JSON Object Notation

- RTSON - Rational Type Safe Object Not
- Simple yet versatile file forma
- Addresses missing consistent
- Adds support for Primitive &

- GeSON Is a superset of JSON
- It adds single line and multilin
- It supports a bunch of primitiv
- It supports dates & times, rege

- GSON supports also complex function calls and named obje
- To an extent, it even supports

Example

```
// this is a comment
@unsigned someInt:12
someBigDecimal:123366558877.3255D
@SafeConfig(lastLoaded:2024-06-12)
page(simple:[1]){
    load(file:"sub-config.tson")
    positions:[(1,2),(3,4)]
    params:{
        name:"name",
        dimension: percentOf(3)
    }
}
resultMatrix:[
    1+i5, i1 ;
```

Literals an example

- `true, false`

Strings an example

- "a" 'b' `c` simple strings

```
\"""
    block string
\"""
'''b'''
```c```
```
```

Regex an example

- /ab*/

Date is an example

- 2024-02-05-10-10 //CHECK ME

Numbers an example

```
// regular numbers
1 (int), 12L (long), 1.0 (double), 1445.0f
// big numbers
1245LL (big int), 1.2336LL (big decimal)
// can include '_'
123_100 // int
// can have unit that starts with '_' or '
123_100L_amp // int with unit 'amp'
10%   // int with unit '%'
23%g  // int with unit '%g'
```

Numbers have a range

```
0u1_12 // this is a byte (8bits) with value
0u2_12 // this is a short (16bits) with val
0u4_12 // this is an int (32bits) with valu
          // this is equivalent to 12
0u8_12 // this is an int (64bits) with valu
          // this is equivalent to 12L
```

Binary Numbers

- 1010_0101 (int)
  0u1x01 (byte)
  0u2x0101 (short)
  0u4x0101 (int)
  0u8x0101 (long)
  0101_0101_01_0101LL (big integer)
  0101_amp (int with unit 'amp')

Octal Number example

```
4556 (int)
0u1o71 (byte)
0u2o71 (short)
0u4o71 (int)
0u8o71 (long)
71LL (big integer)
71LL_amp (big integer with unit 'amp')
71% (int with unit '%')
```

Hexadecimal Numbers

- 4556 (int)
  0u1x71 (byte)
  0u2x71 (short)
  0u4x71 (int)
  0u8x71 (long)
  71LL (big integer)
  71LL_amp (big integer with unit 'amp')
  71% (big integer with unit '%')

Complex Numbers

- 1.2+i15.0 (double complex)
  1.2f+i15.0f (float complex)
  1.2LL+i15.0f (big decimal complex)

Palmis is an example

- `name:"myObject"`

Uplist is an example

- (name:"myObject", enabled, 3)

Objects can key value pairs

```
{
    name:"myObject"
    count:12
    enabled:true
}
```

Or any other constructs

Arrays & Matrices

- `[ 1, 2, 3, 4]`

  `[ 1  2  3  4]`

  `[ 1  2`

  `  3  4]`

- These are matrices (use ';' for r

- `[ 1, 2 ; 3, 4]`

  `[ 1  2 ; 3  4]`

Function example

- `myFunction(arg1,arg2)`

Named Object example

```
myObject{
        item1,
        item2
}
```

- and this is a named object wit

```
myObject(color:red){
        item1,
```

Nested Array example

```
myArray[
        item1,
        item2
]
```

Annotations are metadata that

```
@unsigned int:12
@Entity object{
    @Id column:string
}
```

- Stream can include any binary data
  (identified by the '^' char)
- ^SAFE_TAG[
  ANY DATA
  ]SAFE_TAG

Talhakeyosulah@gmail.com