Chirun

Release 0.6

E-Learning Unit, Newcastle University

Dec 13, 2021

INTRODUCTION

Fig. 1: (ky-run)

Note: The Chirun project is under active development and so the contents of this documentation is subject to change.

Chirun produces flexible and accessible course notes, in a variety of formats, from LaTeX or Markdown source. It is aimed primarily at notes in the mathematical sciences.

Multiple outputs are supported automatically. So, for example, the same source document can produce a HTML web page that can be easily viewed on desktop, mobile or tablet; slides that can be used to give a presentation or lecture; a PDF ready for printing; a Jupyer notebook; and more.

Chirun can be used to convert either a single input file at a time, or a collection of files combined as a Chirun "course" package. It generates HTML output, which can be uploaded to a web server or VLE for distribution to learners.

The *chirun* tool is provided as a Python package, and can be installed on a Linux, macOS or Windows (via WSL) system running Python. The *chirun* tool provides a command line interface to convert course notes.

Alternatively, a free to use public Chirun builder is available at https://mas-coursebuild.ncl.ac.uk/public/, minimising the need to install software. The public builder produces zipped HTML packages that will need to be hosted for the web by another service.

An open source Chirun LTI provider is also available. The LTI provider manages an interface for uploading notes for conversion, distributing the resulting HTML package, and providing secure access for learners.

The LTI provider is designed to integrate with your institution's VLE, and so an instance must be setup on an appropriate Linux server and registered to work with the VLE by your local IT system administrators.

A sample Chirun course package is provided at https://github.com/chirun-ncl/sample_course, demonstrating the use of both Markdown and LaTeX source and multiple item types. Mathematical notes, embedded content and presentation slides are all included as part of the sample course structure.

CHAPTER ONE

SAMPLE OUTPUT

The result after running this course through Chirun can be viewed online at https://chirun-ncl.github.io/sample_course/

CHAPTER TWO

LICENSING AND OSS STACK

2.1 Chirun

The Chirun Builder Python package is licensed under the Apache License, Version 2.0.

The output content produced by Chirun uses the following third party projects:

- Markdown parsing with Python Markdown
- LaTeX parsing with plasTeX
- PDF output produced with LaTeX
- PDF manipulation with PyPDF2
- Image manipulation with Pillow
- Templating with Jinja
- Jupyter Notebook conversion with nbconvert
- Default theme styles provided by Bootstrap
- Icons provided by Font Awesome
- Slides output provided by Reveal.js
- Further Python libraries and packages, listed in the file `requirements.txt`

2.2 Chirun LTI Provider

The Chirun LTI Provider is licenced used the GNU General Public License v3.0.

In addition to the above, the Chirun LTI provider makes use of the following third party projects:

- Docker & Docker Compose
- Twig
- Confd

CHAPTER

THREE

GETTING STARTED

Chirun converts documents from a LaTeX or Markdown source into an accessible HTML format.

Multiple outputs for the same content can be automatically built. So, for example, a single source document can produce a HTML web page that can be easily viewed on desktop, mobile or tablet; slides that can be used to give a presentation or lecture; a PDF ready for printing; a Jupyer notebook; and more.

Chirun can be used to convert either a single input file at a time, or a collection of files combined as a Chirun "course package".

Running Chirun

There are multiple ways to use Chirun. Usually, you will only need to choose one of the three methods described below. In each case, the output is a directory containing HTML, which can be uploaded to a web server or provided via a VLE for distribution to learners.

- If your institution has already setup an instance of the *Chirun LTI Provider*, we recommend making use of that method, as it integrates directly with your institution's VLE.
- Those unfamiliar with the system or just wanting to convert a few independent documents should use the *Chirun Public Content Builder*.
- More advanced users or those wanting to author many documents in Chirun should choose to use the *Chirun Python Package*.

3.1 Chirun Public Content Builder

A free to use public Chirun package and document builder is available at https://mas-coursebuild.ncl.ac.uk/public/, removing the requirement to install the Chirun software and its prerequisites to your local machine.

Warning: The public builder produces zipped HTML packages that must be downloaded and hosted for the web by some other service. The converted notes expire from the public builder servers after a short while.

3.1.1 Convert a Standalone Document

The Chirun public builder can be used to convert a single document from LaTeX or Markdown source into an accessible web-based HTML package. The process is as follows:

- Visit https://mas-coursebuild.ncl.ac.uk/public/
- Click the button labelled "Choose Files" and select your LaTeX or Markdown files to be uploaded and converted
- (Optional) Click Show/hide settings to tweak the build settings
- Click the button labelled "Upload"

Your document will be converted and the build log showing the output from the Chirun tool will be shown on screen. This output will be useful to help debug if anything goes wrong in the conversion process. For help and advice on Chirun build errors, see the *Troubleshooting* section.

