##### 1 Drawing Application Extension

##### 1.1 Additional Preliminaries

Save your solutions for this problem to a file named draw.rkt.

Run the following expression (you must have required [extras.rkt](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt)) to check that your file is properly named and is in the proper directory:

(check-location "12" "draw.rkt")

Add these additional provides at the top of your file (below the requires), so that we can test your solution:

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) INITIAL-WORLD)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) handle-mouse)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) Shape<%>)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) get-world-shapes)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) create-triangle)

##### 1.2 Problem Description

Extend your [drawing program from set11](http://www.ccs.neu.edu/course/cs5010sp15/set11.html#%28part._set11~3adraw%29) with triangles. If your code is well-organized, this should be a straightforward addition.

Specifically, add the capability to draw equilateral triangles. When drawing the triangle, the location of the initial "button-down" determines the triangle center. The location of the "button-up" determines one of the corners, with the radius of the triangle being the distance from any corner to its center. During "drag" (before "button-up"), the program should display a triangle silhouette, just like before, anchored at its center, with a corner at the current mouse location (and the other corners should be equidistant to the center), rotating accordingly to mouse "move" events.

Grabbing one of the triangle corners with the select tool should resize and rotate the triangle with its center anchored.

UPDATE 2015-04-07: The “control region” is a square containing anything within 5 pixels of the corner point itself, just as it was for the rectangles in set11.

Grabbing the body of the triangle with its select tool should smooth drag it without changing its orientation.

UPDATE 2015-04-08: The signature and purpose statement for the required create-triangle function is:

|  |
| --- |
| ; create-triangle : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> Shape<%> |
| ; Creates a triangle Shape<%> object from a center |
| ; posn and one arbitrary corner. |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) (create-triangle center corner) ...) |

UPDATE 2015-05-07: We have released a [triangle library](http://www.ccs.neu.edu/course/cs5010sp15/files/triangle.rkt) (right-click and choose save-as) with the following helper functions.

|  |
| --- |
| ; Posn functions ——————————————– |
|  |
| ; subtract-posn : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) |
| ; GIVEN: Two points, considered "positive" and "negative", respectively. |
| ; RETURNS: The difference between those points, componentwise. Considering the |
| ;    result as a vector, the vector points from the negative point to the |
| ;    positive one. |
|  |
| ; add-posn : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) |
| ; GIVEN: Two points |
| ; RETURNS: The sum of those points, componentwise. |
|  |
| ; distance : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> Real |
| ; GIVEN: Two points, |
| ; RETURNS: the scalar distance between them. |
|  |
| ; Triangle functions ————————————– |
|  |
| ; in-triangle? : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> Boolean |
| ; GIVEN: One "candidate" point and three corner points, |
| ; RETURNS: true iff the candidate is inside the triangle defined |
| ;   by the last three points. |
|  |
| ; render-equilateral-triangle : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) String String [Image](http://www.ccs.neu.edu/course/cs5010sp15/predatadefs.html#%28tech._image%29) -> [Image](http://www.ccs.neu.edu/course/cs5010sp15/predatadefs.html#%28tech._image%29) |
| ; GIVEN: A center Posn, an arbitrary corner Posn, an opacity String, a color |
| ;    String, and a background Image, |
| ; RETURNS: An Image like the background but with an equilateral triangle placed |
| ;    onto it, such that its center is as given, one of its corners is at the |
| ;    given corner, and its opacity and color as given. (The "center" is the |
| ;    point that's equidistant from the three corners.) |
|  |
| ; compute-corners : [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) -> (list [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29) [posn](http://docs.racket-lang.org/htdp/index.html#%28def._%28%28lib._lang%2Fposn..rkt%29._posn%29%29)) |
| ; Computes the corners of an equilateral triangle from a center and one corner. |
| ; The given corner is also included in the output. |

##### 2 Interpreter, OO Style

##### 2.1 Additional Preliminaries

Save your solutions for this problem to a file named eval.rkt.

