

# Dealing with unsightly data in the real world

Alexander Dutton  
Lead Developer, Mobile Oxford

Oxford University Computing Services  
PyCon Atlanta 2010



# What's this all about?

- You need/want someone else's data
- The data isn't in a format you'd like
- The data provider is unable to give you 'better' data
- You've got to make do with what you've got



# A few examples

## Screenscraping

- lxml.html
- BeautifulSoup

## Mapping between mark-ups

- xml.handler.ContentHandler
- generators/coroutines
- regular expressions

## Mapping between protocols

- 3rd party libraries



# Checklist

- Get permission (if necessary)
- Reverse engineer the data source
- Write code to pull the data
- Put an API on it
- Test!
- Deploy



# Permission

- Read the terms of use
- The provider may not be happy
- If unsure, contact them
- Be gentle
- If told to stop, you'd better stop!



# Reverse engineering

## Things to consider:

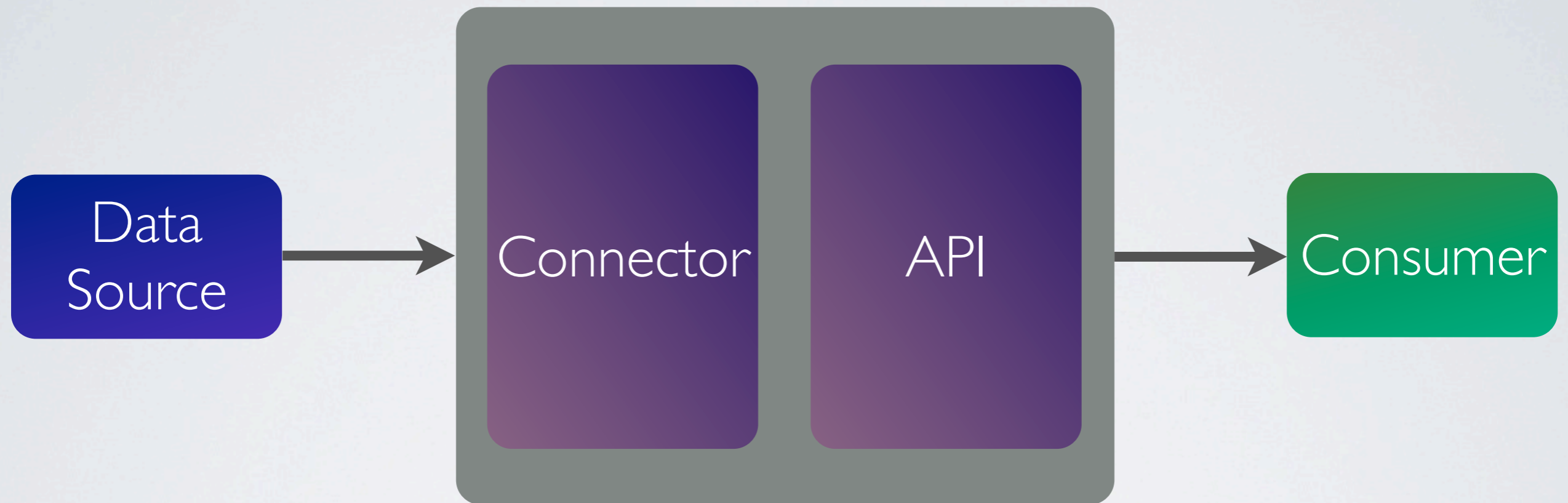
- Have you covered all the corner cases?
- How stable is the source data?
- Will the provider warn you of changes?

## Tools:

- Documentation
- Python shell
- Firebug or equivalent
- Wireshark



# Model



# Interacting with the data source

Make it as resilient as possible

- Coerce individual data to a defined type/range
- Error checking
- Log exceptions, but handle them gracefully
- “Be strict in what you give and forgiving in what you receive”





# Defining an API

- Be generic
- Be specific
- Document the API
- Write more tests



# Testing

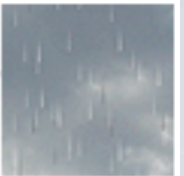
This has been a running theme.  
You'll do well to have unit tests for each part of your module.  
When it breaks (and it will), you'll want to know.



# BBC Weather

## BBC – Weather Centre – Forecast for Eastbourne, United Kingdom

by the BBC Weather Centre in association with the Met Office



### Sunday: light rain, Max Temp: 7°C (45°F), Min Temp: 6°C (43°F)

21 February 2010 02:35

Max Temp: 7°C (45°F), Min Temp: 6°C (43°F), Wind Direction: SSW, Wind Speed: 23mph, Visibility: moderate, Pressure: 994mb, Humidity: 92%, UV risk: low, Pollution: low, Sunrise: 06:58GMT, Sunset: 17:26GMT

### Monday: heavy rain, Max Temp: 9°C (48°F), Min Temp: 5°C (41°F)

21 February 2010 02:35

Max Temp: 9°C (48°F), Min Temp: 5°C (41°F), Wind Direction: SSE, Wind Speed: 24mph, Visibility: poor, Pressure: 982mb, Humidity: 93%, UV risk: low, Pollution: low, Sunrise: 06:56GMT, Sunset: 17:28GMT

