Spis treści (skrócony)

Spis treści (z prawdziwego zdarzenia)

Wprowadzenie

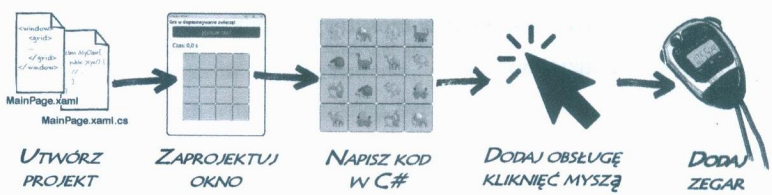
C# według Twojego mózgu. Siedzisz i próbujesz się czegoś *nauczyć*, ale *mózg* stale Ci powtarza, że cała ta nauka *nie jest ważna*. Twój umysł mówi: „Lepiej wyjdź z pokoju i zajmij się ważniejszymi sprawami — dowiedz się, których dzikich zwierząt unikać, a także sprawdź, czy jazda na snowboardzie nago to dobry pomysł". W jaki sposób *możesz* oszukać mózg, aby uznał, że Twoje życie naprawdę zależy od nauki C#?

Zacznij pisać programy w języku C#

1 Zbuduj coś wspaniałego... szybko

Chcesz tworzyć fantastyczne aplikacje... od razu?

C# to nowoczesny język programowania i **cenne narzędzie**, które masz na wyciągnięcie ręki. Ponadto w postaci **Visual Studio** otrzymujesz fantastyczne środowisko programistyczne z wysoce intuicyjnymi funkcjami, które sprawiają, że pisanie kodu jest niezwykle łatwe. Visual Studio jest nie tylko znakomitym środowiskiem do pisania kodu, ale też **bardzo przydatnym narzędziem do nauki** i eksplorowania języka C#. Brzmi nieźle? ***Pora zabrać się za programowanie!***

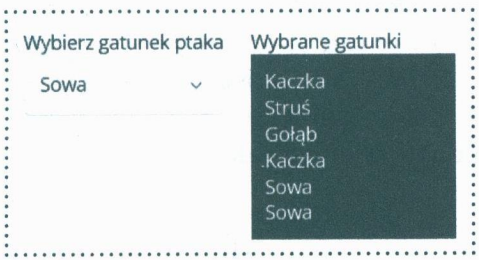


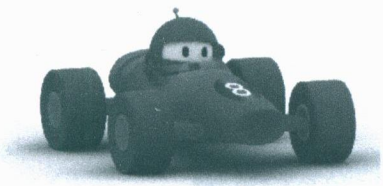
Zmienne, instrukcje i metody

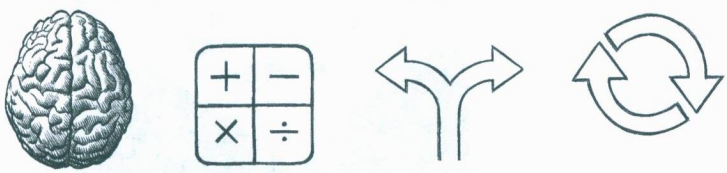
2 Zanurz się w kod C#

Nie jesteś jedynie użytkownikiem IDE. Jesteś programistą.

Za pomocą IDE możesz wykonać wiele zadań, ale nawet to środowisko nie zrobi wszystkiego za Ciebie. Visual Studio jest jednym z najbardziej zaawansowanych narzędzi programistycznych, jakie kiedykolwiek stworzono, ale **rozbudowane IDE** to tylko punkt wyjścia. Pora ***zagłębić się w kod C#*** i dowiedzieć się, jaką ma budowę, jak działa i jak można przejąć nad nim kontrolę. Nie ma ograniczeń w tym, co mogą robić Twoje aplikacje.



