Python Language Support in SNAPpy

Modules

SNAPpy supports import of user-defined as well as standard predefined Python source library modules.

from *module* import * # Supported

import module # Not supported

Variables

Local and Global variables are supported. On RAM constrained devices, SNAPpy images are limited to 64 system globals and 64 concurrent locals.

Functions

Up to 255 "public" functions may be defined. These are remotely callable using SNAP RPC protocol.

Non-public functions (prefixed with underscore) are limited only by size of flash memory.

Data Types

SNAPpy supports the following fundamental Python data types:

NoneType, int, bool, string, function

int is a signed 16-bit integer

string has max size of 255 bytes

SNAPpy does *not* currently support the following common Python types:

float, long, complex, tuple, list, dict, set

User-defined objects (*class* types)

Keywords

The following Python reserved identifiers are supported in SNAPpy:

and from not while elif global or else if pass break import print continue return def is

The following identifiers are reserved, but not yet supported in SNAPpy:

del as with assert yield except class exec in raise finally for lambda try

Operators

SNAPpy supports all Python operators, with the exception of *floor* and *power*.

+ - * ** / // %

<< >> & | ^ ~

< > <= >= == != <>

Slicing

Slicing is supported for **string** data types. The current version of SNAPpy is constrained to a single dynamic "slice buffer," which is 64-bytes in size. Subsequent slices will overwrite this buffer.

Concatenation

Concatenation is supported for **string** data types. The current version of SNAPpy is constrained to a single dynamic "concatenation buffer," which is 64-bytes in size. Subsequent concatenation will overwrite this buffer.

Subscripting

Subscripting is supported for **string** data types.

Expressions

SNAPpy supports all Python Boolean, Binary bit-wise, Shifting, arithmetic, and comparison expressions – including the ternary **if** form.

Builtins

Supported Python built-ins: **len, ord, chr, int, str**

Additionally, many RF module-specific embedded network and control built-ins are supported.