### Tuesday: heavy rain, Max Temp: 7°C (45°F), Min Temp: 7°C (45°F)

21 February 2010 02:35

Max Temp: 7°C (45°F), Min Temp: 7°C (45°F), Wind Direction: NE, Wind Speed: 10mph, Visibility: moderate, Pressure: 989mb, Humidity: 97%, UV risk: low, Pollution: low, Sunrise: 06:54GMT, Sunset: 17:29GMT



# BBC Weather

```
logger = logging.getLogger("app.weather")
FORECAST_URL = "http://newsrss.bbc.co.uk/weather/forecast/%d/Next3DaysRSS.xml"

def get_forecasts():
    FORECAST_RE = re.compile(
        r'Max Temp: (?P<max_temperature>-?\d+|N\A).+Min Temp: (?P<min_temperature>-?\d+|N\A) '
        + r'.+Wind Direction: (?P<wind_direction>[NESW]{0,3}|N\A), Wind Speed: '
        + r'(?P<wind_speed>\d+|N\A).+Visibility: (?P<visibility>[A-Za-z\ / ]+), '
        + r'Pressure: (?P<pressure>\d+|N\A).+Humidity: (?P<humidity>\d+|N\A).+'
        + r'UV risk: (?P<uv_risk>[A-Za-z]+|N\A), Pollution: (?P<pollution>[A-Za-z]+|N\A), '
        + r'Sunrise: (?P<sunrise>\d\d:\d\d)[A-Z]{3}, Sunset: (?P<sunset>\d\d:\d\d)[A-Z]{3}'
    )

    try:
        xml = ET.parse(urllib2.urlopen(FORECAST_URL % bbc_id))
    except Exception:
        logger.exception("Could not parse feed")
        return {}

    forecasts = {}
    for elem in xml.findall('.//item/description'):
        data = FORECAST_RE.match(elem.text)
        if data is None:
            logger.error("Weather not matched by RE")
            return {}
        data = data.groupdict()
        forecasts[data['observed_date']] = data

    return forecasts
```



# Libraries

Library information systems are queried using Z39.50, a stateful binary protocol.

```
class OLISearch(object):
    def __init__(self, query):
        self.connection = zoom.Connection(
            getattr(settings, 'Z3950_HOST'),
            getattr(settings, 'Z3950_PORT', 210),
        )
        self.connection.databaseName = getattr(settings, 'Z3950_DATABASE')
        self.connection.preferredRecordSyntax = getattr(settings, 'Z3950_SYNTAX', 'USMARC')

        self.query = zoom.Query('CCL', query)
        self.results = self.connection.search(self.query)

    def __iter__(self):
        for r in self.results:
            yield OLISResult(r)

    def __getitem__(self, key):
        if isinstance(key, slice):
            if key.step:
                raise NotImplementedError("Stepping not supported")
            return map(OLISResult, self.results.__getslice__(key.start, key.stop))
        return OLISResult(self.results[key])

    def __len__(self):
        return len(self.results)
```



# Libraries

That was easy; right?



# Libraries

Exposing this over HTTP is a problem.

Each HTTP request requires a new Z39.50 connection.

Three ways to solve:

- Pull all the results for a query and cache them
- Create a bijection between the HTTP and Z39.50 sessions
- Create a connection manager which abstracts the state away



\*sigh\*

## No session

Your session has either ended due to inactivity, or you haven't started one yet.

Return to the [OLIS web OPAC start page](#) to start a new session.

Or visit the [OLIS home page](#) to find out more about the catalogue.





# Libraries

There's too much code for one slide

- We've got a connection manager in a separate process
- Exposes API using the multiprocessing module
- Query passed from Django to the CM with the sessionkey
- Finds connections[sessionkey]
- Checks query against previous query
- Requeries if necessary
- Returns an object implementing the list protocol
- 'Old' connections get timed out and closed



# Bus locations

BUSNET- Oxford map - Mozilla Firefox

http://oxfordshire.acislive.com/pda/basic\_pda.asp?SysId=35

Current area: **Oxford**

Applet started.

BUSNET- City Centre map - Mozilla Firefox

http://oxfordshire.acislive.com/pda/basic\_pda.asp?sysid=35&mapid=196&mapLevel=2&naptan=0

Current area: **City Centre**

Applet started.

The stops available for this area are:

- Bertie Place
- Bertie Place
- Binsey Lane
- Binsey Lane
- Bonn Square D4
- Canning Crescent
- Canning Crescent
- Carfax
- Carfax i1
- Carfax i2
- Castle Street E1
- Castle Street E2
- Castle Street E3
- Castle Street M1
- Castle Street M2
- Chatham Road
- Chatham Road
- Cherwell Street
- Clive Booth Hall
- Clive Booth Hall
- County Hall E4
- County Hall E5
- County Hall E6
- Crown Court P2
- East Avenue
- East Avenue
- Edgeway Road
- Edgeway Road
- Freelands Road
- Freelands Road
- Frideswide S R10



# Java. Oh Dear.

How does it work?  
No source to inspect.



