

# Org Mode マニュアル

---

リリース 9.1.14 (release'9.1.14)

**by Carsten Dominik**

with contributions by Bastien Guerry, Nicolas Goaziou, Eric Schulte, Jambunathan K, Dan Davison, Thomas Dye, David O'Toole, and Philip Rooke.

---

このマニュアルは、Org-mode 9.1.14 (release 9.1.14) に対応しています。

Copyright © 2004–2018 Free Software Foundation, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with the Front-Cover Texts being “A GNU Manual,” and with the Back-Cover Texts as in (a) below. A copy of the license is included in the section entitled “GNU Free Documentation License.”

(a) The FSF’s Back-Cover Text is: “You have the freedom to copy and modify this GNU manual.”

## Short Contents

|    |                                      |     |
|----|--------------------------------------|-----|
| 1  | Introduction .....                   | 1   |
| 2  | Document structure .....             | 6   |
| 3  | Tables .....                         | 19  |
| 4  | Hyperlinks .....                     | 38  |
| 5  | TODO items .....                     | 46  |
| 6  | Tags .....                           | 59  |
| 7  | Properties and columns .....         | 64  |
| 8  | Dates and times .....                | 73  |
| 9  | Capture - Refile - Archive .....     | 89  |
| 10 | Agenda views .....                   | 102 |
| 11 | Markup for rich export .....         | 131 |
| 12 | Exporting .....                      | 138 |
| 13 | Publishing .....                     | 188 |
| 14 | Working with source code .....       | 199 |
| 15 | Miscellaneous .....                  | 229 |
| A  | Hacking .....                        | 242 |
| B  | MobileOrg .....                      | 256 |
| C  | History and acknowledgments .....    | 258 |
| D  | GNU Free Documentation License ..... | 264 |
|    | Concept index .....                  | 272 |
|    | Key index .....                      | 283 |
|    | Command and function index .....     | 288 |
|    | Variable index .....                 | 292 |

# Table of Contents

|          |                                |           |
|----------|--------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>            | <b>1</b>  |
| 1.1      | Summary                        | 1         |
| 1.2      | Installation                   | 2         |
| 1.3      | Activation                     | 3         |
| 1.4      | Feedback                       | 3         |
| 1.5      | 本マニュアルで使われる植字ルール               | 4         |
| <b>2</b> | <b>Document structure</b>      | <b>6</b>  |
| 2.1      | Outlines                       | 6         |
| 2.2      | Headlines                      | 6         |
| 2.3      | Visibility cycling             | 6         |
| 2.3.1    | Global and local cycling       | 6         |
| 2.3.2    | Initial visibility             | 8         |
| 2.3.3    | Catching invisible edits       | 8         |
| 2.4      | Motion                         | 8         |
| 2.5      | Structure editing              | 9         |
| 2.6      | Sparse trees                   | 11        |
| 2.7      | Plain lists                    | 12        |
| 2.8      | Drawers                        | 15        |
| 2.9      | Blocks                         | 16        |
| 2.10     | Footnotes                      | 16        |
| 2.11     | The Orgstruct minor mode       | 17        |
| 2.12     | Org syntax                     | 18        |
| <b>3</b> | <b>Tables</b>                  | <b>19</b> |
| 3.1      | 組み込まれたテーブルエディタ                 | 19        |
| 3.2      | Column width and alignment     | 22        |
| 3.3      | Column groups                  | 23        |
| 3.4      | Orgtbl マイナーモード                 | 23        |
| 3.5      | The spreadsheet                | 24        |
| 3.5.1    | References                     | 24        |
| 3.5.2    | Formula syntax for Calc        | 26        |
| 3.5.3    | 数式としての Emacs Lisp 形式           | 28        |
| 3.5.4    | Durations and time values      | 29        |
| 3.5.5    | Field and range formulas       | 29        |
| 3.5.6    | Column formulas                | 30        |
| 3.5.7    | Lookup functions               | 30        |
| 3.5.8    | Editing and debugging formulas | 31        |
| 3.5.9    | Updating the table             | 33        |
| 3.5.10   | Advanced features              | 34        |
| 3.6      | Org-Plot                       | 35        |

|          |  |           |
|----------|--|-----------|
| <b>4</b> | <b>Hyperlinks</b>                        | <b>38</b> |
| 4.1      | Link format                              | 38        |
| 4.2      | Internal links                           | 38        |
| 4.2.1    | Radio targets                            | 39        |
| 4.3      | External links                           | 39        |
| 4.4      | Handling links                           | 41        |
| 4.5      | Using links outside Org                  | 43        |
| 4.6      | Link abbreviations                       | 44        |
| 4.7      | Search options in file links             | 45        |
| 4.8      | Custom Searches                          | 45        |
| <b>5</b> | <b>TODO items</b>                        | <b>46</b> |
| 5.1      | 基本的な TODO の機能                            | 46        |
| 5.2      | Extended use of TODO keywords            | 47        |
| 5.2.1    | ワークフローの状態としての TODO キーワード                 | 47        |
| 5.2.2    | 種類としての TODO キーワード                        | 48        |
| 5.2.3    | 同一ファイル内での複数のキーワードセット                     | 48        |
| 5.2.4    | Fast access to TODO states               | 49        |
| 5.2.5    | Setting up keywords for individual files | 49        |
| 5.2.6    | Faces for TODO keywords                  | 49        |
| 5.2.7    | TODO dependencies                        | 50        |
| 5.3      | Progress logging                         | 52        |
| 5.3.1    | Closing items                            | 52        |
| 5.3.2    | Tracking TODO state changes              | 52        |
| 5.3.3    | Tracking your habits                     | 53        |
| 5.4      | Priorities                               | 55        |
| 5.5      | タスクをサブタスクに細分化する。                         | 55        |
| 5.6      | Checkboxes                               | 56        |
| <b>6</b> | <b>Tags</b>                              | <b>59</b> |
| 6.1      | Tag inheritance                          | 59        |
| 6.2      | Setting tags                             | 59        |
| 6.3      | Tag hierarchy                            | 62        |
| 6.4      | Tag searches                             | 63        |
| <b>7</b> | <b>Properties and columns</b>            | <b>64</b> |
| 7.1      | Property syntax                          | 64        |
| 7.2      | Special properties                       | 66        |
| 7.3      | Property searches                        | 66        |
| 7.4      | プロパティの継承                                 | 67        |
| 7.5      | Column view                              | 67        |
| 7.5.1    | Defining columns                         | 68        |
| 7.5.1.1  | Scope of column definitions              | 68        |
| 7.5.1.2  | Column attributes                        | 68        |
| 7.5.2    | Using column view                        | 70        |
| 7.5.3    | Capturing column view                    | 71        |
| 7.6      | プロパティ API                                | 72        |

|           |   |            |
|-----------|---|------------|
| <b>8</b>  | <b>Dates and times</b>                      | <b>73</b>  |
| 8.1       | タイムスタンプ、デッドラインおよびスケジューリング                   | 73         |
| 8.2       | Creating timestamps                         | 74         |
| 8.2.1     | The date/time prompt                        | 75         |
| 8.2.2     | Custom time format                          | 77         |
| 8.3       | Deadlines and scheduling                    | 77         |
| 8.3.1     | デッドラインおよびスケジュールの挿入                          | 78         |
| 8.3.2     | Repeated tasks                              | 79         |
| 8.4       | Clocking work time                          | 81         |
| 8.4.1     | Clocking commands                           | 81         |
| 8.4.2     | The clock table                             | 83         |
| 8.4.3     | Resolving idle time and continuous clocking | 85         |
| 8.5       | Effort estimates                            | 87         |
| 8.6       | Taking notes with a timer                   | 87         |
| <b>9</b>  | <b>Capture - Refile - Archive</b>           | <b>89</b>  |
| 9.1       | Capture                                     | 89         |
| 9.1.1     | Setting up capture                          | 89         |
| 9.1.2     | Using capture                               | 89         |
| 9.1.3     | Capture templates                           | 90         |
| 9.1.3.1   | Template elements                           | 91         |
| 9.1.3.2   | Template expansion                          | 93         |
| 9.1.3.3   | Templates in contexts                       | 94         |
| 9.2       | Attachments                                 | 95         |
| 9.3       | RSS フィード                                    | 96         |
| 9.4       | 外部アクセスのためのプロトコル                             | 97         |
| 9.4.1     | store-link protocol                         | 97         |
| 9.4.2     | capture protocol                            | 97         |
| 9.4.3     | open-source protocol                        | 98         |
| 9.5       | Refile and copy                             | 99         |
| 9.6       | Archiving                                   | 100        |
| 9.6.1     | ツリーをアーカイブファイルへ移動                            | 100        |
| 9.6.2     | Internal archiving                          | 101        |
| <b>10</b> | <b>Agenda views</b>                         | <b>102</b> |
| 10.1      | Agenda files                                | 102        |
| 10.2      | アジェンダのコマンド選択画面                              | 103        |
| 10.3      | The built-in agenda views                   | 104        |
| 10.3.1    | The weekly/daily agenda                     | 104        |
| 10.3.2    | The global TODO list                        | 106        |
| 10.3.3    | Matching tags and properties                | 107        |
| 10.3.4    | Search view                                 | 110        |
| 10.3.5    | Stuck projects                              | 110        |
| 10.4      | Presentation and sorting                    | 111        |
| 10.4.1    | Categories                                  | 111        |
| 10.4.2    | Time-of-day specifications                  | 111        |
| 10.4.3    | Sorting agenda items                        | 112        |

|           |   |            |
|-----------|---|------------|
| 10.4.4    | Filtering/limiting agenda items.....                                    | 113        |
| 10.5      | Commands in the agenda buffer.....                                      | 115        |
| 10.6      | Custom agenda views.....  | 124        |
| 10.6.1    | Storing searches.....   | 124        |
| 10.6.2    | Block agenda.....   | 125        |
| 10.6.3    | Setting options for custom commands.....                                | 125        |
| 10.7      | Exporting agenda views.....   | 127        |
| 10.8      | Using column view in the agenda.....                                    | 129        |
| <b>11</b> | <b>Markup for rich export.....</b>                                      | <b>131</b> |
| 11.1      | 段落、改行、引用.....   | 131        |
| 11.2      | Emphasis and monospace.....   | 131        |
| 11.3      | Horizontal rules.....   | 132        |
| 11.4      | 画像と表.....   | 132        |
| 11.5      | Literal examples.....   | 132        |
| 11.6      | Special symbols.....  | 134        |
| 11.7      | Subscripts and superscripts.....  | 135        |
| 11.8      | Embedded L <sup>A</sup> T <sub>E</sub> X.....                           | 135        |
| 11.8.1    | L <sup>A</sup> T <sub>E</sub> X の断片的なコード.....                           | 135        |
| 11.8.2    | Previewing L <sup>A</sup> T <sub>E</sub> X fragments.....               | 136        |
| 11.8.3    | Using C <sup>D</sup> L <sup>A</sup> T <sub>E</sub> X to enter math..... | 137        |
| <b>12</b> | <b>Exporting.....</b>   | <b>138</b> |
| 12.1      | The export dispatcher.....  | 138        |
| 12.2      | Export settings.....  | 139        |
| 12.3      | Table of contents.....  | 142        |
| 12.4      | Include files.....  | 143        |
| 12.5      | Macro replacement.....  | 144        |
| 12.6      | Comment lines.....  | 145        |
| 12.7      | ASCII/Latin-1/UTF-8 export.....   | 145        |
| 12.8      | Beamer export.....  | 147        |
| 12.8.1    | Beamer export commands.....   | 147        |
| 12.8.2    | Beamer specific export settings.....                                    | 147        |
| 12.8.3    | Sectioning, Frames and Blocks in Beamer.....                            | 148        |
| 12.8.4    | Beamer specific syntax.....   | 149        |
| 12.8.5    | Editing support.....  | 149        |
| 12.8.6    | A Beamer example.....   | 150        |
| 12.9      | HTML export.....  | 150        |
| 12.9.1    | HTML エクスポートのコマンド.....   | 151        |
| 12.9.2    | HTML Specific export settings.....                                      | 151        |
| 12.9.3    | HTML doctypes.....  | 152        |
| 12.9.4    | HTML preamble and postamble.....  | 153        |
| 12.9.5    | Quoting HTML tags.....  | 153        |
| 12.9.6    | Links in HTML export.....   | 153        |
| 12.9.7    | Tables in HTML export.....  | 154        |
| 12.9.8    | Images in HTML export.....  | 154        |
| 12.9.9    | Math formatting in HTML export.....                                     | 155        |
| 12.9.10   | Text areas in HTML export.....  | 155        |

|          |   |     |
|----------|---|-----|
| 12.9.11  | CSS support .....   | 156 |
| 12.9.12  | ウェブページの表示に関する JavaScript のサポート .....                                  | 157 |
| 12.10    | L <sup>A</sup> T <sub>E</sub> X のエクスポート .....                         | 158 |
| 12.10.1  | L <sup>A</sup> T <sub>E</sub> X エクスポートのコマンド .....                     | 158 |
| 12.10.2  | L <sup>A</sup> T <sub>E</sub> X specific export settings .....        | 159 |
| 12.10.3  | L <sup>A</sup> T <sub>E</sub> X header and sectioning structure ..... | 160 |
| 12.10.4  | Quoting L <sup>A</sup> T <sub>E</sub> X code .....                    | 160 |
| 12.10.5  | L <sup>A</sup> T <sub>E</sub> X エクスポートにおけるテーブル .....                  | 161 |
| 12.10.6  | L <sup>A</sup> T <sub>E</sub> X エクスポートにおける画像 .....                    | 162 |
| 12.10.7  | Plain lists in L <sup>A</sup> T <sub>E</sub> X export .....           | 163 |
| 12.10.8  | Source blocks in L <sup>A</sup> T <sub>E</sub> X export .....         | 164 |
| 12.10.9  | Example blocks in L <sup>A</sup> T <sub>E</sub> X export .....        | 164 |
| 12.10.10 | Special blocks in L <sup>A</sup> T <sub>E</sub> X export .....        | 165 |
| 12.10.11 | Horizontal rules in L <sup>A</sup> T <sub>E</sub> X export .....      | 165 |
| 12.11    | Markdown export .....   | 165 |
| 12.12    | OpenDocument Text export .....  | 166 |
| 12.12.1  | Pre-requisites for ODT export .....                                   | 166 |
| 12.12.2  | ODT export commands .....   | 166 |
| 12.12.3  | ODT specific export settings .....                                    | 167 |
| 12.12.4  | Extending ODT export .....  | 167 |
| 12.12.5  | Applying custom styles .....  | 168 |
| 12.12.6  | Links in ODT export .....   | 168 |
| 12.12.7  | Tables in ODT export .....  | 168 |
| 12.12.8  | Images in ODT export .....  | 169 |
| 12.12.9  | Math formatting in ODT export .....                                   | 170 |
| 12.12.10 | Labels and captions in ODT export .....                               | 171 |
| 12.12.11 | Literal examples in ODT export .....                                  | 172 |
| 12.12.12 | Advanced topics in ODT export .....                                   | 172 |
| 12.13    | Org export .....  | 177 |
| 12.14    | Texinfo export .....  | 177 |
| 12.14.1  | Texinfo export commands .....   | 177 |
| 12.14.2  | Texinfo specific export settings .....                                | 177 |
| 12.14.3  | Texinfo file header .....   | 178 |
| 12.14.4  | Texinfo title and copyright page .....                                | 178 |
| 12.14.5  | Info directory file .....   | 179 |
| 12.14.6  | Headings and sectioning structure .....                               | 179 |
| 12.14.7  | Indices .....   | 180 |
| 12.14.8  | Quoting Texinfo code .....  | 180 |
| 12.14.9  | Plain lists in Texinfo export .....                                   | 180 |
| 12.14.10 | Tables in Texinfo export .....  | 181 |
| 12.14.11 | Images in Texinfo export .....  | 181 |
| 12.14.12 | Special blocks .....  | 181 |
| 12.14.13 | A Texinfo example .....   | 181 |
| 12.15    | iCalendar export .....  | 183 |
| 12.16    | Other built-in back-ends .....  | 184 |
| 12.17    | Advanced configuration .....  | 184 |
| 12.18    | Export in foreign buffers .....                                       | 187 |



