The Leeds BookML guide

Vincenzo Mantova

24th July 2024

Abstract

A self-contained guide to BookML as used at the University of Leeds: how to convert (virtually any) LaTeX file to zip and SCORM packages with both HTML and PDF versions of all files.

Contents

1 Installation

1.1 Prerequisites

Full list of prerequisites

- LaTeXML (minimum 0.8.5, recommended 0.8.6 or later)
- Any image handling: the Perl module `Image::Magick`
- Support for EPS, PDF images: Ghostscript
- BookML images: Ghostscript, `latexmk`, `preview.sty`, `dvisvgm` (minimum 1.6, recommended 2.7 or later)
- Automatic PDF, HTML, and zip creation: GNU Make, `latexmk`, `zip`, optionally `texfot`

The packages `latexmk`, `preview.sty`, `dvisvgm` and `texfot` are distributed by MiKTeX, TeX Live, and virtually all Linux distributions.

For the rest of the software, follow the instructions below.

macOS (MacPorts)

- Open the Terminal app.
- Run `xcode-select --install` to get the Command Line Developer Tools.
- Install MacPorts as per its official instructions **from point 2** (no need for full Xcode!).
- If you use MacTeX, run:

  ```
  sudo port install LaTeXML +mactex
  ```

  For the Ghostscript prerequisites, depending on your specific version of MacTeX, you may need to install the Ghostscript-Extras package.

  Otherwise

  ```
  sudo port install LaTeXML
  ```

- If you use MacPorts for LaTeX, you can add the optional packages via:

  ```
  # for automatic PDF, html, zip creation
  sudo port install dvisvgm latexmk
  # for BookML images (preview.sty)
  sudo port install texlive-latex-extra
  # for texfot (to reduce latex output during PDF creation)
  sudo port install texlive-bin-extra
  ```

- To upgrade:

  ```
  sudo port selfupdate
  sudo port upgrade outdated
  ```

- The remaining packages (e.g. GNU Make) are already available.

macOS (Homebrew)

The Homebrew version has some packaging issues around ImageMagick and I **strongly recommend** you use MacPorts, but if you really want to, run

```
brew install latexml
```

Functionality related to images will likely be broken.

Linux Debian-based (Ubuntu, Debian, Mint, etc)

Download the package for the **future** Ubuntu releases at https://launchpad.net/ubuntu/+source/latexml. At the time of writing, this is `latexml_0.8.8-1_all.deb`. Install `ghostscript`, `make`, `latexmk`, `dvisvgm`, `preview-latex-style`, `texlive-extra-utils` (for `texfot`), `zip`, according to your needs.

```
sudo dpkg -i latexml_0.8.8-1_all.deb
sudo apt -f install
sudo apt install ghostscript make latexmk dvisvgm preview-latex-style texlive-extra-util
```

Linux RPM-based (Red Hat, CentOS, AlmaLinux, etc)

Not figured out yet!

Linux School PC (presumably only desktop connected via cable)

Run the following each time you open a new terminal:

```
module load latexml
module load texlive
```

If it does not work, run the following command and try again:

```
module use /apps/linsw1/modulefiles/7/
```

Everything else should already be available.

Windows (AppsAnywhere — easiest, now works offline too)

By far the easiest method. It also work in the Windows Virtual Desktop (very slowly!).

- Make sure to have AppsAnywhere installed.
- Install Ghostscript, ImageMagick, MiKTEX, StrawberryPerl.
- Open StrawberryPerl and run:

Windows (AppsAnywhere — easiest, now works offline too) (cont)

```
cpanm --verbose Image::Magick
cpanm --notest --verbose LaTeXML
```

Everything else should then be available. Re-do all of the above to update. You may have to reinstall the apps every one or two months (open the CloudPaging Player to check their status).

