Project Progress Report

Specifications

- 3+5, etc.
- Functions from R to R
- Functions in 3D
- Implicit functions
- Operations with function
- Zoom in and out
- Rotate viewpoint

+ Parametric curves and surfaces
+ Complex functions
+ Derivatives and integrals

CRC Models

Classes

Entities

- Expression
- Axes
- Viewpoint

Use Cases

- ExpressionCreator
- RendererUseCase
- ViewpointEditor

Controllers/Presenters

- ExpressionReader
- Renderer

Gateways

- UserInterface

List of CRC Cards

Entities:

- Expression
  - NumberExpression
  - FunctionExpression
  - VariableExpression
  - OperatorExpression
- Axes/Space
- Viewpoint

Controllers/Presenters:

- ExpressionReader
- Renderer

Use Cases:

- ExpressionCreator
- RenderUseCase
- ViewpointEditor

UserInterface

- CommandLineRunner
- GUIRunner (for later)

Expression (Abstract)

Responsibilities:

- Stores various items (Number, Variable, Operator, etc.)
- Provides method signature for evaluate method (to be overridden by subclasses)

Collaborators:

- NumberExpression
- FunctionExpression
- VariableExpression
- OperatorExpression
- ExpressionCreator

Expression

NumberExpression

Responsibilities:

- Extends Expression class
- Handles Number inputs
- Stores numbers (calls the constructor for the Parent class Expression to store the item and initiate the item variable)
- Evaluates its expression (override from abstract method)

Collaborators:

- ExpressionCreator

Expression

OperatorExpression

Responsibilities:

- Input to initializer: an operation and 2 expressions
- Stores the operation and 2 expressions
- Overrides evaluate from parent class Expression.
- Can evaluate the expression for appropriate operations (addition, subtraction, multiplication, division, exponents). Returns type double.

Collaborators:

- ExpressionCreator

VariableExpression

- Extends Expression class
- Accepts and stores strings “x”, “y”, and/or “z”.
- Overrides evaluate by checking a map to determine which value has been assigned to it.

Collaborators:

Expression (abstract parent)

FunctionExpression (abstract)

Responsibilities:

- Extends Expression class
- For functions like cos, sin, sqrt, etc.
- Handles multiple inputs of any kind (Number, Operator, etc.)
- Stores the name of the function (calls the constructor for the Parent class Expression to store the item and initiate the item variable)
- Stores an Expression for how a functions should be evaluated (e.g. if a user defines f(x) = x^2 + 5, then a FunctionExpression will be created that is named f and stores the expression “x^2 + 5”.)

Collaborators:

- Expression (Abstract Parent Class)
- ExpressionCreator

Axes/Space

Responsibilities:

- The Euclidean space in which graphs are graphed
- Handles scale of the axes
- Stores a list of all expressions that have been input

Interacts with:

- Expression

Viewpoint

- Stores attributes of viewpoint e.g. zoom level, scale, orientation, etc.

Interacts with:

Renderer

ViewpointEditor

Responsibilities:

- Changes attributes of viewpoint e.g. zoom level, scale, orientation, etc.

Interacts with:

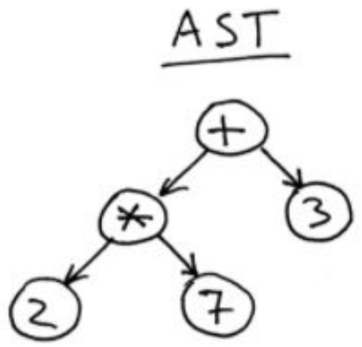
Viewpoint

RendererUseCase

AST

Topics

- Dihan: entity classes
- Ted: ExpressionReader
- Rishibh: ExpressionCreator



Example of AST

ExpressionReader

Responsibilities:

Takes string input and converts to a list that expression creator can use to create an expression with

E.g.

