

# cofeecam

Bartosz Pieńkowski, Barnaba Turek

19 października 2011

## Streszczenie

Celem serii projektów z przedmiotu Grafika Komputerowa jest utworzenie prostego silnika graficznego (i zapoznanie się od strony graficznej z podstawowymi zadaniami grafiki komputerowej, takimi jak rzutowanie, eliminacja elementów zasłoniętych i cieniowanie).

Zdecydowaliśmy się zaimplementować projekt w języku coffeescript.

Pierwsza część projektu polega na utworzeniu wirtualnej kamery. Kamera ma obsługiwać zmianę pozycji, ogniskowej i kąta, w którym jest skierowana.

## 1 Wirtualna kamera

### 1.1 Struktura projektu

Projekt podzielony jest na pliki .coffee, z których każdy zostaje skomplikowany do pliku .js (javascript). Zawartość każdego pliku wynikowego jest opakowana w anonimową funkcję, aby zniwelować ryzyko konfliktu nazw z innymi skryptami.

Zmienne i funkcje potrzebne do komunikacji pomiędzy modułami dostępne są za pomocą zmiennej coffecam.

Nie wszystkie obiekty przedstawione na diagramie 1 są w istocie klasami - z wyjątkiem klas **Camera** i **Controller** nie potrzebowaliśmy przechowywać żadnego stanu, a tylko logicznie podzielić udostępniane funkcje.

Plik *site* jest swojego rodzaju funkcją main, która uruchamia się po załadowaniu dokumentu przez przeglądarkę.

### 1.2 Działanie kamery

Obiekt kamery przechowuje macierz transformacji, początkowo ustawioną na macierz jednostkową (z dokładnością do znaku, ze względu na skrętności układów)[1]. Macierz transformacji obliczana jest na nowo po każdym ruchu kamery. Obliczenie nowej macierzy polega na pomnożeniu aktualnej macierzy przez żądaną transformację.

Dostępne transformacje to przesunięcia i obroty kamery (Transformacje utworzyliśmy na podstawie wykładu[2]).