|           |                                      |            |
|-----------|--------------------------------------|------------|
| <b>13</b> | <b>Publishing</b>                    | <b>188</b> |
| 13.1      | Configuration                        | 188        |
| 13.1.1    | <code>org-publishing-alist</code> 変数 | 188        |
| 13.1.2    | ファイルの送り元と送り先                         | 188        |
| 13.1.3    | Selecting files                      | 189        |
| 13.1.4    | Publishing action                    | 189        |
| 13.1.5    | Options for the exporters            | 190        |
| 13.1.6    | 公開ファイル間のリンク                          | 194        |
| 13.1.7    | サイトマップの生成                            | 194        |
| 13.1.8    | Generating an index                  | 195        |
| 13.2      | Uploading files                      | 196        |
| 13.3      | Sample configuration                 | 196        |
| 13.3.1    | 例：シンプルな公開用の設定                        | 196        |
| 13.3.2    | 例：複雑な公開用の設定                          | 197        |
| 13.4      | Triggering publication               | 197        |
| <b>14</b> | <b>Working with source code</b>      | <b>199</b> |
| 14.1      | Structure of code blocks             | 200        |
| 14.2      | Editing source code                  | 201        |
| 14.3      | Exporting code blocks                | 202        |
| 14.4      | Extracting source code               | 203        |
| 14.5      | Evaluating code blocks               | 204        |
| 14.6      | Library of Babel                     | 205        |
| 14.7      | Languages                            | 205        |
| 14.8      | Header arguments                     | 206        |
| 14.8.1    | Using header arguments               | 206        |
| 14.8.2    | Specific header arguments            | 208        |
| 14.8.2.1  | <code>:var</code>                    | 209        |
| 14.8.2.2  | <code>:results</code>                | 213        |
| 14.8.2.3  | <code>:file</code>                   | 214        |
| 14.8.2.4  | <code>:file-desc</code>              | 214        |
| 14.8.2.5  | <code>:file-ext</code>               | 214        |
| 14.8.2.6  | <code>:output-dir</code>             | 214        |
| 14.8.2.7  | <code>:dir</code> とリモートでの実行          | 215        |
| 14.8.2.8  | <code>:exports</code>                | 215        |
| 14.8.2.9  | <code>:tangle</code>                 | 216        |
| 14.8.2.10 | <code>:mkdirp</code>                 | 216        |
| 14.8.2.11 | <code>:comments</code>               | 216        |
| 14.8.2.12 | <code>:padline</code>                | 216        |
| 14.8.2.13 | <code>:no-expand</code>              | 216        |
| 14.8.2.14 | <code>:session</code>                | 217        |
| 14.8.2.15 | <code>:noweb</code>                  | 217        |
| 14.8.2.16 | <code>:noweb-ref</code>              | 218        |
| 14.8.2.17 | <code>:noweb-sep</code>              | 219        |
| 14.8.2.18 | <code>:cache</code>                  | 219        |
| 14.8.2.19 | <code>:sep</code>                    | 220        |
| 14.8.2.20 | <code>:hlines</code>                 | 220        |
| 14.8.2.21 | <code>:colnames</code>               | 221        |

|                   |   |            |
|-------------------|---|------------|
| 14.8.2.22         | <code>:rownames</code> .....                      | 222        |
| 14.8.2.23         | <code>:shebang</code> .....                       | 222        |
| 14.8.2.24         | <code>:tangle-mode</code> .....                   | 222        |
| 14.8.2.25         | <code>:eval</code> .....                          | 223        |
| 14.8.2.26         | <code>:wrap</code> .....                          | 223        |
| 14.8.2.27         | <code>:post</code> .....                          | 223        |
| 14.8.2.28         | <code>:prologue</code> .....                      | 224        |
| 14.8.2.29         | <code>:epilogue</code> .....                      | 224        |
| 14.9              | Results of evaluation .....                       | 224        |
| 14.9.1            | Non-session .....                                 | 225        |
| 14.9.1.1          | <code>:results value</code> .....                 | 225        |
| 14.9.1.2          | <code>:results output</code> .....                | 225        |
| 14.9.2            | Session .....                                     | 225        |
| 14.9.2.1          | <code>:results value</code> .....                 | 225        |
| 14.9.2.2          | <code>:results output</code> .....                | 225        |
| 14.10             | Noweb reference syntax .....                      | 226        |
| 14.11             | Key bindings and useful functions .....           | 227        |
| 14.12             | Batch execution .....                             | 228        |
| <b>15</b>         | <b>Miscellaneous .....</b>                        | <b>229</b> |
| 15.1              | Completion .....                                  | 229        |
| 15.2              | Easy templates .....                              | 229        |
| 15.3              | Speed keys .....                                  | 230        |
| 15.4              | コードの評価とセキュリティの問題 .....                            | 230        |
| 15.5              | Customization .....                               | 231        |
| 15.6              | バッファ中での設定の要約 .....                                | 231        |
| 15.7              | The very busy C-c C-c key .....                   | 235        |
| 15.8              | より見やすいアウトラインビュー .....                             | 235        |
| 15.9              | Using Org on a tty .....                          | 237        |
| 15.10             | Interaction with other packages .....             | 237        |
| 15.10.1           | Packages that Org cooperates with .....           | 238        |
| 15.10.2           | Packages that conflict with Org mode .....        | 239        |
| 15.11             | <code>org-crypt.el</code> .....                   | 240        |
| <b>Appendix A</b> | <b>Hacking .....</b>                              | <b>242</b> |
| A.1               | Hooks .....                                       | 242        |
| A.2               | Add-on packages .....                             | 242        |
| A.3               | Adding hyperlink types .....                      | 242        |
| A.4               | Adding export back-ends .....                     | 243        |
| A.5               | Context-sensitive commands .....                  | 244        |
| A.6               | Tables and lists in arbitrary syntax .....        | 244        |
| A.6.1             | Radio tables .....                                | 244        |
| A.6.2             | L <sup>A</sup> T <sub>E</sub> X でのラジオテーブルの例 ..... | 245        |
| A.6.3             | Translator functions .....                        | 247        |
| A.6.4             | Radio lists .....                                 | 247        |
| A.7               | Dynamic blocks .....                              | 248        |
| A.8               | Special agenda views .....                        | 249        |
| A.9               | Speeding up your agendas .....                    | 251        |

|                                   |                                       |            |
|-----------------------------------|---------------------------------------|------------|
| A.10                              | Extracting agenda information .....   | 251        |
| A.11                              | Using the property API .....          | 252        |
| A.12                              | Using the mapping API .....           | 254        |
| <b>Appendix B</b>                 | <b>MobileOrg .....</b>                | <b>256</b> |
| B.1                               | Setting up the staging area .....     | 256        |
| B.2                               | Pushing to MobileOrg .....            | 256        |
| B.3                               | Pulling from MobileOrg .....          | 257        |
| <b>Appendix C</b>                 | <b>History and acknowledgments</b>    |            |
|                                   | .....                                 | <b>258</b> |
| C.1                               | From Carsten .....                    | 258        |
| C.2                               | From Bastien .....                    | 259        |
| C.3                               | List of contributions .....           | 259        |
| <b>Appendix D</b>                 | <b>GNU Free Documentation License</b> |            |
|                                   | .....                                 | <b>264</b> |
| <b>Concept index</b>              | .....                                 | <b>272</b> |
| <b>Key index</b>                  | .....                                 | <b>283</b> |
| <b>Command and function index</b> | .....                                 | <b>288</b> |
| <b>Variable index</b>             | .....                                 | <b>292</b> |

# 1 Introduction

## 1.1 Summary

Org is a mode for keeping notes, maintaining TODO lists, and project planning with a fast and effective plain-text system. It also is an authoring system with unique support for literate programming and reproducible research.

Org is implemented on top of Outline mode, which makes it possible to keep the content of large files well structured. Visibility cycling and structure editing help to work with the tree. Tables are easily created with a built-in table editor. Plain text URL-like links connect to websites, emails, Usenet messages, BBDB entries, and any files related to the projects.

Org develops organizational tasks around notes files that contain lists or information about projects as plain text. Project planning and task management makes use of metadata which is part of an outline node. Based on this data, specific entries can be extracted in queries and create dynamic *agenda views* that also integrate the Emacs calendar and diary. Org can be used to implement many different project planning schemes, such as David Allen's GTD system.

Org files can serve as a single source authoring system with export to many different formats such as HTML, L<sup>A</sup>T<sub>E</sub>X, Open Document, and Markdown. New export backends can be derived from existing ones, or defined from scratch.

Org files can include source code blocks, which makes Org uniquely suited for authoring technical documents with code examples. Org source code blocks are fully functional; they can be evaluated in place and their results can be captured in the file. This makes it possible to create a single file reproducible research compendium.

Org keeps simple things simple. When first fired up, it should feel like a straightforward, easy to use outliner. Complexity is not imposed, but a large amount of functionality is available when needed. Org is a toolbox. Many users actually run only a (very personal) fraction of Org's capabilities, and know that there is more whenever they need it.

All of this is achieved with strictly plain text files, the most portable and future-proof file format. Org runs in Emacs. Emacs is one of the most widely ported programs, so that Org mode is available on every major platform.

There is a website for Org which provides links to the newest version of Org, as well as additional information, frequently asked questions (FAQ), links to tutorials, etc. This page is located at <https://orgmode.org>.

An earlier version (7.3) of this manual is available as a paperback book from Network Theory Ltd. (<http://www.network-theory.co.uk/org/manual/>)

## 1.2 Installation

Org is part of recent distributions of GNU Emacs, so you normally don't need to install it. If, for one reason or another, you want to install Org on top of this pre-packaged version, there are three ways to do it:

- By using Emacs package system.
- By downloading Org as an archive.
- By using Org's git repository.

We **strongly recommend** to stick to a single installation method.

### Using Emacs packaging system

Recent Emacs distributions include a packaging system which lets you install Emacs libraries. You can install Org with *M-x package-install RET org*.

**Important:** you need to do this in a session where no `.org` file has been visited, i.e., where no Org built-in function have been loaded. Otherwise autoload Org functions will mess up the installation.

Then, to make sure your Org configuration is taken into account, initialize the package system with `(package-initialize)` in your Emacs init file before setting any Org option. If you want to use Org's package repository, check out the Org ELPA page (<https://orgmode.org/elpa.html>).

### Downloading Org as an archive

You can download Org latest release from Org's website (<https://orgmode.org/>). In this case, make sure you set the load-path correctly in your Emacs init file:

```
(add-to-list 'load-path "~/path/to/orgdir/lisp")
```

The downloaded archive contains contributed libraries that are not included in Emacs. If you want to use them, add the `contrib` directory to your load-path:

```
(add-to-list 'load-path "~/path/to/orgdir/contrib/lisp" t)
```

Optionally, you can compile the files and/or install them in your system. Run `make help` to list compilation and installation options.

### Using Org's git repository

You can clone Org's repository and install Org like this:

```
$ cd ~/src/
$ git clone https://code.orgmode.org/bzg/org-mode.git
$ cd org-mode/
$ make autoloads
```

Note that in this case, `make autoloads` is mandatory: it defines Org's version in `org-version.el` and Org's autoloads in `org-loaddefs.el`.

Remember to add the correct load-path as described in the method above.

You can also compile with `make`, generate the documentation with `make doc`, create a local configuration with `make config` and install Org with `make install`. Please run `make help` to get the list of compilation/installation options.

For more detailed explanations on Org's build system, please check the Org Build System page on Worg (<https://orgmode.org/worg/dev/org-build-system.html>).

## 1.3 Activation

Org mode buffers need font-lock to be turned on: this is the default in Emacs<sup>1</sup>.

There are compatibility issues between Org mode and some other Elisp packages, please take the time to check the list (see Section 15.10.2 [Conflicts], page 239).

The four Org commands `org-store-link`, `org-capture`, `org-agenda`, and `org-switchb` should be accessible through global keys (i.e., anywhere in Emacs, not just in Org buffers). Here are suggested bindings for these keys, please modify the keys to your own liking.

```
(global-set-key "\C-cl" 'org-store-link)
(global-set-key "\C-ca" 'org-agenda)
(global-set-key "\C-cc" 'org-capture)
(global-set-key "\C-cb" 'org-switchb)
```

Files with the `.org` extension use Org mode by default. To turn on Org mode in a file that does not have the extension `.org`, make the first line of a file look like this:

```
MY PROJECTS    -*- mode: org; -*-
```

which will select Org mode for this buffer no matter what the file's name is. See also the variable `org-insert-mode-line-in-empty-file`.

Many commands in Org work on the region if the region is *active*. To make use of this, you need to have `transient-mark-mode` turned on, which is the default. If you do not like `transient-mark-mode`, you can create an active region by using the mouse to select a region, or pressing *C-SPC* twice before moving the cursor.

## 1.4 Feedback

If you find problems with Org, or if you have questions, remarks, or ideas about it, please mail to the Org mailing list `emacs-orgmode@gnu.org`. You can subscribe to the list on this web page (<https://lists.gnu.org/mailman/listinfo/emacs-orgmode>). If you are not a member of the mailing list, your mail will be passed to the list after a moderator has approved it<sup>2</sup>.

For bug reports, please first try to reproduce the bug with the latest version of Org available—if you are running an outdated version, it is quite possible that the bug has been fixed already. If the bug persists, prepare a report and provide as much information as possible, including the version information of Emacs (*M-x emacs-version RET*) and Org (*M-x org-version RET*), as well as the Org related setup in the Emacs init file. The easiest way to do this is to use the command

```
M-x org-submit-bug-report RET
```

which will put all this information into an Emacs mail buffer so that you only need to add your description. If you are not sending the Email from within Emacs, please copy and paste the content into your Email program.

