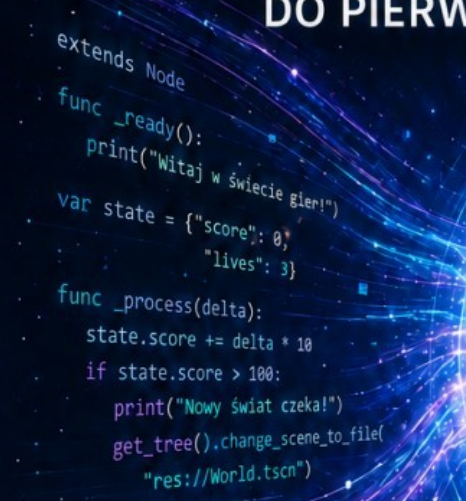
GODOT ENGINE I GDSCRIPT

poswojsku

CZĘŚĆ 2

OD PUSTEGO PROJEKTU DO PIERWSZEGO PROGRAMU



DARIUSZ GOŁĘBIOWSKI

GODOT ENGINE I GDSCRIPT POSWOJSKU Część 2: Od pustego projektu do pierwszego programu

PORADNIK POWSTAŁ NA BAZIE
DOŚWIADCZENIA OSOBISTEGO AUTORA
Z PRZYJACIELSKIM WSPARCIEM AI

NOTA WYDAWCY

UWAGA!

PORADNIK JEST KONTYNUACJĄ CZĘŚCI 1: Wprowadzenie do świata gier i pierwsze skrypty. Dla lepszego zrozumienia zawartości tej książki, Autor i Wydawnictwo zalecają rozpocząć naukę GODOT Engine od Części 1.

OD AUTORA:

Pamiętaj proszę- uszanuj zaangażowanie oraz godziny pracy, które spędziłem nad napisaniem oraz opracowaniem książki:

GODOT ENGINE I GDSCRIPT POSWOJSKU Część 2: Od pustego projektu do pierwszego programu - używaj tylko leganie kupiony ebook.

Wydawnictwo poswojsku.pl – kontakt:

Strona firmowa: www.poswojsku.pl

e-mail: bok@poswojsku.pl

ul. Paprocka 86, 98–220 Zduńska Wola

ISBN: 978-83-68360-50-9

Autor: Gołębiowski Dariusz

W publikacji wykorzystano fonty:

- **Lato udostępniany na licencji SIL Open Font License,**
- **Roboto, udostępniane na licencji Apache License, Version 2.0.**
- **Zrzuty ekranowe i grafiki z gry „Escape from the Seventh Planet", której bezpłatną wersję (5 poziomów) możesz pobrać na stronie wydawnictwa www.poswojsku.pl oraz na stronie projektu www.poswojsku.pl .**

Informacja o wykorzystaniu sztucznej inteligencji

Podczas przygotowywania publikacji wykorzystano narzędzia AI jako wsparcie w opracowywaniu koncepcji, porządkowaniu materiału, redagowaniu wybranych fragmentów oraz przygotowywaniu części elementów graficznych. Całość została zweryfikowana, uzupełniona i zatwierdzona przez autora, który ostatecznie wykonał: dobór, układ i treść publikacji.

Wybrane ilustracje mają charakter syntetyczny i zostały przygotowane lub zmodyfikowane z wykorzystaniem narzędzi generatywnej sztucznej inteligencji.

Od Wydawnictwa Cyfrowego poswojsku.pl

KILKA SŁÓW O FORMACH ŻEŃSKICH I MĘSKICH

Wspaniała Czytelniczko - Drogi Czytelniku

Jak nowoczesnemu wydawnictwu, zależy nam, aby nasz publikacje były przyjazne dla wszystkich osób. Jednocześnie staramy się, aby nasi autorzy pisali w sposób prosty, naturalny i wygodny w czytaniu.

Dlatego w książce czasami używane są formy męskie, a czasami żeńskie. Wszystkie odnoszą się do każdej osoby czytającej tę publikację, niezależnie od płci, wieku, pochodzenia, stanowiska czy sposobu, w jaki sama siebie określa.

Nie stosujemy konsekwentnie zapisów typu „czytelnik/ czytelniczka”, „nauczyciel/ nauczycielka” itp., ponieważ utrudniają one płynność czytania.