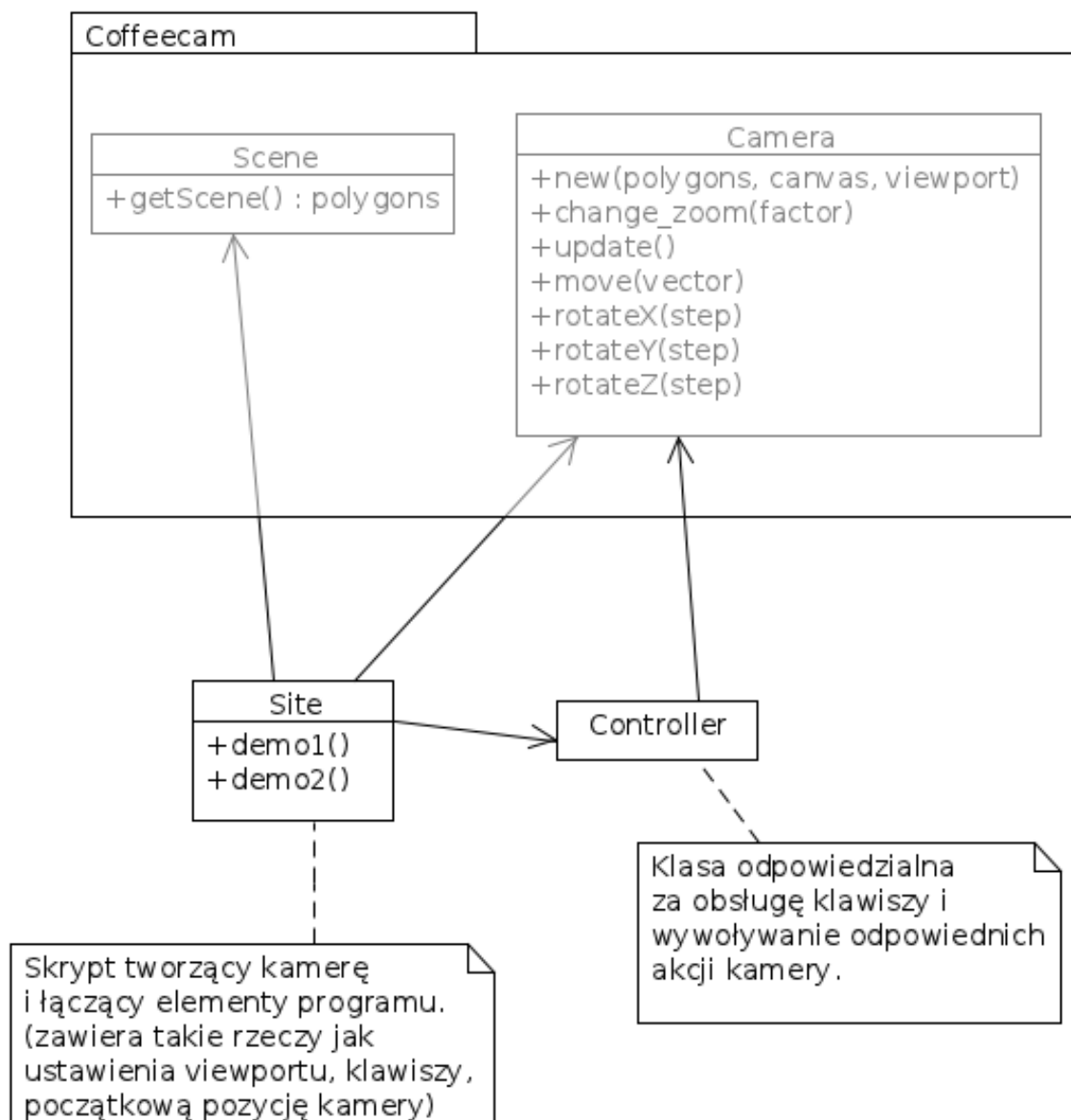
Po transformacji macierz transformacji mnożona jest przez macierz rzutowania (Obliczaną kiedy kamera jest tworzona i przy zmianie ogniskowej)[1]. Następnie każdy z punktów sceny mnożony jest przez macierz wynikową. Przygotowanie każdej nowej klatki realizuje metoda *update()* kamery.

### 1.3 Scena

Obiekty, które kamera wyświetla to wielokąty. Kamera zakłada, że punkty obiektu podane są w sposób uporządkowany.

Scena tworzona jest z czworokątów, ze względu na to, że rysujemy głównie prostopadłościany.

Rysunek 1: Struktura projektu



## 1.4 Pozostałe informacje o projekcie

Statyczne części projektu (strony, style) są tworzone za pomocą frameworku staticmatic, na podstawie źródeł (W językach HAML i SASS).

W projekcie wykorzystaliśmy javascriptową bibliotekę Sylvester wspierającą operacje na macierzach i wektorach.

Kod źródłowy projektu (wraz z aktualną dokumentacją) dostępny jest na serwisie Github (głównie katalog /src/coffee).

Projekt jest dostępny do testowania na następujących serwerach:

- Serwer 1<sup>1</sup>
- Volt<sup>2</sup> (Niekoniecznie aktualna wersja)

## Literatura

- [1] Song Ho Ahn, OpenGL Projection Matrix, [http://www.songho.ca/opengl/gl\\_projectionmatrix.html](http://www.songho.ca/opengl/gl_projectionmatrix.html)
- [2] Dariusz Sawicki, Grafika komputerowa i wizualizacja, [http://wazniak.mimuw.edu.pl/index.php?title=Grafika\\_komputerowa\\_i\\_wizualizacja](http://wazniak.mimuw.edu.pl/index.php?title=Grafika_komputerowa_i_wizualizacja)

---

<sup>1</sup><http://barnex.mo00.com>

<sup>2</sup><http://volt.iem.pw.edu.pl/~pienkowb/>