



Helion 

# PRAKTYCZNE TWORZENIE GIER **UNITY<sup>®</sup> i BLENDER<sup>™</sup>**

Alan Thorn

Tytuł oryginału: Practical Game Development with Unity and Blender

Tłumaczenie: Zbigniew Waśko

ISBN: 978-83-283-0269-3

© 2015 Cengage Learning PTR.

CENGAGE and CENGAGE LEARNING are registered trademarks of Cengage Learning, Inc., within the United States and certain other jurisdictions.

ALL RIGHTS RESERVED. No part of this work covered by the copyright herein may be reproduced, transmitted, stored, or used in any form or by any means graphic, electronic, or mechanical, including but not limited to photocopying, recording, scanning, digitizing, taping, Web distribution, information networks, or information storage and retrieval systems without the prior written permission of the publisher.

Unity is a registered trademark of Unity Technologies. Blender is a trademark of Blender. All other trademarks are the property of their respective owners.

All images © Cengage Learning unless otherwise noted.

© 2015 Helion S.A.

All rights reserved.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie bierze jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Wydawnictwo HELION nie ponosi również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION  
ul. Kościuszki 1c, 44-100 GLIWICE  
tel. 32 231 22 19, 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:  
<ftp://ftp.helion.pl/przyklady/unible.zip>

Drogi Czytelniku!  
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres  
<http://helion.pl/user/opinie/unible>  
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# SPIS TREŚCI



<b>O autorze .....</b>	<b>11</b>
<b>Wstęp .....</b>	<b>13</b>
Niezbędne oprogramowanie .....	14
Czytelnik docelowy .....	15
Czy taka książka jest rzeczywiście potrzebna? .....	17
Czy w tej książce jest wszystko na temat tworzenia gier? .....	17
Jak należy czytać tę książkę? .....	18
Czy ta książka jest jeszcze aktualna? .....	19
Dziesięcioetapowy tok pracy .....	19
Pliki powiązane z książką .....	20
<b>Rozdział 1. Dziesięcioetapowy tok pracy .....</b>	<b>21</b>
Wprowadzenie do dziesięcioetapowego toku pracy .....	22
Etap 1. Burza mózgów .....	24
Etap 2. Projekt wstępny .....	27
Zarys gry .....	28
Opis szczegółów .....	29
Załączniki .....	29
Etap 3. Tworzenie prototypu .....	29
Etap 4. Dopracowanie projektu .....	30
Zarządzanie projektem .....	31
Zinwentaryzuj środki .....	32
Zmniejsz odległości .....	32
Zaplanuj pracę .....	33
Etap 5. Tworzenie elementów gry .....	33
Etap 6. Importowanie elementów gry do silnika .....	34
Etap 7. Projektowanie poziomów .....	36
Etap 8. Kodowanie .....	37
Etap 9. Testowanie .....	39
Etap 10. Budowanie .....	40
Zalecenia praktyczne .....	41
Podsumowanie .....	42

<b>Rozdział 2. Od Blendera do Unity .....</b>	<b>45</b>
Konfigurowanie interfejsu Blendera .....	46
Ciemny motyw .....	46
Etykiety bez pythonowych instrukcji .....	48
Kontrolki z programu Maya .....	49
„Błąd” zamykania bez zapisu .....	51
Przenoszenie modeli z Blendera do Unity .....	53
Pliki .blend .....	54
Ćwiczenie: ręczny eksport do FBX .....	55
Zawartość pliku FBX .....	70
Ćwiczenie: importowanie plików FBX w Unity .....	71
Współrzędne UV mapy światła .....	71
Współczynnik skali .....	73
Podsumowanie .....	74
<b>Rozdział 3. Modułowe środowiska i siatki statyczne .....</b>	<b>77</b>
Zalety metody modułowej .....	79
Rozpoczynanie prac nad środowiskiem modułowym .....	79
Używanie klocka podstawowego .....	83
Opracowywanie modułów w Blenderze .....	83
Odwracanie normalnych .....	84
Ukrywanie ścianek odwróconych tyłem .....	85
Funkcja przyciągania .....	85
N-kąty .....	88
Wyszukiwanie n-kątów .....	91
Cofanie operacji i usuwanie duplikatów .....	92
Mirroring .....	93
Grupowanie wierzchołków .....	94
Parametry wyświetlania siatki .....	97
Mapowanie UV i tworzenie tekstur .....	98
Wyznaczanie szwów, mapowanie UV i modelowanie .....	98
Atlas tekstur i pokrywanie się współrzędnych UV .....	100
Ustalanie gęstości tekselowej .....	102
Importowanie i konfigurowanie środowisk w Unity .....	104
Stosowanie prefabrykatów .....	107
Wsad statyczny .....	109
Podsumowanie .....	109
<b>Rozdział 4. Teren .....</b>	<b>111</b>
Tworzenie terenu w Unity .....	111
Parametry terenu .....	113
Rzeźbienie terenu .....	114
Malowanie terenu teksturami .....	115
Ocena terenów generowanych przez Unity .....	118

Modelowanie terenu w Blenderze .....	119
Metoda edycji proporcjonalnej .....	121
Metoda tekstury przemieszczeń .....	124
Metoda rzeźbienia .....	127
Rozdzielczość terenu .....	130
Malowanie terenu teksturą .....	131
Wyznaczanie współrzędnych UV terenu .....	132
Generowanie tekstury .....	132
Malowanie w oknie UV/Image Editor .....	134
Malowanie w oknie 3D View .....	138
Malowanie teksturami .....	140
Tworzenie dróg i ścieżek .....	143
Modelowanie dróg .....	144
Podsumowanie .....	148
<b>Rozdział 5. Tok pracy animacyjnej .....</b>	<b>149</b>
Klatka kluczowa jako jednostka animacji .....	150
Przygotowanie Blendera do tworzenia animacji .....	151
Wykorzystaj specjalny, animacyjny układ interfejsu Blendera .....	152
Uważaj na automatyczne kluczowanie .....	152
Wstawiaj pojedyncze klatki kluczowe .....	153
Długość animacji .....	155
Eksportowanie animacji do formatu FBX .....	156
Praca z wieloma animacjami .....	156
Prosta animacja kluczowana — od Blendera do Unity .....	158
Animowanie ruchu wzdłuż ścieżki i wypalanie animacji .....	166
Miksowanie kształtów i klucze kształtu .....	172
Kości i rigowanie .....	177
Zawsze nadawaj nazwy poszczególnym kościom .....	178
Szkielety symetryczne a funkcja X-Axis Mirror .....	178
Kinematyka prosta i odwrotna .....	180
Kości deformowane i sterujące .....	182
Eksportowanie zrigowanych postaci .....	182
Importowanie zrigowanych postaci do Unity .....	184
Podsumowanie .....	186
<b>Rozdział 6. Obiekty, zależności i programowanie zdarzeniowe .....</b>	<b>187</b>
Zależności zaprogramowane .....	188
Rozwiązanie DI — projektowanie komponentowe i komunikaty .....	190
Projektowanie komponentowe .....	190
Komunikaty .....	191
Funkcja BroadcastMessage i hierarchie .....	194
Wysyłanie komunikatów do wybranych obiektów .....	195
Wysyłanie komunikatów do obiektów nadrzędnych .....	197

System powiadomień .....	197
NotificationsManager w szczegółach .....	200
Singletony .....	201
Komunikaty a obiekty aktywne .....	202
Przemierzanie hierarchii obiektów .....	203
Podsumowanie .....	204
<b>Rozdział 7. Retopologizacja .....</b>	<b>205</b>
Siatki high-poly, czyli o dużej gęstości .....	206
Siatki wysokorozdzielcze a gry czasu rzeczywistego .....	208
Retopologizacja w praktyce .....	209
Etap 1. Wymodeluj metodą pudełkową wstępną wersję obiektu .....	209
Etap 2. Zwiększ rozdzielczość siatki .....	212
Etap 3. Rzeźbij i dziel .....	216
Etap 4. Retopologizuj .....	221
Dziesiątkowanie .....	231
Podsumowanie .....	234
<b>Rozdział 8 . Zapisywanie stanu gry a trwałość danych .....</b>	<b>235</b>
Dane trwałe .....	236
Preferencje gracza .....	237
Preferencje gracza — ciąg dalszy .....	238
Wybieranie danych trwałych .....	239
Pliki XML — a może JSON lub binarne .....	240
Pliki JSON .....	241
Pliki binarne .....	241
Serializacja klasy .....	242
Przygotowanie danych do serializacji .....	243
Przesyłanie danych do pliku XML .....	245
Odczytywanie danych z pliku XML .....	246
Dodatkowe uwagi na temat klasy SaveState .....	248
Podsumowanie .....	249
<b>Rozdział 9 . Wypalanie .....</b>	<b>251</b>
Czym jest wypalanie? .....	252
Wypalanie oświetlenia statycznego .....	252
Wypalanie oświetlenia dynamicznego .....	252
Wypalanie nawigacji .....	253
Przygotowanie mapowania światła w Unity .....	255
Mapowanie światła. Rozdzielczość mapy światła .....	257
Tryb mapowania światła .....	259
Oświetlenie pośrednie i okluzja otoczenia .....	260
Wypalanie map światła .....	263
Wypalanie map w Blenderze .....	265

Komponowanie renderingów w GIMP-ie .....	271
Wypalanie oświetlenia dynamicznego z użyciem próbników światła .....	274
Wypalanie nawigacji .....	278
Podsumowanie .....	285
<b>Rozdział 10 . Unity, Blender i inne programy .....</b>	<b>287</b>
Inne programy .....	287
MakeHuman .....	288
GIMP .....	288
Inkscape .....	290
MyPaint i Krita .....	292
Synfig Studio .....	293
Tiled .....	293
MonoDevelop .....	294
BMFont .....	295
TexturePacker .....	296
LibreOffice .....	297
Anime Studio Pro .....	298
Audacity .....	298
SFXR .....	299
Podsumowanie .....	300
<b>Dodatek A Inne źródła wiedzy .....</b>	<b>301</b>
Witryny internetowe .....	301
Książki .....	302
Filmy wideo .....	302
<b>Skorowidz .....</b>	<b>303</b>







# OD BLENDERA DO UNITY

*Prostota jest szczytem wyrafinowania.*

— *Leonardo da Vinci*

Po przeczytaniu tego rozdziału powinieneś:

- umieć przygotować Blendera do współpracy z Unity;
- rozumieć zagadnienia związane z przechodzeniem od Blendera do Unity;
- zdawać sobie sprawę z trudności w importowaniu i eksportowaniu siatek;
- umieć obchodzić się z ostrymi krawędziami i modyfikatorem *Edge Split* (rozcinięcie krawędzi);
- umieć eksportować modele w formacie FBX.

Niemal w każdej grze wideo mamy do czynienia z wyświetlaniem modeli złożonych z wielokątów i dlatego zwanych siatkami. Ich geometryczna forma jest definiowana za pomocą wierzchołków, krawędzi i ścianek. W ten właśnie sposób przedstawiane są w zasadzie wszystkie przedmioty widoczne w grze — od elementów uzbrojenia po ludzi i wszelkie potwory. Artyści tworzący na potrzeby gier mają do dyspozycji wiele rozmaitych programów ułatwiających generowanie takich siatek, ale my będziemy używać wyłącznie Blendera. Jest to darmowy i działający na wielu platformach program do modelowania obiektów trójwymiarowych; można go pobrać ze strony: [www.blender.org/](http://www.blender.org/).

## Uwaga

---

Jeśli chciałbyś zobaczyć konkretne przykłady tego, co potrafi Blender, obejrzyj krótki film pt. *Sintel* zamieszczony na stronie: [www.youtube.com/watch?v=eRsGyueVLvQ](http://www.youtube.com/watch?v=eRsGyueVLvQ).

---

Nie będę opisywał Blendera od podstaw. Jest mnóstwo darmowych i komercyjnych poradników oraz kursów obsługi tego programu kierowanych również do początkującego użytkownika. Można tam nie tylko dowiedzieć się wszystkiego na temat interfejsu tego programu i stosowanych w nim skrótów klawiszowych, ale również poznać podstawy modelowania, teksturowania itp.

### Wskazówka

---

Jeśli jeszcze nigdy nie używałeś Blendera albo potrzebujesz odświeżyć swoją wiedzę o nim, zanim przystąpisz do dalszej lektury tego rozdziału, to zajrzyj do dodatku A. W części zatytułowanej „Książki” znajdziesz kilka pozycji, które z pewnością ułatwią Ci wstępne poznanie tego programu.

---

Głównym tematem tego rozdziału jest współpraca Blendera z Unity, ale będzie też okazja, żeby wspomnieć o modelowaniu i teksturowaniu. Jednak zgodnie z tym, co przed chwilą powiedziałem, skoncentruję się przede wszystkim na praktycznym aspekcie współdziałania tych dwóch programów. W szczególności pokażę, jak można skonfigurować interfejs Blendera, aby dało się w nim pracować szybciej i wydajniej. Następnie zobaczysz, na czym polega optymalny sposób eksportowania wymodelowanych siatek do programu Unity z zachowaniem odpowiedniego cieniowania, rozmiarów i struktury. Jeśli w tym momencie nie wiesz za bardzo, o czym mówię, nie przejmuj się. Wkrótce wszystko się wyjaśni.

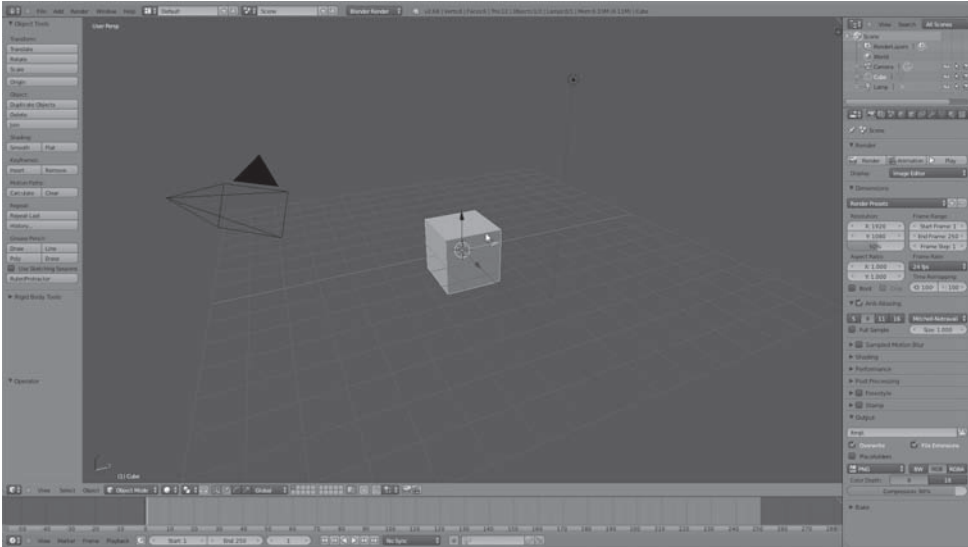
## Konfigurowanie interfejsu Blendera

Gdy uruchomisz Blendera po raz pierwszy i nie zmienisz ustawień domyślnych, jego interfejs będzie wyglądał mniej więcej tak jak na rysunku 2.1. Mówię „mniej więcej”, a nie „dokładnie”, bo Blender jest dość często aktualizowany i jest niemal pewne, że wersja używana przez Ciebie będzie nowsza od mojej, a zatem możesz mieć przed sobą nieco inny widok.

Wielu ludziom odpowiadają te domyślne ustawienia i używają oni Blendera bez zmieniania czegokolwiek. Jednak ja się do nich nie zaliczam. Wprowadzam zmiany, i to dość znaczące. Opiszę je wszystkie wraz z uzasadnieniem, abyś mógł w pełni ocenić, czy moje podejście jest godne naśladowania, czy nie. Oto te modyfikacje.

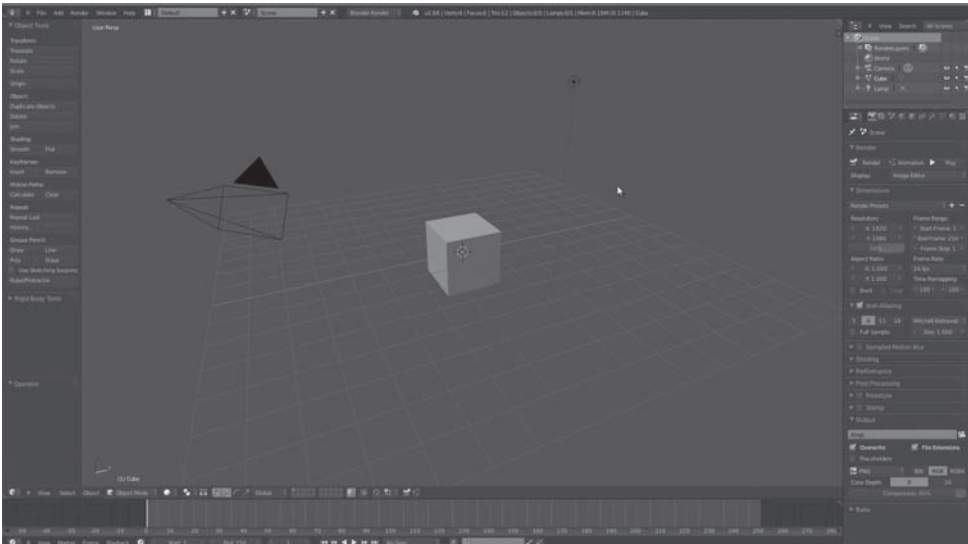
### Ciemny motyw

Kolorystyka interfejsu Blendera jest zdefiniowana w postaci zestawów barw zwanych **motywami**. Motyw domyślny zawiera średnie odcienie szarości pokazane na rysunku 2.1. Stwierdziłem, że przy dłuższej pracy taki interfejs działa jednak rozpraszająco — oko ludzkie w naturalny sposób kieruje się w stronę miejsc jaśniejszych. A skoro naszą uwagę przyciąga interfejs, to trudniej się nam skupić na opracowywanym modelu. Aby temu zaradzić, przyciemniam interfejs, włączając motyw *Elsyiun* pokazany na rysunku 2.2.