Sometimes you might face a problem due to an error in your Emacs or Org mode setup. Before reporting a bug, it is very helpful to start Emacs with minimal customizations and

<sup>1</sup> If you don't use font-lock globally, turn it on in Org buffer with `(add-hook 'org-mode-hook 'turn-on-font-lock)`

<sup>2</sup> Please consider subscribing to the mailing list, in order to minimize the work the mailing list moderators have to do.

reproduce the problem. Doing so often helps you determine if the problem is with your customization or with Org mode itself. You can start a typical minimal session with a command like the example below.

```
$ emacs -Q -l /path/to/minimal-org.el
```

However if you are using Org mode as distributed with Emacs, a minimal setup is not necessary. In that case it is sufficient to start Emacs as `emacs -Q`. The `minimal-org.el` setup file can have contents as shown below.

```
;;; Minimal setup to load latest 'org-mode'

;; activate debugging
(setq debug-on-error t
      debug-on-signal nil
      debug-on-quit nil)

;; add latest org-mode to load path
(add-to-list 'load-path "/path/to/org-mode/lisp")
(add-to-list 'load-path "/path/to/org-mode/contrib/lisp" t)
```

もし Org-mode を使っていてエラーが発生したら、バックトレースがとても役立ちます（作り方は次節を参照してください）。しばしば例となる小さなファイルが問題解決の手助けになります。それらには次のような明瞭な情報を含んでいます。

1. 正確に何を実行したのか
2. 何が起きることを期待していたのか
3. 期待と異なり何が起こったのか

Org-mode の改善にご協力いただき感謝します。

有用なバックトレースを生成する方法

Org-mode を利用していて理解できないメッセージのエラーが発生したら、バグを発見した可能性があります。エラーを報告する最良の方法は、すでに説明したバグレポートの書式に加えて、*backtrace* を提供することです。バックトレースは、ビルトインされたデバグによるエラーの発生箇所とどのように発生したかについての情報を含みます。以下に、有用なバックトレースを生成する手順を示します。

1. Reload uncompiled versions of all Org mode Lisp files. The backtrace contains much more information if it is produced with uncompiled code. To do this, use

```
C-u M-x org-reload RET
```

もしくは、Org -> Refresh/Reload -> Reload Org uncompiledをメニューから選択します。

2. Go to the Options menu and select Enter Debugger on Error.
3. エラーを再現するために必要な操作を行ってください。実行した操作を忘れずにメモしておいてください。
4. エラーが再現されると、**\*Backtrace\***バッファが画面上に表示されます。このバッファを別のファイルとして保存し（例えば `C-x C-w` を使って）、バグレポートに添付します。

## 1.5 本マニュアルで使われる植字ルール

## TODO keywords, tags, properties, etc.

Org mainly uses three types of keywords: TODO keywords, tags and property names. In this manual we use the following conventions:

### TODO

**WAITING**    TODO キーワードは、すべて大文字で記述されます。ユーザーが定義する場合も同様です。

### boss

**ARCHIVE**    ユーザー定義のタグは、小文字で記述されます。特別な意味を持つビルトインされたタグは、すべて大文字で記述されます。

### Release

**PRIORITY**   ユーザー定義のプロパティは、大文字で始めて残りが小文字で記述されます。特別な意味を持つビルトインされたプロパティは、すべて大文字で記述されます。

Moreover, Org uses *option keywords* (like `#+TITLE` to set the title) and *environment keywords* (like `#+BEGIN_EXPORT html` to start a HTML environment). They are written in uppercase in the manual to enhance its readability, but you can use lowercase in your Org file.

## Key bindings and commands

The manual suggests a few global key bindings, in particular `C-c a` for `org-agenda` and `C-c c` for `org-capture`. These are only suggestions, but the rest of the manual assumes that these key bindings are in place in order to list commands by key access.

Also, the manual lists both the keys and the corresponding commands for accessing a functionality. Org mode often uses the same key for different functions, depending on context. The command that is bound to such keys has a generic name, like `org-metaright`. In the manual we will, wherever possible, give the function that is internally called by the generic command. For example, in the chapter on document structure, `M-RIGHT` will be listed to call `org-do-demote`, while in the chapter on tables, it will be listed to call `org-table-move-column-right`. If you prefer, you can compile the manual without the command names by unsetting the flag `cmdnames` in `org.texi`.



## 2 Document structure

Org-mode は、Outline mode をベースとしており、ドキュメントの構造を編集するためにフレキシブルなコマンドを用意しています。

### 2.1 Outlines

Org is implemented on top of Outline mode. Outlines allow a document to be organized in a hierarchical structure, which (at least for me) is the best representation of notes and thoughts. An overview of this structure is achieved by folding (hiding) large parts of the document to show only the general document structure and the parts currently being worked on. Org greatly simplifies the use of outlines by compressing the entire show/hide functionality into a single command, `org-cycle`, which is bound to the `TAB` key.

### 2.2 Headlines

Headlines define the structure of an outline tree. The headlines in Org start with one or more stars, on the left margin<sup>12</sup>. For example:

```
* Top level headline
** Second level
*** 3rd level
    some text
*** 3rd level
    more text

* Another top level headline
```

Note that a headline named after `org-footnote-section`, which defaults to ‘Footnotes’, is considered as special. A subtree with this headline will be silently ignored by exporting functions.

Some people find the many stars too noisy and would prefer an outline that has white-space followed by a single star as headline starters. Section 15.8 [Clean view], page 235, describes a setup to realize this.

An empty line after the end of a subtree is considered part of it and will be hidden when the subtree is folded. However, if you leave at least two empty lines, one empty line will remain visible after folding the subtree, in order to structure the collapsed view. See the variable `org-cycle-separator-lines` to modify this behavior.

### 2.3 Visibility cycling

#### 2.3.1 Global and local cycling

アウトラインによって、バッファの中で、テキストの一部を隠すことが可能となります。Org-mode はバッファ内での表示の状況を変更するために、`TAB`と `S-TAB`とに結びついた 2 つのコマンドを使用します。

<sup>1</sup> See the variables `org-special-ctrl-a/e`, `org-special-ctrl-k`, and `org-ctrl-k-protect-subtree` to configure special behavior of `C-a`, `C-e`, and `C-k` in headlines.

<sup>2</sup> Clocking only works with headings indented less than 30 stars.

|                        |  |                                    |
|------------------------|--|------------------------------------|
| <b>TAB</b>             |  | <b>org-cycle</b>                   |
|                        | <i>Subtree cycling</i> : カレントのサブツリーの状態を順番に表示します。   |                                    |
|                        | <pre> -&gt; FOLDED -&gt; CHILDREN -&gt; SUBTREE --. '-----' </pre>   |                                    |
|                        | The cursor must be on a headline for this to work <sup>3</sup> .   |                                    |
| <b>S-TAB</b>           |  | <b>org-global-cycle</b>            |
| <b>C-u TAB</b>         | <i>Global cycling</i> : Rotate the entire buffer among the states  |                                    |
|                        | <pre> -&gt; OVERVIEW -&gt; CONTENTS -&gt; SHOW ALL --. '-----' </pre>  |                                    |
|                        | When <i>S-TAB</i> is called with a numeric prefix argument N, the CONTENTS view up to headlines of level N will be shown. Note that inside tables, <i>S-TAB</i> jumps to the previous field.   |                                    |
|                        | You can run global cycling using TAB only if point is at the very beginning of the buffer, but not on a headline, and <code>org-cycle-global-at-bob</code> is set to a non-nil value.  |                                    |
| <b>C-u C-u TAB</b>     |  | <b>org-set-startup-visibility</b>  |
|                        | Switch back to the startup visibility of the buffer (see Section 2.3.2 [Initial visibility], page 8).  |                                    |
| <b>C-u C-u C-u TAB</b> |  | <b>outline-show-all</b>            |
|                        | Show all, including drawers.   |                                    |
| <b>C-c C-r</b>         |  | <b>org-reveal</b>                  |
|                        | Reveal context around point, showing the current entry, the following heading and the hierarchy above. Useful for working near a location that has been exposed by a sparse tree command (see Section 2.6 [Sparse trees], page 11) or an agenda command (see Section 10.5 [Agenda commands], page 115). With a prefix argument show, on each level, all sibling headings. With a double prefix argument, also show the entire subtree of the parent. |                                    |
| <b>C-c C-k</b>         |  | <b>outline-show-branches</b>       |
|                        | Expose all the headings of the subtree, CONTENTS view for just one subtree.  |                                    |
| <b>C-c TAB</b>         |  | <b>outline-show-children</b>       |
|                        | Expose all direct children of the subtree. With a numeric prefix argument N, expose all children down to level N.  |                                    |
| <b>C-c C-x b</b>       |  | <b>org-tree-to-indirect-buffer</b> |
|                        | Show the current subtree in an indirect buffer <sup>4</sup> . With a numeric prefix argument N, go up to level N and then take that tree. If N is negative then go up that many levels. With a <i>C-u</i> prefix, do not remove the previously used indirect buffer.   |                                    |

<sup>3</sup> see, however, the option `org-cycle-emulate-tab`.

<sup>4</sup> The indirect buffer (see Section “Indirect Buffers” in *GNU Emacs Manual*) will contain the entire buffer, but will be narrowed to the current tree. Editing the indirect buffer will also change the original buffer, but without affecting visibility in that buffer.

`C-c C-x v` `org-copy-visible`  
 Copy the *visible* text in the region into the kill ring.

### 2.3.2 Initial visibility

When Emacs first visits an Org file, the global state is set to OVERVIEW, i.e., only the top level headlines are visible<sup>5</sup>. This can be configured through the variable `org-startup-folded`, or on a per-file basis by adding one of the following lines anywhere in the buffer:

```
#+STARTUP: overview
#+STARTUP: content
#+STARTUP: showall
#+STARTUP: showeverything
```

Furthermore, any entries with a `VISIBILITY` property (see Chapter 7 [Properties and columns], page 64) will get their visibility adapted accordingly. Allowed values for this property are `folded`, `children`, `content`, and `all`.

`C-u C-u TAB` `org-set-startup-visibility`  
 Switch back to the startup visibility of the buffer, i.e., whatever is requested by startup options and `VISIBILITY` properties in individual entries.

### 2.3.3 Catching invisible edits

Sometimes you may inadvertently edit an invisible part of the buffer and be confused on what has been edited and how to undo the mistake. Setting `org-catch-invisible-edits` to non-`nil` will help prevent this. See the docstring of this option on how Org should catch invisible edits and process them.

## 2.4 Motion

以下のコマンドはバッファの中で他の見出しにジャンプするものです。

|   |   |
|---|---|
| <code>C-c C-n</code>  | <code>org-next-visible-heading</code>     |
| Next heading.   |   |
| <code>C-c C-p</code>  | <code>org-previous-visible-heading</code> |
| Previous heading.   |   |
| <code>C-c C-f</code>  | <code>org-forward-same-level</code>       |
| Next heading same level.  |   |
| <code>C-c C-b</code>  | <code>org-backward-same-level</code>      |
| Previous heading same level.  |   |
| <code>C-c C-u</code>  | <code>outline-up-heading</code>           |
| Backward to higher level heading.   |   |
| <code>C-c C-j</code>  | <code>org-goto</code>                     |
| Jump to a different place without changing the current outline visibility. Shows the document structure in a temporary buffer, where you can use the following keys to find your destination: |   |

---

<sup>5</sup> When `org-agenda-inhibit-startup` is non-`nil`, Org will not honor the default visibility state when first opening a file for the agenda (see Section A.9 [Speeding up your agendas], page 251).

|  |                                    |
|--|------------------------------------|
| TAB  | Cycle visibility.                  |
| DOWN / UP  | Next/previous visible headline.    |
| RET  | Select this location.              |
| /  | Do a Sparse-tree search            |
| The following keys work if you turn off <code>org-goto-auto-isearch</code> |                                    |
| n / p  | Next/previous visible headline.    |
| f / b  | Next/previous headline same level. |
| u  | One level up.                      |
| 0-9  | Digit argument.                    |
| q  | Quit                               |

See also the option `org-goto-interface`.

## 2.5 Structure editing

|  |  |
|--|--|
| <i>M-RET</i>   | <code>org-meta-return</code>                         |
| Insert a new heading, item or row.   |  |
| If the command is used at the <i>beginning</i> of a line, and if there is a heading or a plain list item (see Section 2.7 [Plain lists], page 12) at point, the new heading/item is created <i>before</i> the current line. When used at the beginning of a regular line of text, turn that line into a heading. |  |
| When this command is used in the middle of a line, the line is split and the rest of the line becomes the new item or headline. If you do not want the line to be split, customize <code>org-M-RET-may-split-line</code> .   |  |
| Calling the command with a <code>C-u</code> prefix unconditionally inserts a new heading at the end of the current subtree, thus preserving its contents. With a double <code>C-u C-u</code> prefix, the new heading is created at the end of the parent subtree instead.  |  |
| <i>C-RET</i>   | <code>org-insert-heading-respect-content</code>      |
| Insert a new heading at the end of the current subtree.  |  |
| <i>M-S-RET</i>   | <code>org-insert-todo-heading</code>                 |
| Insert new TODO entry with same level as current heading. See also the variable <code>org-treat-insert-todo-heading-as-state-change</code> .   |  |
| <i>C-S-RET</i>   | <code>org-insert-todo-heading-respect-content</code> |
| Insert new TODO entry with same level as current heading. Like <i>C-RET</i> , the new headline will be inserted after the current subtree.   |  |
| <i>TAB</i>   | <code>org-cycle</code>                               |
| In a new entry with no text yet, the first TAB demotes the entry to become a child of the previous one. The next TAB makes it a parent, and so on, all the way to top level. Yet another TAB, and you are back to the initial level.   |  |
| <i>M-LEFT</i>  | <code>org-do-promote</code>                          |
| Promote current heading by one level.  |  |
| <i>M-RIGHT</i>   | <code>org-do-demote</code>                           |
| Demote current heading by one level.   |  |

