Individualized Creation of Educational Material for Generation Z

Gregor von Laszewski, Josiah Clemons, Kang Jie Gan,
Naimesh Chaudhari, Geoffrey C. Fox
Indiana University, Bloomington, IN 47408
laszewski@gmail.com

This draft is available at
https://github.com/cyberaide/bookmanager-service/blob/master/paper/vonLaszewski-bookmanager.md
https://cyberaide.github.io/bookmanager-service/vonLaszewski-bookmanager.pdf

***Abstract*—|As the needs of online students rapidly change today, we propose a novel new method for their curriculum needs- |an online Bookmanager service that allows them to customize their learning process. Through a drag and drop Flask model |of book chapters that is hosted on a webservice provider, we allow students to organize their learning as best fits their needs.**

A. Introduction

The pervasive importance of computing and cyberinfrastructure is broadly acknowledged in many areas of commercial, government, and academic endeavors. This is reflected in Indiana University's (IU) activities to build its new Intelligent Systems Engineering (ISE) program with a strong computational and information technology basis and situate it in the School of Informatics, Computing, and Engineering (SICE). As students are changing the curriculum needs to change and integrate modern concepts and practices in a rapid fashion as well. Thus, we must not only support the rapid change of course material but also support the learning modes of Generation Z. Our work reported here is targeting this area, while we have learned from a four-year undergraduate curriculum designed *ab initio* and taught so far to our first two undergraduate classes. We aim to invest it into developing active modules that are customized for Cyberinfrastructure Contributors (CIC) communities nucleated, built, and sustained via the dynamic use of GitHub. This includes content that targets Cloud Computing, Big Data Applications and Analytics, Networking, High-Performance Computing, Artificial Intelligence/Machine Learning, and Information Visualization. The content is exposed in a modular format and reflects the needs of today's tech-savvy students by incorporating successful community-building tools. In particular, for the work described here we leverage GitHub repositories to prepare the content and make them available on-demand for later learners and educators. This has the advantage that via a GitHub-based community model all can contribute to the course improving the text, the software, and providing a set of examples. We hope in this project to show that we can build both learning and sustainability communities by using the proven techniques of the open-source software community.

One of the important observations is that today students in classes have vastly different backgrounds. What may be common knowledge to one student may not be known by others. Thus it is important to be able to customize the learning experience not only to participate in a *standard* class, but be able to adapt the teaching material for individual students into a *personalized* class.

Thus we are in need of a tool that can customize, for example, the creation of material form content located in different GitHub repositories. Such a tool has been written by von Laszewski [1] and has been in use for over a year. However, this tool is command-line based and may not be convenient for beginner students that are not yet familiar with installing programs on their own computers. Therefore this work develops a webserver-hosted framework that allows the creation of individualized course material from templates and convenient graphical web pages with a table of content to be assembled through drag and drop. Moreover, if during the study of the material it is found that material is missing, it can easily be integrated into the individualized course table of contents.

The result of this work is two-fold:

- This service is developed as code and made available at [2].
- We will experiment with pacing the service during a test phase online and have students from class evaluate the tool and its features to obtain feedback for improvements.

B. Summary

We have a number of distributed documentation written by different groups and organizations and users. Although we can put an html link up with the collection, students, researchers and tutorial participants benefit from a tool that integrates all of them into a single document rather than forcing them into an assembly of multiple of links. This allows for a book-like distribution of the content written by various authors and hosted on various GitHub repositories.

In our case it also serves as a mechanism for generating class proceedings of reports developed by students so they can take them "home" after completing class work.

1) The solution: For this problem, we have a simple tool called Bookmanager that creates an epub of documents specified in a yaml table of contents. The documents will be fetched from the url and then the images downloaded and an epub generated. This epub (as long as it is not too big) can then be browsed not only on your computer, but also on tablets and mobile devices, so you can peruse course materials at will as part of your online "book collection".

2) Sample yaml file:

- https://github.com/cyberaide/bookmanager/blob/master/tests/python.yml

3) Summary of Benefits:

- Simple to use
- Leverages Pandoc, so more formats in future will be supported
- Pulls together information from several sources
- An auto-generated title page
- ePublished

C. Idea

We have a tool called "cyberaide bookmanager" [3][1] that can create books based on markdown files that are then distributed online in Github (it is not only for data science and cloud computing, but that is where we have most of the material).

