The UCI thesis package *

Hassán Lombera Rodríguez
hlombera@uci.cu

2016/07/28

Abstract

This class is a contribution of Center Vertex Interactive 3D Environment to the University of Computer Science at Havana, Cuba. It provides a useful and intuitive template for making a thesis document elaboration less painful. In this sense, it decouples the document presentation from its content. The package is in charge of the presentation, while the content is up to the user. The package scope includes thesis and master thesis.

Contents

*LaTeX 2ε thesis template for the University of Computer Science. This document was fully developed at Center Vertex Interactive 3D Environments and corresponds to thesis.cls v1.0.3, dated 2016/07/28.

1 Introduction

The `thesis` class provides some facilities to become your thesis elaboration an easy process. Its purpose is to decouple the document presentation from its content, in an intuitive way. To work with this class it is preferable having some LaTeX knowledge in advance. The in-training course "Introduction to the typesetting system LaTeX", provided by Center Vertex, would be a good choice to start. If you are already familiar with this class and you only need a template to start, go straight to Section 6.

2 Class options

The following options may be passed to this class. If more than one option is needed, please use comma separated values. If no value is used for an option, `true` is set by default. Default options will be highlighted with the tag ⟨*default*⟩.

`oneside` Enables oneside printing for the document ⟨*default*⟩.
Usage:
`\documentclass[`⟨*oneside*⟩`]{thesis}`

`twoside` Enables twoside printing for the document.
Usage:
`\documentclass[`⟨*twoside*⟩`]{thesis}`

`english` Tells `thesis` the document will be written in English ⟨*default*⟩. Hyphenation and keywords will be used properly.
Usage:
`\documentclass[`⟨*english*⟩`]{thesis}`

`spanish` Tells `thesis` the document will be written in Spanish. Hyphenation and keywords will be used properly.
Usage:
`\documentclass[`⟨*spanish*⟩`]{thesis}`

`noloa` Tells `thesis` not to list your algorithms in a dedicated page. Useful for removing such a page if your thesis contains no algorithms. By default, the class tries to list them and saves a page for that purpose. The option stands for **no** **l**ist **o**f **a**lgorithms.
Usage:
`\documentclass[`⟨*noloa*⟩`]{thesis}`

`nolol` Tells `thesis` not to list your listings in a dedicated page. Useful for removing such a page if your thesis contains no listings. By default, the class tries to list them and saves a page for that purpose. The option stands for **no** **l**ist **o**f **l**istings.
Usage:
`\documentclass[`⟨*nolol*⟩`]{thesis}`

`predefense` Tells `thesis` your document is ready for the predefense act. Thus, the class prepares a document version with no draft watermarks, comments or highlightings. In addition, it adds a blue seal on the left margin declaring the current document version is a digital compilation for a predefense purpose. This option declares your document almost ready.
Usage:
`\documentclass[`⟨*predefense*⟩`]{thesis}`

`pagelimitmode` Sets the mode on which `thesis` will count pages, for alerting when the limit has been reached. Strict mode ⟨*strict*⟩ counts 80 pages from the first page to the last one ⟨*default*⟩. Flex mode ⟨*flex*⟩ from just the introduction to the end.
Usage:
`\documentclass[`⟨*pagelimitmode=flex*⟩`]{thesis}`

`authorshippage` Sets the mode on which `thesis` will present the authorship page, under an odd total of participants (author + tutors). Possible values are **l**ast **t**utor **d**own ⟨*ltd*⟩ and **f**irst **a**uthor **u**p ⟨*fau*⟩. The first option is useful for the case of two authors and one tutor, for aligning all authors to the same level. Second option is on by ⟨*default*⟩.
Usage:
`\documentclass[`⟨*authorshippage=ltd*⟩`]{thesis}`

`bibitemsep` Controls the vertical space between the individual entries in the bibliography. Useful when a few entries remain alone in the last page of the bibliography. A proper change of this attribute can make all entries come closer, and you can save a page. The default value is 1ex. This option is on by ⟨*default*⟩.
Usage:
`\documentclass[`⟨*bibitemsep=1ex*⟩`]{thesis}`

`print` Makes all color links to disappear and properly changes the seal text if the predefense mode is on. In addition, backrefs are removed.
Usage:
`\documentclass[`⟨*print*⟩`]{thesis}`

`final` Declares your document ready, so it removes all watermarks, balloons and comments. This state is available for the digital and printable version.
Usage:
`\documentclass[`⟨*final*⟩`]{thesis}`

3 Prerequisites

This section gives an overview of all resources required by this class and discusses compatibility issues.

