W3C Publishing Business Group: Epubcheck Development & Maintenance

Definition of the Problem 2
Proposed Implementation 3
  Option 1: Initial rewrite, Python 3
  Option 2: Initial refactor, Java 4
Rates and hours 4
  Option 1: Initial rewrite, Python 5
  Option 2: Initial refactor, Java 5
  Ongoing maintenance: 5
Company Overview 5
  History 5
  Contact Information 6
References 6

Deborah Kaplan
Suberic Networks

16 July 2018

Definition of the Problem

The Publishing Business Group of the World Wide Web Consortium seeks a developer/technical lead to oversee enhancements, developments, maintenance, and future direction of the Epubcheck validation tool.

The current tool:

- Is written in Java, and distributed as a jar file with accompanying bat/sh scripts to enable easy command line usage.
- Is internationalized, with translation strings in seven languages.
- Supports EPUB 2 and 3.0, but not 3.2.
- Uses test-driven development.
- Has an API.
- Validates file formats, various XML schemas, image files, and EPUB-specific rulesets.
- Validates EDUPUB, EPUB Dictionaries and Glossaries, EPUB Indexes, and EPUB Previews.

The tool moving forward, needs:

- Versions:
    - Support for EPUB 3.2 as well as existing version support.
    - Flexibility for future versions of EPUB.
- Scalability and Sustainability:
    - Modular code without duplicated functionality, with generic checks (e.g. does the file exist) separated from format-specific and schema-specific checks.
    - A structure which enables easy creation of new checkers.
    - A structure which makes community contributions simple and clear, especially including the simple creation of new tests.
    - No reinvention of the wheel. Where open source validators already exist, are easily integrated, and are of high quality, they shall be used. This includes everything from CSS validation to binary image file verification. Where limitations in the external validators cause problems, the first step shall be submitting bug reports or patches, with fallback to replicating the functionality used only as a last resort.
    - A clean, well-documented, versioned API.
    - Human-readable, comprehensible code, either with self-documenting code (e.g. adding meaningful labels to the translation string names) or via comments where self-documenting functionality is inappropriate.
- Tests:
    - A more comprehensive test suite, including unit tests as well as both feature and failure tests.
    - New tests for complex features interactions (e.g. JavaScript, CSS, SVG, WebGL).
    - One EPUB per test.

    - Testing with exploded EPUB (except where tests are validating checkers for package integrity) for ease of editing
- Ease of implementation:
    - The ability to be integrated into Java, Python, and JavaScript code.
    - Executables for Windows and OS X.
    - An clear way to integrate with existing EPUB libraries (e.g. Ace by DAISY, Ebooklib).
    - A modular way Epubcheck users can add local-only tests to their local instance.
    - A simple, documented way Epubcheck users can configure their local settings by either configuration file or command line settings.

The existing tool, which has grown organically over the years, needs a large-scale refactor if not a complete rewrite. There are large swaths of code that are duplicated multiple times across the tool. The API needs to be versioned, stable, and documented, with consistent reporting that can be either machine-or human-readable, Naming conventions and encodings must be made consistent. A substantial number of the checkers are in a secondary package which uses different testing mechanisms and runs a second pass of parsers.

Proposed Implementation

The Contractor will use GitHub for publishing and sharing all code, under the open source license of the Client's choice.

If the Client has developers who wish to provide detailed technical code review, the Contractor is happy to work with the Client's code reviewers.

Option 1: Initial rewrite, Python

Although the existing codebase creates a complete, robust Epubcheck, using Java dissuades many of the possible contributors to the project. Ideally, Epubcheck will be written in an easy-to-learn language which is broadly used among developers in the publishing sector. A scan of GitHub projects confirms the anecdotal data the Contractor has gathered from conversations in the EPUB open source community: the plurality of open source EPUB development is in Python, followed closely by JavaScript.

We propose rewriting Epubcheck in Python. Though several modern software projects in this area are written in JavaScript (most importantly Ace, by DAISY, which ideally should have easy interaction with Epubcheck), Python is an ideal language for evaluating the validity of XML and HTML files. Python has a long history with this problem space, and robust, well-tested toolsets. It is low overhead to learn, and has wide adoption among developers in the publishing community. For a task which is, at its core, about validating and verifying HTML and XML files, Python is the optimal language.

It's appealling to rely on the existing code, but we believe that is a sunk cost fallacy. A rewrite in Python will not waste the effort of prior developers, whose algorithms and best practices will be translated into a new codebase.

This solution will be built with Python as well as widely-used open source libraries. It will use test-driven development practices.

Milestones:

1. Minimal framework that can open an EPUB, identify its version, and perform basic validation, with tests.
2. Replication of the existing base set of checkers, with tests.
3. Support for EDUPUB, EPUB Dictionaries and Glossaries, EPUB Indexes, and EPUB Previews.
4. Support for EPUB 3.2.
5. A versioned API.
6. Executables for various platforms and libraries for various languages.
7. Documentation.
8. Additional tests as needed.