D. Design Aspects

1. We aim to host the creation of such a book in a container and place it online on a cloud or web service. (Containerization is easy and mostly done for the command line, but not for the HTML/GUI portion)
2. We would like to allow people that visit a web page and select class modules that are then automatically integrated in customized books for that student
3. The integration could be as simple as clicking on some chapters in a table of content that we distribute
4. The interface could be enabled with Javascript, where you drag and drop the chapters into a hierarchy with chapter titles that could be added or are derived from the included Markdown
5. We would like to facilitate a Markdown test that prior to the inclusion tests if the Markdown is ok while using Markdownlint (MDL).
6. Finally, an (optional) extension we woud like to include would be the ability to switch to different formats, such as PDF, Word, rst, .txt, .org mode (this might be easier as we already use the container Pandoc that supports these formats.)

E. Requirements

Obviously, you need to have some Javascript/HTML/Webpage experience. These days almost all students have them so this is doable. We are open to suggestions for Javascript framework, but hear Electron may be a good choice also so we can host the service in the cloud in addition to running locally.

Other than a manual and programming the project has a 2 page report requirement in markdown not LaTeX. The reason this is so low is that you spend more time on programming than writing the report. You should think of the report also like a mini manual, so you let the users know why it is helpful and how they can use it.

F. Bookmanager

Bookmanager is a tool to create a publication from a number of sources on the internet. It is especially useful to create customized books, lecture notes, or handouts. Content is best integrated in Markdown as this format is able to produce the output very quickly. At present we only produce epubs, but it will be easy to also create PDF, HTML, Word, ODT and others in the future. As we use Pandoc we can support the formats that are usually used in conjunction.

1) Implemented Features::

- Table of contents with indentation levels can be specified via YAML
- Special variable substitution of elements defined in the YAML file
- Documents are fetched from Github
- The documents will be inspected and the images found in them fetched (we assume the images are related to the document, HTTP links will not be modified)
- Automatic generation of a cover page
- Output is generated in a destination directory

2) Planned enhancements::

- integration of References via Pandoc Citeref
- integration of Section, Table, Image references via pandoc crossref

If you like to help get in contact with Gregor von Laszewski laszewski@gmail.com

```
$ pip install cyberaide-bookmanager
```

3) Usage: The manual page is listed in Section -G

4) Cover Page: Bookmanager can create a simple cover page for you.

The example is given at

- https://github.com/cyberaide/bookmanager/blob/master/tests/example/cover.png

5) Example creation:

```
$ git clone https://github.com/cyberaide/bookmanager.git
$ cd bookmanager
$ pip install -e .
```

Introduction to Python
for Cloud Computing

Gregor von Laszewski
Editor

laszewski@gmail.com

April 13, 2019 - 12:20 PM

https://github.com/cyberaide/bookmanager

Figure 1: Cover Page

```
$ bookmanager tests/python.yaml get
$ open dest/book.epub
```

6) Requirements: Book manager requires the existence of some cloudmesh yaml files, In future releases we intend to remove them. Simply run

```
$ mkdir -p ~/.cloudmesh
$ wget -P ~/.cloudmesh https://raw.githubusercontent.com/cloudmesh/cloudmesh-configuration/master/cloudmesh/
```

In addition an up-to-date version of Pandoc is required. Please consult with the Pandoc documentation on how to do this. Unfortunately the versions distributed with Ubuntu are outdated. On Ubuntu you can say:

```
wget -q https://github.com/jgm/pandoc/releases/download/2.7.2/pandoc-2.7.2-1-amd64.deb
sudo dpkg -i pandoc-2.7.2-1-amd64.deb
pandoc --version
```

We recommend Pandoc version 2.7.2.

7) Example YAML file: The example in the appendix Section I shows a YAML file including a table of contents that is used to assemble the document from Github locations.

8) Automated Github links: It is possible to replace the local link that will be added to the files with a link to a Github repository. At this time this is only supported for documents that are in the same repository.

Simply add the following in case your local files are in `../chapter`. While specifying it in the `base`. This variable specifies the link to the source. The variable "github" will be used to replace the base with a link to the Github repository.

```
file:
  "github": "https://github.com/cloudmesh-community/book/l
  "base": "../chapters"
```

9) Links:

- Bookmanager home page
- [3] Bookmanager on Pypi
- [1] Bookmanager on GitHub
- [2] Bookmanager Service on Github
- Example Yamle file

G. Manual Page

```
bookmanager -- a helper to create books
               from markdown files in a yaml
               TOC.

Usage:
  bookmanager version
  bookmanager YAML cover
  bookmanager YAML get [--format=FORMAT]
                       [--force]
  bookmanager YAML download
  bookmanager YAML level
  bookmanager YAML epub [--force]
  bookmanager YAML pdf
  bookmanager YAML html
  bookmanager YAML docx
  bookmanager YAML check [--format=FORMAT]
  bookmanager YAML urls [--format=FORMAT]
  bookmanager YAML list [--format=FORMAT]
                        [--details]

Arguments:
  YAML   the yaml file

Options:
  -h --help
  -f, --format=FORMAT     [default: markdown]
  -d, --details           [default: False]
```

Description

In principle, you only need one command at this time. All other commands are available for test purposes.

