Scientific Writing

This book contains the informsation on hwo to write scientific papers and reports

Contents

1 PREFACE

1.1 Known Bugs

1. The figure prefix in the main section is currently set to fig. 1, however, we want that to be Fiure 1. This is a configuration in pandoc-crossref.
2. Much of the IU section on plagiarizm is copied. We may not have put in quotes to indicate the copied sections.
3. The instalation instructions may be outdted or wrong. For example we no longer use vagrant but have a docker container.
4. The current book has only been tested to be created on ubuntu 20.04
5. We like to move the entire book creation into a docker container so it is more portable
6. Images from graphviz and mermaide are not included.
7. The tables in the plagiarizm section are not properly formatted
8. The docker image generation has not been tested recently.
9. The dockerfiles are outdated.

1.2 Creating the ePubs from source

In case you wish to create the ePub from source, we have included this section.

The creation of the book is based on bookmanager.

The easiest way is to use our Docker container as described in Section 1.2.1.

1.2.1 Docker

We recommend the docker creation method for

- Ubuntu
- Windows 10
- macOS

1.2.1.1 Using OSX The easiest way to create a system that can compile the book on macOS, is to use a docker container. To do so, you need first to install Docker on macOS while following the simple instructions at

- https://docs.docker.com/docker-for-mac/install/

Once you have docker installed, you can follow the instructions in Section 1.2.1.

1.2.1.2 Using the Docker Image In case you have docker installed on your computer, you can create ePubs with our docker image. To create that image by hand, we have included a simple makefile. Alternatively, you can use our image from dockerhub if you like, it is based on Ubuntu and uses our Dockerfile.

First, you need to download the repository:

```
$ git clone https://github.com/cloudmesh-community/book.git
$ cd book
```

To open an interactive shell into the image, you say

```
$ make shell
```

The container image includes

- Python 3.7.4
- Pandoc 2.7.3
- pandoc-citeproc

Now you can skip to Section 1.2.4 and compile the book just as documented there.

Please note that we have not integrated pandoc-mermaid and pandoc-index at this time in our docker image. If you like to contribute them, please try it and make a pull request once you got them to work.

In case you want to create or recreate the image from our Dockerfile (which is likely not necessary, you can use the command

```
$ make image
```

1.2.2 Using the Native System

In case you like to use your native environment (which is typically faster than the container) you need to make sure you have an up to date environment.

Please note that you must have at least Pandoc version 2.5 installed as earlier versions do not work. We recommend that you use pandoc version 2.7.3 or newer. We recommend that you use Python version 3.7.4 or newer to run the scripts needed to assemble the document. However, earlier versions of Python 3 may also work but are not tested. You can check the versions with

```
$ pandoc --version
$ python --version
```

1.2.3 Using Vagrant

In case you have installed vagrant on your computer which is available for macOS, Linux, and Windows 10, you can use our vagrant file to start up a virtual machine that has all software installed to create the ePub.

First, you need to download the repository:

```
$ git clone https://github.com/cloudmesh-community/book.git
$ cd book
```

Next, you have to create the virtual machine with

```
$ vagrant up
```

You can login to the VM with

```
$ vagrant ssh
```

The book folder is mounted in the VM and you can follow the instructions in Section 1.2.1.

1.2.4 Creating a Book

Once you have decided on one of the methods, you can create a book.

To create a book, you have first to check out the book source from GitHub with if you have not yet done so (for example if you were to use the docker container method):

```
git clone git@github.com:cloudmesh-community/book.git
```

Books are organized in directories. We currently have created the following directories

```
./book/books/cloud/
./book/books/big-data-applications/
./book/books/pi
./book/books/writing
./book/books/222
./book/books/516
```

To compile a book, go to the directory and make it. Let us assume you like to create the cloud book for cloud

```
$ git clone https://github.com/cloudmesh-community/book.git
$ cd book/books/cloud
$ make
```

To view it you say

```
$ make view
```

After you have done modifications, you need to do one of two things. In case you add new images you need to use

```
$ make
```

The structure of the books is maintained in the yaml file in the directory where you execute the make in. It typically has the form `NAMEOFDIR.yaml`. Simply do an ls in the directory to see its name or inspect the Makefile. You can add new chapters to the YAML file, but discuss this first with Gregor. Typically, we have for incoming or draft chapters a special `draft` book to make sure the integration is done smoothly first in the draft.

1.2.5 Publishing the Book to GitHub

> ⚠ *This task is only to be done by* Gregor von Laszewski*. You will not have to do this step.*

To publish the book say

```
$ make publish
```

1.2.6 Creating Drafts

Drafts are maintained in the draft folder

```
$ cd book/books/cloud
$ make
```

We recommend that you use the following tools to clean up your files.

- mdl - markdownlint to cleanup your markdown
- biber - to cleanup your bibtex file

We still only use bibtex and not biblatex, but can use biber for doing some verification. Once you have installed them, you can verify your documents with

```
mdl filename.md
biber -V -tool filename.bib
```

Please remember that we have many thousands of references in our bib folder, so before you add a duplicate entry, please check in that folder. An easy way to do this is to use jabref loading the bibfiles.

1.2.7 Creating a New Book

Let us assume you like to create a new book. The easiest way to start is to copy from an existing book. However, make sure not to copy old files in dest. Let us assume you like to call the book gregor and you copy from the python directory.

You have to do the following.

```
$ cd book/books/python
$ make clean
$ cd ..
$ cp -r python gregor
$ cd ../gregor
$ mv python.yaml gregor.yaml
```

edit the Makefile and replace the NAME with gregor. make modifications to the table of contents in that YAML file and then compile with

```
$ make
```

1.2.8 Managing Images

In case you have added images to the book, they must be on the same level as your contribution, but in a directory called images. E.g.

```
./chapters/cloud/mydocument.md
./chapters/cloud/images/myimage.md
```

In the document, the image is then referred to as

```
![My imaage caption](images/myimage.md){#fig:cloud-myimage}
```

The label `#fig:cloud-myimage` must be unique in all of the documents. While adding the directory cloud before the image name, this is the case in our example.

1.2.9 Managing References

References are all managed in bibtex format while using pandoc-crosreff to cite them. There are many examples of the different entry types available in the bib directory. Do not duplicate entries, instead reuse them. Make sure you have a unique and meaningful label.

1.3 ePub Readers

This document is distributed in ePub format. Every OS has a suitable ePub reader to view the document. Such readers can, in many cases, also be integrated into a Web browser so that when you click on an ePub, it is automatically opened in your browser. As we use ePubs, the document can be scaled based on the user's preference. If you ever see content that does not fit on a page, we recommend you zoom out to make sure you can see the entire content.

We have made good experiences with the following readers:

- **macOSX**: Books, which is a build in ebook reader
- **Windows 10**: calibre
- **Linux**: calibre

If you have an iPad or Tablet with enough memory, you may also be able to use them.

Other browsers are available. Please report if you like one more than the once specified here, so we add your suggestion.

You may want to adjust the zoom of your reader to increase or decrease it. Please adjust your zoom to a level that is comfortable for you. On macOS with a larger monitor, we found that zooming out multiple times results in very good rendering allowing you to see the source code without horizontal scrolling.