If the conversion is successful, the message `Finished!` will be displayed at the bottom of the build log and the Download Output Package section will be displayed. Two new buttons are presented,

- Clicking "Preview Content" will open your converted document in a new tab. This form of the output is indended only as a preview and will expire after a short while.
- Clicking "Download Package" will download the converted HTML output as a `.zip` file, ready to be distributed to learners, for example by extracting and uploading the content to a web hosting service.

3.1.2 Tweaking the Build Settings

When uploading a single document to the public builder, the build settings that would normally be controlled by populating a Chirun `config.yml` file can instead be tweaked as part of the upload form.

The following settings are available:

- The item type can be changed (see *Content Item Types* for further information)
- For longer `.tex` documents the content can be automatically split on chapter or section
- The item title can be customised
- Showing a sidebar in the default theme can be enabled or disabled
- Building a PDF version of the document can be turned on of off

Note: Multiple source files can be selected for upload. Alternatively, source files can be compressed into a `.zip` file before uploading.

3.1.3 Build Chirun Course Packages

Chirun supports compiling multiple documents (forming a Chirun "course package") into a single output website. The source files are compiled into separate HTML pages and hyperlinked together.

Chirun course packages are controlled by a course configuration file named `course.yml`. To compile a Chirun course package, first compress all the source documents and a valid `course.yml` into a single `.zip` file. Then, follow the instructions in the previous section and upload the `.zip` file to the public builder.

More information on building a valid Chirun `course.yml` file can be found in the *Compile a Chirun Course Package* section.

The *.zip* file will be extracted and the configuration file will be automatically recognised and used. The properties in the `course.yml` file will override the settings selected in the "Show/hide settings" section.

3.1.4 Compile the Sample Course

The Chirun sample course can be compiled with the public builder and is a good place to start if you'd prefer to modify a working template rather than start from scratch.

- The sample course is available on GitHub. First, either download the sample course files or clone the repository using git:

```
git clone https://github.com/chirun-ncl/sample_course.git
```

- Once you have the source files, enter the `sample_course` directory.
- If you'd like to edit the sample course files to make some changes, do so now.
- Compress the contents of the `sample_course` directory into a `.zip` file ready to upload to the public builder.
- Follow the instructions in the section *Convert a Standalone Document*, but upload the compressed `.zip` file for conversion. All other instructions remain the same.

Note:

- Ensure that your `config.yml` file is at the *root* of the `.zip` package uploaded to the Chirun public builder.
- You should not use the "Show/hide settings" section, as the settings there are overridden by the `config.yml` file in the sample course.

3.2 Chirun Python Package

3.2.1 Installation Instructions

Prerequisites

First, prepare the environment by installing prerequisites required for Chirun to function.

Linux (Ubuntu 18.10+)

These instructions are written with Ubuntu in mind, but other Linux system package managers should be able to be used, albeit with different package names.

- Ensure Git, Python3 and virtualenv are installed:

```
apt install python3 python3-virtualenv
```

- Ensure a system TeX distribution is installed, such as TeX Live:

```
apt install texlive-full
```

- Install the packages pdf2svg, pdftoppm, pdftk and libyaml:

```
apt install pdf2svg poppler-utils libyaml-dev pdftk-java
```

MacOS

- Install a system TeX distribution, such as MacTeX from https://tug.org/mactex/
- Install Homebrew by following the instructions at https://brew.sh
- Use the `brew` command to install pdf2svg, pdftoppm, pdftk and libyaml:

```
brew install poppler
brew install pdf2svg
brew install libyaml
brew install pdftk-java
```

- Install virtualenv by running:

```
pip3 install virtualenv
```

- If you are not using the default Apple-provided build of Python 3 (e.g. Python is installed under `/Applications/Python 3.X`, where `3.X` is the version), ensure that the SSL CA certificates are installed by running:

```
sudo /Applications/Python\ 3.X/Install\ Certificates.command
```

Windows

Chirun can be used by following the Linux instructions in WSL.

Further Windows documentation will be added in future.

Quick Installation

Follow these instructions to install the Chirun Python package. If you plan to modify Chirun or perform development work, follow the instructions in the next section instead.

- Ensure you have installed the prerequisites shown in the previous section.
- Create a Python3 virtualenv and activate it:

```
virtualenv -p python3 chirun_env
source ./chirun_env/bin/activate
```

- Install the Chirun Python package:

```
pip install git+https://github.com/chirun-ncl/chirun.git
```

The command `chirun` is now available for use whenever the virtualenv is active. You should now continue to the next section and compile the sample course to ensure everything works.

Upgrade Instructions

Run the following command with the virtualenv active to upgrade the installed version of Chirun:

```
pip install --upgrade git+https://github.com/chirun-ncl/chirun.git
```

Note: You may need to run the above command with an extra `--force-reinstall` argument if the version number has not been changed between updates.

Development Installation

You should only follow these instructions if you plan to modify Chirun or perform development work.

