Todo list

remover lista de todo da versão final... 1

1: remover indice remissivo se não for necessário 33

remover lista de todo da versão final...

IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

DOUGLAS RODRIGUES PEREZ SP1665626

ANÁLISE DOS MECANISMOS DE CONTRATAÇÃO DE FUNCIONÁRIOS DA INQUESTI

São Paulo - SP - Brasil

06/09/2020

IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

DOUGLAS RODRIGUES PEREZ SP1665626

ANÁLISE DOS MECANISMOS DE CONTRATAÇÃO DE FUNCIONÁRIOS DA INQUESTI

Trabalho da disciplina de A5RHU com o objetivo de realizar uma análise dos mecanismos de contratação de funcionários da empresa em que o autor trabalhe.

Professor: LUIS FERNANDO AIRES BRANCO MENEGUETI

IFSP - Instituto Federal de Educação, Ciência e Tecnologia
Câmpus São Paulo

Tecnologia em Análise e Desenvolvimento de Sistemas

A5RHU - Recursos Humanos

São Paulo - SP - Brasil

06/09/2020

Este trabalho é dedicado à minha noiva que,
me apoia mesmo quando faço besteira.

Agradecimentos

Os agradecimentos principais são direcionados à Mylena Martins de Oliveira, meus pais e todos os meus amigos e colegas que estão ao meu lado.

"Eu me tornei insano,
com longos intervalos de horrível sanidade.
(Edgar Allan Poe (1809 - 1849))

Resumo

FAZER POR ULTIMO DE ACORDO COM A MYLENA

Palavras-chaves: APENAS PALAVRAS SEPARADAS POR PONTO

Abstract

This is the english abstract.

Key-words: ONLY WORDS WITH DOT

Lista de ilustrações

Lista de tabelas

Lista de quadros

Lista de abreviaturas e siglas

Lista de símbolos

Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
$\in$	Pertence

Sumário

1 INTRODUÇÃO 14
1.1 Histórico da Empresa 14
1.2 Elementos públicos de planejamento estratégico 14
1.2.1 Visão 14
1.2.2 Missão 14
1.2.3 Valores 14

2 MECANISMOS DE RECRUTAMENTO, SELEÇÃO E CONTRATAÇÃO DA INQUESTI 15
2.1 Etapas 15
2.2 Cronogramas 15
2.3 Análises 15

3 MODELO TEÓRICO E PRESSUPOSTOS (OU HIPÓTESES) DA PESQUISA 16

4 MÉTODOS DE PESQUISA 17
4.1 Tipo de Pesquisa 17

5 RESULTADOS DA PESQUISA 18
5.1 Assunto 1 18

6 CONSIDERAÇÕES FINAIS 19
6.1 Resposta à Questão de Pesquisa 19

Conclusão 20

ANEXOS 21

ANEXO A – MANUAL TODONOTES(PARCIAL) 22

ANEXO B – MANUAL PDFPAGES(PARCIAL) 26

ANEXO C – MANUAL ACRONYM(PARCIAL) 29

ANEXO D – CRAS NON URNA SED FEUGIAT CUM SOCIIS NATOQUE PENATIBUS 33

1 Introdução

A InQuesti é uma empresa de inteligência analítica, focada em inteligência de negócios e RPA (Robotic Process Automation, ou automação de processos com o uso de robôs), fundada em 2012, a empresa tem mais de 100 empresas atendidas, com mais de 500 projetos e com um time especializado.

1.1 Histórico da Empresa

A InQuesti é uma empresa com 8 anos, foi fundada em 2012 por 3 sócios e atualmente, conta com apenas dois: Vinicius Siqueira e Luciano Kassen, tem projetos em vários tipos de empresa e consta com cerca de 20 funcionários, sendo que 5 deles são funcionários internos.