# Hello, Wireshark

We sniffed its HTTP requests to work out what it was up to.

This led us to a URL to play with and some example requests.

```
NEW | 1024 | 4 | x5 | Operators/common/bus/1 | 45,302
```



# Documentation

Before we 'wrote' any code we blogged about how it works.

<http://blogs.oucs.ox.ac.uk/inapickle/2010/01/14/live-bus-locations-from-acis-oxontime/>



# Implementation

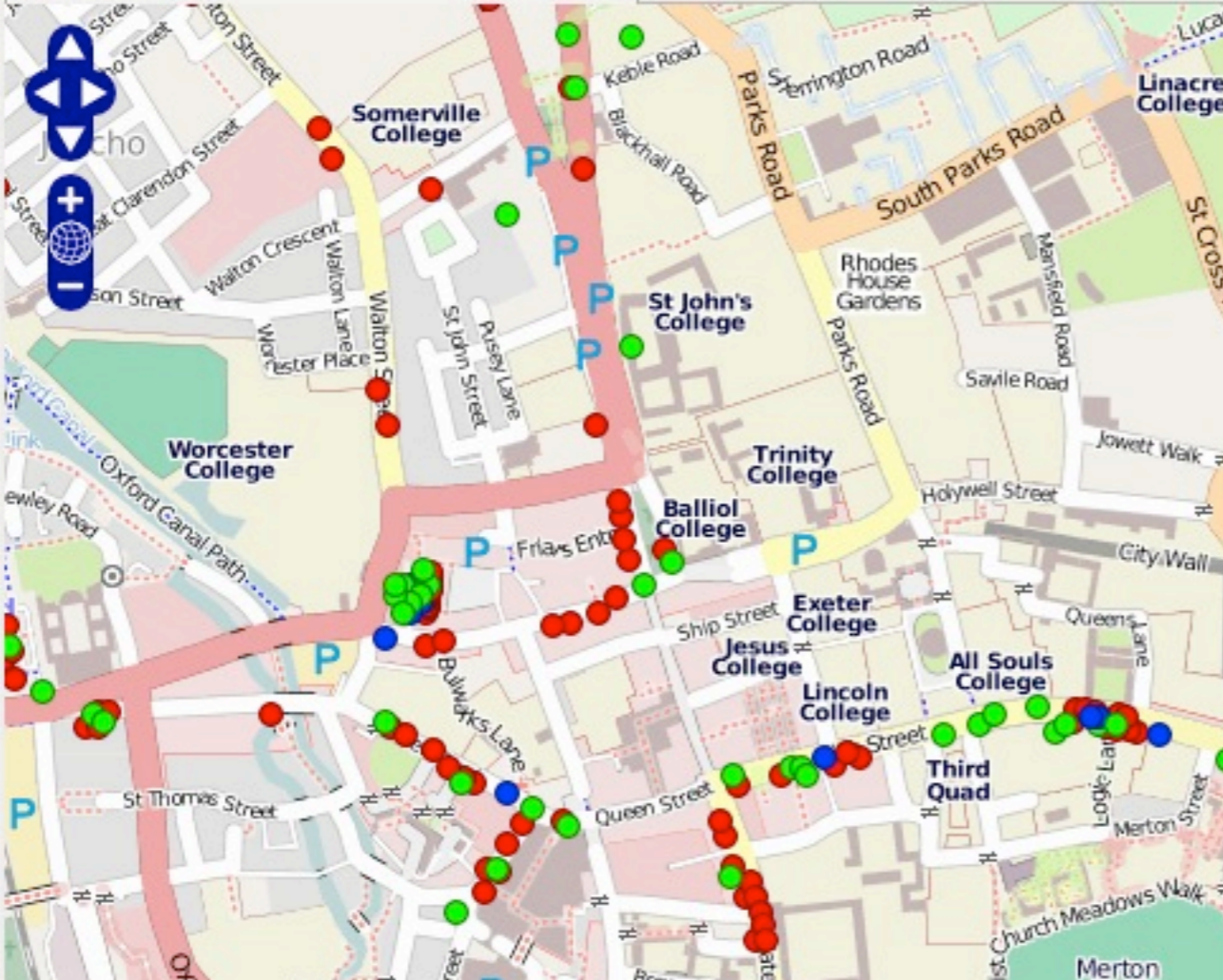
Oxford Buses - Mozilla Firefox

File Edit View History Bookmarks Split Tools Help

http://localhost:8000/

Most Visited Release Notes Fedora Project Red Hat Free Content

Oxford Buses



Stop: Opposite Keble Road, on Banbury Road

Service	Dest. Subsequent	Expected
7A	Kidlington	3 mins
2A	Kidlington	23 mins
S5	Bicester & Glory Fm	50 mins

Services

- [15](#) (4)
- [5](#) (8)
- [13](#) (2)
- [1](#) (9)
- [X80](#) (5)
- [TUBE](#) (18)
- [2](#) (3)
- [7A](#) (4)
- [X3](#) (3)
- [U1](#) (6)
- [X13](#) (4)
- [280](#) (2)

Done

βήτα  
mox



That's it  
Questions?

