

Wprowadzenie (13)

Część I Elementy C++ (21)

Rozdział 1. Dziedziczenie (23)

- Klasy (23)
- Dziedziczenie (24)
- Polimorfizm i funkcje wirtualne (27)
- Dziedziczyć czy nie dziedziczyć? (30)
 - o Zasada 1. Dziedziczenie kontra zawieranie (30)
 - o Zasada 2. Zachowanie kontra dane (31)
- Kiedy stosować dziedziczenie, a kiedy go unikać (31)
- Implementacja dziedziczenia (zaawansowane) (33)
- Analiza kosztowa (zaawansowane) (35)
- Alternatywy (zaawansowane) (38)
- Dziedziczenie a architektura programu (zaawansowane) (39)
- Wnioski (41)
- Zalecana lektura (41)

Rozdział 2. Dziedziczenie wielobazowe (43)

- Stosowanie dziedziczenia wielobazowego (43)
 - o Wszystko w jednym (44)
 - o Wersja z zawieraniem (44)
 - o Dziedziczenie zwyczajne (46)
 - o Ratunek w dziedziczeniu wielobazowym (47)
- Problemy dziedziczenia wielobazowego (47)
 - o Niejednoznaczność (48)
 - o Topografia (48)
 - o Architektura programu (51)
- Polimorfizm (51)
- Kiedy stosować dziedziczenie wielobazowe, a kiedy go unikać (53)
- Implementacja dziedziczenia wielobazowego (zaawansowane) (54)
- Analiza kosztowa (zaawansowane) (55)
 - o Rzutowanie (55)
 - o Funkcje wirtualne z drugiej (kolejnej) klasy nadrzędnej (57)
- Wnioski (58)
- Zalecana lektura (58)

Rozdział 3. O stałości, referencjach i o rzutowaniu (59)

- Stałość (59)
 - o Koncepcja stałości (60)
 - o Stałość a wskaźniki (60)
 - o Stałość a funkcje (61)
 - o Stałość a klasy (64)
 - o Stałe, ale zmienne - słowo mutable (65)
 - o Słowo porady odnośnie const (66)
- Referencje (67)
 - o Referencje kontra wskaźniki (67)
 - o Referencje a funkcje (68)

- o Zalety referencji (69)
 - o Kiedy stosować referencje (71)
- Rzutowanie (72)
 - o Potrzeba rzutowania (72)
 - o Rzutowanie w konwencji C++ (73)
- Wnioski (76)
- Zalecana lektura (76)

Rozdział 4. Szablony (77)

- W poszukiwaniu kodu uniwersalnego (77)
 - o Podejście pierwsze: lista wbudowana w klasę (78)
 - o Podejście drugie: makrodefinicja (79)
 - o Podejście trzecie: dziedziczenie (80)
 - o Podejście czwarte: uniwersalna lista wskaźników void (81)
- Szablony (83)
 - o Szablony klas (83)
 - o Szablony funkcji (85)
 - o Powrót do problemu listy - próba z szablonem (85)
- Wady (87)
 - o Złożoność (87)
 - o Zależności (87)
 - o Rozdęcie kodu (88)
 - o Obsługa szablonów przez kompilator (88)
- Kiedy stosować szablony (88)
- Specjalizacje szablonów (zaawansowane) (89)
 - o Pełna specjalizacja szablonu (90)
 - o Częściowa specjalizacja szablonu (91)
- Wnioski (91)
- Zalecana lektura (92)

Rozdział 5. Obsługa wyjątków (93)

- Jak sobie radzić z błędami (93)
 - o Ignorować! (93)
 - o Stosować kody błędów (94)
 - o Poddać się (z asercjami)! (95)
 - o Stosować wywołania setjmp() i longjmp() (95)
 - o Stosować wyjątki C++ (96)
- Stosowanie wyjątków (97)
 - o Wprowadzenie (98)
 - o Zrzucanie wyjątków (98)
 - o Przechwytywanie wyjątków (100)
- Odporność na wyjątki (103)
 - o Pozyskiwanie zasobów (104)
 - o Konstruktory (107)
 - o Destruktory (109)
- Analiza kosztowa (109)
- Kiedy stosować wyjątki (111)
- Wnioski (112)