|   |  |
|---|--|
| <i>M-S-LEFT</i>   | <code>org-promote-subtree</code>               |
| Promote the current subtree by one level.   |  |
| <i>M-S-RIGHT</i>  | <code>org-demote-subtree</code>                |
| Demote the current subtree by one level.  |  |
| <i>M-UP</i>   | <code>org-move-subtree-up</code>               |
| Move subtree up (swap with previous subtree of same level).   |  |
| <i>M-DOWN</i>   | <code>org-move-subtree-down</code>             |
| Move subtree down (swap with next subtree of same level).   |  |
| <i>M-h</i>  | <code>org-mark-element</code>                  |
| Mark the element at point. Hitting repeatedly will mark subsequent elements of the one just marked. E.g., hitting <i>M-h</i> on a paragraph will mark it, hitting <i>M-h</i> immediately again will mark the next one.  |  |
| <i>C-c @</i>  | <code>org-mark-subtree</code>                  |
| Mark the subtree at point. Hitting repeatedly will mark subsequent subtrees of the same level than the marked subtree.  |  |
| <i>C-c C-x C-w</i>  | <code>org-cut-subtree</code>                   |
| Kill subtree, i.e., remove it from buffer but save in kill ring. With a numeric prefix argument N, kill N sequential subtrees.  |  |
| <i>C-c C-x M-w</i>  | <code>org-copy-subtree</code>                  |
| Copy subtree to kill ring. With a numeric prefix argument N, copy the N sequential subtrees.  |  |
| <i>C-c C-x C-y</i>  | <code>org-paste-subtree</code>                 |
| Yank subtree from kill ring. This does modify the level of the subtree to make sure the tree fits in nicely at the yank position. The yank level can also be specified with a numeric prefix argument, or by yanking after a headline marker like <code>****</code> .   |  |
| <i>C-y</i>  | <code>org-yank</code>                          |
| Depending on the options <code>org-yank-adjusted-subtrees</code> and <code>org-yank-folded-subtrees</code> , Org's internal <code>yank</code> command will paste subtrees folded and in a clever way, using the same command as <i>C-c C-x C-y</i> . With the default settings, no level adjustment will take place, but the yanked tree will be folded unless doing so would swallow text previously visible. Any prefix argument to this command will force a normal <code>yank</code> to be executed, with the prefix passed along. A good way to force a normal yank is <i>C-u C-y</i> . If you use <code>yank-pop</code> after a yank, it will yank previous kill items plainly, without adjustment and folding. |  |
| <i>C-c C-x c</i>  | <code>org-clone-subtree-with-time-shift</code> |
| Clone a subtree by making a number of sibling copies of it. You will be prompted for the number of copies to make, and you can also specify if any timestamps in the entry should be shifted. This can be useful, for example, to create a number of tasks related to a series of lectures to prepare. For more details, see the docstring of the command <code>org-clone-subtree-with-time-shift</code> .  |  |

|                      |  |
|----------------------|--|
| <code>C-c C-w</code> | <code>org-refile</code><br>Refile entry or region to a different location. See Section 9.5 [Refile and copy], page 99.   |
| <code>C-c ^</code>   | <code>org-sort</code><br>Sort same-level entries. When there is an active region, all entries in the region will be sorted. Otherwise the children of the current headline are sorted. The command prompts for the sorting method, which can be alphabetically, numerically, by time (first timestamp with active preferred, creation time, scheduled time, deadline time), by priority, by TODO keyword (in the sequence the keywords have been defined in the setup) or by the value of a property. Reverse sorting is possible as well. You can also supply your own function to extract the sorting key. With a <code>C-u</code> prefix, sorting will be case-sensitive. |
| <code>C-x n s</code> | <code>org-narrow-to-subtree</code><br>Narrow buffer to current subtree.  |
| <code>C-x n b</code> | <code>org-narrow-to-block</code><br>Narrow buffer to current block.  |
| <code>C-x n w</code> | <code>widen</code><br>Widen buffer to remove narrowing.  |
| <code>C-c *</code>   | <code>org-toggle-heading</code><br>Turn a normal line or plain list item into a headline (so that it becomes a subheading at its location). Also turn a headline into a normal line by removing the stars. If there is an active region, turn all lines in the region into headlines. If the first line in the region was an item, turn only the item lines into headlines. Finally, if the first line is a headline, remove the stars from all headlines in the region.   |

When there is an active region (Transient Mark mode), promotion and demotion work on all headlines in the region. To select a region of headlines, it is best to place both point and mark at the beginning of a line, mark at the beginning of the first headline, and point at the line just after the last headline to change. Note that when the cursor is inside a table (see Chapter 3 [Tables], page 19), the Meta-Cursor keys have different functionality.

## 2.6 Sparse trees

An important feature of Org mode is the ability to construct *sparse trees* for selected information in an outline tree, so that the entire document is folded as much as possible, but the selected information is made visible along with the headline structure above it<sup>6</sup>. Just try it out and you will see immediately how it works.

Org mode contains several commands for creating such trees, all these commands can be accessed through a dispatcher:

|                    |   |
|--------------------|---|
| <code>C-c /</code> | <code>org-sparse-tree</code><br>This prompts for an extra key to select a sparse-tree creating command. |
|--------------------|---|

---

<sup>6</sup> See also the variable `org-show-context-detail` to decide how much context is shown around each match.

|  |                             |
|--|-----------------------------|
| <code>C-c / r</code> or <code>C-c / /</code>   | <code>org-occur</code>      |
| Prompts for a regexp and shows a sparse tree with all matches. If the match is in a headline, the headline is made visible. If the match is in the body of an entry, headline and body are made visible. In order to provide minimal context, also the full hierarchy of headlines above the match is shown, as well as the headline following the match. Each match is also highlighted; the highlights disappear when the buffer is changed by an editing command <sup>7</sup> , or by pressing <code>C-c C-c</code> . When called with a <code>C-u</code> prefix argument, previous highlights are kept, so several calls to this command can be stacked. |                             |
| <code>M-g n</code> or <code>M-g M-n</code>   | <code>next-error</code>     |
| Jump to the next sparse tree match in this buffer.   |                             |
| <code>M-g p</code> or <code>M-g M-p</code>   | <code>previous-error</code> |
| Jump to the previous sparse tree match in this buffer.   |                             |

For frequently used sparse trees of specific search strings, you can use the option `org-agenda-custom-commands` to define fast keyboard access to specific sparse trees. These commands will then be accessible through the agenda dispatcher (see Section 10.2 [Agenda dispatcher], page 103). For example:

```
(setq org-agenda-custom-commands
      '(("f" occur-tree "FIXME")))
```

will define the key `C-c a f` as a shortcut for creating a sparse tree matching the string ‘FIXME’.

The other sparse tree commands select headings based on TODO keywords, tags, or properties and will be discussed later in this manual.

To print a sparse tree, you can use the Emacs command `ps-print-buffer-with-faces` which does not print invisible parts of the document. Or you can use `C-c C-e C-v` to export only the visible part of the document and print the resulting file.

## 2.7 Plain lists

Within an entry of the outline tree, hand-formatted lists can provide additional structure. They also provide a way to create lists of checkboxes (see Section 5.6 [Checkboxes], page 56). Org supports editing such lists, and every exporter (see Chapter 12 [Exporting], page 138) can parse and format them.

Org-mode では、数字付きのリスト、順序のないリスト、そして記述リストを解釈します。

- *Unordered* list items start with ‘-’, ‘+’, or ‘\*’<sup>8</sup> as bullets.
- *Ordered* list items start with a numeral followed by either a period or a right parenthesis<sup>9</sup>, such as ‘1.’ or ‘1)’<sup>10</sup>. If you want a list to start with a different value (e.g., 20),

<sup>7</sup> This depends on the option `org-remove-highlights-with-change`

<sup>8</sup> When using ‘\*’ as a bullet, lines must be indented or they will be seen as top-level headlines. Also, when you are hiding leading stars to get a clean outline view, plain list items starting with a star may be hard to distinguish from true headlines. In short: even though ‘\*’ is supported, it may be better to not use it for plain list items.

<sup>9</sup> You can filter out any of them by configuring `org-plain-list-ordered-item-terminator`.

<sup>10</sup> You can also get ‘a.’, ‘A.’, ‘a)’ and ‘A)’ by configuring `org-list-allow-alphabetical`. To minimize confusion with normal text, those are limited to one character only. Beyond that limit, bullets will automatically fallback to numbers.

start the text of the item with `[@20]`<sup>11</sup>. Those constructs can be used in any item of the list in order to enforce a particular numbering.

- 説明 のリストアイテムは順序のないリストアイテムで、説明内容と用語の記述を区別するために ‘`::`’ といった区分するための記号を含んでいます。

Items belonging to the same list must have the same indentation on the first line. In particular, if an ordered list reaches number ‘10.’, then the 2-digit numbers must be written left-aligned with the other numbers in the list. An item ends before the next line that is less or equally indented than its bullet/number.

A list ends whenever every item has ended, which means before any line less or equally indented than items at top level. It also ends before two blank lines. In that case, all items are closed. Here is an example:

```
** Lord of the Rings
  My favorite scenes are (in this order)
  1. The attack of the Rohirrim
  2. Eowyn's fight with the witch king
    + this was already my favorite scene in the book
    + I really like Miranda Otto.
  3. Peter Jackson being shot by Legolas
    - on DVD only
      He makes a really funny face when it happens.
  But in the end, no individual scenes matter but the film as a whole.
  Important actors in this film are:
  - Elijah Wood :: He plays Frodo
  - Sean Astin :: He plays Sam, Frodo's friend. I still remember
    him very well from his role as Mikey Walsh in The Goonies.
```

Org supports these lists by tuning filling and wrapping commands to deal with them correctly, and by exporting them properly (see Chapter 12 [Exporting], page 138). Since indentation is what governs the structure of these lists, many structural constructs like `#+BEGIN_...` blocks can be indented to signal that they belong to a particular item.

If you find that using a different bullet for a sub-list (than that used for the current list-level) improves readability, customize the variable `org-list-demote-modify-bullet`. To get a greater difference of indentation between items and their sub-items, customize `org-list-indent-offset`.

The following commands act on items when the cursor is in the first line of an item (the line with the bullet or number). Some of them imply the application of automatic rules to keep list structure intact. If some of these actions get in your way, configure `org-list-automatic-rules` to disable them individually.

**TAB** **org-cycle**

Items can be folded just like headline levels. Normally this works only if the cursor is on a plain list item. For more details, see the variable `org-cycle-include-plain-lists`. If this variable is set to `integrate`, plain list items will be treated like low-level headlines. The level of an item is then given by

<sup>11</sup> If there's a checkbox in the item, the cookie must be put *before* the checkbox. If you have activated alphabetical lists, you can also use counters like `[@b]`.



the indentation of the bullet/number. Items are always subordinate to real headlines, however; the hierarchies remain completely separated. In a new item with no text yet, the first **TAB** demotes the item to become a child of the previous one. Subsequent **TABs** move the item to meaningful levels in the list and eventually get it back to its initial position.

- M-RET** **org-insert-heading**  
 Insert new item at current level. With a prefix argument, force a new heading (see Section 2.5 [Structure editing], page 9). If this command is used in the middle of an item, that item is *split* in two, and the second part becomes the new item<sup>12</sup>. If this command is executed *before item's body*, the new item is created *before* the current one.
- M-S-RET** Insert a new item with a checkbox (see Section 5.6 [Checkboxes], page 56).
- S-up**  
**S-down** Jump to the previous/next item in the current list<sup>13</sup>, but only if **org-support-shift-select** is off. If not, you can still use paragraph jumping commands like **C-UP** and **C-DOWN** to quite similar effect.
- M-up**  
**M-down** Move the item including subitems up/down<sup>14</sup> (swap with previous/next item of same indentation). If the list is ordered, renumbering is automatic.
- M-left**  
**M-right** 一つのアイテムのインデントを増減します。子のアイテムを残したままで。
- M-S-LEFT**  
**M-S-RIGHT** サブアイテムを含んだまま、アイテムのインデントを増減します。初期設定では、アイテムのツリーはカレントのインデントに基づいて選択されます。直接連続してこれらのコマンドが何度も実行されたときは、たとえ新しいインデントが異なる階層であるとかわかっていても、初期に選択されたリージョンが使用されます。新しい階層を使用するために、カーソルを移動させるコマンドの連鎖をブレイクする必要があります。
- As a special case, using this command on the very first item of a list will move the whole list. This behavior can be disabled by configuring **org-list-automatic-rules**. The global indentation of a list has no influence on the text *after* the list.
- C-c C-c** アイテムの行にチェックボックス (see [\[チェックボックス\]](#), page [\[undefined\]](#)) がある場合には、チェックボックスの状態を切り替えます。ともかく、リスト全体について bullet とインデントの整合性を検証します。
- C-c -** Cycle the entire list level through the different itemize/enumerate bullets ('-', '+', '\*', '1.', '1') or a subset of them, depending on **org-plain-list-ordered-item-terminator**, the type of list, and its indentation. With a numeric prefix argument N, select the Nth bullet from this list. If there is an active region

<sup>12</sup> If you do not want the item to be split, customize the variable **org-M-RET-may-split-line**.

<sup>13</sup> If you want to cycle around items that way, you may customize **org-list-use-circular-motion**.

<sup>14</sup> See **org-list-use-circular-motion** for a cyclic behavior.

when calling this, all selected lines are converted to list items. With a prefix argument, selected text is changed into a single item. If the first line already was a list item, any item marker will be removed from the list. Finally, even without an active region, a normal line will be converted into a list item.

**C-c \*** Turn a plain list item into a headline (so that it becomes a subheading at its location). See Section 2.5 [Structure editing], page 9, for a detailed explanation.

**C-c C-\*** Turn the whole plain list into a subtree of the current heading. Checkboxes (see Section 5.6 [Checkboxes], page 56) will become TODO (resp. DONE) keywords when unchecked (resp. checked).

#### **S-LEFT/RIGHT**

bullet 上、またはアイテムの行のどこかにカーソルが置かれているときに、このコマンドは、また、bullet のスタイルを切り替えます。詳細は **org-support-shift-select** に依存します。

**C-c ^** Sort the plain list. You will be prompted for the sorting method: numerically, alphabetically, by time, by checked status for check lists, or by a custom function.

## 2.8 Drawers

Sometimes you want to keep information associated with an entry, but you normally don't want to see it. For this, Org mode has *drawers*. They can contain anything but a headline and another drawer. Drawers look like this:

```
** これは見出しです
    ここはまだ引き出しの外側です
    :DRAWERNAME:
    これは引き出しの内側です。
    :END:
    引き出しの後です。
```

You can interactively insert drawers at point by calling **org-insert-drawer**, which is bound to **C-c C-x d**. With an active region, this command will put the region inside the drawer. With a prefix argument, this command calls **org-insert-property-drawer** and add a property drawer right below the current headline. Completion over drawer keywords is also possible using **M-TAB**<sup>15</sup>.

Visibility cycling (see Section 2.3 [Visibility cycling], page 6) on the headline will hide and show the entry, but keep the drawer collapsed to a single line. In order to look inside the drawer, you need to move the cursor to the drawer line and press **TAB** there. Org mode uses the **PROPERTIES** drawer for storing properties (see Chapter 7 [Properties and columns], page 64), and you can also arrange for state change notes (see Section 5.3.2 [Tracking TODO state changes], page 52) and clock times (see Section 8.4 [Clocking work time], page 81) to be stored in a drawer **LOGBOOK**. If you want to store a quick note in the **LOGBOOK** drawer, in a similar way to state changes, use

<sup>15</sup> Many desktops intercept **M-TAB** to switch windows. Use **C-M-i** or **ESC TAB** instead for completion (see Section 15.1 [Completion], page 229).

**C-c C-z** Add a time-stamped note to the LOGBOOK drawer.

You can select the name of the drawers which should be exported with `org-export-with-drawers`. In that case, drawer contents will appear in export output. Property drawers are not affected by this variable: configure `org-export-with-properties` instead.

## 2.9 Blocks

Org mode uses begin...end blocks for various purposes from including source code examples (see Section 11.5 [Literal examples], page 132) to capturing time logging information (see Section 8.4 [Clocking work time], page 81). These blocks can be folded and unfolded by pressing TAB in the begin line. You can also get all blocks folded at startup by configuring the option `org-hide-block-startup` or on a per-file basis by using

```
#+STARTUP: hideblocks
#+STARTUP: nohideblocks
```

## 2.10 Footnotes

Org mode supports the creation of footnotes.