**Rysunek 2.1.** Blender w konfiguracji domyślnej (wersja 2.68a). Jeśli masz wersję późniejszą, jej interfejs może wyglądać nieco inaczej

Źródło: Blender.



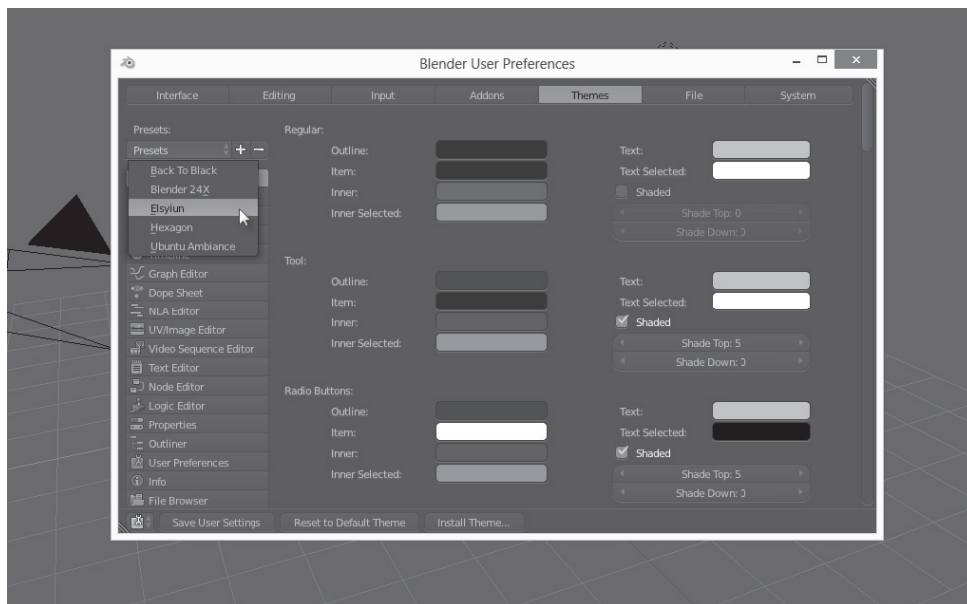
**Rysunek 2.2.** Motyw Elsyiun przyciemnia interfejs Blendera

Źródło: Blender.

Aby włączyć ten motyw kolorystyczny, wykonaj następujące czynności:

1. Wybierz *File/User Preferences* (plik/preferencje użytkownika) lub wciśnij klawisze *Ctrl+Alt+U*.
2. W oknie dialogowym *Blender User Preferences*, które się otworzy, kliknij zakładkę *Themes* (motywy).

3. Rozwiń listę *Presets* (ustawienia predefiniowane) i wybierz pozycję o nazwie *Elsyiun*, tak jak pokazano na rysunku 2.3. Kolory interfejsu natychmiast się zmienią.



**Rysunek 2.3.** W oknie dialogowym *Blender User Preferences* wybierz motyw o nazwie *Elsyiun*

Źródło: Blender.

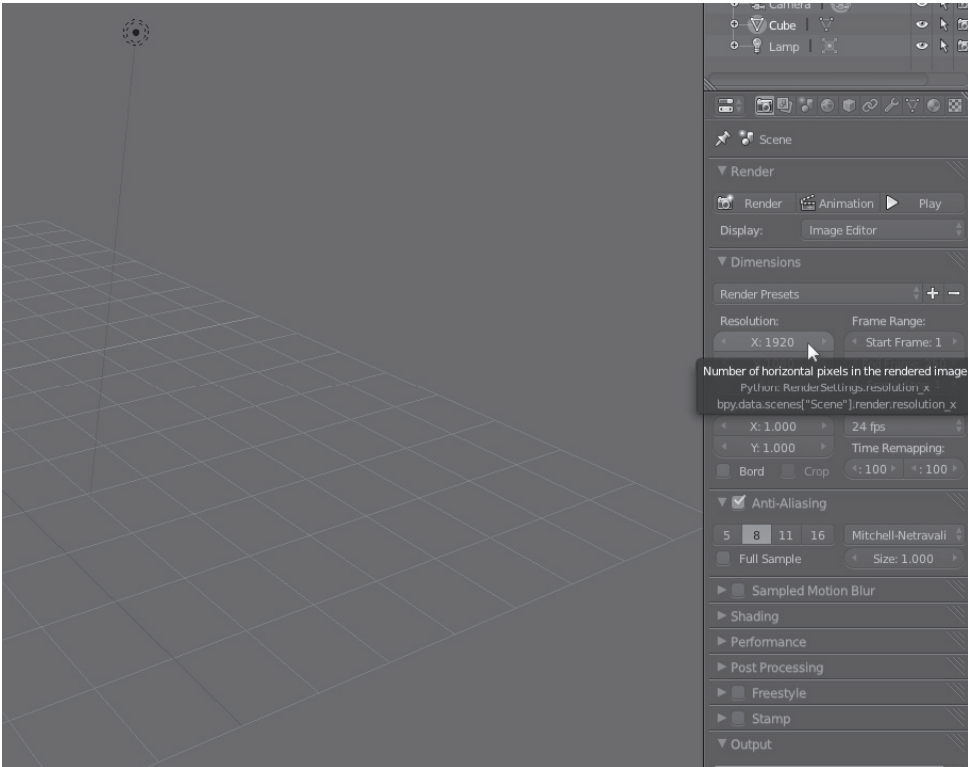
4. W celu zatwierdzenia wprowadzonej zmiany kliknij przycisk *Save User Settings* (zapisz ustawienia użytkownika) widoczny w dolnej części okna preferencji. Następnie zamknij to okno, aby wrócić do głównej przestrzeni roboczej.

Jeśli później uznasz, że jednak motyw domyślny był lepszy, możesz go przywrócić przez kliknięcie przycisku *Reset to Default Theme* (przywróć motyw domyślny) w oknie *Blender User Preferences*.

## Etykiety bez pythonowych instrukcji

Blendera używają zarówno artyści tworzący modele, jak i programiści piszący skrypty rozwijające możliwości programu. Dlatego w ustawieniach domyślnych starano się uwzględnić potrzeby jednych i drugich. Jednak my, twórcy gier, będziemy działać głównie jako artyści, więc niepotrzebne nam będą elementy przydatne programistom.

Jednym z takich elementów są instrukcje w języku Python (język programowania dostępny w Blenderze jako język skryptowy) wyświetlane w etykietkach narzędziowych, które ukazują się, gdy przytrzymasz wskaźnik myszy na jakimś przycisku lub innym elemencie interfejsu (rysunek 2.4). Jako artyście informacje takie nie będą Ci potrzebne. Możesz je śmiało ukryć, aby dodatkowy tekst nie angażował bez potrzeby Twojej uwagi.



**Rysunek 2.4.** Przy domyślnych ustawieniach Blendera w etykietkach narzędziowych wyświetlane są informacje zarówno dla artystów, jak i programistów

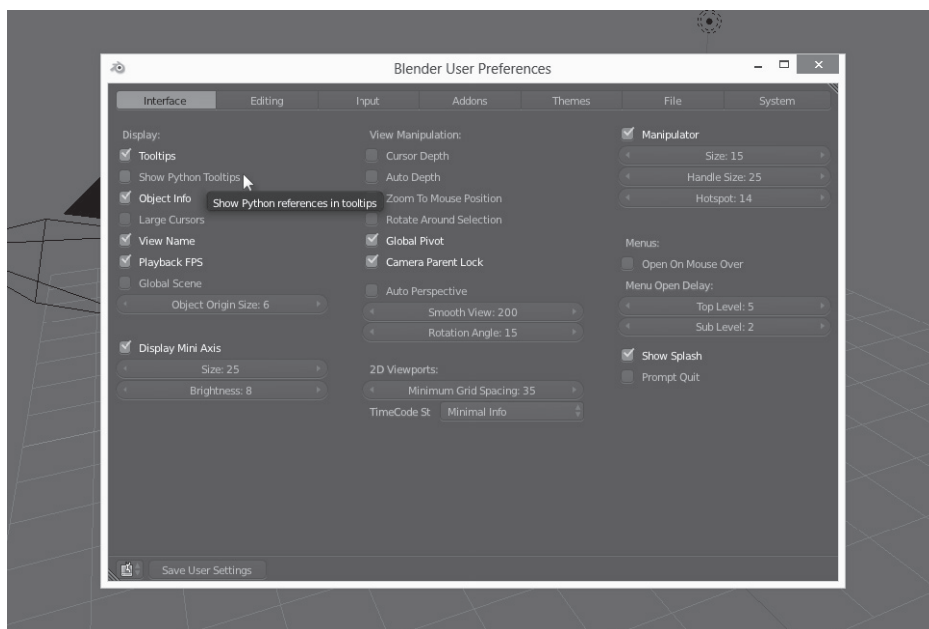
Źródło: Blender.

Aby wyłączyć wyświetlanie informacji pythonowych, wykonaj następujące czynności:

1. Otwórz okno dialogowe *Blender User Preferences* przez wybranie polecenia *File/User Preferences*.
2. Kliknij zakładkę *Interface* (interfejs).
3. Wyłącz opcję *Show Python Tooltips* (pokaż etykiety narzędziowe Pythona) — rysunek 2.5.
4. Kliknij przycisk *Save User Settings* (zapisz ustawienia użytkownika). (Jeśli tego nie zrobisz, dawne ustawienia powrócą po ponownym uruchomieniu programu).

## Kontrolki z programu Maya

Dla mnie największym usprawnieniem w pracy z Blenderem i Unity jest możliwość wykonywania wielu czynności w Blenderze w sposób podobny jak w programie Maya. Przy ustawieniach domyślnych Blender oferuje własny zestaw skrótów klawiszowych i kontrolki do nawigowania w oknach widokowych. Niestety, wszystko odbywa się tu inaczej niż w programie Unity, który pod tym względem nie różni się od programu Maya.



**Rysunek 2.5.** Wyłączenie informacji pythonowych w oknie *Blender User Preferences*

Źródło: Blender.

Ponieważ często będziemy musieli używać Blendera na przemian z Unity, byłoby dobrze, gdyby dało się uniknąć częstego przestawiania się z jednego systemu sterowania na inny. Z doświadczenia wiem, że bez ujednolicenia tych systemów łatwo o pomyłkę, a zastosowanie skrótu blenderowego w Unity (lub na odwrót) nie wróży nic dobrego. Rozwiązanie tego problemu tkwi w predefiniowanym ustawieniu o nazwie *Maya*, za pomocą którego można sprawić, że Blender będzie działał tak jak Unity. Po uaktywnieniu tego ustawienia otrzymujemy jeden spójny system sterowania obowiązujący w obu aplikacjach.

## Uwaga

Ktoś może zapytać, dlaczego upodobiłem Blendera do Unity, a nie na odwrót. Powód jest wręcz banalny: po prostu Blender jest pod względem konfigurowalności bardziej elastyczny i nawet ma już wbudowane gotowe ustawienia, więc po co się męczyć, skoro można to zrobić szybko i skutecznie.

Zmianę systemu sterowania Blenderem można przeprowadzić na dwa sposoby. Pierwszy polega na wybraniu odpowiedniej opcji na ekranie powitalnym, a drugi wymaga posłużenia się oknem dialogowym *Blender User Preferences*. W pierwszym przypadku należy:

1. Uruchomić Blendera.
2. Na ekranie powitalnym rozwinąć listę *Interactions* (interakcje) i wybrać z niej opcję *Maya* — tak jak na rysunku 2.6.



Rysunek 2.6. Wybieranie systemu sterowania Blenderem na ekranie powitalnym

Źródło: Blender.

## Wskazówka

Ekran powitalny można wyświetlić również w trakcie pracy z Blenderem. W tym celu należy kliknąć ikonkę Blendera widoczną w nagłówku okna informacyjnego, który domyślnie znajduje się przy górnej krawędzi interfejsu (na rysunku 2.6 ikonka ta jest zaznaczona kółkiem).

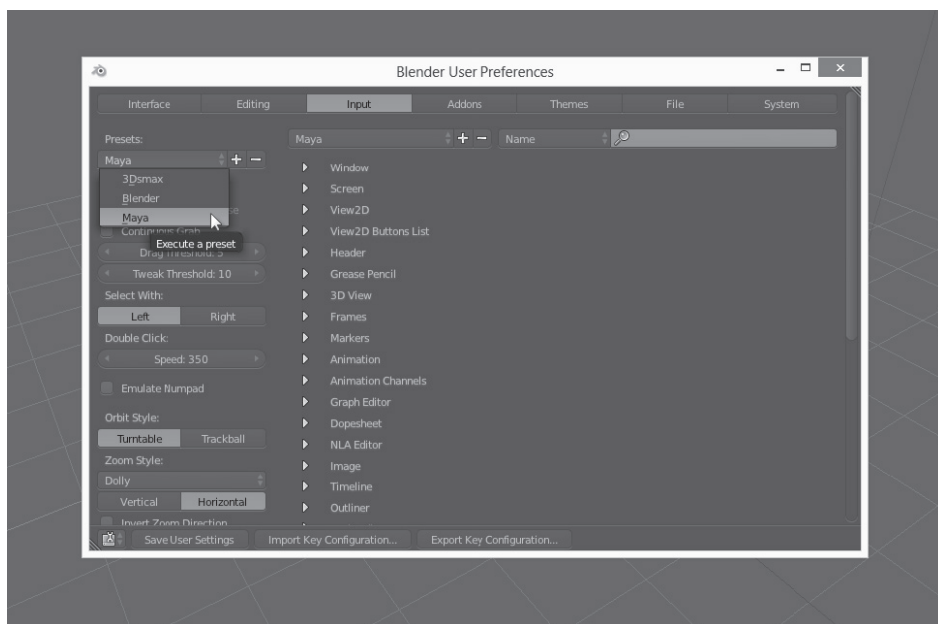
Drugi sposób wymaga następujących czynności:

1. Wybierz polecenie *File/User Preferences*.
2. W oknie dialogowym *Blender User Preferences*, które się otworzy, kliknij zakładkę *Input* (wejście).
3. Rozwiń listę *Presets* i wybierz pozycję o nazwie *Maya* (rysunek 2.7)<sup>1</sup>.
4. W celu zatwierdzenia wprowadzonej zmiany kliknij przycisk *Save User Settings*.

## „Błąd” zamykania bez zapisu

Zanim przejdziemy do bardziej zasadniczych spraw, chciałbym omówić kwestię pozornego błędu, który stał się już sławny w społeczności blenderowej. Otóż istnieje niepisane prawo, które mówi, że każda aplikacja użytkowa powinna reagować na polecenie zamknięcia odpowiednim komunikatem, jeśli użytkownik nie zapisał aktualnego

<sup>1</sup> Listy są dwie (na rysunku rozwinięta jest tylko jedna): jedna odnosi się do operacji wykonywanych za pomocą myszy, a druga do skrótów klawiszowych. Ekran powitalny ustawia obie listy w należyty sposób — *przyp. tłum.*



**Rysunek 2.7.** Ustawianie systemu sterowania Blenderem w oknie dialogowym Blender User Preferences

Źródło: Blender.

stanu swojej pracy. Niemal wszystkie programy, od pakietów Microsoft Office i LibreOffice aż po zaawansowane programy graficzne, takie jak Unity czy GIMP, w pełni respektują to prawo, a Blender nie! Gdy go zamykasz, po prostu znika z ekranu i robi tak niezależnie od tego, czy zapisałeś rezultat swojej pracy, czy nie. A gdy go uruchamiasz ponownie, jakby nigdy nic wyświetla nową, świeżą scenę i nadal nie daje żadnego znaku, że tamta niezapisana scena nadal istnieje i możesz do niej wrócić. Już niejedynemu użytkownikowi udało się wiele godzin pracy zostać bezpowrotnie stracone. W takich chwilach rzeczywiście można wpaść w furję, zwłaszcza jeśli sobie uświadomimy, że przecież można było wszystkiego uniknąć, gdyby tylko Blender „zachował się” w sposób konwencjonalny i wyświetlił stosowne ostrzeżenie.

