Jamy & Nory

Michalis Kamburelis

Kilka fragmentów kodu i część dokumentacji:
Michał Sałaban
Rafał Wawrzkowicz

Spis treści

1. Ogólny opis projektu

Jamy & Nory to gra utrzymana w konwencji MUD, czyli *multiple user dungeon*. Projekt składa się z dwóch zasadniczych komponentów: programu serwera i programu klienta. Serwer zarządza światem gry i pozwala przyłączać się do gry (poprzez Internet) klientom. Klient pozwala użytkownikowi włączyć się do świata działającego na wskazanym serwerze. Użytkownik połączony z serwerem za pomocą swojego klienta staje się *graczem*. Gracz może poruszać się po świecie gry i wpływać na niego na wiele sposobów – podnosząc przedmioty, zabijając potwory, a także walcząc i rozmawiając z innymi graczami.

Gracz może w dowolnej chwili opuścić grę (używając polecenia `koniec`). Również nieoczekiwane zakończenie procesu klienta i idące za tym zerwanie połączenia w „brutalny” sposób nie spowodują utraty informacji na serwerze. Bez względu na to w jaki sposób klient odłączy się od gry, aktualny stan jego postaci zostanie zachowany na serwerze. Użytkownik może potem połączyć się ponownie i będzie mógł kontynuować grę swoją dotychczasową postacią. Przy ponownym podłączeniu się będzie musiał podać imię swojej postaci oraz jej hasło (wszystkie postacie graczy przechowywane na serwerze są zabezpieczone hasłem aby nikt nie mógł grać postacią stworzoną przez kogoś innego).

Nawiasem mówiąc rodzi to pewne problemy, bo pozwala graczowi zawsze uciec przed walką poprzez opuszczenie gry. Taka sytuacja jest nie do uniknięcia – nie możemy np. karać gracza śmiercią postaci za opuszczenie gry w czasie walki, gdyż w ten sposób awaria oprogramowania, sprzętu lub zerwanie połączenia (nie z winy klienta) powodowałyby utratę dotychczasowego dorobku postaci, co z pewnością nie byłoby mile widziane. Zresztą gra działa w taki sposób, że od walki i tak zazwyczaj można uciec po prostu przechodząc do sąsiedniego pomieszczenia.

Po świecie gry może poruszać się dowolnie wielu graczy. Z drugiej strony, gra nie zostaje zawieszona nawet wówczas gdy wszyscy klienci odłączą się od serwera. Niezależnie od graczy świat cały czas podlega przemianom, postacie sterowane sztuczną inteligencją same z siebie podejmują różne akcje – chodzą po świecie, atakują się wzajemnie, itd.

2. Przed uruchomieniem

Oficjalna strona dystrybucji programu jest na stronach Michalisa[1]. Stamtąd możesz ściągnąć program w wersji skompilowanej, dokumentację (tą którą właśnie czytasz) wygenerowaną w różnych formatach oraz źródła do programu. Jeśli chcesz po prostu możliwie łatwo pograć w grę to polecam ściągnąć wersję skompilowaną.

Program jest napisany w Javie. Aby go uruchomić potrzebujesz Java Runtime Environment[2].

Po zainstalowaniu Java Runtime Environment musisz ustawić zmienną środowiskową CLASSPATH tak żeby zawierało pełną nazwę pliku (tzn. ze ścieżką) do jamy_i_nory.jar.

- **Przykład dla użytkowników Linuxa**
  Załóżmy że rozpakowałeś wersję skompilowaną *Jam & Nor* do kata-

[1] http://www.camelot.homedns.org/~michalis/jamy_i_nory.php
[2] http://java.sun.com/getjava/

logu `/home/michal/jamy/`. Więc ustaw `$CLASSPATH` żeby zawierało plik `/home/michal/jamy/jamy_i_nory.jar`, np. dopisując do `.bashrc`

```
export CLASSPATH="$CLASSPATH:/home/michal/jamy/jamy_i_nory.jar"
```

- **Przykład dla użytkowników Windowsa**
  Załóżmy że rozpakowałeś wersję skompilowaną *Jam & Nor* do katalogu `c:/jamy/`. Więc ustaw `$CLASSPATH` żeby zawierało plik `c:/jamy/jamy_i_nory.jar`, np. pod Windowsami 95, 98 dopisz do `autoexec.bat`

  ```
  set CLASSPATH=c:/jamy/jamy_i_nory.jar
  ```

  a pod Windowsem 2000 Professional wejdź w Panel Sterowania -> System -> Zmienne środowiskowe i dodaj zmienną CLASSPATH o wartości `c:/jamy/jamy_i_nory.jar`.