A footnote is started by a footnote marker in square brackets in column 0, no indentation allowed. It ends at the next footnote definition, headline, or after two consecutive empty lines. The footnote reference is simply the marker in square brackets, inside text. Markers always start with `fn:`. For example:

```
The Org homepage[fn:1] now looks a lot better than it used to.
...
[fn:1] The link is: https://orgmode.org
```

Org mode extends the number-based syntax to *named* footnotes and optional inline definition. Here are the valid references:

`[fn:name]`

A named footnote reference, where `name` is a unique label word, or, for simplicity of automatic creation, a number.

`[fn::This is the inline definition of this footnote]`

参照のポイントに直接定義がなされる L<sup>A</sup>T<sub>E</sub>X のような無記名の脚注。

`[fn:name:a definition]`

脚注のインラインでの定義、それはまた、ノートのための名前を明確に規定します。Org-mode は同じノートに対して多重の参照を許容するので、新たな参照を作成するために、`[fn:name]`を使用することができます。

脚注のラベルは自動的に作成することができます。そうしないならば、あなた自身で名前を作成することができます。これは `org-footnote-auto-label` 変数で操作され、`#+STARTUP` キーワードに対応します。詳細については変数の説明を参照してください。

以下のコマンドはプロパティを操作する助けとなります。

**C-c C-x f** 脚注の動作のコマンド。

カーソルが脚注参照上にあるときに、定義部分にジャンプします。カーソルが定義部分にあるときに、(最初の) 参照されている部分にジャンプします。

Otherwise, create a new footnote. Depending on the option `org-footnote-define-inline`<sup>16</sup>, the definition will be placed right into the text as part of the reference, or separately into the location determined by the option `org-footnote-section`.

前置引数と一緒にコマンドが呼び出された場合は、追加のオプションのメニューが提示されます。

- s** Sort the footnote definitions by reference sequence. During editing, Org makes no effort to sort footnote definitions into a particular sequence. If you want them sorted, use this command, which will also move entries according to `org-footnote-section`. Automatic sorting after each insertion/deletion can be configured using the option `org-footnote-auto-adjust`.
- r** Renumber the simple `fn:N` footnotes. Automatic renumbering after each insertion/deletion can be configured using the option `org-footnote-auto-adjust`.
- S** Short for first **r**, then **s** action.
- n** Normalize the footnotes by collecting all definitions (including inline definitions) into a special section, and then numbering them in sequence. The references will then also be numbers.
- d** Delete the footnote at point, and all definitions of and references to it.

`org-footnote-auto-adjust`<sup>17</sup> と `nofnadjust` の変数に依存し、それぞれの挿入と削除のあとに、番号の振り直しと脚注の並び替えが自動的に行われます。

**C-c C-c** If the cursor is on a footnote reference, jump to the definition. If it is a the definition, jump back to the reference. When called at a footnote location with a prefix argument, offer the same menu as **C-c C-x f**.

**C-c C-o** or `mouse-1/2`

Footnote labels are also links to the corresponding definition/reference, and you can use the usual commands to follow these links.

**C-c '**

**C-c '** Edit the footnote definition corresponding to the reference at point in a separate window. The window can be closed by pressing **C-c '**.

## 2.11 The Orgstruct minor mode

If you like the intuitive way the Org mode structure editing and list formatting works, you might want to use these commands in other modes like Text mode or Mail mode as well. The minor mode `orgstruct-mode` makes this possible. Toggle the mode with `M-x orgstruct-mode RET`, or turn it on by default, for example in Message mode, with one of:

```
(add-hook 'message-mode-hook 'turn-on-orgstruct)
(add-hook 'message-mode-hook 'turn-on-orgstruct++)
```

<sup>16</sup> The corresponding in-buffer setting is: `#+STARTUP: fninline` or `#+STARTUP: nofninline`

<sup>17</sup> 対応するインバッファのオプションは `fnadjust` と `nofnadjust` になります。

When this mode is active and the cursor is on a line that looks to Org like a headline or the first line of a list item, most structure editing commands will work, even if the same keys normally have different functionality in the major mode you are using. If the cursor is not in one of those special lines, Orgstruct mode lurks silently in the shadows.

When you use `orgstruct++-mode`, Org will also export indentation and autofill settings into that mode, and detect item context after the first line of an item.

You can also use Org structure editing to fold and unfold headlines in *any* file, provided you defined `orgstruct-heading-prefix-regexp`: the regular expression must match the local prefix to use before Org's headlines. For example, if you set this variable to `";; "` in Emacs Lisp files, you will be able to fold and unfold headlines in Emacs Lisp commented lines. Some commands like `org-demote` are disabled when the prefix is set, but folding/unfolding will work correctly.

## 2.12 Org syntax

A reference document providing a formal description of Org's syntax is available as a draft on Worg (<https://orgmode.org/worg/dev/org-syntax.html>), written and maintained by Nicolas Goaziou. It defines Org's core internal concepts such as **headlines**, **sections**, **affiliated keywords**, **(greater) elements** and **objects**. Each part of an Org file falls into one of the categories above.

To explore the abstract structure of an Org buffer, run this in a buffer:

```
M-: (org-element-parse-buffer) RET
```

It will output a list containing the buffer's content represented as an abstract structure. The export engine relies on the information stored in this list. Most interactive commands (e.g., for structure editing) also rely on the syntactic meaning of the surrounding context.

You can check syntax in your documents using `org-lint` command.

## 3 Tables

Org comes with a fast and intuitive table editor. Spreadsheet-like calculations are supported using the Emacs `calc` package (see *GNU Emacs Calculator Manual*).

### 3.1 組み込まれたテーブルエディタ

Org makes it easy to format tables in plain ASCII. Any line with ‘|’ as the first non-whitespace character is considered part of a table. ‘|’ is also the column separator<sup>1</sup>. A table might look like this:

```
| Name | Phone | Age |
|-----+-----+-----|
| Peter | 1234 | 17 |
| Anna | 4321 | 25 |
```

テーブルの中で、TABやRETもしくはC-c C-cを押す度に、テーブルの表示が自動的に更新されます。TABを押すとカーソルが次のフィールドに移ります (RETの場合は次の行へ)。また、テーブルの端もしくは水平ラインの直前の行で押せば、テーブルに新しい行が追加されます。テーブルのインデントは一行目目で決定されます。‘|-’で開始するどんな行も水平ラインとして解釈され、次にテーブルが更新される時に、水平ラインはテーブル幅いっぱいに拡張されます。つまり、上記のテーブルを作成するためには、次のように入力するだけでよかったのです。

```
|Name|Phone|Age|
|-
```

続いてTABを押してテーブルを整列し、フィールドへの入力を始めます。さらに素早いテーブルの作成方法は、|Name|Phone|Ageに続いてC-c RETと入力することです。

When typing text into a field, Org treats DEL, Backspace, and all character keys in a special way, so that inserting and deleting avoids shifting other fields. Also, when typing *immediately after the cursor was moved into a new field with TAB, S-TAB or RET*, the field is automatically made blank. If this behavior is too unpredictable for you, configure the option `org-table-auto-blank-field`.

#### Creation and conversion

C-c | org-table-create-or-convert-from-region

Convert the active region to a table. If every line contains at least one TAB character, the function assumes that the material is tab separated. If every line contains a comma, comma-separated values (CSV) are assumed. If not, lines are split at whitespace into fields. You can use a prefix argument to force a specific separator: C-u forces CSV, C-u C-u forces TAB, C-u C-u C-u will prompt for a regular expression to match the separator, and a numeric argument N indicates that at least N consecutive spaces, or alternatively a TAB will be the separator. If there is no active region, this command creates an empty Org table. But it is easier just to start typing, like |Name|Phone|Age RET |- TAB.

#### Re-aligning and field motion

C-c C-c org-table-align

Re-align the table and don't move to another field.

<sup>1</sup> To insert a vertical bar into a table field, use `\vert` or, inside a word `abc\vert{}def`.

|   |   |
|---|---|
| <i>C-c SPC</i>  | <code>org-table-blank-field</code>        |
| Blank the field at point.   |   |
| <i>TAB</i>  | <code>org-table-next-field</code>         |
| Re-align the table, move to the next field. Creates a new row if necessary.   |   |
| <i>S-TAB</i>  | <code>org-table-previous-field</code>     |
| Re-align, move to previous field.   |   |
| <i>RET</i>  | <code>org-table-next-row</code>           |
| Re-align the table and move down to next row. Creates a new row if necessary. At the beginning or end of a line, <i>RET</i> still does <i>NEWLINE</i> , so it can be used to split a table.   |   |
| <i>M-a</i>  | <code>org-table-beginning-of-field</code> |
| Move to beginning of the current table field, or on to the previous field.  |   |
| <i>M-e</i>  | <code>org-table-end-of-field</code>       |
| Move to end of the current table field, or on to the next field.  |   |
| <b>Column and row editing</b>   |   |
| <i>M-LEFT</i>   | <code>org-table-move-column-left</code>   |
| <i>M-RIGHT</i>  | <code>org-table-move-column-right</code>  |
| Move the current column left/right.   |   |
| <i>M-S-LEFT</i>   | <code>org-table-delete-column</code>      |
| Kill the current column.  |   |
| <i>M-S-RIGHT</i>  | <code>org-table-insert-column</code>      |
| Insert a new column to the left of the cursor position.   |   |
| <i>M-UP</i>   | <code>org-table-move-row-up</code>        |
| <i>M-DOWN</i>   | <code>org-table-move-row-down</code>      |
| Move the current row up/down.   |   |
| <i>M-S-UP</i>   | <code>org-table-kill-row</code>           |
| Kill the current row or horizontal line.  |   |
| <i>M-S-DOWN</i>   | <code>org-table-insert-row</code>         |
| Insert a new row above the current row. With a prefix argument, the line is created below the current one.  |   |
| <i>C-c -</i>  | <code>org-table-insert-hline</code>       |
| Insert a horizontal line below current row. With a prefix argument, the line is created above the current line.   |   |
| <i>C-c RET</i>  | <code>org-table-hline-and-move</code>     |
| Insert a horizontal line below current row, and move the cursor into the row below that line.   |   |
| <i>C-c ^</i>  | <code>org-table-sort-lines</code>         |
| Sort the table lines in the region. The position of point indicates the column to be used for sorting, and the range of lines is the range between the nearest horizontal separator lines, or the entire table. If point is before the first column, you will be prompted for the sorting column. If there is an active region, the |   |

mark specifies the first line and the sorting column, while point should be in the last line to be included into the sorting. The command prompts for the sorting type (alphabetically, numerically, or by time). You can sort in normal or reverse order. You can also supply your own key extraction and comparison functions. When called with a prefix argument, alphabetic sorting will be case-sensitive.

リージョン

**C-c C-x M-w** **org-table-copy-region**

テーブルの矩形領域を特別なクリップボードにコピーします。ポイントとマークは、矩形領域を構成する末端となるフィールドを決定します。アクティブリージョンがなければ、カーソル位置のフィールドだけをコピーします。この処理は、テーブルの水平ラインを無視します。

**C-c C-x C-w** **org-table-cut-region**

テーブルの矩形領域を特別なクリップボードにコピーし、領域内の全てのフィールドを空にします。つまりこれは「カット」操作です。

**C-c C-x C-y** **org-table-paste-rectangle**

テーブルに矩形領域をペーストします。領域の左上がカーソル位置のフィールドに上書きされます。ペーストする領域に重なるすべてのフィールドは上書きされます。対象とするテーブルに矩形領域が合わないならば、必要に応じてテーブルは拡張されます。この処理は、テーブルの水平ラインを無視します。

**M-RET** **org-table-wrap-region**

カーソル位置でフィールドの文字列を分割し、カーソル以降を一つ下のフィールドの文頭に移動します。アクティブリージョンが存在し、またポイントとマークの両方が同じ列にあるとき、列に含まれるテキストは、与えられた行数を最小化するように改行されます（訳注：余計な空白が消される）。プレフィックスで指定する整数値は、希望する行数に合わせるために使われます（訳注：M行をN行に圧縮できます）。もし選択領域がない状態でプレフィックスを指定すると、カーソル位置のフィールドは空白になり、元々あった文字列は一つ上のフィールドの文末に付け加えられます。

計算機能

**C-c +** **org-table-sum**

カーソル位置の列、もしくは、アクティブリージョンで定められた矩形領域に含まれる数値を合計します。計算結果はエコー領域に表示され、**C-y**で挿入できます。

**S-RET** **org-table-copy-down**

When current field is empty, copy from first non-empty field above. When not empty, copy current field down to next row and move cursor along with it. Depending on the option **org-table-copy-increment**, integer field values will be incremented during copy. Integers that are too large will not be incremented. Also, a 0 prefix argument temporarily disables the increment. This key is also used by shift-selection and related modes (see Section 15.10.2 [Conflicts], page 239).

Miscellaneous

**C-c `** **org-table-edit-field**

Edit the current field in a separate window. This is useful for fields that are not fully visible (see Section 3.2 [Column width and alignment], page 22). When



called with a `C-u` prefix, just make the full field visible, so that it can be edited in place. When called with two `C-u` prefixes, make the editor window follow the cursor through the table and always show the current field. The follow mode exits automatically when the cursor leaves the table, or when you repeat this command with `C-u C-u C-c ``.

#### `M-x org-table-import RET`

ファイルをテーブルとしてインポートします。テーブルは、タブもしくは空白で区切られている必要があります。この機能は、例えば、スプレッドシートのテーブルやデータベースの情報をインポートするために利用します。というのも、一般的にこれらのプログラムが、タブ区切りのテキストフィールドを書き出せるためです。このコマンドは、ファイルの内容をバッファに挿入することで処理が始まり、その領域をテーブルに変換します。コンバータに与えられるすべてのプレフィックスは、セパレータを決定するために利用されます。

`C-c |`

#### `org-table-create-or-convert-from-region`

org-mode のバッファにテーブル状のテキスト（訳注：文字列がタブで区切られているテキスト領域など）をペーストすることでも org-mode のテーブルを作成できます。`C-x C-x`でバッファにペーストされているテキストを選択して、`C-c |`で org-mode のテーブルに変換します（前述の *Creation and conversion* を参照してください）。

#### `M-x org-table-export RET`

Export the table, by default as a TAB-separated file. Use for data exchange with, for example, spreadsheet or database programs. The format used to export the file can be configured in the option `org-table-export-default-format`. You may also use properties `TABLE_EXPORT_FILE` and `TABLE_EXPORT_FORMAT` to specify the file name and the format for table export in a subtree. Org supports quite general formats for exported tables. The exporter format is the same as the format used by Orgtbl radio tables, see Section A.6.3 [Translator functions], page 247, for a detailed description.