Jeżeli ktoś zauważy, że

- liczba końcówek męskich nie zgadza się z liczbą końcówek żeńskich,
- istnieją inne końcówki, które nie zostały użyte,

z góry przepraszamy za nas i naszych Autorów. Nie jest to działanie zamierzone – po prostu poswojsku jest wydawcą książek ;), a nie firmą księgową, dbającą o zgodność liczby końcówek językowych.

Wszystkim

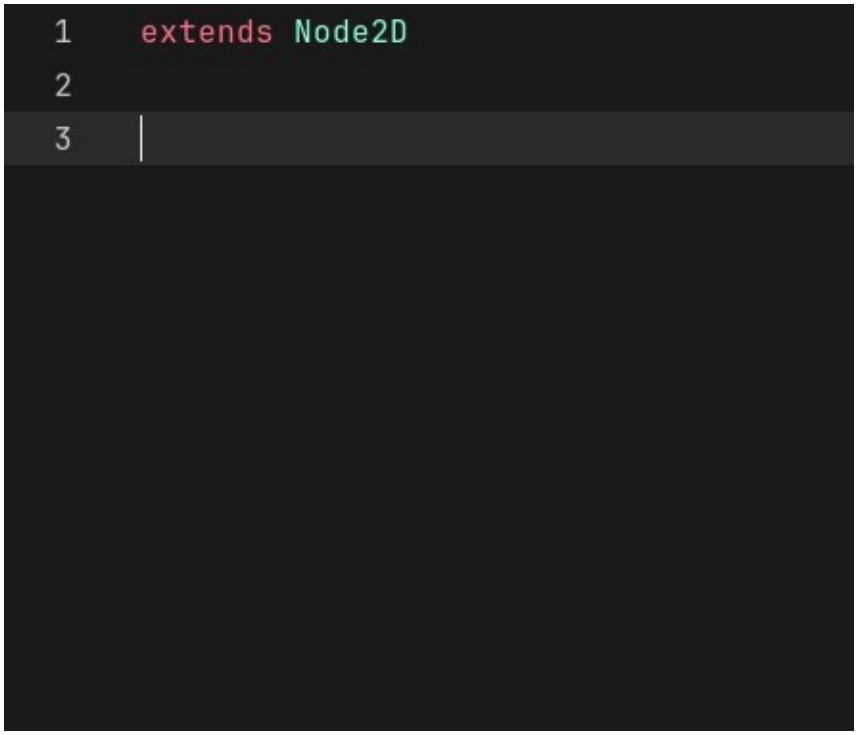
Wspaniałym Czytelniczkom oraz

Drogim Czytelnikom

życzymy przyjemnej lektury tej książki.

Spis treści

Wprowadzenie



Pusty projekt ma w sobie coś tajemniczego ;).

Na ekranie nie ma jeszcze bohaterki, przeciwników ani nawet punktów do zdobycia. Nie istnieją zasady, zwycięstwa ani porażki. Nic nie reaguje na Twoje polecenia. Jest jedynie przestrzeń czekająca na pierwszą decyzję twórcy. To brzmi dumnie: TWÓRCA, czyli TY :).

Doskonale znam to uczucie. Kiedy zaczynałem kodować *Escape From The Seventh Planet* – moją autorską grę osadzoną w uniwersum Sagi CyberJestestwa – wpatrywałem się dokładnie w taką samą, bezkresną pustkę Godota. A potem pojawił się kod.

Jakieś zmienne, warunki, funkcje i sygnały. Z każdą kolejną instrukcją projekt zaczynał ożywać. Najpierw nieśmiało reagował na kliknięcie. Później odliczał czas, poruszał postacią, przyznawał punkty i rozpoznawał zdarzenia. Nawet jeśli miał błędy, przestawał być pustą planszą. Stawał się prawdziwym programem. I właśnie temu procesowi ożywiania poświęcona jest druga część książki „Godot Engine i GDScript poswojsku”.