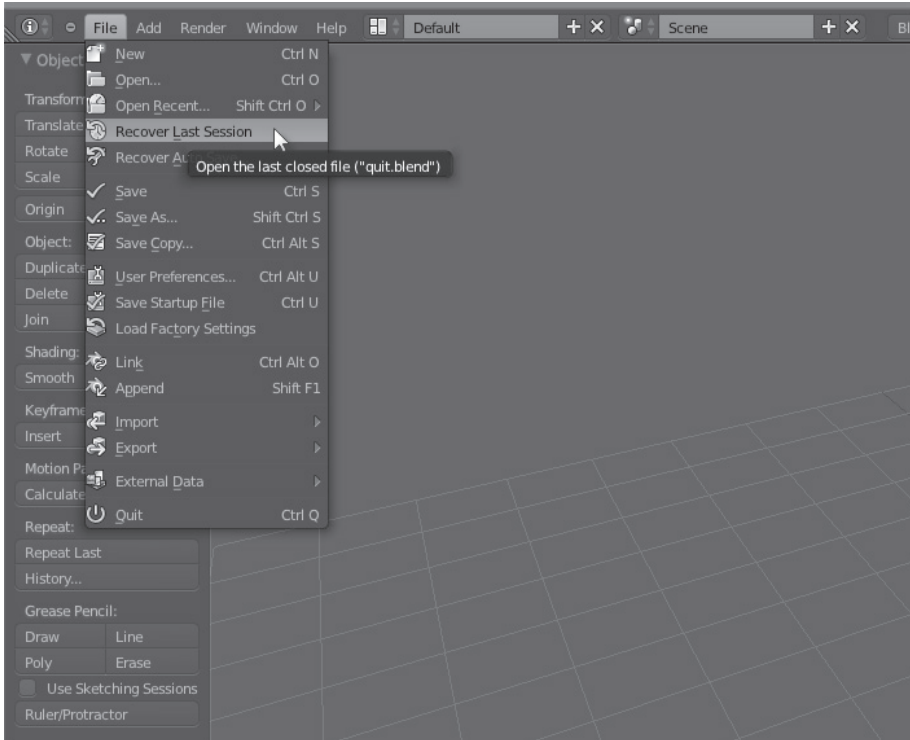
### Uwaga

Niektórzy próbowali uzasadniać „brak tej funkcji”, twierdząc, że użytkownik Blendera powinien być profesjonalistą i dbać o rezultaty swojej pracy. Uważam jednak takie podejście za co najmniej niewłaściwe, ponieważ ignoruje ono fakt, że każdemu (również profesjonalistom) może się zdarzyć przypadkowe zamknięcie aplikacji bez zapisania opracowywanego pliku. Na szczęście problem jest tylko pozorny! Czytaj dalej...

Przekonanie, że Blender pozwala na utratę niezapisanych zmian, jest błędne. Jego twórcy przewidzieli takie sytuacje i zaimplementowali metodę odzyskiwania rezultatów poprzedniej sesji. Wystarczy po prostu wybrać polecenie *File/Recover Last Session*



(plik/odtwórz ostatnią sesję) — rysunek 2.8. Polecenie działa, ponieważ Blender tuż przed zamknięciem zapisuje bieżącą scenę w pliku *quit.blend* i wystarczy ten plik otworzyć. Wydaje się to całkiem proste (i tak jest — wszystko sprowadza się przecież tylko do kliknięcia odpowiedniego polecenia w menu), ale przeświadczenie o niedopracowaniu Blendera powoduje, że możliwość przywrócenia poprzedniej sesji często uchodzi uwadze użytkownika.



**Rysunek 2.8.** Przywracanie poprzedniej sesji w celu odzyskania niezapisanych danych

Źródło: Blender.

Być może to, co napisałem, nie do końca Cię przekonuje i nadal uważasz, że wzorem innych aplikacji Blender powinien wyświetlać ostrzeżenie o niezapisanych zmianach. Proponuję jednak odłożyć te dywagacje na bok. Nie piszę książki o tym, jakie funkcje dany program powinien zawierać, a jakich nie. Piszę o tym, jaki program jest i co można za jego pomocą zrobić. Sądzę, że tylko przy takim nastawieniu da się w pełni wykorzystać te możliwości programu, które w nim tkwią.

## Przenoszenie modeli z Blendera do Unity

Jednym z zagadnień, które są najczęściej dyskutowane w rozmowach na temat toku pracy z użyciem Blendera i Unity, jest przenoszenie modeli wygenerowanych w Blenderze do silnika Unity. Popularną metodą jest zapisanie blenderowej sceny w pliku *.blend*

(natywny format Blendera) za pomocą polecenia *File/Save* (plik/zapisz), a następnie otwarcie tego pliku w Unity. Resztą zajmuje się już Unity — przynajmniej w teorii! Czasami — zależnie od okoliczności — otrzymasz dokładnie to, o co Ci chodziło, i nie musisz niczego poprawiać, ale najczęściej rezultaty będą trudne do zaakceptowania. Może się również zdarzyć, że ta prosta metoda w ogóle nie zadziała. Wyjaśnię teraz, dlaczego tak się dzieje, a potem na konkretnym przykładzie przedstawię właściwe rozwiązanie.

### Uwaga

---

Na razie zajmujemy się eksportowaniem blenderowych modeli do Unity w sensie ogólnym. Pomijamy szczegóły dotyczące modeli animowanych, takich jak rigowane siatki postaci człokształtnych lub innych stworów. Wrócimy do nich w dalszej części książki.

---

## Pliki .blend

Blender domyślnie tworzy pliki w formacie *.blend*. Dlatego gdy wydasz mu polecenie *File/Save* (plik/zapisz), zapisze bieżącą scenę w pliku z rozszerzeniem *.blend*. Na szczęście Unity potrafi odczytywać takie pliki, więc możesz ich używać do przenoszenia danych z jednego programu do drugiego. Po prostu przeciągnij plik z Eksploratora lub Findera i upuść w obrębie interfejsu Unity.

Metoda wydaje się interesująca, ale jest kilka powodów, dla których zdecydowanie odradzam jej stosowanie.

- **Unity do zaimportowania pliku *.blend* potrzebuje Blendera.** Unity odczytuje pliki *.blend*, ale nie bezpośrednio. Tak naprawdę akceptuje tylko formaty: FBX, DAE, 3DS, DXF i OBJ. Gdy ma zaimportować dane z innego pliku, uruchamia w tle program macierzysty (Blendera dla pliku *.blend*) i za jego pomocą konwertuje ten plik na format FBX. A zatem każdy, kto chce importować pliki *.blend* do Unity, musi mieć zainstalowanego Blendera, nawet gdy nie ma zamiaru go używać. Unity potrzebuje go do przeprowadzenia konwersji. Wprawdzie Blender jest darmowy, ale już sama konieczność jego instalacji tylko po to, by otworzyć jakiś plik, może być irytująca — tym bardziej, że da się tego uniknąć (o tym za chwilę).
- **Pliki *.blend* ograniczają liczbę opcji eksportu i zwiększają ryzyko niekompatybilności.** Gdy Unity przeprowadza konwersję, nie masz na ten proces żadnego wpływu. Stosowane są wtedy ustawienia domyślne, które przecież nie zawsze są optymalne, i w rezultacie otrzymujesz niekoniecznie prawidłową siatkę. Oczywiście może się zdarzyć, że wszystko pójdzie dobrze, ale nie mając na nic wpływu, możesz liczyć tylko na szczęśliwy przypadek. Poza tym dochodzi jeszcze ryzyko wynikające z częstych modyfikacji samego Blendera. Jeśli do konwersji pliku zostanie użyta inna wersja niż ta, za pomocą której go wygenerowano, rezultat może być trudny do przewidzenia.

- **Pliki *.blend* są zbyt duże.** Nie jest to zarzut pod adresem tego formatu, lecz stwierdzenie, że przy ustawieniach domyślnych Unity z pomocą Blendera przeniesie do formatu FBX całą zawartość sceny, włącznie z obiektami, które nie powinny być importowane, np.: kamery, lampy, niewidoczne siatki, obiekty pozorne. W Blenderze są przydatne, ale ich import do Unity na ogół nie ma sensu.

---

### Uwaga

Więcej informacji na temat formatów 3D i ich technicznych uwarunkowań znajdziesz w instrukcji obsługi programu Unity dostępnej pod adresem: <http://docs.unity3d.com/Documentation/Manual/3D-formats.html>.

---

## Ćwiczenie: ręczny eksport do FBX

Po przeczytaniu powyższych uwag raczej nie będziesz przynosił modeli z Blendera do Unity za pomocą plików *.blend*. O ile na wczesnym etapie prac, podczas testowania siatek, możesz się jeszcze nimi posługiwać, o tyle do importowania gotowych zasobów powinieneś używać wyłącznie plików w formacie FBX — tym bardziej, jeśli będą to zasoby przekazywane innym członkom zespołu.

Jak już wspominałem, Unity posługuje się wewnątrz formatem FBX nawet wtedy, gdy importuje pliki *.blend*. Ale jeśli sam wyeksportujesz z Blendera plik FBX, będziesz miał większą kontrolę nad jego zawartością niż podczas automatycznej konwersji pliku *.blend*. Ponadto uniezależnisz Unity od innych programów, bo z formatem FBX radzi sobie bez żadnej pomocy z zewnątrz.

W ćwiczeniu tym pokażę na przykładzie zwykłego stożka, jak takie eksportowanie należy przeprowadzać. Naturalnie Twoje modele będą bardziej rozbudowane i skomplikowane, ale wszystko, co tutaj powiem, ma zastosowanie do wszystkich modeli: prostych i złożonych. Więc zaczynamy!

### Tworzenie modeli

To ćwiczenie zaczniemy od zupełnie nowej sceny w Blenderze (zob. rysunek 2.1 lub 2.2). W codziennej pracy do eksportowania będziesz przystępował ze sceną zawierającą gotowe modele. Jednak tym razem usuwamy wszystko z wyjątkiem kamery. W pustej scenie umieść zwykły stożek — wybierz polecenie *Add/Mesh/Cone* (dodaj/siatkę/stożek). Będzie to model, który wyeksportujesz z przeznaczeniem umieszczenia go w Unity (rysunek 2.9).

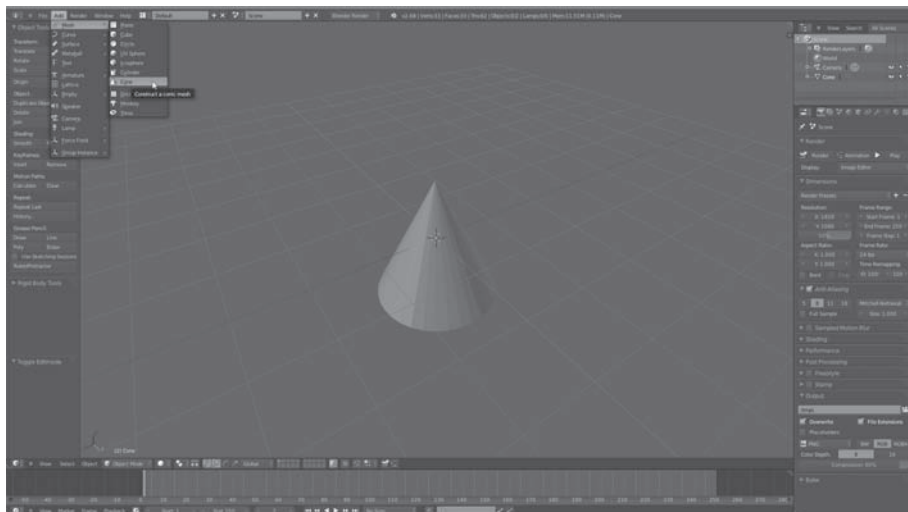
### Radzenie sobie z wadami cieni

---

#### Uwaga

Ten punkt i następny zachowują ważność, pod warunkiem że używasz Blendera w wersji wcześniejszej niż 2.69. Właśnie w tym wydaniu aplikacji dodano funkcję dzielenia normalnych, która likwiduje omawiane tu usterki. Więcej informacji na ten temat znajdziesz pod adresem: [http://wiki.blender.org/index.php/Dev:Ref/Release\\_Notes/2.69](http://wiki.blender.org/index.php/Dev:Ref/Release_Notes/2.69). Jeśli używasz wersji 2.69 lub nowszej, możesz od razu przejść do punktu zatytułowanego „Środek obiektu”.

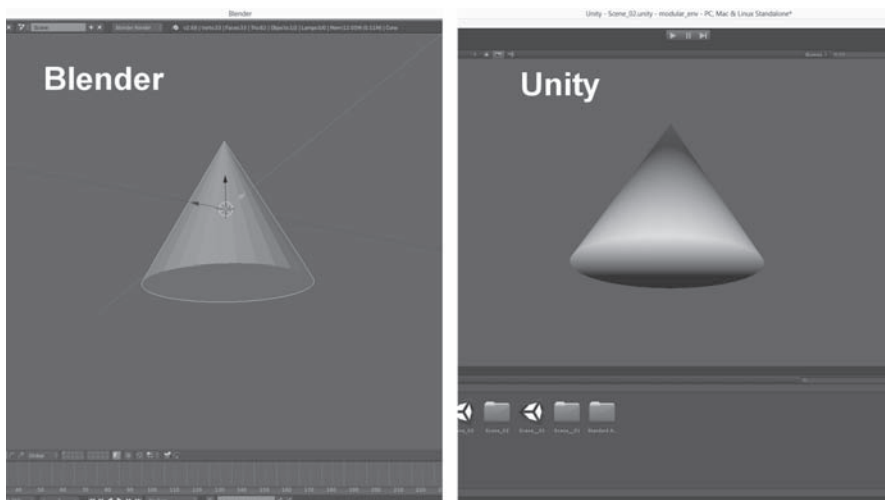
---



**Rysunek 2.9.** W pustej scenie Blendera umieść stożek

Źródło: Blender.

Idźmy dalej. Jeśli od razu przeniesiesz siatkę stożka z Blendera do Unity, czy to w pliku *.blend*, czy FBX (o którym więcej już za chwilę), najprawdopodobniej będziesz miał problem z wyrenderowaniem prawidłowych cieni na jej powierzchni. Zauważysz to, gdy tylko spojrzysz na siatkę w oknie widokowym Unity lub w polu podglądu na panelu *Inspector* (inspektor). Siatka zostanie wygładzona i będzie wyglądała dziwnie. Problem wynika z odmiennego podejścia obu programów do roli, jaką powinny odgrywać normalne. Rozbieżności w wyglądzie stożka wyświetlanego w Blenderze i w Unity są pokazane na rysunku 2.10.

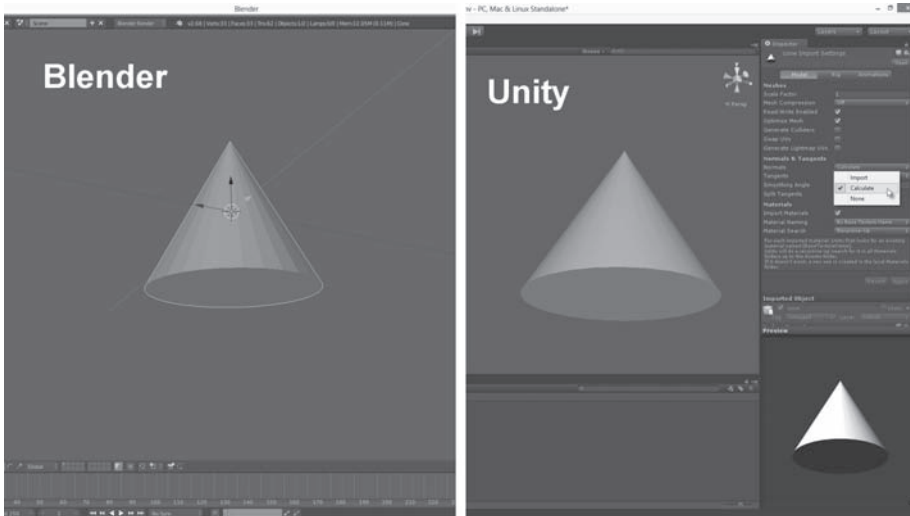


**Rysunek 2.10.** Problem z cieniowaniem powierzchni siatki. Tutaj rozbieżności w cieniowaniu między Blenderem i Unity najlepiej widać przy krawędzi łączącej powierzchnię boczną stożka z jego podstawą

Źródło: Blender.