  (TODO: jak tylko będzie mi się chciało wejść w Windowsa muszę sprawdzić czy ta ścieżka „Panel Sterowania -> ...” jest dobra)

3. Uruchamianie

3.1. Łączenie klienta i serwera

Serwer gry rozpoczyna pracę po uruchomieniu skryptu `jamy_server` z podkatalogu `bin/` naszego projektu. Składnia wywołania jest następująca:

```
jamy_server [--help] [--port PORT] [parametry określające interfejs]...
```

Serwer naszej gry może czekać na połączenia na dowolnym porcie TCP, domyślnie działa na porcie 1111 ale można to zmienić parametrem `--port`, np. polecenie

```
jamy_server --port 2323
```

uruchomi serwer nasłuchujący na porcie 2323.

Proces klienta rozpoczynamy wydając polecenie o składni

```
jamy_client <adres-serwera> [--help] [--port PORT] [parametry określające interfejs]...
```

gdzie `<adres-serwera>` to numer IP serwera (np. `123.456.123.456`) lub nazwa serwera rozpoznawana przez DNS (np. `ii.uni.wroc.pl`). Klient spróbuje połączyć się z działającym serwerem naszej gry na podanym komputerze. Domyślnie będzie próbował połączyć się z serwerem na porcie 1111, ale, podobnie jak w przypadku serwera, można to zmienić podając parametr `--port`, np.

```
jamy_client ii.uni.wroc.pl --port 2323
```

połączy się z serwerem działającym na komputerze `ii.uni.wroc.pl` nasłuchującym na porcie 2323.

3.1.1. Przykłady

Prosty przykład. Aby uruchomić serwer i klienta na tym samym komputerze można wydać polecenia

```
jamy_server
jamy_client 127.0.0.1
```

Zamiast ostatniego polecenia można równoważnie użyć

```
jamy_local_client
```

Przykład prosty ale z problemami. Jeżeli w powyższym przykładzie serwer przy próbie uruchomienia się odpowie *Address already in use* to port 1111 komputera jest już zajęty przez inny program. Musimy wtedy nakazać serwerowi naszej gry aby działał na innym porcie. Załóżmy że chcemy uruchomić nasz serwer na porcie 2323. Wpisujemy wtedy

```
jamy_server --port 2323
```

Aby połączyć się z naszym serwerem z tego samego komputera użyjemy polecenia

```
jamy_local_client --port 2323
```

Jeżeli nasz komputer jest widoczny w sieci pod adresem `my.computer.net` to ktokolwiek w sieci chcąc dołączyć się do naszej gry będzie mógł wydać polecenie

```
jamy_client my.computer.net --port 2323
```

Więcej przykładów? Można wykorzystać parametry `--port` aby na jednym komputerze uruchomić wiele wersji naszego serwera. Każdy taki serwer prowadziłby oddzielną grę w oddzielnym świecie. Gracze mogliby wtedy wybierać spośród kilku serwerów do których chcą się przyłączyć.

3.2. JamySingle

Poza serwerem i klientem w katalogu `single/` znajduje się jeszcze klasa `JamySingle`, niesieciowa wersja gry dla jednego użytkownika. Realizuje wszystkie potrzebne dla jednego gracza funkcje. Gramy wtedy sami z komputerem, czyli gra przestaje być grą typu MUD – staje się zwykłą tekstową przygodówką.

Aby zagrać w *JamySingle* należy wpisać polecenie o składni

```
jamy_single [--help] [parametry określające interfejs]...
```

Klasa `JamySingle` powstała przy okazji, w trakcie robienia serwera i klienta gry, ale zostaje już na stałe w programie bo dowodzi że kod „Jam i nor” jest elegancko zorganizowany i można w nim z łatwością oddzielić fragmenty zajmujące się komunikacją sieciową od mechanizmów gry.

