KnoxPy Open Source Extravaganza

2019-02-07

Dale Visser

PySolr and DependencyCheck

Overview

Project	Implementation Language	License
PySolr	Python	3-Clause BSD
DependencyCheck	Java	Apache 2.0

PySolr

- ...is a client for Solr. What is Solr?
- IDATA – why I cared
- Submission/Acceptance experience

Apache Solr

“Solr is the popular, blazing-fast, open source enterprise search platform built on Apache Lucene™.”

http://lucene.apache.org/solr/

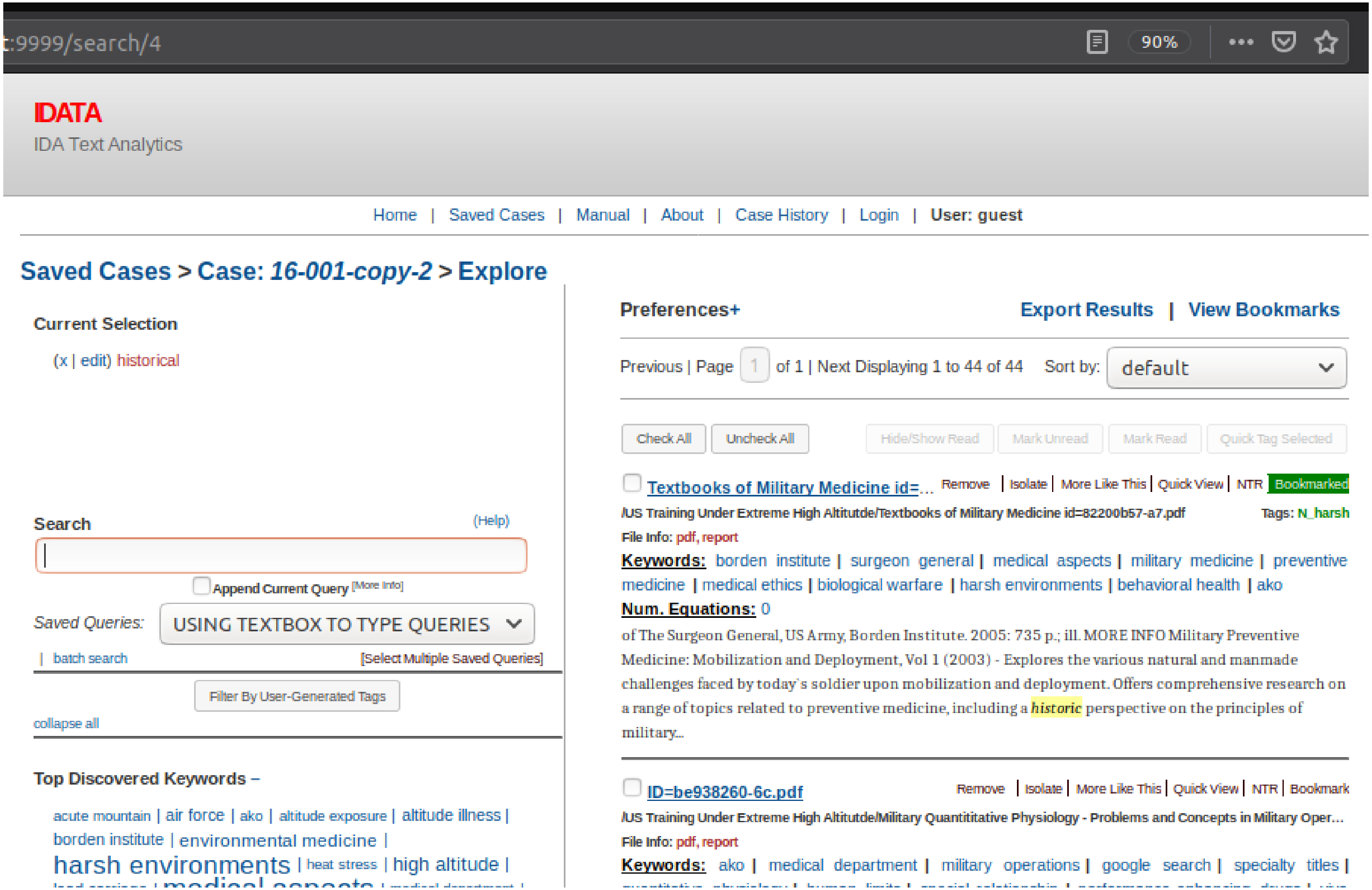
PySolr

“`pysolr` is a lightweight Python client for Apache Solr. It provides an interface that queries the server and returns results based on the query.”