<FALAR COM O VINI PARA FAZER O LEVANTAMENTO DO HISTÓRICO>

1.2 Elementos públicos de planejamento estratégico

<EXPLICAÇÃO DOS ELEMENTOS PÚBLICOS DE PLANEJAMENTO ESTRATÉGICO>

1.2.1 Visão

<EXPLICAÇÃO DE VISÃO DE ACORDO COM A LITERATURA>

<VISÃO DA INQUESTI>

1.2.2 Missão

<EXPLICAÇÃO DE MISSÃO DE ACORDO COM A LITERATURA>

<MISSÃO DA INQUESTI>

1.2.3 Valores

<EXPLICAÇÃO DE VALORES DE ACORDO COM A LITERATURA>

<VALORES DA INQUESTI>

2 Mecanismos de recrutamento, seleção e contratação da InQuesti

2.1 Etapas

2.2 Cronogramas

2.3 Análises

3 Modelo Teórico e Pressupostos (ou Hipóteses) da Pesquisa

4 Métodos de Pesquisa

4.1 Tipo de Pesquisa

5 Resultados da Pesquisa

5.1 Assunto 1

6 Considerações finais

6.1 Resposta à Questão de Pesquisa

Conclusão

Sed consequat tellus et tortor. Ut tempor laoreet quam. Nullam id wisi a libero tristique semper. Nullam nisl massa, rutrum ut, egestas semper, mollis id, leo. Nulla ac massa eu risus blandit mattis. Mauris ut nunc. In hac habitasse platea dictumst. Aliquam eget tortor. Quisque dapibus pede in erat. Nunc enim. In dui nulla, commodo at, consectetuer nec, malesuada nec, elit. Aliquam ornare tellus eu urna. Sed nec metus. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Phasellus id magna. Duis malesuada interdum arcu. Integer metus. Morbi pulvinar pellentesque mi. Suspendisse sed est eu magna molestie egestas. Quisque mi lorem, pulvinar eget, egestas quis, luctus at, ante. Proin auctor vehicula purus. Fusce ac nisl aliquam ante hendrerit pellentesque. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Morbi wisi. Etiam arcu mauris, facilisis sed, eleifend non, nonummy ut, pede. Cras ut lacus tempor metus mollis placerat. Vivamus eu tortor vel metus interdum malesuada.

Sed eleifend, eros sit amet faucibus elementum, urna sapien consectetuer mauris, quis egestas leo justo non risus. Morbi non felis ac libero vulputate fringilla. Mauris libero eros, lacinia non, sodales quis, dapibus porttitor, pede. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Morbi dapibus mauris condimentum nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam sit amet erat. Nulla varius. Etiam tincidunt dui vitae turpis. Donec leo. Morbi vulputate convallis est. Integer aliquet. Pellentesque aliquet sodales urna.

Anexos

ANEXO A – Manual todonotes(parcial)

The todonotes package*

Henrik Skov Midtiby
henrikmidtiby@gmail.com

July 9, 2015

Abstract

The `todonotes` package allows you to insert to–do items in your document. At any point in the document a list of all the inserted to–do items can be listed with the `\listoftodos` command.

Contents

1 Introduction **2**
1.1 Usage . . . 2
1.2 Package options . . . 4
1.3 Options for the todo command . . . 5
1.4 Options for the missingfigure command . . . 7
1.5 Options for the listoftodos command . . . 8
1.6 Known issues . . . 9
1.7 Things to improve . . . 11
1.8 Usage methods . . . 12

2 Implementation **19**
2.1 Declaration of options for the package . . . 19
2.2 Options for the todo command . . . 23
2.3 The main code part . . . 25

*This document corresponds to todonotes .dtx, dated 2015/07/09.

1 Introduction

The `todonotes` package makes three commands available to the user: `\todo[]{}`, `\missingfigure{}` and `\listoftodos`. `\todo[]{}` and `\missingfigure{}` makes it possible to insert notes in your document about things that has to be done later (todonotes . . .). I developed the basic functionality of the package while I worked on my bachelor project.

Some alternatives for the todonotes package are:

- easy-todo
  Depends on `color`, `tocloft` and `ifthen`, small feature set.
- fixmetodonotes
  Depends on `graphicx`, `color`, `transparent`, `watermark`,`fix-cm`, `ulem` and `tocloft`, small feature set.
- todo
  Depends on `amssymb`, medium feature set.
- fixme
  Large package with a lot of features.

The main reason for considering other packages is that the todonotes package is quire large and relies heavily on tikz. This can slow down compilation of large documents significantly. The mentioned alternatives have a different feature set and does not rely on tikz, which makes them require less ressoureces.

1.1 Usage

`\todo` My most common usage of the todonotes package, is to insert an todonotes somewhere in a latex document. An example of this usage is the command

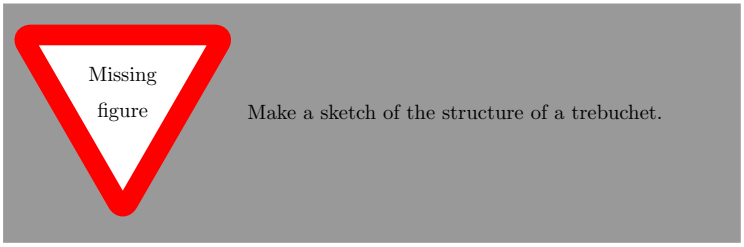
`\todo{Make a cake \ldots}`,

which renders like. The `\todo` command has this structure: `\todo[`⟨*options*⟩`]{`⟨*todo text*⟩`}`. The `todo text` is the text that will be shown in the todonote and in the list of todos. The optional argument `options`, allows the user to customize the appearance of the inserted todonotes. For a description of all the options see section 1.3.