3.3. Pomoc

Zarówno proces klienta jak i serwera można uruchomić z parametrem `--help`. Wypisują wowczas krótką instrukcję: jak należy je uruchomić oraz jakie są dozwolone parametry w linii poleceń. Parametry które nie zostały opisane w tej sekcji, oznaczane jako `[parametry określające interfejs]...`, są omówione w sekcji 5 o obsłudze.

4. Mechanika gry

Gracze wcielają się w przedstawiciela jednej z ras (ludzie, elfy, krasnoludy). Od wyboru rasy zależą początkowe zdolności postaci. Postacie mogą

- poruszać się po świecie gry używając wyznaczonych połączeń między lokacjami,
- operować przedmiotami (podnosić je z ziemi, upuszczać na ziemię, zakładać na siebie (zbroje), używać (np. użyć klucz do otwarcia drzwi) itp.,
- atakować inne postacie (innych graczy lub stwory sterowane przez komputer).

Gra nigdzie nie nakazuje graczom walczyć lub współpracować ze sobą nawzajem. Podobnie istoty sterowane przez komputer – mogą być groźne dla innych graczy (i innych istot sterowanych przez komputer) ale nie muszą być. Ideą gry jest aby do pewnego stopnia symulować świat rzeczywisty i pozwalać graczom na względną swobodę działań.

4.1. Postacie

Postacie mogą być kierowane przez ludzi lub przez serwer gry. Gracz patrząc na drugą postać nie jest w stanie powiedzieć czy jest ona sterowana przez serwer gry czy przez innego człowieka. Generalnie, gracze mogą się przed sobą ujawnić prowadząc rozmowę za pomocą komendy *powiedz* (patrz opis dostępnych komend gry 5). Chociaż zawsze pozostaje niepewność że wmontowaliśmy do naszej gry niezwykle zmyślną sztuczną inteligencję a'la Eliza....

Zdolności każdej z postaci opisane są przez poniższy zbiór cech:

- *życie* – określa odporność na obrażenia odniesione w walce. Każde zranienie postaci zabiera jej pewną ilość punktów życia. Gdy liczba punktów życia spada do zera postać umiera. Ekwipunek umierającej postaci jest upuszczany na ziemię w miejscu śmierci postaci.

  Aby mieć punkt odniesienia wyznaczamy sobie że typowy, zdrowy człowiek ma około 20 punktów życia.

- *inteligencja* – jest to ogólna zdolność rozeznania postaci. Im wyższa tym więcej uzyskujemy informacji o napotkanych postaciach – ich cechach oraz nastawieniu do innych uczestników gry. Z drugiej strony, istoty o wyższej inteligencji trudniej „rozpracować”.

  Przyjmujemy że przeciętny człowiek ma inteligencję 10, „normalny” człowiek ma +/- 3 a niepospolici mistrzowie/kretyni mogą mieć do +/- 6. Informatycy mogą mieć +/- 1024.

- *siła* – określa maksymalną wagę przedmiotów, które może nosić postać i ma decydujący wpływ na przebieg walki, zarówno podczas ataku jak i obrony. Wraz z siłą wzrasta ilość punktów życia odbieranych przy ataku oraz szansa na skuteczną obronę.

  Skala: taka jak inteligencja, tzn. przeciętny człowiek ma 10.

- *zręczność* – podobnie jak siła, ma wpływ na wynik walki. Ponadto decyduje o tym jak dużo czasu musi upłynąć zanim postać będzie mogła wykonać kolejną akcję. Im większa zręczność, tym krótszy jest ten czas.

  Skala: taka jak inteligencja, tzn. przeciętny człowiek ma 10.

- *obrona* – gdy postać zostanie zraniona wielkość zranienia jest zmniejszana o obronę postaci. W rezultacie postać o dużej obronie traci mniej punktów życia w wyniku zranienia.

  W przypadku mniej więcej normalnych, „człekopodobnych" ras (a prawdopodobnie takie będą najczęściej prowadzone przez graczy) obrona będzie wynikała głównie z rodzaju używanej zbroi. Natomiast specjalne rasy potworów (np. smoki) mogą posiadać również wrodzoną obronę (mówimy wtedy o *obronie skóry*).