3.1 Requirements

The class `thesis` makes extensive use of the LaTeX packages repository[1]. The resources listed in this section are strictly required for `thesis` to function. The class will not work if they are not available. For some packages a minimum version will be indicated in brackets. The class triggers warnings when a package expected version is not found. They are:

- algorithm
- algpseudocode
- amsmath
- amssymb
- amsthm
- appendix
- aurical
- babel [2014/09/25]
- biber [2]
- biblatex [2014/06/25]
- bigstrut
- bookmark
- calligra
- caption [2013/05/02]
- csquotes
- datetime
- draftwatermark [2012/01/10]
- enumitem
- environ
- epstopdf
- etoolbox
- fancyhdr
- fncychap
- fontenc
- fp
- geometry
- glossaries [2015/03/16]
- graphicx
- hhline
- hyperref
- ifdraft
- ifthen
- inputenc
- kbordermatrix
- lipsum
- listings
- longtable
- mdframed [2013/07/01]
- multirow
- pdflscape
- pdfpages
- rotating
- setspace
- soul
- subcaption
- textpos
- tocloft
- todonotes [2012/07/25]
- txfonts
- upquote
- xcolor
- xkeyval

4 Installation

For being able to use this class, ALL packages related at Section 3.1 must be "visible" to LaTeX. To install the `thesis` class, run LaTeX on the installation script

[1] Packages installation should occur on the fly if the repository has been properly configured and updated from ctan.

[2] It is not a package, but the BibTeX replacement for users of biblatex.

`thesis.ins` and follow the instructions provided by the script itself. To produce the documentation run the file `thesis.dtx` through LaTeX.
Usage:
```
pdflatex thesis.ins
```
To produce the documentation ⟨*highly recommended*⟩ run the file thesis.dtx through LaTeX as follows.

Usage:
```
pdflatex thesis.dtx
makeindex -s gind.ist -o thesis.ind thesis.idx
makeindex -s gglo.ist -o thesis.gls thesis.glo
pdflatex thesis.dtx
pdflatex thesis.dtx
```

Expected files after installation:

1. `thesis.cls`
2. `masterthesis.sty`
3. `LogoUCI.eps`
4. `thesis.pdf` ⟨*documentation*⟩
5. `example.tex`
6. `README.txt`

5 User guide

The following are user-callable macros. Its usage will be shown when needed. If no, its regular usage should be assumed[3].

5.1 The title page

The title page can be easily generated by the following commands. All they should be placed in the preamble and are mandatory but the one referring to the UCI center[4].

`\addauthor` Command used for defining document authors. It can be used to support multiple authors definition, two at most. If three or more authors are defined they are ignored. It also supports role adjustment, via command options. The latter is useful for adjusting the authors' gender at the title and authorship pages.
Usage:
```
\addauthor[Autora]{Maybel Díaz Capote}
```

[3] Arguments in braces.
[4] This command is not mandatory as no all students are linked to a productive center. Thanks to Madelis Pérez Gil <mgil@uci.cu> from Faculty 2, for clarification.

`\author` Regular command for defining document authors. It is still available for backward compatibility. It is a wrapper for command `\addauthor`.
Usage:
```
\author[Autora]{Jane Roe}
```

`\title` Regular command for defining the document title. It is used for the same purpose.

`\ucicenter` Command used for defining the UCI center where the thesis document belongs to. It is not a mandatory command.

`\facultynum` Command used for defining the UCI faculty where the thesis document belongs to. The command expects a number as an argument.