- Zalecana lektura (112)

Część II Wydobywanie mocy C++ (113)

Rozdział 6. Wydajność (115)

- Wydajność i optymalizacje (115)
 - W czasie programowania (117)
 - Pod koniec programowania (118)
- Rodzaje funkcji (119)
 - Funkcje globalne (119)
 - Statyczne funkcje klas (120)
 - Niewirtualne składowe klas (120)
 - Funkcje wirtualne przy dziedziczeniu pojedynczym (121)
 - Funkcje wirtualne przy dziedziczeniu wielobazowym (123)
- Rozwijanie funkcji w miejscu wywołania (124)
 - Potrzeba rozwijania w funkcji (124)
 - Funkcje rozwijane w miejscu wywołania (125)
 - Kiedy stosować rozwijanie zamiast wywołania? (127)
- Jeszcze o narzutach wywołań funkcji (128)
 - Parametry funkcji (128)
 - Wartości zwracane (130)
 - Funkcje puste (132)
- Unikanie kopiowania (133)
 - Argumenty wywołań (133)
 - Obiekty tymczasowe (134)
 - Jawność intencji (135)
 - Blokowanie kopiowania (135)
 - Dopuszczanie kopiowania (136)
 - Przeciążanie operatorów (137)
- Konstruktory i destruktory (139)
- Pamięci podręczne i wyrównywanie danych w pamięci (zaawansowane) (143)
 - Wzorce odwołań do pamięci (144)
 - Rozmiar obiektów (145)
 - Rozmieszczanie składowych w obiektach (146)
 - Wyrównanie pamięci (146)
- Wnioski (147)
- Zalecana lektura (148)

Rozdział 7. Przydział pamięci (149)

- Stos (149)
- Sterta (150)
 - Wydajność przydziału (151)
 - Fragmentacja pamięci (152)
 - Inne problemy (154)
- Przydziały statyczne (155)
 - Zalety i wady przydziału statycznego (156)
 - Kiedy korzystać z przydziałów statycznych (157)
- Przydziały dynamiczne (158)
 - Łańcuch wywołań (158)

- o Globalne operatory new i delete (159)
 - o Operatory new i delete dla klas (161)
- Własny menedżer pamięci (163)
 - o Kontrola błędów (163)
 - o Przeglądanie stert (166)
 - o Zakładki i wycieki (166)
 - o Sterty hierarchiczne (167)
 - o Inne rodzaje przydziałów (169)
 - o Narzut mechanizmu zarządzania pamięcią (170)
- Pule pamięci (171)
 - o Implementacja (172)
 - o Podłączanie puli do sterty (175)
 - o Pule uniwersalne (176)
- W nagłych wypadkach... (177)
- Wnioski (178)
- Zalecana lektura (179)

Rozdział 8. Wzorce projektowe w C++ (181)

- Czym są wzorce projektowe? (181)
- Wzorzec Singleton (183)
 - o Przykład: menedżer plików (183)
 - o Implementacja Singletona (184)
- Wzorzec Façade (185)
 - o Fasada dla systemu "w budowie" (188)
 - o Fasada dla przeróbek (189)
- Wzorzec Observer (190)
- Wzorzec Visitor (195)
- Wnioski (199)
- Zalecana lektura (199)

Rozdział 9. Kontenery STL (201)

- Przegląd STL (201)
- Korzystać czy nie korzystać? (203)
 - o Wykorzystanie gotowego kodu (203)
 - o Wydajność (204)
 - o Wady (205)
- Kontenery sekwencyjne (206)
 - o Kontener vector (206)
 - o Kontener deque (211)
 - o Kontener list (214)
- Kontenery asocjacyjne (217)
 - o Kontenery set i multiset (218)
 - o Kontenery map i multimap (222)
 - o Kontenery haszowane (226)
- Adaptory kontenerów (230)
 - o Stos (231)
 - o Kolejka (231)
 - o Kolejka priorytetowa (232)