- Create a Python3 virtualenv and activate it:

  ```
  virtualenv -p python3 chirun_env
  source ./chirun_env/bin/activate
  ```

- Clone the Chrirun Python package repository:

  ```
  git clone https://github.com/chirun-ncl/chirun.git
  ```

- Enter the chirun package directory, `cd chirun`
- Install all the requirements:

  ```
  pip install -r requirements.txt
  ```

- Install the chirun tool into your environment:

  ```
  pip install -e .
  ```

The command `chirun` is now available for use. You should now compile the sample course and ensure everything works.

Development Upgrade Instructions

To upgrade the development installation pull the latest changes from this git repository and install any new requirements:

```
cd chirun
git pull
pip install -r requirements.txt
```

3.2.2 Convert a Standalone Document

Chirun can be used to convert a single document in LaTeX or Markdown format as a *standalone* item. The conversion process has two steps,

1. Create a Chirun configuration file
2. Compile the document using the Chirun CLI tool

Create the config.yml file

Take the following template and save it as `config.yml` in the same directory as your document source.

```
author: 'Ann Example'
structure:
  - type: standalone
    topbar: False
    source: source_file.tex
    title: 'Example Document'
build_pdf: true
```

Note: Don't forget to replace the author's name, document title, and the source filename for the document, `source_file.tex`.

Run the Chirun CLI tool

Once your `config.yml` file has been created, run Chirun to convert your document into accessible HTML format:

```
chirun
```

Once the job is complete, the HTML output can be found in the `build` directory. Open the file `build/index.html` to view the results.

Warning: If you see the error message `FileNotFoundError: [Errno 2] No such file or directory` ensure that the file `source_file.tex` exists, or the source property in `config.yml` is updated to match your document's source filename.

Distribution

The contents of the `build` directory can be uploaded to a web space or your VLE to distribute to learners.

3.2.3 Compile a Chirun Course Package

Chirun supports compiling multiple documents (forming a Chirun "course package") into a single output website. The source files are compiled into separate HTML pages and hyperlinked together.

Compiling a course package is very similar to the building a stanalone item. The file `config.yml` controls the structure and appearance of the Chirun course items.

Create `config.yml` with multiple items

Extra documents can be included by populating the `structure` section of the `config.yml` file.

As an example, the following configuration file instructs Chirun to build two items of type `chapter`. An index page hyperlinking the items together will be included as part of the HTML output.

```
author: 'Ann Example'
title: 'Example Course'
structure:
  - type: chapter
    source: item_one.tex
    title: 'Item One'
  - type: chapter
    source: item_two.tex
    title: 'Item Two'
build_pdf: True
```

Save the above example in a file named `config.yml`. Then, in the same directory, add two documents to be included as part of the output produced Chirun. Finally, update the document titles and source filenames in `config.yml` to match your documents and the course package is ready to be built using Chirun.

The Sample Course

A full sample course is provided as a demonstration of how to build a Chirun course package and populate the `config.yml` file with items of various types. The following instructions describe how to obtain and build the sample course.

- Use git to obtain a copy of the sample course package:

  ```
  git clone https://github.com/chirun-ncl/sample_course.git
  ```

- Change into the directory and run `chirun` to build the sample course.
- The HTML output will be in the `build` directory.

3.3 Chirun LTI Provider

The Chirun LTI Provider is designed to integrate with your institution's Virutal Learning Envionment (VLE). Before the Chirun LTI Provider can be used, it must be setup on a server run by your institution and registered with the VLE by your IT System Administrators.

3.3.1 Creating a Chirun Content Item in the VLE

Once the Chirun LTI Provider has been setup and registered with your VLE, you should be able to add a Chirun LTI item to your course. The precise method to add a Chirun LTI item will vary based on the VLE, but the action should be labelled similarly to:

- Add activity or resource
- Add external tool
- Insert LTI item/link
- LTI teaching tool

After a source file(s) has been uploaded by an instructor, students can select the new activity listed in the VLE to access the HTML content output by Chirun.

3.3.2 Uploading Content

The simplest way to use the Chirun LTI Provider is to convert a single document from LaTeX or Markdown source. Chirun will convert the document into an accessible web page, and then display the web page to a learner when they click on the LTI item in the VLE.

The process to upload your content to the Chirun LTI Provider is very similar to uploading content to the *Chirun Public Content Builder*. The major difference is rather than providing a package to download, the LTI Provider will automatically host your resulting package on the web, in a location accessible only to the learners registered for the course.

Follow the instructions below to upload content,

- Create a Chirun LTI activity and access the LTI item as an instructor to load the dashboard
- When there is no content associated with the Chirun LTI item, the instructor dashboard will redirect to the Upload page
- Click the button labelled "Choose Files" and select your LaTeX or Markdown files to be uploaded and converted
- (Optional) Click Show/hide settings to tweak the build settings
- Click the button labelled "Upload"

The following build settings are available:

- The content item type can be changed (see *Content Item Types* for further information)
- For longer `.tex` documents, optionally choose to split the document at certain levels
- The content item's title can be customised
- Showing a sidebar in the default theme can be turned on or off
- Additionally building a PDF version of the document can be turned on of off

3.3.3 Build Log

Once you have uploaded your content it will begin to be converted and you will be redirected to the build log page. The output as Chirun runs and processes your document will be shown on screen.