Do każdej książki potrzebny jest bohater. Tutaj będzie nim statek do podróży kosmicznych wykorzystujący Portale Wodne: ***Pokorny Dzwon*** :). Wziąłem go z mojej powieści. A oto jego dwie wersje:

Pokorny Dzwon bez osłon ochronnych

Pokorny Dzwon z aktywnymi osłonami

Od poznawania kodu do tworzenia zasad

W pierwszej części zbudowaliśmy fundament: poznaliśmy środowisko, sceny, węzły i skrypty. Teraz pora na kolejny krok. Nie będziemy już tylko wydawać programowi pojedynczych poleceń. Nauczymy go wybierać właściwą drogę, powtarzać działania, reagować na zdarzenia i przekazywać informacje. To moment, w którym programowanie staje się budowaniem własnego świata, a każdy warunek staje się jego żelaznym prawem:

- **jeżeli** gracz dotknie statku — zrób coś (np. przyznaj punkty);
- **jeżeli** czas dobiegnie końca — zrób coś (np. zakończ rozgrywkę).

Pętle pozwolą nam zautomatyzować żmudne działania. Funkcje uporządkują kod, a sygnały umożliwią obiektom komunikowanie się bez tworzenia chaosu. To już nie będzie zbiór przypadkowych instrukcji. To będzie system. Twój system!

Kod ma być zrozumiały

Programowanie bywa niesłusznie przedstawiane jako wiedza tajemna dla garstki wybrańców. Nie zgadzam się z tym. Kod to po prostu sposób opisywania zasad. Zamiast uczyć się GDScriptu poprzez bezrefleksyjne przepisywanie linijek tekstu, każde zagadnienie rozłożymy na mniejsze części. Nie chodzi o to, aby wkuć każdą instrukcję na pamięć. Chodzi o to, aby potrafić zadać właściwe pytania:

- Co powinno wydarzyć się w programie?
- Jaki warunek musi zostać spełniony?
- Które działanie należy powtórzyć?
- Który obiekt powinien poinformować resztę o zdarzeniu?
- Dlaczego kod zachowuje się inaczej, niż zaplanowaliśmy?

Ta umiejętność jest znacznie cenniejsza niż znajomość dziesiątek poleceń.

Od warunku do działającej mini-gry

Zaczniemy od instrukcji warunkowych, dzięki którym gra zacznie podejmować decyzje. Zrozumiemy potęgę pętli, ale też pułapki z nimi związane (jedna źle napisana pętla potrafi przecież „zawiesić” cały system).

Wyjaśnimy magię parametru `delta` i to, dlaczego ruch w grze musi być niezależny od liczby klatek wyświetlanych przez monitor.

Zdobytą wiedzę wykorzystamy w praktyce, tworząc mini-grę „Klikaj, zanim czas minie!”. Nie będzie to suchy, oderwany od rzeczywistości przykład, ani też żadne zaawansowane dzieło kodowania :). Ale od czegoś trzeba zacząć.

Zbudujemy działający projekt z dobrymi zasadami kodowania, upływającym czasem i mechaniką, którą będziesz mógł samodzielnie rozwijać. Z każdym nowo stworzonym sygnałem i funkcją pusty projekt zacznie żyć.

Błędy są częścią tej podróży

Podczas pracy coś na pewno nie zadziała. Sygnał nie zostanie połączony, zrobisz literówkę w zmiennej, licznik pokaże wartość ujemną, a statek kosmiczny naliczy punkty podwójnie. To absolutnie normalne.

Błąd nie oznacza, że nie nadajesz się do programowania. Oznacza jedynie, że komputer wykonał instrukcję dokładnie tak, jak mu kazano – choć inaczej, niż tego chciałeś. Nauczysz się tu nie tylko pisać kod, ale też go diagnozować. Programista to nie ktoś, kto nigdy się nie myli. To ktoś, kto potrafi znaleźć drogę od błędu do rozwiązania.

Nie bój się eksperymentować

Książkowe przykłady to punkt wyjścia, a nie meta. Zmieniaj wartości. Przestawiaj obiekty. Psuj i naprawiaj. Jeśli projekt przestanie działać – ustal dlaczego. A jeśli zadziała inaczej, niż planowałeś – zastanów się, czy przypadkiem nie odkryłeś nowej,

genialnej mechaniki. Prawdziwe zrozumienie programowania zdobywa się w działaniu.

