Podstawy programowania gier

Wykład 3: Podstawy obiektowości i sterowania

Mgr inż. Staniszewski Hubert

Programowanie obiektowe

Paradygmat programowania, w którym programy definiuje się za pomocą obiektów.

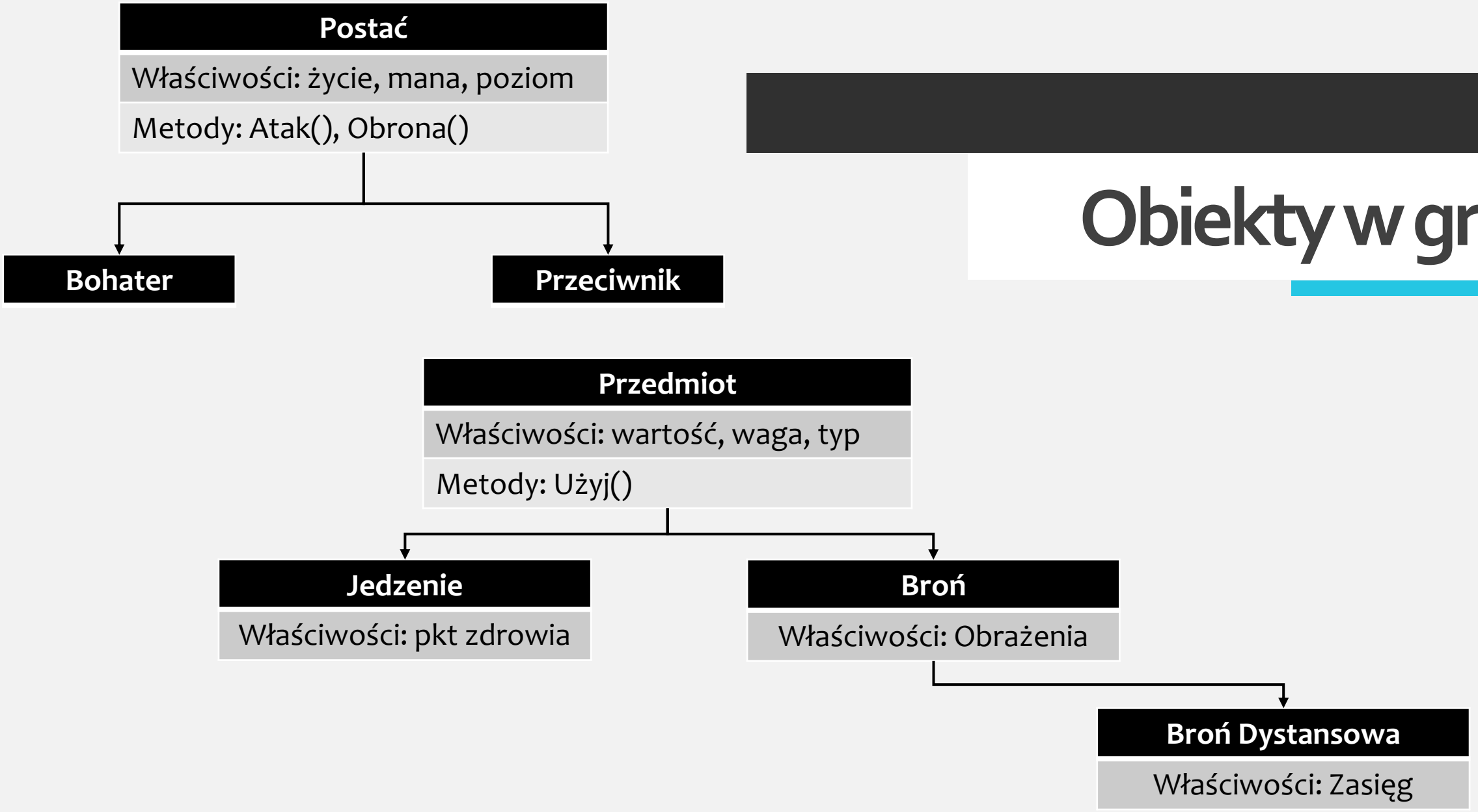
Programowanie obiektowe a proceduralne

Obiektowe	Proceduralne
• Podział na klasy i obiekty – klasa szablonem dla obiektu • Enkapsulacja – ograniczenie widoku dla użytkownika • Dziedziczenie • Polimorfizm – zdolność do używania tego samego interfejsu dla różnych obiektów • Abstrakcja – Upraszczanie złożonych rzeczywistości przez definiowanie abstrakcyjnych klas i obiektów.	• Procedury/Funkcje – podział na funkcje • Sekwencyjność – wykonywanie liniowe. • Modułowość