4.2. Przedmioty

Każdemu przedmiotowi przypisany jest krótki opis oraz waga (w kilogramach). Przedmiot może należeć do jednej z poniższych klas:

- *Broń* – ma decydujący wpływ na przebieg walki, określa jak duże obrażenia może zadawać dana postać. Bez broni wielkość zadawanych obrażeń jest zdeterminowana przez rasę postaci, dla ras „człekopodobnych" (prowadzonych przez graczy) jest to zaledwie 1 punkt.

  Magiczna broń może też swobodnie modyfikować cechy postaci (pozytywnie lub negatywnie – np. przeklęty miecz może zmniejszać inteligencję używającego go gracza).

- *Zbroja* – zwiększa obronę postaci.

  Podobnie jak broń, magiczna zbroja może swobodnie modyfikować cechy gracza.

- *Pierścień* – może swobodnie modyfikować cechy gracza.

Postać może jednocześnie używać jednej broni, jednej zbroi i dwóch pierścieni. Ponadto może nosić w plecaku dowolne przedmioty, jednak magiczne przedmioty noszone w plecaku nie mogą modyfikować cech gracza, zbroja w plecaku nie daje obrony, a bronią w plecaku nie można zadawać obrażeń.

W grze występują też przedmioty które zawierają w sobie inne przedmioty, np. wszelkie plecaki. szkatułki, skrzynki itp. Waga takich przedmiotów jest zawsze podawana jako suma wagi opakowania (np. pustego plecaka) i wag przedmiotów w środku. Przedmioty takie można rozpakowywać poleceniem `rozpakuj`.

4.3. Modyfikatory

Wszystkie cechy postaci oraz przedmiotów są opisane przez modyfikatory. Pełen obraz zdolności postaci jest obliczany poprzez zsumowanie wszystkich aktualnych modyfikatorów działających na postać.

4.3.1. Modyfikator cech

Opisuje liczbę punktów którą dodaje się lub odejmuje od cech postaci. Może on być przypisany rasie (determinując początkowe cechy postaci), przedmiotom oraz fragmentom świata (np. na bagnie zręczność wszystkich postaci będzie zredukowana).

4.3.2. Modyfikator uderzenia

Jest podstawą do określania liczby punktów życia, które traci postać zraniona przy użyciu danej broni. Zapisujemy go jako $n\mathrm{k}m+c$. Za każdym razem gdy dana broń rani postać obliczamy ile punktów życia zabiera uderzenie przez zsumowanie n wyników rzutów kostką m-ścienną i dodanie stałego czynnika c.

Cecha ta charakteryzuje każdą broń oraz rasę (aby określić jakie obrażenie zadaje nieuzbrojona postać, tzn. walcząca pięściami, pazurami, mackami itd.)

4.4. Walka

Wynik walki, w której postać A atakuje postać B obliczany jest następująco:

$$A_{pkt} = \frac{A_{sila} + A_{zrecznosc}}{2} + (1k6)$$

$$B_{pkt} = \frac{B_{sila} + B_{zrecznosc}}{2} + (1k6)$$

Jeśli $A_{pkt} > B_{pkt}$ to mówimy że *cios trafia* i zadaje obrażenia wynikające z modyfikatora uderzenia broni używanej przez A zmniejszone o wartość obronną zbroi i skóry postaci B.

5. Obsługa

I klient i serwer działają na zasadzie przyjmowania wpisywanych poleceń i wyświetlania odpowiedzi, np. klient na polecenie `opisz mnie` odpowiada dokładnym opisem gracza a serwer w odpowiedzi na polecenie `gracze` wypisuje wszystkich graczy znajdujących się aktualnie w świecie. Ponadto niektóre informacje mogą być wypisywane bez ostrzeżenia – np. klient może w każdej chwili poinformować gracza że nowa postać weszła do pomieszczenia, a serwer może w każdej chwili poinformować że do gry przyłączył się nowy klient.

Podczas obsługi serwera możliwość wydawania komend nie będzie zbyt często wykorzystywana – w końcu serwer powinien działać sprawnie bez żadnego nadzoru. Natomiast podczas pracy z klientem gracz będzie zazwyczaj nieustannie wpisywał kolejne komendy i czytał odpowiedzi.