Zaczynamy

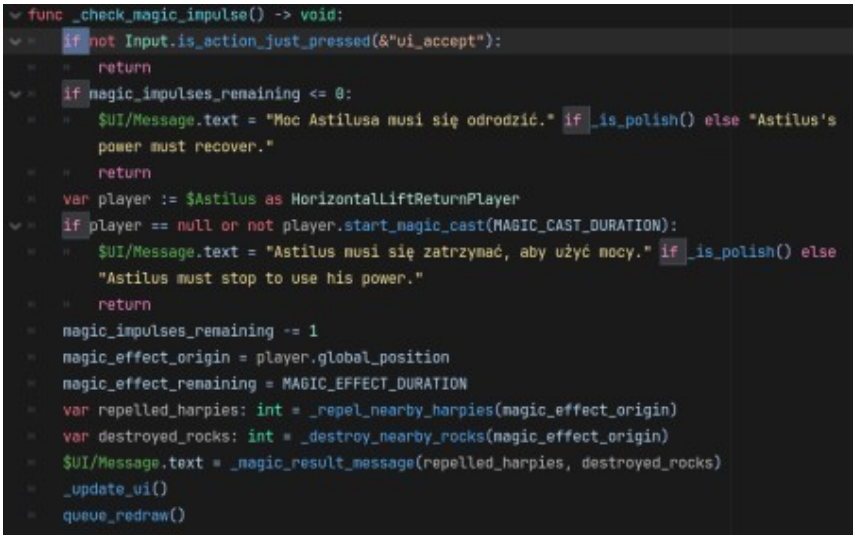
Przed nami warunki, pętle, funkcje i sygnały. Nie musisz wiedzieć wszystkiego od razu. Wystarczy, że wykonasz ten jeden, kolejny krok. Otwórz Godot Engine. Utwórz projekt. Dodaj pierwszą scenę. Pusty świat już na Ciebie czeka.

Nadajmy mu zasady. Wprawmy go w ruch.

Napiszmy pierwszy prawdziwy program.

Praktycznie. Zrozumiale. poswojsku.

Moduł 3 – Instrukcje warunkowe i pętle w akcji



```
func _check_magic_impulse() -> void:
    if not Input.is_action_just_pressed(&"ui_accept"):
        return
    if magic_impulses_remaining <= 0:
        $UI/Message.text = "Moc Astilusa musi się odrodzić." if _is_polish() else "Astilus's
        power must recover."
        return
    var player := $Astilus as HorizontalLiftReturnPlayer
    if player == null or not player.start_magic_cast(MAGIC_CAST_DURATION):
        $UI/Message.text = "Astilus musi się zatrzymać, aby użyć mocy." if _is_polish() else
        "Astilus must stop to use his power."
        return
    magic_impulses_remaining -= 1
    magic_effect_origin = player.global_position
    magic_effect_remaining = MAGIC_EFFECT_DURATION
    var repelled_harpies: int = _repel_nearby_harpies(magic_effect_origin)
    var destroyed_rocks: int = _destroy_nearby_rocks(magic_effect_origin)
    $UI/Message.text = _magic_result_message(repelled_harpies, destroyed_rocks)
    _update_ui()
    queue_redraw()
```

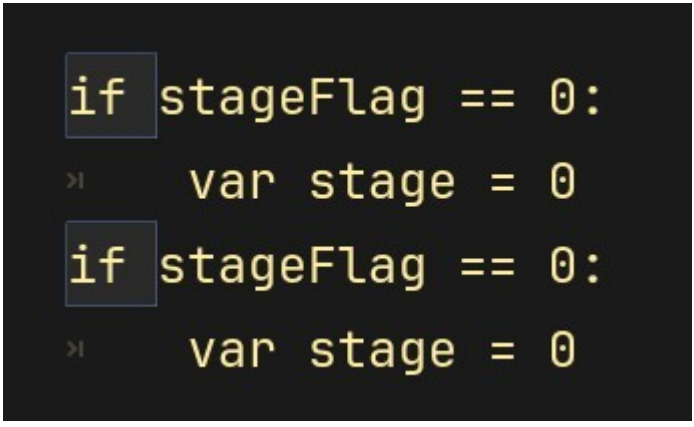
Cele modułu:

- Nauczyć się kontrolować przebieg programu,
- Podejmować decyzje,
- Powtarzać działania,
- Reagować na upływ czasu,
- Sterować stanem prostej gry.