“x^2 + 5” -> [“x”, “^”, “3”, “+”, “5”]

“cos(x + y)” -> [“cos”, “(“, “x”, “+”, “y”, “)”]

Interacts with:

ExpressionCreator

UserInterface (when upgrading to graphical interface)

ExpressionCreator

Responsibilities:

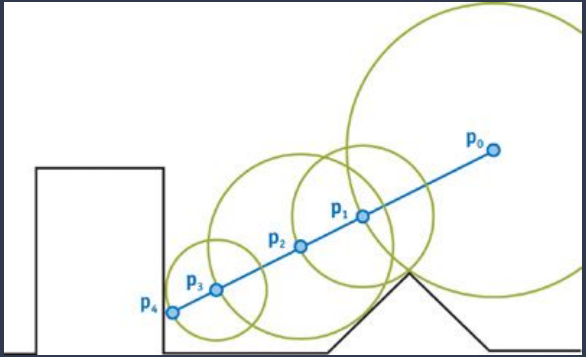
Converts a list into an Expression, where the elements corresponds to a ‘unit’ of information that implements Abstract Syntax Trees

Inputs are of the format that ExpressionReader ‘spits’ out

Interacts with:

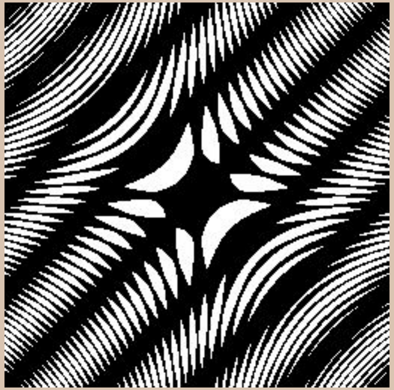
Expression (and its subclasses)

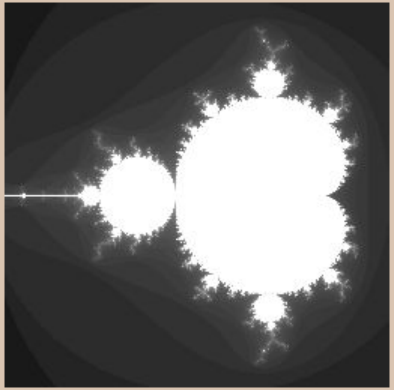
Graphics



Evaluatable

Evaluatable3D





RendererUseCase

Responsibilities:

“Main method”, calls renderer with axes + viewpoint etc

Is bridge between core renderer(s) and UI layers

Pass parameters to Renderer

Interacts with:

- Viewpoint
- Renderer
- Axes

Renderer (interface)

Responsibilities:

- Generates image from pixel array
- Save image

Interacts with:

- RendererUseCase

UserInterface

Responsibilities:

- Accept user input and passes it to ExpressionReader
- Although we will be starting out with a command line interface, we ideally want to transition to a graphical user interface

Interacts with:

ExpressionReader

Scenario Walkthrough

A user inputs an expression like f(x) = x^2 + 5 . First an empty Backend.Axes object is created. This is passed on to Backend.ExpressionReader . The Backend.ExpressionReader converts x^2 + 5 into a list [“x”, “^”, “2”, “+”, “5”] and passes that on to Backend.ExpressionCreator . Backend.ExpressionCreator will convert all the variables, operators and numbers into appropriate Expressions (Backend.NumberExpression , Backend.VariableExpression , Backend.OperatorExpression) and combines them into an Abstract Syntax Tree, then create a Backend.FunctionExpression that stores the function name “f” along with the expression that it evaluates. Backend.ExpressionReader will then return this Backend.FunctionExpression which will then be added to the Backend.Axes object.

The Renderer takes an Backend.Axes (currently just an Backend.Expression), a Viewpoint , and other parameters for the desired image (such as size, scales of axes, etc), then generates an int[] array representing pixels of an image. RendererUseCase will pass the parameters of the image to Renderer which will then convert the pixel array into an image and save it to a file.