

## Część 4 – obszary, obiekty i limity

W części 3 kursu wyświetliliśmy na ekranie obrazek pomieszczenia, ale na razie kursor nie miał zastosowania. W tej części to zmienimy.

### Definiowanie obszaru

Obszar jest określony poprzez jego pozycję i wielkość. Otwieramy z "game\areas" plik areas\_def.h i dodajemy obiekt będący pomieszczeniem:

```
////////////////////////////////////  
// IDs of areas  
////////////////////////////////////  
  
enum {  
    OBJECT_LIVING_ROOM,  
    AREA_LAST_AREA  
};
```

W pliku areas\_def.c definiujemy obszar, poprzez ustalenie jego pozycji (x,y) i jego wielkości w pikselach (width,height). W tym przypadku chcemy, żeby obszar dotyczył całości naszego ekranu.

```
// OBJECT_LIVING_ROOM  
{ 0,0, 160,160},
```

Pozycję na obrazku można sprawdzić wersją Windowsową, która na bieżąco pokazuje miejsce, w którym znajduje się kursor.

### Obiekty

W Adventure Studio każdy obszar jest obiektem i można mu przypisać właściwości. Mamy już zdefiniowany obszar, teraz trzeba zdefiniować odpowiadający mu obiekt.

Tu postępujemy inaczej niż zwykle i **nie dopisujemy** identyfikatora obiektu do game\objects\objects\_def.h. Nie dopisujemy, ponieważ ten identyfikator już przed chwilą nadaliśmy. Innymi słowy obszary są obiektami, ale mają dodatkowo pozycję ekranie określoną w pliku areas\_def.c.

Musimy jednak stworzyć definicję obiektu. W tym celu otwieramy plik "game\objects\objects\_def.c"

```
// OBJECT_LIVING_ROOM  
{  
    // Name, variable  
    "Living Room", AREA_ACTIVE,  
    // Actions on object  
    {  
        {"Examine", object_living_room_examine},  
        {"Leave", object_living_room_leave},  
    }  
},
```

W definicji obiektu ustalamy jego nazwę, w przypadku obszaru domyślny stan (AREA\_ACTIVE lub AREA\_INACTIVE), a następnie akcje, które można na obiekcie wykonać (nazwa, procedura akcji).

## Limity

Próba skompilowania tego kodu aktualnie zakończy się błędem, ponieważ nie mamy jeszcze zdefiniowanych akcji `object_living_room_examine` i `object_living_room_leave`, ale jest jeszcze jeden błąd:

error C2078: too many initializers dla linii `{"Leave", object_living_room_leave}`

Jest on spowodowany tym, że domyślnie są nadane ograniczenia na wielkości tablic, w celu zaoszczędzenia pamięci. Ograniczenia te są w pliku `game\config\game_cfg.h` i dla pustego szablonu z Tutorial Part 2 wyglądają następująco:

```
#define MAX_AREAS_IN_ROOM 1
#define MAX_OBJECT_ACTIONS 1
#define MAX_OBJECTS_IN_CONTAINER 1
#define MAX_PICTURE_OBJECT_MAPPINGS 1
```

`MAX_AREAS_IN_ROOM` to liczba obszarów, które występują w pomieszczeniu. Na pewno będziemy chcieli mieć ich kilka, więc ustalmy tę liczbę na 3.

`MAX_OBJECT_ACTIONS` to liczba akcji, które można wykonać na obiekcie. To właśnie przekroczenie tej wartości powodowało błąd. Ustalmy liczbę akcji na 2.

`MAX_OBJECTS_IN_CONTAINER` to liczba obiektów, które można umieścić w kontenerze jakim jest np. inwentarz gracza, szafa lub dowolny inny pojemnik występujący w grze. Dajmy na 2.

`MAX_PICTURE_OBJECT_MAPPINGS` na razie pozostawmy o limicie 1. Zostanie wyjaśniony przy zarządzaniu inwentarzem.

Czyli mamy:

```
#define MAX_AREAS_IN_ROOM 3
#define MAX_OBJECT_ACTIONS 2
#define MAX_OBJECTS_IN_CONTAINER 2
#define MAX_PICTURE_OBJECT_MAPPINGS 1
```

---

*UWAGA: Zwiększając limity należy wypełniać powstałą lukę "pustymi wartościami", np. Gdyby `MAX_OBJECT_ACTIONS` wynosiło 3, to definicję obiektu należałoby dopełnić za pomocą:*

```
// Actions on object
{
    {"Examine", object_living_room_examine},
    {"Leave", object_living_room_leave},
    {TEXT_NONE, OBJECT_NONE},
}
```

*Jest tak, ponieważ kompilator wypełnia brakujące pola wartościami 0, zaś nasze `OBJECT_NONE` to 0xFF. 0 to identyfikator pierwszego obiektu, więc pozostawienie tego kompilatorowi może przynieść nieplanowane zachowanie gry.*

---

## Akcje

Musimy stworzyć teraz akcje dla poleceń Examine i Leave. Robimy to jak poprzednio – dodajemy je w actions\_def.h:

```
void object_living_room_examine(void);
void object_living_room_leave(void);
```

i actions\_def.c:

```
void object_living_room_examine(void)
{
    console_print("An old fashioned living room.",true);
}

void object_living_room_leave(void)
{
    console_print("The kitchen room is not defined yet!",true);
}
```

Polecenie console\_print wypisuje tekst. Dodanie na końcu parametru true oznacza, że po pokazaniu tekstu program będzie czekał na naciśnięcie “fire”, po czym wyczyści konsolę tekstową.

## Przypisanie obszarów do pomieszczeń

Zdefiniowaliśmy już obszar i akcje obiektu, ale jeszcze nie przypisaliśmy tego obiektu do konkretnego pomieszczenia. W tym celu otworzmy plik game\rooms\rooms\_def.c i zmodyfikujmy definicję naszego pomieszczenia:

```
// 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_LIVING_ROOM,
        OBJECT_NONE,
        OBJECT_NONE,
    },
},
```

Maksymalną liczbę obszarów, które możemy przypisać do pomieszczenia zdefiniowaliśmy w MAX\_AREAS\_IN\_ROOM

Kompilujemy program i możemy wykonać dwie akcje na obszarze/obiekcie jakim jest cały ekran:



W kolejnej części zajmiemy się właściwościami obiektów.