Instrukcja dla programisty

Program był pisany tak, aby dopisywanie kolejnych rozszerzeń nie sprawiało problemów. Tym samym projekt został podzielony na wiele niezależnych od siebie modułów:

- RayTracer
  - App - moduł startowy programu wywoływany przez entry point znajdujący się w pliku `Main.hs`
  - AABB - implementacja struktury bryły brzegowej, opisującej prostopadłościan na obiekcie bądź zbiorze obiektów
  - AABBTree - wariacja na temat drzewa ósemkowego wykorzystująca strukturę zaimplementowaną w module `AABB`. Służy głównie do przyspieszania obliczeń
  - Camera - implementacja kamery oraz przekształceń świat <-> kamera.
  - Light - moduł opisujący dostępne światła. Dodawanie nowych źródeł światła wymaga edycji tego pliku.
  - Material - moduł opisujący dostępne materiały. Dodawanie nowych typów powierzchni wymaga edycji tego pliku.
  - Primitive - moduł opisujący dostępne obiekty **oraz** typ algebraiczny i funkcje do obsługi zderzenia się promienia z obiektem. Dodanie nowych prymitywów wymaga edycji tego pliku.
  - Ray - moduł implementujący interfejs promienia (półprostej)
  - Scene - moduł opisujący scene
  - SceneParser - moduł odpowiedzialny za parsowanie pliku sceny i zwracanie obiektu z modułu `Scene`. Ten moduł musi być edytowany aby udostępnić użytkownikowi nowy materiał/światło/rodzaj obiektu.
  - Vector - moduł implementujący podstawowe operacje na wektorach

Kod w większości jest okomentowany natomiast w wielu miejsach zostało to pominięte gdyż jest on wystarczająco deskryptywny.

W kodzie znajdują się dwie stałe które można edytować:

- numSamples (`RayTracer/App.hs`) - ilość sampli generowanych na jeden piksel obrazu
- getMaxRayDeph (`RayTracer/Ray.hs`) - maksymalna ilość odbić (inaczej: max. głębokość rekursji) jaką może wykonać pojedynczy promień.

Przykładową rzeczą, którą można relatywnie łatwo dodać jest światło słoneczne (nie zostało to zaimplementowane w oryginalnym projekcie przez skończenie się czasu) tworzące, w przeciwieństwie do światła punktowego, miękkie cienie. Przez fakt, że liczenie intensywności światła wymaga policzenia wielu losowych promieni od powierzchni źródła w stronę obiektu potrzebujemy generatora liczb pseudolosowych. Najprostszym sposobem jest użycie monady stanowej i stworzenie w oparciu o nią monady RNG. Po zmianie typu głównej funkcji generującej kolory pikseli trzeba także zmodyfikować typ funkcji liczącej kolor dla konkretnego światła w module `Light`. Tu najprościej wykorzystać transformator monady Maybe: `MaybeT`. Teraz została sama implementacja światła która jest już prosta. Dodatkowo, dzięki dokonanym zmianą otrzymaliśmy dostęp do generatora liczb losowych dzięki czemu możemy zimplementować ciekawsze materiały wykorzystujące metodę Monte-Carlo

Samo dodawanie nowych materiałów/źródeł światła/primitywów jest w większości przypadków bardzo proste i ogranicza się do rozszerzenia typu algebraicznego w odpowiednim module oraz napisania właściwej implementacji dla nowej rzeczy (przez rozszerzenie istniejącej funkcji). Wynika to z faktu, że główne funkcje w modułach są "generyczne" względem konkretnego typu algebraicznego, przez co z użyciem pattern-matching'u możemy napisać rozłączne implementacje dla różnych konstruktorów.