## 3.2 Column width and alignment

テーブルの各列の幅はテーブルエディタによって自動的に決定されます。また、列の配置も自動的に決定されます。具体的には、列の中で数値でないフィールド数に対する、数値（と解釈できる）フィールド数の割合に応じて決まります。

Sometimes a single field or a few fields need to carry more text, leading to inconveniently wide columns. Or maybe you want to make a table with several columns having a fixed width, regardless of content. To set the width of a column, one field anywhere in the column may contain just the string ‘<N>’ where ‘N’ is an integer specifying the width of the column in characters. The next re-align will then set the width of this column to this value.

|                                  |       |            |
|----------------------------------|-------|------------|
| ---+-----                        |       | ---+-----  |
|                                  |       | <6>        |
| 1   one                          |       | 1   one    |
| 2   two                          | ----\ | 2   two    |
| 3   This is a long chunk of text | ----/ | 3   This=> |
| 4   four                         |       | 4   four   |
| ---+-----                        |       | ---+-----  |

Fields that are wider become clipped and end in the string ‘=>’. Note that the full text is still in the buffer but is hidden. To see the full text, hold the mouse over the field—a tool-tip window will show the full content. To edit such a field, use the command `C-c `` (that is `C-c` followed by the grave accent). This will open a new window with the full field. Edit it and finish with `C-c C-c`.

幅が狭められた列を持つテーブルが記述されているファイルを訪問する段階では、まだ文字列の隠蔽は実行されていません。見た目を良くするにはテーブルを整列する必要があります。オプションの `org-startup-align-all-tables` を設定すると、ファイルを訪問するときにファイルに記述された全てのテーブルが自動的に整列されます。ただしスタートアップが少し遅くなります。以下の設定を利用すれば、ファイルごとにこのオプションを設定することもできます。

```
#+STARTUP: align
#+STARTUP: noalign
```

If you would like to overrule the automatic alignment of number-rich columns to the right and of string-rich columns to the left, you can use ‘<r>’, ‘<c>’<sup>2</sup> or ‘<l>’ in a similar fashion. You may also combine alignment and field width like this: ‘<r10>’.

書式の設定情報のみを含む行は、ドキュメントをエクスポートするときに自動的に削除されます。

### 3.3 Column groups

When Org exports tables, it does so by default without vertical lines because that is visually more satisfying in general. Occasionally however, vertical lines can be useful to structure a table into groups of columns, much like horizontal lines can do for groups of rows. In order to specify column groups, you can use a special row where the first field contains only ‘/’. The further fields can either contain ‘<’ to indicate that this column should start a group, ‘>’ to indicate the end of a group, or ‘<>’ (no space between ‘<’ and ‘>’) to make a column a group of its own. Boundaries between column groups will upon export be marked with vertical lines. Here is an example:

| N | N <sup>2</sup> | N <sup>3</sup> | N <sup>4</sup> | ~sqrt(n)~ | ~sqrt[4](N)~ |
|---|----------------|----------------|----------------|-----------|--------------|
| / | <              |                | >              | <         | >            |
| 1 | 1              | 1              | 1              | 1         | 1            |
| 2 | 4              | 8              | 16             | 1.4142    | 1.1892       |
| 3 | 9              | 27             | 81             | 1.7321    | 1.3161       |

```
#+TBLFM: $2=$1^2::$3=$1^3::$4=$1^4::$5=sqrt($1)::6=sqrt(sqrt(($1)))
```

表示させたいすべての垂直ラインの後ろに、列のグループ開始を指定するだけでも十分です。

| N | N <sup>2</sup> | N <sup>3</sup> | N <sup>4</sup> | sqrt(n) | sqrt[4](N) |
|---|----------------|----------------|----------------|---------|------------|
| / | <              |                |                | <       |            |

### 3.4 Orgtbl マイナーモード

If you like the intuitive way the Org table editor works, you might also want to use it in other modes like Text mode or Mail mode. The minor mode Orgtbl mode makes this

<sup>2</sup> Centering does not work inside Emacs, but it does have an effect when exporting to HTML.

possible. You can always toggle the mode with `M-x orgtbl-mode RET`. To turn it on by default, for example in Message mode, use

```
(add-hook 'message-mode-hook 'turn-on-orgtbl)
```

さらに、いくつかの特別な手順を追加することで、orgtbl モードの任意のシンタックスを使ってテーブルをメンテナンスできます。たとえば、orgtbl モードの簡単さと機能を用いて  $\text{\LaTeX}$  のテーブルを構築できます。これは表計算機能も含んでいます。さらなる詳細は、[\[任意の構文によるテーブル\]](#), page [\[undefined\]](#). を参照してください。

## 3.5 The spreadsheet

org-mode のテーブルエディタは、表計算機能を実装するために Emacs の `calc` を利用します。異なるフィールドの値を使って様々なフィールドの値を算出するために、Emacs Lisp の書式も評価できます。十分な表計算機能が備わっていますが、org-mode の実装が他の表計算ソフトと全く同等というわけではありません。たとえば、org-mode は列の数式を概念を理解しています。これは、関連する各フィールドに数式をコピーすることなく、ヘッダーを除いた列にある全てのフィールドに適用されます。数式のデバッグもあります。また、数式に含まれる参照に対応するフィールドを、テーブル内でハイライトする機能と、それらの参照に矢印キーで移動する機能を持つ数式エディタがあります。

### 3.5.1 References

テーブル内部のフィールドを他のフィールドの値を使って計算するに、数式は、他のフィールドか範囲を必ず参照していなければなりません。org-mode では、名前、絶対的または相対的な位置によってフィールドを参照することができます。フィールドの位置がどこかを特定するためには、そのフィールドで `C-c ?` を押してください。もしくは、グリッド表示をトグルするために `C-c }` を使用してください。

#### フィールドの参照

数式は別フィールドの値を 2 つの方法で参照できます。他の表計算ソフトと同じように、B3 のような文字と数値の組み合わせでフィールドを参照できます。B3 は 3 行目の第 2 フィールドを意味しています。However, Org prefers<sup>3</sup> to use another, more general representation that looks like this:

`@row$column`

Column specifications can be absolute like `$1`, `$2,...$N`, or relative to the current column (i.e., the column of the field which is being computed) like `+$1` or `-$2`. `$<` and `$>` are immutable references to the first and last column, respectively, and you can use `$>>>` to indicate the third column from the right.

The row specification only counts data lines and ignores horizontal separator lines (hlines). Like with columns, you can use absolute row numbers `@1`, `@2,...@N`, and row numbers relative to the current row like `@+3` or `@-1`. `@<` and `@>` are immutable references the first and last<sup>4</sup> row in the table, respectively. You may also specify the row relative to one of the hlines: `@I` refers to the first hline, `@II` to the second, etc. `@-I` refers to the first

<sup>3</sup> Org will understand references typed by the user as 'B4', but it will not use this syntax when offering a formula for editing. You can customize this behavior using the option `org-table-use-standard-references`.

<sup>4</sup> For backward compatibility you can also use special names like `$LR5` and `$LR12` to refer in a stable way to the 5th and 12th field in the last row of the table. However, this syntax is deprecated, it should not be used for new documents. Use `@>$` instead.

such line above the current line, @+I to the first such line below the current line. You can also write @III+2 which is the second data line after the third hline in the table.

@0 and \$0 refer to the current row and column, respectively, i.e., to the row/column for the field being computed. Also, if you omit either the column or the row part of the reference, the current row/column is implied.

org-mode の符号無し数値を使った参照は、静的な参照です。つまり、二つの異なるフィールドにある数式の中で同じ参照を利用すれば、常に同じフィールドが参照されます。符号付き数値を使った参照は、動的な参照です。これは、見た目上は同じ参照であっても、数式で計算されるフィールドに依存して、異なるフィールドを参照できるためです。

いくつかの例を示します。

|         |  |
|---------|--|
| @2\$3   | 2nd row, 3rd column (same as C2)                   |
| \$5     | column 5 in the current row (same as E&)           |
| @2      | current column, row 2                              |
| @-1\$-3 | the field one row up, three columns to the left    |
| @-I\$2  | field just under hline above current row, column 2 |
| @>\$5   | field in the last row, in column 5                 |

#### 範囲参照

複数のフィールドを集めた矩形の範囲を参照できます。この範囲は、二つの参照を二つのドット‘.’で接続して指定します。二つの参照が共にカーソル位置の行にあるとき、単純に‘\$2..\$7’と指定できます。しかし、一方のフィールドが異なる列にあるときは、少なくとも最初に指定するフィールドについて@row\$columnのような省略しない形式を使う必要があります（つまり、正しく解釈させるために参照を‘@’で始めなければなりません）。具体例は次のようになります。

|              |  |
|--------------|--|
| \$1..\$3     | first three fields in the current row                          |
| \$P..\$Q     | range, using column names (see under Advanced)                 |
| \$<<<..\$>>  | start in third column, continue to the last but one            |
| @2\$1..@4\$3 | 6 fields between these two fields (same as A2..C4)             |
| @-1\$-2..@-1 | 3 fields in the row above, starting from 2 columns on the left |
| @I..II       | between first and second hline, short for @I..@II              |

Range references return a vector of values that can be fed into Calc vector functions. Empty fields in ranges are normally suppressed, so that the vector contains only the non-empty fields. For other options with the mode switches ‘E’, ‘N’ and examples see Section 3.5.2 [Formula syntax for Calc], page 26.

#### 数式中のフィールドの座標

One of the very first actions during evaluation of Calc formulas and Lisp formulas is to substitute @# and \$# in the formula with the row or column number of the field where the current result will go to. The traditional Lisp formula equivalents are `org-table-current-dline` and `org-table-current-column`. Examples:

```
if(@# % 2, $#, string(""))
```

Insert column number on odd rows, set field to empty on even rows.

```
$2 = '(identity remote(F00, @@#$1))
```

Copy text or values of each row of column 1 of the table named F00 into column 2 of the current table.

```
@3 = 2 * remote(F00, @1$$#)
```

Insert the doubled value of each column of row 1 of the table named F00 into row 3 of the current table.

For the second/third example, the table named F00 must have at least as many rows/columns as the current table. Note that this is inefficient<sup>5</sup> for large number of rows/columns.

#### 名前付き参照

‘\$name’ is interpreted as the name of a column, parameter or constant. Constants are defined globally through the option `org-table-formula-constants`, and locally (for the file) through a line like

```
#+CONSTANTS: c=299792458. pi=3.14 eps=2.4e-6
```

Also properties (see Chapter 7 [Properties and columns], page 64) can be used as constants in table formulas: for a property ‘:Xyz:’ use the name ‘\$PROP\_Xyz’, and the property will be searched in the current outline entry and in the hierarchy above it. If you have the `constants.el` package, it will also be used to resolve constants, including natural constants like ‘\$h’ for Planck’s constant, and units like ‘\$km’ for kilometers<sup>6</sup>. Column names and parameters can be specified in special table lines. These are described below, see Section 3.5.10 [Advanced features], page 34. All names must start with a letter, and further consist of letters and numbers.

#### リモート参照

別テーブルの定数、フィールドそして範囲を参照できます。現在のファイルでも、別ファイルにある場合でも参照できます。シンタックスは、

```
remote(NAME-OR-ID, REF)
```

where NAME can be the name of a table in the current file as set by a `#+NAME: Name` line before the table. It can also be the ID of an entry, even in a different file, and the reference then refers to the first table in that entry. REF is an absolute field or range reference as described above for example `@3$3` or `$somename`, valid in the referenced table.

Indirection of NAME-OR-ID: When NAME-OR-ID has the format `@ROW$COLUMN` it will be substituted with the name or ID found in this field of the current table. For example `remote($1, @>$2) => remote(year_2013, @>$1)`. The format `B3` is not supported because it can not be distinguished from a plain table name or ID.

### 3.5.2 Formula syntax for Calc

A formula can be any algebraic expression understood by the Emacs Calc package. Note that `calc` has the non-standard convention that ‘/’ has lower precedence than ‘\*’, so that ‘a/b\*c’ is interpreted as ‘a/(b\*c)’. Before evaluation by `calc-eval` (see Section “Calling Calc from Your Lisp Programs” in *GNU Emacs Calc Manual*), variable substitution takes place according to the rules described above. 範囲指定するベクトルは‘vmean’や‘vsum’のような Calc のベクトル関数に直接渡されます。

<sup>5</sup> The computation time scales as  $O(N^2)$  because the table named F00 is parsed for each field to be read.

<sup>6</sup> `constants.el` can supply the values of constants in two different unit systems, `SI` and `cgs`. Which one is used depends on the value of the variable `constants-unit-system`. You can use the `#+STARTUP` options `constSI` and `constcgs` to set this value for the current buffer.

A formula can contain an optional mode string after a semicolon. This string consists of flags to influence Calc and other modes during execution. By default, Org uses the standard Calc modes (precision 12, angular units degrees, fraction and symbolic modes off). The display format, however, has been changed to (`float 8`) to keep tables compact. The default settings can be configured using the option `org-calc-default-modes`.

List of modes:

|                             |   |
|-----------------------------|---|
| <code>p20</code>            | Set the internal Calc calculation precision to 20 digits.   |
| <code>n3, s3, e2, f4</code> | Normal, scientific, engineering or fixed format of the result of Calc passed back to Org. Calc formatting is unlimited in precision as long as the Calc calculation precision is greater.   |
| <code>D, R</code>           | Degree and radian angle modes of Calc.  |
| <code>F, S</code>           | Fraction and symbolic modes of Calc.  |
| <code>T, t, U</code>        | Duration computations in Calc or Lisp, see Section 3.5.4 [Durations and time values], page 29.  |
| <code>E</code>              | If and how to consider empty fields. Without ‘E’ empty fields in range references are suppressed so that the Calc vector or Lisp list contains only the non-empty fields. With ‘E’ the empty fields are kept. For empty fields in ranges or empty field references the value ‘nan’ (not a number) is used in Calc formulas and the empty string is used for Lisp formulas. Add ‘N’ to use 0 instead for both formula types. For the value of a field the mode ‘N’ has higher precedence than ‘E’. |
| <code>N</code>              | Interpret all fields as numbers, use 0 for non-numbers. See the next section to see how this is essential for computations with Lisp formulas. In Calc formulas it is used only occasionally because there number strings are already interpreted as numbers without ‘N’.   |
| <code>L</code>              | Literal, for Lisp formulas only. See the next section.  |

Unless you use large integer numbers or high-precision-calculation and `-display` for floating point numbers you may alternatively provide a ‘`printf`’ format specifier to reformat the Calc result after it has been passed back to Org instead of letting Calc already do the formatting<sup>7</sup>. A few examples:

|                                |  |
|--------------------------------|--|
| <code>\$1+\$2</code>           | Sum of first and second field                        |
| <code>\$1+\$2;%.2f</code>      | Same, format result to two decimals                  |
| <code>exp(\$2)+exp(\$1)</code> | Math functions can be used                           |
| <code>\$0;%.1f</code>          | Reformat current cell to 1 decimal                   |
| <code>(\$3-32)*5/9</code>      | Degrees F -> C conversion                            |
| <code>\$c/\$1/\$cm</code>      | Hz -> cm conversion, using <code>constants.el</code> |
| <code>tan(\$1);Dp3s1</code>    | Compute in degrees, precision 3, display SCI 1       |
| <code>sin(\$1);Dp3%.1e</code>  | Same, but use printf specifier for display           |

<sup>7</sup> The ‘`printf`’ reformatting is limited in precision because the value passed to it is converted into an ‘`integer`’ or ‘`double`’. The ‘`integer`’ is limited in size by truncating the signed value to 32 bits. The ‘`double`’ is limited in precision to 64 bits overall which leaves approximately 16 significant decimal digits.