1.4 Notation

The material here uses the following notation. This is especially helpful if you contribute content, so we keep the content consistent.

If you like to see the details on how to create them in the markdown documents, you can to look at the file source while clicking on the cloud in the heading of the Notation section (Section 1.4). This brings you to the markdown text. However, to see it you still have to look at the raw content to see the details.

or `![Github](images/github.png)`

> If you click on the or in a heading, you can go directly to the document in GitHub that contains the next content. This is convenient to fix errors or make additions to the content. The cloud is automatically added upon the inclusion of a new markdown file that includes in its first line a section header.

$

> Content in bash is marked with verbatim text and a dollar sign
>
> ```
> $ This is a bash text
> ```

[1]

> References are indicated with a number and are included in the reference chapter [1]. Use it in markdown with `[@las14cloudmeshmultiple]`. References must be added to the `references.bib` file in BibTex format.

Error

Chapters marked with this emoji are not yet complete or have some issue that we know about. These chapters need to be fixed. If you like to help us fixing this section, please let us know. Use it in markdown with `\faBug` or if you like to use the image with `![No](images/no.png)`.

REST 36:02

Example for a video with the `![Video](images/video.png)` emoji. Use it in markdown

```
[![Video](images/video.png)REST 36:02](https://youtu.be/
↪ xjFuA6q5Nh_U)
```

Slides 10

Example for slides with the `![Presentation](images/presentation.png)` emoji. These slides may or may not include audio.

Slides 10

Slides without any audio. They may be faster to download. Use it in markdown with `[![Presentation](images/presentation.png)Slides 10](TBD)`.

A set of learning objectives with the `![Learning](images/learning.png)` emoji.

A section is released when it is marked with this emoji in the syllabus. Use it in markdown with `![Ok](images/ok.png)`.

> Indicates opportunities for contributions. Use it in markdown with `![Question](images/question.png)`.

> Indicates sections that are worked on by contributors. Use it in markdown with `![Construction](images/co`

> Sections marked by the contributor with this emoji `![Smiley](images/smile.png)` when they are ready to be reviewed.

> Sections that need modifications are indicated with this emoji `![Comment](images/comment.png)`.

> A warning that we need to look at in more detail `![Warning](images/warning.png)`

> Notes are indicated with a bulb `![Idea](images/idea.png)`

Other emojis

> Other emojis can be found at https://gist.github.com/rxaviers/7360908. However, note that emojis may not be viewable in other formats or on all platforms. We know that some emojis do not show in Calibre, but they do show in macOS iBooks and MS Edge

> This is the list of emojis that can be converted to PDF. So if you like a PDF, please limit your emojis to

`\faGithub` `\faBug` `:relaxed:` :relaxed: `:sunny:` :sunny: `:baseball:` :baseball: `:spades:` :spades: `:hearts:` :hearts: `:clubs:` :clubs: `:diamonds:` :diamonds: `:hotsprings:` :hotsprings: `:warning:` :warning: `:parking:` :parking: `:a:` :a: `:b:` :b: `:recycle:` :recycle: `:copyright:` :copyright: `:registered:` :registered: `:tm:` :tm: `:bangbang:` :bangbang: `:interrobang:` :interrobang: `:scissors:` :scissors: `:phone:` :phone:

1.4.1 Figures

Figures have a caption and can be referred to in the ePub simple with a number. We show such a reference pointer while referring to Figure 1.

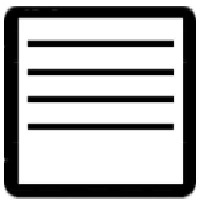

Figure 1: Figure example [1]

Figures must be written in the md as

```
![Figure example [@las14cloudmeshmultiple]](images/code.png){#fig:
   ↪ code-example width=1in}
```

Note that the text must be in one line and must not be broken up even if it is longer than 80 characters. You can refer to them with `@fig:code-example`. Please note for numbering to work, figure references must include the `#fig:` followed by a unique identifier. Please note that identifiers must be unique and that identifies such as `#fig:cloud` or similar simple identifiers are a poor choice and will likely not work. To check, please list all lines with an identifier, such as. Also not that if the image is copied form the internet you must not use the http link, but you must copy the image into the images folder. In addition for these images you must create a bibtex entru to the source where this image originated from. This applies also to images that you reused in other papers of yours. However if you have created the image yoruslef and it is not just a redrawing of somone elses work, you do not need a citation.

```
$ grep -R "#fig:" chapters
```

and see if your identifier is genuinely unique.

1.4.2 Hyperlinks in the document

To create hyperlinks in the document other than images, we need to use proper markdown syntax in the source. This is achieved with a reference, for example, in sections headers. Let us discuss the reference header for this section, e.g., Notation. We have augmented the section header as follows:

```
# Notation {#sec:notation}
```

Now we can use the reference in the text as follows:

```
In @sec:notation we explain ...
```

It will be rendered as: In Section 1.4 we explain …

1.4.3 Equations

Equations can be written as

~~~

and used in text:

$$a^2+b^2=c^2$${#eq:pythagoras}

It will render as: As we see in @eq:pythagoras.

The equation number is optional. Inline equations just use one dollar
sign and do not need an equation number:

```This is the Pythagoras theorem: $a^2+b^2=c^2$```

Which renders as:

This is the Pythagoras theorem: $a^2+b^2=c^2$.