Po ukończeniu tego modułu będziesz potrafić:

- budować warunki za pomocą `if`, `elif` i `else`;
- łączyć warunki operatorami logicznymi;
- korzystać z pętli `for` i `while`;
- przerywać pętlę albo pomijać jej wybrane wykonania;
- odróżniać pętlę od funkcji wywoływanej w każdej klatce;
- rozumieć podstawową rolę parametru `delta`;
- sterować prostym ruchem postaci;
- przygotować licznik czasu;
- wykorzystać instrukcję `match`;
- zbudować mini-grę opartą na czasie i punktach.

1. Logika programu: „jeżeli wydarzy się to, wykonaj tamto”



Silnik Godot Engine wykonuje instrukcje zgodnie z przygotowanymi (w kodzie) przez programistkę zasadami.

W grach i aplikacjach wiele takich zasad ma postać warunków:

- jeżeli gracz ma właściwy klucz, otwórz określone drzwi;
- jeżeli punkty życia spadną do zera, zakończ grę;
- jeżeli czas się skończy, zablokuj możliwość zdobywania punktów;
- jeżeli użytkownik poda poprawne hasło, pozwól mu przejść dalej;
- jeżeli przeciwnik zobaczy gracza, rozpocznij pościg;
- jeżeli postać stoi na podłożu, pozwól jej skoczyć.

Warunek odpowiada na pytanie, którego wynikiem jest wartość logiczna:

true

albo:

false

Przykładowo:

health <= 0

oznacza pytanie:

Czy wartość zmiennej `health` jest mniejsza lub równa zeru?

Jeżeli odpowiedź brzmi „tak”,

wynikiem warunku jest `true`.

2. Instrukcja `if`

Najprostsza instrukcja warunkowa wygląda następująco:

```
if warunek:
    instrukcja
```

Czyli jeżeli ***warunek*** da logiczną wartość *TRUE*, wówczas wykonywana jest ***instrukcja***

Przykład:

```
func check_health(health: int) -> void:
    if health <= 0:
        print("Koniec gry!")
```

Komunikat zostanie wyświetlony tylko wtedy, gdy wartość `health` będzie mniejsza lub równa zeru.

Kod należący do warunku musi być prawidłowo wcięty, czyli ustawiony od lewej krawędzi, a pozwól iż przypomnę, że 4 spacje w przypadku takich języków kodowania jak Python czy używany w Godot Engine GDScript, niekoniecznie równy jest spacji. Warto o tym pamiętać, bo inaczej możesz się spotkać z sytuacją, w której wizualnie – wszystko wygląda o.k., ale program „wyrzuca" błąd. Poniżej – zasy wcięcia kodu dla omawianej instrukcji if:

```
if health <= 0:
    print("Ta instrukcja należy do warunku.")
print("Ta instrukcja zostanie wykonana niezależnie od warunku.")
```

Wcięcia nie służą wyłącznie poprawieniu wyglądu kodu. W GDScript są częścią jego składni.

3. Kilka możliwych ścieżek

```
func _update_interaction_hint() -> void:
    var polish: bool = _is_polish()
    var nearest := _nearest_container()
    if nearest != null:
        $UI/Interaction.text = "SPACJA — zbadaj skrytkę"
        if polish else "SPACE — examine cache"
    elif opened_containers >= CACHE_ORDER.size():
        $UI/Interaction.text = "ARSENAŁ ZABEZPIECZONY —
        IDŹ DO WINDY" if polish else "ARSENAL SECURED —
        REACH THE LIFT"
    else:
        $UI/Interaction.text = "SZUKAJ KOLEJNEGO ZNAKU
        MERLINA" if polish else "FIND THE NEXT SIGN OF
        MERLIN"
```

Jeżeli program powinien wybrać jedną z kilku możliwości, możemy użyć

if,

elif,

else:

```
func show_health_status(health: int) -> void:
    if health <= 0:
        print("Koniec gry!")
    elif health < 20:
        print("Uważaj, masz mało punktów życia!")
    elif health < 60:
        print("Postać jest ranna.")
    else:
        print("Postać jest w dobrej formie.")
```

