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

HENRIQUE HIROMI SHIMADA	SP3039421
ISABELA SOUZA DUARTE	SP3030083
MATEUS SOUZA DA SILVA	SP3022374
VINICIUS GOMES MOREIRA	SP3039587
WELEN MOTA SOUSA	SP146616X

Metaverso - Ferramenta de Chamados de TI (ITSM) para pessoas físicas e pequenas empresas

São Paulo - SP - Brasil

ABRIL DE 2022

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

HENRIQUE HIROMI SHIMADA	SP3039421
ISABELA SOUZA DUARTE	SP3030083
MATEUS SOUZA DA SILVA	SP3022374
VINICIUS GOMES MOREIRA	SP3039587
WELEN MOTA SOUSA	SP146616X

Metaverso - Ferramenta de Chamados de TI (ITSM) para pessoas físicas e pequenas empresas

Professor: JOSE BRAZ DE ARAUJO
Professor: MARCELO TAVARES DE SANTANA

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

Curso Técnico em Informática Integrado ao Ensino Médio

PI1A5 - Projeto Integrado I

São Paulo - SP - Brasil
ABRIL DE 2022

Este trabalho é dedicado às crianças adultas que,
quando pequenas, sonharam em se tornar cientistas.

Agradecimentos

Os agradecimentos principais são direcionados à Gerald Weber, Miguel Frasson, Leslie H. Watter, Bruno Parente Lima, Flávio de Vasconcellos Corrêa, Otavio Real Salvador, Renato Machnievscz[1] e todos aqueles que contribuíram para que a produção de trabalhos acadêmicos conforme as normas ABNT com LaTeX fosse possível.

Agradecimentos especiais são direcionados ao Centro de Pesquisa em Arquitetura da Informação[2] da Universidade de Brasília (CPAI), ao grupo de usuários *latex-br*[3] e aos novos voluntários do grupo *abnTeX2*[4] que contribuíram e que ainda contribuirão para a evolução do abnTeX2.

[1] Os nomes dos integrantes do primeiro projeto abnTeX foram extraídos de <http://codigolivre.org.br/projects/abntex/>

[2] <http://www.cpai.unb.br/>

[3] <https://groups.google.com/group/latex-br>

[4] <https://groups.google.com/group/abntex2> e <http://abntex2.googlecode.com/>

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito."
(Bíblia Sagrada, Romanos 12, 2)

Resumo

De acordo com a norma NBR6028:2003 (3.1-3.2) , o resumo deve ressaltar o contexto, o objetivo, o método, os resultados e as conclusões do documento (portanto deve ser escrito por ultimo). A ordem e a extensão destes itens dependem do tipo de resumo (informativo ou indicativo) e do tratamento que cada item recebe no documento original. O resumo **deve ter um paragrafo único** e deve **ter entre 150 e 500 palavras para trabalhos acadêmicos ou entre 100 e 250 para artigos de periódicos**. O resumo deve ser precedido da referência do documento, com exceção do resumo inserido no próprio documento. (. . .) As palavras-chave devem figurar logo abaixo do resumo, antecedidas da expressão **Palavras-chave**:, separadas entre si por ponto e finalizadas também por ponto.

Palavras-chaves: latex. abntex. editoração de texto.

Abstract

This is the english abstract.

Keywords: latex. abntex. text editoration.

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

De acordo com a PNAD de 2019 **??**, 82,7% dos domicílios brasileiros contam com acesso à internet. Mesmo tendo acesso de banda larga em 80,2% dos dispositivos móveis do país, apenas 45,1% os domicílios da amostra têm acesso via computador. Assim, entendemos que embora muitos dos usuários têm alguma fluência em aplicativos móveis, as diferenças de interfaces pode ser um desafio para o usuário que têm tarefas diferentes das desempenhadas em aplicativos móveis.

Ainda, considerando que existem diversas soluções corporativas de suporte, em contraste, para esses usuários, as soluções que mais se apresentam são fóruns, que ainda que sejam gratuitas, ainda demandam algum trabalho e interação que os usuários alvo são pouco familiarizados ou têm compreensão limitados em relação à dinâmica de tais ferramentas.

Assim, propomos uma ferramenta com capacidade de suportar os usuários que sentem necessidade de suporte personalizado.