Note: if `Image::Magick` fails complaining about needing a C++ compiler, run

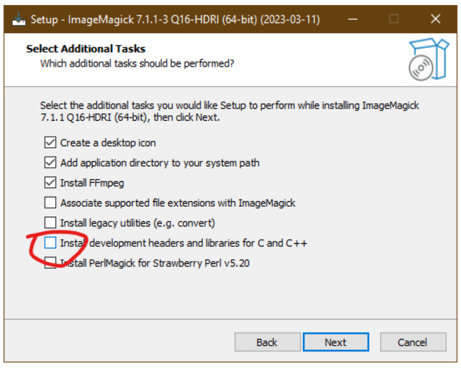
```
cpanm --build-args=CC=c++ --verbose Image::Magick
```

You may also need to add `--notest` if some tests fail (this usually happen when `Image::Magick` is a few versions behind ImageMagick).

Windows (with admin rights)

For University laptops: you can gain admin rights by right-clicking an the installer and choosing "Request Run as Administrator".

- Install StrawberryPerl `64bit` version.
- Install ImageMagick `x64-dll`. During installation, enable 'Install development headers and libraries for C and C++':



Setup - ImageMagick 7.1.1-3 Q16-HDRI (64-bit) (2023-03-11)

Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing ImageMagick 7.1.1 Q16-HDRI (64-bit), then click Next.

- ☑ Create a desktop icon
- ☑ Add application directory to your system path
- ☑ Install FFmpeg
- ☐ Associate supported file extensions with ImageMagick
- ☐ Install legacy utilities (e.g. convert)
- ☐ Install development headers and libraries for C and C++
- ☐ Install PerlMagick for Strawberry Perl v5.20

  Be very careful **not** to choose 32bit, portable, or static variants.

- Install Ghostscript 64bit.
- In StrawberryPerl, run

```
cpanm --verbose Image::Magick
cpanm --notest --verbose LaTeXML
```

Everything else should then be available. Re-do all of the above to update.

Note: if `Image::Magick` fails complaining about needing a C++ compiler, run

Windows (with admin rights) (cont)

```
cpanm Image::Magick --build-args 'CC=g++' --verbose
```

You may also need to run `cpanm --notest --verbose Image::Magick` if some tests fail (this usually happen when `Image::Magick` is a few versions behind ImageMagick).

Windows (without admin rights)

Install the Scoop package manager (no admin required). Then run:

```
scoop install perl
scoop install imagemagick
scoop install ghostscript
cpanm --verbose Image::Magick
cpanm --notest --verbose LaTeXML
```

Everything else should then be available. Use `scoop update --all` to update.

Note: if `Image::Magick` fails complaining about needing a C++ compiler, run

```
cpanm Image::Magick --build-args 'CC=g++' --verbose
```

You may also need to run `cpanm --notest --verbose Image::Magick` if some tests fail (this usually happen when `Image::Magick` is a few versions behind ImageMagick).

Docker/Podman/etc

You can also run BookML using Docker. The default image will compile automatically all the content of the /source folder. For instance:

```
docker run -t -v.:/source ghcr.io/vlmantova/bookml
```

Please note that the Docker image includes a full copy of TeX Live 2021.

1.2 BookML

Unzip the `template.zip` file from the latest BookML release. Open a terminal in the directory containing `Makefile`, `template.tex`, and run

```
make detect
```

. . . or `gmake detect` on Windows.

How do I run commands in the terminal?

You simply type them and press `<ENTER>`. To open the terminal in a specific folder:

▶

How do I run commands in the terminal? (cont)

Windows

Install the Windows Terminal. The easiest way is from the Microsoft Store. If the Store is blocked, you can use

```
winget install --id Microsoft.WindowsTerminal -e
```

You will then be able to right-click on the folder and select "Open in Windows Terminal".

macOS

Right-click on a folder and select "New Terminal at Folder".

Linux

Many file browsers have an option "Open in Terminal" when you right-click on a folder.

Use `gmake detect` on Windows. You should get something like the following:

```
    Main files: template.tex
        BookML: v0.4.3 OK
      GNU Make: 4.4.1 OK
           TeX: MiKTeX 24.1 OK
          perl: v5.38.0 OK (optional)
       LaTeXML: 0.8.7 OK
 Image::Magick: 7.1.1 OK (required for any image handling)
   Ghostscript: 10.02.1 OK (required for EPS, PDF, BookML images)
       dvisvgm: 3.1.1 OK (required for SVG, BookML images)
dvisvgm/libgs: 9.25 OK (required for SVG, BookML images)
       latexmk: 4.82a OK
        texfot: 1.48 OK (optional)
   preview.sty: 13.3 OK (required for BookML images)
           zip: 3.1b OK
```

Anything missing will show a red `NOT FOUND`. If the version is too old, there will be a red or a yellow prompt to upgrade.

