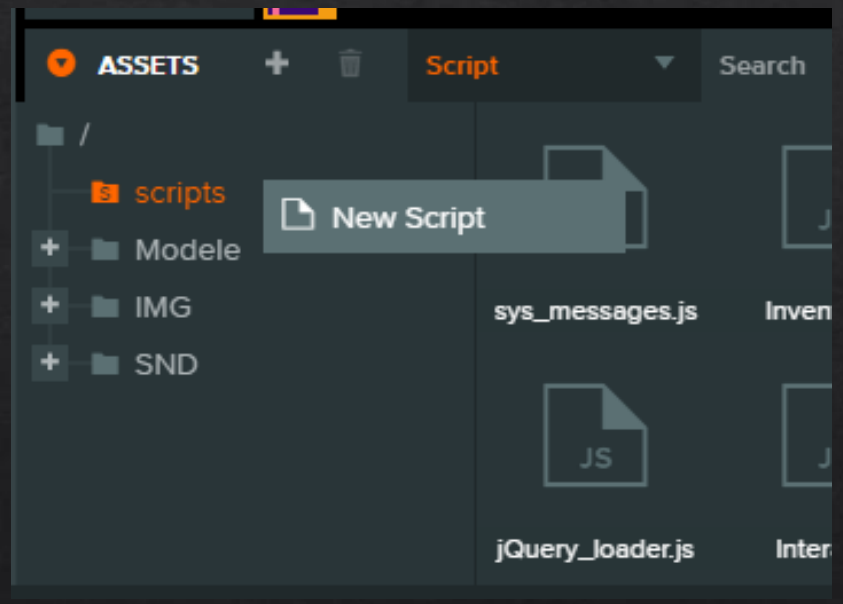
Na dodawanie kolejnych skryptów pozwala przycisk ADD SCRIPT.

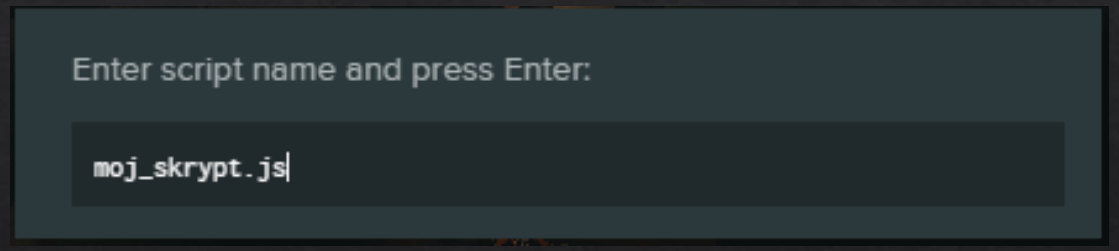
Skrypty mogą mieć własne parametry. Zostanie to omówione w rozdziale dotyczącym skryptów.

Skrypty

Nowy skrypt

Utworzenie nowego skryptu nie jest skomplikowane. Wystarczy kliknąć prawym przyciskiem myszy na folder Scripts i wybrać opcję New Script (lub w dowolnym miejscu drzewa plików i wówczas wybrać New Asset → Script) i podać jego nazwę.





Ogólna budowa skryptu

Po utworzeniu skryptu można go otworzyć klikając na niego dwukrotnie. Skrypt jest wstępnie przygotowany przez środowisko. Każdy skrypt zawiera od razu definicję nowej klasy o nazwie takiej jak nazwa pliku skryptu (jednak zaczynając od pierwszej dużej litery).

PLAYCANVAS CODE EDITOR

```
pc.script.create('moj_skrypt', function (app) {
    // Creates a new Moj_skrypt instance
    var Moj_skrypt = function (entity) {
        this.entity = entity;
    };

    Moj_skrypt.prototype = {
        // Called once after all resources are loaded and before the first update
        initialize: function () {
        },

        // Called every frame, dt is time in seconds since last update
        update: function (dt) {
        }
    };

    return Moj_skrypt;
});
```

Klasa od razu zawiera dwie wymagane funkcje – initialize() i update() (jeśli nie będą potrzebne, można nie definiować ich ciała, ale deklaracje muszą pozostać).

Funkcja initialize()

Jest to funkcja uruchamiana tylko raz, przy wczytywaniu. Tutaj umieszczamy wszelkie inicjalizacje, ustawienia początkowych wartości itd. Przykład zastosowania poniżej.

PLAYCANVAS CODE EDITOR

```
pc.script.create('moj_skrypt', function (app) {
    // Creates a new Moj_skrypt instance
    var Moj_skrypt = function (entity) {
        this.entity = entity;
    };

    Moj_skrypt.prototype = {
        // Called once after all resources are loaded and before the first update
        initialize: function () {

            this.variable1 = 7;
            this.variable2 = 'aaa';

            console.log('moj_skrypt loaded on entity:' + this.entity.name);
        },

        // Called every frame, dt is time in seconds since last update
        update: function (dt) {
        }
    };

    return Moj_skrypt;
});
```

Podczas wczytywania gry od razu zostaną do obiektu dodane 2 zmienne, *variable1* oraz *variable2* wraz z początkowymi wartościami, a następnie do konsoli wypisana zostanie informacja wraz z nazwą obiektu.

Słowem kluczowym *this* odwołujemy się do obiektu, do którego podpięty jest edytowany skrypt. Oczywiście dany skrypt może być podpięty do wielu obiektów.

Funkcja update()

Jest uruchamiana w każdej klatce. Tutaj umieszczamy wszelkie działania, które powinny wykonywać się cały czas, jak np. odczytywanie wciskanych klawiszy. Funkcję update() można porównać do pętli.

