Rule Book

Definitions:

1. **Whitespace character:** A whitespace character can be:-
   a. a space character
   b. a tab character
   c. a carriage return character
   d. a new line character
   e. a vertical tab character
   f. a form feed character

2. **Digit:** Any character from 0 - 9

3. **Alphabet:** Any character from a-z or A-Z

4. **Alphanumeric:** Any character from a-z or A-Z or 0-9

5. **Word character:** Any character from a-z or A-Z or 0-9 or _

Salience:

1. Begin with the keyword “salience” or “priority” (without quotes)
2. followed by zero or more whitespace characters
3. followed by a colon (:)
4. followed by zero or more whitespace characters
5. followed by priority/salience value which is:-
   5.1. an **optional** minus (-) sign
   5.2. followed by any combination of one or more digits
6. followed by zero or more whitespace characters
7. **(Optional)** End with a comma (,) or a semicolon (;)

Examples:
   1. salience : 12
   2. priority : 0;
   3. salience : -2;

Agenda Group:

1. Begin with the keyword “agenda-group” or “agendaGroup” (without quotes)
2. followed by zero or more whitespace characters
3. followed by a colon (:)
4. followed by zero or more whitespace characters
5. followed by name of agenda which:-
   - 5.1. Begins with an alphabet or a dollar ($) or an underscore (_)
   - 5.2. followed by any combination of alphanumeric or dollar ($) or underscore (_) characters

     or
   - 5.1. Begins with a double quote (“)
   - 5.2. followed by any combination of zero or more characters (except double quote)
   - 5.3. Ends with a double quote (“)

     or
   - 5.1. Begins with a single quote (‘)
   - 5.2. followed by any combination of zero or more characters (except single quote)
   - 5.3. Ends with a single quote (‘)
6. followed by zero or more whitespace characters
7. (Optional) End with a comma (,) or a semicolon (;)

Examples:

1. agenda-group : ‘a&b’
2. agendaGroup : $abc
3. agenda-group : “a^b”;

Auto-focus:

1. Begin with the keyword “auto-focus” or “autoFocus” (without quotes)
2. followed by zero or more whitespace characters
3. followed by a colon (:)
4. followed by zero or more whitespace characters
5. followed by auto-focus value which can be:
   - 5.1. “true” (without quotes)

     or
   - 5.1. “false” (without quotes)
6. followed by zero or more whitespace characters

7. (Optional) End with a comma (,) or a semicolon (;)

Examples:

1. auto-focus : true
2. autoFocus : false;

TODO /*Hash constraint:

1. Begin with a curly brace "{" (without quotes)
2. followed by a key value pair which is
   - 2.1. followed by a key which is:
     - 2.1.1. (Optional) a single quote (') or a double quote (")
     - 2.1.2. followed by an optional dollar ($)
     - 2.1.3. followed by any combination of one or more word characters
     - 2.1.4. (Optional, according to step 2.1.1) End with a single quote (') or a double quote (") corresponding to step 2.1.1
   - 2.2. followed by zero or more whitespace characters
   - 2.3. followed by a colon (:)
   - 2.4. followed by zero or more whitespace characters
   - 2.5. followed by a value which is
     - 2.5.1. (Optional) a single quote (') or a double quote (")
     - 2.5.2. followed by an optional dollar ($)
     - 2.5.3. followed by any combination of one or more word characters
     - 2.5.4. (Optional, according to step 2.5.1) End with a single quote (') or a double quote (") corresponding to step 2.5.1
3. followed by zero or more spaces
4. followed by */

Rule Condition:

1. Begin with an optional dollar ($)
2. followed by any combination of one or more word characters
3. followed by zero or more spaces
4. followed by a colon (:)
5. followed by zero or more spaces
6. followed by any combination of one or more word characters
7. followed by any combination of zero or more characters (except newline or line terminator)

Examples:

1. $m : Message
2. s : String s == ‘hello’

Predicate Expression:

1. Begin with any combination of one or more word characters
2. followed by zero or more spaces
3. followed by an opening parentheses “(“ (without quotes)
4. followed by any combination of zero or more characters (except newline or line terminator)
5. End with a closing parentheses “)” (without quotes)

Note:- The first word (word in rule 1) should be “not” or “or” (without quotes).

1. If it is “not”, there should only be a single Rule Condition in the parentheses.
2. If it is “or”, there can be multiple Rule Conditions inside the parentheses separated by a comma (,). There should not be any comma after the last Rule Condition.