## Uwaga

Dalszy ciąg tego punktu przedstawia sytuację hipotetyczną, więc nie musisz tego czytać. Chciałem Ci tylko pokazać, na jaki problem możesz natrafić, jeśli postąpisz zgodnie z podanym opisem. Innym efektem, jaki napotkasz, będą dużo mniejsze wymiary siatki w Unity aniżeli w Blenderze. Wynika to z domyślnie przyjmowanej przez Unity skali o wartości 0,01. Możesz się o tym przekonać, sprawdzając tę wartość w panelu *Inspector*, gdy siatka jest zaznaczona (rysunek 2.11). Aby to naprawić, po prostu zwiększ parametr *Scale Factor* (współczynnik skali) do poziomu 1. (Zagadnienie to będzie jeszcze omawiane).



**Rysunek 2.11.** Przez ponowne przeliczenie normalnych w Unity można szybko naprawić błędy cieniowania

Źródło: Blender.

W przypadku modeli tak prostych jak sześciiany, stożki czy kule problem cieniowania można szybko rozwiązać w Unity. Trzeba tylko zaznaczyć właściwą siatkę w panelu *Project* (projekt), po czym w panelu *Inspector* wybrać z listy rozwijanej *Normals* (normalne) opcję *Calculate* (oblicz). (Nie zapomnij kliknąć potem przycisku *Apply*, czyli „zastosuj”, aby zatwierdzić wprowadzone zmiany). Unity na podstawie kątów między ściankami ustali, które krawędzie mają być wygładzone, a które powinny pozostać ostre. W rezultacie otrzymasz stożek wyglądający niemal tak samo jak w Blenderze (zob. rysunek 2.11). Możesz zatem powiedzieć, że między aplikacjami występuje relacja podobieństwa 1:1.

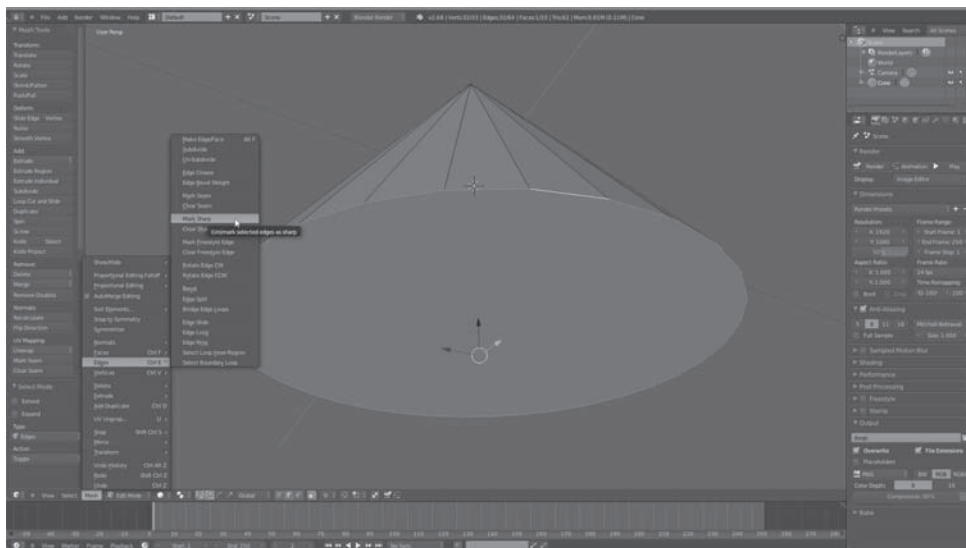
Powyzsza metoda „naprawiania” błędów cieniowania ma jednak istotne ograniczenia. Jest to działanie automatyczne i nie masz żadnego wpływu na to, które krawędzie zostaną zacieniowane ostro, a które łagodnie. W przypadku stożka jest to wystarczające, ale przecież nie tylko takie siatki chciałbyś importować. Dlatego musimy wrócić do Blendera i tam odpowiednio przygotować model do eksportu. A zatem, jeśli wraz ze mną przenośiłeś stożek za pośrednictwem pliku *.blend*, usuń go teraz z Unity i ponownie otwórz Blendera.

### Ostrość krawędzi i modyfikator *Edge Split*

W poprzednim punkcie pokazałem jeden sposób na rozwiązanie problemu złego cieniowania, a teraz zaprezentuję metodę pewniejszą i najczęściej stosowaną.

Jeśli chcesz, żeby po zaimportowaniu do Unity siatka wyglądała prawidłowo i nie ufasz zbyt wielu mechanizmom przeliczania normalnych w tej aplikacji, musisz jeszcze w Blenderze wskazać dokładnie, które krawędzie mają pozostać ostre. **Ostra krawędź** to takie miejsce na powierzchni siatki, gdzie spotykają się dwie ścianki, a ich cieniowania pozostają całkowicie odrębne — nie są w żaden sposób uśredniane. Zazwyczaj są to kany takich brył jak sześcian czy graniastosłup, ale mogą to być także nagłe załamania ścian pod kątem 90°. Natomiast większość obiektów o kształtach organicznych, np. ludzka twarz, ma powierzchnie gładkie i nie ma tam miejsca na ostre krawędzie.

W przykładowym stożku powinieneś wskazać jako ostre krawędzie tworzące obwód jego podstawy, tam bowiem łączą się dwie zupełnie odrębne powierzchnie. Zaznacz więc cały obwód podstawy, po czym wybierz polecenie *Mesh/Edges/Mark Sharp* (siatka/krawędzie/oznacz jako ostre) — tak jak na rysunku 2.12.



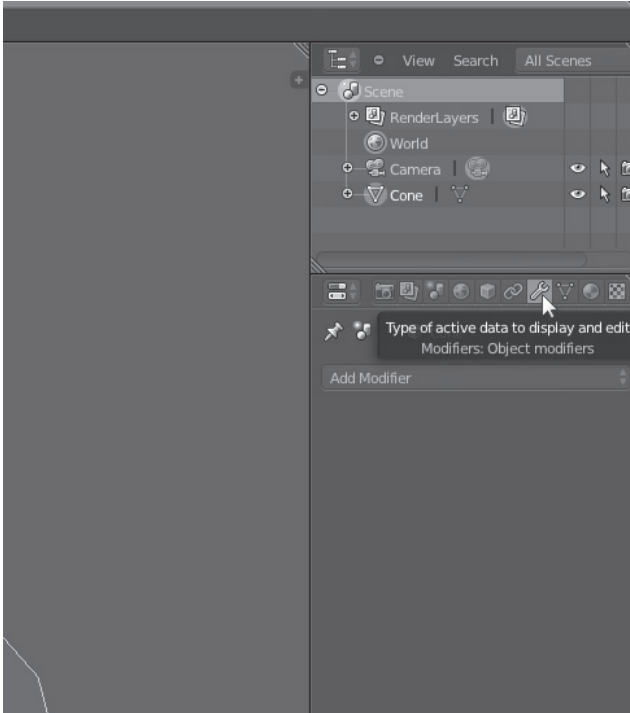
**Rysunek 2.12.** Krawędzie na obwodzie podstawy stożka oznacz jako ostre

Źródło: Blender.

Wskazanie krawędzi, które powinny być ostre, to dopiero pierwszy krok. Teraz trzeba jeszcze poddać siatkę działaniu modyfikatora o nazwie *Edge Split* (rozcinianie krawędzi), który porozcina siatkę wzdłuż ostrych krawędzi i w ten sposób wymusi na Unity odrębne cieniowanie rozłącznych powierzchni. Nie bój się, że przez taki zabieg zniszczysz siatkę, bo wszystko dzieje się w sposób odwracalny — modyfikator działa w Blenderze i zawsze możesz go wyłączyć, a wtedy siatka powróci do stanu pierwotnego (z nierozciętymi krawędziami).

Aby zastosować modyfikator *Edge Split*, wykonaj następujące czynności:

1. Zaznacz siatkę stożka.
2. Otwórz zakładkę *Object modifiers* (modyfikatory obiektu) panelu *Properties* (właściwości) — rysunek 2.13.



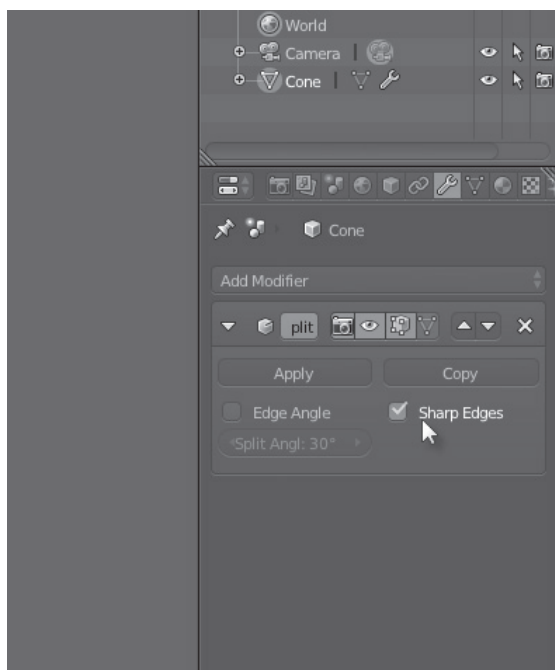
**Rysunek 2.13.** Modyfikator *Edge Split* znajduje się na zakładce *Object modifiers*

Źródło: Blender.

3. Na zakładce *Object modifiers* rozwiń listę *Add Modifier* (dodaj modyfikator) i wskaż opcję *Edge Split*. Modyfikator o takiej właśnie nazwie zostanie przypisany do obiektu stożka.
4. W właściwościach modyfikatora wyłącz opcję *Edge Angle* (kąt krawędzi), ale pozostaw włączoną funkcję *Sharp Edges* (ostre krawędzie) — tak jak na rysunku 2.14. W ten sposób nakażesz Blenderowi rozcięcie siatki tylko wzdłuż tych krawędzi, które oznaczyłeś jako ostre, a nie na podstawie kąta między przyległymi ściankami.

### Ostrzeżenie

Nie klikaj przycisku *Apply* (zastosuj) w właściwościach modyfikatora (zob. rysunek 2.14), bo spowodujesz zatwierdzenie wprowadzonych zmian, a to będzie oznaczało, że staną się nieodwracalne. Po prostu zostaw modyfikator takim, jakim jest. Wrócisz do niego później, gdy będziesz eksportował stożek do pliku FBX.



**Rysunek 2.14.** Dodaj do stożka modyfikator *Edge Split*, aby wyostrzyć oznaczone krawędzie

Źródło: Blender.

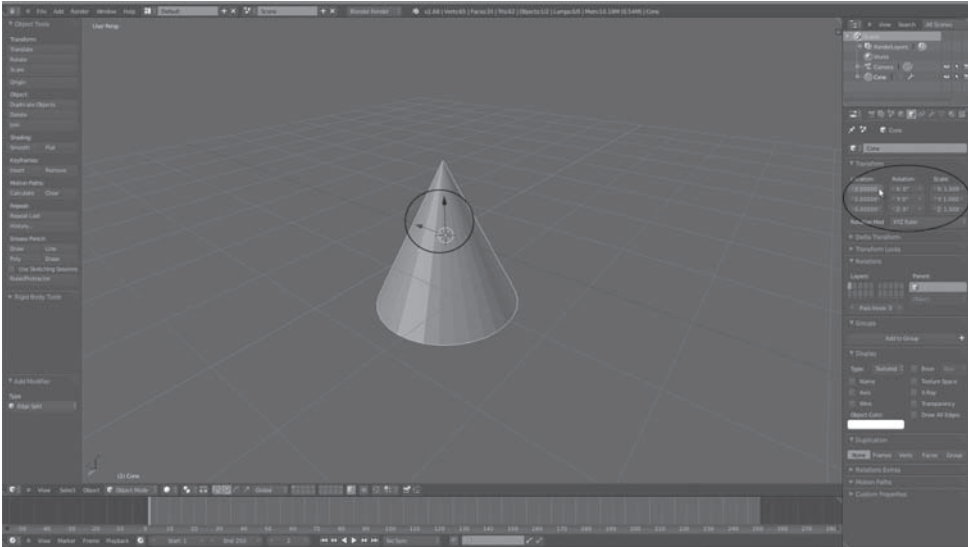
## Uwaga

Zatwierdzenie (zastosowanie) modyfikatora przypisanego siatce wprowadza zmiany „nieodwracalne”, albo inaczej „stałe”, w sensie umownym. Należy to rozumieć jako niemożność odłączenia modyfikatora od obiektu i w ten sposób anulowania skutków jego działania. Jedynym sposobem na przywrócenie stanu pierwotnego jest ręczne zmodyfikowanie siatki — przez ponowne ustawienie wierzchołków, krawędzi i ścianek.

## Środek obiektu

Zarówno w Unity, jak i w Blenderze każdy obiekt ma określony punkt, zwany w Unity środkiem transformacji (*pivot*), a w Blenderze po prostu środkiem (*origin*). (Ja będę używał nazwy blenderowej). Jest to punkt odniesienia dla geometrii siatki. Jeśli ją przesuniesz do nowego położenia o określonych współrzędnych scenicznych, to właśnie tam znajdzie się jej środek. Jeśli ją będziesz obracał, to właśnie ten punkt będzie środkiem obrotu. Krótko mówiąc: jest to początek lokalnego układu współrzędnych siatki. Gdy ją eksportujesz z Blendera, zapewne chciałbyś, aby położenie tego punktu było takie jak należy. Właśnie teraz pokażę Ci, jak to osiągnąć. Na rysunku 2.15 zaznaczyłem zarówno środek stożka, jak i miejsce na zakładce *Object* (obiekt), gdzie możesz odczytać jego położenie we współrzędnych globalnych. (Gizmo transformacji jest umieszczone dokładnie w środku siatki).





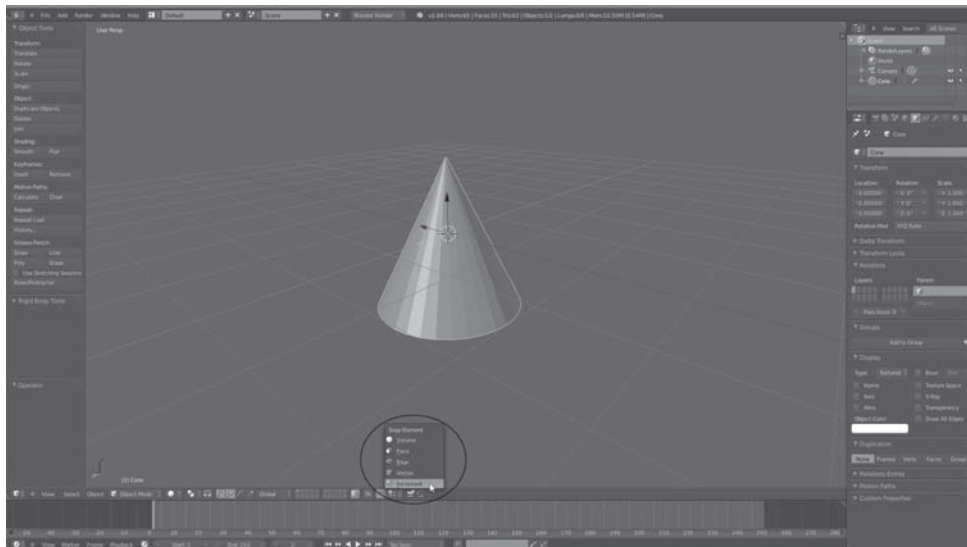
**Rysunek 2.15.** Globalne (światowe, sceniczne) współrzędne położenia zaznaczonego obiektu można odczytać na zakładce Object panelu Properties. W oknie widokowym środek obiektu jest wskazywany przez gizmo transformacji

Źródło: Blender.

Podczas modelowania w Blenderze często środek obiektu ląduje nie tam, gdzie byśmy chcieli. Czasami trzeba go umieścić w ściśle określonym położeniu względem siatki. Na przykład gdy obiektem jest postać człekokształtna, wskazane jest umieszczenie środka jej siatki na dolnej powierzchni stopy, żeby po zaimportowaniu do Unity właśnie tą częścią stykała się z podłożem.

