# LaTeX 2 edX

Ben Weeks
Eric Heubel

Open edX Conference Nov. 2014

# Overview

LaTeX2edX essentially is a translator for creating XML course content from LaTeX source based on the python-based plasTeX compiler.

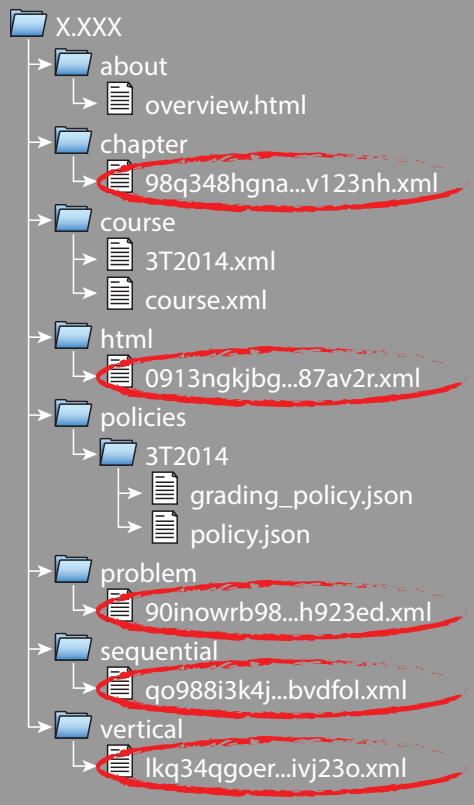
More advanced than Studio's "Advanced Editor" interface

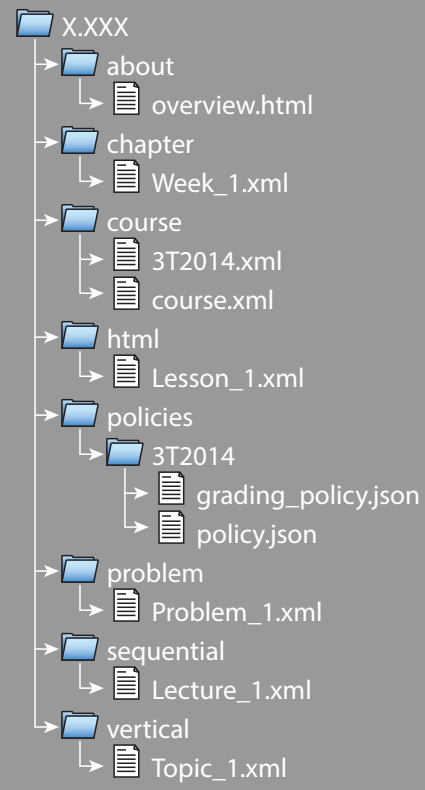
Generates entire course tree with specified filenames.

## Example

A demo LaTeX course may be found at:
`github.com/mitocw/content-mit-latex2edx-demo`
and the course content may be accessed at:
`edge.edx.org/courses/MITx/MIT.latex2edx/2014_Spring`

# Advantages




Editing the exported Studio XML directly is hindered by the auto-naming of files with 32 character alphanumeric strings.

`course.tex` can be converted to several formats: single all-containing course.xbundle XML file, the full course tree (identical to edX), even generate some policy file content.

# Usage

The `latex2edx` executable compiled from python source code uses similarities between XML markup and LaTeX directives to simplify editing edX's OLX.

Much like XML, there is only one course file, and therefore one call to `\begin{edXcourse}{}`.

The entire course may be written in one file, or it may be parsed into components each included into the main TeX document using `\include{}`.

From a terminal prompt ($) in the working directory containing `course.tex`

- `$ latex2edx course.tex`
- TeX is first parsed to XML using plasTeX
- A python script then finishes the translation into OLX

LaTeX2edX is available on GitHub at:
`https://github.com/mitocw/latex2edx`

# Features

Since the source is LaTeX, users can define custom macros using the `\def` command.

It has the additional benefit of being able to generate nicely structure PDF documents for use as handouts.

With this in mind two boolean arguments were added:

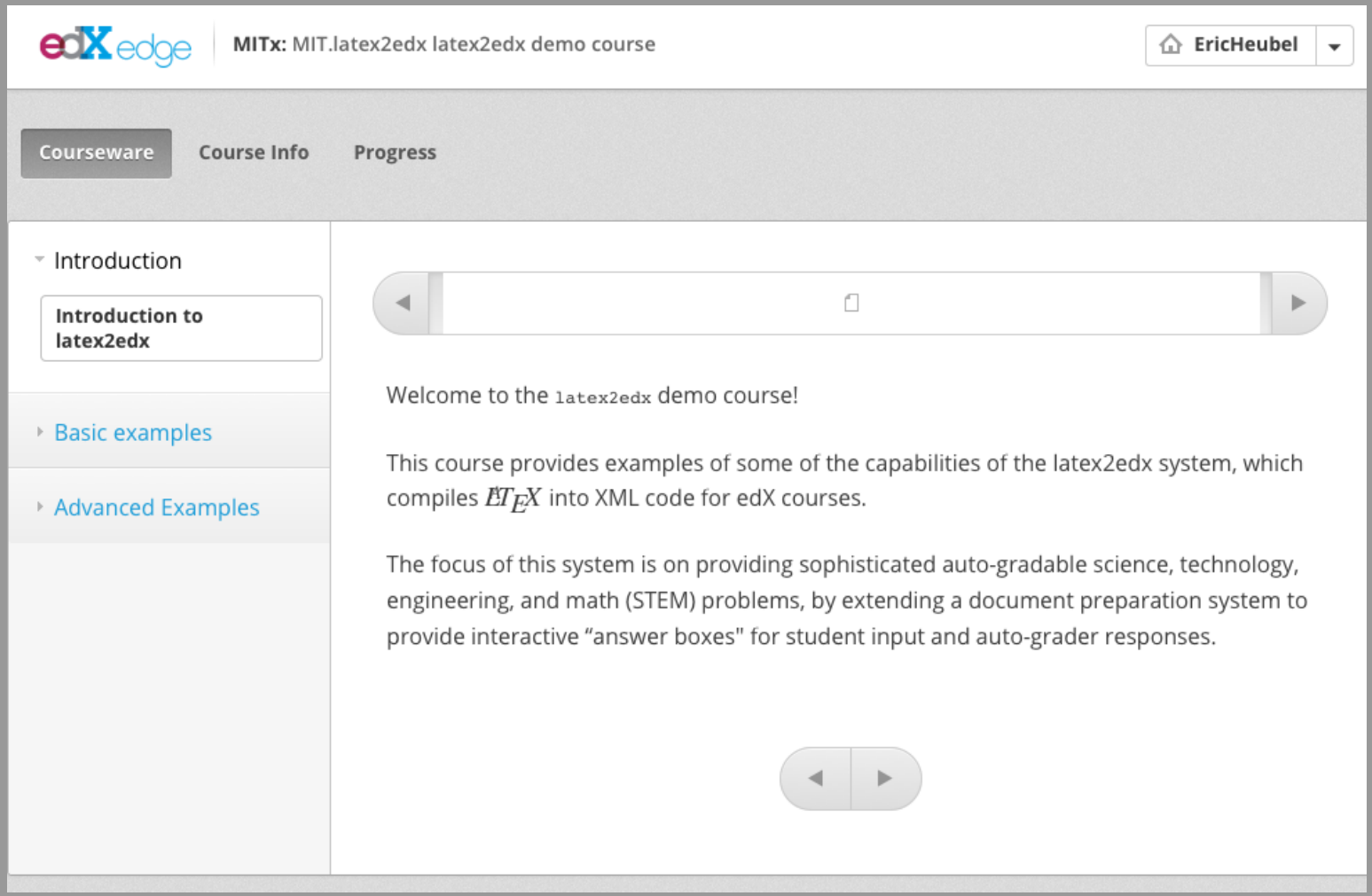
- `\Showanswer` - allows solutions to be hidden when generating a PDF
- `\ShowAbox` - allowing assessment boxes to be hidden from PDF output

This is done using the included style file `edXpsl.sty` found at: `https://github.com/mitocw/latex2edx` under: `/latex2edx/texinputs/`

# course.tex

```
% main latex source file for the latex2edx demo course
% This file can be compiled using
%
%   latex2edx mitx.tex
%
% to generate all the content files for the course.

\documentclass[11pt]{article}

\usepackage{edXpsl} % edX style file

\input{ikedoc} % user-defined macros

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{document}

\begin{edXcourse}{MIT.latex2edx}{latex2edx demo course}[semester="2014_Spring"]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{edXchapter}{Introduction}

\begin{edXsequential}{Introduction to latex2edx}

\begin{edXvertical}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{edXtext}{intro}

Welcome to the \texttt{latex2edx} demo course!

This course provides examples of some of the capabilities of the
latex2edx system, which compiles \LaTeX\ into XML code for edX courses.

The focus of this system is on providing sophisticated auto-gradable
science, technology, engineering, and math (STEM) problems, by
extending a document preparation system to provide interactive
```

# On Edge


edX edge
MITx: MIT.latex2edx latex2edx demo course
EricHeubel
Courseware
Course Info
Progress
Introduction
Introduction to latex2edx
Basic examples
Advanced Examples
Welcome to the latex2edx demo course!
This course provides examples of some of the capabilities of the latex2edx system, which compiles LaTeX into XML code for edX courses.
The focus of this system is on providing sophisticated auto-gradable science, technology, engineering, and math (STEM) problems, by extending a document preparation system to provide interactive “answer boxes" for student input and auto-grader responses.

# Contribute!

Please feel free to contribute to the repository.
`https://github.com/mitocw/latex2edx`

Notes:

- `edXpsl.sty` uses LaTeX3 directives
- Macros are defined in `edXpsl.sty` for PDF output
- A corresponding `plastexpy/edXpsl.py` file AND `render/edXpsl.zpts` OR `render/math.zpts` for XML interpretation.
- Post-XML manipulations are done in `main.py`