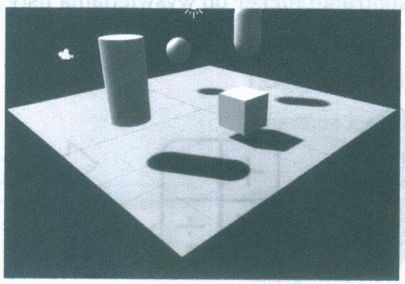


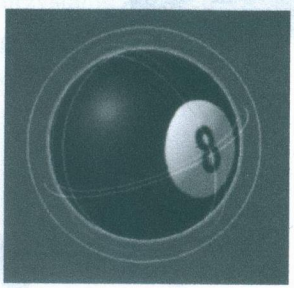


Ćwiczenia z Unity nr 1

Poznawaj C# z Unity

Witaj w pierwszych **ćwiczeniach z Unity** w ***C#. Rusz głową***.
Pisanie kodu jest umiejętnością i, podobnie jak inne umiejętności, wymaga **praktyki i eksperymentowania**, jeśli chcesz robić postępy. Unity jest wartościowym narzędziem, które Ci w tym pomoże. W tych ćwiczeniach zaczniesz stosować w praktyce wiedzę, jaką zdobyłeś w rozdziałach 1. i 2.



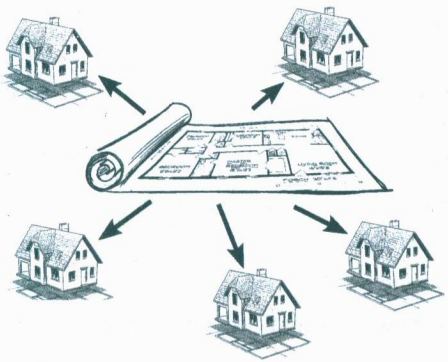


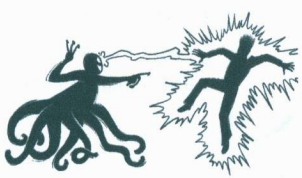
Przestrzenie nazw i klasy

3 Organizowanie kodu

Świetni programiści dbają o organizację kodu i danych.

Jaką pierwszą rzecz robisz podczas tworzenia aplikacji? Myślisz o tym, **co ma robić**, niezależnie od tego, czy rozwiązujesz problem, tworzysz grę, czy po prostu się bawisz. Ale nie zawsze jest oczywiste, jak poszczególne instrukcje wpasowują się w ogólny obraz aplikacji. I właśnie w tym miejscu pojawiają się **klasy**. Pozwalają one **organizować kod** na podstawie tworzonych funkcji i rozwiązywanych przez aplikację problemów. Klasy mogą również pomóc w **organizowaniu danych**, ponieważ umożliwiają tworzenie **obiektów** reprezentujących dowolne „rzeczy" potrzebne w aplikacji. Projektowane klasy stanowią „wzorce" dla obiektów używanych w aplikacji.



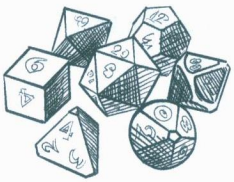


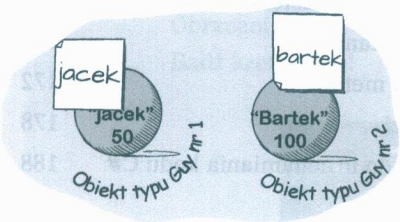
Dane, typy, obiekty i referencje

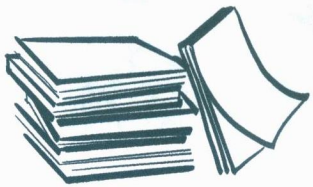
4 Zarządzanie danymi aplikacji

Dane i obiekty to cegiełki aplikacji.

Czym byłyby aplikacje bez danych? Zastanów się nad tym przez chwilę. Bez danych programy są... no cóż, trudno nawet wyobrazić sobie pisanie kodu bez danych. Potrzebujesz **informacji** od użytkownika i korzystasz z nich do generowania nowych informacji do zwrócenia. Prawie wszystko, co robisz w trakcie programowania, w jakiś sposób dotyczy **pracy z danymi**. W tym rozdziale poznasz tajniki **typów danych** i **referencji**, zobaczysz, jak pracować z danymi w programach, a także dowiesz się kilku nowych rzeczy na temat **obiektów** (i wiesz co? — obiekty także są danymi!).