1.1 Questão de Pesquisa

Existência de potencial relevante de usuários de pouca fluência com soluções de tecnologia que têm dificuldades em resolver problemas frustantes e corriqueiros com computadores pessoais que incomodam ou impedem o uso esperado do dispositivo por não conseguirem desfrutar do dispositivo ou, mesmo, trabalhar.

O projeto elaborado propõe uma solução que atenda pessoas com baixa fluência em sistemas de computação em situações cotidianas em que seus dispositivos não funcionem de acordo com o esperado pelo usuário.

Entende-se que a solução tem como alvo pessoas físicas e pequenas empresas, que normalmente têm acesso limitado ou nenhum a ferramentas tradicionais de suporte de tecnologia da informação (*ITSM - Information Technology Service Management*).

Para tanto, foram definidos os seguintes objetivos para criação do serviço: elaboração do mínimo produto viável e suas ferramentas essenciais, com pontos de checagem (*check point*) para verificação do avanço da solução.

1.2 Objetivo Principal

Disponibilizar um serviço de assistência a problemas em sistemas computacionais domésticos e pequenas empresas para usuários finais com pouca ou nenhuma familiaridade

a problemas cotidianos.

1.3 Objetivos Secundários

Para que o produto de ITSM - *Information Technology Service Management* seja considerado viável, será necessário que as seguintes ferramentas sejam disponibilizadas as funcionalidades a seguir:

- Familiarizar o usuário com pouca habilidade para resolver problemas de soluções simples com computadores pra que possam ser independentes;
- Facilitar a identificação e aplicação de resolução a problemas com computadores;
- Orientar os gestores da ferramenta sobre as questões e dificuldades mais comuns;
- Facilitar o processo de resolução em caso de atendimento por terceiro credenciado

1.4 Justificativa

A utilização de recursos computacionais têm sido disponibilizados de forma acessível a mais e mais pessoas. Entretanto, a acessibilidade a tais recursos não significa que os indivíduos que adquirirem esses recursos como micro computadores e celulares inteligentes têm plenas capacidades de resolver problemas em seus dispositivos, mas que poderiam ser solucionadas com pesquisas simples em sites de busca.

A solução Metaverso é uma gratuita que facilita a identificação e resolução de problemas de usuários por conta própria ao mesmo tempo que ensine aos seus usuários como resolver suas dificuldades imediatas fortalece o vínculo comercial para resolver situações mais complexas através de uma ferramenta proprietária de chamados.

2 Revisão da Literatura

2.1 ITSM

2.2 Metodologia ágil

3 Gerenciamento do Projeto

3.1 Metodologia utilizada

A metodologia aplicada no desenvolvimento da ferramenta fez usos de práticas combinadas das metodologias Scrum e Kamban, com alterações pontuais para melhor adequação aos prazos da disciplina e práticas familiarizadas pelos componentes do grupo.

Assim, a equipe aaderiu aos seguintes artefatos do Scrum:

- Sprint Planning

  Em conjunto com as práticas do scrum, utilizamos o trello como ferramenta kanban para organizar as tarefas sendo desenvolvidas.

- Diagrama de caso de usos

  A fim de minimizar a questão de limitação de tempo, aplicou-se o diagrama de caso de uso. Assim, contorna-se a necessidade de refinar novas tarefas durante o desenvolvimento.

3.2 Equipe e Projeto

A equipe é composta por seis integrantes, a fim de cumprir o requisito de conclusão do curso de Análise e Desenvolvimento de Sistemas.

As tarefas foram distribuídas entre os integrantes, mas não restrita ao indivíduo. Essa distribuição tem por objetivo priorizar e nomear pontos focais.

- Henrique Hiromi Shimada

  Atuou como desenvolvedor de software por aproximadamente um ano, utilizando principalmente as linguagens Java e Python, serviços gerenciados da nuvem AWS. Atualmente, atua como consultor de entrega de pacotes em soluções corporativas na área dados IBM.

- Isabela Souza Duarte
- Mateus Souza da Silva
- Vinicius Gomes Moreira
- Welen Mota Sousa

SVN	https://svn.spo.ifsp.edu.br/viewvc/A6PGP/S202201-PI-NOT/Metaverso/
Blog	https://metaversoifsp.blogspot.com/
Youtube	https://www.youtube.com/channel/UCNXlPi5ADeGx1tZbMo_xOqw

Tarefa	Henrique	Isabela	Mateus	Vinicius	Welen
Front-end					
Back-end					
Banco de Dados					
Infraestrutura					
Testes					
Cobertura					
Documentação					
Blog					
Vídeos					
Gerenciamento					

3.2.1 Comunicação do Projeto

O progresso das etapas do projeto foi publicado em um blog, de acordo com a orientação dos professores.