Na rysunku 2.15 środek stożka domyślnie pokrywa się z jego środkiem masy. (Wskazuje na to położenie gizma transformacji). Widać też, że pokrywa się on również z początkiem globalnego układu współrzędnych. Załóżmy, że chcesz go umieścić na podstawie stożka, aby przy współrzędnych globalnych równych (0,0,0) była spoczywała na podłożu, a nie pod nim. Żeby to uzyskać, wykonaj następujące czynności:

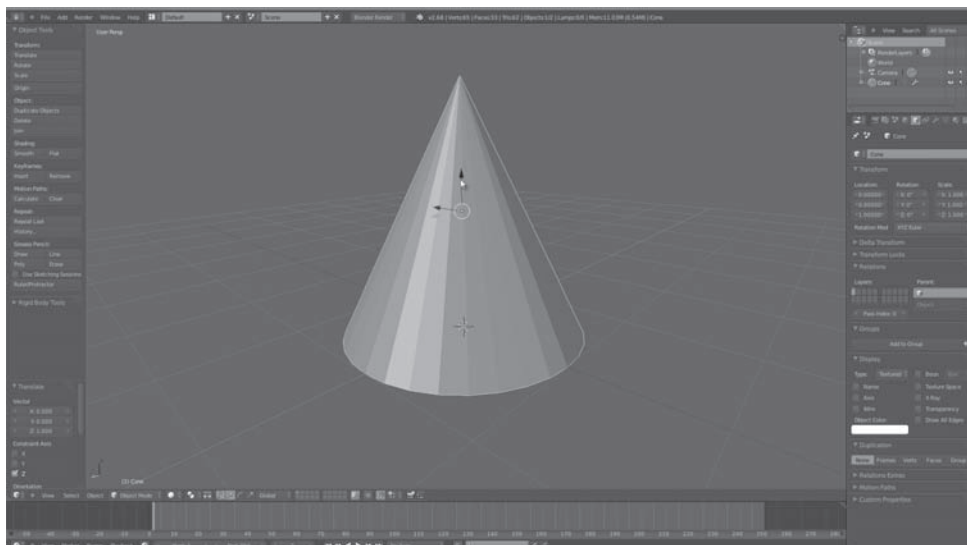
1. W oknie widokowym Blendera zaznacz stożek.
2. Włącz przyciąganie przyrostowe (*Incremental Snapping*) w sposób pokazany na rysunku 2.16. Jeśli teraz będziesz przesuwając stożek, będzie się on przemieszczał skokowo zgodnie z odległościami między liniami blenderowej siatki konstrukcyjnej, a nie płynnie jak przy przesuwaniu swobodnym.
3. Za pomocą gizma transformacji przesun stożek w górę, aby jego podstawa znalazła się w siatce konstrukcyjnej. Zauważ, że w Blenderze oś skierowana do góry to oś Z i w trakcie przesuwania stożka w górę zmienia się wartość tej współrzędnej. Teraz położenie stożka powinno być określone wartościami (0,0,1). Jednak środek siatki przesunął się wraz z nią. Spójrz na rysunek 2.17. Stożek spoczywa na podłożu, tak



**Rysunek 2.16.** Włącz przyciąganie przyrostowe, aby móc przesuwać obiekt skokowo zgodnie z siatką konstrukcyjną

Źródło: Blender.

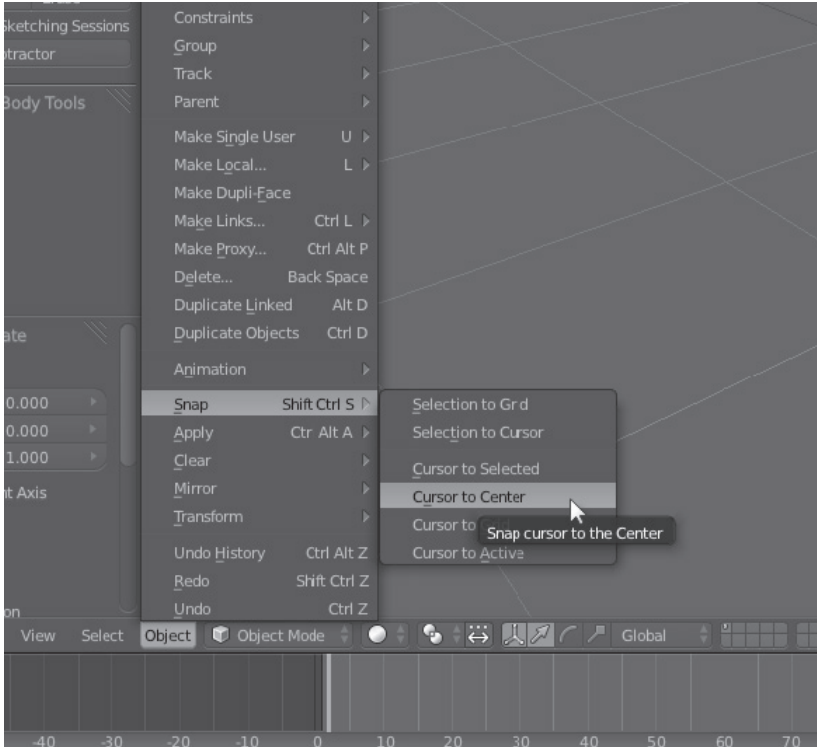
jak chcieliśmy, ale jego środek (wskazywany przez gizmo transformacji) nadal pokrywa się ze środkiem masy, a nie z podłożem. Musimy to tak skorygować, żeby bez przesuwania siatki jej środek znów znalazł się w punkcie o współrzędnych globalnych (0,0,0).



**Rysunek 2.17.** Przesunięcie stożka w górę spowodowało, że jego środek nie jest już w początku globalnego układu współrzędnych

Źródło: Blender.

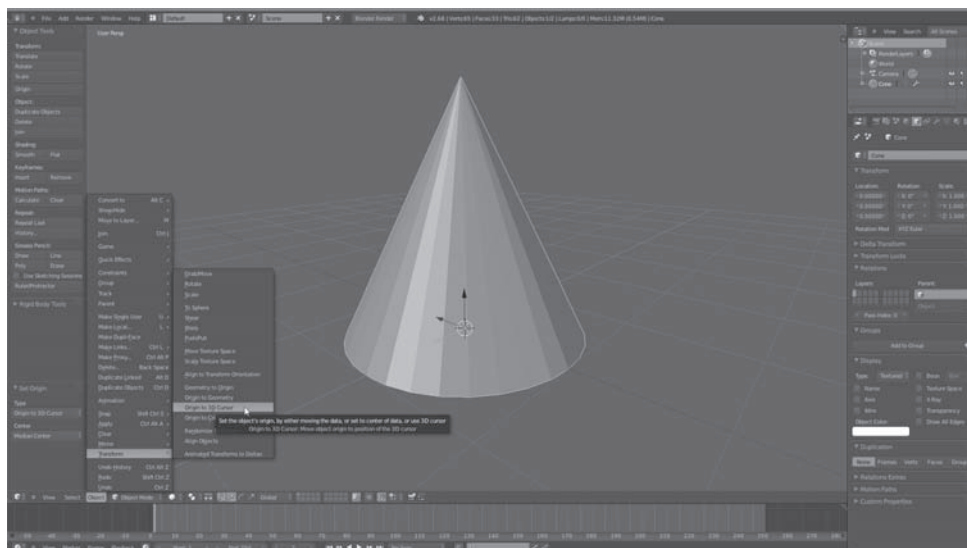
4. Aby z powrotem umieścić środek stożka w początku globalnego układu współrzędnych, użyj blenderowego kursora 3D (widoczny w oknie widokowym jako włosowaty krzyżyk celowniczy). Ustaw ten kursor w początku układu współrzędnych globalnych. W tym celu wybierz polecenie *Object/Snap/Cursor to Center* (obiekt/przyciągnij/kursor do środka) — tak jak na rysunku 2.18.



**Rysunek 2.18.** W ramach przygotowań do przesunięcia środka stożka ustaw kursor 3D w początku globalnego układu współrzędnych

Źródło: Blender.

5. Po ustawieniu kursora 3D w początku globalnego układu współrzędnych przyciągnij do niego środek stożka. W ten sposób początek lokalnego układu współrzędnych siatki pokryje się z początkiem układu globalnego. Aby to zrealizować, zaznacz siatkę stożka (jeśli jeszcze nie jest zaznaczona), a następnie wybierz *Object/Transform/Origin to 3D Cursor* (obiekt/przekształć/środek do kursora 3D). Gizmo transformacji przesunie się do środka globalnego układu współrzędnych, a to będzie oznaczało, że tam też znajduje się środek stożka. Co więcej, współrzędna Z stożka będzie teraz miała wartość 0, a nie, jak dotychczas, 1. Rezultat jest pokazany na rysunku 2.19.



**Rysunek 2.19.** Umieść środek siatki w środku sceny, a proces importu w Unity będzie bardziej przewidywalny

Źródło: Blender.

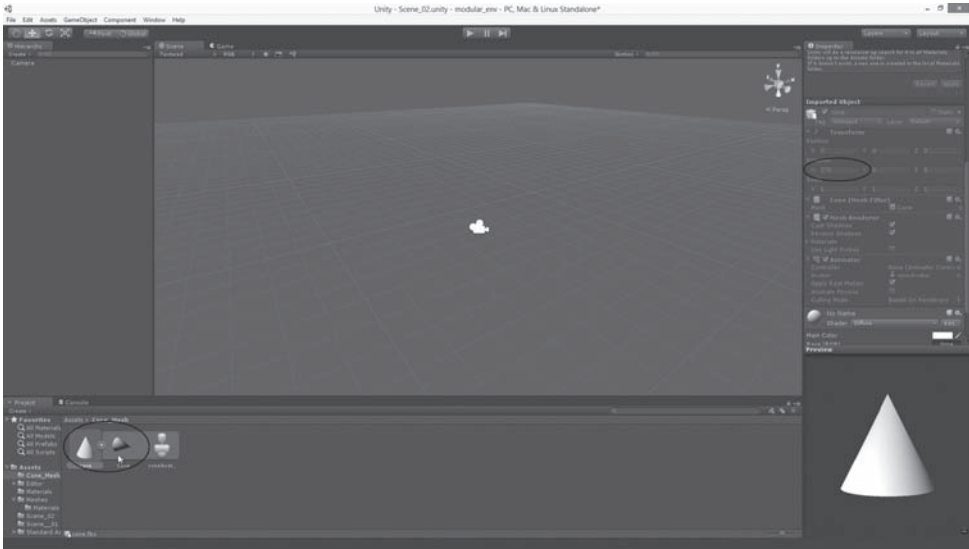
### Obracanie obiektów

Niestety, to nie wszystko. Jest jeszcze kilka rzeczy do zrobienia, jeśli chcesz, aby przenoszenie siatki z Blendera do Unity przebiegło bezboleśnie i zakończyło się w sposób przewidywalny.

Aby zilustrować kolejne zadanie, jakie zostało do wykonania, rozważmy następujący scenariusz. Jeśli teraz przeniesiesz stożek z Blendera do Unity — czy to za pomocą pliku *.blend*, czy FBX — jego wygląd w panelu *Project* na pierwszy rzut oka będzie prawidłowy. Lecz jeśli przyjrzyś się uważniej, zauważysz, że coś jest nie tak z orientacją stożka. Wygląda, jakby był obrócony o  $90^\circ$  lub  $270^\circ$ . I rzeczywiście tak jest (rysunek 2.20). Zaznaczenie siatki w panelu *Project* i sprawdzenie jej transformacji w panelu *Inspector* w pełni potwierdza taką diagnozę. Podczas importu w Unity siatka została obrócona o  $270^\circ$  wokół osi X, chociaż w Blenderze jej obroty wokół każdej osi były zerowe (u Ciebie oś obrotu może być inna). Istnieje tutaj wyraźna rozbieżność. Mamy więc do czynienia z niedopasowaniem domyślnych orientacji obiektu w Blenderze i Unity.

### Uwaga

Jeśli w tym stanie przeciągniesz siatkę z panelu *Project* i umieścisz ją w scenie Unity, najprawdopodobniej jej orientacja zostanie skorygowana automatycznie, a nawet jeśli tak się nie stanie, możesz to zrobić ręcznie za pomocą stosownego manipulatora. Niektórych takie rozwiązanie zadowala, ale mnie nie. Ja zawsze dążę do tego, by importowane siatki miały ustawienia standardowe — położenie (0,0,0), obrót (0,0,0) i skalę (1,1,1). Dzięki temu unikam kłopotów związanych z niewłaściwymi wartościami początkowymi tych trzech transformacji.



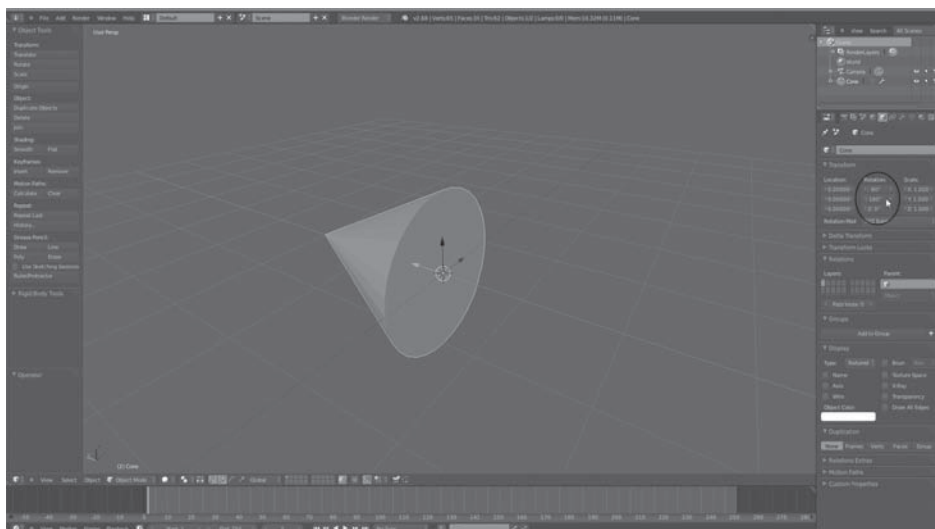
**Rysunek 2.20.** Podczas przenoszenia z Blendera do Unity siatka jest obracana wokół jednej z osi o  $90^\circ$  lub  $270^\circ$

Źródło: Unity Technologies.

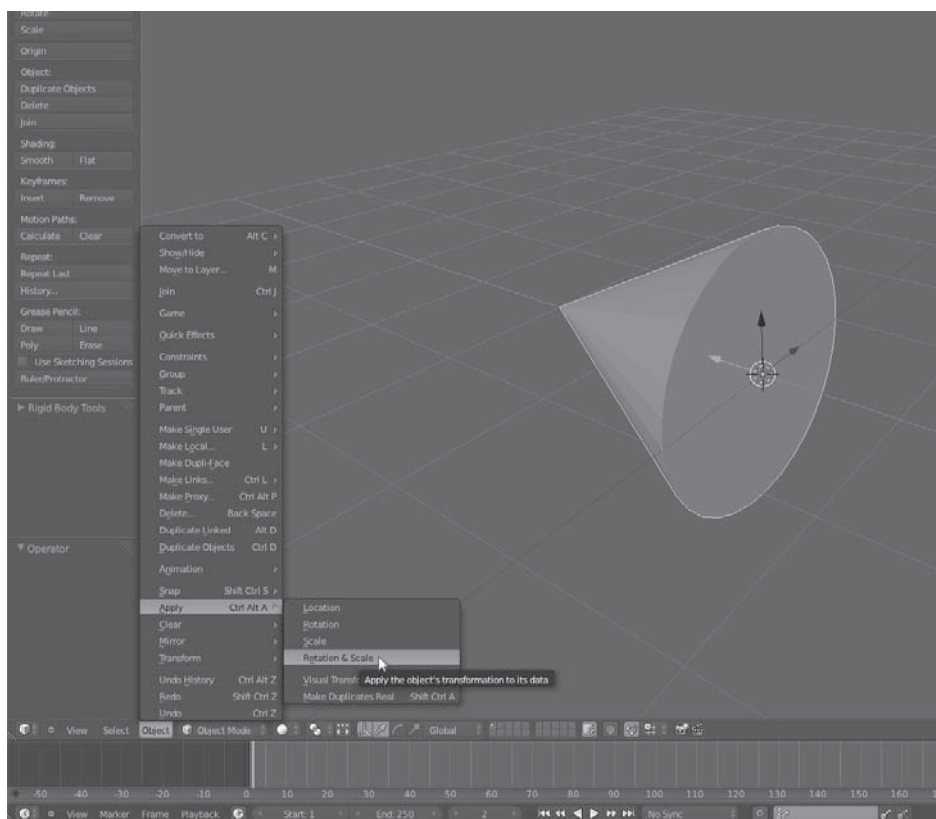
Problem z obrotem wynika z fundamentalnej rozbieżności między układami współrzędnych w Blenderze i w Unity. W Unity osią skierowaną do góry jest Y, a w Blenderze taką osią jest Z. Dlatego przy przejściu do przestrzeni o innej orientacji obiekt jest obracany o  $90^\circ$  lub  $270^\circ$ . Rozwiązanie polega na wykonaniu obrotu kompensacyjnego w programie macierzystym (Blenderze). Zabieg jest dwuetapowy. Pierwszy krok wykonamy teraz, a drugi dopiero po zapoznaniu się z blenderowym eksporterem FBX.

A zatem, jeśli razem ze mną wykonywałeś poprzednie czynności, usuń siatkę stożka w Unity i wróć do Blendera. W oknie widokowym zaznacz stożek i za pomocą odpowiedniej kontrolki w panelu *Properties* obróć go o  $-90^\circ$  wokół osi X. Niektóre siatki będziesz musiał także obrócić o  $180^\circ$  wokół osi Y, aby po zaimportowaniu do Unity były zwrócone do ekranu przodem, a nie tyłem. Spójrz na rysunek 2.21.