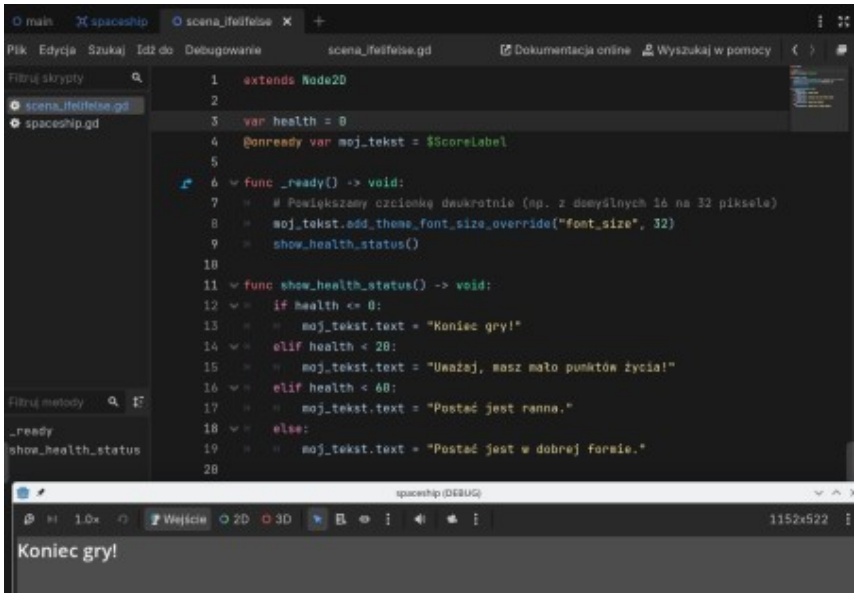
Godot sprawdza warunki od góry.

Jeżeli pierwszy warunek jest prawdziwy, wykonuje należący do niego blok i pomija pozostałe części tej konstrukcji.

Dla wartości:

health = 0

prawdziwe są zarówno warunki `health <= 0`, jak i `health < 20`.



Zostanie jednak wykonany tylko pierwszy pasujący blok:

print("Koniec gry!")

Dlatego kolejność warunków ma znaczenie. Zwykle zaczynamy od przypadków najbardziej szczegółowych albo najważniejszych.

4. Łączenie warunków

Pokazany powyżej Pokorny Dzwon - pokazuje trzy sposoby podejmowania decyzji:

- u góry brama otwiera się, gdy świecą **obydwa** sygnały (and);
- pośrodku do celu prowadzi **jedna z dwóch** tras (or);
- na dole statek wybiera drogę, na której **nie ma** zagrożenia (not).

Operatory: and or not

Za pomocą operatorów and, or i not możemy budować bardziej rozbudowane warunki.

Operator and

Obie części warunku muszą być prawdziwe:

```
func can_open_door(has_key: bool, door_is_locked: bool) -> bool:
    return has_key and door_is_locked
```

Operator or

Wystarczy, że przynajmniej jedna część warunku będzie prawdziwa:

```
func can_enter(has_ticket: bool, is_admin: bool) -> bool:
    return has_ticket or is_admin
```

Operator `not`

Operator `not` odwraca wartość logiczną:

```
func show_door_status(door_is_open: bool) -> void:
    if not door_is_open:
        print("Drzwi są zamknięte.")
```

Jeżeli `door_is_open` ma wartość `false`, wyrażenie:

```
not door_is_open
```

będzie miało wartość `true`.

5. Warunki zagnieżdżone

```
func _process_scan(delta: float) -> void:
    if not turn_completed:
        _process_turn_to_surface(delta)
        return
    phase_time = maxf(phase_time - delta, 0.0)
    scan_disruption = maxf(scan_disruption - delta, 0.0)
    spawn_timer -= delta
    if spawn_timer <= 0.0:
        _spawn_interference()
        spawn_timer = 2.8
    for hazard: Dictionary in hazards:
        hazard.position += hazard.velocity * delta
        if Vector2(hazard.position).distance_to($TargetReticle.position) <
        float(hazard.radius) + 22.0:
            hazard.dead = true
            scan_disruption = 1.5
            lock_progress = 0.0
    _remove_dead(hazards)
    if lock_index < LOCK_POINTS.size():
        var target: Vector2 = _current_lock_position(lock_index)
        if Input.is_action_pressed(&"ui_accept") and scan_disruption <= 0.0 and
        $TargetReticle.position.distance_to(target) <= LOCK_DISTANCE:
            lock_progress += delta
            if lock_progress >= LOCK_HOLD_TIME:
                locked_points[lock_index] = true
                lock_index += 1
                lock_progress = 0.0
        else:
            lock_progress = maxf(lock_progress - delta * 0.65, 0.0)
    if lock_index >= LOCK_POINTS.size():
        _enter_phase(PHASE_DEFENCE)
    elif phase_time <= 0.0:
        _restart(_tr("CZAS SKANU MINĄŁ — ROZWIĄZANIE OGNIOWE UTRACONE", "SCAN
        TIME EXPIRED — FIRING SOLUTION LOST"))
```

Witaj w gronie twórców. Do dzieła!