The build log will help you debug the problem if anything goes wrong in the conversion process. For help and advice on Chirun build errors, see the *Troubleshooting* section.

If the conversion is successful, the message `Finished!` will be displayed at the bottom of the build log.

3.3.4 Student Preview

After your content has been built successfully, you can preview the content as a student by selecting "View Content" at the top of the instructor dashboard, then clicking "View as Student". The converted content will open in a new tab. If your VLE supports masquerading as a learner, you can also open the Chirun activity from the VLE while the student role is active.

The "View All Content" button is used to view all content, regardless of Chirun *Adaptive Release* settings.

3.3.5 Convert a Chirun Course Package

The above section describes uploading a single document for conversion with Chirun. However, Chirun also supports compiling multiple documents (forming a Chirun "course package"). The source files are compiled into separate HTML pages and hyperlinked together.

Using this method the entire uploaded content, made up of multiple source files, can be accessed via a single Chirun LTI item in your VLE.

Chirun course packages are controlled by a course configuration file named `course.yml`. To compile a Chirun course package, first compress all the source documents and a valid `course.yml` into a single `.zip` file. Then, follow the instructions in the previous sections and upload the `.zip` file to the Chirun LTI Provider.

More information on building a valid Chirun `course.yml` file can be found in the *Compile a Chirun Course Package* section.

The *.zip* file will be extracted and the configuration file will be automatically recognised and used. The properties in the `course.yml` file will override the settings selected in the "Show/hide settings" section.

3.3.6 Link to Existing Content

Once you have uploaded content to the Chirun LTI Provider, the same content can be linked to different LTI items. This is useful when combining with *Direct Linking* so that several LTI items point to the same piece of uploaded content, but with each LTI item displaying a different piece of content when accessed by learners.

To link an empty Chirun LTI item to existing content, you will need to know the GUID code for the content. The GUID for existing content can be found on the "Selected Content" dashboard page of the LTI item where they have been successful uploaded.

Once you know the GUID,

- Scroll down the Upload page to the "Existsing Document" section.
- Enter an existing GUID in the box labelled "GUID"
- Finally, click "Submit"

Existing content must have been uploaded by the same user currently logged in.

3.3.7 Adaptive Release

When uploading a split document, or a Chirun "course package" consisting of multiple documents, Chirun allows you to adaptively release content, hiding the content from learners until a certain date/time has passed or the content is manually released.

To access the adaptive release settings, select "Access Control" at the top of the instructor dashboard, then "Adaptive Release". If your LTI item has multiple pieces of content associated with it, they will be shown on the page.

Manually Hide Content

To manually hide a piece of content, locate it in the adaptive release table and select the "Force hidden" checkbox. Then click "Save Schedule". Access to the content will be restricted and the content will not be shown to students in introduction or part pages.

Follow the same procedure, but unticking the box, to make the content available again.

Adaptive Release Schedule

To show a piece of content only after a certain time, only until a certain time, or both, locate the content in the table and populate the "Start Date & Time" and/or "End Date & Time" entries using the date picker.

After selecting the dates and times for all of the rows you want to release by schedule, click the "Save Schedule" button to commit the changes.

- Content with a "Start Date & Time" set will be hidden unless the current date is after the start date.
- Content with an "End Date & Time" set will be hidden unless the current date is before the end date.
- Content with both dates set will only be available between the two dates.

Warning: If you are using a course GUID to populate multiple Chirun LTI items with the same content, you must set the adaptive release setting on every indiviual LTI item in the VLE.

The adpative release settings only apply to the current LTI item, even if there are mutliple LTI items sharing uploaded content.

3.3.8 Public Access

Uploading content to the Chirun LTI Provider generates a content link; the output HTML generated by Chirun is automatically served at that link. By default, the link is secured by the server so that only learners registered for the course on the VLE can access the content.

This means that students must access the content by clicking on the LTI item shown in the VLE – the web URL associated with the content cannot be shared directly.

As an alternative, the content can be made publically available, so that anyone with the URL can access the content.

To enable public access:

- Click "Access Control" at the top of the instructor dashboard
- Then click "Public Access"
- Tick the box labelled "Enable public access to content"
- Finally, click "Update"

Warning: Do not enable Public Access for content that should only be accessed by course learners.

3.3.9 Direct Linking

By default, when a learner clicks a Chriun LTI item with multiple pieces of content available they are first directed to an introduction page. As an alternative, a piece of content can be set as "direct linked". Direct linked items are shown to the learner when they first enter the LTI item via the VLE, instead of the introduction page.

This setting is useful when you have several LTI items pointing to the same piece of uploaded content (through use of the content GUID). The same content can be associated with several LTI items, but by direct linking to different content, each item displaying a different piece of content when accessed by learners.

