Sceny DMX

Zapamiętywanie ulubionych kolorów lamp RGB jest podstawowym wymogiem sterowania kolorowym oświetleniem. Wychodząc na przeciw oczekiwaniom stworzyliśmy rozwiązanie, które umożliwia zapisywanie dowolnej liczby scen świetlnych DMX.

Czego nauczysz się w tym samouczku?

Po przeczytaniu tego samouczka będziesz umiał tworzyć interfejsy umożliwiające Ci zapisanie/wywoływanie dowolnej liczby scen DMX, zarówno z poziomu menu **Remote**, jak i wizualizacji **Display**. Zapisywanie scen zrealizowane będzie w zakładce **Logika**. Cały niezbędny kod został zawarty w skrypcie `dmxscene.lua`, który jest dostępny w aktualizacji komponentu `resources` w wersji `130311` dla modułu **Base**.

Kod skryptu można podejrzeć przechodząc do zakładki **Zasoby** > **Skrypty** oraz otwierając plik `dmxscene.lua`. Jeżeli z jakiegoś powodu w Twoim module nie ma skryptu, możesz go dodać manualnie po kliknięciu na przycisk **Dodaj**, a następnie wskazując plik na dysku lokalnym.

Skrypt jest też dostępny jako załącznik do niniejszego samouczka.

1. Deklaracja grup i scen

Aby rozpocząć tworzenie scen świetlnych należy w pierwszej kolejności zaimportować skrypt `dmxscene.lua` do zakładki **Logika**. Do importu skryptów służy polecenie `import '<skrypt>'`, w naszym przypadku `import 'dmxscene'`.

Aby zarządzanie scenami było bardziej intuicyjne skrypt będzie je automatycznie grupował wg podanej przez Ciebie nazwy. Dzięki temu będziesz mógł utworzyć wiele scen związanych np. z danym pomieszczeniem lub konkretnym źródłem oświetlenia. Do grupy można dodać dowolną liczbę scen. Tworzenie nowej grupy jest bardzo proste i ogranicza się do dodania w zakładce **Logika** pojedynczej linii kodu wg poniższego schematu:

`<zmienna> = dmxscene('<grupa>',<slot1>,...,<slotn>)`, gdzie `<zmienna>` to dowolna nazwa zmiennej, przez którą odwołujemy się do grupy. W dalszej części samouczka te zmienne będziemy nazywać obiektami. `<grupa>` to dowolna nazwa grupy. Po nazwie grupy należy podać adresy dowolnej liczby kanałów/slotów DMX, które mają zostać przypisane do danej grupy.

Przykłady gotowych deklaracji grup:

```
salon = dmxscene('salon',2,3,4,5,6,7,8,9,10)
```

```
salonsufit = dmxscene('sufit',2,3,4)
```

Te same kanały/sloty DMX mogą występować w kilku grupach. Dzięki takiemu ich wprowadzeniu możliwe będzie zapisywanie scen dla całych pomieszczeń, jak również niezależnych scen dla konkretnych źródeł światła.

Skrypt umożliwia zapisywanie wielu scen odnoszących się do tego samego zestawu kanałów DMX (do tej samej grupy). Służy do tego funkcja `save`. Do funkcji `save` należy przekazać, jako argument nazwę sceny, którą chcemy zapisać. Składnia jest następująca: `obiekt:save('nazwa')`, np `salon:save('relaks')`. Takie wywołanie funkcji spowoduje, że do sceny o nazwie `'relaks'` zostaną zapisane wartości kanałów DMX po-

dane w deklaracji grupy, w odniesieniu do powyższego przykładu będą to numery od 2 do 10.

Sceny zapisywane są w nieulotnej pamięci modułu **Base** (jako zmienne typu MEM). Każdorazowe wywołanie funkcji `save` nadpisuje scenę aktualnymi wartościami kanałów DMX, przypisanych do danej sceny. Wartości wszystkich scen można podejrzeć w zakładce **Stan**. Każda scena jest odrębną zmienną MEM o nazwie tożsamej z nazwą grupy i sceny. Wartość danej zmiennej MEM to poszczególne wartości kanałów wyrażone w kodzie heksadecymalnym. Ogólna forma zapisu jest następująca: `MEM.dmxs.<nazwa>.<scena>=<wartość>`.

