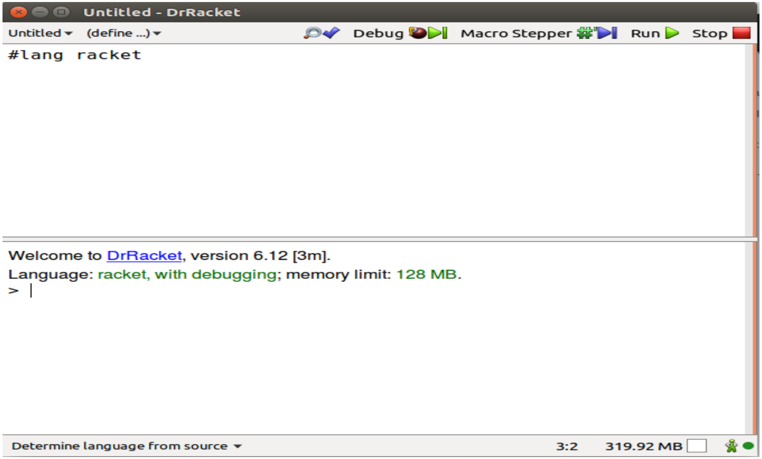
Lecture 2: Our Language

15CSE402 :: SICP

R Sreekumar

August 3, 2020

Dr. Racket



```
#lang racket
```

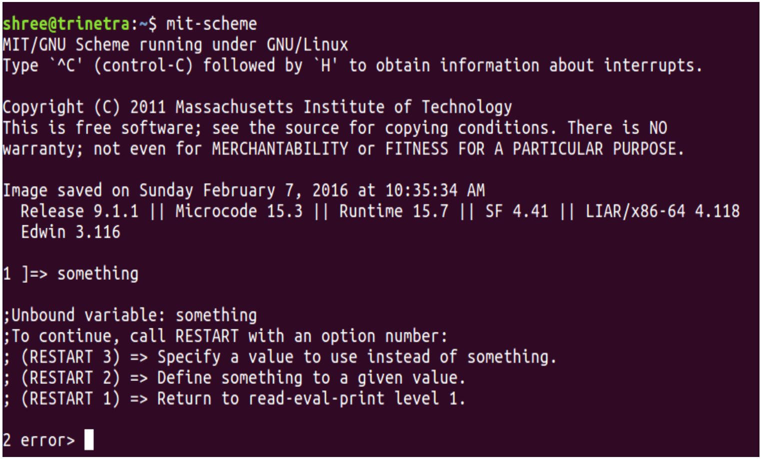
```
Welcome to DrRacket, version 6.12 [3m].
Language: racket, with debugging; memory limit: 128 MB.
> 
```

Expressions

```
> 142
142
>
```

```
> something
. . something: undefined;
 cannot reference undefined identifier
>
```

Error in MIT-SCHEME



```
shree@trinetra:~$ mit-scheme
MIT/GNU Scheme running under GNU/Linux
Type `^C' (control-C) followed by `H' to obtain information about interrupts.

Copyright (C) 2011 Massachusetts Institute of Technology
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Image saved on Sunday February 7, 2016 at 10:35:34 AM
  Release 9.1.1 || Microcode 15.3 || Runtime 15.7 || SF 4.41 || LIAR/x86-64 4.118
  Edwin 3.116

1 ]=> something

;Unbound variable: something
;To continue, call RESTART with an option number:
; (RESTART 3) => Specify a value to use instead of something.
; (RESTART 2) => Define something to a given value.
; (RESTART 1) => Return to read-eval-print level 1.

2 error> 
```

Combinations

```
> (+ 34 65)
99
> (* 5 99)
495
>
```

```
> (+ 2 4 6 8)
20
>
```

Expressions (contd.)

```
> (+ (* 3 (+ (* 2 4) (+ 3 5))) (+ (- 10 7) 6))
45
```

```
> (+ (* 3
        (+ (* 2 4)
           (+ 3 5)))
     (+ (- (- 10 7)
           6)))
45
```

Primitive Procedures

```
> (define size 2)
> size
2
> (* 4 size)
8
```

```
> (define (square x)
    (* x x))
> (square 10)
100
```

Conditional Expressions and Predicates

```
> (define (absolute x)
    (contd ((> x 0) x)
          ((= x 0) 0)
          ((< x 0) (- x))))
> (absolute -3)
  3
```

```
> (define (absolute x)
    (if (< x 0) (- x)
        x))
> (absolute -3)
3
```

Conditionals (contd)

```
(and (> x 0) (> y 0))
(or (> x 0) (> y 0))
(not (> x 0))
```

Few more

So far, we have seen

- define
- contd
- if
- and, or, not

We have few more procedures

- lambda
- let
- quote
- list
- cons, car, cdr

This is more than enough for us explain the computational process. We don't need any more.

End of Syntax

End of Syntax

Thus completes our programming language.

A word of Caution

- Both Dr.Racket Documentation[1] and MIT-Scheme Reference Manual[2] consists of lots of primitive.
- We need not worry about that for two reasons:
    - This syntax is more than enough to describe procedure
    - A new procedure (as in the references), can be easily created with the above primitives.

  However, interested students are encouraged to go through these references.

[1] `https://docs.racket-lang.org/`

[2] `https://www.gnu.org/software/mit-scheme/documentation/stable/mit-scheme-ref.pdf`