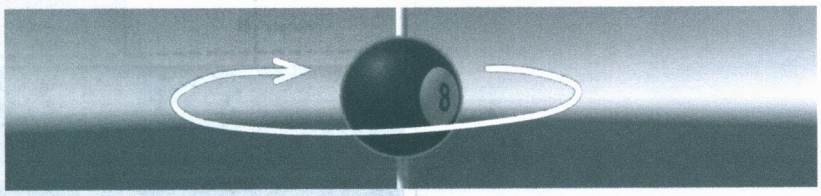




Ćwiczenia z Unity nr 2

Pisanie kodu C# w Unity

Unity nie jest *tylko* rozbudowanym, międzyplatformowym silnikiem i edytorem do budowania gier 2D i 3D oraz symulacji. Jest też **doskonałym narzędziem do nabierania praktyki w pisaniu kodu C#**. W tych ćwiczeniach nabierzesz wprawy w pisaniu kodu C# w projekcie w Unity.



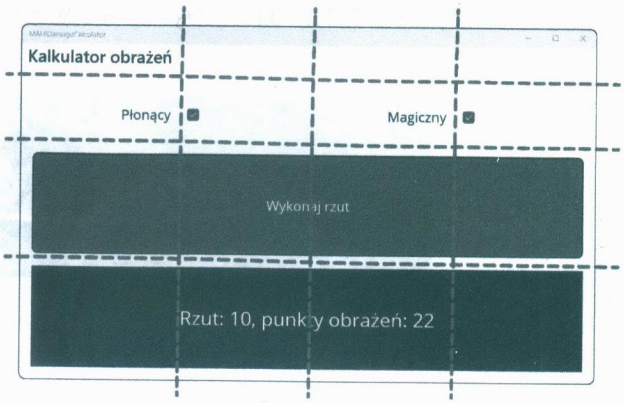
Hermetyzacja

5 O tym, jak obiekty strzegą swoich tajemnic

Czy marzyłeś kiedyś o odrobinie prywatności?

Obiekty czasem czują się podobnie. Tak jak Ty nie chcesz, aby osoby, którym nie ufasz, czytały Twój dziennik lub przeglądały wyciągi bankowe, dobre obiekty nie pozwalają ***innym*** obiektom podglądać swoich pól. W tym rozdziale poznasz wartość **hermetyzacji**. Jest to mechanizm programistyczny pomagający pisać kod łatwy do modyfikowania i stosowania oraz odporny na próby niewłaściwego użycia. **Zapewnisz prywatność danym obiektów** oraz dodasz **właściwości** zabezpieczające dostęp do tych danych. Ponadto zabezpieczysz ważne dane obiektu przed **wyciekaniem do innych obiektów**, aby nie zostały przypadkowo niewłaściwie wykorzystane.

- *ABY OBLICZYĆ PUNKTY OBRAŻEŃ (PO) PO ATAKU MIECZEM, WYKONAJ RZUT 3D6 (TRZEMA KOSTKAMI SZEŚCIOŚCIENNYMI) I DODAJ „OBRAŻENIA BAZOWE" RÓWNE 3PO.*
- *PŁONĄCE MIECZE POWODUJĄ DODATKOWE 2PO.*
- *JEŚLI MIECZ JEST MAGICZNY, RZUT 3D6 JEST MNOŻONY PRZEZ 1,75, A WYNIK NALEŻY ZAOKRĄGLIĆ W DÓŁ. DO WYNIKU NALEŻY DODAĆ OBRAŻENIA BAZOWE I OBRAŻENIA OD OGNIA.*