`\missingfigure` The `\missingfigure` command inserts an image containing an attention sign and the given text. The command takes only one argument `\missingfigure{`⟨*text*⟩`}`, a text string that could describe what the figure should consist of. An example of its usage could be

`\missingfigure{Make a sketch of the structure of a trebuchet.}`

which renders like.



Make a sketch of the structure of a trebuchet.

`\listoftodos` The `\listoftodos` command inserts a list of all the todos in the current document. `\listoftodos` takes no arguments. For this document the list of to–do's looks like.

Todo list

Make a cake ... 2
Figure: Make a sketch of the structure of a trebuchet. 2
And a green note 5
Anything but default colors 5
A note with no line connecting it to the placement in the original text. 5
A todonote placed in the text 6
Fill those circles ... 6
A note with a large font size. 6
Note with very small font size. 6
Short note 6
Short note with prepend 6
Short note with noprepend 6
Testing. 6
Testing author option. 7
Testing author option. 7
Figure: Testing a long text string 7
Figure: Testing a long text string 7
Figure: Add a test image ... 7
Figure: Testing 8
Figure: Testing figcolor 8
Does this eat the space? 9
Test of newly defined command. 12
Test of newly defined command, requesting a green color. 12
1: A numbered todonote. 13
2: Another numbered todonote. 13
Comment [HSM1]: Testing first time. 13
Comment [HSM2]: Testing second time. 13

ANEXO B – Manual pdfpages(parcial)

The pdfpages Package*

Andreas Matthias
andreas.matthias@gmail.com

2013/08/25

Abstract

This package simplifies the insertion of external multi-page PDF or PS documents. It supports pdfTeX, VTeX, and XeTeX.

Contents

1 Introduction **1**

2 Usage **2**
2.1 Package Options 2
2.2 Commands 2
2.3 The Layout 10
2.4 Pitfalls 11

3 Required Packages **11**

4 Acknowledgment **11**

1 Introduction

When creating PDF documents, it is sometimes useful to insert pages of external PDF documents. This can be done with the `\includegraphics` command from the `graphics` package. But a simple `\includegraphics{doc.pdf}` normally produces '`Overfull \hbox`' and '`Overfull \vbox`' warnings, because the size of the inserted pages does not match the print space.

The `pdfpages` package makes it easy to insert pages of external PDF documents without worrying about the print space. Here are some features of the `pdfpages` package: Several logical pages can be arranged onto each sheet of paper and the layout can be changed individually. A lot of hypertext operations are supported, like links to the inserted pages, links to the original PDF document, threads, etc. When working with VTeX the same is possible with PostScript documents, too. Note that PostScript documents are only supported by VTeX and *not* by pdfLaTeX.

*This file has version number v0.4v, last revised 2013/08/25.

When producing DVI output `pdfpages` cannot insert pages of a PDF documents. But instead of interrupting execution `pdfpages` will insert empty pages. This feature is important when using packages like `pst-pdf`, which need to produce DVI output at the first run.

Links and other interactive features of PDF documents When including pages of a PDF only the so called content stream of these pages is copied but no links. Up to now there are no TeX-engines (pdfTeX, XeTeX, ...) available that can copy links or other interactive features of a PDF document, too. Thus, all kinds of links[1] will get lost during inclusion. (Using `\includepdf`, `\includegraphics`, or other low-level commands.)

However, there's a gleam of hope. Some links may be extracted and later reinserted by a package called *pax* which can be downloaded from CTAN [3]. Have a look at it!

2 Usage

2.1 Package Options

`\usepackage[`⟨*options*⟩`]{pdfpages}`

⟨*option*⟩ – **`final`:** Inserts pages. This is the default.

`draft`: Does not insert pages, but prints a box and the filename instead.

`enable-survey`: Activates survey functionalities. (*experimental, subject to change*)

2.2 Commands

`\includepdf` Inserts pages of an external PDF document.

`\includepdf[`⟨*key=val*⟩`]{`⟨*filename*⟩`}`

⟨*key=val*⟩ – A comma separated list of options using the ⟨*key*⟩=⟨*value*⟩ syntax.

