sanest

sane
nested
dictionaries and lists

wouter bolsterlee

@wbolster

```
pip install sanest
```

nested operations

type checking

data
model
restrictions

```
{
  "data": {
    "users": [
      {"id": 12, "name": "alice"},
      {"id": 34, "name": "bob"}
    ]
  }
}
```

```
d = json.loads(...)
for user in d['data']['users']:
      print(user['name'])
```

```
TypeError:
string indices must be integers
```

```
TypeError:
list indices must be integers or slices,
not str
```

```
TypeError:
'NoneType' object is not subscriptable
```

…

or no error at all :(

validation?

sane
middle
ground

type safe
and
fail-fast

```
d = json.loads(...)
for user in d['data']['users']:
    print(user['name'])
```

```
d = json.loads(...)
wrapped = sanest.dict.wrap(d)
for u in wrapped['data', 'users':[dict]]:
    print(u['name':str])
```

data model

nested operations

```
'users', 0, 'name'
```

```
d['users', 0, 'name']
```

```
path = ['users', 0, 'name']
d[path]
```

```
d.get(['users', 0, 'name'], 'default')
```

type checking

```
user['name':str]
```

```
user.get('name', type=str)
```

```
d['users', 0, 'name':str]
```

wrapping

```
d = sanest.dict.wrap(existing_dict)
```

```
d.unwrap()
```

error handling

missing data

```
LookupError
    KeyError
    IndexError
```

```
>>> d['users', 0, 'xyz']
Traceback (most recent call last):
...
KeyError: ['users', 0, 'xyz']
```

```
>>> d['users', 123, 'name']
Traceback (most recent call last):
...
IndexError: ['users', 123]
```

problematic data

```
ValueError
    sanest.DataError
        sanest.InvalidStructureError
        sanest.InvalidValueError
```

```
>>> d['users', 0, 'name':int]
Traceback (most recent call last):
...
InvalidValueError: expected int, got str
at path ['users', 0, 'name']: 'alice'
```

recap

```
d = json.loads(...)
for user in d['data']['users']:
    print(user['name'])
```

```
d = json.loads(...)
wrapped = sanest.dict.wrap(d)
for u in wrapped['data', 'users':[dict]]:
    print(u['name':str])
```

sanest.rtfd.io

wbolster/sanest