Warning: Direct linking does not prevent learners accessing other content associated with the LTI item. To restrict access to content, use *Adaptive Release*.

3.4 Troubleshooting

The Chirun project makes use of GitHub Issues for bug reports and discussing issues. You can use the search function on the Chirun issues page to see if your problem has been seen before.

Help! My LaTeX notes won't compile!

Chirun compiles LaTeX documents using the plasTeX Python package. While it supports a wide array of TeX and LaTeX features, not all LaTeX packages are compatible. Complex packages or packages relying on special features of the PDF format must be re-implemented in the Python lanaguge, and this process has not been completed for many packages.

If your document fails to build, the first thing to do is to take a look at which LaTeX packages you are using. Try removing one or more, or simplifying your notes, until things start working.

If your notes build, but parts of the output are broken, you should check indiviual mathematical equations in your document. Chirun renders mathematics on the web with MathJax, and again not all of the features available in LaTeX work fully in MathJax out of the box.

In short, it is a good idea to start with short and simple LaTeX documents and slowly build up complexity once they are building successfully.

I get the error `AttributeError: module 'yaml' has no attribute 'CLoader'`

Reinstall pyyaml, ensuring that it is linked to the system *libyaml* by issuing the command:

```
pip --no-cache-dir install --verbose --force-reinstall -I pyyaml
```

Running makecourse throws an error related to certificate validation on macOS

To fix this problem on macOS run the command:

```
sudo /Applications/Python\ 3.X/Install\ Certificates.command
```

(where `3.X` is your version of Python) to install the appropriate SSL CA certificates. This change allows the headless version of Chromium to download successfully.

CHAPTER

FOUR

CONTENT ITEM TYPES

The Chirun configuration file `config.yml` contains an list of items to be compiled. The top level array property `structure` holds the information, and each item in the list has an associated `type` (and other properties), described below.

With the major exceptions of the `title`, `source` and `content` properties, most item properties are optional and so can be omitted in `config.yml`.

4.1 Chapter

A `chapter` item should be used when including a short simple document, or when including a single chapter of a longer document. The entire document is output as single web page as part of the Chirun output.

4.1.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The file name of the source for the item.	None
`sidebar`	Show the theme's sidebar including ToC?	`True`
`topbar`	Show the theme's topbar at the top of the page?	`True`
`footer`	Show the theme's footer at the bottom of the page?	`True`
`buildPDF`	Build a PDF for this item?	`True`
`js`	An array of paths to JS files to include in HTML output	`[]`
`css`	An array of paths to CSS files to include in HTML output	`[]`

4.1.2 Supported Source Formats

- LaTeX with *Chirun LaTeX Package*
- Markdown with *Chirun Markdown Extensions*

4.1.3 Outputs

- HTML web page
- PDF document

4.1.4 Example

```
structure:
  - type: chapter
    title: Some LaTeX notes
    source: latex_notes.tex
    sidebar: False
```

4.2 Part

A `part` item allows you to group items into a collection. The items contained in the part are built as normal, but are grouped together in the output hierarchy. Chirun will render a part item as a list of links to the containing items, labelled by the linked item titles. Part items can be nested.

An example use of the part type could be to combine several `chapter` items into an organised collection. Another use could be to organise several items of content into weekly blocks.

4.2.1 Item Properties

Property	Description	Default Value
`title`	The title for the part collection.	None
`content`	An array of items to be built as part of this collection.	None

4.2.2 Outputs

- HTML web page

4.2.3 Example

```
structure:
  - type: part
    title: Some Notes
    content:
      - type: chapter
        title: LaTeX Chapter 1
        source: latex_notes_1.tex
      - type: chapter
        title: LaTeX Chapter 2
        source: latex_notes_2.tex
```

4.3 Document

A *document* item is similar to a *Chapter* item, but intended for longer documents or books.

A document item allows for content to be split at the chapter or section level, building up a hierarchy of part items and chapter subitems automatically. Both the HTML and PDF outputs are split as part of this process.

Note: Currently, the document item type only works with LaTeX source documents. For longer Markdown documents, split up your content into multiple files and build the structure manually using part and chapter item types.

4.3.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The file name of the source for the item.	None
`splitlevel`	At what *level* should the document be split?	`0`
`sidebar`	Show the theme's sidebar including ToC?	`True`
`topbar`	Show the theme's topbar at the top of the page?	`True`
`footer`	Show the theme's footer at the bottom of the page?	`True`
`buildPDF`	Build a PDF for this item?	`True`
`js`	An array of paths to JS files to include in HTML output	`[]`
`css`	An array of paths to CSS files to include in HTML output	`[]`

4.3.2 Split Levels

Description	Split level
Entire Document (no splitting)	-2
Part	-1
Chapter	0
Section	1
Subsection	2

4.3.3 Supported Source Formats

- LaTeX with *Chirun LaTeX Package*

4.3.4 Outputs

- HTML web page
- PDF document

4.3.5 Example

```yaml
structure:
  - type: document
    title: Some LaTeX Book
    source: latex_book.tex
    splitlevel: 0
```

4.4 Standalone

A `standalone` item type is the same as a *Chapter* item type, but intended for when there is only a single piece of content being built.

