

Projekt 1: Komunikacja międzyprocesowa

Materiały wykładowe dostępne w usłudze Pliki w Chmurze Politechniki Łódzkiej:

(<https://pwc.p.lodz.pl/d/60cd5c043f12486990a7/>) <http://bit.ly/sysopy2>.

Jeżeli w tekście są braki lub niejasności to skontaktuj się z koordynatorem przedmiotu (Tomasz Jaworski, tjaworski@iis.p.lodz.pl) w celu korekty instrukcji.

Zadanie

W ramach projektu z komunikacji międzyprocesowej należy przygotować prostą grę planszową dla czterech niezależnych graczy, grających równolegle. Zarówno gracze jak i serwer mają być oddzielnymi procesami. Ponieważ graczem może być zarówno komputer jak i człowiek to sugerowane jest, aby całość składała się z trzech programów: serwera gry, klienta gracza, klienta bota.

Gracze (1, 2, 3, 4) zamknięci są w labiryncie i mają za zadanie zbierać pojawiające się skarby w postaci monet (c, t, T). Gracz, który zbierze wystarczająco dużo skarbów to zanoszi je do obozowiska (A) i tam zostawia.

Gracz może w danej chwili nosić dowolną liczbę monet (carried) ale może je stracić w wyniku ataku dzięki bestii (*) lub poprzez zderzenie się z innym graczem.

- W przypadku ataku dzięki bestii gracz ginie (deaths) a zebrany przez niego łup pozostaje w miejscu śmierci (D). Gracz respawnuje się w swoim punkcie startowym.
- W przypadku zderzenia z innym graczem łupy obu pozostają w miejscu zderzenia (D) a gracze respawnują się w swoich punktach startowych.
- Pozostawiony łup (D) ma swoją wartość. W przypadku zderzenia jest to suma noszonych monet obu graczy.

Gracz pozbywa się swoich monet w obozowisku (A) gdzie zapisywane są one na jego konto (brought). Po zdaniu skarbu gracz kontynuuje poszukiwania, zaczynając od obozowiska.

Typy graczy

Należy przygotować dwa typy graczy: komputer – bot (CPU) oraz człowiek (HUMAN).

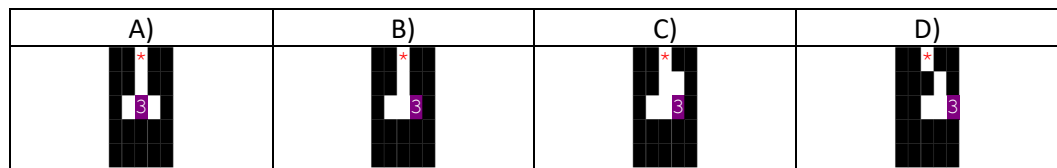
- Postać **gracza-bota** porusza się autonomicznie na podstawie mapy przekazywanej przez serwer.
- Postać **gracza-człowieka** poruszana jest za pomocą klawiszy strzałek (w górę, w dół, w lewo, w prawo).

Gracz CPU może poruszać się z wykorzystaniem dowolnego algorytmu, np. chaotycznie, A* w dowolny punkt, A* z eksploracją, RL, lewa ściana, itp. Gracz CPU powinien również reagować na pojawienie się bestii, serwując się ucieczką.

Przeciwnicy

W grze należy przewidzieć co najmniej jeden typ przeciwnika – „dziką bestię” (*) – która ma zachowywać się w następujący sposób:

- Jeżeli gracz nie ma w polu widzenia to bestia może bezcelowo poruszać się w dowolnym kierunku lub stać w miejscu (lub łączyć te dwa zachowania).
- Jeżeli gracz pojawił się w polu widzenia to bestia podejmuje pościg aż do ucieczki gracza z pola widzenia. Po zgubieniu gracza bestia może wrócić do bezcelowego poruszania się lub dotrzeć do miejsca, w którym ostatnio widziała gracza.
 - Zgubienie gracza ma miejsce tylko wtedy, gdy linia prosta między środkiem postaci gracza a środkiem postaci bestii jest przecięta przez obiekt ściany:
a) bestia widzi gracza (pościg); b) bestia nie widzi gracza; c) bestia widzi gracza; d) bestia widzi gracza, ale nie ma jak podjąć pościgu.



