

Część 3 - Pierwsze pomieszczenie

W tej części kursu nauczymy się tworzyć pomieszczenie i wyświetlać wcześniej przygotowany obrazek.

Definujemy pomieszczenie

Jeżeli nie chcemy modyfikować silnika gry (co nie jest potrzebne przy jej tworzeniu), to definicja całej gry znajduje się w katalogu "game". W tym katalogu znajdują się definicje:

actions – akcje
areas – obszary
containers – kontenery
objects – obiekty
pictures – obrazki i palety
rooms – pomieszczenia

Nas interesuje katalog "rooms". Znajdują się w nim dwa pliki: rooms_def.h i rooms_def.c. Pliki .h to pliki nagłówkowe w języku C. Najogólniej mówiąc zawierają deklaracje, które są dzielone pomiędzy różnymi plikami .c

W naszym przypadku w pliku rooms_def.h musimy wyliczyć wszystkie pomieszczenia, które będą użyte w grze. Na razie potrzebujemy tylko jedno przykładowe pomieszczenie "Living room", któremu numer przypisujemy następująco (enum to typ wyliczeniowy automatycznie numerujący od 0 w górę):

```
enum {  
    ROOM_LIVING_ROOM,  
    ROOM_LAST_ROOM  
};
```

ROOM_LAST_ROOM to oznaczenie końca definicji. Na jego podstawie ustalana jest wielkość tablicy zawierającej definicje pomieszczeń. ROOM_LIVING_ROOM ma wartość 0, więc ROOM_LAST_ROOM to 1.

Czas na zdefiniowanie pomieszczenia. Otwieramy plik rooms_def.c i zmieniamy szablon pomieszczenia Room_XXX na:

```
// ROOM_LIVING_ROOM  
{  
    // Actions  
    room_living_room_on_draw, // on room draw  
    ACTION_NONE, // on room enter  
    ACTION_NONE, // on room exit  
    // Areas. The main room area should be the last.  
    {  
        OBJECT_NONE,  
    }  
},
```

room_living_room_on_draw to akcja (procedura), która zostanie wywołana w przypadku zdarzenia "on draw", czyli rysowania pomieszczenia. Rysowanie odbywa się automatycznie po wejściu do pomieszczenia, ale może też być wywołane po zmianach, które zaszły w pomieszczeniu lub gdy chcemy odświeżyć grafikę zamalowaną wyświetleniem inwentarza. Aktualnie definiowanie akcji **room_living_room_on_draw** zignorujemy i przejdźmy do definicji obrazka.

Definiujemy obrazek

Definicje obrazków i palet kolorów znajdują się w katalogu "game\pictures" w plikach pictures_def.c i pictures_def.h. Podobnie jak w przypadku pomieszczeń musimy najpierw wyliczyć używane obrazki i palety w pictures_def.h:

```
enum {  
    PICTURE_LIVING_ROOM,  
    PICTURE_LAST_PICTURE  
};  
  
enum {  
    PALETTE_LIVING_ROOM,  
    PALETTE_LAST_PALETTE  
};
```

Teraz możemy zdefiniować obrazek i jego paletę w pliku pictures_def.c.

Definicja obrazka:

```
// PICTURE_LIVING_ROOM,  
// Filename, width, height, palette_id, data pointer  
{ "livroom.mic", 160, 128, PALETTE_LIVING_ROOM, 0 },
```

Definicja palety:

```
// PALETTE_LIVING_ROOM,  
// Filename, filesize, data pointer  
{ "livroom.dli", 512, 0 },
```

Definicja obrazka składa się z nazwy pliku, szerokości obrazka, wysokości, identyfikatora palety i na końcu 0. Identyfikator palety to identyfikator zdefiniowanej palety. W tym przypadku używamy palety, którą właśnie zdefiniowaliśmy.

Nazwę pliku zmieniamy z "room1.mic0,0(160,128).mic" na krótką "livroom.mic", żeby nie miał z nią problemu Atari DOS.

Wielkość obrazka można odczytać z nazwy pliku wygenerowanej przez narzędzie gfx_slicer (wartość w nawiasach).

Definicja palety składa się z nazwy pliku, wielkości pliku palety i na końcu 0.

Pliki MIC i DLI należy skopiować pod zdefiniowanymi nazwami do katalogu "game\pictures". Z tego miejsca podczas kompilacji automatycznie zostaną skopiowane na obraz ATR,

Wyświetlamy obrazek po wejściu do pomieszczenia

Do wyświetlenia obrazka trzeba zdefiniować akcję. Definicje akcji są w "game\actions", w plikach actions_def.h i actions_def.c. Każda akcja to procedura formatu:

```
void nazwa_akcji(void);
```

My w pliku actions_def.h dodajemy:

```
void room_living_room_on_draw(void);
```

Zaś w pliku actions_def.c implementujemy tę akcję:

```
void room_living_room_on_draw(void)  
{  
    picture_draw(PICTURE_LIVING_ROOM, 0, 0);  
}
```

Kompilujemy, uruchamiamy i naszym oczom ukazuje się obrazek w tle pierwszego pomieszczenia:



W następnej części kursu nauczymy się jak definiować obszary, które można wybrać kursorem i poznamy zarządzanie obiektami gry.