⟨*filename*⟩ – Filename of the PDF document. (The filename *must not* contain any blanks!)

The following list describes all possible options of `\includepdf`. All options are using the ⟨*key=value*⟩ syntax.

- Main options:

pages Selects pages to insert. The argument is a comma separated list, containing page numbers (`pages={3,5,6,8}`), ranges of page numbers (`pages={4-9}`) or any combination. To insert empty pages use `{}`.

E.g.: `pages={3,{},8-11,15}` will insert page 3, an empty page, and pages 8, 9, 10, 11, and 15.

[1] Actually not only links but all kinds of *PDF annotations* will get lost.

Page ranges are specified by the following syntax: ⟨*m*⟩-⟨*n*⟩. This selects all pages from ⟨*m*⟩ to ⟨*n*⟩. Omitting ⟨*m*⟩ defaults to the first page; omitting ⟨*n*⟩ defaults to the last page of the document. Another way to select the last page of the document, is to use the keyword `last`. (This is only permitted in a page range.)

E.g.: `pages=-` will insert *all* pages of the document, and `pages=last-1` will insert all pages in reverse order.

(Default: `pages=1`)

nup Puts multiple logical pages onto each sheet of paper. The syntax of this option is: `nup=`⟨*xnup*⟩`x`⟨*ynup*⟩. Where ⟨*xnup*⟩ and ⟨*ynup*⟩ specify the number of logical pages in horizontal and vertical direction, which are arranged on each sheet of paper. (Default: `nup=1x1`)

landscape Specifies the format of the sheet of paper, which is rotated by 90 degrees. This does *not* affect the logical pages, which will *not* be rotated by the 'landscape' option. To rotate the logical pages use the 'angle' option (e.g. 'angle=90'). Either 'true' or 'false' (or no value, which is equivalent to 'true'). (Default: `landscape=false`)

- Layout options:

delta Puts some horizontal and vertical space between the logical pages. The argument should be two dimensions, separated by space. See Chapter 2.3 and Figure 1. (Default: `delta=0 0`).

offset Displaces the origin of the inserted pages. The argument should be two dimensions, separated by space. In 'oneside' documents positive values shift the pages to the *right* and to the *top* margin, respectively, whereas in 'twoside' documents positive values shift the pages to the *outer* and to the *top* margin, respectively. See Chapter 2.3 and Figure 1. (Default: `offset=0 0`)

frame Puts a frame around each logical page. The frame is made of lines of thickness `\fboxrule`. Either 'true' or 'false' (or no value, which is equivalent to 'true'). (Default: `frame=false`)

column Pdfpages normally uses 'row-major' layout, where successive pages are placed in rows along the paper. The `column` option changes the output into a 'column-major' layout, where successive pages are arranged in columns down the paper. Either 'true' or 'false' (or no value, which is equivalent to 'true'). (Default: `column=false`)

columnstrict By default the last page is not set in a strict 'column-major' layout, if the logical pages do not fill up the whole page. The `columnstrict` option forces a strict 'column-major' layout for the last page. Either 'true' or 'false' (or no value, which is equivalent to 'true'). (Default: `columnstrict=false`)

1	4	
2	5	
3		

`columnstrict=true`

1	3	5
2	4	

`columnstrict=false`

ANEXO C – Manual acronym(parcial)

An Acronym Environment for LaTeX 2ε*

Tobias Oetiker

2015/03/21

1 Introduction

When writing a paper on cellular mobile radio I started to use a lot of acronyms. This can be very disturbing for the reader, as he might not know all the used acronyms. To help the reader I kept a list of all the acronyms at the end of my paper.

This package makes sure, that all acronyms used in the text are spelled out in full at least once.

2 The user interface

The package provides several commands and one environment for dealing with acronyms. Their appearance can be controlled by two package options and three macros.

2.1 Acronyms in the Text

`\ac` To enter an acronym inside the text, use the

`\ac{`⟨*acronym*⟩`}`

command. The first time you use an acronym, the full name of the acronym along with the acronym in brackets will be printed. If you specify the `footnote` option while loading the package, the full name of the acronym is printed as a footnote. The next time you access the acronym only the acronym will be printed.

`\acresetall` The 'memory' of the macro `\ac` can be flushed by calling the macro `\acresetall`. Afterwards, `\ac` will print the full name of any acronym and the acronym in brackets the next time it is used.

`\acf` If later in the text again the Full Name of the acronym should be printed, use the command

`\acf{`⟨*acronym*⟩`}`

*This file has version number v1.41, last revised 2015/03/21.

to access the acronym. It stands for "full acronym" and it always prints the full name and the acronym in brackets.