First try? Run `make` (or `gmake` on Windows). After a bit, you will find a folder `template` and two zip files `template.zip`, `SCORM.template.zip`. Please open `template/index.html` and verify that it looks as you would expect. The zip files are set up for upload on Minerva.

To update BookML, simply replace the `bookml` folder with the content of a newly downloaded `bookml.zip`.

2 Converting your files

2.1 How to convert

Once your are satisfied that the template is working, drop your own files next to `template.tex`: each file containing `\documentclass` will be treated the same as `template.tex`, and will be compiled to PDFs, SCORM and zip packages. Run `make detect` again, and check that 'Main files' contains such new files.

There is now a reasonable chance that your files are already working (unless you are using Ti*k*Z, which needs some care), but before your first try, you should truncate your files with an early `\end{document}`, as compilation times can be long and most errors will originate in the preamble anyway.

To compile, do the following:

- Run

  ```
  make
  ```

  On Windows, it is `gmake` rather than `make`.

  After a bit, you will have files named like `lecturenotes.zip`, `SCORM.lecturenotes.zip`. The former works with 'Upload Zip package'; the latter with 'Create SCORM package'.

  To generate one particular package, for instance the one for the file `lecturenotes.tex`, run

  ```
  make lecturenotes.zip
  ```

  or, if you are using SCORM packages,

  ```
  make SCORM.lecturenotes.zip
  ```

- When you change a file and want to regenerate the zip files, just run `make` again. Only the files that need updating will be recompiled.

- You may occasionally get errors like

  ```
  No rule to make target 'bookml-short-guide.toc', needed by 'auxdir/bookml-short-guide
  ```

  If that happens, run `make` a second time. If the error is still there, run `make clean-aux`, or delete the `auxdir` folder, to reset BookML.

What is `make` doing? Step-by-step look under the hood

Each time you call `make`, it does the following.

0. Read the file `Makefile` in the folder you are in. That file will import instructions from the `bookml` folder about what to do next.

1. Check all `.tex` files in the folder and find the ones containing `\documentclass`.

   In `example.zip`, it finds `main.tex` and `secondfile.tex`.

What is `make` doing? Step-by-step look under the hood (cont)

2. Arrange to generate two 'targets', a zip package and a SCORM package, for every such file in step 1.

   In the example, the targets are `main.zip`, `SCORM.main.zip`, `secondfile.zip`, `SCORM.secondfile.zip`.

3. Check if the targets of step 2 exist, and if so, if any of their 'prerequisites' are newer, in which case the targets must be updated. The prerequisites themselves are checked recursively to see if they also need to be updated. The prerequisite chain looks like '`zip => html => xml => pdf => tex`'.

   In the example, on your first try, Make will follow the following prerequisite chain: `main.zip => main/index.html => main.xml => main.pdf => main.tex`. It will then build the prerequisites backwards until all is in place to create `main.zip`. Likewise for the other targets.

4. Build the PDF of each LaTeX file found in step 1, using `latexmk` to run `pdflatex`, `makeindex`, `bibtex` and similar as many times as necessary; record which files are `\input`'d and mark them as prerequisites for step 3.

   In the example, it builds `main.pdf`, `secondfile.pdf`, and marks `chapter1.tex` as prerequisite of `main.pdf`. Any update to `chapter1.tex` will cause `main.pdf` to be updated next time you call `make`.

5. Call LaTeXML to build an XML file from each file in step 1.

   In the example, `main.xml`, `secondfile.xml`.

6. Try to build or update the alternative formats requested using `\bmlAltFormat`. For now, BookML only knows how to build PDF files from LaTeX files that have the same name; if you need other formats, you need to build them yourself, or add the relevant instructions (called 'recipes') in the `Makefile`.

   In the example, `main-sans.pdf` and `main-sans-large.pdf` are the alternative formats requested in the preamble of `main.tex`, on top of `main.pdf` which has been built already. Since the example contains `main-sans.tex`, `main-sans-large.tex`, BookML will know what to do, and generate the alternative PDFs by repeating step 4.

