FUNCTION PARSER REFERENCE

KATUN

BLOCKCHAIN

CONTENT

1. OBJECTIVE

In order to provide a tool that simplifies and ensures the right way to input information in SmartContract features, function parser enableS to analyze the syntax of a string with the nomenclature of the smart contract methods/functions to be invoked and parse them into Katun BlokChain set of objects to be used in invoke/execute smart contracts events.

2. USAGES

- Syntax

  function:<function_name>:['\n'|<space>|<blank>]
  arg:<data_type>:{<value>}:['\n'|<space>|<blank>]
  …
  arg:<data_type>:{<value>}:

  arg = element to start the definition of an argument belonging to a function.

  data_type =
  <boolean|byte|char|double|float|integer|long|object|string>
  value = [any value introduced by the user compatible with the datatype defined in the argument].

- Supported types
  boolean, byte, char, double, float, integer, long, object, string

Notice that the **arg** elements correspond only to the arguments to be input in the method/function to Its execution as we can see in the next example

- Example

  The next code is a simple java class to be used as a SmartContract. This contract has 2 methods (**function**(s)), sum (with arguments (**arg**) "a" int/integer type and "b" int/integer type and message (no arguments).

```
public class test{
        int sum(int a, int b){
```

```
            return a+b;
        }
        String message(){
            return "this is a test";
        }
}
```

The nomenclature to generate the execution/invoke string might be as follows:

function:sum:
arg:integer:{10}:
arg:integer:{10}:
function:message:

Expecting to get 2 katun.smartcontract.ui.common.entities.UIFunction objects with 2 katun.smartcontract.ui.common.objects.Attribute objects with name arg[0..n] (in this case arg[0-1]) and datatype as native int, and no arg, respectively.