\acs To get the short version of the acronym, use the command

\acs{⟨*acronym*⟩}

\acl Gives you the expanded acronym without even mentioning the acronym.

\acl{⟨*acronym*⟩}

\acp Works in the same way as \ac, but makes the short and/or long forms into plurals.

\acfp Works in the same way as \acf, but makes the short and long forms into plurals.

\acsp Works in the same way as \acs, but makes the short form into a plural.

\aclp Works in the same way as \acl, but makes the long form into a plural.

\acfi Prints the Full Name acronym (\acl) in italics and the abbreviated form (\acs) in upshaped form.

\acused Marks an acronym as used, as if it had been called with \ac, but without printing anything. This means that in the future only the short form of the acronym will be printed.

\acsu Prints the short form of the acronym and marks it as used.

\aclu Prints the long form of the acronym and marks it as used.

Example: \acl{lox}/\acl{lh2} (\acsu{lox}/\acsu{lh2})

\iac Works in the same way as the \ac command but prefixes it with an appropriate indefinite article.

\Iac Works in the same way as the \ac command but prefixes it with an appropriate upper case indefinite article.

\...* The following commands do the same as their unstarred forms, except that the acronym will not be marked as used. If you work with the 'onlyused' option then macros which have only been used with starred commands will not show up. \ac*, \acs*, \acl*, \acf*, \acp*, \acsp*, \aclp*, \acfp*, \acfi*, \acsu*, \aclu*, \iac* and \Iac*.

2.2 Customization

The appearance of \acs and \acf can be configured in various ways. Of main importance are the package options:

footnote makes the full name of the acronym appear as a footnote.

smaller lets the acronyms appear a bit smaller than the surrounding text. This is in accord with typographic convention. The relsize package is required.

There are three lower-level macros controlling the output. Any acronym printed by \acs is formatted by \acsfont. Similarly, unless the option footnote is specified, \acffont handles the output of \acf, where the included acronym goes through \acfsfont (and \acsfont). The plural forms are treated accordingly. Usually the three macros do nothing. To give an example, the option smaller makes \acsfont use the command \textsmaller from the relsize package:

\acsfont
\acffont
\acfsfont

```
\renewcommand*{\acsfont}[1]{\textsmaller{#1}}
```

2.3 Defining Acronyms

Acronyms can either defined from an environment specifically introduced for that purpose or by direct definitions.

`acronym` The `acronym` environment allows one to define all the acronyms needed by a document at a single place and is self-documenting, since a table of acronyms is automatically produced.

`\acro` In the `acronym` environment, acronyms are defined with the command:

`\acro{`⟨*acronym*⟩`}[`⟨*short name*⟩`]{`⟨*full name*⟩`}`

The first argument ⟨*acronym*⟩ is the acronym string itself and is used in the commands of the previous section such as `\ac` or `\acl`, that print the different forms of the acronym.

Because internal commands take ⟨*acronym*⟩ for storing the different forms of the acronym, the TEX code for the acronym is limited by `\csname`. If the acronym requires problematic or complicate TEX stuff (font commands, ...), then this code can be given in the optional argument ⟨*short name*⟩. The first argument ⟨*acronym*⟩ is then a simpler string to identify the acronym. For example, an acronym for water can look like this:

```
\acro{H2O}[$\mathrm{H_2O}$]{water}
```

Then `\acs{H2O}` gets "H_2O" and `\acl{H2O}` prints "water".

`\acroextra` Inside the `acronym` environment additional information can be added to the list of acronyms with the `\acroextra` command that will not be included in the normal inline acronyms.

`\acroextra{`⟨*additional info*⟩`}`

for example:

```
\acro{H2O}[$\mathrm{H_2O}$]
     {Dihydrogen Monoxide\acroextra{ (water)}}
\acro{NA}[\ensuremath{N_{\mathrm A}}]
     {Number of Avogadro\acroextra{ (See \S\protect\ref{A1})}}
```

Note that `\acroextra` must be inserted inside the `\acro` definition and that fragile commands must be protected. Be careful of unnecessary spaces.

The standard format of the acronym list is a `\description` environment. If you pass an optional parameter to the `acronym` environment, the width of the acronym-column will be fitted to the width of the given parameter (which should be the longest acronym). For example, if *HBCI* is the longest acronym used, the list should start with

```
\begin{acronym}[HBCI]
```

ANEXO D – Cras non urna sed feugiat cum sociis natoque penatibus

1: remover indice remissivo se não for necessário