Content built with the standalone item type becomes the index page for the course, and no introduction page is generated.

4.4.1 Example

```yaml
structure:
  - type: standalone
    title: Some LaTeX Notes
    source: latex_notes.tex
```

4.5 Introduction

An `introduction` item produces the index page for the course. The index page shows some basic information about the course, such as the author, course title, year and code (if populated) course properties set in the `config.yml` file. In addition, the other content items described in the `structure` property are linked to from this introduction page.

A source document can be optionally associated with the introduction item to display content as part of the introduction page. Alternatively, text content can be set as properties on the introduction item directly.

If no introduction or standalone item is included in the course structure, a basic introduction item is automatically generated.

4.5.1 Item Properties

Property	Description	Default Value
`source`	The source file name for optional content.	None
`location`	Display the `source` content above or below the content links?	`below`
`leading_text`	Optional text shown under the title and author.	None

4.5.2 Supported Source Formats

- LaTeX with *Chirun LaTeX Package*
- Markdown with *Chirun Markdown Extensions*

4.5.3 Outputs

- HTML web page

4.5.4 Example

```
structure:
  - type: introduction
    leading_text: "This is a short paragraph that will be
    inserted into the introduction page, just under the author and year."
  - type: chapter
    title: Some LaTeX notes
    source: latex_notes.tex
```

4.6 Slides

The `slides` item type is intended to be used for content primarily presented as a presentation and/or set of slides. A slides item is built as a *Chapter*-style HTML web page, a slides pack for presenation, and a printable PDF output.

The precise output format for a slides item depends on the source format.

4.6.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The file name of the source for the item.	None
`js`	An array of paths to JS files to include in HTML output	`[]`
`css`	An array of paths to CSS files to include in HTML output	`[]`

4.6.2 Supported Source Formats

LaTeX with the Beamer Package

LaTeX documents can be converted as a slides item type when using the LaTeX package Beamer. Two output formats are produced,

- A HTML web page, in the style of a *Chapter* item.
- The PDF output, as produced by LaTeX, containing the slides that can be presented with a PDF viewer or printed.

An example of Beamer slides output can be found here in the sample course. The PDF output is provided as a link in the sidebar of the HTML webpage.

Markdown with Chirun Markdown Extensions

Slides written in Markdown using the *Chirun Markdown Extensions* produces three output formats,

- A HTML web page, in the style of a *Chapter* item.
- Web-based slides, using the Reveal.js presentation framework.
- A printable PDF download showing the slides.

An example of Markdown slides can be found here in the sample course. Both the Reveal.js and PDF download are provided as links in the sidebar of the HTML page.

Note: The source document for the above Markdown slides can be found on GitHub at https://raw.githubusercontent.com/chirun-ncl/sample_course/master/markdown/lecture.md

Slides items currently have the same properties as *Chapter* items.

4.6.3 Example

```yaml
structure:
  - type: slides
    title: Beamer Slides
    source: lecture1.tex
  - type: slides
    title: Markdown Slides
    source: lecture2.md
```

4.7 Notebook

A *notebook* item is similar to a *Chapter* item, but intended for documents with many code blocks and authored in a style that would fit well as a Jupyer notebook.

The content is built in the style of a *Chapter* item, but with an additional download link provided to a Jupyter notebook version of the same content. Code blocks become runnable cells in the notebook, while other content becomes information-only cells.

Note: Currently, the notebook item type only works with Markdown source documents.

4.7.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The file name of the source for the item.	None
`sidebar`	Show the theme's sidebar including ToC?	`True`
`topbar`	Show the theme's topbar at the top of the page?	`True`
`footer`	Show the theme's footer at the bottom of the page?	`True`
`buildPDF`	Build a PDF for this item?	`True`
`js`	An array of paths to JS files to include in HTML output	`[]`
`css`	An array of paths to CSS files to include in HTML output	`[]`

4.7.2 Supported Source Formats

- Markdown with *Chirun Markdown Extensions*

4.7.3 Outputs

- HTML web page
- Jupyter notebook

4.7.4 Example

```
structure:
  - type: notebook
    title: Programming Handout
    source: handout.md
```

An example of the output from a notebook item can be found here in the sample course. Both the Jupyter notebook and PDF download are provided as links in the sidebar of the HTML page.

Note: The source document for the above Markdown slides can be found on GitHub at https://raw.githubusercontent.com/chirun-ncl/sample_course/master/markdown/handout.md

4.8 URL

A `url` item type is used to link to external URLs or static documents. For example, data file could be distributed verbatim by using the URL item type. URL item types are added to the introduction or part pages, but do not cause any extra content pages to be built; the item is linked to directly.