Po wykonaniu kompensacyjnego obrotu musisz to przekształcenie zatwierdzić. Chodzi o takie „wypalenie” lub „zakodowanie” tego przekształcenia, żeby z blenderowego punktu widzenia nadal wszystkie obroty miały wartość zerową. Żeby coś takiego osiągnąć, upewnij się, że siatka jest zaznaczona, po czym wybierz polecenie *Object/Apply/Rotation & Scale* (obiekt/zastosuj/obróć i skalowanie) — rysunek 2.22. I to na razie wszystko! Wrócimy do tej kwestii, gdy będziemy eksportować model do pliku FBX.



**Rysunek 2.21.** Przed wyeksportowaniem do formatu FBX nadaj siatce odpowiednią orientację  
*Źródło: Blender.*

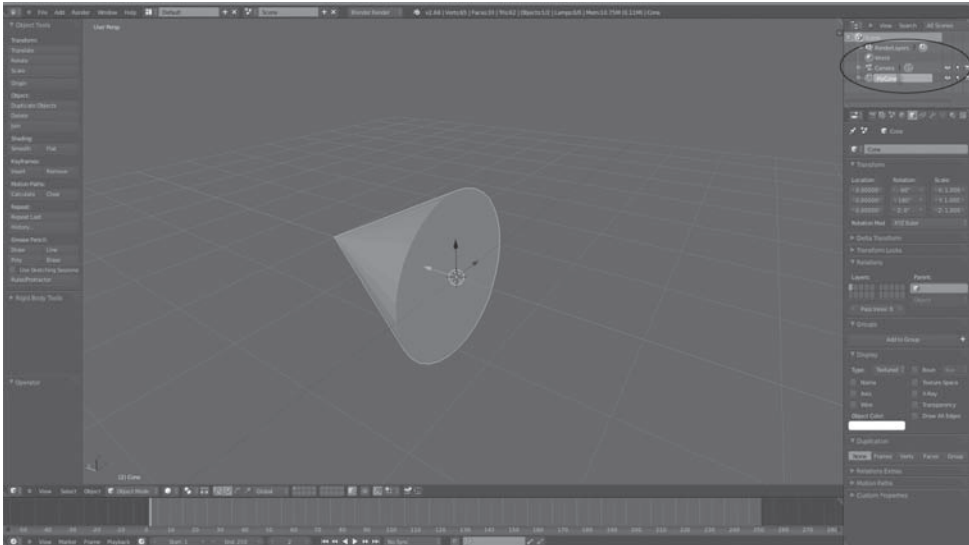


**Rysunek 2.22.** Zatwierdź transformację obiektu  
*Źródło: Blender.*

## Nazywanie obiektów

Ten etap procesu eksportowania jest opcjonalny. Przecież możesz z powodzeniem przenieść model do Unity bez nadawania mu specjalnej nazwy, a dla gracza nie będzie to miało żadnego znaczenia. Lecz jeśli gra ma zawierać dużo rozmaitych modeli, to warto poświęcić trochę czasu i ponadawać im jakieś sensowne nazwy, aby później można było łatwiej znaleźć to, czego się szuka. Nie lekceważ tej sprawy, bo bez dobrego systemu nazewnictwa szybko wkrada się bałagan i organizacja pracy zaczyna kuleć. Dlatego nadawaj nazwy modelom już w Blenderze.

Blender automatycznie przypisuje stożkowi nazwę *Cone* (stożek). Jednak na ogół będziesz chciał zmienić taką domyślną nazwę na bardziej unikatową. Spróbujmy zatem nadać stożkowi nazwę *MyCone* (mój stożek). W tym celu zaznacz bryłę w oknie widokowym, dwukrotnie kliknij jej dotychczasową nazwę w panelu *Outliner* (organizator) i zastąp ją nową (rysunek 2.23). I to wszystko! Teraz możemy rozpocząć eksportowanie siatki do pliku FBX.



**Rysunek 2.23.** Nadaj obiektowi sensowną nazwę, aby później móc sprawniej pracować

Źródło: Blender.

## Eksportowanie do pliku FBX

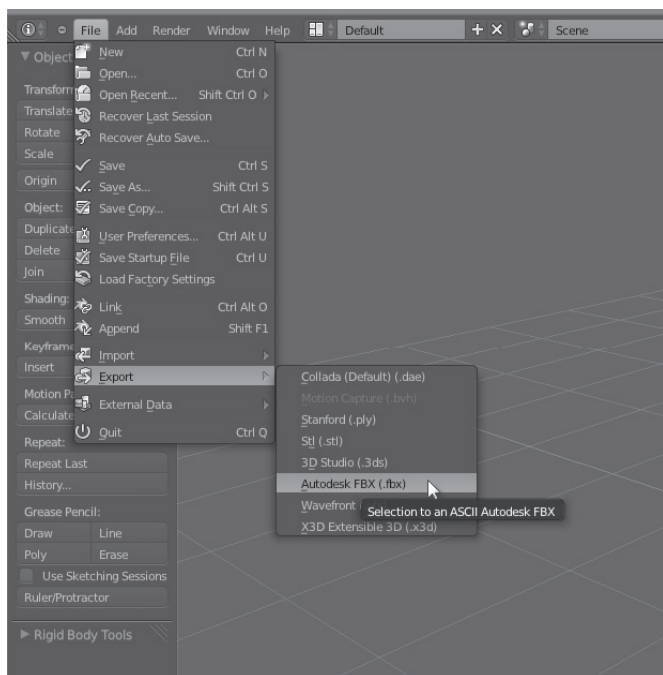
Wszystko, co robiliśmy do tej pory, było przygotowaniem do tego kroku. Teraz użyjemy blenderowego narzędzia eksportującego i za jego pomocą zapiszemy stożek w pliku FBX, aby stamtąd już bez żadnych niespodzianek mógł być zaimportowany w Unity.

Wcześniej uzasadniałem, dlaczego lepiej jest importować siatki w Unity z plików FBX, a nie z plików *.blend*, a teraz, podtrzymując wszystkie tamte stwierdzenia, chcę powiedzieć, że pliki *.blend* również powinny odgrywać ważną rolę w Twojej pracy. Nigdy nie

zapominaj o zapisywaniu swoich scen właśnie w tych plikach, bo tylko dzięki temu będziesz miał do nich pełny dostęp i w razie potrzeby będziesz mógł wprowadzać niezbędne zmiany. Jeśli zdecydujesz się na modyfikowanie siatki, nie wczytuj jej do Blendera z pliku FBX, lecz załaduj ją z pliku *.blend*. Plik FBX traktuj wyłącznie jako medium pośredniczące w przenoszeniu modeli z Blendera do Unity. Dlatego zanim wyeksportujesz siatkę do pliku FBX, zapisz ją w pliku *.blend*, wydając Blenderowi standardowe polecenie *File/Save* (plik/zapisz).

Aby zaznaczoną siatkę wyeksportować do pliku FBX, wykonaj następujące czynności:

1. Zaznacz przeznaczoną do eksportu siatkę i wybierz *File/Export/Autodesk FBX (.fbx)* (plik/eksportuj/Autodesk FBX) — rysunek 2.24. Otworzy się okno narzędzia eksportującego z wieloma opcjami i parametrami eksportu. Dla siatek nieanimowanych, czyli takich jak stożek, najlepsze będą ustawienia pokazane na rysunku 2.25.

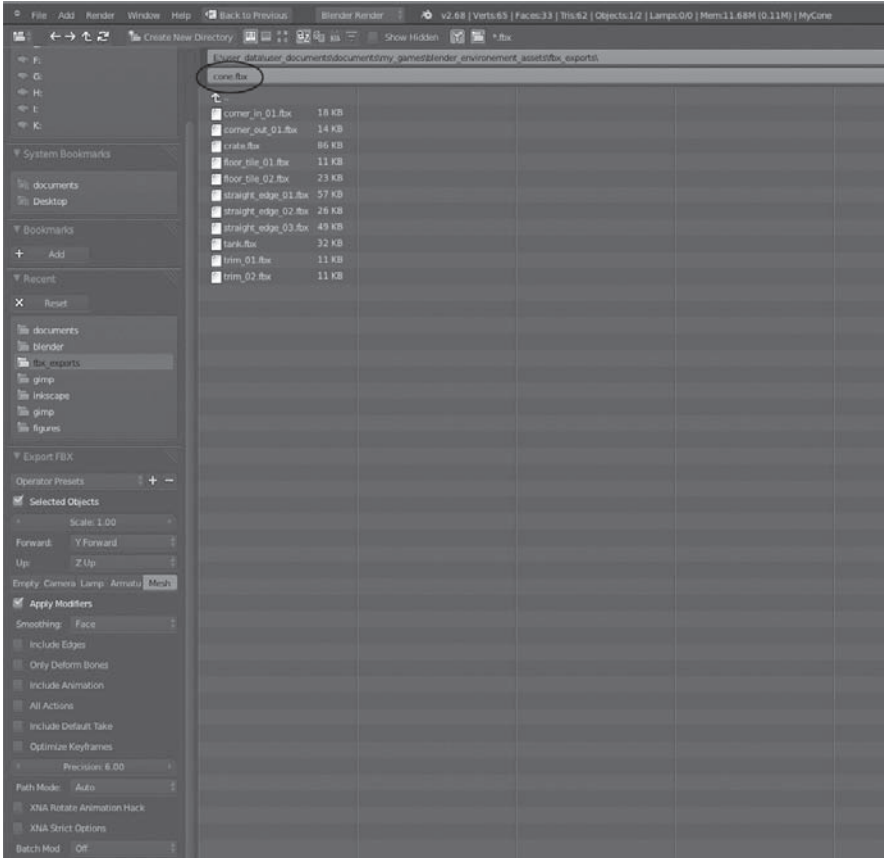


**Rysunek 2.24.** Przed rozpoczęciem eksportu zaznacz właściwą siatkę

Źródło: Blender.

2. Nadaj plikowi sensowną nazwę. Ja wpisałem *Cone.fbx*, co widać na rysunku 2.25. Każda eksportowana siatka powinna mieć własną, unikatową i opisową nazwę.
3. Na rolicie *Export FBX* (eksport FBX) zaznacz opcję *Selected Objects* (zaznaczone obiekty). Spowoduje to, że wyeksportowane zostaną tylko obiekty zaznaczone, a nie cała scena. Jeśli nie włączysz tej opcji, wszystkie siatki zostaną skomasowane w jedną grupę i wyeksportowane.





Rysunek 2.25. Ustawienia eksportu dla siatek statycznych

Źródło: Blender.

## Uwaga

Parametr *Scale* (skala) ma domyślną wartość 1.00 i w większości przypadków byłaby ona odpowiednia. Jednakże, jak się wkrótce przekonasz, Unity zupełnie pomija ten parametr bez względu na jego wartość. W zamian stosuje własną skalę o wartości 0.01, czy tego chcesz, czy nie. Później pokażę, jak sobie z tym problemem radzić.

- Określ kierunki *Forward* (w przód) i *Up* (w górę). Jest to drugi etap procesu rozpoczętego w punkcie „Obracanie obiektów”. Ustawienia te w połączeniu z odpowiednią orientacją obiektu w scenie gwarantują mu właściwą orientację również po zaimportowaniu w Unity. Zazwyczaj kierunki te należy ustawiać zgodnie z kierunkami osi w Blenderze, czyli w polu *Forward* należy ustawić *Y Forward*, a w polu *Up* — *Z Up*. Czasami będziesz musiał trochę pokombinować z tymi ustawieniami, zanim siatka zostanie zaimportowana ze standardowymi wartościami transformacji, czyli z położeniem (0,0,0), obrotem (0,0,0) i skalą (1,1,1), a mimo to będzie miała właściwą orientację.

5. Upewnij się, że zaznaczona jest opcja *Apply Modifiers* (zastosuj modyfikatory). Jest to konieczne, aby wszystkie modyfikatory przypisane siatce, włącznie z *Edge Split* (opisanym w punkcie „Ostrość krawędzi i modyfikator Edge Split”), zostały zaktualizowane w eksportowanej siatce. (Opcja ta nie spowoduje zatwierdzenia żadnego modyfikatora w pierwotnej blenderowej scenie; ma wpływ jedynie na siatkę, która jest generowana podczas eksportu i zapisywana w pliku FBX).
6. Dla siatek nieanimowanych (statycznych) wyłącz opcje *Include Animation* (dołącz animację) i *Optimize Keyframes* (optymalizuj klatki kluczowe). O eksportowaniu siatek animowanych będzie mowa w dalszej części książki.

## Uwaga

Zamiast FBX możesz użyć formatu Collada (.dae). Użyj wtedy polecenia *File/Export/Collada* (plik/eksportuj/Collada). Format ten obsługuje siatki zarówno statyczne, jak i animowane. Więcej informacji na jego temat znajdziesz pod adresem: <https://collada.org/>.

## Zawartość pliku FBX

Podczas eksportowania Blender tworzy plik FBX możliwy do odczytania w Unity. Przyjrzyjmy mu się nieco dokładniej. Warto to zrobić, bo plik zawiera czytelne dla człowieka (tekst ASCII) i edytowalne zapisy właściwości eksportowanej siatki. W razie potrzeby można tu jeszcze coś poprawić. Do przeglądania i edycji zawartości takiego pliku można użyć dowolnego edytora tekstowego, takiego jak Notepad, Notepad++ czy nawet MonoDevelop (rysunek 2.26).

**Rysunek 2.26.** Pliki FBX zawierają czytelny dla człowieka tekst ASCII, który można modyfikować w dowolnym edytorze tekstowym lub zintegrowanym środowisku programistycznym (IDE)

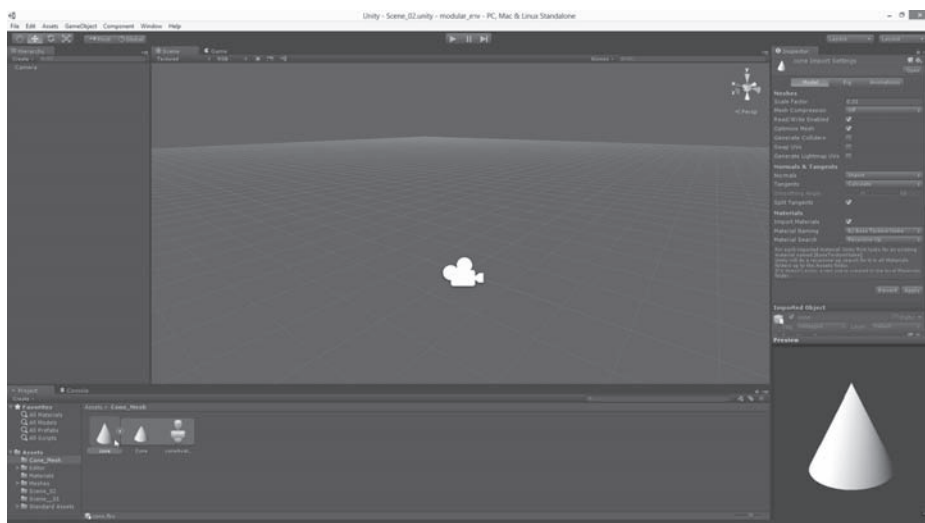
Źródło: MonoDevelop.

## Uwaga

Teoretycznie ręczne manipulowanie w prawidłowo wyeksportowanych plikach FBX nie powinno być potrzebne. Ale mimo wszystko warto wiedzieć, że taka możliwość istnieje. Warto też wiedzieć, że pliki te można analizować i przetwarzać również w trybie wsadowym.

## Ćwiczenie: importowanie plików FBX w Unity

Po wyeksportowaniu siatki FBX z Blendera zapewne będziesz chciał ją zaimportować w Unity, aby tam dołączyć ją do gry. W tym celu po prostu przeciągnij odpowiedni plik z Eksploratora (Windows) lub Findera (Mac OS) na panel *Project* w Unity (rysunek 2.27). Ale to jeszcze nie wszystko. Najprawdopodobniej pewne ustawienia będą wymagały dostosowania. W tym ćwiczeniu zajmiemy się również takimi sprawami.



Rysunek 2.27. Importowanie plików FBX do panelu *Projekt* w Unity

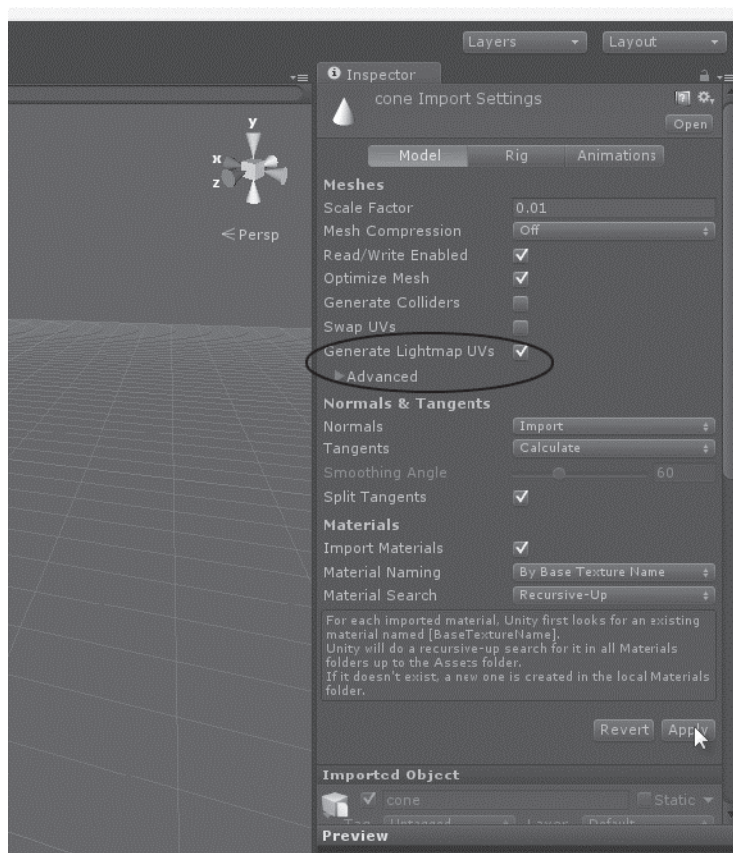
Źródło: Unity Technologies.