https://github.com/django-haystack/pysolr

https://pypi.org/project/pysolr/

IDATA


:9999/search/4
90%
IDATA
IDA Text Analytics
Home | Saved Cases | Manual | About | Case History | Login | User: guest
Preferences+
Export Results | View Bookmarks
Check All
Uncheck All
Hide/Show Read
Mark Unread
Mark Read
Quick Tag Selected
Remove | Isolate | More Like This | Quick View | NTR
Bookmarked
Bookmark
(Help)
[More Info]
batch search
[Select Multiple Saved Queries]
Filter By User-Generated Tags
collapse all

Saved Cases > Case: *16-001-copy-2* > Explore

Current Selection

(x | edit) historical

Search

☐ Append Current Query

Saved Queries: USING TEXTBOX TO TYPE QUERIES

Top Discovered Keywords –

acute mountain | air force | ako | altitude exposure | altitude illness | borden institute | environmental medicine | harsh environments | heat stress | high altitude |

Previous | Page 1 of 1 | Next Displaying 1 to 44 of 44 Sort by: default

☐ Textbooks of Military Medicine id=...

/US Training Under Extreme High Altitutde/Textbooks of Military Medicine id=82200b57-a7.pdf Tags: N_harsh

File Info: pdf, report

Keywords: borden institute | surgeon general | medical aspects | military medicine | preventive medicine | medical ethics | biological warfare | harsh environments | behavioral health | ako

Num. Equations: 0

of The Surgeon General, US Army, Borden Institute. 2005: 735 p.; ill. MORE INFO Military Preventive Medicine: Mobilization and Deployment, Vol 1 (2003) - Explores the various natural and manmade challenges faced by today`s soldier upon mobilization and deployment. Offers comprehensive research on a range of topics related to preventive medicine, including a *historic* perspective on the principles of military...

