Racket Assignment #1: Getting Acquainted with Racket/DrRacket + LEL Sentence Generation

Abstract

This assignment will delve into the Racket programming language and the DrRacket interface. I will do this by constructing a LEL sentence generator. This is an activity that will help me understand the syntax and the mechanics in the Racket programming language.

Code for the LEL sentence generator

```
#lang racket
;------------------------------------------
; LEL sentence generator with helper PICK,
; several applications of APPEND, several
; applications of LIST, and one use of MAP
; with a LAMBDA function.

(define(pick list)
   (list-ref list(random(length list)))
)

(define(noun)
   (list(pick'(robot baby toddler hat dog)))
)

(define(verb)
  (list(pick '(kissed hugged protected chased hornswoggled)))
)

(define(article)
  (list(pick '(a the)))
)

(define(qualifier)
  (pick '((howling)(talking)(dancing)
          (barking)(happy)(laughing)
          () () () () () ()
          )
  )
)

(define(noun-phrase)
  (append(article)(qualifier)(noun))
)

(define(sentence)
  (append(noun-phrase)(verb)(noun-phrase))
)

(define(ds);display a sentence
  (map
    (lambda(w)(display w)(display " "))
    (sentence)
  )
  (display ""); an artificial something
)
```

Demo for the LEL Sentence Generator

```
Welcome to DrRacket, version 8.7 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (pick '(red yellow blue))
'blue
> (pick '(red yellow blue))
'blue
> (pick '(red yellow blue))
'red
> (pick '(red yellow blue))
'yellow
> (pick '(Racket Prolog Haskell Rust))
'Racket
> (pick '(Racket Prolog Haskell Rust))
'Prolog
> (pick '(Racket Prolog Haskell Rust))
'Prolog
> (noun)
'(toddler)
> (noun)
'(dog)
> (noun)
'(dog)
> (noun)
'(hat)
> (verb)
'(hugged)
> (verb)
'(kissed)
> (verb)
'(protected)
> (verb)
'(hugged)
> (article)
```

```
'(the)
> (article)
'(the)
> (article)
'(a)
> (article)
'(the)
> (qualifier)
'()
> (qualifier)
'(happy)
> (qualifier)
'(howling)
> (qualifier)
'(happy)
> (qualifier)
'(laughing)
> (qualifier)
'()
> (qualifier)
'(talking)
> (qualifier)
'(barking)
> (qualifier)
'()
> (qualifier)
'(laughing)
> (qualifier)
'(howling)
> (qualifier)
'()
> (qualifier)
'()
> (qualifier)
'(talking)
> (qualifier)
'()
> (qualifier)
'(barking)
> (noun-phrase)
'(a talking hat)
> (noun-phrase)
'(a toddler)
> (noun-phrase)
'(the dancing dog)
> (noun-phrase)
'(a robot)
> (noun-phrase)
'(the dog)
> (noun-phrase)
'(a baby)
> (noun-phrase)
```

```
'(a baby)
> (noun-phrase)
'(the toddler)
> (sentence)
'(the toddler hugged the baby)
> (sentence)
'(the hat kissed a toddler)
> (sentence)
'(a toddler hornswoggled a talking toddler)
> (sentence)
'(a dancing hat protected a baby)
> (sentence)
'(a dancing hat hornswoggled the toddler)
> (sentence)
'(the dancing hat protected the barking dog)
> (sentence)
'(the robot chased the toddler)
> (sentence)
'(a baby kissed the happy toddler)
> (ds)
the dog kissed a toddler
> (ds)
the barking hat chased a dancing robot
> (ds)
a robot hugged a baby
> (ds)
the howling toddler chased the hat
> (ds)
a happy robot hugged a baby
> (ds)
a hat hugged a robot
> (ds)
the dog hornswoggled a dancing baby
> (ds)
the robot protected the dancing baby
> (ds)
a robot kissed the howling baby
> (ds)
a howling toddler hugged the barking dog
> (ds)
a barking toddler chased a happy dog
> (ds)
a howling baby hugged the dancing baby
>
```