`\addtutor` Command used for defining the thesis tutors. It can be used many times to support multiple tutors definition. It also supports role adjustment, via command options. They are useful for adjusting the tutors' gender at the title and authorship pages.
Usage:
```
\addtutor[Tutora]{MSc. Maybel Díaz Capote}
```

Example 1:

```
1.  \addauthor{Jane Roe}
2. \author{John Doe}
3. \title{\LaTeXe{} thesis template for the
4.        University of Computer Science}
5. \ucicenter{Vertex, Interactive 3D Environments}
6. \facultynum{5}
7. \addtutor[Tutora]{MSc. Maybel Díaz Capote}
```

`\city` Command used for defining the city where the thesis will be read. Havana is set by ⟨*default*⟩.
Usage:
```
\city{Madrid}
```

5.2 Thought, dedicatory and acknowledgment

The following commands should be placed in the preamble and are mandatory.

`\thought` Command used for quoting your prefer thought. Inside the command you can break lines at any time by placing the new line symbol \\. The text is automatically flush right.

`\dedicatory` Command used for setting your dedicatory.

`\acknowledgement` Command used to thank anybody.

Example 2:

```
\thought{What is well set on your mind will be
         for sure well set on your lips\\
         José Martí}
\dedicatory{To my wife, my endless love}
\acknowledgment{I am greatly indebted to
                professor John...}
```

5.3 The authorship page

The `thesis` class is in charge of this page. It is automatically generated from the authors and tutors data previously provided.

5.4 The abstract and keywords

The following commands should be placed in the preamble and are mandatory.

`\abstract` Command used for defining the thesis abstract.

`\keywords` Command used for defining the thesis keywords. Several keywords should be comma separated.

Example 3:

```
\abstract{Lorem ipsum dolor sit amet, consectetuer
          adipiscing elit. Ut purus elit, vestibulum ut,
          placerat ac, adipiscing vitae, felis. Curabitur
          dictum gravida mauris. Nam arcu libero, nonummy
          eget, consectetuer id, vulputate a, magna.}
\keywords{word1, word2, word3}
```

5.5 The table of content

The `thesis` class is in charge of this page. It is automatically generated from chapters, sections and subsections.

5.6 Lists of figures, algorithms and listings

The `thesis` class is in charge of these pages. They are automatically generated from figures, algorithms and listings. If your work has no algorithms or listings, it has no sense to show pages with no lists. Thus, the class allows to remove those pages. See Section 2 for proper class options.

5.7 Glossaries

The `thesis` class makes use of the `glossaries` package to include a glossary in your document. Thus, it should be defined in the preamble. We recommend to define them in a separate file and then to include the latter in the preamble.
Usage:

```
\input{Glossary}
```

Within the `Glossary` file we should follow the `glossaries` package syntax, e.g.

```
\newglossaryentry{Engine}{name=engine, description={part of a program
 which handles certain types of data (Computers)}}
```

5.8 Acronyms

The `thesis` class makes use of the `glossaries` package to include acronyms in your document. Thus, it should be defined in the preamble. We recommend to define them in a separate file and then to include the latter in the preamble.
Usage:

```
\input{Acronyms}
```

Within the `Acronyms` file we should follow the `glossaries` package syntax for this purpose, e.g.

```
\newacronym{cad}{CAD}{Computer Aided Design}
```

`\newacronymeng` Useful when an acronym in English is required to be used in a document written in Spanish. It automatically adds *"por sus siglas en inglés"* on its first use.
Usage:

```
\newacronymeng{cad}{CAD}{Diseño Asistido por Ordenador}
```

Then, whenever you require this acronym use the command `\ac{cad}`.

5.9 Bibliography

The `thesis` class is in charge of properly placing and formatting your bibliography. It makes use of package `biblatex` and `biber`. The default bibliography style is ISO 690 author-year. You only need to add your bib file location, in the preamble.
Usage:

```
\addbibresource{example.bib}
```

5.10 Regular sections

After setting the preamble of your thesis it is time to define the regular sections of your work. Inside the document, everything is similar to regular LaTeX documents. The first thing to do is calling the macro `\maketitle`. It is in charge of assembling the title page after collecting the data provided in the preamble. This call is mandatory. The remaining sections are very common. A typical introduction, the chapters, conclusions, suggestions and appendices. We will briefly depict how to work with them. Chapters work as expected. Only the sections introduction,

conclusions, suggestions and appendixes must begin with a command with these same names respectively. We recommend to define them in separate files and then to include them in your document via the `\input` command. See the Example 5.