Do wywoływania scen służy funkcja `restore`. Funkcja `restore` przyjmuje, jako argument nazwę sceny, którą chcemy wywołać. Składnia jest następująca: `obiekt:restore('nazwa')`, np `salon:restore('relaks')`.

Mając zadeklarowane grupy możemy przystąpić do tworzenia interfejsu sterującego scenami.

Przed przejściem do kolejnego kroku pamiętaj o zapisaniu zakładki **Logika**.

2. Remote

Interfejs w aplikacji **Remote** składać się będzie z co najmniej dwóch elementów:

Światło RGB – wybór barwy i jasności światła oraz kontrolki

Przycisk – przycisk do zapisywania i wywołania sceny.

Krótkie naciśniecie przycisku będzie wywoływało scenę, natomiast długie przyciśnięcie zapisze aktualne wartości kanałów DMX wskazanych w deklaracji grupy. Jeżeli dla danego zestawu kanałów DMX jest zadeklarowane kilka scen, wówczas należy dodać adekwatną liczbę **Przycisków**.

Procedura opisana poniżej odnosi się do deklaracji grupy `'salon'` przedstawionej w poprzedniej sekcji.

1. Dodaj nowy element **Światło RGB**, a następnie kliknij na nim dwukrotnie i uzupełnij jego właściwości. W kolejnych polach wpisz numery kanałów DMX przypisane do danej barwy światła, np. `DMX.2`, `DMX.3`, `DMX.4`. Do naszej grupy należą kanały od 2 do 10, dlatego ten krok należy powtórzyć 3 razy.
2. Dodaj **Przycisk**, kliknij na nim dwukrotnie i uzupełnij jego właściwości:
   - W komórce **Etykieta** wpisz opis przycisku.
   - Wybierz zakładkę **Przyciśnięcie**.
   - Kliknij na **Dodaj komendę** i w oknie, które się wyświetli w polu **Nazwa** wpisz: `C.LOGIC`, natomiast w polu **Wartość**: `<nazwa>:restore('<scena>')`, w naszym przypadku `salon:restore('relaks')`.
   - Przejdź do zakładki **Przytrzymanie** i powtórz poprzedni krok zmieniając zawartość pola **Wartość** na `<nazwa>:save('<scena>')`, w naszym przypadku `salon:save('relaks')`.

Przykładowy interfejs przedstawiono na poniższym obrazku:

3. Wizualizacja

Wywoływanie scen DMX można w prosty sposób zaimplementować na wizualizacji. Jako elementy sterujące należy tutaj użyć kontrolek typu **Przycisk**. Definicja kontrolki ogranicza się jedynie do przypisania **Przyciskom** polecenia i opcjonalnie etykiety.

Polecenie musi mieć następującą składnię: `LOGIC=obiekt:restore('scena')`, w naszym przypadku: `LOGIC=salon:restore('relaks')`.

```
--
-- DMX Scenes
--
-- Copyright 2013 DOMIQ Sp. z o.o.
--

function dmxscene(group,...)
      assert(group, "Group name required")
      for _,v in ipairs(arg) do
            assert(v >= 0 and v <= 255, "Invalid slot value "..v)
      end
      local t = {}

      function t:save(name)
            local val = {}
            for _,v in ipairs(arg) do
                  table.insert(val,
                        string.format('%02x',
                              math.ceil((get('DMX.'..v))*2.55)))
            end
            set(string.format('MEM.dmxs.%s.%s',group,name),
                  table.concat(val))
      end

      function t:restore(name)
            local val = get(string.format('MEM.dmxs.%s.%s',group,name))
            assert(val,"Missing scene "..name)
            local c = 0
            for v in string.gmatch(val,"(%x%x)") do
                  c = c + 1
                  command('C.DMX.'..arg[c],
                        math.floor(tonumber(v,16)/2.55+0,5))
            end
      end

      return t
end
```