Option 2: Initial refactor, Java:

Alternatively, the Contractor could do the necessary refactoring, upgrading, and maintenance in Java, in order to maintain the existing knowledge base among current contributors, and in order to make use of existing code and spend less time on rewrites. However, we believe this would retain an existing limitation: lack of widespread Java knowledge among potential contributors. Additionally, this will retain existing licensing problems with legacy code which was contributed without appropriate licenses.

This solution will be built with the existing toolset (e.g. Java, Maven), though will likely incorporate some modern libraries to replace legacy code.

Milestones:

1. Identification of replicated and defunct code.
2. Refactoring to remove defunct code and modularize replicated code.
3. Refactoring to reorganize the remaining checkers, merge them into a single package, and reorganize according to principles identified in the work plan.
4. Refactoring the test suite according to the work plan.
5. Support for EPUB 3.2.
6. A versioned API.
7. Executables for various platforms.
8. Documentation.
9. Additional tests as needed.

Rates and hours

All time estimates are considered to be rough estimates. The Contractor will communicate with the Client if discoveries during the work will cause substantial changes to the time estimates.

The Contractor will communicate with the Client on a weekly basis to give an update on work progress and milestones reached. If extensions are necessary, this will be negotiated between the Client and the Contractor in writing.

Option 1: Initial rewrite, Python

We estimate that the process will take Twenty (20) person-weeks. We propose to perform this work at the flat rate of $60,000 dollars.

The Client will invoice the Contractor at the end of each calendar month during the work period of five months, for an amount equal to one fifth of the flat rate: $12,000. The Client will pay within 30 days by either EFT or check.

Option 2: Initial refactor, Java:

We estimate that the process will take Seventeen (17) person-weeks. We propose to perform this work at the flat rate of $50,000 dollars.

The Client will invoice the Contractor at the end of each calendar month during the work period of four months, for an amount equal to one quarter of the flat rate: $12,500. The Client will pay within 30 days by either EFT or check.

Ongoing maintenance:

Ongoing maintenance will include bug fixes, new features, new checkers, new tests, and modifications as necessary to adapt to changes in the EPUB specification. The Contractor will provide:

- A documented method for submitting bugs and feature requests.
- Two-day turnaround on initial response to bugs (United States business-day calendar).
- Two-day turnaround on initial response to submitted code (United States business-day calendar).
- Participation in appropriate W3C task forces, community groups, and other discussion forums.
- Monitoring of the appropriate IRC channels, and participation in discussion with people reporting bugs, requesting help, and submitting code.

We propose to commit to an agreed-upon number of hours per calendar month for ongoing maintenance, at the rate of $85 per hour. If the Client and the Contractor agree that an excess of the agreed-upon time is necessary in a calendar month, that excess time will be billed at $85 per hour for each additional hour.

Company Overview

History

Suberic Networks has provided hosting solutions for local organizations and nonprofits since 1997. It has been a full-scale software solutions vendor since 2016. Suberic Networks focuses on building accessible, scalable tools for organizations in the publishing, library, and archives sectors. We have also provided technical and accessibility training for myriad organizations.

- Internet Archive: We developed an open source library to assist in the IA's pipeline from scanned documents and optical character recognition to accessible EPUB 3. Since many of the OCR-produced source documents were quite large, the project

required extensive performance tuning of Python XML parsing, in order to produce fast-runtimes without using too much memory. Although the source documents provided by IA often lack enough information for true accessibility, the provided library maximizes accessible packages and accessibility metadata for the best possible experience. <https://github.com/deborahgu/abbyy-to-epub3/>

- Kirkus Reviews: We developed the Django back end, the reviewer and editor interfaces, and the taxonomy used by Kirkus Collections <https://www.kirkusreviews.com/diversity/>, an exciting set of curated collections and an accompanying toolset in a partnership between Kirkus Reviews and Baker & Taylor. This was an entirely new product for Kirkus, but had to exist, fully-tested, within a weakly-tested legacy codebase with high uptime requirements.

Suberic Networks has two employees and a network of subcontractors who have many years of experience with all elements of EPUB, Epubcheck, and publishing, as well as with Python, Java, and open-source projects. We are located in Arlington, MA.

Contact Information

Deborah Kaplan
Suberic Networks
107 Grafton Street
Arlington, MA 02474
USA
deborah.kaplan@suberic.net
https://suberic.net/

References

Denise Paolucci
Founder
Dreamwidth Studios
Dreamwidth Studios, LLC, PO Box 39608, Baltimore MD, 21212
denise@dreamwidth.org

Vicky Smith
Kirkus Collections and Children's Editor
Kirkus Reviews
Kirkus Media LLC, 65 West 36th St., Suite 700, New York, N.Y. 10018
vsmith@kirkus.com

Brenton Cheng
Senior Engineer
Open Library / Internet Archive
Internet Archive, 300 Funston Avenue, San Francisco, CA 94118
415-561-6767
brenton@archive.org