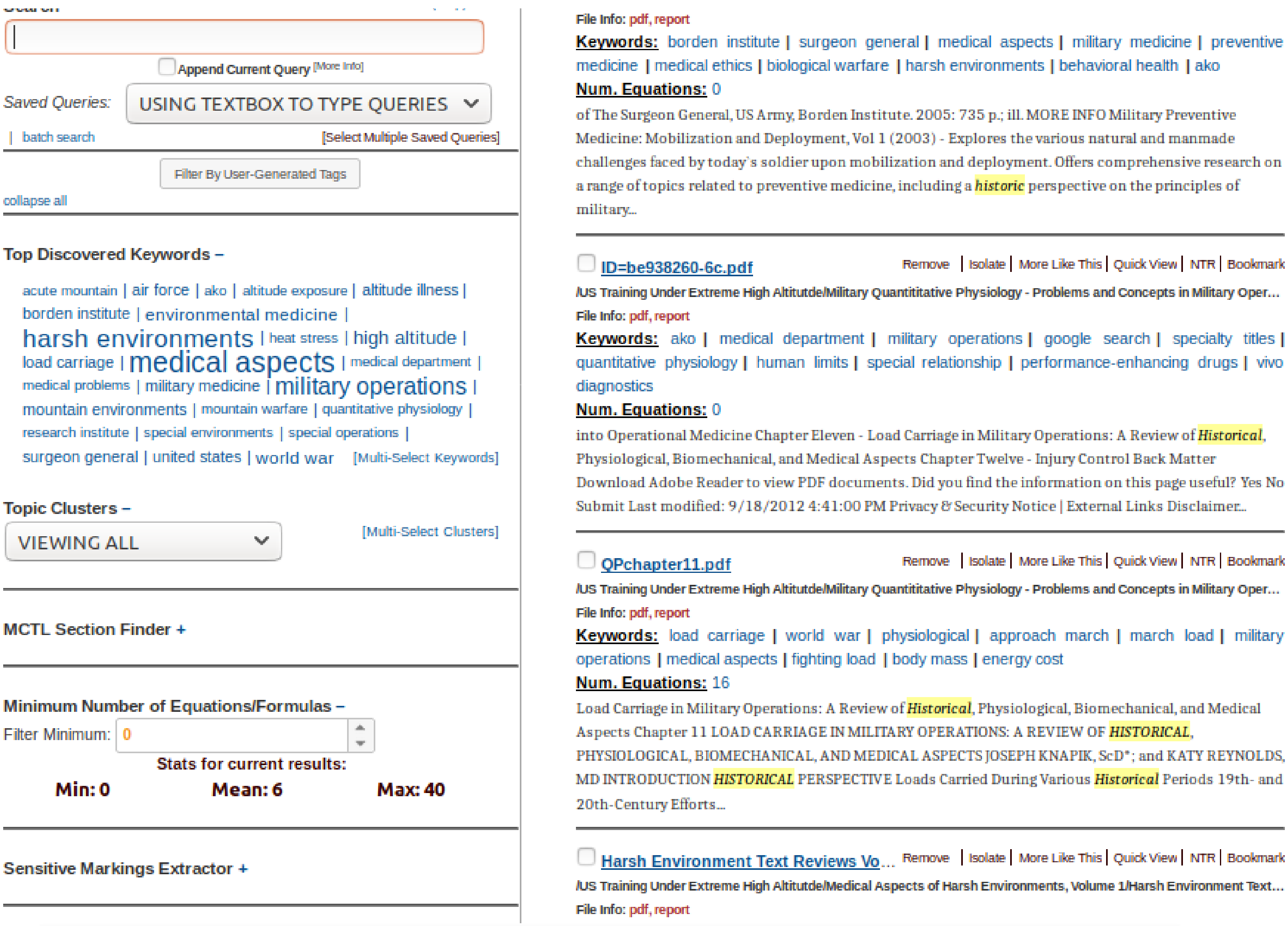
☐ ID=be938260-6c.pdf

/US Training Under Extreme High Altitutde/Military Quantititative Physiology - Problems and Concepts in Military Oper...

File Info: pdf, report

Keywords: ako | medical department | military operations | google search | specialty titles |

IDATA



□ Append Current Query [More Info]

Saved Queries: USING TEXTBOX TO TYPE QUERIES

| batch search [Select Multiple Saved Queries]

Filter By User-Generated Tags

collapse all

Top Discovered Keywords –

acute mountain | air force | ako | altitude exposure | altitude illness | borden institute | environmental medicine | harsh environments | heat stress | high altitude | load carriage | medical aspects | medical department | medical problems | military medicine | military operations | mountain environments | mountain warfare | quantitative physiology | research institute | special environments | special operations | surgeon general | united states | world war [Multi-Select Keywords]

Topic Clusters –

VIEWING ALL [Multi-Select Clusters]

MCTL Section Finder +

Minimum Number of Equations/Formulas –

Filter Minimum: 0

Stats for current results:

Min: 0 **Mean: 6** **Max: 40**

Sensitive Markings Extractor +

File Info: pdf, report

Keywords: borden institute | surgeon general | medical aspects | military medicine | preventive medicine | medical ethics | biological warfare | harsh environments | behavioral health | ako

Num. Equations: 0

of The Surgeon General, US Army, Borden Institute. 2005: 735 p.; ill. MORE INFO Military Preventive Medicine: Mobilization and Deployment, Vol 1 (2003) - Explores the various natural and manmade challenges faced by today`s soldier upon mobilization and deployment. Offers comprehensive research on a range of topics related to preventive medicine, including a *historic* perspective on the principles of military...

□ ID=be938260-6c.pdf Remove | Isolate | More Like This | Quick View | NTR | Bookmark

/US Training Under Extreme High Altitutde/Military Quantititative Physiology - Problems and Concepts in Military Oper...

File Info: pdf, report

Keywords: ako | medical department | military operations | google search | specialty titles | quantitative physiology | human limits | special relationship | performance-enhancing drugs | vivo diagnostics

Num. Equations: 0

into Operational Medicine Chapter Eleven - Load Carriage in Military Operations: A Review of *Historical*, Physiological, Biomechanical, and Medical Aspects Chapter Twelve - Injury Control Back Matter Download Adobe Reader to view PDF documents. Did you find the information on this page useful? Yes No Submit Last modified: 9/18/2012 4:41:00 PM Privacy & Security Notice | External Links Disclaimer...

□ QPchapter11.pdf Remove | Isolate | More Like This | Quick View | NTR | Bookmark

/US Training Under Extreme High Altitutde/Military Quantititative Physiology - Problems and Concepts in Military Oper...