## Współrzędne UV mapy światła

Gdy będziesz importował statyczną siatkę terenu, budowli lub innego elementu środowiska gry, zapewne będziesz chciał ją oświetlić metodą mapowania światła. Do realizacji tego rodzaju oświetlenia możesz użyć narzędzia *Beast Lightmapper* dostępnego w Unity po wybraniu polecenia *Window/Lightmapping* (okno/mapowanie światła). **Mapowanie światła** polega na określaniu, jak dana siatka reaguje na panujące w jej otoczeniu warunki oświetleniowe — jakie rzuca cienie, jakie tworzy odbłaski — i zapisywaniu takich informacji w formie specjalnej tekstury. Tekstura taka, zwana **mapą światła**, jest potem nakładana na powierzchnię siatki, co sprawia, że ta ostatnia wygląda, jakby rzeczywiście była oświetlona.

Dzięki takim zabiegom ani procesor graficzny, ani główny nie muszą na bieżąco obliczać efektów oświetleniowych. Jednak żeby coś takiego się udało, zaimportowane siatki muszą mieć przypisane współrzędne mapy światła. Współrzędne te to specjalne dane matematyczne określające, w jaki sposób tekstura z mapą światła ma być odwzorowana na powierzchni siatki. Bez nich mapowanie światła może dać wynik daleki od oczekiwanego. Wartości tych współrzędnych możesz wyznaczyć ręcznie w Blenderze, ale możesz też zlecić to zadanie odpowiednim funkcjom Unity.

Jeśli zdecydujesz się na ten drugi sposób, zacznij od zaznaczenia siatki w panelu *Project*, aby wyświetlić jej właściwości w panelu *Inspector*. Potem po prostu włącz opcję *Generate Lightmap UVs* (generuj współrzędne mapy światła) i kliknij przycisk *Apply* (zastosuj) — tak jak na rysunku 2.28. W większości przypadków otrzymasz rezultat prawidłowy, kłopot mogą sprawić jedynie siatki o kształtach okrągłych lub organicznych i wtedy będziesz musiał sam wyznaczyć te współrzędne za pomocą odpowiednich narzędzi Blendera. Więcej informacji na ten temat znajdziesz w następnym rozdziale.

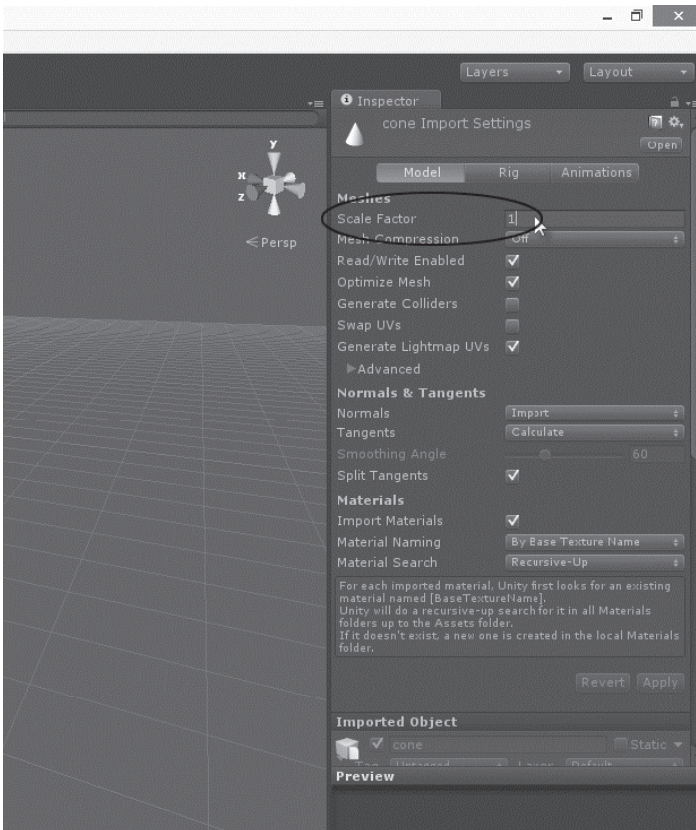


**Rysunek 2.28.** Automatyczne generowanie współrzędnych mapy światła dla zaimportowanej siatki

Źródło: Unity Technologies.

## Współczynnik skali

Dla każdej importowanej siatki FBX Unity automatycznie ustala współczynnik skali o wartości 0,01 i nie ma tu znaczenia, jaka wartość jest zapisana w importowanym pliku. W rezultacie siatka przyjmuje rozmiary dużo mniejsze niż te, które powinna mieć zgodnie z projektem. Czasami będziesz musiał wykonać mocne zbliżenie, żeby ją w ogóle zobaczyć. Problem można rozwiązać przez zwiększenie wartości parametru *Scale Factor* (współczynnik skali) do poziomu 1 (rysunek 2.29). Jednakże przy większej liczbie importowanych siatek lub wielokrotnym importowaniu tej samej siatki (ze względu na częste modyfikacje) ciągle powtarzanie tej samej czynności szybko staje się męczące.



**Rysunek 2.29.** Zmiana współczynnika skali na 1 (zamiast 0,01)

Źródło: Unity Technologies.

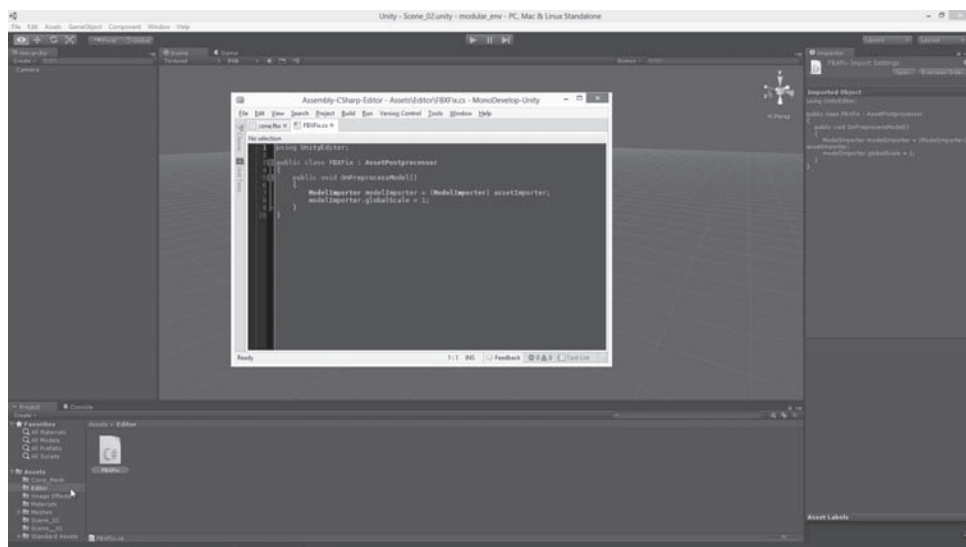
Metoda alternatywna polega na napisaniu skryptu, który zmusi Unity do określonego zachowania podczas importowania siatek — w tym przypadku do ustawiania za każdym razem współczynnika skali o wartości 1. Skrypt będzie działał automatycznie i już nigdy nie będziesz musiał ręcznie poprawiać tego parametru. Aby to uzyskać, utwórz w panelu *Project* nowy skrypt w języku C# i nazwij go `FBXFix.cs` (poprawianie FBX). Następnie umieść w tym skrypcie następujący kod:

```

using UnityEditor;
public class FBXFix : AssetPostprocessor
{
    public void OnPreprocessModel()
    {
        ModelImporter modelImporter = (ModelImporter) assetImporter;
        modelImporter.globalScale = 1;
    }
}

```

Gotowy skrypt umieść w folderze *Editor* (edytor). (Jeśli taki folder nie istnieje w bieżącym projekcie, utwórz go). Żeby skrypt działał, musi się znajdować właśnie w takim folderze (rysunek 2.30). Aby go przetestować, zaimportuj nowy plik FBX. Współczynnik skali powinien mieć teraz wartość 1!



**Rysunek 2.30.** Odpowiedni skrypt w folderze *Editor* ustawi współczynnik skali importowanej siatki na 1

Źródło: Unity Technologies.

A zatem możesz już eksportować modele z Blendera i importować je do Unity z zachowaniem niemal pełnej wierności.

## Podsumowanie

Rozdział ten poświęciłem opisowi czynności związanych z przenoszeniem modeli między Blenderem a Unity. Mówiąc krótko: gdy dwa różne programy wymieniają między sobą dane, bardzo często część tych danych jest tracona lub zmieniana. W przypadku Blendera i Unity do przenoszenia danych można użyć plików *blend* lub FBX. W tym rozdziale starałem się wykazać, że pliki FBX nadają się do tego celu znacznie lepiej. Ale jak miałeś okazję się przekonać, temat jest znacznie szerszy niż tylko wybór

formatu plików. Żeby siatka przeniesiona z Blendera do Unity wyglądała jak najlepiej, trzeba wykonać jeszcze kilka dodatkowych czynności. Niezbędne jest oznakowanie ostrych krawędzi siatki, nadanie jej odpowiedniej orientacji, właściwe umiejscowienie środka transformacji itp. Wszystko, co tutaj napisałem, odnosi się do każdej siatki, niezależnie od jej rodzaju. Siatki animowane mają dodatkowe wymagania, ale tym zagadnieniem zajmiemy się dokładniej w jednym z dalszych rozdziałów. Natomiast tematem następnego rozdziału będzie specyficzny rodzaj siatek statycznych, mianowicie modułowe siatki środowiskowe.





# SKOROWIDZ



## A

agent, 282  
animacja, Animation, 149, 152  
    automatyczne kluczowanie, 152  
    eksportowanie, 156  
    klatka kluczowa, 150  
    liczba klatek, 155  
    retargetowanie, 161  
    wypalanie, 166, 170  
    wzdłuż ścieżki, 166  
animowanie ruchu, 166  
animuj ścieżkę, Animate Path, 169  
aplikacja  
    Anime Studio Pro, 298  
    Audacity, 298  
    Blender, 14  
    BMFont, 295  
    GIMP, 14, 271, 288  
    Inkscape, 290  
    Krita, 292  
    LibreOffice, 297  
    MakeHuman, 288  
    MonoDevelop, 294  
    MyPaint, 292  
    SFXR, 299  
    Synfig Studio, 293  
    TexturePacker, 296  
    Tiled, 293  
    Unity, 14  
aproxymowanie, 229  
atak na obiekty, 201  
atlas tekstur, 100

automatyczne  
    generowanie współrzędnych, 72  
    kluczowanie, 152  
awatar, 186

## B

Blender, 45  
    tworzenie animacji, 151  
    tworzenie terenu, 119  
    wypalanie map, 265  
błędy, 30  
błędy cieniowania, 55, 57  
budowanie, 40  
budowanie grafu animatora, 165  
burza mózgów, 24

## C

cel, Target, 168  
cel przyciągania, Snap Target, 225  
chmury, Clouds, 141  
ciemny motyw, 46  
cienie, 55  
cieniowanie, Shading, 56, 85  
cień kontaktowy, contact shadow, 262  
cofanie operacji, 92  
cykliczność, 145

## D

dane kolizyjne, 105  
dane trwałe, persistent data, 235  
definiowanie kluczy kształtu, 174  
DI, Dependency Injection, 190

długość, Length, 119  
 animacji, 155  
 łańcucha, Chain Length, 180  
 dodaj teksturę, Add Texture, 116  
 dodatek Texture Paint Layer Manager, 137  
 dodawanie tekstury, 116, 117  
 dołącz animację, Include Animation, 70  
 domyślna orientacja obiektu, 64  
 dopracowanie projektu, 30  
 droga, 143  
 duplikaty, 92  
 dzielenie siatki, 121  
 dziesiętkowanie, Decimate, 130, 231  
 dziesięcioetapowy tok pracy, 21

## E

edycja proporcjonalna, 121, 123  
 edytor  
 współrzędnych UV, 103  
 wykresów, Graph Editor, 159  
 edytuj tekstury, Edit Textures, 116  
 eksport do FBX, 55  
 eksportowanie  
 animacji, 156, 157  
 do formatu FBX, 66  
 do pliku FBX, 67  
 postaci, 157  
 zrigowanej siatki, 183, 184  
 zrigowanych postaci, 182  
 elementy gry, 33  
 etap  
 budowanie, 40  
 burza mózgow, 24  
 dopracowanie projektu, 30  
 importowanie elementów gry, 34  
 kodowanie, 37  
 projekt wstępny, 27  
 projektowanie poziomów, 36  
 testowanie, 39  
 tworzenie elementów gry, 33  
 tworzenie prototypu, 29  
 etykiety, 48

## F

filtr siatki, Mesh Filter, 190  
 folder  
 Editor, 74  
 Preferences, 239

format  
 Collada, 182  
 DAE, 184  
 FBX, 55, 156, 183  
 JSON, 241  
 formaty  
 dźwięku, 35  
 filmów, 35  
 obrazów, 35  
 plików, 35  
 siatek, 35  
 FSM, finite state machine, 164  
 funkcja  
 ApplyDamage, 194  
 Auto-Key, 152  
 BroadcastMessage, 194  
 DealDamage, 189, 192  
 OnCastFireBallSpell, 189  
 Proportional Editing, 121  
 przyciągania, 85, 88  
 SendMessage, 194, 202  
 Sharp Edges, 59  
 X-Axis Mirror, 178, 179

## G

gatunek, 25  
 GDD, game design document, 27  
 generowanie  
 chmur, 125  
 siatki nawigacyjnej, 279  
 tekstury, 124, 132, 140  
 terenu, 112  
 współrzędnych UV, 106  
 generuj zderzacze, Generate Colliders, 105  
 gęstość  
 siatki, 120  
 tekselowa, 102  
 gizmo transformacji, 60, 62  
 gładkie, Smooth, 121  
 głębokość, Depth, 166  
 głębokość fazowania, 168  
 graf animatora, 165  
 grupa docelowa, 26  
 grupowanie wierzchołków, 94  
 gry czasu rzeczywistego, 208

## H

hierarchia obiektów, 203  
 hierarchie, 194

**I**

IDE, Integrated Development Environment, 294  
 importowanie  
   elementów gry, 34  
   kluczy kształtów, 176  
   plików, 71  
   pliku, 54  
   zrigowanych postaci, 184  
 intensywność odbicia, Bounce Intensity, 261  
 interaktywne reorganizowanie siatki, 89  
 interfejs Blendera, 46

**J**

język Python, 48

**K**

kant, Crease, 219  
 kąt krawędzi, Edge Angle, 59  
 kinematyka  
   odwrotna, 181  
   prosta, 180  
 klasa  
   Enemy, 189  
   GUI, 197  
   PlayerPrefs, 237, 238  
   SaveState, 245–248  
   Wizard, 189, 194, 201  
 klatka  
   kluczowa, keyframe, 70, 150  
   końcowa, End Frame, 155  
   początkowa, Start Frame, 158  
   pośrednia, tween, 150  
 klocek podstawowy, 82, 83  
 klucze kształtu, Shape Keys, 173  
 kluczowanie  
   kształtów, 172  
   wizualne, Visual Keying, 171  
 kodowanie, 37  
 kolekcja prefabrykatów, 107  
 kolor światła firmamentu, 261  
 komponent  
   Box Collider, 190  
   klasa Enemy, 192  
   Light Probe Group, 275  
   Mesh Filter, 190  
   Mesh Renderer, 190  
   Nav Mesh Agent, 283, 285

  Transform, 190  
   Wizard, 192  
 komponowanie renderingów, 271  
 komunikat odrodzenia, 195  
 komunikaty, 191, 202  
 konfigurowanie  
   interfejsu Blendera, 46  
   klipu, 163  
 kontroler animatora, 164  
 kontrolki, 49  
 kontury, Wire, 224  
 kopia bezpieczeństwa, 83  
 kości, 177  
   deformowane, 182  
   sterujące, 182  
 kratka UV, 103  
 krawędzie, 58  
 krycie, Opacity, 114  
 krzywa  
   Beziera, 144  
   funkcyjna, F-curve, 150  
   zaniku, 122  
 kursor 3D, 80, 81