5.1. Obsługa klienta

Po uruchomieniu programu klienta, połączeniu z serwerem i stworzeniu postaci (lub wybraniu już istniejącej) użytkownik może rozpocząć grę. Do poruszania się w świecie „Jam i nor” służą poniższe komendy:

- *pomoc* – wyświetla listę dostępnych komend wraz z krótkim opisem.
- *pomoc komenda* – prezentuje opis konkretnej komendy.
- *koniec* – kończy grę.
- *idz nazwa-przejścia* – nakazuje postaci podążyć wskazanym przejściem.
- *opisz miejsce* – podaje szczegółowe informacje o miejscu w którym znajduje się postać, przebywających tam postaciach, dostępnych przejściach oraz leżących przedmiotach.
- *opisz mnie* – prezentuje wszystkie cechy kierowanej przez użytkownika postaci, jej wyposażenie oraz zawartość plecaka.
- *upusc przedmiot* – upuszcza przedmiot na ziemię.
- *podnies przedmiot* – podnosi przedmiot i umieszcza w plecaku.
- *zaloz przedmiot* – bierze wskazany przedmiot z plecaka lub z ziemi i zakłada (tzn. czyni go aktualnie używaną bronią, zbroją itd.). Dotychczas używany przedmiot (o ile taki jest) zajmuje miejsce nowo założonego (tzn. jest odkładany do plecaka lub na ziemię).
- *uzyj przedmiot* – używa przedmiotu w aktualnym pomieszczeniu, np. użycie klucza może otwierać i zamykać drzwi.
- *rozpakuj przedmiot* – rozpakowuje przedmiot w rodzaju plecaka, tzn. wyjmuje z niego wszystkie przedmioty i odkłada je na ziemię (jeżeli rozpakowywany przedmiot leży na ziemi) lub do własnego plecaka (jeżeli rozpakowywany przedmiot był w środku własnego plecaka).
- *atakuj postać* – atakuje wskazaną postać.
- *powiedz wiadomość* – przekazuje wiadomość wszystkim innym graczom w tej lokacji. W ten sposób w naszej grze mamy małego „komunikatora” (*instant messenger*). Zwracam uwagę że istoty sterowane przez komputer też czasem coś „mówią” m.in. aby zmylić gracza.
- *samobojstwo* – powoduje natychmiastową śmierć postaci.

Ponieważ gra jest ciągle rozwijana, powyższa lista może szybko stać się nieaktualna. Najlepszym źródłem informacji jest komenda *pomoc*.

Wersja niesieciowa (*JamySingle*) zachowuje się tak samo jak klient gry, tyle że nie łączy się z żadnym serwerem. Co znaczy tyle że wersja niesieciowa obsługuje identyczny zbiór komend co klient.

5.2. Obsługa serwera

Uruchomiony serwer gry udostępnia podobny interfejs co klient. Administrator serwera widzi rozmaite komunikaty diagnostyczne, przede wszystkim informujące o przyłączaniu i odłączaniu się graczy. Ponadto ma możliwość inspekcji i modyfikacji aktualnego stanu gry.

- *pomoc* – wyświetla listę dostępnych komend wraz z krótkim opisem.
- *koniec* – kończy pracę serwera, w łagodny sposób zamykając wszystkie połączenia.
- *gracze* – pokazuje listę wszystkich graczy.
- *stworzenia* – pokazuję listę wszystkich stworzeń sterowanych przez komputer.
- *mrozonki* – pokazuje listę wszystkich mrożonek, czyli zamrożonych postaci graczy którzy odłączyli się od serwera.
- *przedmioty* – pokazuje wszystkie przedmioty w grze leżące w lokacjach.
- *opisz postać* – opisz dokładnie postać.
- *zabij postać* – zabij postać.

Określenie *postać* oznacza powyżej, tak jak wszędzie, gracza lub postać sterowaną przez komputer (czyli istoty wypisywane przez polecenia *gracze* i *stworzenia*).