`taylor($3,x=7,2)`      Taylor series of \$3, at x=7, second degree

Calc also contains a complete set of logical operations, (see Section “Logical Operations” in *GNU Emacs Calc Manual*). For example

`if($1 < 20, teen, string(""))`

“teen” if age \$1 is less than 20, else the Org table result field is set to empty with the empty string.

`if("$1" == "nan" || "$2" == "nan", string(""), $1 + $2); E f-1`

Sum of the first two columns. When at least one of the input fields is empty the Org table result field is set to empty. ‘E’ is required to not convert empty fields to 0. ‘f-1’ is an optional Calc format string similar to ‘%.1f’ but leaves empty results empty.

`if(typeof(vmean($1..$7)) == 12, string(""), vmean($1..$7); E`

Mean value of a range unless there is any empty field. Every field in the range that is empty is replaced by ‘nan’ which lets ‘vmean’ result in ‘nan’. Then ‘typeof == 12’ detects the ‘nan’ from ‘vmean’ and the Org table result field is set to empty. Use this when the sample set is expected to never have missing values.

`if("$1..$7" == "[]", string(""), vmean($1..$7))`

Mean value of a range with empty fields skipped. Every field in the range that is empty is skipped. When all fields in the range are empty the mean value is not defined and the Org table result field is set to empty. Use this when the sample set can have a variable size.

`vmean($1..$7); EN`

To complete the example before: Mean value of a range with empty fields counting as samples with value 0. Use this only when incomplete sample sets should be padded with 0 to the full size.

You can add your own Calc functions defined in Emacs Lisp with `defmath` and use them in formula syntax for Calc.

### 3.5.3 数式としての Emacs Lisp 形式

It is also possible to write a formula in Emacs Lisp. This can be useful for string manipulation and control structures, if Calc’s functionality is not enough.

If a formula starts with an apostrophe followed by an opening parenthesis, then it is evaluated as a Lisp form. The evaluation should return either a string or a number. Just as with `calc` formulas, you can specify modes and a `printf` format after a semicolon.

With Emacs Lisp forms, you need to be conscious about the way field references are interpolated into the form. By default, a reference will be interpolated as a Lisp string (in double-quotes) containing the field. If you provide the ‘N’ mode switch, all referenced elements will be numbers (non-number fields will be zero) and interpolated as Lisp numbers, without quotes. If you provide the ‘L’ flag, all fields will be interpolated literally, without quotes. I.e., if you want a reference to be interpreted as a string by the Lisp form, enclose the reference operator itself in double-quotes, like “\$3”. Ranges are inserted as space-separated fields, so you can embed them in list or vector syntax.

Here are a few examples—note how the ‘N’ mode is used when we do computations in Lisp:

```
'(concat (substring $1 1 2) (substring $1 0 1) (substring $1 2))
    Swap the first two characters of the content of column 1.

'(+ $1 $2);N
    Add columns 1 and 2, equivalent to Calc's $1+$2.

'(apply '+ '($1..$4));N
    Compute the sum of columns 1 to 4, like Calc's vsum($1..$4).
```

### 3.5.4 Durations and time values

If you want to compute time values use the T, t, or U flag, either in Calc formulas or Elisp formulas:

| Task 1  | Task 2   | Total    |
|---------|----------|----------|
| 2:12    | 1:47     | 03:59:00 |
| 2:12    | 1:47     | 03:59    |
| 3:02:20 | -2:07:00 | 0.92     |

```
#+TBLFM: @2$3=$1+$2;T::@3$3=$1+$2;U::@4$3=$1+$2;t
```

Input duration values must be of the form HH:MM[:SS], where seconds are optional. With the T flag, computed durations will be displayed as HH:MM:SS (see the first formula above). With the U flag, seconds will be omitted so that the result will be only HH:MM (see second formula above). Zero-padding of the hours field will depend upon the value of the variable `org-table-duration-hour-zero-padding`.

With the t flag, computed durations will be displayed according to the value of the option `org-table-duration-custom-format`, which defaults to 'hours and will display the result as a fraction of hours (see the third formula in the example above).

Negative duration values can be manipulated as well, and integers will be considered as seconds in addition and subtraction.

### 3.5.5 Field and range formulas

特定のフィールドに数式を割り当てるためには、‘:=’に続けて、直接フィールドに書き込みます。たとえば、‘:=vsum(@II..III)’のようにします。カーソルがフィールドにある状態で、TABやRET、もしくはC-c C-cを押すと、入力した数式はそのフィールド専用の数式として保存され、評価された後、フィールドの表示が演算結果で置き換わります。

Formulas are stored in a special line starting with `#+TBLFM:` directly below the table. If you type the equation in the 4th field of the 3rd data line in the table, the formula will look like ‘@3\$4=\$1+\$2’. When inserting/deleting/swapping columns and rows with the appropriate commands, *absolute references* (but not relative ones) in stored formulas are modified in order to still reference the same field. To avoid this, in particular in range references, anchor ranges at the table borders (using @<, @>, \$<, \$>), or at hlines using the @I notation. Automatic adaptation of field references does of course not happen if you edit the table structure with normal editing commands—then you must fix the equations yourself.

Instead of typing an equation into the field, you may also use the following command



**C-u C-c =** **org-table-eval-formula**  
 Install a new formula for the current field. The command prompts for a formula with default taken from the **#+TBLFM:** line, applies it to the current field, and stores it.

多数の別フィールドに数式を割り当てるために、等式の左側は特別な表現を指定できます。範囲数式を入力するショートカットはありません。これを入力するには、数式エディタ (see [\(undefined\)](#) [数式の編集とデバッグ], page [\(undefined\)](#)) を用いるか、**#+TBLFM:** 行に直接記述します。

**\$2=** 列の数式。列の全体に対して有効。よく利用されるため、org-mode はこれらの数式を特別な方法で扱います。 [\(undefined\)](#) [列の数式], page [\(undefined\)](#). を参照してください。

**@3=** Row formula, applies to all fields in the specified row. **@>=** means the last row.

**@1\$2..@4\$3=**  
 範囲数式、与えられた矩形領域の全てのフィールドに適用する。これはある行の全てのフィールドではなく一部に対して数式を割り当てることにも利用できます。

**\$name=** 名前付きフィールド ([\(undefined\)](#) [一歩進んだ機能], page [\(undefined\)](#) を参照)

### 3.5.6 Column formulas

When you assign a formula to a simple column reference like **\$3=**, the same formula will be used in all fields of that column, with the following very convenient exceptions: (i) If the table contains horizontal separator hlines with rows above and below, everything before the first such hline is considered part of the table *header* and will not be modified by column formulas. Therefore a header is mandatory when you use column formulas and want to add hlines to group rows, like for example to separate a total row at the bottom from the summand rows above. (ii) Fields that already get a value from a field/range formula will be left alone by column formulas. These conditions make column formulas very easy to use.

To assign a formula to a column, type it directly into any field in the column, preceded by an equal sign, like **=\$1+\$2**. When you press **TAB** or **RET** or **C-c C-c** with the cursor still in the field, the formula will be stored as the formula for the current column, evaluated and the current field replaced with the result. If the field contains only **=**, the previously stored formula for this column is used. For each column, Org will only remember the most recently used formula. In the **#+TBLFM:** line, column formulas will look like **\$4=\$1+\$2**. The left-hand side of a column formula cannot be the name of column, it must be the numeric column reference or **\$>**.

フィールドに数式を書き入れる代わりに、次のコマンドも利用できます。

**C-c =** **org-table-eval-formula**  
 Install a new formula for the current column and replace current field with the result of the formula. The command prompts for a formula, with default taken from the **#+TBLFM** line, applies it to the current field and stores it. With a numeric prefix argument(e.g., **C-5 C-c =**) the command will apply it to that many consecutive fields in the current column.

### 3.5.7 Lookup functions

Org has three predefined Emacs Lisp functions for lookups in tables.

(org-lookup-first VAL S-LIST R-LIST &optional PREDICATE)  
 Searches for the first element S in list S-LIST for which

**(PREDICATE VAL S)**

is `t`; returns the value from the corresponding position in list `R-LIST`. The default `PREDICATE` is `equal`. Note that the parameters `VAL` and `S` are passed to `PREDICATE` in the same order as the corresponding parameters are in the call to `org-lookup-first`, where `VAL` precedes `S-LIST`. If `R-LIST` is `nil`, the matching element `S` of `S-LIST` is returned.

**(org-lookup-last VAL S-LIST R-LIST &optional PREDICATE)**

Similar to `org-lookup-first` above, but searches for the *last* element for which `PREDICATE` is `t`.

**(org-lookup-all VAL S-LIST R-LIST &optional PREDICATE)**

Similar to `org-lookup-first`, but searches for *all* elements for which `PREDICATE` is `t`, and returns *all* corresponding values. This function can not be used by itself in a formula, because it returns a list of values. However, powerful lookups can be built when this function is combined with other Emacs Lisp functions.

If the ranges used in these functions contain empty fields, the `E` mode for the formula should usually be specified: otherwise empty fields will not be included in `S-LIST` and/or `R-LIST` which can, for example, result in an incorrect mapping from an element of `S-LIST` to the corresponding element of `R-LIST`.

These three functions can be used to implement associative arrays, count matching cells, rank results, group data etc. For practical examples see this tutorial on Worg (<https://orgmode.org/worg/org-tutorials/org-lookups.html>).

### 3.5.8 Editing and debugging formulas

You can edit individual formulas in the minibuffer or directly in the field. Org can also prepare a special buffer with all active formulas of a table. When offering a formula for editing, Org converts references to the standard format (like `B3` or `D&`) if possible. If you prefer to only work with the internal format (like `@3$2` or `$4`), configure the option `org-table-use-standard-references`.

`C-c =` or `C-u C-c =` `org-table-eval-formula`  
 カーソル位置の列やフィールドに結びついている数式をミニバッファで編集する。〈undefined〉 [列の数式], page 〈undefined〉 と 〈undefined〉 [フィールドと範囲の数式], page 〈undefined〉 を参照してください。

`C-u C-u C-c =` `org-table-eval-formula`  
 カーソル位置のフィールドにアクティブな数式 (フィールドの数式、もしくは列の数式) を再挿入します。これはフィールドの中でそのまま編集できるようにするためです。ミニバッファでの編集よりも有利なのは、`C-c ?` を使えることです。

`C-c ?` `org-table-field-info`  
 テーブルのフィールド内で数式を編集するときに、数式内のカーソルが置かれている参照によって指し示されたフィールドをハイライトする。

`C-c }` `org-table-toggle-coordinate-overlays`  
 オーバーレイ (`org-table-toggle-coordinate-overlays`) を使用して、テーブルの行と列の番号を表示するようにトグルする。テーブルが整列される度に表示が変わり、`C-c C-c` で強制的に表示を更新できます。

|  |   |
|--|---|
| <code>C-c {</code>                           | 数式のデバッガを ON / OFF する ( <code>org-table-toggle-formula-debugger</code> )。  |
| <code>C-c '</code>                           | <code>org-table-edit-formulas</code><br>現在のテーブルにあるすべての数式を、特別なバッファで編集する。バッファでは数式はラインごとに一つずつ表示される。カーソル位置のフィールドにアクティブな数式があるとき、カーソルエディタのカーソルはその数式をマークします。この特別なバッファの中では、 <code>org-mode</code> は、カーソルがあるどんなフィールドや範囲参照も自動的にハイライトします。数式の編集、削除、追加が可能です、さらに次のコマンドを利用できます。  |
| <code>C-c C-c</code> or <code>C-x C-s</code> | <code>org-table-fedit-finish</code><br>Exit the formula editor and store the modified formulas. With <code>C-u</code> prefix, also apply the new formulas to the entire table.  |
| <code>C-c C-q</code>                         | <code>org-table-fedit-abort</code><br>Exit the formula editor without installing changes.   |
| <code>C-c C-r</code>                         | <code>org-table-fedit-toggle-ref-type</code><br>Toggle all references in the formula editor between standard (like B3) and internal (like @3\$2).   |
| <code>TAB</code>                             | <code>org-table-fedit-lisp-indent</code><br>Pretty-print or indent Lisp formula at point. When in a line containing a Lisp formula, format the formula according to Emacs Lisp rules. Another <code>TAB</code> collapses the formula back again. In the open formula, <code>TAB</code> re-indents just like in Emacs Lisp mode. |
| <code>M-TAB</code>                           | <code>lisp-complete-symbol</code><br>Complete Lisp symbols, just like in Emacs Lisp mode. <sup>8</sup>  |
| <code>S-UP/DOWN/LEFT/RIGHT</code>            | Shift the reference at point. For example, if the reference is B3 and you press <code>S-RIGHT</code> , it will become C3. This also works for relative references and for hline references.   |
| <code>M-S-UP</code>                          | <code>org-table-fedit-line-up</code>  |
| <code>M-S-DOWN</code>                        | <code>org-table-fedit-line-down</code>  |
|  | Move the test line for column formulas in the Org buffer up and down.   |
| <code>M-UP</code>                            | <code>org-table-fedit-scroll-down</code>  |
| <code>M-DOWN</code>                          | <code>org-table-fedit-scroll-up</code>  |
|  | Scroll the window displaying the table.   |
| <code>C-c }</code>                           | Turn the coordinate grid in the table on and off.   |

Making a table field blank does not remove the formula associated with the field, because that is stored in a different line (the `#+TBLFM` line)—during the next recalculation the field will be filled again. To remove a formula from a field, you have to give an empty reply when prompted for the formula, or to edit the `#+TBLFM` line.

You may edit the `#+TBLFM` directly and re-apply the changed equations with `C-c C-c` in that line or with the normal recalculation commands in the table.

<sup>8</sup> Many desktops intercept `M-TAB` to switch windows. Use `C-M-i` or `ESC TAB` instead for completion (see Section 15.1 [Completion], page 229).