PLAYCANVAS CODE EDITOR

```
1  pc.script.create('moj_skrypt', function (app) {
2      // Creates a new Moj_skrypt instance
3      var Moj_skrypt = function (entity) {
4          this.entity = entity;
5      };
6
7      Moj_skrypt.prototype = {
8          // Called once after all resources are loaded and before the first update
9          initialize: function () {
10
11             this.variable1 = 7;
12             this.variable2 = 'aaa';
13
14             console.log('moj_skrypt loaded on entity:' + this.entity.name);
15         },
16
17         // Called every frame, dt is time in seconds since last update
18         update: function (dt) {
19
20             if(app.keyboard.wasReleased(pc.KEY_F))
21             {
22                 this.variable1++;
23
24                 console.log(this.variable2 + ' ' + this.variable1);
25             }
26         }
27     };
28
29     return Moj_skrypt;
30 });
```

Ten skrypt w każdej klatce sprawdza, czy puszczony został klawisz [F]. Jeśli tak, zwiększa zmienną *variable1* obiektu, a następnie wypisuje do konsoli string będący wartościami zmiennych *variable2* i *variable1*, rozdzielonych spacją.

Atrybuty

Atrybuty skryptów to wspaniała rzecz. Jeśli w trakcie tworzenia gry chcemy zmienić coś w kodzie, np. prędkość biegania postaci, w normalnej sytuacji wymaga to zmiany wartości w skrypcie, zapisania skryptu i uruchomienia projektu od nowa. Atrybuty pozwalają na zmianę wartości na bieżąco, nawet w trakcie uruchomienia projektu.

PLAYCANVAS CODE EDITOR

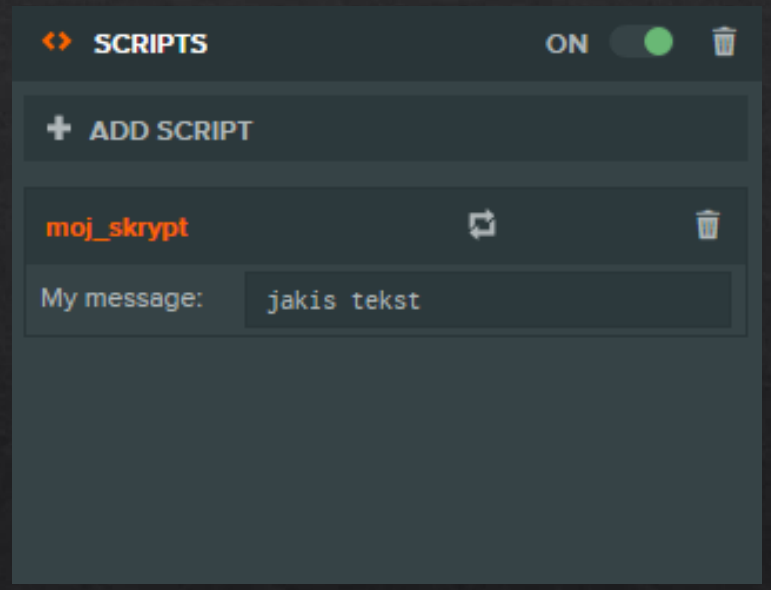
```
1  pc.script.attribute('custom_msg', 'string', 'jakis tekst', {displayName: 'My message: '});
2
3  pc.script.create('moj_skrypt', function (app) {
4      // Creates a new Moj_skrypt instance
5      var Moj_skrypt = function (entity) {
6          this.entity = entity;
7      };
8
9      Moj_skrypt.prototype = {
10         // Called once after all resources are loaded and before the first update
11         initialize: function () {
12
13             this.variable1 = 7;
14             this.variable2 = 'aaa';
15
16             console.log('moj_skrypt loaded on entity:' + this.entity.name);
17         },
18
19         // Called every frame, dt is time in seconds since last update
20         update: function (dt) {
21
22             if(app.keyboard.wasReleased(pc.KEY_F))
23             {
24                 this.variable1++;
25
26                 console.log(this.variable2 + ' ' + this.variable1);
27                 console.log(this.custom_msg);
28             }
29         }
30     };
31
32     return Moj_skrypt;
33 });
```

Nad definicją klasy dodajemy definicję atrybutu. Kolejne argumenty to:

- nazwa zmiennej, pod jaką atrybut będzie dostępny
- typ zmiennej (typy JavaScript)
- wartość domyślna, używana w przypadku braku określenia innej w edytorze)
- obiekt z dodatkowymi parametrami, jak na przykład nazwa wyświetlana

Atrybuty

Atrybuty skryptów to wspaniała rzecz. Jeśli w trakcje tworzenia gry chcemy zmienić coś w kodzie, np. prędkość biegania postaci, w normalnej sytuacji wymaga to zmiany wartości w skrypcie, zapisania skryptu i uruchomienia projektu od nowa. Atrybuty pozwalają na zmianę wartości na bieżąco, nawet w trakcie uruchomienia projektu.



W edytorze, po kliknięciu w ikonkę odświeżenia pod nazwą skryptu pojawi się dodany argument. Jego wartość można zmieniać w dowolnym momencie bez konieczności zmieniania kodu.

Koniec

Szczegóły dotyczące skryptowania w PlayCanvas :
http://developer.playcanvas.com/

Wszelkie nowości i zmiany dotyczące platformy można śledzić na bieżąco na oficjalnym blogu produktu:
http://blog.playcanvas.com/

Po odpowiedzi na nurtujące pytania oraz pomoc w przypadku problemów lub niejasności zaprasza oficjalne forum:
http://forum.playcanvas.com/