

The Canvas learning management system and L^AT_EXML

The L^AT_EX workflow is still the best

Will Robertson

July 20, 2018

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Background

- The University of Adelaide recently switched to *Canvas* as its LMS
- The LMS has an API!
- I wanted to sensibly manage 'content'
- Generate both HTML and PDF
- A good excuse to learn \LaTeX XML...

Introduction

What is a learning management system?

This is an LMS

The screenshot shows a web browser window displaying a course page on the University of Adelaide's LMS. The browser address bar shows the URL <https://myuni.adelaide.edu.au/courses/36028>. The course title is **3800_MECH_ENG_X_0001**.

Left Navigation Menu:

- Account
- Dashboard
- Courses
- Calendar
- Inbox
- Commons
- Help

Course Navigation:

- Home (selected)
- Announcements
- Modules
- Collaborations
- Discussions
- People
- Assignments
- Quizzes
- Grades
- Echo 360
- Select Course Readings
- Group Management
- Toolkit
- LMS Analytics
- Outcomes
- Pages
- Syllabus

Recent announcements:

- Mechanical Workshop Submissions**
Dear honours project stude...
Posted on: 28 Jun 2018 at 12:34
- Reminder: prelim peer assessment**
Dear honours project stude...
Posted on: 13 Jun 2018 at 0:00
- Prelim report peer assessment**
Dear honours project stude...
Posted on: 5 Jun 2018 at 17:09

Honours Project (4143A_4143B...)

Welcome to the Mechanical Engineering Honours Project, 2018.

- Please see the [COURSE HANDBOOK](#) for (most) information you need. This is also available in [PDF form](#).
- Ensure that you *subscribe* to the [DISCUSSION FORUM](#) and use it to ask general questions about the course.
- Lectures and deadlines are outlined in the [SCHEDULE](#).

Right-hand Side:

- Import from Commons
- Choose home page
- View Course Stream
- Course setup checklist
- New announcement
- Student view
- View Course Analytics

To do:

- Grade Meeting minutes weeks 1-2
5 points • 12 Mar at 23:59
- Grade Preliminary Report Submission
100 points • Multiple due dates
- Grade Meeting minutes weeks 11-12
5 points • 4 Jun at 23:59
- Grade Meeting minutes weeks 11-12
5 points • 29 Oct at 23:59

Introduction

Specifications for the project

Motivation

1. In the old days, had a monolithic (impossible) Word file
2. Care and maintenance of content
 - Click click click
 - No good tools to manage content globally
3. Re-use of content for a comprehensive PDF
 - 'One stop shop' for reference
 - Handy to be able to distribute
(for students, for supervisors, for accreditation)

Introduction

Specifications for the project

- One source, multiple outputs
- Macros to ensure up-to-date info
 - Names of certain people
 - Due dates
 - Weightings of assessments
- Reliable (easy setup, actively developed, etc.)
- L^AT_EX based (sorry, not sorry)

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

The authoring interface

Which 'HTML' tool to choose?

A number of L^AT_EX to HTML possibilities, including:

- T_EX4ht — (?), in T_EX Live
- l_warp — Lua, in T_EX Live (951 p. manual)
- L^AT_EXML — Perl, not in T_EX Live
- HEVEA — OCaml, not in T_EX Live

(Non-exhaustive list. All are actively developed.)

The authoring interface

Which 'HTML' tool to choose?

- Hopefully the choice is not too important!
- I.e., the conversion aspect should be modular
- I wanted to try $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{XML}$, and I've been happy (enough) with it to date.

Architecture of L^AT_EX XML

Caveat: I'm only a user!

- Reimplements some T_EX scanning in Perl
- Therefore handles basic L^AT_EX 2_ε programming
 - `\newcommand`
 - `\newcounter`, `\stepcounter`
 - etc.
- Provides Perl interfaces to emulate classes and packages

A combination of L^AT_EX 2_ε, Perl, XSLT, HTML, CSS, ...

Anyway, the basics works well:

```
\newcommand\honourscoord{Will Robertson}
```

The authoring interface

L^AT_EX_ML overview

Running L^AT_EX_ML

Two-phase process: latexml then latexmlpost

```
latexml tex/$FILENAME.tex | latexmlpost - \  
  --xsltparameter=SIMPLIFY_HTML:true \  
  --sourcedirectory=tex \  
  --format=html5 \  
  --destination=html/$FILENAME.html \  
  --splitat=chapter \  
  --splitnaming=label
```

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Canvas programming

- A so-called REST API
Wikipedia: 'REST-compliant web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations.'

- Using curl:

```
curl -X GET -H "$CANVASAUTH" $CANVASCOURSE/$1
```

- \$CANVASAUTH = secret token
- \$CANVASCOURSE = URL to course
- \$1 = users or rubrics or assignments?search_term=charter etc.

- E.g.:

```
curl -X GET -H 'Authorization: Bearer 81...Jx'  
https://myuni.adelaide.edu.au/api/v1/courses/36028/assignments
```

Uploading a file to *Canvas*

Clear as mud

```
curl -X POST -H "$CANVASAUTH" "$CANVASCOURSE/files"\
    -F "name=$1" -F "parent_folder_path=upload" > tmp.json ;
URL=`cat tmp.json | jq '.upload_url'` ;
KEYS=`cat tmp.json | jq '.upload_params' | jq -r -j \
    "to_entries | map(\"-F \(.key)=\(.value|tostring) \")|.[]"` ;
echo curl -D response.tmp $URL $KEYS -F file=@$1 | bash ;
LOC=`sed -n -e 's/Location: \(.*/\1/p' response.tmp`;
LOC=${LOC%$'\r'}
curl -X POST -H "$CANVASAUTH" "$LOC" | jq ;
```

- I have forgotten how this works!
- Allows me to, say, upload the typeset PDF automatically after updating content.