Tables {#sec:tables}

Tables can be placed in the text as follows:
~~~

Table 1: Sample Data Table

x	y	z
1	2	3
4	5	42

As usual, make sure the label is unique. When compiling, it results in an error if labels are not unique. Additionally, there are several md table generators available on the internet and make creating table more efficient.

2 SECTION WRITING

2.1 Plagiarism

We start our discussion with the review of a most important topic. Typically you have had alreday a course that covers the topic of plagiarism. However, we like you to review this chapter anyways and provide us with a proof that you passed the plagiarizm test. Having done so will avoid any confusion about if you understand what plagiarizm is or not.

2.1.1 Plagiarism Definition

In academic life it is important to understand and avoid plagiarism. The dictionary defines plagiarism as follows dictionary.com:

pla·gia·rism

> “the practice of taking someone else’s work or ideas and passing them off as one’s own.”

2.1.2 Plagiarism Policies

Organizations and universities will have policies in place do address plagiarism. An example is provided for Indiana University [2]. We quote:

> “Honesty requires that any ideas or materials taken from another source for either written or oral use must be fully acknowledged. Offering the work of someone else as one’s own is plagiarism. The language or ideas thus taken from another may range from isolated formulas, sentences, or

> paragraphs to entire articles copied from books, periodicals, speeches, or the writings of other students. The offering of materials assembled or collected by others in the form of projects or collections without acknowledgment also is considered plagiarism. Any student who fails to give credit for ideas or materials taken from another source is guilty of plagiarism.
>
> (Faculty Council, May 2, 1961; University Faculty Council, March 11, 1975; Board of Trustees, July 11, 1975)"

Faculty members at Universities are also bound by policies that mandate reporting. At Indiana University the following policy applies (for a complete policy see the Web page):

> "Should the faculty member detect signs of plagiarism or cheating, it is his or her most serious obligation to investigate these thoroughly, to take appropriate action with respect to the grades of students, and *in any event* to report the matter to the Dean for Student Services (or equivalent administrator). The necessity to report every case of cheating, whether or not further action is desirable, arises particularly because of the possibility that this is not the student's first offense, or that other offenses may follow it. Equity also demands that a uniform reporting practice be enforced; otherwise, some students will be penalized while others guilty of the same actions will go free.
>
> (Faculty Council, May 2, 1961)"

Naturally if a student has any questions about understanding plagiarism the University can provide assistance. If a student is in doubt and asks for help this is not considered at that time plagiarism.

As you can see from the previous policies, the faculty do not have any choice but reporting cases of plagiarism to the university administration. Thus you must not hold them personally responsible as this is part of the tasks they are required to do if they like it or not. Instead, it is **the responsibility of the authors of the document** to assure no plagiarism occurs. If you are a student of a class that writes a paper or project report this naturally also all applies to you. In addition, if you work in a team you need to assure the entire team addresses plagiarism appropriately.

In practice this means that the teachers of a course expect you know what plagiarism is and you need to be abble to apply it in such a form in your writings that it does not occur. This is typically taught in other courses. However, as it is often overlooked by the student we are pointing it out here so we can make sure you contribute to courses that require you to write papers and reports. This also means you can not claim you did not know what plagiarism is. You are required to know what it is, know how to detect it and know how to avoid it. The resources provided next will give you the necessary tools and background.

2.1.3 Plagiarism Resources

The School of Education at Indiana University has a significant set of resources to get educated about plagiarism. These resources are intended to

> "preparing educators, advancing knowledge, and improving education" [2].

This includes:

- IU Definition of Plagiarism from Student Code of Conduct
- Overview How to give proper credit, steps.
- Cases of Plagiarism in the US, in the news, and elsewhere
- Examples Word for word, paraphrasing
- Practice with feedback on word-for-word and paraphrasing plagiarism
- Test 10 questions on recognizing plagiarism
- Tutorial Site Map Expanded table of contents
- Resources Websites, books, dictionary links, references for learning more about plagiarism

Furthermore, a number of tutorials are offered by Indiana University Instructional Systems Technology Department Web pages dealing with plagiarism. These include:

- Plagiarism Tutorial
- Understanding Plagiarism

2.1.4 How to Recognize Plagiarism

We are listing fifteen patterns of plagiarism that are defined at

- https://www.indiana.edu/~istd/patterns.html

which we copied into the next table. As such we have not included quotes but refer to their Web page but instead copied the tables with a reference. Naturally we do not want to be accused of plagiarize in a chapter about plagiarism.

Table 2: Plagiarism [3]

Name	Plagiarism Type	Reason
Clueless Quote	word-for-word	no quotes, no citation, no referenc
Crafty Cover-up	proper paraphrase but word-for-word	also present

Name	Plagiarism Type	Reason
Cunning Cover-up	paraphrasing	no citation, no reference
Deceptive Dupe	word-for-word	no quotes, no citation, but has refe
Delinked Dupe	word-for-word	no reference, even though quotes and citation
Devious Dupe	correct quote but word-for-word	also present
Dippy Dupe	word-for-word	quotes missing, even though full c and reference
Disguised Dupe	looks like proper para, but actually word-for-word	no quotes, no locator
Double Trouble	word-for-word and paraphrasing	although has reference
Linkless Loser	word-for-word	citation and reference lacking, alth has quotes and locator
Lost Locator	word-for-word	missing locator, although has quot citation, and reference
Placeless Paraphrase	paraphrasing	no reference, although citation pre
Severed Cite	paraphrasing	reference but no citation
Shirking Cite	word-for-word	lacks locator and reference, althou quotes and citation present
Triple D–Disguised Disconnected Dupe	word-for-word	looks like proper para, but no quot no reference, no locator

Table 3: Three patterns of non-plagiarism [3]

Name	Type	Description
Correct Quote	non-plagiarism	takes another’s words verbatim and acknowledges with quotation mark full in-text citation with locator, and reference
Proper Paraphrase	non-plagiarism	summarizes another’s words and acknowledges with in-text citation

Name	Type	Description
		and reference
Parroted Paraphrase	non-plagiarism	appears to be paraphrasing, and technically may not be plagiarism …

2.1.5 Citation Styles

Each journal or workshop and books will have their own citation styles. Often we find styles such as APA and Harvard. However these styles are often not used in scientific writing and tutorials that use them may provide you with information that is not applicable to other publications. Hence you need to review the rules for your specific publication venue carefully. In some cases this will lead to a significant improvement in readability as well as reduction in space.

While in APA style you may mention

> "John von Neumann (John von Neumann; 1927) describes a turing machine in his ground braking paper … that can do this and that and more.".

However in our community we often focus more on the technology than on the person or team that did this. Also citations will be precisely numbered. Hence this will be sufficient:

> "In [1] a turing machine is introduced that can do …"

Please be reminded that non of our assignments will use APA style citing. We use numbered ACM or IEEE proceedings and journal citations.

2.1.6 Exercise

Plagiarism.1: Read this document in order for you to undesrtand what plagiarizm is and how to cite

Plagiarism.2: What is the difference between APA and IEEE style citing? Why do you think in science a numbered citing is often prefered?

Plagiarism.3: Pass the plagiarism certification. This test is mandatory for all students taking this class. Typically it should have been done as part of another class, but we found through experience that some student that have not taken the certification have issues with this topic. Please note, that the test requires you to prepare for it with the material that is provided on the education web page.

2.2 Writing a Scientific Article

An important part of any scientific research is to communicate and document it. Previously we used LaTeX in this class to provide the ability to contribute professional looking documents. As part of this we also noticed that **any** document from **any** student that ever submitted a document in MSWord was inferior to those using structured documents such as provided by LaTeX or markdown. Although we would have loved to continue our work in LaTeX, a small number of students made the full adaptation of LaTeX as document system not possible. Thus we have adopted an even simpler approach. All documents written for this class are written in structured markdown. As we use a uniform format all documents in the class will be made available online, including your project reports. This allows anyone giving feedback to the class artifacts and not only by the instructors. IT also allows open sharing of ideas as any project in class must be unique and must not have been previously done. If it has been, a significant enhancement must be made.