`\introduction` Your introduction must start with this macro at its first line. It is in charge of properly configuring references, its entry in the table of content and the section title.

`\conclusions` Your conclusions must start with this macro at its first line. It is in charge of properly configuring references, its entry in the table of content and the section title.

`\suggestions` Your suggestions must start with this macro at its first line. It is in charge of properly configuring references, its entry in the table of content and the section title.

`\appendixes` Your appendixes must start with this macro at its first line. It tells to `thesis` class your work will have appendices. Thus, the class can save a place for them. You should add your appendices inside the predefined environment `addendum`. It is in charge of placing those at the end of bibliography. Otherwise it is up to you (not recommended).

Example 4:

```
1. \appendixes
2. \begin{addendum}
3.     \chapter{Projects}
4.         \section{An appendix example}
5.             \includegraphics[scale=0.45]{ProjectReport}
6. \end{addendum}
```

If your appendices are external pages you want to embed in your document, the `thesis` class still comes to your rescue. Use the following after the `\appendixes` macro.

```
\addappendix[linkname]{The_appendix_name}
```

For this case, the command option is the link name you will use to make reference to this appendix. Within your document do the following for referencing an external appendix:

```
\hyperlink{linkname.1}{The_word_which_makes_reference_to_your_appendix}
```

Example 5:

```
1. \begin{document}
2.   \maketitle
3.
4.   \input{Introduction}
5.   \input{Chapter1}
6.   \input{Chapter2}
```

```
 7.   \input{Chapter3}
 8.   \input{ChapterN}
 9.   \input{Conclusions}
10.   \input{Suggestions}
11.   \input{Appendices}
12. \end{document}
```

5.11 Figures and tables

Figures and tables are included as in regular LaTeX documents. Since v0.98 the `thesis` class searches for figures additionally in a folder named `images` ⟨*no case sensitive*⟩. On the other hand, we recommend the `longtable` package for splitting tables across pages.

5.12 Math equations

Math equations are included as in regular LaTeX documents. We strongly recommend the use of the `align` environment for which the class has a better performance.

5.13 Software engineering support

As for software engineering, the `thesis` class supports several artifacts for XP Methodology in an easy way: User story, CRC cards, Effort estimation by user story, Iteration plan, Acceptance test and Engineering tasks or Development tasks.

5.13.1 User story

`userstory` This environment is intended for creating a user story table in an easy way. The class is in charge of numbering tables, creating cells needed, splitting the table across pages, inserting captions and references in the list of tables and ensuring the user provides all data needed. The user has a macro for each cell he needs to fill. If some mandatory field is missing the class throws an error. For mandatory macros the tag ⟨*mandatory*⟩ will be used. The environment is not concerned about the order in which you provide the data. It is in charge of properly placing them.

`\storyname` Defines the story name ⟨*mandatory*⟩.

`\storyuser` Defines the story username ⟨*mandatory*⟩.

`\storyiter` Defines the iteration where this user story is assigned ⟨*mandatory*⟩.

`\storypriority` Defines the priority of this user story ⟨*mandatory*⟩.

`\storyrisk` Defines risks for this user story ⟨*mandatory*⟩.

`\storypoints` Defines the point estimation for this story user ⟨*mandatory*⟩.

`\storyprogrammer` Defines the programmer of this user story ⟨*mandatory*⟩.

`\storydescription` Sets the description for this user story ⟨*mandatory*⟩.

`\storyobservation` Defines the observation for this user story ⟨*mandatory*⟩.

`\storyinterface` Shows how the interface of this user story should look like ⟨*optional*⟩.