Serdecznie dziękujemy za poświęcenie Twojego czasu na zapoznanie się z początkiem drugiej części poradnika dla osób szukających wiedzy w obszarze gier, a w szczególności ich tworzenia w Silnku Godot Engine i języku kodowania GDScript.

Zapraszamy także do nabycia pełnej wersji tego ebooka. Więcej informacji znajdziesz na stronie wydawnictwa - www.poswojsku.pl

Pamiętaj, że na moim kanale:

youtube.com/@poswojsku

znajdziesz sporo dodatkowych poradników do silnika Godot Engine oraz języka GDScript.

Zapraszam także ***Ciebie*** do skorzystania z ***bezpłatnej gry 2D - 5 poziomów gry „Escape from the Seventh Planet”*** — od prologu i starcia z Harpiami oraz Mronami, przez uruchomienie statku i obronę przed dronami, aż po pierwszą lekcję lotu. Gra jest inspirowana powieściami z cyklu „Saga CyberJestestwa”. Możesz ją pobrać m.in. ze strony Wydawnictwa Cyfrowego poswojsku.pl

PODZIĘKOWANIA

Serdecznie dziękuję za zapoznanie się z zawartością mojego poradnika. Mam nadzieję, że przyniósł on Tobie Droga Czytelniczko/ Wspaniały Czytelniku - dużo wiedzy oraz zrozumienia odnośnie sposobów efektywnego wykorzystania narzędzi związanych z AI.

Zapraszam Ciebie także do kolejnych części poradnika „AI w edukacji".

W poradniku wykorzystano:

- własne materiały graficzne,
- prace AI,
- cliparty z programu LibreOffice na licencji CCO.

DZIĘKUJĘ ZA UWAGĘ

Autor poradnika: DARIUSZ GOŁĘBIOWSKI

Poznaj inne książki poswojsku.pl

Jeśli spodobał się Tobie ten poradnik i chcesz dalej rozwijać swoją wiedzę o cyberbezpieczeństwie, technologii i świadomym korzystaniu z internetu, oto moje inne książki, które mogą w tym pomóc. Każda z nich powstała z myślą o osobach, które szukają praktycznych wskazówek, konkretnych przykładów i języka zrozumiałego dla każdego.

Twoje bezpieczeństwo w świecie cyber i sztucznej inteligencji – Część 1: Wprowadzenie

Kompleksowy wstęp do tematyki ochrony danych, prywatności i bezpiecznego korzystania z sieci. To książka dla tych, którzy chcą szybko zrozumieć, na czym polegają najważniejsze zagrożenia i jak zacząć się przed nimi skutecznie bronić – bez skomplikowanego żargonu. Znajdziesz tu przykłady z życia i proste instrukcje krok po kroku.

Twoje bezpieczeństwo w świecie cyber i sztucznej inteligencji – Część 2: Cyberhigiena

Praktyczny przewodnik po codziennych nawykach, które realnie zwiększają Twoje bezpieczeństwo. Dowiesz się, jak tworzyć silne hasła, chronić urządzenia, wykrywać próby

oszustw i przygotować swoją rodzinę na zagrożenia cyfrowe. To pozycja obowiązkowa, jeśli chcesz, żeby cyberhigiena była naturalną częścią Twojego życia.

Twoje bezpieczeństwo w świecie cyber i sztucznej inteligencji – Część 3: Dziecko i Ty

Poradnik stworzony specjalnie dla rodziców i opiekunów, którzy chcą wprowadzać dzieci w cyfrowy świat z rozsądkiem i spokojem. Znajdziesz tu nie tylko zasady i rekomendacje, ale też gotowe sposoby rozmowy o bezpieczeństwie i budowania zaufania. Ta książka pomoże Ci chronić najmłodszych bez straszenia i nadmiernej kontroli.