Examples:

1. not(s : String)
2. or(m : Message m.message =~ /^hello/, m:Message m.message =~ /world$/)

Note:- There can be any number of Predicate Expressions and Rule Conditions separated by semicolon (;) in a “when” block, with an optional semicolon (;) after the last one.

When block:

1. Begin with the keyword “when” (without quotes)
2. followed be zero or more whitespace characters
3. followed by an opening curly bracket “{“ (without quotes)
4. End with a closing curly bracket “}” (without quotes)

Note:- All constraints(rule conditions and predicate expressions) of a rule are written inside the ‘When block’.

Then block:

1. Begin with keyword “then” (without quotes)
2. followed by zero or more whitespace characters
3. followed by an opening curly bracket “{“ (without quotes)
4. End with a closing curly bracket “}” (without quotes)

Note:- Actions corresponding to a rule are written inside the ‘Then block’.

Block comment:

1. Begin with “/*” (without quotes)
2. End with “*/” (without quotes)

Note:- Block comments cannot be nested inside one another.

Examples:

1. /* This is a comment */

Define block:

1. Begin with the keyword “define” (without quotes)
2. followed by zero or more whitespace characters
3. followed by name of define block which:
    - 3.1. Begins with an alphabet or a dollar ($) or an underscore (_)
    - 3.2. followed by any combination of zero or more alphanumeric or dollar ($) or underscore (_) characters
4. followed by zero or more whitespace characters
5. followed by an opening curly bracket “{“ (without quotes)
6. End with a closing curly bracket “}” (without quotes)

Note:- ‘Define block ‘ consists of key-value pairs similar to javascript hash object. The values can be functions or other javascript objects. A special key ‘constructor’ can be used to define the constructor of the class corresponding to the define block.

Examples:

1. define Result { }
2. define Message {

```
        message : '',
        constructor : function(mesg) {
                this.message = mesg;
        }
}
```

Global declaration:

1. Begin with the keyword "global" (without quotes)
2. followed by zero or more whitespace characters
3. followed by variable name which:
   3.1. Begins with an alphabet or a dollar ($) or an underscore (_)
   3.2. followed by any combination of zero or more alphanumeric or dollar ($) or underscore (_) characters
   3.3. followed by zero or more whitespace characters
4. followed by an equals (=) sign
5. End with a semicolon (;)

Examples:

1. global x = 5;
2. global list = [1, 2, 3, 4];

Function declaration:

1. Begin with the keyword "function" (without quotes)
2. followed by zero or more whitespace characters
3. followed by function name which:
   3.1. Begins with an alphabet or dollar ($) or underscore (_)
   3.2. followed by any combination of zero or more alphanumeric or dollar ($) or underscore (_) characters
   3.3. followed by zero or more whitespace characters
4. followed by an opening parentheses "(" (without quotes)
5. followed by an optional "Arguments List"(similar to those in javascript)
6. followed by a closing parentheses ")" (without quotes)
7. followed by zero or more whitespace characters

8. followed by an opening curly bracket “{“ (without quotes)
9. End with a closing curly bracket “}” (without quotes)

Examples:

1. function sum(a,b) {
            return a+b;
   }
2. function doSomething() {
   }

Rule block:

1. Begin with the keyword “rule” (without quotes)
2. followed by zero or more whitespace characters
3. followed by rule name which:
   3.1. Begins with an alphabet or a dollar ($) or an underscore (_)
   3.2. followed by any combination of zero or more alphanumeric or dollar ($) or underscore (_) characters

   or

   3.1. Begins with a double quote (“)
   3.2. followed by any combination of zero or more characters (except double quote)
   3.3. Ends with a double quote (“)

   or

   3.1. Begins with a single quote (‘)
   3.2. followed by any combination of zero or more characters (except single quote)
   3.3. Ends with a single quote (‘)
4. followed by zero or more whitespace characters
5. followed by an opening curly bracket “{“ (without quotes)
6. Ending with a closing curly bracket “}” (without quotes)

Examples:

1. rule “Rule 1” { }
2. rule Rule1 { }

Note:- Salience, Agenda group, Auto focus, When block and Then block are written inside the Rule block.

Rule Actions:

Actions of a rule have to be written in pure javascript. The aliases defined in when block can be used as to refer to the instances of facts. There is no ‘modify’, ‘assert’ or ‘retract’ function. Facts can be modified by direct reference to them by their aliases.