Example 6:

```
\begin{userstory}[tb:test]
   \storyname{Load File}
   \storyuser{Specialist}
   \storyiter{1}
   \storypriority{High}
   \storyrisk{Low}
   \storypoints{0.8}
   \storyprogrammer{John Doe}
   \storydescription
   {
      \lipsum[1]
      \begin{itemize}
         \item red
         \item white
      \end{itemize}
   }
   \storyobservation{\lipsum[1].}
   \storyinterface
   {
      \includegraphics[scale=0.5]{Images/DSC00461}
   }
\end{userstory}
```

The string in brackets `[`⟨*tb:test*⟩`]`, next to the beginning of the environment, stands for the label the user may use for referencing this table, as usual.

5.13.2 CRC Cards

`crccard` This environment is intended for creating a CRC card table in an easy way. The class is in charge of numbering tables, creating cells needed, splitting the table across pages, inserting captions and references in the list of tables and ensuring the user provides all data needed. The user has a macro for each cell he needs to fill. If some mandatory field is missing the class throws an error. For mandatory macros the tag ⟨*mandatory*⟩ will be used. The environment is not concerned about the order in which you provide the data. It is in charge of properly placing them.

`\crcclass` Defines the CRC class name ⟨*mandatory*⟩.

`\crcresp` Defines the class responsabilities ⟨*mandatory*⟩.

`\crccolab` Defines the colaboration with this class ⟨*mandatory*⟩.

Example 7:

```
\begin{crccard}[tb:mytable]
   \crcclass{GeneticAlgorithm}
   \crcresp
   {
      \begin{itemize}
         \item Create gens
         \item Create mutation
         \item Create recombination
      \end{itemize}
   }
  \crccolab
  {
     Population\\
     ProbabilityOperator\\
  }
\end{crccard}
```

The string in brackets `[`⟨*tb:mytable*⟩`]`, next to the beginning of the environment, stands for the label the user may use for referencing this table, as usual.

5.13.3 Effort estimation

`effortestimation` This environment is intended for creating the effort estimation table in an easy way. The class is in charge of numbering tables, creating cells needed, splitting the table across pages, inserting captions and references in the list of tables and ensuring the user provides all data needed. The user has a macro for each cell he needs to fill. If some mandatory field is missing the class throws an error.

`\addentry` Fundamental macro for adding entries to this table. The optional argument stands for the iteration this entry belongs to. First argument stands for the user story name. Second argument stands for the point estimation, in weeks. Despite it is not needed to set iterations in order, it is mandatory to group them[5]. See the Example 8 for details.

Example 8:

```
\begin{effortestimation}
   \addentry[4]{Graph assembly sequence}{1.1}
   \addentry[3]{Determine assembly sequence}{2.5}
   \addentry[1]{Loading file}{0.8}
   \addentry[1]{Manage tool change constraint}{0.5}
   \addentry[2]{Graph assembly sequence}{0.3}
   \addentry[2]{Graph assembly sequence}{0.3}
   \addentry[2]{Graph assembly sequence}{0.2}
   \addentry[2]{Graph assembly sequence}{0.2}
\end{effortestimation}
```

[5]This constraint will be removed soon.

5.13.4 Iteration plan

The Iteration plan is generated automatically from the `effortestimation` if the command `\geniterationplan` is used. The corresponding sums are calculated automatically. If the result is not what you expect you still can set the value manually, using the command `\setiterresult{iter}{result}`. For this case, the latter must be used as an argument of command `\geniterationplan`, or inside the `iterationplan` environment.

`\geniterationplan` Automatically generates the Iteration plan from the `effortestimation` environment which must be defined in advance.

`iterationplan` This environment alternatively creates as well the Iteration plan table. Define in brackets your label for referencing this table later.

`\setiterresult` Macro for modifying an entry for the iteration plan. Useful when you prevent an iteration will take a different time than the one calculated. First argument stands for the iteration you want to change. Second argument stands for the new value.

Example 9:

```
% First possible usage
\geniterationplan

% Second possible usage
\geniterationplan[tb:yourlabel]{\setiterresult{1}{3}}

% Third possible usage
\begin{iterationplan}[tb:yourlabel]
   \setiterresult{1}{3}
\end{iterationplan}
```

5.13.5 Engineering tasks

`engineeringtask` This environment is intended for creating an engineering task table in an easy way. The class is in charge of numbering tables, creating cells needed, splitting the table across pages, inserting captions and references in the list of tables and ensuring the user provides all data needed. The user has a macro for each cell he needs to fill. If some mandatory field is missing the class throws an error. For mandatory macros the tag ⟨*mandatory*⟩ will be used. The environment is not concerned about the order in which you provide the data. It is in charge of properly placing them.