5.3. Różne interfejsy

Niniejszym zostanie wyjaśnione co znaczyły parametry oznaczone w sekcji 3 o Uruchamianiu jako

```
[parametry określające interfejs]...
```

Zarówno dla klienta, dla serwera jak i dla wersji niesieciowej dostępne są trzy rodzaje interfejsu. Każdy z nich zapewnia taką samą funkcjonalność – to którego interfejsu używać pozostaje do wyboru użytkownika.

- Interfejs konsolowy. Aby go używać należy uruchomić program z parametrem `--console`. To jest domyślny interfejs serwera (tzn. jeśli uruchomimy serwer bez żadnych parametrów to będzie to tak jakbyśmy uruchomili serwer z parametrem `--console`).

  Interfejs konsolowy działa czytając standardowe wejście i zapisując do standardowego wyjścia (więc *de facto* nie jest to nawet interfejs działający na konsoli – jest to interfejs używający tylko standardowo dostępnych dla programu deskryptorów plików).

- Interfejs okienkowy. Aby go używać należy uruchomić program z parametrem `--simple-swing`. To jest domyślny interfejs klienta i wersji niesieciowej.

  Interfejs okienkowy został napisany z użyciem komponentów Javy *Swing*.

- Bardziej stylowa wersja interfejsu okienkowego. Jest ładniejsza ale działa znacząco wolniej niż wersja prosta. Aby go używać należy uruchomić program z parametrem `--stylish-swing`.

5.3.1. Aplecik

Osoby znające Javę może zainteresować fakt że dostępna jest też wersja programu klienta jako aplet Javy. Ten aplet nie jest jeszcze zainstalowany na żadnej stronie w Internecie bo nie ma jeszcze nigdzie żadnego oficjalnego serwera gry, który działałby 24 godziny na dobę. Więc i tak trzeba sobie samemu uruchamiać serwer gry na własnym komputerze (albo uzywać wersji niesieciowej) więc równie dobrze można sobie samemu uruchamiać program klienta. Więc aplet nie ma w tym momencie zastosowania, jako że zasadniczą jego zaletą jest fakt że użytkownik nie musiałby nic ściągać ani ustawiać na swoim komputerze - wystarczyłoby żeby jego przeglądarka WWW obsługiwała Javę i aplet „sam by się uruchomił", połączył z odpowiednim serwerem itp.

Tym niemniej wersja testowa apletu (jeszcze nie w pełni skończona – na pewno skończę ją gdy będzie realny sens używania tego apletu) jest dostępna – jest to klasa *ClientApplet* w pakiecie `jamy_i_nory.client`. Jeśli ktoś jest zainteresowany to może ten aplet uruchomić w swojej przeglądarce WWW. Posiadając źródła programu można też stworzyć odpowiedni plik jar, `jamy_i_nory_client_applet.jar`, zawierający tylko klasy potrzebne temu apletowi.

5.3.2. Standardowa konsola Windows

Uwaga dla użytkowników standardowej konsoli pod Windows (nie chodzi tu o powłokę, command.com czy np. bash, ale o samo używanie standardowej konsoli Windows):

- Błąd w konsoli Windows sprawia, że w czasie gry nie widać na bieżąco wpisywanego z klawiatury tekstu. Widać go dopiero po naciśnięciu klawisza Enter (czyli już po zaakceptowaniu polecenia).

- Konsola Windows domyślnie używa czcionki zawierającej polskie znaczki w kodowaniu DOSa (Codepage 852). Jest to kodowanie niekompatybilne z Windowsowym standardem kodowania polskich znaczków (Windows-1250), nie wspominając nawet o zgodności ze standardem powszechnie przyjętym w Internecie i we wszystkich innych systemach operacyjnych (ISO-8859-2).

  Mógłbym się do tego dostosować wypisując polskie znaczki w kodowaniu Cp852 ale w ten sposób straciłbym możliwość współpracy z jakąkolwiek inną konsolą Windows (jak `rxvt` Cygwina). Fakt że 99,9% użytkowników Windows nie zna innych konsoli niż ta wbudowana w Windows niczego tu nie zmienia.

Podsumowując: pod Windows nie należy używać interfejsu konsolowego lub też używać innej konsoli niż ta dostarczona z systemem – jak już wspomniałem polecam konsolę `rxvt` Cygwina.