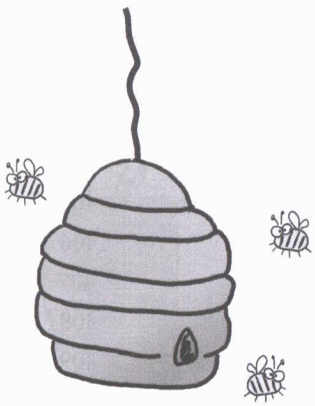


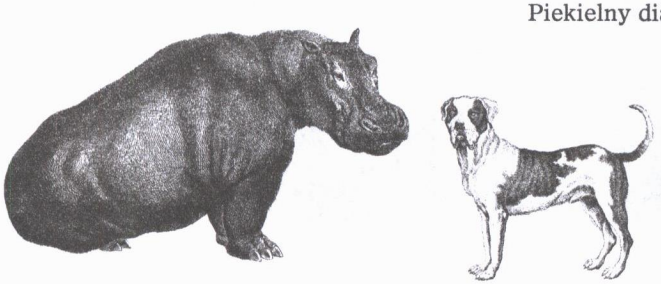
Dziedziczenie

6 Drzewo genealogiczne obiektów

Czasem CHCESZ być taki jak Twoi rodzice.

Czy natrafiłeś kiedyś na klasę, która robi ***prawie*** dokładnie to, co ma robić ***Twoja*** klasa? Czy naszła Cię myśl, że gdybyś tylko mógł ***zmienić kilka rzeczy***, klasa byłaby idealna? Dzięki **dziedziczeniu** możesz **rozszerzać** istniejące klasy. W ten sposób nowa klasa otrzyma wszystkie operacje starej, zachowując **swobodę** wprowadzania zmian, a Ty będziesz mógł dostosować nową klasę do swoich potrzeb. Dziedziczenie jest jedną z najważniejszych koncepcji i technik w języku C#. Dzięki niemu możesz **uniknąć powtórzeń w kodzie**, uzyskać dokładniejszy **model rzeczywistego świata** i budować aplikacje, które są **łatwiejsze w konserwacji** i **mniej narażone na błędy**.

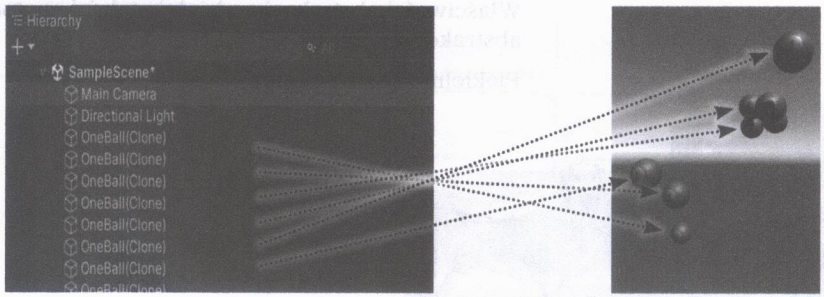




Ćwiczenia z Unity nr 3

Instancje obiektów gry

C# jest językiem obiektowym, a ponieważ celem ćwiczeń z Unity w książce *C#. Rusz głową* jest **praktykowanie pisania kodu w C#**, zrozumiałe jest, że w tych ćwiczeniach skupisz się na tworzeniu obiektów.

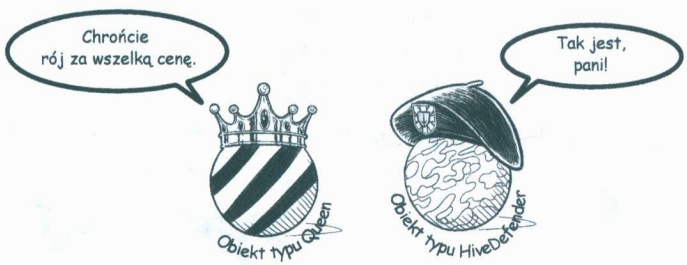


Interfejsy, rzutowanie i instrukcja „is"

7 Spełnianie obietnic przez klasy

Chcesz, aby obiekt wykonywał określone zadanie? Zastosuj interfejs.