File Info: pdf, report

Keywords: load carriage | world war | physiological | approach march | march load | military operations | medical aspects | fighting load | body mass | energy cost

Num. Equations: 16

Load Carriage in Military Operations: A Review of *Historical*, Physiological, Biomechanical, and Medical Aspects Chapter 11 LOAD CARRIAGE IN MILITARY OPERATIONS: A REVIEW OF *HISTORICAL*, PHYSIOLOGICAL, BIOMECHANICAL, AND MEDICAL ASPECTS JOSEPH KNAPIK, ScD*; and KATY REYNOLDS, MD INTRODUCTION *HISTORICAL* PERSPECTIVE Loads Carried During Various *Historical* Periods 19th- and 20th-Century Efforts...

□ Harsh Environment Text Reviews Vo... Remove | Isolate | More Like This | Quick View | NTR | Bookmark

/US Training Under Extreme High Altitutde/Medical Aspects of Harsh Environments, Volume 1/Harsh Environment Text...

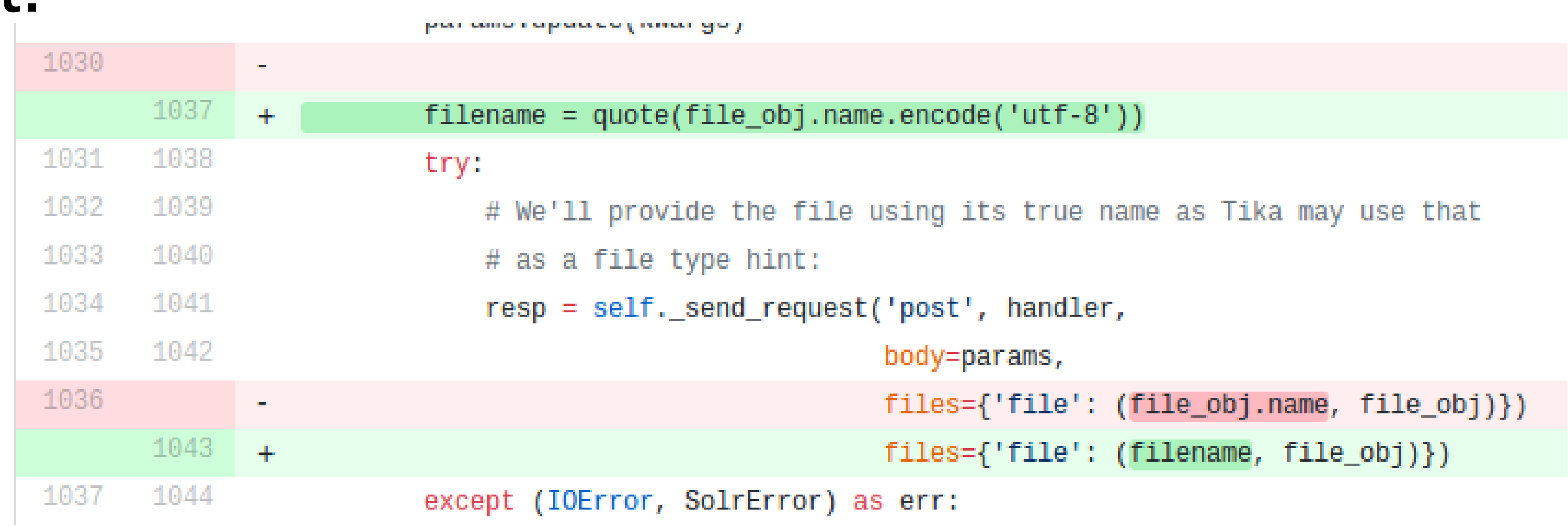
File Info: pdf, report

The issue to fix

- def extract(self, file_obj, ..., **kwargs):
- POSTs a file to the Solr index, relying on Solr to extract text and metadata
- Failing with spaces and unicode characters in filenames
- GitHub issue tracker existed for years, along with suggested fix!
- We were actually manually applying this fix.

