A bit of a Review: Stop Light Problem

```
CLIPS> (defrule red-light
         (light red)
        =>
         (printout t "Stop" crlf))

CLIPS>(defrule green-light
         (light green)
      =>
         (printout t "Go" crlf))
CLIPS> (rules)
red-light
green-light
For a total of 2 defrules.
CLIPS> (agenda)
CLIPS> (assert (light red))
<Fact-1>
CLIPS> (agenda)
0      red-light: f-2
For a total of 1 activation.
CLIPS> (run)
Stop
CLIPS> (assert (light green))
<Fact-1>
CLIPS> (run)
Go
```

Two rules are created.
With no facts there is nothing on the agenda.
When a fact is presented, the corresponding rule is on the agenda.
Running activates the rule

Two Patterns in a rule

```
CLIPS> (clear)
CLIPS> (defrule take-a-drive
     (status driving)
     (light green)
=>
     (printout t "Go" crlf))
CLIPS> (defrule take-a-walk
     (status walking)
     (walk-sign walk)
=>
     (printout t "Go" crlf))
CLIPS> (rules)
take-a-walk
take-a-drive
For a total of 2 defrules.
CLIPS> (assert (status walking))
<Fact-0>
CLIPS> (agenda)
CLIPS> (assert (light green))
<Fact-1>
CLIPS> (agenda)
CLIPS> (assert (walk-sign walk))
<Fact-2>
CLIPS> (agenda)
0      take-a-walk: f-0,f-2
For a total of 1 activation.
CLIPS> (run)
Go
```

Two rules each with two conditional elements are defined. The appropriate rule does not appear on the agenda until the both patterns are present.

Strategy: Which Rule Fires First

Depth Strategy
New Activations are placed on the Agenda
After
Those with higher Salience
Before
Those of equal Salience or less

Salience
A user defined (range -10000 to +10000) priority to a rule
Default: 0
Set within rule using: **(declare (salience 100))**

Strategy: Many Strategies Available

Clips offers several strategies available:

- depth
- breadth
- LEX
- MEA
- complexity
- simplicity
- random

The default is **depth**.

defacts: Initialization of Facts

```
CLIPS> (clear)
CLIPS>
 (deffacts walk "Some Facts about walking"
     (status walking)
     (walk-sign walk))
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-1     (status walking) CF 1.00
f-2     (walk-sign walk) CF 1.00
For a total of 3 facts.
CLIPS> (clear)
CLIPS> (facts)
CLIPS>
```

The facts of **deffacts** are re-initiated after reset. This is one of the differences between (reset) and (clear).

(reset)

```
CLIPS> (defrule red-light
         (light red)
=>
         (printout t "Stop" crlf))
CLIPS> (assert (animal duck))
<Fact-0>
CLIPS> (defrule quack
          (animal duck) => (printout t "Quack" crlf))
CLIPS> (deffacts walk "Some Facts about walking"
          (light red)
          (walk-sign walk))
CLIPS> (agenda)
0      quack: f-0
For a total of 1 activation.
CLIPS> (facts)
f-0     (animal duck) CF 1.00
For a total of 1 fact.
CLIPS> (rules)
red-light
quack
For a total of 2 defrules.
CLIPS> (reset)
CLIPS> (agenda)
0      red-light: f-1
For a total of 1 activation.
CLIPS> (clear)
CLIPS> (agenda)
CLIPS> (facts)
CLIPS> (rules)
CLIPS>
```

The reset function:

- Removes Existing Facts from the fact-list, which may remove activated rules from the agenda.
- Asserts (initial-fact)
- Asserts facts from deffacts

Eliminate deffacts

```
CLIPS> (deffacts walk "Some Facts about walking"
     (light red)
     (walk-sign walk))
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-1     (light red) CF 1.00
f-2     (walk-sign walk) CF 1.00
For a total of 3 facts.
CLIPS> (undeffacts walk)
CLIPS> (reset)
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
For a total of 1 fact.
```

Dribble

(dribble-on *filename*)

(dribble-on *filename*)

Debugging: Stepping Through Firings

```
CLIPS> (defrule red-light
         (light red) => (printout t "Stop" crlf))
CLIPS> (defrule walking
             (status walk)  => (printout t "Walking" crlf))
CLIPS> (deffacts initial
                 (light red)
                 (status walk))
CLIPS> (agenda)
0      walking: f-2
0      red-light: f-1
For a total of 2 activations.
CLIPS> (facts)
f-0     (initial-fact) CF 1.00
f-1     (light red) CF 1.00
f-2     (status walk) CF 1.00
For a total of 3 facts.
CLIPS> (run)
Walking
Stop
CLIPS> (reset)
CLIPS> (run 1)
Walking
CLIPS> (run 1)
Stop
```

Seeing what is matched

```
CLIPS> (defrule take-a-vacation
    (work done)
    (money plenty)
    (reservations made)
=>
    (printout t "Let's Go" crlf))
CLIPS> (assert (work done))
<Fact-0>
CLIPS> (matches take-a-vacation)
Matches for Pattern 1
f-0
Matches for Pattern 2
 None
Matches for Pattern 3
 None
Partial matches for CEs 1 - 2
 None
Partial matches for CEs 1 - 3
 None
Activations
 None
```

```
CLIPS> (assert (money plenty))
<Fact-1>
CLIPS> (matches take-a-vacation)
Matches for Pattern 1
f-0
Matches for Pattern 2
f-1
Matches for Pattern 3
 None
Partial matches for CEs 1 - 2
f-0,f-1
Partial matches for CEs 1 - 3
 None
Activations
 None
```