Code Style Documentation

Class Comments

```python
class SampleClass:
    """Summary of class here.

    Longer class information....
    Longer class information....

    Attributes:
        likes_spam: A boolean indicating if we like SPAM or not.
        eggs: An integer count of the eggs we have laid.
    """

    def __init__(self, likes_spam=False):
        """Inits SampleClass with blah."""
        self.likes_spam = likes_spam
        self.eggs = 0

    def public_method(self):
        """Performs operation blah."""
```

Function/Method Comments

```python
def fetch_smalltable_rows(table_handle: smalltable.Table,
                          keys: Sequence[Union[bytes, str]],
                          require_all_keys: bool = False,
) -> Mapping[bytes, Tuple[str]]:
    """Fetches rows from a Smalltable.

    Retrieves rows pertaining to the given keys from the Table instance
    represented by table_handle.  String keys will be UTF-8 encoded.

    Args:
        table_handle: An open smalltable.Table instance.
        keys: A sequence of strings representing the key of each table
          row to fetch.  String keys will be UTF-8 encoded.
        require_all_keys: Optional; If require_all_keys is True only
          rows with values set for all keys will be returned.

    Returns:
        A dict mapping keys to the corresponding table row data
        fetched. Each row is represented as a tuple of strings. For
        example:

        {b'Serak': ('Rigel VII', 'Preparer'),
         b'Zim': ('Irk', 'Invader'),
         b'Lrrr': ('Omicron Persei 8', 'Emperor')}

        Returned keys are always bytes.  If a key from the keys argument
is
        missing from the dictionary, then that row was not found in the
        table (and require_all_keys must have been False).

    Raises:
        IOError: An error occurred accessing the smalltable.
    """
```

Use False evaluation in the place of `0, None, [], {}, ''`

```
users = []
if not users:
    // do something when there are users
```

Use `@property` decorator when possible

- Example

```
import math

    class Square:
        """A square with two properties: a writable area and a read-only
perimeter.

        To use:
        >>> sq = Square(3)
        >>> sq.area
        9
        >>> sq.perimeter
        12
        >>> sq.area = 16
        >>> sq.side
        4
        >>> sq.perimeter
        16
        """

        def __init__(self, side):
            self.side = side

        @property
        def area(self):
            """Area of the square."""
            return self._get_area()

        @area.setter
        def area(self, area):
            return self._set_area(area)

        def _get_area(self):
            """Indirect accessor to calculate the 'area' property."""
            return self.side ** 2

        def _set_area(self, area):
            """Indirect setter to set the 'area' property."""
            self.side = math.sqrt(area)

        @property
        def perimeter(self):
            return self.side * 4
```

References

- Google Python Style Guide