
pySFeel

Release 1.3.4

Jan 01, 2021

Contents:

1	the sFeelParse function	1
2	Data Types	3
3	List and Context Filters	5
4	Assignment and Variable names	7
5	Usage	9
6	Built-in Functions	11
7	Indices and tables	15
	Python Module Index	17
	Index	19

the sFeelParse function

```
class pySFeel.SFeelParser
```

```
    sFeelParse (text)
```

Parse S-FEEL text

This routine parses the passed text, which must be valid S-FEEL

Parameters **param1** (*str*) – The S-FEEL text to be parsed

Returns

(status, value)

'status' is a list of any parsing errors.

'value' is the Python native value of the parsed S-FEEL text.

For an assignment statement the 'value' will be the Python native value assigned to the named variable.

For all other expressions the 'value' will be the Python native value of the S-FEEL expression.

Return type tuple

CHAPTER 2

Data Types

pySFeel converts S-FEEL data into the nearest equivalent Python native data type.

S-Feel data type	Python native data type
number	float
string	str
boolean	bool
days and time duration	datetime.timedelta
year and months duration	int
time	datetime.time
date	datetime.date
date and time	datetime.datetime
List	list
Context	dict

Literal strings (@"PT5H") are also implemented as bare strings (PT5H). @"PT5H" > @"PT4H" can be written as PT5H > PT4H and would return True

List and Context Filters

pySFeel supports List and Context filters with one deliberate deviation from the standard - the key word 'item' in a Context filters is **not** optional.

`[{x:1,y:2},{x:2,y:3}][x=1]` is not valid (as `x=1` is either True or False)
`[{x:1,y:2},{x:2,y:3}][item x=1]` is valid and will return `{x:1,y:2}`.

Similarly, `fred.y` is **not** the 'y' filter on the List of Contexts named 'fred' (as `fred.y` is a valid name).
However `(fred).y` is the 'y' filter on the list of Contexts named `fred`.

CHAPTER 4

Assignment and Variable names

There's one extension - an assignment operator (<-) which will store a Python internal value against a named variable. Named variables are valid in S-FEEL expressions in pySFeel.

```
fred <- 7 bill <- 9 fred = bill
```

This will return False

```
fred <- [{x:1,y:2},{x:2,y:3}] (fred).y
```

This will return [2,3]


```
import pySFeel
parser = pySFeel.SFeelParser()
sfeelText = '7.3 in [2.0 .. 9.1]'
(status, retVal) = parser.sFeelParse(sfeelText)
if 'errors' in status:
    print('With errors:', status['errors'])
```

- retVal will be True
- The dictionary 'status' will have the key 'errors' if you have errors in your sfeelText.
- status['errors'] is a list of strings. It may help in diagnosing your S-FEEL syntax errors.

CHAPTER 6

Built-in Functions

pySFeel has support all the standard FEEL built-in functions with some differences because pySFeel is a Python implementation.

Name(parameters)	Parameter Domain	pySFeel implementation notes
date(from)	date string	Uses dateutil.parser - strict ISO format is not required. pySFeel will convert a string that is in ISO format into a datetime.date, datetime.time or datetime.datetime
date(from)	date and time	Truncates datetime.datetime to datetime.date
date(year,month,day)	year,month,day are numbers	
date and time(date,time)	date is a date or date time; time is a time	
date and time(from)	date time string	Uses dateutil.parser - strict ISO format is not required. pySFeel will convert a string that is in ISO format into a datetime.date, datetime.time or datetime.datetime
time(from)	time string	Uses dateutil.parser - strict ISO format is not required. pySFeel will convert a string that is in ISO format into a datetime.date, datetime.time or datetime.datetime
time(from)	time, date and time	
number(from,grouping, separator,decimal separator)	string,string, string	pySFeel does the appropriate rounding, but the returned value is a float. Trailing zeros are not retained.
string(from)	non null	
duration(from)	duration string	

Continued on next page

Table 1 – continued from previous page

Name(parameters)	Parameter Domain	pySFeel implementation notes
years and months duration(from, to)	both are date or both are date and time	
not(negand)	boolean	
substring(string,start, position,length?)	string,number	
string length(string)	string	
upper case(string)	string	
lower case(string)	string	
substring before (string,match)	string,string	
substring after (string,match)	string,string	
replace(input,pattern, replacement,flags?)	string	
contains(string,match)	string	
starts with(string,match)	string	
ends with(string,match)	string	
matches(input,pattern, flags?)	string	
split(string,delimiter)	string	
list contains(list,element)	list,any element of the semantic domain	
count(list)	list	
min(list) min(C1,...,Cn),N>0 max(list) max(C1,...,Cn),N>0	non-empty list of comparable items or argument list of one or more comparable items	
sum(list) sum(N1,...,Nn),N>0	list of 0 or more numbers or argument list of one or more numbers	
mean(list) mean(N1,...,Nn),N>0	non-empty list of numbers or argument list of one or more numbers	
all(list) all(B1,...,Bn),N>0	list of Boolean items of argument list of one or more Boolean items	
any(list) any(B1,...,Bn),N>0	list of Boolean items of argument list of one or more Boolean items	
sublist(list,start position, length?)	list,number, number	
append(list,item...)	list,any element	
concatenate(list...)	list	
insert before(list,position, newItem)	list,number,any element	
remove(list,position)	list,number	
reverse(list)	list	
index of(list,match)	list,any element	
union(list,...)	list	
distinct values(list)	list	
flatten(list)	list	
product(list) product(N1,...,Nn)	list is a list of numbers. N1..Nn are numbers	
median(list) median(N1,...,Nn)	list is a list of numbers. N1..Nn are numbers	
stddev(list) stddev(N1,...,Nn)	list is a list of numbers. N1..Nn are numbers	

Continued on next page

Table 1 – continued from previous page

Name(parameters)	Parameter Domain	pySFeel implementation notes
mode(list) mode(N1,...,Nn)	list is a list of numbers. N1..Nn are numbers	
deciman(n,scale)	number,number	pySFeel does the appropriate rounding, but the returned value is a float. Trailing zeros are not retained.
floor(n)	number	
ceiling(n)	number	
abs(n)	number	
modulo(dividend,divisor)	number,number	
sqrt(n)	number	
log(n)	number	
exp(n)	number	
odd(n)	number	
even(n)	number	
is(expr, expr)	expr, expr	
before(range, range)	range, range	
after(range, range)	range, range	
meets(range, range)	range, range	
met by(range, range)	range, range	
overlaps(range, range)	range, range	
overlaps before(range, range)	range, range	
overlaps after(range, range)	range, range	
finishes(range, range)	range, range	
finished by(range, range)	range, range	
includes(range, range)	range, range	
during(range, range)	range, range	
starts(range, range)	range, range	
started by(range, range)	range, range	
coincides(range, range)	range, range	
day of year(date)	date, date and time	
day of week(date)	date, date and time	
month of year(date)	date, date and time	
week of year(date)	date, date and time	

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

p

pySFeel, [1](#)

P

`pySFeel` (*module*), 1

S

`sFeelParse()` (*pySFeel.SFeelParser method*), 1

`SFeelParser` (*class in pySFeel*), 1