Czasem programista chce pogrupować obiekty na podstawie tego, co potrafią robić, a nie na bazie klas, po jakich dziedziczą. Interfejsy to umożliwiają. Możesz użyć interfejsu do zdefiniowania **określonego zadania**. Każda instancja klasy **implementującej** dany interfejs *gwarantuje wykonanie tego zadania* niezależnie od tego, po jakich klasach dziedziczy. Aby ten mechanizm działał, każda klasa implementująca interfejs musi zobowiązać się do spełnienia wszystkich obietnic — w przeciwnym razie kompilator połamie jej kolana, rozumiesz?



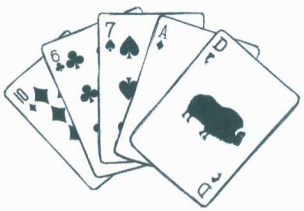
Wyliczenia i kolekcje

8 Porządkowanie danych

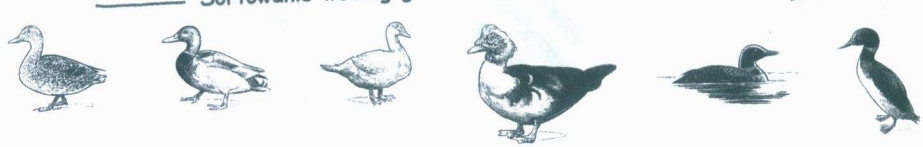
Dane nie zawsze są tak uporządkowane, jak byś sobie tego życzył.

W praktyce nie otrzymujesz danych w postaci uporządkowanych porcji i elementów. Nic z tego — dane będą trafiać do Ciebie w formie rozmaitych ładunków, stert i wiązek. Będziesz potrzebować rozbudowanych narzędzi, aby je wszystkie uporządkować. Na szczęście C# udostępnia wszystko, czego potrzebujesz. **Wyliczenia** to typy, które pozwalają zdefiniować poprawne wartości do kategoryzowania danych. **Kolekcje** są specjalnymi obiektami przechowującymi wiele wartości. Pozwalają **przechowywać i sortować** wszelkie dane, jakie program musi przetwarzać, oraz **zarządzać** nimi. Dzięki temu możesz przeznaczyć czas na zastanowienie się nad programem pracującym z danymi, a obsługą ich samych zajmą się kolekcje.

Rzadko używana karta „książę bawołów".

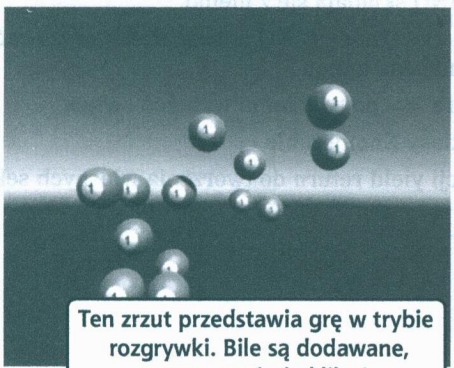


Sortowanie według gatunków kaczek.

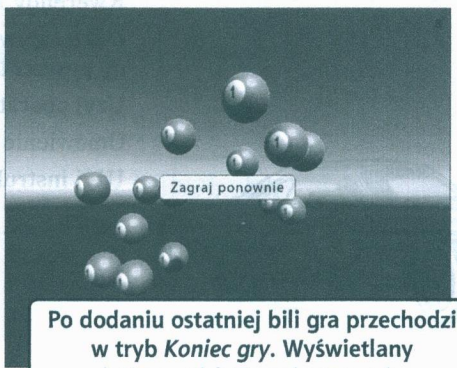


Ćwiczenia z Unity nr 4
Interfejsy użytkownika

W poprzednich ćwiczeniach z Unity zacząłeś pisać grę, używając obiektów prefab do tworzenia obiektów gry, które pojawiają się w losowych punktach przestrzeni gry trójwymiarowej i latają po ekranie. W tych ćwiczeniach z Unity będziesz kontynuować te prace, co pozwoli Ci wykorzystać wiedzę na temat interfejsów z C# i inne informacje.