While using markdown and github, you will also be able to collaboratively write documents in a group, while the instructors can inspect who has contributed to the document what while utilizing the history of the document.

In addition we will be able to use bibtex as reference manager and leverage hundrets of thousands professionally curated bibliography citations while not needing to worry about IEEE, ACM, or APA style formating requirements. The bibliography format is automatically created for you. In particular markdown is very useful for creating deployment and code documentations through easy to use verbatim or code environments. At the same time it allows us to use mathematical equations as defined in LaTeX and avoid images for equation inclusion.

Having a uniform report format not only helps us to be able to integrate the papers into a proceedings so you can yourself compare the paper length and effort as part of teaching a course with other students contributions. As such we have made available many previous projects through publicly available proceedings. We will furthermore explain how you can create such proceedings yourself.

2.2.1 Format

We distinguish typically two different artifacts. Such documents could be integrated in our book. Therefore you must not use the term *project*, *paper*, or *chapter* in the text or title. The artifacts are:

1. a small technology overview
2. a project report

A technology overview is a document that is multiple pages long and contains an overview about the technology, and a number of references. It is used to prepare you for writing a project report. Such technology overviews can be integrated as chapters into our book. You can look for example at the *graphql* section in the Cloud Computing book] for an example on how such a section looks like

A template for such a section is provided at

https://github.com/cloudmesh-community/proceedings-fa18/tree/master/paper

To use this template you must copy the md files precisely as a raw file and not just copy and paste from the rendered github output. We provide in the markdown section presice documentation on how to add images, citations and references.

2.2.2 Github limitations

Please note that github does not render your document precisely. Instead you need to install pandoc and create the document yourself if you like to see it. However, do never add the pdf or docx file to your github repository, as our proceedings script will automatically create the final version and your versions in github will conflict with them.

You will have exactly one md file and one bib file in the directory you manage, as well as an images directory that contains all images.

2.2.3 Working in a Team

Today research is done in potentially large research teams. This also include the writing of a document and code. In order for you not to learn additional systems you will be using github for collaboratively writing code and the document with your team mates.

2.2.4 Time Management

Obviously writing a paper takes time and you need to carefully make sure you devote enough time to it. The important part is that the paper should not be an after-thought but should be the initial activity to conduct and execute your research. Remember that

1. It takes time to read the information
2. It takes time understand the information
3. It takes time to do the research

For deadlines the following will get you in trouble:

1. *There are still 10 weeks left till the deadline, so let me start in 4 week . . . .* Procrastination is your worst enemy.

2. If you work in a team that has time management issues address them immediately
3. Do not underestimate the time it takes to prepare the final submission into the submission system. Prepare automated scripts that can deliver the package for submission in minutes rather than hours by hand.

2.2.5 Paper and Report Checklist

In this section we summarize a number of checks that you may perform to make sure your paper is properly formatted and in excellent shape. Naturally this list is just a partial list and if you find things we should add here, let us know.

One good way is to either copy the checklist into a file or print out just this pages and check with a pen if the particular issue occurs.

2.2.5.1 Team

1. If you work in a team make sure you only store the document once in github. Remove all duplicates. E.g. select one HID under which the report is stored.
2. Update your README.yml files `group:` where the first HID is the HID in which your report or paper is stored. Do this as soon as you have identified your team.
3. Once you have your team identified create the appropriate directory and copy the template into it. Upload it to github.com in your dedicated repository.

2.2.5.2 Content

1. Is the paper formal paper and not an experience report?
2. Do not include phrases such as "In week 1 we did this"
3. Do not include the words paper, chapter, or report in your text to refer to itself. Instead use general terms such as "We describe …". It is not important if it is a chapter, report, or paper. In fact if your pepre would be selected to be put in the book a paper becomes a chapter and you would have to change it. Therefore, we avoid it altogether.
4. When writing the *proposal* do not use the word *proposal* write the document as if it would be the final paper. We see too many reports at the end forgetting to remove the word proposal in the final paper, so we can not tell if you did it or if it is still a proposal. As the final paper is not a proposal we reject such papers and you get a 1/3 grade reduction. To avoid this, just do not use the word proposal.
5. When writing the abstract do not make it a proposal. Abstracts are no proposals. Avoid phrases such as We propose to do, We intend to show and so on. If the paper intends to show things you are still in the draft phase of the paper. However, if you say We show, that you later will not

have to change the text. Let us just assume you intended to show something but did not achieve then you can say *We intended to show this but we it was not possible to verify. We have provided reasons for this in the section A*. As you can see not only the intention is communicated, but the result. If you just focus on the intent that is just a proposal and is not a proper abstract.

6. Add meaningful up to 5 keywords to the paper
7. Do not add the class or your HID to the keywords. Instead, just add the HID after the title. THis way it can be easily seen in the proceedings.
8. If your paper is an introduction or overview paper, please do not assume the reader to be an expert. Provide enough material for the paper to be useful for an introduction into the topic.
9. A typical report has about 🐞number of words
10. If your paper limit is x number of words but you want to hand in > x + x 10 words this is an issue. If however you page limit is 2 pages and you hand in 4 or 6 pages that is no issue.

2.2.5.3 Submission

1. Do not make changes to your paper during grading, when your repository should be frozen.
2. Do not use filenames and directory names that have spaces in them only use `[a-z0-9-]`. Have all directory names be lower case.
3. Make all file names lower case other than Makefile and README.yml
4. You are required to run

   ```
   $ yamllint README.yml
   ```

   on all team members README.yml including your own and correct mistakes other than line length. Whenever you make modifications, please rerun yamllint. Do not add unnecessary spaces. Take a look at the previous class for examples. If you do not have yamllint you can write one in python. Its 3 lines of code. It is part of the class requirement to know how to write a yml file. We will deduct 10% of your grade if you repeatedly make mistakes in the yaml file.
5. Have you included the paper in the submission system (In our case git). This includes all images, bibliography files and other material that is needed to build the paper from scratch?
6. Have you made sure your paper compiles with *make* and the provided Makefile before you committed? 🐞a makefile will be provided to you so you can check if the document is correct
7. Are all images checked in?
8. Did you submit the report.bib file?

9. Remember that your document will be integrated in a proceedings that requires unique bibtex labels image labels and other references. Thus you could append them with your hid to make them unique.

We may experiment this year with a joint bibliography so you can reuse existing bibtex files and citations. 🐞stay tuned on this.

2.2.5.4 Bibliography

1. Are you managing your references in jabref
2. In the author field, authors are separated with an *andand not a comma.
3. The filename for the bibliography ends in .bib.
4. Bibtex labels must not have any spaces, _ or & in it
5. Fix citations in text that show as [?]. This means either your bib file is not up-to-date or there is a spelling error in the label of the item you want to cite.
