Tworzenie własnego narzędzia graficznego

Skoro pobraliśmy już Processing, użyjmy go do tworzenia narzędzia graficznego, które pomaga w określeniu liczby rozwiązań równania. Zaczniemy od utworzenia siatki niebieskich linii, która wygląda jak papier w kratkę. Następnie utworzymy osie x i y, używając czarnych linii.

Ustawianie wymiarów wykresu

Aby utworzyć siatkę dla naszego narzędzia graficznego, musimy najpierw ustawić wymiary okna wyświetlania. W Processingu do określania szerokości i wysokości ekranu w pikselach możemy użyć funkcji `size()`. Domyślny rozmiar ekranu to 600 na 600 pikseli, ale na potrzeby naszego narzędzia graficznego będziemy tworzyć wykres, który zawiera wartości x i y z zakresu od −10 do 10.

Otwórzmy nowy plik w Processingu i zapiszmy go pod nazwą *grid.pyde*. Sprawdźmy, czy włączony jest tryb Pythona. Wpiszmy kod z listingu 4.6, aby zadeklarować zakres wartości x oraz y wyświetlanych na naszym wykresie.

grid.pyde

```
# ustawianie zakresu wartości x
xmin = -10
xmax = 10

# zakres wartości y
ymin = -10
ymax = 10

# obliczanie zakresu
rangex = xmax - xmin
rangey = ymax - ymin

def setup():
    size(600,600)
```

Listing 4.6. Ustawianie zakresu wartości x i y dla wykresu

Na listingu 4.6 tworzymy dwie zmienne `xmin` i `xmax`, do przechowywania minimalnej i maksymalnej wartości x w naszej siatce, a następnie powtarzamy ten proces dla wartości y. Później deklarujemy zmienną `rangex` dla zakresu x i zmienną `rangey` dla zakresu y. Obliczamy wartość `rangex`, odejmując `xmin` od `xmax` i robimy to samo dla wartości y.

Ponieważ nie potrzebujemy wykresu o wymiarach 600 na 600 jednostek, musimy zawęzić zakres współrzędnych, mnożąc współrzędne x i y przez współczynniki skali. Rysując wykres, musimy pamiętać, aby mnożyć wszystkie współrzędne x i współrzędne y przez te współczynniki skali – w przeciwnym razie nie zostaną one prawidłowo wyświetlone na ekranie. W tym celu dostosowujemy istniejący kod funkcji `setup()`, dodając wiersze z listingu 4.7.

grid.pyde

```
def setup()
    global xscl, yscl
    size(600,600)
    xscl = width / rangex
    yscl = height / rangey
```

Listing 4.7. Skalowanie współrzędnych przy użyciu współczynników skali

Zaczynamy od zadeklarowania globalnych zmiennych `xscl` oraz `yscl`, których użyjemy do skalowania ekranu. Zmienna `xscl` oraz zmienna `yscl` reprezentują odpowiednio współczynnik skali x oraz współczynnik skali y. Na przykład współczynnik skali x wynosiłby 1, gdybyśmy chcieli, aby zakresem x było 600 pikseli, czyli cała szerokość ekranu. Natomiast gdybyśmy chcieli, aby ekran obejmował zakres od −150 do 150, zastosujemy współczynnik skali x równy 2, uzyskiwany przez podzielenie `width` (600) przez zakres `rangex` (300).

W naszym przypadku możemy obliczyć współczynnik skalowania, dzieląc 600 przez zakres x, który wynosi 20 (od −10 do 10). Dlatego współczynnik skali musi wynosić 30. Od tej pory musimy mnożyć wszystkie nasze współrzędne x i y przez współczynnik skali 30, aby były one odpowiednio wyświetlane na ekranie. Dobra wiadomość jest taka, że to komputer będzie wykonywać dzielenie i skalowanie. Jednak musimy pamiętać, aby użyć `xscl` oraz `yscl` podczas rysowania wykresu.

Rysowanie siatki

Skoro ustawiliśmy już odpowiednie współrzędne dla naszego wykresu, możemy narysować linie siatki, jak na papierze w kratkę. Kod zawarty w funkcji `setup()` zostanie uruchomiony jednokrotnie. Następnie tworzymy nieskończoną pętlę przy użyciu funkcji o nazwie `draw()`. Funkcje `setup()` i `draw()` to wbudowane funkcje biblioteki Processingu i ich nazwy muszą pozostać niezmienione, aby szkic mógł być uruchamiany. Dodajmy kod z listingu 4.8 w celu utworzenia funkcji `draw()`.

grid.pyde

```
# ustawianie zakresu wartości x
xmin = -10
xmax = 10

# zakres wartości y
ymin = -10
ymax = 10

# obliczanie zakresu
rangex = xmax - xmin
rangey = ymax - ymin

def setup():
    global xscl, yscl
    size(600,600)
    xscl = width / rangex
    yscl = height / rangey

def draw():
    global xscl, yscl
    background(255) # biały
    translate(width/2,height/2)
    # cyjanowe linie
    strokeWeight(1)
    stroke(0,255,255)
    for i in range(xmin,xmax + 1):
        line(i,ymin,i,ymax)
```

Listing 4.8. Tworzenie niebieskich linii siatki dla wykresu