External links must begin `http://`, `https://` or `ftp://`.

Internal static files should be placed in a directory `static` in the same directory as the `config.yml` file. The contents of this directory will be automatically copied into the output directory by Chirun. Files in `static` can then be referenced relatively for URL items.

4.8.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The URL to be linked to.	None

4.8.2 Example

```yaml
structure:
  - type: url
    title: The BBC website
    source: https://bbc.co.uk
  - type: url
    title: Some static content
    source: static/data/dataset.RData
```

4.9 HTML

A *html* item is similar to a *Chapter* item, but intended for including raw HTML as part of the Chirun output in style consistent with the rest of the output pages.

Rendering is performed in the same way as for a chapter item, but rather than converting the document from its original source. The raw html file provided as the `source` file is inserted into the produced HTML web page in the place where processed document content would normally be placed.

Note: A HTML item is not reproduced verbatim as part of the output, but is processed to form a page in the style of a a `chapter` item. To include a `.html` file verbatim with no modifications, create an internal static *URL* item instead.

4.9.1 Item Properties

Property	Description	Default Value
`title`	The content item's title.	None
`source`	The file name of the source for the item.	None
`sidebar`	Show the theme's sidebar including ToC?	`True`
`topbar`	Show the theme's topbar at the top of the page?	`True`
`footer`	Show the theme's footer at the bottom of the page?	`True`
`js`	An array of paths to JS files to include in HTML output	`[]`
`css`	An array of paths to CSS files to include in HTML output	`[]`

4.9.2 Supported Source Formats

- HTML

4.9.3 Outputs

- HTML web page

4.9.4 Example

```
structure:
  - type: html
    title: Include raw HTML
    source: files/raw/document.html
```

CHAPTER

FIVE

REFERENCE

5.1 Command Line Arguments

5.1.1 Usage

```
usage: chirun [-h] [-o BUILD_PATH] [-v] [-vv] [-d] [-a] [--config CONFIG_FILE] [dir]
```

5.1.2 Arguments

Argument	Description	Default Value
`dir`	Path to a chirun compatible source directory	Current directory
`-h`	Show a help message and exit	
`-o BUILD_PATH`	Set a directory to put build files in	`build`
`-v`	Verbose output	
`-vv`	Very verbose output	
`-d`	Delete auxiliary files	
`-a`	Output using absolute file paths, relative to `root_url`	
`--config CONFIG_FILE`	Path to a config file	`config.yml`

5.2 `config.yml` Properties

The following is a list of top level properties that can be set in `config.yml` when creating a Chirun course package.

Argument	Description
`title`	The title used for the output package
`author`	The name(s) of the output package author(s)
`code`	An optional "course code" for the output package
`year`	An optional year for the output package
`structure`	An array of content items to be built
`themes`	An array of themes to be used
`build_pdf`	Should PDFs be built by default (where possible)?
`base_dir`	The base URL used when building packages with absolute file paths
`js`	An array of paths to JS files to include in HTML output
`css`	An array of paths to CSS files to include in HTML output

Note: At the moment the *Chirun Public Content Builder* and *Chirun LTI Provider* use the `code` property internally when building packages and so overrides any course code given in `config.yml`.

This is expected to change in a future version.

5.3 Chirun Markdown Extensions

The flavour of Markdown used in Chirun is Python Markdown with PyMdown Extensions and some further Chirun Extensions.

Chirun specific extensions are described below.

5.3.1 Sectioning

Documents are automatically wrapped into logical sections based on headings. This allows styles to be applied to blocks of content by using Attribute Lists. A logical section is automatically ended by a new heading of equal or lower level.

```
# H1 level section
Here is some text.

## H2 level section {: #someid .someclass style='background-color:#111; color: #EEE;'}
Here is some more text.

# Another H1 section
Finally, some ending text.
```

A logical section can also be forcefully ended in the following way,

```
## H2 level section {: #someid .someclass style='background-color:#111; color: #EEE;'}
Here is some more text.
## ---

This text will be outside of the above logical section.
```

Note: The default theme includes the classes `.exercise` and `.interlude` that can be used in this way.

5.3.2 Including Images

Include images by using the filename path relative to the source document. Chirun will detect images included in this way and will copy them to the output directory automatically. Attribute Lists can be used to customise image style.

```
![A plot of y=sin(x)](images/lecture_sine2.png){width="70%"}
```

5.3.3 Markdown Slides

Slides can be written in Markdown and converted to a HTML page, reveal.js slides and printable PDF by creating a *Slides* item type.

Note: Be aware that the Markdown used by Chirun is slightly different to the Markdown used when using reveal.js's `data-markdown` attribute. You need to use the Markdown variant described here.

An example of Markdown slides for Chirun can be found here in the sample course.

The source document for the above Markdown slides can be found on GitHub at https://raw.githubusercontent.com/chirun-ncl/sample_course/master/markdown/lecture.md

Slide Separator

Separate slides in Chirun Markdown by inserting a line containing nothing but (at least) 3 dashes, surrounded by a blank line above and below:

```
---
```

The slide separator renders as a horizontal rule in the HTML web page version of the content.