`\engtaskuserstory` Sets the user story number this engineering task belongs to.

`\engtaskname` Sets the name for this engineering task.

`\engtasktype` Sets the type for this engineering task.

`\engtaskpointestimation` Sets the point estimation for this engineering task.

`\engtaskstartdate` Sets the starting date for this engineering task.

`\engtaskenddate` Sets the end date for this engineering task.

`\engtaskdescription` Sets the description for this engineering task.

`\engtaskprogrammer` Defines the programmer in charge of this engineering task ⟨*optional*⟩.

Example 10:

```
\begin{engineeringtask}[tb:engtask] % label in brackets
   \engtaskuserstory{4}
   \engtaskname{Register user}
   \engtasktype{Development}
   \engtaskpointestimation{1}
   \engtaskstartdate{6}{5}{2014} % day, month, year
   \engtaskenddate{24}{10}{2014}
   \engtaskdescription{User provides his data
                       to complete the process.}
   \engtaskprogrammer{John Doe}
\end{engineeringtask}
```

5.13.6 Development task

`developmenttask` Actually, this environment adds support to development tasks, for XP Methodology. The support consists of a wrapper for each command and the environment belonging to engineering tasks. The commands keep the same name but the first three letters. Change `eng` by `dev` on each. For instance, the `\engtaskuserstory` command is now `\devtaskuserstory`. Only one of these two environments (`engineeringtask` or `developmenttask`) should be used along the document. If no, a warning is thrown, but the package still runs. The wrapper is robust. Counters are properly defined.

`\devtaskuserstory` Sets the user story number this development task belongs to.

`\devtaskname` Sets the name for this development task.

`\devtasktype` Sets the type for this development task.

`\devtaskpointestimation` Sets the point estimation for this development task.

`\devtaskstartdate` Sets the starting date for this development task.

`\devtaskenddate` Sets the end date for this development task.

`\devtaskdescription` Sets the description for this development task.

`\devtaskprogrammer` Defines the programmer in charge of this development task ⟨*optional*⟩.

Example 11:

```
\begin{developmenttask}[tb:engtask] % label in brackets
   \devtaskuserstory{4}
   \devtaskname{Register user}
   \devtasktype{Development}
   \devtaskpointestimation{1}
   \devtaskstartdate{6}{5}{2014} % day, month, year
   \devtaskenddate{24}{10}{2014}
   \devtaskdescription{User provides his data
                       to complete the process.}
   \devtaskprogrammer{John Doe}
\end{developmenttask}
```

5.13.7 Acceptance test

`acceptancetest` This environment is intended for creating an acceptance test table in an easy way. The class is in charge of numbering tables, creating cells needed, splitting the table across pages, inserting captions and references in the list of tables and ensuring the user provides all data needed. The user has a macro for each cell he needs to fill. If some mandatory field is missing the class throws an error. For mandatory macros the tag ⟨*mandatory*⟩ will be used. The environment is not concerned about the order in which you provide the data. It is in charge of properly placing them.

`\testcasecode` Define the code for the acceptance test.

`\testcasedescription` Define the description for the acceptance test.

`\testcaseexeccond` Define the execution condition for the acceptance test.

`\testcaseexecstep` Define the execution steps for the acceptance test.

`\testcaseexpresult` Define the expected results for the acceptance test.

`\testcasename` Define the acceptance test name.

`\testcaseuserstory` Set here the number of the user story this acceptance test belongs to.