6. Urls in citations are never placed in howpublished, instead we use url = { }. `howpublished` is just used for a text sting such as *Web page*, *Blog*, Repository and others like that. Do not use just the word Web. To be uniform use the word *Web Page*.
7. Do not use the only urls in the text, instead use a citation behind the url.
8. Are you references correct? References to a paper are no afterthought, they should be properly cited. Use jabref and make sure the citation type of the reference is correct and fill out as many fields as you can. Some journals and conferences have for example special requirements that go beyond the requirements of for example jabref. One example is that many conferences require you that wne you cite papers form another conference to augment the conferences not only with the location where the conference took place, but also with the dates the conference took place. Unfortunately, this is information that is often only available through additional google queries and many reference entries you find in the internet do not have this information readily available.

2.2.5.5 Writing

1. Have you spellchecked the paper?
2. Have you grammar checked the paper?
3. Use proper capitalization in the title, see: https://capitalizemytitle.com/
4. Are you using *a* and *the* properly?
5. Short form of verbs is for spoken language. Do not use them in scientific writing. Example: can't is incorrect, cannot is correct.
6. Do not use phrases such as *shown in the Figure bellow*. Instead, use *as shown in Figure 3*, when referring to the 3rd figure, but use the *ref* label macros. How to automatically use figure numbers

is explained in our template. You must use this automated figure numbering.

7. Do not use the word *instead* use *we* even if you are the sole author. In many cases you may want to avoid using the word *we* also.
8. Do not use the phrase *In this paper/report/chapter we show* instead use *We show*. It is not important if this is a paper or a report and does not need to be mentioned.
9. If you want to say *and* do not use *&* but use the word *and*.
10. Use a space after `. , :`
11. Headers are never just capitalized, E.g. `## INTRODUCTION` is wrong but `## Introduction` is correct.
12. Use proper indentation of the headers with `#` characters. The title has one `#` All others will have more
13. All headers are not numbered. numbers will be automatically added by our scripts

2.2.5.6 Citation Issues and Plagiarism

1. It is your responsibility to make sure no plagiarism occurs.
2. When stating claims you added the proper citations.
3. Do avoid paraphrasing long quotations (whole sentences or longer) form other papers.
4. Double check your paper if you have quote from other papers and included the citation.
5. The `[@label]` command must not be in the beginning of the sentence or paragraph, but in the end, before the period mark. Example: …a library called Message Passing Interface (MPI) `[7]`.
6. Put a space between the citation mark and the previous word.
7. There must not be any citation in the abstract or conclusion.
8. Citations cannot be included in section headings they need to be included in the text.

2.2.5.7 Character Errors

The following errors are very often found and must be avoided.

1. To emphasize a word, use *emphasize* and not “quote”. Quotes are reserved for quotes from other papers and must not be used to emphasize words or phrases. to put around a word that you like to emphasize.

2. When quoting we want you not only to use the `""` chars, but also the `>` char

   ```
   > "This is a proper quote" [@label].
   ```

3. Generally we do not us **bold* text. Instead use *italic*.

4. When using `<text>` in a text it must be quoted `<` `>` without quotes are interpreted as html and will likely render wrongly.

5. Pasting and copying from the Web often results in non-ASCII characters to be used in your text, please remove them and replace accordingly. This is the case for **all*quotes, dashes and all the other special characters such as three dots, copyright and so on.

2.2.5.8 Structural Issues

1. Does your paper include an Acknowledgement section.
2. Is the acknowledgment including all the people appropriately that helped you in your activity.
3. In case of a class and if you do a multi-author paper, you need to add an appendix called *Work-break Down* describing who did what in the paper,after the bibliography
4. Do you fulfill the minimum page length such as defined in the submission guideline. Remember that images, tables and figures do not count towards the page length.
5. Do not artificially inflate your paper if you are below the page limit.
6. In case you have an appendix it is included after the bibliography

2.2.5.9 Figures and Tables

1. Images must be at least 300dpi if they are not in a scalable format such as PDF which you can generate from Powerpoint and other drawing programs.
2. If you use Microsoft products, use ppt 4:3 ratio for drawing correct images. In case there is a powerpoint in the submission, the image must be exported as PDF.
3. If you have OSX, you are allowed to use omnigraffle.
4. Make sure you capitalize Table 1 when used in a sentence.
5. Do use `@fig:yourfigurelabel` to automatically refer to the figure in the text.
6. Table captions must be above the image.
7. Do not include the titles of the figures in the figure itself but instead use the caption or that information.
8. All images must be in native format, e.g. `.graffle`, `.pptx`, `.png`, `.jpg` in the images directory
9. Do not submit eps images. Instead, convert them to PDF
10. The image files must be in a single directory named *images*.
11. Make the figures large enough so we can read the details. If needed make the figure over two columns
12. Do not worry about the figure placement if they are at a different location than you think. Figures are allowed to float.
13. In case you copied a figure from another paper you need to ask for copyright permission. In case of a class paper, you must include a reference to the original in the caption. In general we like to avoid this for the reports and like that you produce original pictures. In case you can not, make

sure to put a citation in the caption.
14. Remove any figure that is not referred to explicitly in a sentence.

If you observe something missing let us know.

2.2.6 Acknowledgements

In many cases you not only want but must to include an acknowledgement section. In some cases you may be tempted to eliminate this section as you think you are out of space and the acknowledgement section may give you some additional space. This however is the wrong strategy and you should not do this. Instead you should shorten your paper elsewhere and leave enough space for acknowledgements.

In some cases where you get financial support from a university or a funding agency for a project such as from NIH or NSF specific information **must** be included. The best way is to verify with your coauthors. Additional acknowledgements may have to be added and you need te evaluate if for example significant help on the paper or the work that lead up to the paper warrants co-authorship.

An issue that we have seen often is for example when a professor has helped significantly on the paper but is not properly acknowledged. This can even lead to the professor asking you to remove him from the acknowledgement. A bad acknowledgement example is the following:

> We like to thank Professor Zweistein for his help in teaching me how to write the paper.

We do not think that the professor will be happy with this acknowledgement as it sounds like the only thing that was provided was the help on that you should have done anyways without the help of the professor. Ask yourself, if he introduced you to the field, has helped you with preparing the text, has given you insights, has corrected things in your paper, made suggestions. So instead of the previous acknowledge example maybe a more general term such as *helped with the paper* would be more appropriate. If you feel like your professor did not help you, leaving it off is more appropriate. In some cases you may wan to invite your professor to become a co-author. In some cases you may want to even include this handbook as a citation.

2.2.7 Exercises

Report.1: Install pandoc and jabref on your system

Report.2: Check out the report example directory. Create a PDF with pandoc and view it. Modify and recompile.

Report.4: Learn about the different bibliographic entry formats in bibtex

Report.5: What is an article in a magazine? Is it really an Article or a Misc?

Report.6: What is an InProceedings and how does it differ from Conference?

Report.7: What is a Misc?

Report.8: Why are spaces, underscores in directory names problematic and why should you avoid using them for your projects

Report.10: Why is it advantageous that directories are lowercase have no underscore or space in the name?

2.3 Markdown

Markdown is a simple markup language, however there is no uniform precise standard defined for it and implementations may have features not supported by other implementations. Nevertheless, when using the basic features, it provides a simple and easy way to quickly develop clean structured documents. The emphasize here is on structure, as in contrast to WYSIWYG editors it is not only important that your document looks good, but that the structure of the document is reflected in its layout. Thus you need to use headings and not just make headings look like headings with a bold font.

2.3.1 Markdown format

To convert the markdown to other formats with `pandoc`