7. Convert the XML files to HTML, using `latexmlpost`.

   Thus generate the folders `main`, `secondfile`, each containing an `index.html`.

8. Zip the folder and pack the SCORM package (the latter requiring a couple more steps I will not explain), using `zip`

   At last, you will get `main.zip`, `SCORM.main.zip`, `secondfile.zip`, `SCORM.secondfile.zip`.

2.2 Necessary adjustments

Consult `template.tex` for the minimal requirements in the preamble (e.g. you **must** provide a `\title` command). You should create copies of `template-sans.tex`, `template-sans-large.tex` for each of your main files, at least as a baseline. Alternative versions can be customized and removed, as explained in the next subsection. We recommend offering some alternative versions as good practice. You should also follow the key requirements below.

- Use the `babel` package to set the document language (crucial for screen readers to work correctly). For instance:

```
\usepackage[british]{babel}
```

- Set the document metadata in the preamble (essential for proper navigation links, web page titles, SCORM package metadata, and so on).

```
\title{LaTeXML + BookML guide}
\author{Vincenzo Mantova}
```

- Ensure all TEX style formatting commands (`\Large`, `\bf`, . . .) are enclosed between braces, and use LATEX alternatives such as `\textbf{}` when possible. If you see the wrong font in the HTML output, it is likely caused by TEX-style font switches that haven't gone well.

```
{\bf some bold text}    % DO
\bf some bold text      % DON'T
\textbf{some bold text} % BEST
```

- Using Ti*k*Z or XY-pic and an old version of LATEXML (before 0.8.7)? Follow the instruction below right away. Without it, LATEXML will take several minutes longer (regardless of the size of the file) and often produce broken images. Since version 0.8.7, LATEXML has become more capable, and 0.8.8 can produce excellent Ti*k*Z pictures as well as `tikzcd` diagrams.

2.3 Alternative formats

To create an alternative PDF meant to be included in the same SCORM or zip package (such as a large print PDF), for instance for the `lecturenotes.zip` package, add the following to the preamble of `lecturenotes.tex`:

```
  \usepackage{bookml/bookml} % if not already in your preamble
  \bmlAltFormat{lecturenotes.LARGE.pdf}{PDF (large print)}
```

and create a corresponding `lecturenotes.LARGE.tex` that

- does **NOT** contain `\documentclass` (or it will be compiled into its own SCORM and zip packages);
- configures e.g. different fonts and margins, then call `\input{lecturenotes.tex}`.

BookML will automatically compile and include `lecturenotes.LARGE.pdf` in your final outputs.

Consult `template.tex`, `template-sans.tex`, `template-sans-large.tex` for some simple techniques to achieve this.

If you instead want **distinct** SCORM and zip packages, for instance compile a problem sheet both with and without solutions, explore `example.zip` to see some possibilities.

2.4 Other adjustments

Once you have language and metadata in place, follow the advice below to improve the chances your file will compile correctly, **especially if you use Ti*k*Z or XY**.

Many or complex Ti*k*Z pictures

BookML has a facility to generate the images via LaTeX, bypassing LaTeXML's internal SVG generator, which sometimes fails to create good Ti*k*Z pictures.

If you are running LaTeXML 0.8.8, most Ti*k*Z figures should render reasonably well. If one particular figure causes issues, wrap it in `\begin{bmlimage}` and `\end{bmlimage}`.

If the issues affect several images, you can run *all* Ti*k*Z figures automatically via BookML by doing the following. First, ensure that **all of the** Ti*k*Z code is either in the preamble or between `\begin{tikzpicture}` and `\end{tikzpicture}` (this will speed up compilation). Then add the following to the preamble, after the bookml package:

```
\bmlImageEnvironment{tikzpicture}
\bmlImageEnvironment{tikzcd} % if using tikzcd
\iflatexml
\else
\usepackage{tikz}
% ... ALL of the TikZ-related preamble code here
\fi
% No TikZ commands after this point!
```

See Figure 1, Figure 2.