- Wnioski (233)
- Zalecana lektura (235)

Rozdział 10. STL: algorytmy i zagadnienia zaawansowane (237)

- Obiekty funkcyjne (funktory) (237)
 - o Wskaźniki funkcji (237)
 - o Funktory (238)
 - o Adaptory funktorów (240)
- Algorytmy (241)
 - o Algorytmy niemodyfikujące (242)
 - o Algorytmy modyfikujące (245)
 - o Algorytmy sortujące (248)
 - o Uogólnione algorytmy numeryczne (249)
- Ciągi znaków (250)
 - o Ciągłe bez ciągów (250)
 - o Klasa string (252)
 - o Wydajność (254)
 - o Pamięć (255)
 - o Alternatywy (256)
- Alokatory (zaawansowane) (257)
- Kiedy STL nie wystarcza (zaawansowane) (260)
- Wnioski (262)
- Zalecana lektura (262)

Rozdział 11. Poza STL: własne struktury i algorytmy (265)

- Grafy - studium przypadku (265)
 - o Powtórka z grafów (265)
 - o Ogólniej o grafie (267)
 - o W kwestii kosztów (267)
 - o Koszt przebycia krawędzi a jakość rozgrywki (268)
- Grafy w C++ (269)
- Zaprząć grafy do pracy (271)
- "Inteligencja" grafów (276)
 - o Życie na krawędzi (276)
 - o Aktualizacja, wracamy do C++ (279)
 - o Implementacja wyznaczania trasy (285)
 - o A może A* (zaawansowane) (293)
- Inne zastosowania grafów (294)
 - o Optymalizacja przejść w interfejsie użytkownika (296)
 - o Brakujące ścieżki powrotne (298)
 - o Zagubione plansze menu (298)
- Wnioski (299)
- Zalecana lektura (299)

Część III Techniki specjalne (301)

Rozdział 12. Interfejsy abstrakcyjne (303)

- Interfejsy abstrakcyjne (303)

- Implementacja interfejsów w C++ (305)
- Interfejsy abstrakcyjne w roli bariery (306)
 - o Nagłówki i wytwórnice (308)
 - o Z życia (310)
- Interfejsy abstrakcyjne w roli charakterystyk klas (312)
 - o Implementacje (313)
 - o Pytanie o interfejs (315)
 - o Rozszerzanie gry (317)
- Nie wszystko złoto... (319)
- Wnioski (320)
- Zalecana lektura (321)

Rozdział 13. Wtyczki (323)

- Po co komu wtyczki (323)
 - o Wtyczki do cudzych programów (324)
 - o Wtyczki do własnych programów (325)
- Architektura wtyczek (326)
 - o Interfejs wtyczek (326)
 - o Tworzenie konkretnych wtyczek (327)
 - o Obsługa różnych rodzajów wtyczek (328)
 - o Ładowanie wtyczek (329)
 - o Menedżer wtyczek (331)
 - o Komunikacja dwukierunkowa (333)
- Żeby miało ręce i nogi... (335)
- Wtyczki w praktyce (335)
 - o Stosowanie wtyczek (336)
 - o Wady (336)
 - o Inne platformy (337)
- Wnioski (338)
- Zalecana lektura (338)

Rozdział 14. C++ a skrypty (339)

- Po co jeszcze jeden język, i to skryptowy? (339)
 - o Złe wieści... (340)
 - o I wieści dobre (341)
- Rozważania o architekturze (344)
 - o Engine kontra gameplay (345)
- Zintegrowany interpreter skryptów - nie tylko w sterowaniu rozgrywką (349)
 - o Konsola (349)
 - o Interaktywne sesje diagnostyczne (350)
 - o Szybkie prototypowanie (352)
 - o Automatyzacja testów (353)
- Wnioski (355)
- Zalecana lektura (356)
 - o Lua (356)
 - o Python (357)
 - o GameMonkey Script (357)
 - o AngelScript (358)