Ten zrzut przedstawia grę w trybie rozgrywki. Bile są dodawane, a gracz może je klikać, aby poprawić wynik.



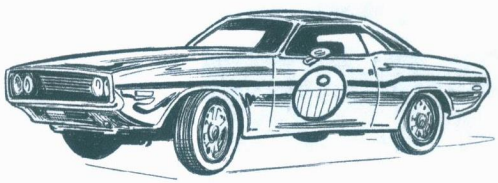
Po dodaniu ostatniej bili gra przechodzi w tryb *Koniec gry*. Wyświetlany jest przycisk *Zagraj ponownie*, a gra nie dodaje nowych bil.

LINQ i lambdy

9 Kontroluj swoje dane

Świat jest sterowany przez dane. Wszyscy musimy nauczyć się w nim żyć.

Minęły już dni, kiedy można było dniami, a nawet tygodniami programować bez przetwarzania **dużych ilości danych**. Dziś ***wszystko kręci się wokół danych***. Tu właśnie przydaje się technologia **LINQ**. Jest to mechanizm języka C# i platformy .NET, który nie tylko pozwala w intuicyjny sposób **pisać kwerendy danych** z kolekcji platformy .NET, ale też **grupować dane** i **scalać dane z różnych źródeł**. Zobaczysz, jak używać **obiektów anonimowych** do zarządzania danymi w nowe i interesujące sposoby. W tym rozdziale dodasz **testy jednostkowe**, aby mieć pewność, że kod działa w oczekiwany sposób. Gdy już nauczysz się przekształcać dane na możliwe do przetworzenia porcje, zastosujesz **wyrażenia lambda**, aby zrefaktoryzować kod C#, by był jeszcze bardziej zwięzły.

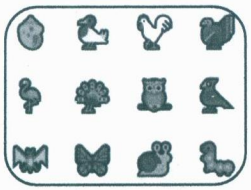


Odczyt i zapis plików

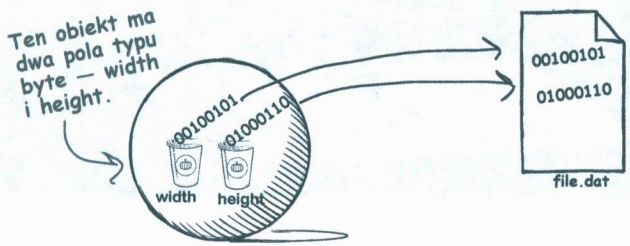
10 Zachowaj dla mnie ostatni bajt

Czasem trwałość się przydaje.

Do tej pory wszystkie omawiane programy miały krótki czas życia. Były uruchamiane, działały przez pewien czas, po czym kończyły pracę. Jednak ten model nie zawsze wystarcza — zwłaszcza gdy przetwarzasz ważne informacje. Potrzebna jest możliwość **zapisywania pracy**. W tym rozdziale zobaczysz, jak **zapisywać dane w pliku**, a także jak **wczytywać informacje** z pliku. Poznasz też **strumienie**, dowiesz się, jak zapisywać obiekty w plikach z użyciem **serializacji**, i przyjrzysz się bitom oraz bajtom w **danych szesnastkowych, Unicode i dwójkowych**.