Other figures, e.g. XY-matrices or `animate`

Most packages producing pictures are not supported by LaTeXML, but you can get around it exactly like with Ti*k*Z:

1. wrap the preamble commands within `\iflatexml\else ... \fi`;
2. if the pictures are their own environments, use `\bmlImageEnvironment` as for Ti*k*Z; see for instance Figure 2;
3. if the pictures are not environments, such as `\xymatrix`, wrap them between `\begin{bmlimage}` and `\end{bmlimage}`: see Figure 3.

You can read more details and see examples in BookML manual.

LaTeXML 0.8.7 has experimental native support for XY-matrices, but it is still not very good.

Alternative text for images (essential for screen readers)

For `\includegraphics` and LaTeXML 0.8.7, just add the option `alt={description of the image}`, as in

▶

Alternative text for images (essential for screen readers) (cont)

```
\includegraphics[alt={Computation of ...}]{figure1}
```

For older versions, and other images such as Ti*k*Z pictures, add `\bmlDescription{text}` right after the image (do not leave an empty line between the image and the text). This will populate the `alt` attribute (or equivalent) and will be read by screen readers in place of the image.

Please keep in mind that sighted users may also benefit from the alternative text. If that is the case, consider using `\begin{figure}` and `\caption{}`, and possibly add a reference in the caption to more explanations (e.g. the definition, a proof, etc.).

Table headers (important for screen readers)

You may need to explicitly mark some table rows as headers. This can be done with some appropriate commands provided by LaTeXML. See Figure 6 for an example.

Split into multiple pages

Add `SPLITAT=chapter` to `Makefile` to split the output in various ways (you can use part, chapter, section. . .). You **must** split long documents, or MathJax will take ages to render your formulas.

You can also specify different splitting strategies for different files: add lines like the following

```
lecturenotes.zip: SPLITAT=section
problemsheet1.zip: SPLITAT=
```

to the end of `Makefile`. The above will split `lecturenotes.tex` into a page per section, and will not split `problemsheet1.tex`. By default, files will be split by chapter.

Disable the bookdown style

If you do not like the bookdown style and prefer a more plain page, like the old `latexmlleeds`, use

```
\usepackage[style=plain]{bookml/bookml}
```

You may wish to disable the additional table of contents, for instance add

```
lecturenotes.zip: LATEXMLPOSTEXTRAFLAGS=--navigationtoc=none
```

at the end of `Makefile`. See ‘Split into multiple pages’ for more details.

Navigation sidebar

This is already included in the bookdown style. If you disable it, but still want the sidebar, add

▶

Navigation sidebar (cont)

```
lecturenotes.zip: LATEXMLPOSTEXTRAFLAGS=--css=LaTeXML-navbar-left.css
```

to the end of `Makefile`.

Customise css (fonts, color)

Create the `bmluser` folder and add CSS files to it.

Embed videos

You can use `\bmlRawHTML{html}` to write arbitrary HTML, in particular output the embedding code for Stream, Mediasite, YouTube, or any other platform. Unfortunately, Microsoft Stream Classic is being phased out and may be broken in some browsers.

See 2.6 below for some reusable code that will also make the video adapt to the size of the page.

Unsupported packages or classes

LaTeXML supports only so many packages (full list). If your package is not supported, or is not supported well, see subsection 2.5.

Resize BookML images (deprecated)

To change the size of the BookML images, use

```
\usepackage[imagescale=2.5]{bookml/bookml}
```

Please note that images are now resized so that the text within the images has roughly the same sizes as the surrounding page. The option `imagescale` will eventually disappear.

For images converted by LaTeXML, starting with 0.8.7, you can try the following experimental and undocumented options:

```
\usepackage[dpi=192,magnify=2,upsample=3,zoomout=2]{latexml}
```

although you are **strongly recommended** to convert EPS and PDF images by yourself separately to SVG, using for instance `dvisvgm`, to get substantially better quality.

Foldable environments

If you want to hide a proof, a solution, or some additional details, you can use the following:

```
\<DETAILS>
  \<SUMMARY>\textbf{Solution.}\</SUMMARY>
```

Foldable environments (cont)

```
  ...details of the solutions...
\</DETAILS>
```

More generally, you can add arbitrary HTML tags by starting them with `\<`, and using the XML syntax (for instance, attributes must have values between quotes, self-closing tags must end with `/>`).