The core of the fix

- Using urllib library’s quote() function.
- There’s more to it, but this is the most important bit:



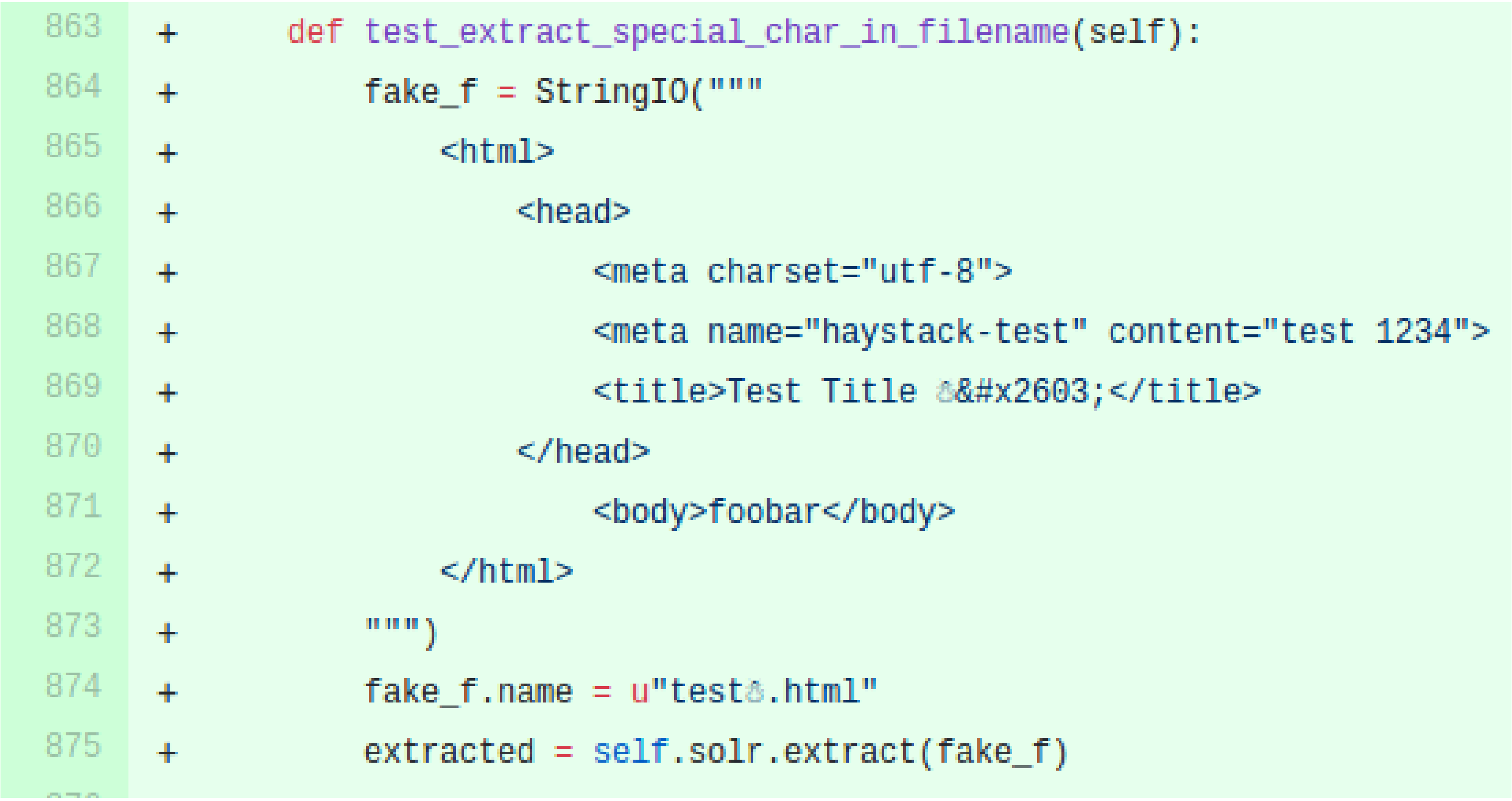
```
        params.update(kwargs)
-
+       filename = quote(file_obj.name.encode('utf-8'))
        try:
            # We'll provide the file using its true name as Tika may use that
            # as a file type hint:
            resp = self._send_request('post', handler,
                                      body=params,
-                                     files={'file': (file_obj.name, file_obj)})
+                                     files={'file': (filename, file_obj)})
        except (IOError, SolrError) as err:
```

My Contribution

- Cleaned up suggested fix a little
- Create new (smoke) test cases that the CI server could run to validate the change
- Create pull request
- Nudge the project administrator every few weeks to merge
- Result: I forgot about this for a few months, and when I checked back, it had quietly been merged and included in the v3.8.1 release.