Programowanie obiektowe w grach

- Modularność i strukturalność,
- Reużywalność kodu,
- Łatwość utrzymania i rozbudowy,
- Abstrakcja złożonych systemów,
- Polimorfizm,
- Ponowne wykorzystanie mechanik w różnych częściach gry,
- Zespołowa praca nad projektem,
- Łatwość debugowania i testowania

Obiekty w grach



Podstawowe mechaniki

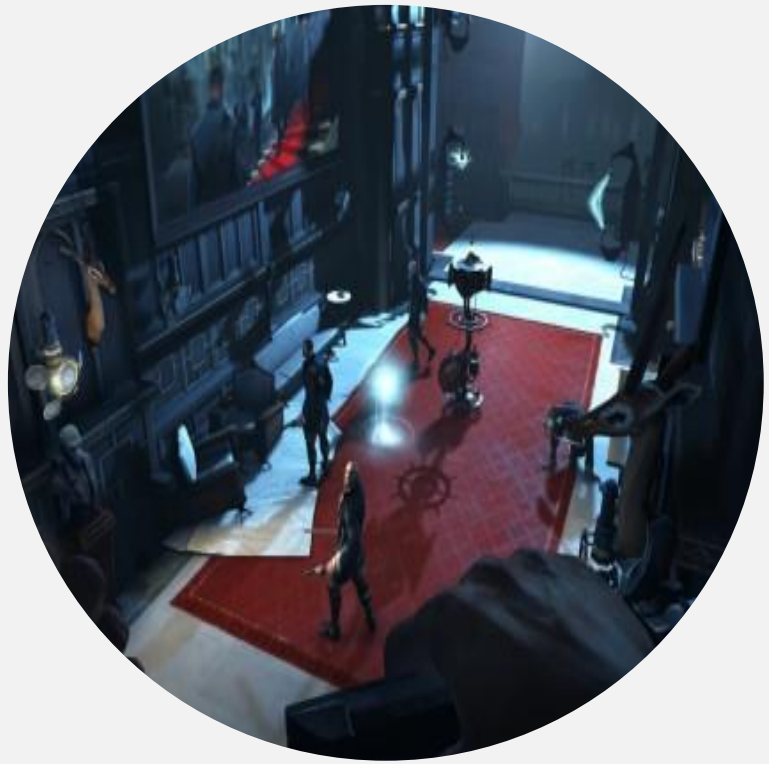
Co to są mechaniki

Mechaniki gier to zbiór systemów reguł i interakcji, które definiują sposób, w jaki gracz wchodzi w interakcję ze światem gry.

Sterowanie postacią gracza

Sterowanie skupia się na wszystkie czynności jakie gracz może wykonać przy pomocy swoich postaci wewnątrz gry. Ruch, interakcja, skok czy ruch kamerą to tylko przykłady.