Evolution of my support scripts

- Started with `curl`
- A few small-ish Bash functions and scripts
- Have now started with Lua programming
 - Requesting data with many items ('all submitted assignments', say) is returned in multiple 'pages' so iteration is required
 - Much prefer doing real programming not in Bash
 - Start looking into the Lua package ecosystem

Package proliferation: not just a L^AT_EX problem

- luajson by harningt
- lunajson by grafi
- rapidjson by xpol
- dkjson by dhkolf
- jwt-jitsi by pawelgawel88
- JSON4Lua by luarocks
- jwt by olivine-labs
- mjolnir._asm.data.json by _asm
- ngxjsonform by rtbz
- dromozoa-json by moyu
- lua-cjson-ol by olivine-labs
- lua-cjson2 by CriztianiX
- json-lua by jiyinyiyong
- ...

Lua equivalent to curl

```
local http    = require("ssl.https")
local ltn12   = require("ltn12")

local body, code, headers, status = http.request{
    method = "GET",
    url = canvas_url .. req .. "?" .. opt,
    headers = {
        ["authorization"] = "Bearer " .. canvas_token,
        ["content-type"]   = "application/json"
    },
    sink = ltn12.sink.table(canvas_result),
}
```

- This returns JSON, which is 'decoded' into a Lua table.

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

The source

```
\documentclass{report}
\usepackage{latexml}
...
\begin{document}
\title{...}\author{...}\date{...}
\maketitle
\tableofcontents

\input{../texdata/data-limits.tex}
\input{../texdata/data-marks.tex}

\part{Introduction}
\label{part-intro}

\input{../pages/introduction}
\input{../pages/course-schedule}
\input{../pages/week-planner}
```

- Each `.tex` file contains one chapter:

```
\chapter{Introduction}
```

```
\label{introduction}% same as filename!
```

```
...
```

- \LaTeX ML does not convert file by file
- Rather, output HTML is split by chapter
- With consistent naming, this produces one `.html` file per `.tex` file
- A 'table of contents' page is also generated

Top matter

```
<!DOCTYPE html>
<html>
<head>
  <title>XX Chapter title</title>
  <meta http-equiv="Content-Type"
        content="text/html; charset=UTF-8">
  <link rel="stylesheet" href="LaTeXML.css"
        type="text/css">
  [...]
</head>
<body>
```

The generated HTML page

Bottom matter

```
<footer class="ltx_page_footer">  
  [...]  
</footer>  
</div>  
</body>  
</html>
```

'Content'

```
<div class="ltx_page_main">
<header class="ltx_page_header">
  [...]
</header>
<div class="ltx_page_content">
  <section class="ltx_chapter ltx_authors_1line">
    <h1 class="ltx_title ltx_title_chapter">
      <span class="ltx_tag ltx_tag_chapter">Chapter X </span>
      Title of chapter goes here
    </h1>
    <div class="ltx_date ltx_role_creation"></div>
    <section id="S1" class="ltx_section">
      [...]
      ALL THE CONTENT
      [...]
    </section>
  </section>
</div>
```


'Content'

```
<section class="ltx_chapter ltx_authors_1line">
  <h1 class="ltx_title ltx_title_chapter">
    <span class="ltx_tag ltx_tag_chapter">Chapter X </span>
    Title of chapter goes here
  </h1>
  <div class="ltx_date ltx_role_creation"></div>
  <section id="S1" class="ltx_section">
    [...]
    ALL THE CONTENT
    [...]
  </section>
</section>
```

One line of awk

- Snipping is done with

```
awk '/<section.*\>/,/<\section\>/' \  
html/$BASE >> snip/$BASE
```
- Possible to take the raw XML and develop own workflows, but unnecessary for me since I want HTML anyway.

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Not much to say here:

- Using sensible file structure, match up location of images for the PDF on the disk, and for the HTML in the server
- Same thing for files, but currently I don't link files! (TODO)

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Uncertainties

- Maths? Not well supported in *Canvas*, yet. That's okay — for this course I don't need maths.
But this is really important!!
- Bash scripts? One step at a time.
- All Lua, eventually — cross-platform and a bit more sensible.
- How far do I go writing a general *Canvas* interface?

Is \LaTeX XML the right approach?

- The Perl layer is a little foreign to me but appears well-designed.
- The XML output appears highly flexible.
- In the long-run, what does \LaTeX itself need to provide?
- Could \LaTeX XML be written in \TeX itself? Or \LuaTeX ?

Live demo???

(Not sure if this is a good idea...)

Introduction

What is a learning management system?

Specifications for the project

The authoring interface

Which 'HTML' tool to choose?

L^AT_EX_{ML} overview

The *Canvas* programming interface

The generated HTML page

Images and files

Uncertainties

Benefits of scripting *Canvas*

Conclusion

Summary

- From \LaTeX (to PDF, and) to \LaTeXML to *Canvas*.
- \LaTeX is still the pre-eminent document preparation system.
- \LuaTeX opens the door for a tightly integrated approach for bidirectional transfer of information between \LaTeX and *Canvas*.