```
# Heading {#sec:unique-heading-label-without-spaces}

## Sub-heading {#sec:unique-sub-heading-label-without-spaces}

### Sub Sub Heading heading

Paragraphs are separated
by a blank line. This is important. There must be one after the
   ↪ heading also

Text attributes include: *italic*, **bold**, `monospace`.

Horizontal rule:
```

~~~
---

Bullet list:

  * item 1
  * item 2
  * item 3

Numbered list:

  1. first
  2. second
  3. third

A [link](http://example.com).

```bash
$ echo "a bash script"
```

```python
print("a python script")
```

Images must all be local and must not have an http reference
All images must be placed in a directory called images/ as
shown in @fig:unique-labelwithoutspaces.

![This is the caption](images/example.png){#fig:unique-
   ↪ labelwithoutspaces}

Any figure used in the text must be referred to with a figure
cation and label. Images can not be embedded in itemized lists.

Quotes are in our publications not only done with "quote" but also
with `>` in front of each quoted line, it must be clearly indicated
before or after from which source:
~~~

```
> "This is a quote" [@label].

Please note that the period is after the label. Alternatively you can
    ↪  say

In [@label] we find the following list of properties.(use an
    ↪ appropriate starting
sentence):

> * item1
> * item1

Please note that we have not used quotes here as it is confusing in
    ↪ lists,
but it is clear from the `>` that we still quote.

URL's can be written as

[Google](http://www.google.com)

or

<http://www.google.com>
```

Frequent errors we see are:

- missing empty lines before and after sections
- use of `-` and `=` for section underline instead of `#`
- ignoring proper numbers of `#` in fromt of sections
- using `#` to do bold
- missing empty lines before and after code block starts and ends
- not left indenting text in code blocks
- not ending code blocks properly
- using wrong spaces in lists or codeblocks to replace lists
- not using a spell checker
- having spaces in front of numbered list items
- not using 80 char block formatting (makes it easier to correct things and display in different editors)

- trying to do to fancy things that are not supported in markdown

2.3.2 Editors

There are several tools that make writing documents in markdown easy. If you do not just look at the output of these documents but follow the structure guides properly. Usually whatever editor you use, it will not rendered the document properly. They are not supposed to be WYSIWYG editore. They help, but you need to make sure the markdown is valid and follows our convention just as you would with any other format.

Examples for such editors are:

Macdown MacDown is an open source Markdown editor for macOS that is available at https://macdown.uranusjr.com/

Emacs The most universal editor that has ever been written. It allows you to edit markdown text. Emacs comes in different flavors dependent on the OS. For macOS we have made good experiences with Aquamacs, CarbonEmacs

PyCharm As we do a lot of python programming, we recommend that your learn Emacs and/or PyCharm. Both come with Markdown editing modes.

Dilinger A HTML5 based cloud enabled editor https://dillinger.io/. It allows to download the created Markdown.

Markdown plus ⚠ *We are not using the many extensions > that are provided by markdown plus.*

We recommend that you use emacs or PyCharm. However, those that want to do advanced markdown outside the class may benefit from markdown plus. Markdown plus has lots of extensions. It is located at https://mdp.tylingsoft.com/. The source code is on github https://github.com/tylingsoft/markdown-plus. A local install can be achieved as follows:

```
$ git clone https://github.com/tylingsoft/markdown-plus.git
$ cd markdown-plus
$ brew install yarn
$ yarn install
$ yarn watch
$ open dist/index.html
```

Remarkable see https://remarkableapp.github.io/

Visualstudio see https://marketplace.visualstudio.com/items?itemName=yzhang.markdown-all-in-one

Textmate see https://macromates.com/

2.3.3 Converters

In addition to editors you can often convert text to markdown. However you need to be careful that the documents you created your text from may not produce the clean markdown you need and you may need to cleanup the text by for example replacing some characters, including proper spaces and other things.

The most powerful converter is pandoc and we recommend that you use it. Pandoc is also able to convert markdown in other formats such as ePub, PDF, HTML.

A word of warning: Although pandoc can convert from MSWord, Word has some limitations in its character sets that require you to clean the markdown after conversion. Experience shows that it cost less time and is far easier to write the document directly in markdown with emacs or pyCharm.

2.3.3.1 Conversion with pandoc Pandoc is a tool to convert file formats between each other. According tho the Web page Pandoc can convert "Markdown, reStructuredText, textile, HTML, DocBook, LaTeX, MediaWiki markup, TWiki markup, TikiWiki markup, Creole 1.0, Vimwiki markup, OPML, Emacs Org-Mode, Emacs Muse, txt2tags, Microsoft Word docx, LibreOffice ODT, ePub, or Haddock markup".

The Web page is located at.

- https://pandoc.org/

To convert files simply use the following command line option `-o` to specify the output format

```
pandoc filename.md -o filename.tex
```

In our example we converted the md file to latex.

As this document is created with pandoc we encourage you to review our Makefile to see how we use some more advanced features

2.3.3.2 Advanced Pandoc pandoc can integrate extensions and filters. We have not made yet use of them but like to explore them over time.

Packages of interest include:

- Include files http://pandoc.org/filters.html#include-files

- Integration of R https://github.com/cdupont/r-pandoc
- Figure numbers https://github.com/tomduck/pandoc-fignos
- Equation numbers https://github.com/tomduck/pandoc-eqnos
- Table numbers https://github.com/tomduck/pandoc-tablenos
- Cross refs with lots of numbering https://github.com/lierdakil/pandoc-crossref
- section numbering https://github.com/chdemko/pandoc-numbering
- CSV table https://github.com/baig/pandoc-csv2table
- inline CSV table https://github.com/mb21/pandoc-placetable

In our framework we use crosref and crosscite

2.3.3.2.1 Mermaid Mermaid is a graph generation tool that lets you create graphs and diagrams with the help of a description language. It includes graphviz and UML like diagrams, as well as gantt charts

A live editor is available at

- Mermaid live editor

A markdown plugin for pandoc is available

- Mermaid
- https://github.com/raghur/mermaid-filter

Installation is done with

```
$ npm install --global mermaid-filter
```

A sequence diagram example is shown next

```
~~~mermaid
sequenceDiagram
Alice->>John: Hello John
    John-->>Alice: Hallo Allice
~~~
```

```
sequenceDiagram
    Alice->>John: Hello John
    John-->>Alice: Hallo Allice
```

A flowchart looks like

```
~~~mermaid
graph LR
    Start --> End
~~~
```

```
graph LR
    Start --> End
```

2.3.4 Presentations in Markdown

Please find some links on hwo to use markdown to create slides

- https://yhatt.github.io/marp/
- http://slidify.org/
- https://rmarkdown.rstudio.com/lesson-11.html
- GitPitch https://github.com/gitpitch/gitpitch/wiki/Slide-Markdown

2.3.4.1 Markdown to PPTX `Pandoc` provides the ability to simply export markdown to power point. This could be useful for transitioning or first developing content in markdown to powerpoint. Simply use

```
pandoc filename.md -o fiename.pptx
```

and you will convert the markdown to a simple powerpoint that you can than improve. The initial improvement is best done in the overview mode of powerpoint so you can organize the bullet points and slides better in case the pagination is not done right.

2.3.5 Checking if the Markdown is valid?

After using this tool we found significant limitations, that do not include the syntactic and semantic checks that we need for our papers. So we recommend that if you use the tool to also inspect the file by hand.

Markdown is such a simple format that you should not have any issue. We recommend that you do a local checkout of the ePub and compile it and look at your section contribution. To work on a single file you can just use markdown editors.

A lint program is available at

- https://github.com/remarkjs/remark-lint

However, we recommend to copy your file into a separate directory and check it there as it installs some other programs into the directory where you do the checking.

2.3.6 References

- https://en.wikipedia.org/wiki/Markdown

2.3.7 Writing Papers and Reports with Markdown

2.3.7.1 Proper use of <> In this section we summarize a couple of issues that you will typically not find in general markdown documentation, that however are very important for us.

One of the frequent mistakes we see is that authors use the greater and smaller characters without proper quoting. THis is for example done when referring to command line parameters or keys.

Such as `<key>` and `command <parameter>`

If they are not done in quotes they break any markdown as they are interpreted as raw html Thus it is important that you quote them properly

2.3.7.2 Urls in markdown Urls must be wrapped in proper markdown syntax this is done through `[text](url)` or `<url>` work. If you just use the url in the text without the `<>` it will not render properly

2.3.7.3 Use star instead of underscore As we do some translations of the markdown, we noticed that whan you use `_` instead of `*` in your markdown this may lead to issues. please use `*italic*` and `**bold**` only.

2.3.7.4 Hyperreferences to other sections in markdown PLease note that you must not

