Supported file types and actions

File types

pyntree currently supports a plethora of file types, all of which can be encrypted:

Pickle (default)

Pickled data is python data saved directly to a file as bytes. This way, you can save objects (ex: datetime) without losing any data or causing any problems while trying to reload the file.

Filetype parameter: `pyn`

Supported file extensions:

- .pyn
- .pyndb
- .pkl

Compressed (and pickled)

Compressed data can be saved in a variety of formats, thanks to the `compress_pickle` module. For a full list of supported file extensions and types, see here.

JSON

JSON data can be saved and loaded with pyntree.

Filetype parameter: `json`

Supported file extensions:

- .json

YAML

YAML data can also be saved and loaded with pyntree.

Filetype parameter: `yaml`

Supported file extensions:

- .yaml
- .yml

Plain text

> Saving python objects to a plaintext file will likely cause loading issues since pyntree will use eval() to load the data. This means that if it tries to create an object from a class it hasn't imported, the loading will fail. Additionally, the reliance on the eval() function means that you should only load trusted plaintext files, as loading untrusted ones could present a security risk.

You can also save the data as a string representation of the dictionary.

Filetype parameter: `txt`

Supported file extensions:

- .txt

The File object

The File object is a helper object to ensure data consistency across nodes. Each node holds a link to the file. Although you don't have to use the File object directly, it is helpful to understand what parameters are available since they can be passed to your root node.

Parameters:

- data (Mandatory)
  - The filename or dictionary to be loaded into the File object
- filetype (Optional - auto-detected, fallback is 'pyn')
  - See File types.
- autosave (Optional - False)
  - Save file on modification
- save_on_close (Optional - False)
  - Save data to the file when the File object is destroyed

Changing the active file

To change the active file, you can use the `switch_to_file` method:

```
File.switch_to_file(filename, filetype=None)
# You can also call Node.switch_to_file for initialized Nodes.
```

If the `filetype` parameter is set to `None`, then the filetype will be inferred.

> This method **will not** reload the data from the file.

To reload the data from the new file, see Reloading/retrieving the data from the file.

Reloading/retrieving the data from the file

To reload the data from the file, use the following method:

```
File.reload()
```

This method is only 1 line of code:

```
self.data = self.read_data()
```

The `read_data` method returns the data stored in the file, after being parsed into a python dictionary.

Encrypting a file

> Encryption is a new feature as of 1.0.0!

> To use encryption, first type `pip install pyntree[encryption]` to install the necessary dependencies.

To encrypt your data, simply specify a password parameter for the Node when saving or loading.

```
db = Node('encrypted.pyn', password='password')
```

Optionally, you can also specify a salt:

```
db = Node('encrypted.pyn', password='password', salt=b'random_salt')
```