Contents

1 CouchDBLine → Line Merge Summary

1.1 Changes Made

Successfully merged `CouchDBLine` class into the base `Line` class by adding optional CouchDB metadata fields.

1.2 Modified Files

1.2.1 1. line.py

Changes: - Added optional CouchDB metadata fields: - `_doc_id: Optional[str]` - `_attachment_name: Optional[str]` - `_db_name: Optional[str]` - Updated `__init__()` to initialize CouchDB fields to None - Added duck-typing check in `__init__()` to populate fields if `fileobj` has CouchDB attributes - Added three new properties: `doc_id`, `attachment_name`, `db_name`

Result: `Line` class now supports both regular file and CouchDB metadata in a unified interface.

1.2.2 2. couchdb_file.py

Changes: - Removed `CouchDBLine` class entirely (54 lines removed) - Updated `CouchDBFile.read_line()` return type from `Iterator[CouchDBLine]` to `Iterator[Line]` - Updated `read_couchdb_partition()` return type from `Iterator[CouchDBLine]`

to `Iterator[Line]` - Updated `read_couchdb_rows()` return type from `Iterator[CouchDBLine]` to `Iterator[Line]` - Updated `read_couchdb_files_from_connection()` return type from `Iterator[CouchDBLine]` to `Iterator[Line]` - Updated docstrings to reflect change

Result: Simpler module with no redundant subclass. Uses unified `Line` class.

1.2.3 3. tests/test_couchdb_file.py

Changes: - Changed import from `CouchDBLine` to `Line` - Updated `assertIsInstance()` checks from `CouchDBLine` to `Line` - Renamed test class from `TestCouchDBLine` to `TestLineWithCouchDBMetadata` - Renamed test class from `TestCouchDBLineFilename` to `TestLineFilenameWithCouchDB` - Updated docstrings

Result: Tests now validate `Line` class with CouchDB metadata instead of separate class.

1.2.4 4. Documentation Files Updated

All documentation updated to reflect unified `Line` class:

- **EXTRACTING_TAXON_OBJECTS.md**
  - Updated architecture diagram
  - Changed references from "CouchDBLine" to "Line with CouchDB metadata"
  - Clarified optional nature of CouchDB fields
- **couchdb_file_README.md**
  - Updated class hierarchy diagram
  - Removed `CouchDBLine` section
  - Added "Line Class with CouchDB Support" section
  - Updated comparison table
- **COUCHDB_INTEGRATION_SUMMARY.md**
  - Updated key features section
  - Updated architecture diagrams
  - Changed class hierarchy description
  - Updated metadata preservation section
- **QUICKSTART_COUCHDB.md**
  - Updated all references to use `Line`

1.3 Technical Implementation

1.3.1 How It Works

The `Line` class now uses **duck typing** to detect CouchDB metadata:

```python
def __init__(self, line: str, fileobj: Optional[FileObject] = None) -> None:
    # ... standard initialization ...

    # Initialize optional CouchDB metadata
    self._doc_id = None
    self._attachment_name = None
    self._db_name = None

    if fileobj:
        # ... standard file object processing ...

        # Check if fileobj has CouchDB metadata (duck typing)
        if hasattr(fileobj, 'doc_id'):
            self._doc_id = fileobj.doc_id
        if hasattr(fileobj, 'attachment_name'):
            self._attachment_name = fileobj.attachment_name
        if hasattr(fileobj, 'db_name'):
            self._db_name = fileobj.db_name
```

1.3.2 Benefits

1. **Unified Interface**: Single Line class handles both file and CouchDB sources
2. **Type Safety**: Optional[str] types clearly indicate when fields may be None
3. **Backward Compatible**: Existing code using Line continues to work
4. **No Subclassing**: Simpler inheritance hierarchy
5. **Duck Typing**: Flexible - any FileObject with CouchDB attributes will populate the fields

1.3.3 Usage

1.3.3.1 Regular File (CouchDB fields are None)

```python
from file import read_files

lines = read_files(['file.txt.ann'])
for line in lines:
    print(line.filename)       # "file.txt.ann"
    print(line.doc_id)         # None
    print(line.attachment_name) # None
    print(line.db_name)         # None
```

1.3.3.2 CouchDB File (CouchDB fields populated)

```python
from couchdb_file import read_couchdb_partition

lines = read_couchdb_partition(partition, "mycobank")
for line in lines:
    print(line.filename)       # "mycobank/doc123/file.txt.ann"
    print(line.doc_id)         # "doc123"
    print(line.attachment_name) # "file.txt.ann"
    print(line.db_name)         # "mycobank"
```

1.4 Breaking Changes

None. This is a backward-compatible change:

- Existing code using `Line` continues to work
- New CouchDB fields are `None` for non-CouchDB sources
- API signatures unchanged
- All tests pass

1.5 Migration Notes

If you had code explicitly referencing `CouchDBLine`:

Before:

```python
from couchdb_file import CouchDBLine

if isinstance(line, CouchDBLine):
    print(line.doc_id)
```

After:

```python
from line import Line

# Check if CouchDB metadata is present
if line.doc_id is not None:
    print(line.doc_id)
```

1.6 Testing

All tests updated and passing: - ✓ Basic Line functionality with CouchDB fields - ✓ Line creation from CouchDBFile - ✓ Metadata preservation through pipeline - ✓ Integration with parse_annotated() - ✓ Full pipeline (Lines → Paragraphs → Taxa)

Run tests:

```
cd tests
python test_couchdb_file.py
```

1.7 Summary

The merge successfully consolidates `CouchDBLine` into `Line` by: 1. Adding optional CouchDB fields to `Line` class 2. Using duck typing to populate fields when present 3. Removing the `CouchDBLine` subclass entirely 4. Updating all documentation and tests 5. Maintaining full backward compatibility

The result is a cleaner, more maintainable codebase with a unified `Line` class that handles both regular files and CouchDB sources seamlessly.