Pole widzenia

Serwer nie przekazuje graczom kompletnej mapy w jakikolwiek sposób. Przekazuje jedynie dane w ramach pola widzenia gracza. Przez pole widzenia gracza należy rozumieć obszar o promieniu dwóch pól dookoła aktualnej pozycji postaci (patrz **Widok gracza**). Wraz z polem widzenia gracz otrzymuje swoją pozycję w świecie (GPS). Graczowi nie wolno zakładać jakichkolwiek informacji o świecie. Jedyne dopuszczalne założenie to maksymalny rozmiar świata – 128x128.

Wraz z informacjami o polu widzenia gracz otrzymuje od serwera swoje współrzędne. Informacje te można wykorzystać do budowy mapy świata po stronie gracza.

Dołączanie do i porzucanie gry

Serwer może obsługiwać maksymalnie czterech graczy. Jeżeli użytkownik spróbuje uruchomić kolejnego gracza (proces gracza/bota) to ten ma wyświetlić komunikat o pełnym serwerze i zakończyć się.

Dla każdego nowo przybyłego gracza losowane jest miejsce (współrzędne XY) spawnu i jego postać jest tam wstawiana. Gracz człowiek jak i bot mają być traktowane tak samo. Z punktu widzenia serwera nie może istnieć faworyzowanie jednego bądź drugiego. Wymusza to wspólny interfejs API między serwerem a jego klientami.

Serwer powinien umieć radzić sobie z sytuacją, gdy gracz opuszcza grę w naturalny sposób (wciskając Q po stronie klienta) lub nagły sposób (zabicie procesu).

Podział czasu gry

Gra składa się z tur, z których każda ma trwać nie więcej niż sekundę (sugerowana parametryzacja na czas pisania kodu). W ramach jednej tury każda postać (bestia/gracz/bot) może przemieścić się nie więcej niż o jedno pole. Może też stać w miejscu. Ruch ma odbywać się na zasadzie informowania serwera o chęci zrobienia kroku w jednym z czterech kierunków. Serwer ma prawo odmówić wykonania ruchu (np. jeżeli gracz będzie chciał przejść przez ścianę).

Jeżeli klient nie zadeklaruje chęci ruchu w danej turze, to gra jest kontynuowana. Ułatwi to proces debugowania klientów, gdy jeden z nich jest wstrzymany w trybie debugowania.