Run the following expression (you must have required [extras.rkt](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt)) to check that your file is properly named and is in the proper directory:

(check-location "12" "eval.rkt")

Add these additional provides at the top of your file (below the requires), so that we can test your solution:

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) mk-Program%)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) mk-Expr%)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) Program<%>)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) Expr<%>)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) Result<%>)

##### 2.2 Problem Description

The [interpreter pattern](http://en.wikipedia.org/wiki/Interpreter_pattern) is a common OO design pattern used to implement (unsurprisingly) a variety of interpreters. Revise your [solution from set09](http://www.ccs.neu.edu/course/cs5010sp15/set09.html#%28part._set09~3aeval%29) to use[interface](http://docs.racket-lang.org/reference/createinterface.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._interface%29%29)s and [class](http://docs.racket-lang.org/reference/createclass.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._class%29%29)es.

Your interpreter should be able to handle the same kinds of expressions as in set09. A different [class](http://docs.racket-lang.org/reference/createclass.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._class%29%29) should represent each possible expression. You should also have a class forPrograms. (You may choose whether to convert definitions, ie Defs, to [class](http://docs.racket-lang.org/reference/createclass.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._class%29%29)es or to leave them as [define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29)s. Be ready to justify your design decision during your code walk.)

Each [class](http://docs.racket-lang.org/reference/createclass.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._class%29%29) should implement one (or more) of the following interfaces. You may extend the given or add new [interface](http://docs.racket-lang.org/reference/createinterface.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._interface%29%29)s if you wish. However, all the requested methods must be implemented.

Make sure to follow the [OOP Style](http://www.ccs.neu.edu/course/cs5010sp15/style.html#%28part._style~3aoop%29) guide.

|  |
| --- |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) Result<%> |
| ([interface](http://docs.racket-lang.org/reference/createinterface.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._interface%29%29) () |
| ; app : ListOf<Result<%>> ListOf<Def> -> Result<%> |
| ; If Result is a lambda, applies it to given arguments, else errors. |
| app)) |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) Expr<%> |
| ([interface](http://docs.racket-lang.org/reference/createinterface.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._interface%29%29) () |
| ; eval : ListOf<Def> -> Result<%> |
| ; Evaluates the expression to a Result<%>, |
| ; in the context of the given definitions. |
| eval |
| ; subst : Result<%> Var -> Expr<%> |
| ; Replaces references to the given var with the given Result<%>. |
| ; Does not replace x with r if x occurs in the body of a lambda |
| ; that shadows x. |
| ; WHERE: the given Result<%> has no unbound variables. |
| subst |
| ; to-expr : -> Expr |
| ; Returns a representation of this expression as a non-object Expr. |
| to-expr |
| ; to-expr/no-var : -> ExprNoVar |
| ; Returns a representation of this expression as a non-object ExprNoVar. |
| to-expr/no-var |
| ; expr=? : Expr<%> -> Boolean |
| ; Returns true if this expression is structurally equivalent to the given Expr<%> |
| expr=?)) |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) Program<%> |
| ([interface](http://docs.racket-lang.org/reference/createinterface.html#%28form._%28%28lib._racket%2Fprivate%2Fclass-internal..rkt%29._interface%29%29) () |
| ; eval : -> ListOf<Result<%>> |
| ; Evaluates expressions in the program to Result<%>s |
| ; WHERE: A function may be called before it is defined. |
| ; WHERE: The results have the same order as their original expressions in the program. |
| eval)) |

In addition, define and [provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) the following functions. If you are familiar with OO patterns, these can be viewed as [static factory methods](http://www.drdobbs.com/jvm/creating-and-destroying-java-objects-par/208403883) (which are essentially just regular functions).

|  |
| --- |
| ; mk-Program% : Program -> Program<%> |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) (mk-Program% p) ...) |
| ; mk-Expr% : Expr -> Expr<%> |
| ([define](http://docs.racket-lang.org/reference/define.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define%29%29) (mk-Expr% e) ...) |

The Program and Expr data definitions are the same as in set09.

**Note**: Since we are using full Racket, [define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29)s must have the #:transparentannotation. In other words, use the following struct definitions for mk-Expr% and other functions.

|  |
| --- |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) def (name params body) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) arith (op args) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) bool (op args) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) cmp (op args) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) if-exp (test branch1 branch2) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) call (fn args) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) lam (params body) #:transparent) |
| ([define-struct](http://docs.racket-lang.org/reference/define-struct.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._define-struct%29%29) lam/no-var (body) #:transparent) |