Cechy dobrego sterowania w grach
Intuicyjność
Precyzja
Responsywność



Sterowanie w silnikach

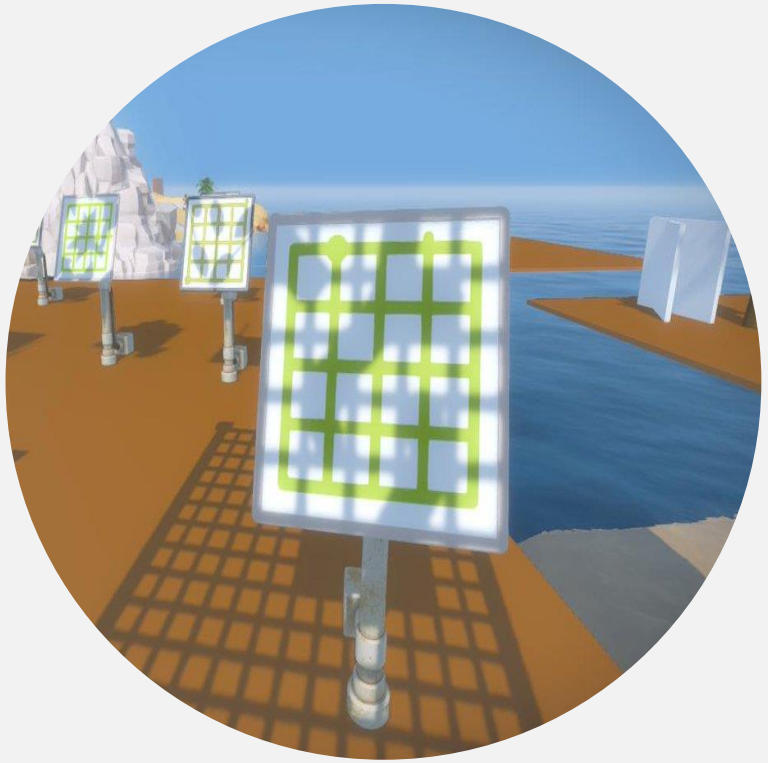
Jest kilka czynności które należy wykonać w każdym z silników aby sterowanie było możliwe. Część elementów takich jak np. kontroler gracza musi komunikować się z postacią.

Sterowanie w silnikach

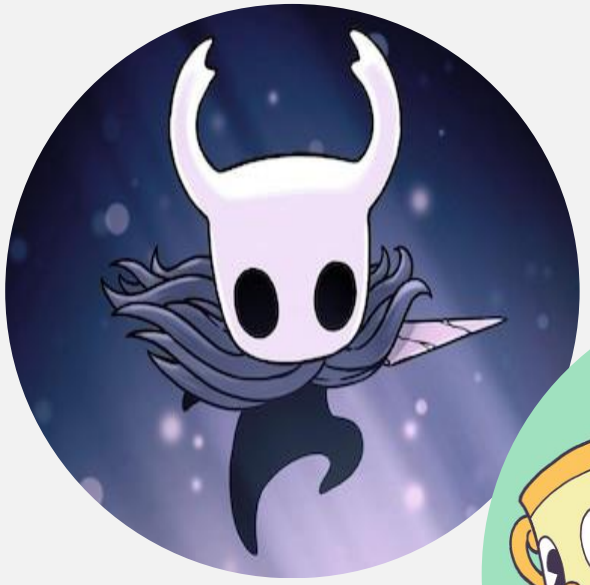
Unity	Unreal Engine	Godot
Używa C# i komponentów takich jak **Rigidbody2D** oraz **Collider2D**. Jest stosunkowo prosty w implementacji i oferuje elastyczne narzędzia do sterowania postacią z dużą kontrolą nad fizyką.	Zastosowanie **Blueprints** pozwala na wizualne tworzenie logiki gry bez konieczności pisania kodu. Z kolei C++ daje zaawansowaną kontrolę nad każdym aspektem ruchu i fizyki postaci, ale jest bardziej złożone.	Opiera się na skryptach w GDScript, które są lekkie i szybkie do napisania. **KinematicBody2D** pozwala na precyzyjną kontrolę ruchu i kolizji bez konieczności używania pełnej symulacji fizyki.