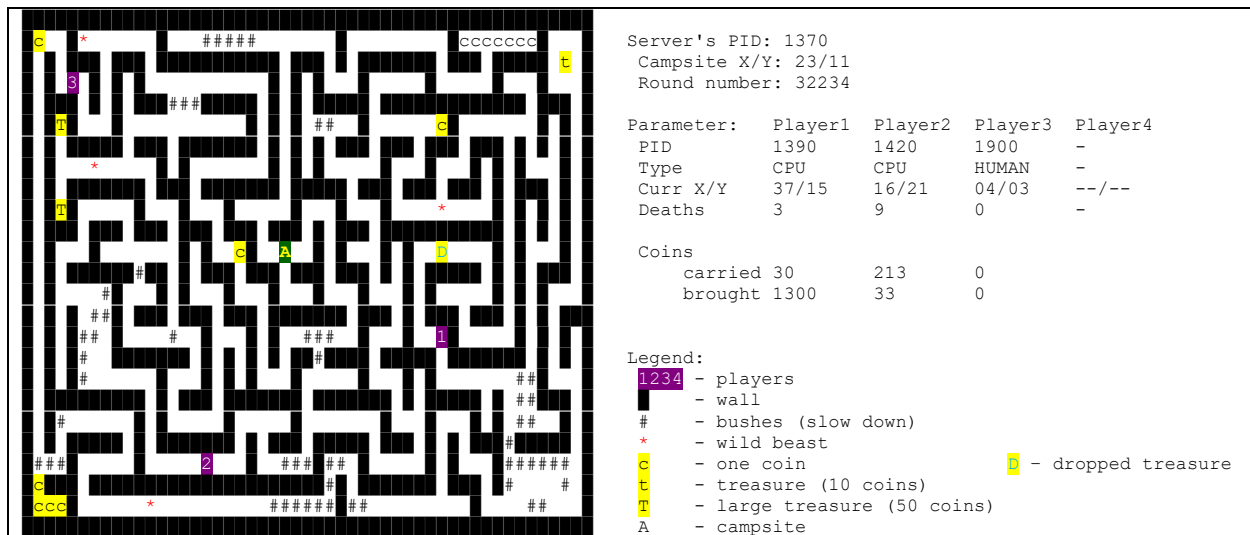
Przeszkody

W grze należy zaimplementować co najmniej dwa typy przeszkód: ścianę (■) oraz krzaki (#). Postać nie może przejść przez ścianę – serwer nie może do tego dopuścić. Przejście przez krzaki jest możliwe, ale wymaga to dwóch tur, w ramach których gracz próbuje przejść przez przeszkodę.

Statystyki

Wszelkie statystyki przechowywane i liczone są po stronie serwera. Gracz jest jedynie informowany o ich wartościach. Dzięki temu gracz nie może ich „sztucznie” zmienić (np. dodając pewną liczbę monet).

Widok serwera



The screenshot displays a game server interface. On the left is a maze map with various symbols: numbers 1-4 for players, '#' for walls, '*' for wild beasts, 'c' for coins, 't' for treasure, and 'A' for a campsite. On the right is a text-based status window.

Server's PID: 1370
Campsite X/Y: 23/11
Round number: 32234

Parameter:	Player1	Player2	Player3	Player4
PID	1390	1420	1900	-
Type	CPU	CPU	HUMAN	-
Curr X/Y	37/15	16/21	04/03	--/--
Deaths	3	9	0	-

Coins

	Player1	Player2	Player3
carried	30	213	0
brought	1300	33	0

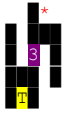
Legend:

- 1234 - players
- - wall
- # - bushes (slow down)
- * - wild beast
- c - one coin
- t - treasure (10 coins)
- T - large treasure (50 coins)
- A - campsite
- - dropped treasure

Serwer musi równolegle z grą obsługiwać klawiaturę, dopuszczając następujące reakcje na wciskane klawisze:

- B/b – dodanie jednej bestii w losowym miejscu labiryntu
- c/t/T – dodanie nowej monety, skarbu, dużego skarbu w losowym miejscu labiryntu
- Q/q – zakończenie gry

Widok gracza:



Server's PID: 1370
Campsite X/Y: unknown
Round number: 32234

Player:
Number: 3
Type: HUMAN
Curr X/Y 04/03
Deaths 0

Coins found 0
Coins brought 0

Legend:
1234 - players
■ - wall
- bushes (slow down)
* - enemy
c - one coin
t - treasure (10 coins)
T - large treasure (50 coins)
A - campsite

Klient gracza-człowieka powinien obsługiwać klawisze strzałek (góra, dół, lewo, prawo) do poruszania postacią gracza. Ponadto klienty gracza-człowieka i bota powinny dawać możliwość kontrolowanego zakończenia gry (klawisz Q). Zabicie procesu gracza **nie jest** kontrolowanym sposobem zakończenia gry.

Warunki zaliczenia

- Serwer oraz klienty muszą komunikować się ze sobą jedynie za pomocą mechanizmów IPC w ramach tej samej maszyny. Niedopuszczalne jest wymienianie danych poprzez fizyczne pliki.
- Kolejność uruchamiania procesów składowych nie może mieć znaczenia.
- Programy nie muszą być napisane w tych samych językach¹.
- Nie wolno stosować gotowych bibliotek, opakowujących systemowe mechanizmy IPC. Jeżeli uważasz, że znaleziona przez Ciebie biblioteka jest w porządku i chcesz ją wykorzystać – uzgodnij to wcześniej z koordynatorem przedmiotu SO2.

Podpowiedzi

- Czy gracz-bot różni się czymś szczególnym od bota bestii?

¹ Każdy może być napisany w innym języku.