3.2.2 Divisão de Tarefas

3.3 Sprints

3.3.1 Sprint 1

4 DESENVOLVIMENTO

4.1 Modelagem de dados

4.2 Histórias de usuários

4.3 Arquitetura da aplicação

4.4 Avaliação SSL

4.5 Análise de resposta HTTP

4.6 Análise HTML

4.7 Métricas do Projeto

4.8 Escolhas e Descartes

4.8.1 Funcionalidades descartadas

4.8.2 Mudanças na infraestrutura

4.9 Problemas enfrentados

4.10 Links do Projeto

4.11 Estatísticas SVN

5 Considerações Finais

Além desse documento ser um modelo de como pode ser criado um documento em LaTeX ele também apresenta diversas informações úteis para as disciplinas de projetos de informática do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) e alguns elementos uteis para as monografias do curso de Pós Graduação em Gestão de TI do IFSP.

Um trabalho de disciplina não tem "Conclusão"

Exemplo de possíveis seções para monografia da pós graduação...

5.1 Resposta à Questão de Pesquisa

5.2 Objetivos Propostos

5.3 Contribuições Acadêmicas e Gerenciais

5.4 Limitações da Pesquisa e Contribuições para Estudo

Glossário

Apêndices

APÊNDICE A – Quisque libero justo

APÊNDICE B – Nullam elementum urna vel imperdiet sodales elit ipsum pharetra ligula ac pretium ante justo a nulla curabitur tristique arcu eu metus

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

*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

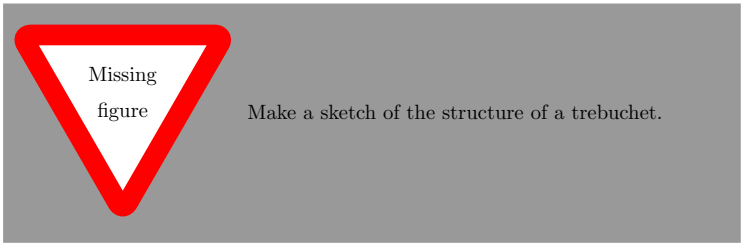
`\todo{Make a cake \ldots}`,

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

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

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})}}
```

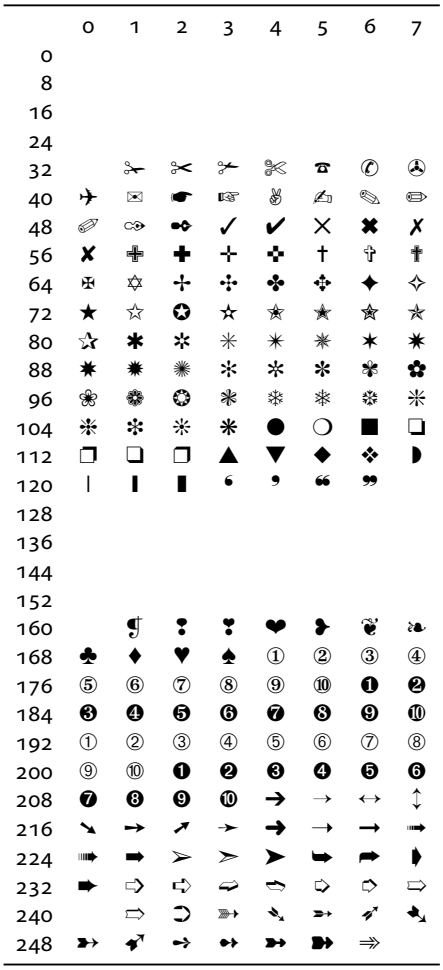
Note that `\acroextra` must be inserted inside the `\acro` definition and that fragile commands must be protected. Be careful of unnecessary spaces.

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 – Referência Rápida pifont

LaTeX pifont quick reference



Above are all of the characters available via the `ding` macro in the LaTeX pifont package. Add the row number to the column number to get the appropriate argument; e.g., `\ding{165}` produces a ❥ character.

http://web.willbenton.com