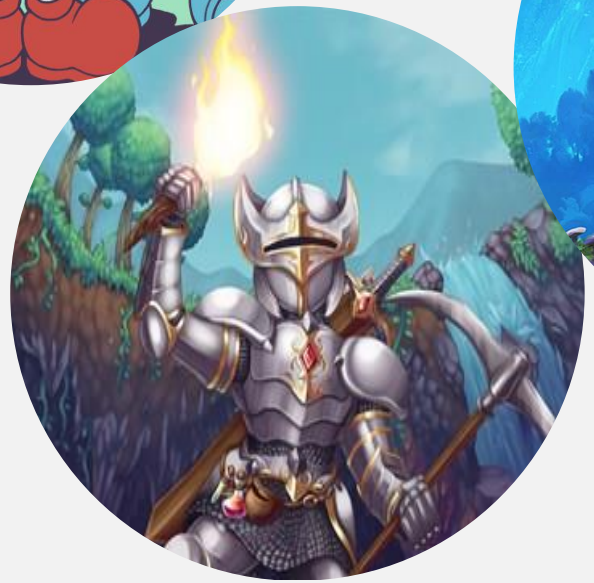
Ogólny schemat sterowania

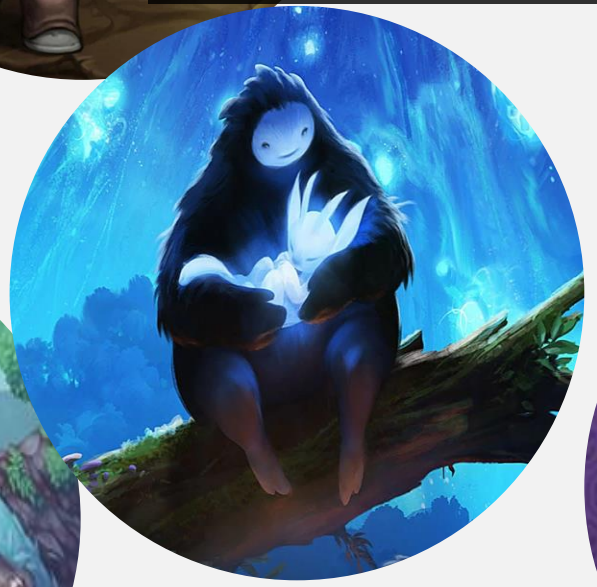
1. Przygotowanie postaci
2. Odczyt wejścia od gracza
3. Implementacja ruchu
4. Obsługa fizyki i kolizji
5. Testowanie i optymalizacja sterowania
6. Dodanie zaawansowanych mechanik (opcjonalne)
7. Integracja z efektami



Przykłady gier 2D z bardzo dobrym sterowaniem

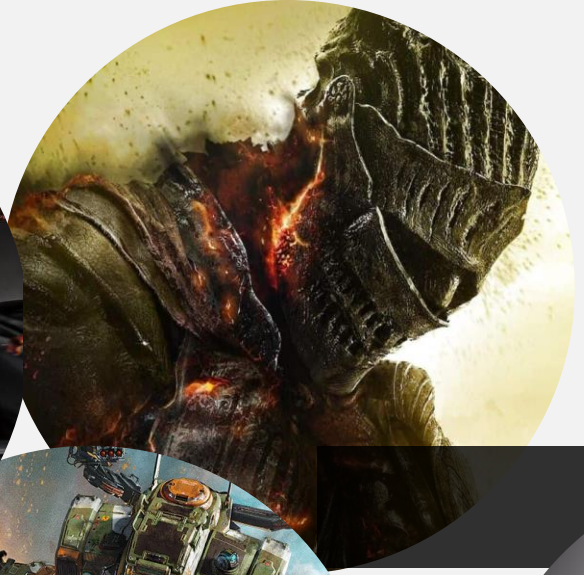






Przykłady gier 3D ze świetnym sterowaniem







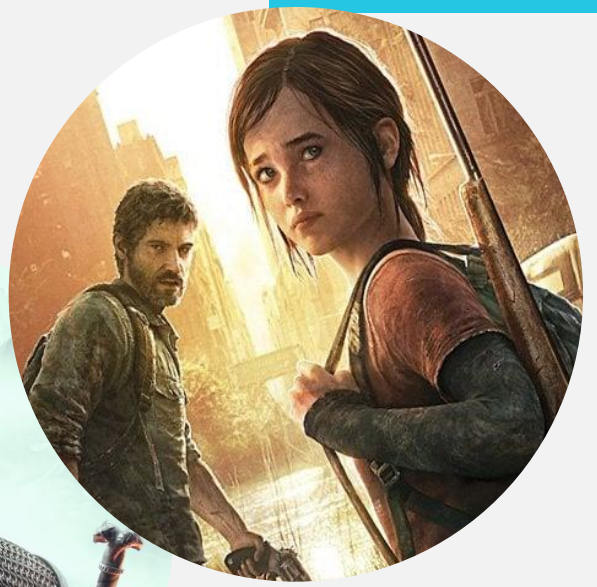


Przykłady gier ze słabym sterowaniem


STEEL BATTALION





Pytania

Dziękuję za uwagę