AI w edukacji – Część 1: Praktyczny poradnik nie tylko dla nauczycieli

Sztuczna inteligencja to nie tylko moda, ale i realne narzędzie, które może usprawnić naukę i pracę. W tej książce pokazuję, jak korzystać z AI w prosty sposób – od generowania treści, przez wspieranie kreatywności, po automatyzację codziennych zadań. Idealna dla edukatorek/edukatorów i osób, które chcą poznać podstawy nowoczesnych technologii.

AI w edukacji – Część 2: Praktyczne pomysły na kreatywną edukację

Kontynuacja 1-ej części – pełna inspiracji, scenariuszy zajęć i ćwiczeń. Dowiesz się, jak prowadzić warsztaty i lekcje, które łączą AI z rozwojem kompetencji cyfrowych, logicznego myślenia i twórczego podejścia do nauki.

Stwórz Grę Mobilną

Praktyczny przewodnik dla osób, które marzą o stworzeniu własnej gry na smartfon. Od absolutnych podstaw programowania w JavaScript i React Native, przez projektowanie rozgrywki, aż po publikację gry. Jeśli chcesz uczyć się kodowania w sposób ciekawy i namacalny, to ta książka będzie Twoim drogowskazem.

Saga CyberJestestwa

Powieść science fiction dla tych, którzy chcą oderwać się od codzienności i zanurzyć w refleksyjnej historii o sensie istnienia, wolności i relacjach w obliczu zmian. To opowieść o ludziach i technologii, o wyborach i konsekwencjach – dla miłośniczek/miłośników literatury, którzy cenią głębsze przesłanie i oryginalny klimat.

Jeśli chcesz być na bieżąco z nowymi książkami, szkoleniami i inspiracjami o cyberbezpieczeństwie, technologii i AI – zapraszam Cię do obserwowania moich profili. Dzięki temu nie przegapisz premier, promocji i wartościowych materiałów które tworzę: często, prosto, przystępnie i z humorem.

- ***Strona internetowa 📍 poswojsku.pl***
- ***Facebook 📍 facebook.com/poswojsku***
- ***YouTube 📍 youtube.com/@poswojsku***
- ***LinkedIn 📍 linkedin.com/in/golebiowski-dariusz***
- ***Instagram 📍 instagram.com/poswojsku***
- ***Threads 📍 threads.com/@poswojsku***
- ***TikTok 📍 tiktok.com/@astilus***
- ***Amazon Author Page 📍 amazon.com/author/dariuszgolebiowski***
- ***Goodreads 📍 goodreads.com/dariuszgolebiowski***

Proszę, dołącz do mnie – razem budujmy bezpieczniejszy i bardziej świadomy świat cyfrowy!

Więcej informacji - **www.poswojsku.pl**

Prawa autorskie

Nazwy handlowe i znaki towarowe

Wszystkie wymienione w publikacji nazwy firm, organizacji, produktów, modeli, usług, platform oraz powiązane z nimi logotypy mogą być znakami towarowymi, zastrzeżonymi znakami towarowymi albo nazwami handlowymi należącymi do odpowiednich właścicieli.

W szczególności:

- **Godot**, **Godot Engine** oraz powiązane z nimi logotypy są znakami towarowymi należącymi do **Godot Foundation**.
- **GDScript** jest nazwą języka programowania stworzonego i rozwijanego w ramach otwartoźródłowego projektu Godot Engine.

Pozostałe nazwy firm, organizacji, produktów, modeli i usług wymienione w publikacji mogą stanowić znaki towarowe, zastrzeżone znaki towarowe albo nazwy handlowe należące do odpowiednich właścicieli.

Ich użycie w niniejszym poradniku służy wyłącznie identyfikacji i omówieniu opisywanych rozwiązań. Autor i wydawca nie roszczą sobie żadnych praw do cudzych znaków, nazw, logotypów ani elementów identyfikacji wizualnej.

Informacje dotyczące narzędzi i usług przedstawiają stan wiedzy na dzień opracowania publikacji. Nazwy, właściciele, funkcjonalności, warunki korzystania oraz dostępność poszczególnych usług mogą ulec zmianie.