If you like the styling used here, just drop the bookmlleeds-details.css file into the `bmluser` folder.

Customize header and footer (e.g. for copyright notice)

Use the environments `lxHeader`, `lxFooter`.

```
\begin{lxFooter}
  Copyright \copyright{} 2021 Vincenzo Mantova, University of Leeds.
\end{lxFooter}
```

The header is omitted in the bookdown (GitBook) style.

Disable MathJax for an equation

Add `\bmlDisableMathJax` within the equation. Please review the output in Firefox, Safari, and Chrome/Edge (from versions 109).

Other options

Visit the BookML documentation, the LaTeXML documentation, or run `latexml --help`, `latexmlpost --help`, `latexmlc --help`.

2.5 Unsupported packages or classes

If LaTeXML does not recognise a particular package or class, it is sometimes easy to make it work, but it can also be near impossible.

- If you use your own custom-made package or class to keep your favourite packages and options (so essentially a fancy preamble): change its extension to `.tex` and use `\input` instead of `\usepackage`.
- If the package is producing images: use the same strategy as for Ti*k*Z, and remember to provide alternative text.
- If LaTeXML supports a similar package: replace it (for instance, use `actuarialangle` instead of `lifecon`).

- If the package is PDF-specific: use `\iflatexml` and `\newcommand` in the preamble to define macros that do something equivalent, or nothing at all, in HTML. Useful, for instance, for references including page numbers, or setting headers and footers.
- If the class is not supported, tell LaTeXML to use a different class:

  ```
  % just like \usepackage, but before \documentclass
  \RequirePackage{bookml/bookml}
  \iflatexml
  \documentclass[12pt]{book}
  \else
  \documentclass[12pt]{memoir}
  \fi
  ```

  Make sure you pass your class options are consistent, for instance use the same font size.
- If none of the above works, copy the missing macros directly from the package (or class) and add them to your preamble (usually within `\makeatletter` and `\makeatother`). It may work for simple packages, or packages partially supported by LaTeXML.
- Generalising the last idea, you can tell LaTeXML to read the entire package or class as if you were using `\input`. To do this, create a file named `package.sty.ltxml` in the same folder as the `.tex` file (replace `package` with the actual package name, and `sty` with `cls` if dealing with a class):

  ```
  use LaTeXML::Package;
  InputDefinitions('package', type => 'sty', noltxml => 1);
  1;
  ```

  This will instruct LaTeXML to read the content of `package.sty`. This may work wonderfully, or crash miserably.

2.6 Examples

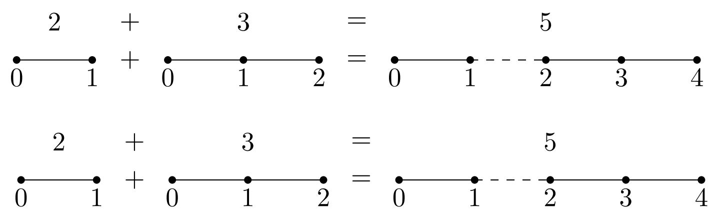


Figure 1: Ordinal sum of 2 and 3 generated in two ways: first via 'bmlimage', then directly by LaTeXML. The latter uses MathJax for the embedded formulas, resulting in improved accessibility but minor alignment issues.

$$\begin{tikzcd}
A \arrow[r, "\phi"] \arrow[dr] & B \\
 & C
\end{tikzcd}$$

$$\begin{tikzcd}
A \arrow[r, "\phi"] \arrow[dr] & B \\
 & C
\end{tikzcd}$$

Figure 2: Example of tikzcd diagram, again generated in two ways via 'bmlimage' and directly by LaTeXML.

2.7 The `latexml.sty` package (advanced)

The `bookml` package automatically loads the `latexml.sty` package (and it includes its own copy if `latexml.sty` is not in your LaTeX installation). `latexml.sty` offers a variety of commands which may be useful. Just open `latexml.sty` (it is very short) to see all the commands, a bit of documentation in the comments, and the occasional example. The source of the LaTeXML documentation contains many examples too. Below are some relevant ones.