Example 12:

```
\begin{acceptancetest}[tb:expresult] % label in brackets
   \testcasecode{HU2\_P1}
   \testcasedescription{Description}
   \testcaseexeccond{Depict here the execution conditions.}
   \testcaseexecstep{Depict here the execution steps.}
   \testcaseexpresult{Define here the expected results}
   \testcasename{The test case name}
   \testcaseuserstory{1}
\end{acceptancetest}
```

5.14 Pseudocodes and true coding

For pseudocodes we recommend the use of the `algorithmic` environment. True coding are supported for the following programming languages: C++, Java, Python, PHP, HTML5, JavaScript and CSS. The last three might be combined. The `thesis` class uses the `listings` package for this purpose, properly modified. If your prefer language is not in the previous list, the `thesis` class will still try to load it from `listings`. You will receive an error if it is not present. If so, or if it does not look like as your expect, please contact the author of this class for a proper change.

Example 13:

```
1. \loadlanguage{C++}
2. \lstinputlisting[tabsize=3, breaklines, frame=lines,
3. firstline=1, lastline=14, caption=Maximum search]{code.cpp}
4.
5. \lstinputlisting[style=htmlcssjs, tabsize=3, breaklines,
6. frame=lines, firstline=89, lastline=146,
7. caption=Example code written in htmlcssjs]{code.cpp}
```

5.15 Page limit

As a requirement for the printing process at the University of Computer Science the page limit is set to 80. After this number, all pages will have a red watermark indicating the page limit has been reached. Keep in mind this feature fits two possible behaviors, for counting pages. See Section 2 for proper class options and descriptions.

5.16 Revision tools

As for revision, it is worth to remember the thesis output is a pdf file, which can be easily commented with Acrobat products. However, this commented file might be easily overwritten for the compiling process, if it is not saved outside the project folder. These circumstances may conduct to many file versions with different revisions each. So, it might be convenient to embed comments in the .tex file itself, to support the revision process in an easier way. Once again `thesis` class comes to your rescue. Thus, comments will be present despite you compile. Comments will be removed automatically when the document is declared as `final`.

`\comment` Macro for embeding comments in the .tex file.

Usage:

```
\comment[Revisor ID]{Comment}
```

Example 14:

```
1. Math might\comment[HLR]{Math is needed!} be useful for
2. games development.
```

`\hlblock` Macro for highlighting an entire paragraph.

`\hl` Macro for highlighting sentences in an easy way.

5.17 The master thesis style

For getting a master style of your document use the `masterthesis` package. It is an expected file after the installation process of `thesis` class. This style file is in charge of properly modifying the title page and some other minor details concerning the master thesis presentation requirements.

Example 15:

```
\documentclass[english]{thesis}
\usepackage{masterthesis}
```

6 An example for the thesis main file

The following is a minimal example for your thesis project. Namely, it depicts the major chapters and sections for your main file. Instructions will be as comments when needed.

Example 16:

```
\documentclass[spanish]{thesis} [2015/30/06 v1.0]

\addauthor{Jane Roe}
\author{John Doe}
\title{\LaTeXe{} thesis template for the
       University of Computer Science}
\ucicenter{Vertex, Interactive 3D Environments}
\facultynum{5}
\addtutor[Tutora]{MSc. Maybel Díaz Capote}

\thought{What is well set on your mind will be
         for sure well set on your lips\\
         José Martí}
\dedicatory{To my wife, my endless love}
\acknowledgment{I am greatly indebted to
                professor John...}

\abstract{Lorem ipsum dolor sit amet, consectetuer
          adipiscing elit. Ut purus elit, vestibulum ut,
          placerat ac, adipiscing vitae, felis. Curabitur
          dictum gravida mauris. Nam arcu libero, nonummy
          eget, consectetuer id, vulputate a, magna.}
\keywords{word1, word2, word3}

% Create a separate file for each. As filenames use the
```

```
% ones in braces.
\input{Glossary}
\input{Acronyms}

% Ensure the bib file is visible to LaTeX
\addbibresource{example.bib}

\begin{document}
  \maketitle

  % Create a separate file for each. As filenames use the
  % ones in braces.
  \input{Introduction}
  \input{Chapter1}
  \input{Chapter2}
  \input{Chapter3}
  \input{ChapterN}
  \input{Conclusions}
  \input{Suggestions}
  \input{Appendices}
\end{document}
```

Change History

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the definition; numbers in roman refer to the pages where the entry is used.