Test code contribution

- This is just the start of the test, but it shows the exercising of the changed function.



```
+       def test_extract_special_char_in_filename(self):
+           fake_f = StringIO("""
+               <html>
+                   <head>
+                       <meta charset="utf-8">
+                       <meta name="haystack-test" content="test 1234">
+                       <title>Test Title ☃&#x2603;</title>
+                   </head>
+                       <body>foobar</body>
+               </html>
+           """)
+           fake_f.name = u"test☃.html"
+           extracted = self.solr.extract(fake_f)
```

OWASP Dependency-Check

Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities. Currently, Java and .NET are supported; additional experimental support has been added for Ruby, Node.js, Python, and limited support for C/C++ build systems (autoconf and cmake)*. The tool can be part of a solution to the OWASP Top 10 2017 A9-Using Components with Known Vulnerabilities previously known as OWASP Top 10 2013 A9-Using Components with Known Vulnerabilities.

https://www.owasp.org/index.php/OWASP_Dependency_Check

Why I contributed

- In 2015, participated in a work task funded by DHS to improve overall open source cybersecurity with targeted contributions
  - OWASP contributions
    - OWASP Dependency-Check
    - OWASP Zed Attack Proxy (ZAP)
  - Start of CII Best Practices Badge program, which keeps growing: https://bestpractices.coreinfrastructure.org/