Rozdział 15. Informacja o typach w czasie wykonania (359)

- Praca bez RTTI (359)
- Używanie i nadużywanie RTTI (361)
- Standardowe RTTI w C++ (363)
 - Operator `dynamic_cast` (363)
 - Operator `typeid` (365)
 - Analiza RTTI w wydaniu C++ (366)
- Własny system RTTI (368)
 - Najprostsze rozwiązanie (368)
 - Z obsługą dziedziczenia pojedynczego (372)
 - Z obsługą dziedziczenia wielobazowego (375)
- Wnioski (378)
- Zalecana lektura (378)

Rozdział 16. Tworzenie obiektów i zarządzanie nimi (379)

- Tworzenie obiektów (379)
 - Kiedy `new` nie wystarcza (380)
 - Wielka selekcja (381)
- Wytwórnice obiektów (382)
 - Prosta wytwórnica (382)
 - Wytwórnica rozproszona (383)
 - Jawne rejestrowanie obiektów wytwórczych (384)
 - Niejawne rejestrowanie obiektów wytwórczych (385)
 - Identyfikatory typów obiektów (387)
 - Od szablonu (388)
- Obiekty współużytkowane (389)
 - Bez wspólnych obiektów (390)
 - Ignorowanie problemu (391)
 - Niech się właściciel martwi (392)
 - Zliczanie odwołań (393)
 - Uchwyty (396)
 - Inteligentne wskaźniki (398)
- Wnioski (401)
- Zalecana lektura (402)

Rozdział 17. Utrwalanie obiektów (405)

- Przegląd zagadnień dotyczących utrwalania jednostek gry (405)
 - Jednostki kontra zasoby (406)
 - Najprostsze rozwiązanie, które nie zadziała (406)
 - Czego potrzebujemy (407)
- Implementacja utrwalania jednostek gry (409)
 - Strumienie (409)
 - Zapisywanie (412)
 - Wczytywanie (416)
- Współ w zespół (419)
- Wnioski (420)
- Zalecana lektura (421)

Rozdział 18. Postępowanie z dużymi projektami (423)

- Struktura logiczna a struktura fizyczna (423)
- Klasy i pliki (425)
- Pliki nagłówkowe (426)
 - o Co ma się znaleźć w pliku nagłówkowym? (426)
 - o Bariery włączania (427)
 - o Dyrektywy `#include` w plikach implementacji (430)
 - o Dyrektywy `#include` w plikach nagłówkowych (432)
 - o Wstępnie kompilowane pliki nagłówkowe (434)
 - o Wzorzec implementacji prywatnej (437)
- Biblioteki (440)
- Konfiguracje (443)
 - o Konfiguracja diagnostyczna (444)
 - o Konfiguracja dystrybucyjna (444)
 - o Konfiguracja diagnostyczna zoptymalizowana (445)
- Wnioski (445)
- Zalecana lektura (446)

Rozdział 19. Zbrojenie gry (447)

- Stosowanie asercji (447)
 - o Kiedy stosować asercje (448)
 - o Kiedy nie stosować asercji (450)
 - o Własne asercje (451)
 - o Co powinno się znaleźć w ostatecznej wersji (452)
 - o Własna przykładowa implementacja asercji (454)
- Zawsze świeżo (455)
 - o Wycieki pamięci (456)
 - o Fragmentacja pamięci (456)
 - o Dryf zegara (456)
 - o Kumulacja błędów (457)
 - o Co robić? (457)
- Postępowanie ze "złymi" danymi (458)
 - o Wykrywać asercjami (459)
 - o Radzić sobie samemu (459)
 - o Kompromis (461)
- Wnioski (463)
- Zalecana lektura (463)

Skorowidz (465)