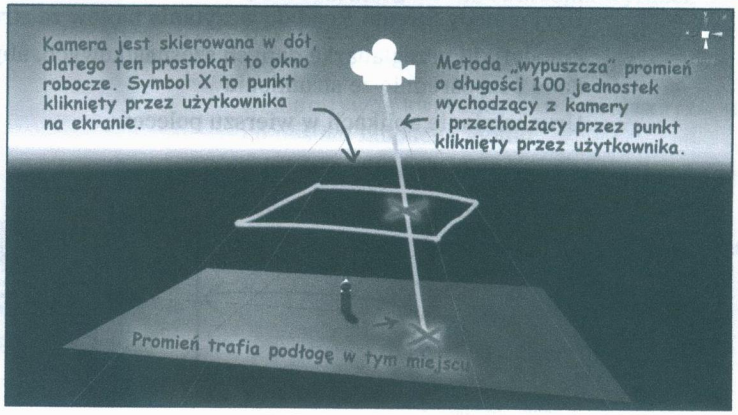


```
0000: 54 6f 20 65 6c 65 6d 65 To eleme
0005: 6e 74 61 72 6e 65 2c 20 ntarne,
0010: 64 72 6f 67 69 20 57 61 drogi Wa
0015: 74 73 6f 6e 69 65       tsonie
```

Ćwiczenia z Unity nr 5
Ray casting

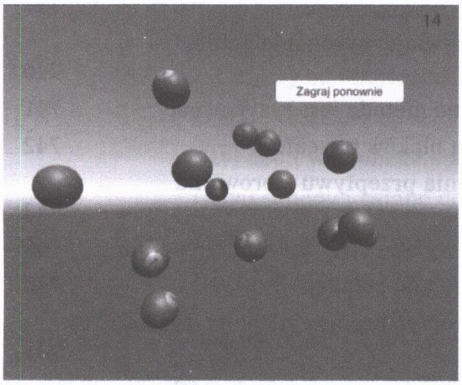
Gdy konfigurujesz scenę w Unity, tworzysz wirtualny trójwymiarowy świat, w którym postacie z gry mogą się poruszać. Jednak w większości gier wiele wydarzeń nie jest bezpośrednio kontrolowanych przez gracza. W jaki więc sposób obiekty poruszają się po scenie? W tych ćwiczeniach zobaczysz, jak C# może w tym pomóc.



KAPITAN WSPANIAŁY

ŚMIERĆ OBIEKTU

C#. Rusz głową	
4 zł	Rozdział 11.



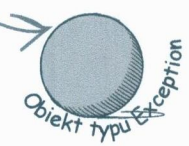
Obsługa wyjątków

12 Gaszenie pożarów robi się nudne

Gdy musisz rozwiązywać błąd za błędem, nazywamy to „gaszeniem pożarów".

Wyobraź sobie, że minęło kilka lat. Spędziłeś cały ten czas na rozwoju umiejętności z zakresu C#. Kontynuujesz naukę i doskonalisz się, a teraz jesteś jednym ze starszych programistów w dużej firmie technologicznej. Ale spanikowani współpracownicy nadal dzwonią do Ciebie w środku nocy, ponieważ **Twój program się zawiesza** lub **nie działa tak, jak powinien**. Chcesz przeznaczać czas na pisanie kodu, a nie na gaszenie pożarów! Nic nie wybija programistów z rytmu tak jak konieczność wyeliminowania dziwnego błędu. Na szczęście C# udostępnia **obsługę wyjątków**, dzięki czemu możesz pisać kod **radzący sobie z problemami**, które się pojawiają. Jeszcze lepsze jest to, że możesz nawet zaplanować działania na wypadek takich problemów, aby program **kontynuował działanie** po ich wystąpieniu.


```
int[] anArray = {3, 4, 1, 11};
int aValue = anArray[15];
```



Ćwiczenia z Unity nr 6

Nawigowanie po scenie

W poprzednich ćwiczeniach z Unity utworzyłeś scenę z podłogą (płaszczyzną) i gracza (kulę powiązaną z walcem). Ponadto użyłeś siatki nawigacyjnej, obiektu *NavMesh Agent* i *ray castingu*, aby gracz podążał w miejsca kliknięcia myszą na scenie. W tych ćwiczeniach dodasz elementy do sceny, używając C#.

Komponent NavMesh Obstacle tworzy poruszającą się lukę w siatce nawigacyjnej, co czasowo uniemożliwia graczowi wejście po rampie. Dodasz skrypt, który umożliwi użytkownikowi przesuwanie tej przeszkody w górę i w dół, aby zablokować lub odblokować rampę.