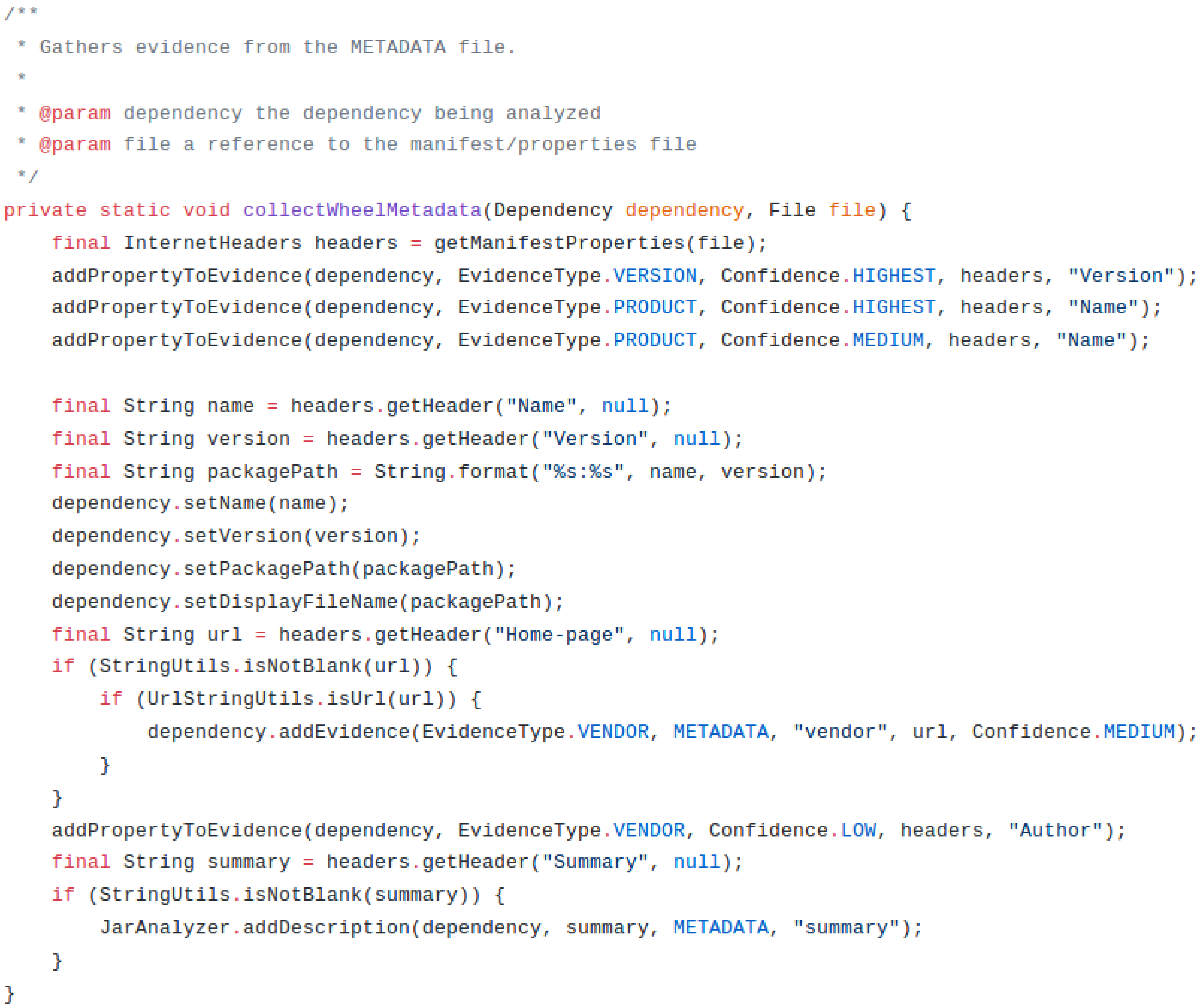
Dependency-Check Architecture

- Updates local data from nvd.nist.gov (National Vulnerability Database)
- Analyzes files/folders/archives for metadata evidence
  - Vendor
  - Product
  - Version
- Assigns a confidence (low, medium, high, highest) to evidence
- Compares to local NVD store in a configurable way

PythonDistributionAnalyzer

- Able to scan a folder for the following
  - .whl files
  - Old-style packages - .egg or .zip extension
- Looks for files (in filesystem or archive formats):
  - EGG-INFO, PKG_INFO, METADATA
  - *.dist-info, *.egg-info
- Leverages javax.mail.internet.InternetHeaders library to examine metadata files for evidence

Some of the DistributionAnalyzer:



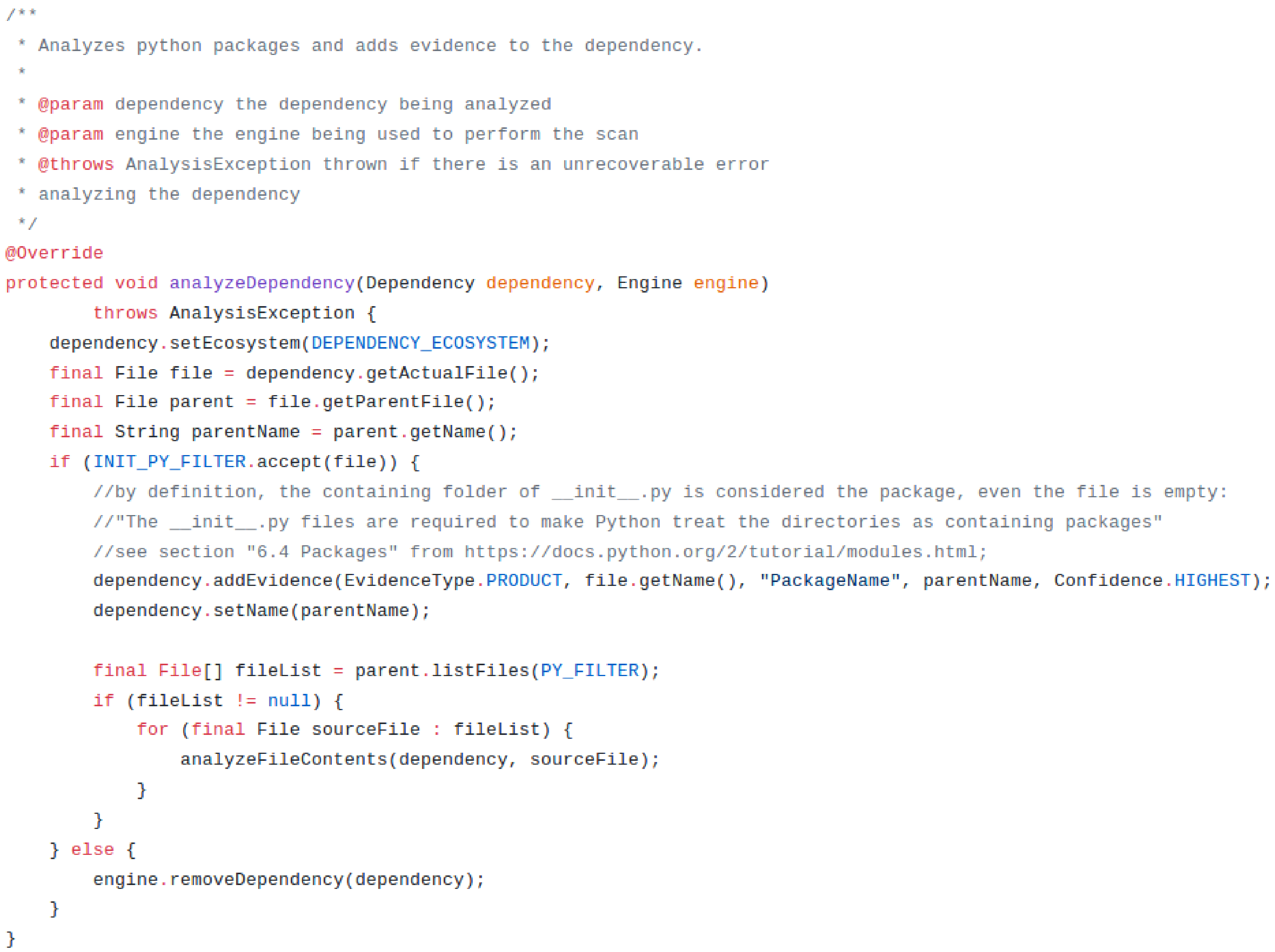
```java
/**
 * Gathers evidence from the METADATA file.
 *
 * @param dependency the dependency being analyzed
 * @param file a reference to the manifest/properties file
 */
private static void collectWheelMetadata(Dependency dependency, File file) {
    final InternetHeaders headers = getManifestProperties(file);
    addPropertyToEvidence(dependency, EvidenceType.VERSION, Confidence.HIGHEST, headers, "Version");
    addPropertyToEvidence(dependency, EvidenceType.PRODUCT, Confidence.HIGHEST, headers, "Name");
    addPropertyToEvidence(dependency, EvidenceType.PRODUCT, Confidence.MEDIUM, headers, "Name");

    final String name = headers.getHeader("Name", null);
    final String version = headers.getHeader("Version", null);
    final String packagePath = String.format("%s:%s", name, version);
    dependency.setName(name);
    dependency.setVersion(version);
    dependency.setPackagePath(packagePath);
    dependency.setDisplayFileName(packagePath);
    final String url = headers.getHeader("Home-page", null);
    if (StringUtils.isNotBlank(url)) {
        if (UrlStringUtils.isUrl(url)) {
            dependency.addEvidence(EvidenceType.VENDOR, METADATA, "vendor", url, Confidence.MEDIUM);
        }
    }
    addPropertyToEvidence(dependency, EvidenceType.VENDOR, Confidence.LOW, headers, "Author");
    final String summary = headers.getHeader("Summary", null);
    if (StringUtils.isNotBlank(summary)) {
        JarAnalyzer.addDescription(dependency, summary, METADATA, "summary");
    }
}
```