```
# This is my header {#s-this-is-my-label}
```

Now, I can refer to Section. References, must not have spaces in it.

2.3.7.5 Code in markdown Code in markdown is easy to integrate with

~~~
```python
code
```
~~~

to see if syntax highlighting works. I have not tried this yet though When doing bash, we also like to try

~~~
```bash
$ script
```
~~~

Note that in order to indicate a new line in bash we use the `$` sign as prefix which indicates the prompt sign.

> ⚠ *Please note that there must be an empty line before and after the code block.*

2.3.7.6 Citations in markdown As we manage many references it is unnecessary to duplicate them. You can reuse them from others. IF you use them and notice errors, we require you to not only fix them in your bib, but also in the bib where you found it. Furthermore, we like you to use the same label and do not use a different label.

The best way to manage bibliographies is with jabref or emacs. We have seen students that try to avoid them while making their life unnecessarily complicated.

You must make sure that your bibliography entries are syntactically correct which either emace or jabref can do.

We will deduct points every time we notice that a bibliography is syntactically wrong. you will place the bibliography dependent on the artifact into a bib file. For papers it is placed in a file called `paper.bib`, for reports it is places in a file called `report.bib`. The markdown file is accordingly called `paper.md` or `report.md` all images must be placed in an `images/` directory

When you cite Then you can cite a references you can simply do this with for example `[@www-google]`. Please not that citation keys must not have spaces or underscore in them.

Example:

```
@Misc{www-google,
   author={{Google}},
```

```
   title={Google Home Page},
   howpublished={Web page},
   url={https://google.com}
}
```

Now you can use:

A good way to search on the internet is to use Google [4].

2.3.7.7 Markdown and bibtex

As you know we do not use LaTeX for this class but simply markdown. you can use pandoc to create your ePubs locally if you wish while following the paradox manual.

However, it is much simpler than that, as we create the proceedings with all your markdown papers for you once a week, so you can check it. OFten we create it multiple times a day.

So you do not have to do much than once in a while looking at the ePubs.

As part of this we like to remind you that we did distribute on day one of the class a document located in

https://github.com/cloudmesh-community/book/blob/master/README.md

That is called Scientific Writing II. This includes a section about LaTeX which you can ignore, but it also includes a section about bibtex and how to do bibtex entries that you may be benefitting from. So take a quick look at the Section 3. It also explains how to improve bibtex entries which is important for your projects.

http://cyberaide.org/papers/vonLaszewski-latex.pdf

Also, we noticed that some do not follow our tips posed as part of the FAQ's we send out here.

So we do recommend that you inspect them. TAs are assigned to also move the FAQs into the handbook. So you can also find them there (after a week).

Please also be reminded that there is an empty line before and after a heading or a quote or

a list or any paragraph. paragraphs are not indented with tabs or spaces

2.3.7.8 Please check your bibtex files

We continue to see that some have unnecessary issues with bibtex. Mostly the reason is they are not using emacs or jabref which come with validation that assures that commas are placed at the right location.

We highly recommend that you use them. In addition, we recommend that if you are familiar with the command line to install and run biber as documented in our Scientific Writing II handbook. Typically this is not needed when using emacs or jabref correctly.

You still need to ensure that there are no spaces in a label, the types are correctly done, and that company authors have two brackets. Please be reminded that an entry may only take you a minute with such tools, but if you do not use them you may spend hours trying to find a space in a label or a missing comma. bibtex is easy when using the right tool.

2.3.7.9 Using pandoc to check your files It is easy to install pandoc on your operating system. Please see the Pandoc web page and install it.

Once installed and you have in a directory the files

```
report.md
report.bib
images/test.png
```

You can easily generate for example a ePub, PDF, or html output with Make sure to also install pandoc-crossref

```
$ pandoc --verbose --filter pandoc-crossref -f markdown+
   ↪ header_attributes -f markdown+smart -f markdown+emoji --
   ↪ indented-code-classes=bash,python,yaml -o paper.epub paper.md
```

Assignment: provide documentation for Linux, OSX, Windows to do this.

A sample report is available at:

- https://github.com/cloudmesh-community/proceedings-fa18/tree/master/project-report

A sample 2 page paper is available at:

- https://github.com/cloudmesh-community/proceedings-fa18/tree/master/paper

Note that the formats of both are different. A two page paper is often too short to justify an abstract or even a conclusion.

2.3.8 Original Reference are a must

It came to our attention that some students forgot to cite the original reference to their technologies.

The first time you mention your technology is a good location for that example

Goggle [4] is a company that offers cloud services.

where www-google is the label to the bibtex entry representing the Google home page

2.4 Emacs

Emacs is one of the most powerfull editors. It originated from MIT AI Lab's Incompatible Timesharing System (ITS) while profiding a collection of macros used for editing. The name *Emacs* originates as an abbreviation for *Editor MACroS*. modern version of Emacs was rewritten in 1984 and since than has been evolved and maintained.

A number of extensive documentation is available at the following links according to the GNU Emacs Home page.

- GNU Emacs manual
- An Introduction to Programming in Emacs Lisp
- Emacs Lisp Reference Manual
- Other Emacs manuals

One of the most useful short manuals for emacs is the following reference card. It takes some time to use this card efficiently, but the most important commands are written on it. Generations of students have literally been just presented with this card and they learned emacs from it.

- https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf

There is naturally also additional material available and a great manual. You could also look at

- https://www.gnu.org/software/emacs/tour/

From the last page we have summarized the most useful and **simple** features. And present them here. One of the gems of emacs is the ability to recreate replay able macros which we include here also. You ought to try it and you will find that for data science and the cleanup of data emacs (applied to smaller datasets) is a gem.

Notation

Key	Description
C	Control
M	Esc (meta character)

Here are some other ways on what to do if you have accidentally pressed a wrong key:

- `C-g` If you pressed a prefix key (e.g. `C-x`) or you invoked a command which is now prompting you for input (e.g. Find file: …), type `C-g`, repeatedly if necessary, to cancel. `C-g` also cancels a long-running operation if it appears that Emacs has frozen.

- `C-/` If you executed a command and Emacs has modified your buffer, use `C-/` to undo that change.

Key	Description
—	**Saving and Exiting**
`C-x C-w`	Write the buffer to file
`C-x C-s`	Write the buffer to file and quit Emacs
—	**Cursor** *use the cursor keys or …*
`C-f`	Forward one character
`C-n`	Next line
`C-b`	Back one character
`C-p`	Previous line
—	**Cursor context move**
`C-a`	Beginning of line
`M-f`	Forward one word
`M-a`	Previous sentence
`M-v`	Previous screen
`M-<`	Beginning of buffer
`C-e`	End of line
`M-b`	Back one word
`M-e`	Next sentence
`C-v`	Next screen
`M->`	End of buffer
—	**Cursor jump**
`M-g` g	Jump to specified line number
—	**Search**
`C-s`	Incremental search forward
`C-r`	Incremental search backward
—	**Replace**

Key	Description
`M-` %	Query replace
—	**Killing (“cutting”) text**
`C-k`	Kill line
`C-y`	Yanks last killed text
—	**Macros**
`M-x (`	Start recording macro
`M-x )`	Stop recording macro
`M-x e`	Play back macro once
`M-5 C-x-e`	Play back macro 5 times

Modes

> “Every buffer has an associated major mode, which alters certain behaviors, key bindings, and text display in that buffer. The idea is to customize the appearance and features available based on the contents of the buffer.”

modes are typically activated by ending such as `.py`, `.java`, `.rst`,...

Key	Description
`M-x python-mode`	Mode for editing Python files
`M-x auto-fill-mode`	Wraps your lines automatically when they get longer than 70 characters.
`M-x flyspell-mode`	Highlights misspelled words as you type.

2.4.1 Org Mode

Emacs has some very advanced features that you can activate via a mode. One such feature is to organize a TODO list via org-mode.

Instead of us designing our own video, we point to a community tutorial such as

Cloud 18:04

2.4.2 Programming Python with Emacs

Emacs comes by default with syntax highlighting for python when you edit a `.py` file. This is really all you need. It also comes with a python ide that you can use and customize.

Python auto-completion for Emacs:

- https://github.com/tkf/emacs-jedi

Some more information is available at

- https://realpython.com/blog/python/emacs-the-best-python-editor/
- https://www.emacswiki.org/emacs/PythonProgrammingInEmacs

2.4.3 Emacs Keys in a Terminal

One of the real great features of knowing emacs is that you can set all your editors to emacs shortcuts. This includes pyCharm, but also bash. In bash you simply say

