noiserandom_documentation

October 6, 2025

1 noiserandom — Jupyter Documentation

This notebook documents the Python package **`noiserandom`**. It explains installation, core concepts, and usage examples corresponding to the code implementation provided.

1.1 Installation

```
pip install noiserandom
```

1.2 Requirements

- A working camera device accessible by OpenCV when using image capture.
- Python packages:
  - `opencv-python` (used via `cv2`)
  - `gmpy2` (for probable-prime testing)
  - `pycryptodome` (for RSA key construction)
  - Standard library modules: `os`, `secrets`, `time`, `sys`

1.3 Importing

```
from noiserandom import NoiseRandom
```

1.4 Quickstart

The generator captures one or more images from a camera and derives randomness from the image data.

```
# Example setup (adjust the path to a writable directory on your system)
from noiserandom import NoiseRandom

nr = NoiseRandom(
    path="./entropy_images",      # directory where captured images will be written
    strength=1,                   # number of images to capture per draw (minimum 1)
    cameras=[0],                  # list of camera indices to sample from
    disable_scramble=False,       # internal scrambling toggle
    disable_delete_images=False, # keep or delete captured images after use
)
```

1.5 Function: `captureImage`

```
captureImage(path: str, total_images: int = 1, camera: int = 0) -> list[str]
```

Captures `total_images` frames from the camera indicated by `camera` and saves them as `.jpg` files in `path`.

Parameters
- `path`: Directory where images are saved.
- `total_images`: Number of images to capture.
- `camera`: Camera index for `cv2.VideoCapture`.

Returns
- `list[str]`: Absolute or relative paths of the saved images.

Raises
- `Exception`: If the camera cannot be opened or a frame cannot be read.

1.6 Class: `NoiseRandom`

1.6.1 Constructor

```
NoiseRandom(
    path: str,
    strength: int = 1,
    cameras: list[int] = [0],
    disable_scramble: bool = False,
    disable_delete_images: bool = False,
)
```

Parameters
- `path`: Directory used to persist captured images.
- `strength`: Number of images captured per draw; values < 1 are coerced to 1.
- `cameras`: List of camera indices to choose from when capturing.
- `disable_scramble`: If `True`, disables scrambling (internal byte shuffling).
- `disable_delete_images`: If `True`, retains captured images after use.

1.6.2 `randomInt`

```
randomInt(get_bytes: bool = False) -> int | bytes
```

Captures images, extracts a portion of the JPEG byte stream, optionally scrambles it, and returns either the raw bytes or a big-endian integer representation.

Parameters
- `get_bytes`: When `True`, returns the derived bytes; otherwise returns an integer.

Returns
- `int | bytes`: Random integer or bytes.

1.6.3 `randomBytes`

```
randomBytes(total_bytes: int, get_bytes: bool = True) -> int | bytes
```

Derives a pool from `randomInt(True)` and selects `total_bytes` bytes from it.

Parameters
- `total_bytes`: Number of bytes to return; must be > 0.
- `get_bytes`: When `True`, returns bytes; otherwise returns an integer built from those bytes.

Returns
- `int | bytes`: Selected bytes or the corresponding big-endian integer.

Raises
- `ValueError`: If `total_bytes` 0.

1.6.4 `randomPrime`

```
randomPrime(total_bytes: int) -> int
```

Selects a candidate integer of `total_bytes` bytes and repeats until a probable prime is found using `gmpy2.is_prime`.

Parameters
- `total_bytes`: Byte length of the prime to return; must be > 0.

Returns
- `int`: A probable prime of the requested byte length.

Raises
- `ValueError`: If `total_bytes` 0.

1.6.5 `generate_rsa_keys`

```
generate_rsa_keys(
    p: int,
    q: int,
    e: int = 65537,
    private_key_name: str = "private.pem",
    public_key_name: str = "public.pem"
) -> None
```

Constructs an RSA key given primes `p` and `q`, then writes a private key (PEM) and public key (PEM) to disk.

Parameters
- `p`, `q`: Prime factors of the modulus.
- `e`: Public exponent.
- `private_key_name`: Output filename for the private key (PEM).
- `public_key_name`: Output filename for the public key (PEM).

Raises
- `ValueError`: If `e` and `phi(n)` are not coprime.

1.6.6 Fixed-size helpers

```
random1024()  -> int  # 1024-bit (128-byte) integer
random2048()  -> int  # 2048-bit (256-byte) integer
```

```
random4096()  -> int  # 4096-bit (512-byte) integer

randomPrime1024()  -> int  # 1024-bit probable prime
randomPrime2048()  -> int  # 2048-bit probable prime
randomPrime4096()  -> int  # 4096-bit probable prime
```

1.7 Usage Examples

1.7.1 1) Draw random bytes

```
# from noiserandom import NoiseRandom
# nr = NoiseRandom(path="./entropy_images", strength=1, cameras=[0])
# b = nr.randomBytes(32, get_bytes=True)
# len(b), b
```

1.7.2 2) Draw a random integer

```
# from noiserandom import NoiseRandom
# nr = NoiseRandom(path="./entropy_images", strength=1, cameras=[0])
# x = nr.randomInt(get_bytes=False)
# type(x), x.bit_length()
```

1.7.3 3) Generate a probable prime of a given size

```
# from noiserandom import NoiseRandom
# nr = NoiseRandom(path="./entropy_images", strength=1, cameras=[0])
# p = nr.randomPrime(256)  # 2048-bit prime
# p.bit_length()
```

1.7.4 4) Generate RSA keys

```
# from noiserandom import NoiseRandom
# nr = NoiseRandom(path="./entropy_images", strength=1, cameras=[0])
# p = nr.randomPrime(256)
# q = nr.randomPrime(256)
# nr.generate_rsa_keys(p, q, e=65537, private_key_name="private.pem", public_key_name="public.
# print("Keys written: private.pem, public.pem")
```

1.8 Internal Methods (for reference)

- `__deleteImages()` — Removes captured images from disk unless `disable_delete_images=True`.
- `__captureImages()` — Captures images according to `strength` and selected camera from `cameras`.
- `__scramble(data: bytes) -> bytes` — Performs in-place byte shuffling on the extracted data and returns the result.