PythonPackageAnalyzer

- Able to scan a folder for Python packages, i.e., folders with __init__.py
- Regex scans all .py files therein
  - Docstrings and/or comments referring to vendor, author, and title
  - assignments to __version__, __title__, __summary__, __uri__, __url__, __homepage__, __author__ and/or all caps versions of same variables

Some of the PackageAnalyzer:



```java
/**
 * Analyzes python packages and adds evidence to the dependency.
 *
 * @param dependency the dependency being analyzed
 * @param engine the engine being used to perform the scan
 * @throws AnalysisException thrown if there is an unrecoverable error
 * analyzing the dependency
 */
@Override
protected void analyzeDependency(Dependency dependency, Engine engine)
        throws AnalysisException {
    dependency.setEcosystem(DEPENDENCY_ECOSYSTEM);
    final File file = dependency.getActualFile();
    final File parent = file.getParentFile();
    final String parentName = parent.getName();
    if (INIT_PY_FILTER.accept(file)) {
        //by definition, the containing folder of __init__.py is considered the package, even the file is empty:
        //"The __init__.py files are required to make Python treat the directories as containing packages"
        //see section "6.4 Packages" from https://docs.python.org/2/tutorial/modules.html;
        dependency.addEvidence(EvidenceType.PRODUCT, file.getName(), "PackageName", parentName, Confidence.HIGHEST);
        dependency.setName(parentName);

        final File[] fileList = parent.listFiles(PY_FILTER);
        if (fileList != null) {
            for (final File sourceFile : fileList) {
                analyzeFileContents(dependency, sourceFile);
            }
        }
    } else {
        engine.removeDependency(dependency);
    }
}
```

Contribution Experience

- The project leader was quite helpful and accepted most contributions
- Just as with PySolr, it was critical to include test cases that exercised my contributed code
- The project retains its Java and .NET focus, with my Python, Ruby, Node.js, etc. items labelled “experimental” and not turned on by default.
- Of course, now each of these communities (Python less so) have widely deployed free tools for auditing dependencies, e.g., `npm audit`