You can create an ePub with

- `bookmanager YAML get [--format=FORMAT]`

The command searches for all images within the Markdown document and fetches them so the document can be created locally with the images. We assume all images in the md document are for now not specified via HTTP locations but via relative locations.

To view the document use your favourite ePub Reader

Other commands include:

- `bookmanager YAML download [--format=FORMAT]` downloads the URLs into the ./dest directory for local processing
- `bookmanager YAML check [--format=FORMAT]` checks if the URLs in the yaml file exist
- `bookmanager YAML urls  [--format=FORMAT]` lists all URLs of the YAML file
- `bookmanager YAML list [--format=FORMAT]` lists the YAML file

Not implemented are the following features:

- pdf: bookmanager pdf book.yml

YAML Table of Contents format:

The table of contents for the book can be controlled with a simple YAML file that has some specific contextual enhancements. This includes the creation of a BOOK section that has the sections outlined in hierarchical form, and contains chapter and section headers without links that are automatically generated.

Examples of YAML files are provided at

- python.yml

Bugs and enhancement suggestions can be submitted via GitHub

I. Yaml file example

```
metadata:
  image: "cover.png"
  title: "Introduction to Python"
  subtitle: "for Cloud Computing"
  author: 'Gregor von Laszewski'
  subauthor: "Editor"
  email: "laszewski@gmail.com"
  url: "https://github.com/cyberaide/bookmanager"
  description: "Book creator"
  abstract: "my abstract"
  keywords: "pandoc"
  stylesheet: "epub.css"
  dest: "./dest/book"
  filename: "vonLaszewski-python.epub"
git:
  "book": "https://raw.githubusercontent.com/cloudmesh-co
  "credit": "https://raw.githubusercontent.com/cyberaide/b
BOOK:
  - PREFACE:
    - "{git.credit}/disclaimer.md"
  - INTRODUCTION:
    - "{git.book}/prg/SECTION-PYTHON.md"
    - "{git.book}/prg/python/python-intro.md"
    - "{git.book}/prg/python/python-install.md"
    - "{git.book}/prg/python/python-interactive.md"
    - "{git.book}/prg/python/python-editor.md"
    - "{git.book}/prg/python/python.md"
  - LIBRARIES:
    - "{git.book}/prg/python/python-libraries.md"
    - "{git.book}/prg/python/python-data.md"
    - "{git.book}/prg/python/python-matplotlib.md"
    - "{git.book}/prg/python/python-docopts.md"
    - "{git.book}/prg/python/python-cmd5.md"
    - "{git.book}/prg/python/python-cmd.md"
    - "{git.book}/prg/python/opencv/opencv.md"
    - "{git.book}/prg/python/opencv/secchi.md"
  - DATA:
    - "{git.book}/SECTION/SECTION-DATA.md"
    - "{git.book}/data/formats.md"
  - MONGO:
    - "{git.book}/data/mongodb.md"
    - "{git.book}/data/mongoengine.md"
  - APPLICATIONS:
    - "{git.book}/prg/python/fingerprint/fingerprint.md"
    - "{git.book}/prg/python/facedetection/facedetection.md"
```

A. Creation of the PDF

We assume that you have `pandoc`, `bookmanager` [3] and `make` installed on your system.

```
git clone git@github.com:cyberaide/bookmanager-service.git
make doc
```

This will create the PDF in the `docs` folder

B. References

https://github.com/dgraziotin/acm_sig_paper_markdown_pandoc

[1] G. von Laszewski, “Bookmanager source code.” GitHub [Online]. Available: https://github.com/cyberaide/bookmanager

[2] Names misisng and G. von Laszewski, “Bookmanager service source code.” GitHub [Online]. Available: https://github.com/cyberaide/bookmanager-service

[3] G. von Laszewski, “Bookmanager.” PyPi [Online]. Available: https://pypi.org/project/cyberaide-bookmanager/