Pause/Fragments

Insert a pause (a fragment, in reveal.js language) by inserting a line containing nothing but 3 dots separated by spaces, surrounded by a blank line above and below:

```
. . .
```

A pause is not rendered in the HTML web page version of the content.

Other Reveal.js Features

Other reveal.js features can be enabled using Attribute Lists to add the required data attributes to sections or individual items. For example:

```
## This slide will have a background color {data-background-color="aquamarine"}
* Item 1
* Item 2

. . .

* Item 3
```

(continues on next page)

(continued from previous page)

```
---
```

5.3.4 Embedding Other Content

Numbas

A Numbas exam can be embedded into a document with,

```
<numbas-embed data-url="https://numbas.mathcentre.ac.uk/[...]" data-id="exercise-1" data-
↪cta="Show Exercise"></numbas-embed>
```

Attribute	Description
`data-url`	The URL to for the embeddable Numbas test
`data-id`	Some unique identifier for this test
`data-cta`	(Optional) Text to show on the button to load the test. Default: "Test Yourself"

YouTube

A YouTube video be embedded into a document with,

```
<youtube-embed data-id="EdyociU35u8"></youtube-embed>
```

Attribute	Description
`data-id`	The YouTube video ID

Vimeo

A Vimeo video be embedded into a document with,

```
<vimeo-embed data-id="8169375"></vimeo-embed>
```

Attribute	Description
`data-id`	The Vimeo video ID

oEmbed

Chirun supports embedding content with providers that support oEmbed.

```
<oembed data-url="[...]"></oembed>
```

Attribute	Description
`data-url`	The URL of the oEmbed compatible content to be embedded

5.3.5 Code Blocks

Code blocks with syntax highlighting can be included using SuperFences.

This example shows two different ways to include code blocks. The first is a code block set to use Python syntax highlighting. The second code block also displays Python code, but also includes a button that can be clicked to show the output from running the code.

~~~
### Print statements

```python
print("Hello", "World")
```

### If statements

```runnable lang="python"
x = 2
if x > 0:
    print('it is true')
```
~~~

5.4 Chirun LaTeX Package

When compiling LaTeX documents in Chirun, a LaTeX package is provided to provide some supporting functionality.

5.4.1 Using Chirun LaTeX Package

Use the `chirun` LaTeX package in your documents by adding the following line to you preamble:

```
\usepackage{chirun}
```

5.4.2 Features

Embed HTML

```
\begin{HTML}
    <div>
        <p>This raw HTML will be produced in the output directly</p>
    </div>
\end{HTML}
```

The raw HTML will not appear in the LaTeX PDF output.

Embed Numbas Test

```
\numbas[Test Yourself:]{https://numbas.mathcentre.ac.uk/[...]}
```

The Numbas test will appear embedded in the HTML web page. In the LaTeX PDF output, a link will be shown to the content.

Embed Youtube/Vimeo

```
\youtube[YouTube:]{EdyociU35u8}
\vimeo[Vimeo:]{8169375}
```

The video will appear embedded in the HTML web page. In the LaTeX PDF output, a link will be shown to the content.

Image Alt Text

```
\begin{figure}
    \includegraphics[width=0.8\textwidth]{images/hist.pdf}
    \caption{A histogram originally provided in .pdf format}
    \alttext{A plot titled "A histogram". The x axis is labelled "x-axis".
            The y axis is labelled "Frequency". The histogram shows a peak at
            a value of approximately 70.}
\end{figure}
```

The content of the `\altext{}` command will be attached to the figure image as alt text in the HTML web page. The LaTeX PDF output is unaffected.

CHAPTER

SIX

EXTENDING CHIRUN'S FEATURES

6.1 LaTeX Package Support

Chirun uses the PlasTeX compiler to parse LaTeX documents. Not all LaTeX packages are supported by PlasTeX, but more can be added by implementing them in Python.

If this is something you are interested in pursuing, see the PlasTeX Documentation on Macros and Packages to learn more.

6.2 Markdown Extensions

Documentation to be added in the near future.

6.3 Filters

Documentation to be added in the near future.

CHAPTER

SEVEN

ADDING NEW THEMES

Documentation to be added in the near future.

CHAPTER EIGHT

API

Warning: This following set of pages are generated automatically from the Chirun Python source code.

The documentation in the Chirun source is not currently optimised for this kind of autosummary, and so the documentation pages will be lacking much information until the situation is improved.

CHAPTER

NINE

CHIRUN LTI PROVIDER SETUP

This Chirun LTI Provider source code is freely available at: https://github.com/chirun-ncl/chirun-lti

The recommended way to install the Chirun LTI provider is via Docker Compose. The required Docker files and documentation for installing the LTI Provider in this way is available at: https://github.com/chirun-ncl/chirun-lti-docker