```
$ set -o emacs
```

in your bash prompt. Additionally, if you do not have a window systems configured, you can run emacs directly in the terminal with

```
$ emacs -nw
```

This you can log in to a remote computer and if it has emacs installed. Use it in the terminal. This would replace editors such as vi, vim, nano, pico or others that work in a terminal.

2.4.4 LaTeX and Emacs

LaTeX is directly supported by emacs and nothing has to be changed. However, a collection of information about additional LaTeX features for emacs is available at

- https://www.emacswiki.org/emacs/LaTeX

Of interest are for example also

- `M-x flyspell-mode`: allowing to do spell checking in the window
- predictive mode: https://www.emacswiki.org/emacs/PredictiveMode
- preview latex
- whizzy tex

However instead of previews and whizzy tex we recommend to use

- https://www.emacswiki.org/emacs/LatexMk

which comes pre-installed and allows you to do editing in one terminal, while previewing the update on change in another window.

2.4.5 LateXMk and Emacs

LatexMk allows one to outomatically compike and preview a document in case theire source is changed. THus if you run LateXMk and edit your files for example with emacs, The PDF viewer such as skim will automatically update the document. This is similar to overleaf, but much faster and without collaborators editing the same file.

2.5 Recording Audio with Autoplay

In some classes you may be asked to prepare a presentation that can be played at any time with recorder audio. Powerpoint provides such a mechanism, while allowing to combine the audio for each page to a consecutive recording.

To help you achieve this, we have provided the following simple demonstration.

Powerpoint with Autoplay and Sound (1:42)

2.6 Graphviz

Graphviz is a tool that allows you to visualize structural information with the help of abstract graphs and networks. It is achieved while providing the graph with automatic layout algorithms so you can focus on the creation of dependencies between nodes through edges. The main Web page is located at

- https://graphviz.gitlab.io/resources/

2.6.1 Installation

On macOS you can install graphviz with

```
$ brew install graphviz
```

On macOS there is also a graphviz version available that includes a GUI. The link to this software is:

- http://www.pixelglow.com/graphviz/

It can be downloaded from

- http://www.pixelglow.com/downloads/graphviz-1.13-v16.dmg

There is also an additional tool that is distributed by the community that is called doteditor and can be installed with

```
$ brew cask install doteditor
```

If you have issues with brew cask install, you can also install it by hand while going to

- https://vincenthee.github.io/DotEditor/

Online versions of graphviz are also available, but we have not tested them

- http://www.webgraphviz.com/
- https://dreampuf.github.io/GraphvizOnline/
- http://viz-js.com/
- http://graphviz.it/\#/gallery/unix.gv

There are many more versions available. Please contribute to this section to improve it

2.6.2 Usage

To use graphviz create a dot file and run the following command.

```
$ dot -Tpng filename.dot -o filename.png
```

This will create a png file. Other formats are also possible such as svg, or PDF

```
$ dot -Tsvg filename.dot -o filename.svg
$ dot -TPDF filename.dot -o filename.pdf
```

For inclusion in latex documents we recommend you create PDF output as it has a much better quality and is smaller in size than png.

2.6.2.1 The Dot Format

An extensive documentation is provided at

- https://graphviz.gitlab.io/documentation/

From there we find also the most simplest Hello World Graph>

```
$ echo "digraph G {Hello->World}" | dot -Tpng > hello.png
```

2.6.3 Exercise

Graphviz.1: Develop a REST service that takes a graph as input and returns a rendered version of the graph in a specified format. Make sure you can pass the format as a parameter.

Graphviz.2: Develop a REST service that takes a graph as input and returns a URL of the rendered graph while storing the output onto a data server. The data server is another rest service, from which the result can be picked up.

Graphviz.3: For IU students. Develop a REST service that takes a graph as input and returns a URL of the rendered graph while storing the output onto a data server. The data server is another rest service, from which the result can be picked up. Use box and/or google drive that are offered by IU as services. Make sure not to expose your passwords or access keys

3 REFERENCES

[1] G. von Laszewski, F. Wang, H. Lee, H. Chen, and G. C. Fox, "Accessing Multiple Clouds with Cloudmesh," in *Proceedings of the 2014 ACM international workshop on software-defined ecosystems*, 2014, p. 8 [Online]. Available: https://github.com/cyberaide/paper-cloudmesh/raw/master/vonLaszewski-cloudmesh.pdf

[2] Indiana University, "University policies: Cheating and plagiarism, ACA-72." Web Page, 1961 [Online]. Available: https://policies.iu.edu/policies/aca-72-cheating-plagiarism/index.html

[3] School of Education, Indiana University, "How to recognize plagiarism - examples: Patterns of plagiarism." Web Page, Aug-2019 [Online]. Available: https://plagiarism.iu.edu

[4] Google, "Google." Web Page, 2019 [Online]. Available: https://cloud.google.com