**L**

liczba ścianek w siatce, 131  
 lustro, Mirror, 94  
 lustrzane odbicie, 218

**Ł**

łączenie, Merge, 130

**M**

malowanie teksturą, Paint Texture, 115, 131,  
 134, 140, 142  
   3D View, 138  
   UV/Image Editor, 134  
 mapa  
   normalnych, 208  
   okluzji otoczenia, 270, 273  
   szachownicowa, 103  
   światła, 71, 257  
   kierunkowe, 260  
   podwójne, 259  
   pojedyncze, 259  
   światła dla siatki, 106  
   wysokości, 124

mapowanie  
 normalnych, 208  
 światła, Lightmapping, 71, 252, 255, 259  
 UV, 98

Maya, 49

metoda

edycji proporcjonalnej, 121, 123  
 kaskadowa, 24  
 modułowa, 36, 78, 79  
 pudełkowa, 209  
 rzeźbienia, 127  
 tekstury przemieszczeń, 124  
 zagęszczania siatki, 120

miękkie zaznaczanie, Soft Selection, 121

miksowanie kształtów, 172

mirroring, 93

mnożenie, Multiply, 272

model, 161

biznesowy, 26  
 szczegółowy, 209  
 uproszczony, 209

modelowanie

3D, 89  
 dróg, 144  
 pudełkowe, 205  
 terenu, 118–120  
 wysokorozdzielcze, 205

moduł, 77

modułowe środowiska, 77

modyfikator

Decimate, 130, 232, 233  
 Displace, 125, 126  
 Edge Split, 58, 59, 60  
 Mirror, 94, 213  
 Multiresolution, 120, 215–217  
 Shrinkwrap, 227–231  
 Subdivision Surface, 213, 215

motyw Elsyiun, 47

motywy, Themes, 46

## N

nakład pracy, 30

narzędzia

do retopologizacji modeli, 89  
 malarskie, 138  
 rzeźbiarskie, 127

narzędzie

Backface Culling, 85  
 Beast Lightmapper, 71

Extrude, 144

Knife Project, 145, 147

Select Faces by Sides, 91

Subdivide, 119, 120

Transform, 87

Triangulate Faces, 90

natężenie światła firmamentu, 261

nazywanie obiektów, 67

n-kąty, 88

normalna, Normal, 84, 139

NotificationsManager, 198–201

numeracja klatek animacyjnych, 158



obiekt nadrzędny, 197

obiekty

aktywne, 202  
 dynamiczne, 109  
 statyczne, 109

oblicz, Calculate, 57

obracanie obiektów, 64

obsługiwane rozdzielczości, 26

odbicia, Bounces, 260

odczyt z pliku, 246

odejmowanie, Subtract, 129

odległość, 32

odłączanie tekstury, 143

odrzućcie ścianek, 85

odstęp między klatkami, Frame Step, 171

odtwarzanie, Frame Rate, 151

odwracanie normalnych, 84

odzyskiwanie rezultatów sesji, 52

ograniczenia

Auto-Key, 154  
 mapy wysokości, 119

okluzja otoczenia, Ambient Occlusion, 260, 262, 269

okno

3D View, 138  
 Blender User Preferences, 50  
 Timeline, 155  
 UV/Image Editor, 134

okrąg, Circle, 228

opcja

Alpha, 268  
 Connected, 121  
 Extend, 92  
 Generic, 162

- Greater Than, 92
- Use Light Probes, 278
- opcje wypalania, 270
- opis szczegółów, 29
- opracowywanie modułów, 83
- optymalizacja tekstur, 79
- optymalność topologii, 118
- ostre krawędzie, Sharp Edges, 58, 59
- oś czasu, Timeline, 152
- oświetlenie
  - bezpośrednie, 260
  - dynamiczne, 252, 274
  - pośrednie, 260
  - statyczne, 252
- outsourcing, 34
- oznacz szew, Mark Seam, 99

## P

### parametr

- Brush Size, 114
- Edge Angle, 59
- Frame Step, 171
- Height, 119
- Heightmap Resolution, 114
- Length, 119
- Opacity, 114
- Preview, 216
- Radius, 129, 282
- Ratio, 232
- Render, 216
- Resolution, 257
- Scale, 69
- Scale Factor, 57, 73
- Sculpt, 216
- Smooth, 121
- Start Frame, 158
- Strength, 126
- Texture, 220
- View, 213
- Width, 119

### parametry

- komponentu Nav Mesh Agent, 283
- mapowania światła, 261, 264
- tekstury, 268
- terenu, 113
- wyświetlania siatki, 97
- pędzel, Brush, 114, 128, 219

- planowanie pracy, 33
- platformy, 26
- plik, 20

- Enemy.cs, 193
- EnemyAI.cs, 284
- Wizard.cs, 193
- z obrazem, 142

### pliki

- .blend, 54, 55
- binarne, 241
- FBX, 70
- JSON, 241
- skryptowe, 192
- XML, 240, 246

- płaszczyzna, Plane, 222

- pobieranie komponentu, 204

- podgląd, Preview, 124, 216

- podgląd awatara, 186

- podziel, Subdivide, 119, 214

### polecenie

- Add Texture, 116
- Add/Curve/Path, 166
- Add/Mesh/Plane, 119, 222
- Animate Path, 169
- Assign, 95
- Bake Action, 170
- Calculate, 57
- Clear Constraints, 171
- Draw All Edges, 224
- Edit Textures, 116
- Extrude, 144
- Flip Normals, 84
- Generate Colliders, 105
- Include Animation, 70
- Insert Single Keyframe, 153
- Join, 101
- Mark Seam, 99
- Optimize Keyframes, 70
- Origin to 3D Cursor, 80
- Recover Last Session, 52
- Remove Doubles, 92
- Select, 95
- Select All, 84
- Smart UV Project, 125
- Subdivide, 119, 120, 214
- Undo, 92
- Unlink datablock, 143
- Unwrap, 99
- User Preferences, 97, 137

popychanie, Nudge, 219  
 poszerz, Extend, 92  
 powierzchnia wielokrotnie dzielona, 206  
 prefabrykat, 107  
 preferencje
 

- gracza, 237, 238
- użytkownika, User Preferences, 97

 problem z cieniowaniem, 56  
 program Maya, 49  
 programowanie zdarzeniowe, 187  
 projekt wstępny, 27  
 projektowanie
 

- komponentowe, 190
- poziomów, 36

 promienie Final Gather, 262  
 promień, Radius, 129  
 prototyp, 29  
 próbnik światła, light probes, 253, 274  
 przekształcenie, Transform, 87, 190  
 przemieszczenie, Displace, 125  
 przenoszenie
 

- animacji, 158
- modeli, 53

 przesyłanie danych, 245  
 przezroczystość, 268  
 przyciąganie, snapping, 225
 

- do powierzchni, 221, 226

 przyrostowe, Incremental Snapping, 61, 81, 86  
 przypisywanie wierzchołków, 96  
 przypisz, Assign, 95  
 przyrost, Increment, 86  
 przywracanie poprzedniej sesji, 53  
 punkty kontrolne, 167  
 Python, 48

## R

raport operatorski, Dopesheet, 159  
 rejestr systemowy, 239  
 rejestracja klasy, 200  
 renderer siatki, Mesh Renderer, 190, 277  
 renderowanie, Render, 151, 216  
 renderowanie cieni, 56  
 rentgen, X-Ray, 224  
 retargeting, 185  
 retargetowanie animacji, 161  
 retopologizacja, 205, 208, 221, 227, 234  
 retopologizacja modeli, 89  
 rigi, 177

rozdzielczość
 

- mapy światła, 257, 258
- siatki, 212
- terenu, 130, 131

 rozmiar
 

- pędzla, Brush Size, 114
- wierzchołka, 97

 rozpraszanie światła, 272  
 rozsyłanie komunikatów, 195, 203  
 rozwiń, Unwrap, 99  
 rysuj wszystkie krawędzie, 224  
 rzeźbiące wyciąganie, SculptDraw, 128  
 rzeźbienie, Sculpt, 127, 216  
 rzeźbienie terenu, 114, 123  
 rzutowanie
 

- elementów, 225
- malowania, Project Paint, 139
- terenu, 126
- tnące, Knife Project, 145

## S

scena, 190  
 serializacja, 243, 244
 

- danych, 240
- klasy, 242

 siatka, Mesh, 67, 99
 

- high-poly, 206
- low-poly, 227
- nawigacyjna, 279, 280
- nawigacyjna, navmesh, 254
- niskorozdzielcza, 231
- statyczna, 77
- terenu, 127
- wielokątna, 88
- wysokorozdzielcza, 208

 silnik, 34  
 siła, Strength, 126  
 singletony, 201  
 składniki kolizyjne, 106  
 skrypt, 48, 73, 192  
 sterownik, Controller, 165  
 sterownik animatora, Animator Controller, 162, 164  
 stosowanie prefabrykatów, 107  
 stożek, Cone, 67  
 symetria, Symmetry, 218  
 system
 

- bytów, entity system, 191

Mecanim, 162  
 NotificationsManager, 198  
 powiadomień, 197  
 sterowania, 50  
 szczypanie, Pinch, 219  
 szerokość, Width, 119  
 sześcian, 80  
 szkielet, Rig, 161  
 szkielety symetryczne, 178  
 sztuczna inteligencja, 284  
 szwy, seams, 98

## Ś

ścieżka animacji, 167  
 środek, 32  
 obiektu, 60  
 transformacji, 146  
 światło otaczające, 256

## T

tablet graficzny, 217  
 tekstura, Texture, 98, 100, 115, 132, 220  
 mapy światła, 266  
 przemieszczenia, 124, 125  
 typu Clouds, 124, 141  
 teren, 111  
 generowanie, 112  
 malowanie teksturami, 115  
 modelowanie, 119, 120  
 parametry, 113  
 rozdzielczość, 130  
 rzeźbienie, 114, 123  
 rzutowanie, 126  
 teselacja, 119  
 testowanie, 39  
 topologia siatki, 118  
 triangulacja ścianek, Triangulate Faces, 89  
 trwałość danych, 235  
 tryb  
 Connected, 121  
 edycji, Edit Mode, 225  
 Increment, 86  
 mapowania światła, 259  
 mieszania, 272  
 rzeźbienia, 127  
 rzeźbienia, Sculpt Mode, 216  
 Single Lightmaps, 263

Subtract, 129  
 Texture Paint, 138  
 układania póź, 180  
 tryby mieszania, 135  
 tworzenie  
 animacji, 151  
 dróg, 143  
 elementów gry, 33  
 grupy, 96  
 kluczy kształtu, 173  
 krzywej, 167  
 krzywej Beziera, 145  
 modeli, 55  
 płaszczyzn, 223  
 prefabrykatu, 108  
 prototypu, 29  
 scen, 196  
 skryptu, 192  
 tekstur, 98, 133  
 terenu, 111  
 tytuł, 25

## U

układ Animation, 152  
 ukrywanie  
 deformowanych kości, 183  
 ścianek, 85  
 Unity, 35, 45  
 importowanie środowisk, 104  
 konfigurowanie środowisk, 104  
 tworzenie terenu, 111  
 ustalanie gęstości tekselowej, 102  
 ustawienia eksportu, 69  
 usuń więzy, Clear Constraints, 171  
 usuwanie, Delete, 130  
 usuwanie duplikatów, 92  
 używanie klocka podstawowego, 83

## W

warstwa okluzji otoczenia, 273  
 warstwy, Layers, 272  
 wczytywanie stanów gry, 248  
 węzeł BasicAnimation, 165  
 widoczność ścieżki, 168  
 widok, View, 213  
 widok z góry, 144  
 wielokrotność użycia, 79

- wielorozdzielczość, 120
  - wierzchołek, Vertex, 97
  - więcej niż, Greater Than, 92
  - więzy, Constraints, 167
    - Follow Path, 168
    - kości, Bone Constraints, 180
  - wizualizacja gry, 30
  - właściwości, Properties, 151
  - włączanie trybu
    - malowania, 135
    - rzeźbienia, 128
  - wsad statyczny, 109
  - współczynnik skali, Scale Factor, 73
  - współrzedne UV, 71
    - mapy światła, 105
    - terenu, 132
  - wstawianie klatek kluczowych, 153, 154
  - wstrzykiwanie zależności, DI, 190
  - wybieranie
    - danych trwałych, 239
    - systemu sterowania, 51
    - typu pędzla, 128
  - wycinanie, Cut, 130
  - wydajność, 79, 118
  - wymiary, Dimensions, 82, 155
  - wymiary klocka podstawowego, 82
  - wypalanie, baking, 208, 251
    - animacji, 166, 169, 171
    - map, 265
    - map światła, 263
    - nawigacji, 253, 278
    - oświetlenia dynamicznego, 252, 274
    - oświetlenia statycznego, 252
    - próbek, Bake Probes, 276
    - tekstur oświetleniowych, 267
    - widoczności obiektów, 285
  - wyrównanie wierzchołków ściany, 87
  - wysokość, Height, 119
  - wysyłanie komunikatów, 195–197
  - wyszukiwanie n-kątów, 91
  - wyświetlanie
    - informacji pythonowych, 49
    - siatki, 97
    - siatki nawigacyjnej, 280
  - wytłaczanie, Extrude, 92, 144, 225
  - wywołanie rysujące, 100
  - wyznaczanie szwów, 98
  - wzmocnienie odbicia, Bounce Boost, 261
- ## Z
- zagęszczanie siatki, 120, 215, 217
  - zalecenia, 41
  - zależności zaprogramowane, 188
  - załączniki, 29
  - zamykanie bez zapisu, 51
  - zapętlenie czasu, Loop Time, 162
  - zapis
    - do pliku, 245
    - mapy okluzji, 270
    - ręczny, 242
    - rozdzielczości, 237
    - stanu gry, 235, 240, 248
  - zarys gry, 28
  - zarządzanie projektami, 31, 33
  - zaznacz, Select, 95
  - zaznaczenie wierzchołka, 97
  - zderzacz
    - prostopadłościenny, Box Collider, 190
    - siatkowy, 105
  - zestaw środowiska modułowego, 104
  - zgłoszenia, 33
  - złącz, Join, 101
  - zmiana typu tekstury, 141



# PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



- 1. ZAREJESTRUJ SIĘ**
- 2. PREZENTUJ KSIĄZKI**
- 3. ZBIERAJ PROWIZJĘ**

Zmień swoją stronę WWW  
w działający bankomat!

**Dowiedz się więcej i dołącz już dzisiaj!**

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

## Zaprezentuj światu swoją wizję elektronicznej rozrywki!

Unity to darmowy silnik do tworzenia zaawansowanych gier komputerowych. Blender to darmowe środowisko do tworzenia grafiki 3D. Co może powstać z połączenia tych dwóch narzędzi? Genialne środowisko, dzięki któremu przygotujesz atrakcyjną wizualnie grę o niesamowitej grywalności! Z tego duetu gromadnie korzystają małe studia opracowujące gry — studia z małym budżetem, ale ze świetnymi pomysłami!

Jeżeli Twoim marzeniem jest wejście na rynek gier komputerowych, lecz nie dysponujesz budżetem, który pozwoliłby Ci zagrozić gigantom elektronicznej rozrywki, to Unity w połączeniu z Blenderem jest Twoją szansą na sukces. Sięgnij po tę książkę i przekonaj się, jak zorganizować proces tworzenia gry w 10 etapach. Z kolejnych rozdziałów nauczysz się, jak przenosić modele z Blendera do Unity, tworzyć tekstury i teren oraz animować obiekty. Na sam koniec dowiesz się, w jaki sposób oświetlać przedmioty i przechowywać stan gry. Znajdziesz tu też opisy różnych narzędzi, które mogą Ci pomóc w codziennej pracy. Ta książka to obowiązkowa lektura dla wszystkich marzycieli, którzy chcą tworzyć gry komputerowe. Dzięki niej marzenia staną się rzeczywistością!

### Sięgnij po tę książkę i:

- zintegruj Unity z Blenderem
- stwórz zaawansowany teren
- wykonaj atrakcyjne modele
- stwórz animację
- zbuduj grę komputerową w 10 sprawdzonych etapach

### Alan Thorn

— matematyk, twórca gier komputerowych. Założyciel niezależnego studia gier, zdobywca wielu nagród branżowych, autor licznych książek poświęconych tej tematyce. Prowadzi wykłady związane z tworzeniem gier.



**Helion**

31292 numer katalogowy

księgarnia internetowa

<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Sprawdź najnowsze promocje:  
● <http://helion.pl/promocje>  
Książki najchętniej czytane:  
● <http://helion.pl/bestsellery>  
Zamów informacje o nowościach:  
● <http://helion.pl/nowości>

Helion SA  
ul. Kościuszki 1c, 44-100 Gliwice  
tel.: 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
<http://helion.pl>

ISBN 978-83-283-0269-3



9 788328 302693

Informatyka w najlepszym wydaniu

cena: 54,90 zł