- `\lxAddClass{class}` and `\lxWithClass{class}{content}` to add CSS classes to the output;
- `\lxBeginTableHead`, `\lxEndTableHead` and variations to mark table headers and footers (read the `latexml.sty` source for how to use them);
- `\lxContextTOC`, `\lxNavbar{arg}`, `\lxHeader{arg}`, `\lxFooter{arg}` to customise the HTML pages;
- `\lxFcn{code}`, `\lxID{code}`, `\lxPunct{code}` to help LaTeXML understand the meaning of mathematical symbols (for instance, read $f(a+b)$ as 'f applied to $a+b$' as opposed to 'f multiplied by $a+b$'); the wrong interpretation may affect screen readers, so it will need to be addressed, but for now this is too hard to deal with.

3 Uploading to Minerva

3.1 SCORM packages

1. On the front page of your module, under ‘Module content’, click the ⊕ button where you want to insert your item.
2. Choose ‘⊕ Create’.
3. Choose ‘SCORM package’.
4. Choose ‘Upload SCORM package’ and select your `SCORM.<...>.zip` file.
5. Disable ‘Grade SCORM’ and click ’Save’.

Note. Title and abstract of the file will become title and description of the Minerva entry. You will be able to edit the Minerva details right after uploading, if necessary.

3.2 Plain ZIP packages

For the initial permission setup, as well as screenshots of the entire process, consult Chris’ guide. Below is a summary of the day-to-day upload process, once permissions have been set up:

1. On the front page of your module, under ‘Module content’, click the ⊕ button where you want to insert your item.
2. Choose ‘Content Collection’, then ‘Browse Content Collection’.
3. Browse to the folder that has been set up with the appropriate permisions.
4. Click ‘Upload’ and choose ‘Upload Zip Package’.
5. Use ‘Browse Local Files’ to upload your `<...>.zip` file. You **must enable** ‘If selected, the system automatically overwrites the existing file with the same name’.
6. Click ‘Submit’, the choose ‘index.html’ as file to be presented on Minerva.

```
\begin{bmlimage}
  \[ \xymatrix{
        A \ar[rd] \ar^\phi[r] & B \\
                              & C } \]
\end{bmlimage}
\bmlDescription{A, B, C drawn in a triangle with C under B,
  an arrow labelled phi from A to B and an arrow from A to C}
```

$$\begin{tikzcd}
A \arrow[r, "\phi"] \arrow[rd] & B \\
 & C
\end{tikzcd}$$

Figure 3: Example of XY-matrix diagram processed using `bmlimage`.

$$\begin{tikzcd}
A \arrow[r, "\phi"] \arrow[rd] & B \\
 & C
\end{tikzcd}$$

Figure 4: Unsatisfying example of XY-matrix generated directly by LaTeXML.

```
% preamble
\newcommand{\includestream}[2]{
  \bmlRawHTML{<div style="max-width: 1920px; width: 100\%">
    <div style="position: relative; padding-bottom: 56.25\%; height: 0; overflow: hidden;"
      <iframe
        src="https://web.microsoftstream.com/embed/video/#1?autoplay=false\&amp;showinfo=t
        title="#2" style="border:none; position: absolute; top: 0; left: 0;
          right: 0; bottom: 0; height: 100\%; max-width: 100\%;"
        allow="picture-in-picture" allowfullscreen="" width="1920" height="1080">
      </iframe>
    </div></div>}
    Watch \href{https://web.microsoftstream.com/video/#1}{#2}
  }
% document
\includestream{ba6b8866-df29-4dea-a47e-13decc5cd409}{Mock recording for Models and Sets}
```

Watch Mock recording for Models and Sets

Figure 5: How to embed a video. Note that the LaTeX special characters are preceeded by a backslash or the output may be invalid.

```
\begin{tabularx}{\textwidth}{c|X||c}
  \lxBeginTableHead{} Header 1 & Header 2 & Header 3 \\
  \hline \lxEndTableHead{}
  Content & Content & Content \\
  More content & content & content \\
  \hline
\end{tabularx}
\caption{A table}
```

Header 1	Header 2	Header 3
Content	Content	Content
More content	content	content

Figure 6: Mark a table row as header. Read the content of `latexml.sty` for more table-related commands.