## Using multiple #+TBLFM lines

You may apply the formula temporarily. This is useful when you switch the formula. Place multiple #+TBLFM lines right after the table, and then press `C-c C-c` on the formula to apply. Here is an example:

```
| x | y |
|----+----|
| 1 |   |
| 2 |   |
#+TBLFM: $2=$1*1
#+TBLFM: $2=$1*2
```

Pressing `C-c C-c` in the line of ‘#+TBLFM: \$2=\$1\*2’ yields:

```
| x | y |
|----+----|
| 1 | 2 |
| 2 | 4 |
#+TBLFM: $2=$1*1
#+TBLFM: $2=$1*2
```

Note: If you recalculate this table (with `C-u C-c *`, for example), you will get the following result of applying only the first #+TBLFM line.

```
| x | y |
|----+----|
| 1 | 1 |
| 2 | 2 |
#+TBLFM: $2=$1*1
#+TBLFM: $2=$1*2
```

### 数式のデバッグ

数式を評価してエラーが生じた時は、フィールドの文字列が‘#ERROR’に変わります。バグを見つけるために、変数の代入と計算の過程で何が起きているかを調べたいならば、`Tbl`メニューにある数式のデバッグを有効にして、計算をやり直してください。たとえば、フィールド内で `C-u C-u C-c = RET` と押します。すると、詳細な情報が表示されます。

## 3.5.9 Updating the table

テーブルの再計算は、通常は自動的に行なわれず、コマンドにより実行する必要があります。〈undefined〉 [一歩進んだ機能], page 〈undefined〉を参照してください。少なくとも半自動で計算されます。

テーブルのある行もしくはテーブル全体を再計算するために、次のコマンドを使います。

`C-c *` `org-table-recalculate`  
最初に保存されている列の数式を用いて、現在行を左から右に再計算します。現在行の全てのフィールドと範囲数式が対象です。

`C-u C-c *`  
`C-u C-c C-c`

行ごとにテーブル全体を再計算します。最初の水平ラインの以前にあるどの行も再計算されません。これらは、テーブルのヘッダーであると想定します。

**C-u C-u C-c \* or C-u C-u C-c C-c** **org-table-iterate**  
 計算結果の変化が生じなくなるまでテーブルの再計算を繰り返します。これは、計算の流れにおいて後段で計算される別フィールドの値を、すでに計算の対象となっているフィールドで利用する場合に必要ななるでしょう。

**M-x org-table-recalculate-buffer-tables RET**  
 現在のバッファに含まれる全てのフィールドを再計算します

**M-x org-table-iterate-buffer-tables RET**  
 テーブル間の依存関係を収束させるために、カレントバッファの全てのテーブルを反復計算する。

### 3.5.10 Advanced features

If you want the recalculation of fields to happen automatically, or if you want to be able to assign *names*<sup>9</sup> to fields and columns, you need to reserve the first column of the table for special marking characters.

**C-#** **org-table-rotate-recalc-marks**  
 第一行の計算用マークを‘!’、‘#’、‘\*’、‘!’、‘\$’の順番に循環します。アクティブリージョンがあれば、その領域のすべてのマークを変更します。

例として、学生の試験結果を集めて自動再計算の機能を使うテーブルを示します。

|    |         |        |        |        |       |      |  |  |
|----|---------|--------|--------|--------|-------|------|--|--|
|    |         |        |        |        |       |      |  |  |
|    | Student | Prob 1 | Prob 2 | Prob 3 | Total | Note |  |  |
| !  |         | P1     | P2     | P3     | Tot   |      |  |  |
| #  | Maximum | 10     | 15     | 25     | 50    | 10.0 |  |  |
| ^  |         | m1     | m2     | m3     | mt    |      |  |  |
| #  | Peter   | 10     | 8      | 23     | 41    | 8.2  |  |  |
| #  | Sam     | 2      | 4      | 3      | 9     | 1.8  |  |  |
|    | Average |        |        |        | 25.0  |      |  |  |
| ^  |         |        |        |        | at    |      |  |  |
| \$ | max=50  |        |        |        |       |      |  |  |

**#+TBLFM: \$6=vsum(\$P1..\$P3):: \$7=10\*\$Tot/\$max;%.1f::\$at=vmean(@-II..@-I);%.1f**

重要: このような特別なテーブルについて、**C-u C-c \***を使った再計算は、‘#’もしくは‘\*’でマークされた行、そして、自身に対して数式が適用されているフィールドだけに影響することに気をつけてください。最初のフィールドが空になっている行には、列の数式が適用されません。

マーキング文字には、次のような意味があります。

- ‘!’ 行に含まれるフィールドが、対応する列の名前を表します。これは‘\$6’ではなく‘\$Tot’として列を参照可能にするためです。
- ‘^’ 上側の列のフィールドについて名前を定義します。この定義を用いることで、テーブルのどのような数式でも、値‘10’を参照するために‘\$m1’を利用できます。また、名前を定義するフィールドに数式を割り当てると、‘\$name=...’として保存されます。

<sup>9</sup> Such names must start by an alphabetic character and use only alphanumeric/underscore characters.

|      |  |
|------|--|
| '_'  | '^'とほぼ同じだが、下側にある列のフィールドに対する名前を定義します。   |
| '\$' | この行のフィールドは、数式のパラメータを定義します。たとえば、'\$'が指定された行で、フィールドが'max=50'を含むならば、テーブルの数式は'\$max'を用いて値'50'を参照できます。パラメータは、正に定数として動作し、テーブルごとに定義されます。    |
| '#'  | 行の中でTAB、RET、またはS-TABを押すと、この行のフィールドは自動的に再計算されます。また、この行はC-u C-c *によるグローバルな再計算に利用されます。このコマンドでは、マークされていない行はそのままに維持されます。                  |
| '*'  | C-u C-c *によるグローバルな再計算にこの行を利用します。ただし、自動的な再計算には利用されません。自動再計算が編集速度を著しく低下させる場合に利用します。  |
| ' '  | Unmarked lines are exempt from recalculation with C-u C-c *. All lines that should be recalculated should be marked with '#' or '*'. |
| '/'  | この行はエクスポートしない。表示幅を狭くする'<N>'マーカ、もしくは列のグループマーカを含む列で使うと便利です。  |

最後に、素晴らしいcalc.elパッケージで何ができるのかを知るという学習欲を刺激するために、一つのテーブルを示します。このテーブルは、いくつかの関数に対して、 $x$ における $n$ 次のテーラー級数を計算します。

|   | Func                    | n | x   | Result  |
|---|-------------------------|---|-----|---|
| # | exp(x)                  | 1 | x   | 1 + x   |
| # | exp(x)                  | 2 | x   | 1 + x + x <sup>2</sup> / 2                      |
| # | exp(x)                  | 3 | x   | 1 + x + x <sup>2</sup> / 2 + x <sup>3</sup> / 6 |
| # | x <sup>2</sup> +sqrt(x) | 2 | x=0 | x*(0.5 / 0) + x <sup>2</sup> (2 - 0.25 / 0) / 2 |
| # | x <sup>2</sup> +sqrt(x) | 2 | x=1 | 2 + 2.5 x - 2.5 + 0.875 (x - 1) <sup>2</sup>    |
| * | tan(x)                  | 3 | x   | 0.0175 x + 1.77e-6 x <sup>3</sup>               |

#+TBLFM: \$5=taylor(\$2,\$4,\$3);n3

## 3.6 Org-Plot

Org-Plot can produce graphs of information stored in org tables, either graphically or in ASCII-art.

### Graphical plots using Gnuplot

Org-Plot produces 2D and 3D graphs using Gnuplot <http://www.gnuplot.info/> and gnuplot-mode <http://xafs.org/BruceRavel/GnuplotMode>. To see this in action, ensure that you have both Gnuplot and Gnuplot mode installed on your system, then call C-c " g or M-x org-plot/gnuplot RET on the following table.

```
#+PLOT: title:"Citas" ind:1 deps:(3) type:2d with:histograms set:"yrange [0:]"
| Sede          | Max cites | H-index |
|-----+-----+-----|
| Chile         | 257.72   | 21.39   |
| Leeds         | 165.77   | 19.68   |
| Sao Paolo     | 71.00    | 11.50   |
| Stockholm     | 134.19   | 14.33   |
| Morelia       | 257.56   | 17.67   |
```

Notice that Org Plot is smart enough to apply the table's headers as labels. Further control over the labels, type, content, and appearance of plots can be exercised through the `#+PLOT:` lines preceding a table. See below for a complete list of Org-plot options. The `#+PLOT:` lines are optional. For more information and examples see the Org-plot tutorial at <https://orgmode.org/worg/org-tutorials/org-plot.html>.

#### プロットオプション

|                |   |
|----------------|---|
| <b>set</b>     | グラフの描画時に設定される <code>gnuplot</code> のオプションを指定します。  |
| <b>title</b>   | プロットのタイトルを設定します。  |
| <b>ind</b>     | x軸として利用するテーブルの列を指定します。  |
| <b>deps</b>    | Lisp スタイルのリストで描画するように列を指定します。括弧で括られ、スペースで分離されます。たとえば、 <code>dep:(3 4)</code> とすると、第三と第四列を描画します（デフォルトでは、 <code>ind</code> の列を除いて他の全ての列が描画されます）。  |
| <b>type</b>    | プロットの種別（2d、3dもしくは <code>grid</code> ）を指定します。  |
| <b>with</b>    | Specify a <code>with</code> option to be inserted for every col being plotted (e.g., <code>lines</code> , <code>points</code> , <code>boxes</code> , <code>impulses</code> , etc...). Defaults to <code>lines</code> .  |
| <b>file</b>    | プロット結果を外部ファイルに出力したいときに、 <code>"path/to/desired/output-file"</code> のように指定します。   |
| <b>labels</b>  | <code>deps</code> で利用されるラベルのリストを指定します。標準では列のヘッダーが使われます（存在する場合）。   |
| <b>line</b>    | Gnuplot のスクリプトに、記述内容がそのまま挿入される行を設定します。  |
| <b>map</b>     | プロットの種別で 3d もしくは <code>grid</code> を指定する場合に、このオプションを <code>t</code> にすると、3d の立体的な傾斜がわかる表示ではなく、平らな表示（訳注：平面に射影した状態）でプロットします。  |
| <b>timefmt</b> | Specify format of Org mode timestamps as they will be parsed by Gnuplot. Defaults to <code>'%Y-%m-%d-%H:%M:%S'</code> .   |
| <b>script</b>  | Gnuplot をさらに制御するために、プロットに利用されるスクリプトファイル（ファイル名をダブルクォーテーションで囲んでください）を指定できます。プロットを始める前に、指定されたスクリプトファイルに含まれるすべての <code>\$datafile</code> の記述は、プロット点を格納するファイル（訳注：org-plot が作る一時的なプロット点データ）へのファイルパスで置き換えられます。仮にこのオプションを指定しても、データファイルの内容に影響を与えることができるように、プロットの種別を制御したくなるかもしれません。 |

## ASCII bar plots

While the cursor is on a column, typing `C-c " a` or `M-x orgtbl-ascii-plot RET` create a new column containing an ASCII-art bars plot. The plot is implemented through a regular column formula. When the source column changes, the bar plot may be updated by refreshing the table, for example typing `C-u C-c *`.

| Sede          | Max cites |              |
|---------------|-----------|--------------|
| Chile         | 257.72    | WWWWWWWWWWWW |
| Leeds         | 165.77    | WWWWWWWh     |
| Sao Paolo     | 71.00     | WWW;         |
| Stockholm     | 134.19    | WWWWW:       |
| Morelia       | 257.56    | WWWWWWWWWWH  |
| Rochefourchat | 0.00      |              |

```
#+TBLFM: $3='(orgtbl-ascii-draw $2 0.0 257.72 12)
```

The formula is an elisp call:

```
(orgtbl-ascii-draw COLUMN MIN MAX WIDTH)
```

**COLUMN** is a reference to the source column.

**MIN MAX** are the minimal and maximal values displayed. Sources values outside this range are displayed as ‘too small’ or ‘too large’.

**WIDTH** is the width in characters of the bar-plot. It defaults to ‘12’.



## 4 Hyperlinks

Like HTML, Org provides links inside a file, external links to other files, Usenet articles, emails, and much more.

### 4.1 Link format

Org will recognize plain URL-like links and activate them as clickable links. The general link format, however, looks like this:

`[[link][description]]` or alternatively `[[link]]`

Once a link in the buffer is complete (all brackets present), Org will change the display so that ‘description’ is displayed instead of ‘[[link][description]]’ and ‘link’ is displayed instead of ‘[[link]]’. Links will be highlighted in the face `org-link`, which by default is an underlined face. You can directly edit the visible part of a link. Note that this can be either the ‘link’ part (if there is no description) or the ‘description’ part. To edit also the invisible ‘link’ part, use `C-c C-l` with the cursor on the link.

If you place the cursor at the beginning or just behind the end of the displayed text and press `BACKSPACE`, you will remove the (invisible) bracket at that location. This makes the link incomplete and the internals are again displayed as plain text. Inserting the missing bracket hides the link internals again. To show the internal structure of all links, use the menu entry `Org->Hyperlinks->Literal links`.

### 4.2 Internal links

If the link does not look like a URL, it is considered to be internal in the current file. The most important case is a link like ‘[[#my-custom-id]]’ which will link to the entry with the `CUSTOM_ID` property ‘my-custom-id’. You are responsible yourself to make sure these custom IDs are unique in a file.

Links such as ‘[[My Target]]’ or ‘[[My Target][Find my target]]’ lead to a text search in the current file.

The link can be followed with `C-c C-o` when the cursor is on the link, or with a mouse click (see Section 4.4 [Handling links], page 41). Links to custom IDs will point to the corresponding headline. The preferred match for a text link is a *dedicated target*: the same string in double angular brackets, like ‘<<My Target>>’.

If no dedicated target exists, the link will then try to match the exact name of an element within the buffer. Naming is done with the `#+NAME` keyword, which has to be put in the line before the element it refers to, as in the following example

```
#+NAME: My Target
| a | table |
|----+-----|
| of | four cells |
```

If none of the above succeeds, Org will search for a headline that is exactly the link text but may also include a `TODO` keyword and tags<sup>1</sup>.

<sup>1</sup> To insert a link targeting a headline, in-buffer completion can be used. Just type a star followed by a few optional letters into the buffer and press `M-TAB`. All headlines in the current buffer will be offered as completions.

During export, internal links will be used to mark objects and assign them a number. Marked objects will then be referenced by links pointing to them. In particular, links without a description will appear as the number assigned to the marked object<sup>2</sup>. In the following excerpt from an Org buffer

```
- one item
- <<target>>another item
Here we refer to item [[target]].
```

The last sentence will appear as ‘Here we refer to item 2’ when exported.

In non-Org files, the search will look for the words in the link text. In the above example the search would be for ‘my target’.

Following a link pushes a mark onto Org’s own mark ring. You can return to the previous position with *C-c &*. Using this command several times in direct succession goes back to positions recorded earlier.

### 4.2.1 Radio targets

Org can automatically turn any occurrences of certain target names in normal text into a link. So without explicitly creating a link, the text connects to the target radioing its position. Radio targets are enclosed by triple angular brackets. For example, a target ‘<<<My Target>>>’ causes each occurrence of ‘my target’ in normal text to become activated as a link. The Org file is scanned automatically for radio targets only when the file is first loaded into Emacs. To update the target list during editing, press *C-c C-c* with the cursor on or at a target.

## 4.3 External links

Org supports links to files, websites, Usenet and email messages, BBDB database entries and links to both IRC conversations and their logs. External links are URL-like locators. They start with a short identifying string followed by a colon. There can be no space after the colon. The following list shows examples for each link type.

|  |                                      |
|--|--------------------------------------|
| <code>http://www.astro.uva.nl/~dominik</code>            | on the web                           |
| <code>doi:10.1000/182</code>                             | DOI for an electronic resource       |
| <code>file:/home/dominik/images/jupiter.jpg</code>       | file, absolute path                  |
| <code>/home/dominik/images/jupiter.jpg</code>            | same as above                        |
| <code>file:papers/last.pdf</code>                        | file, relative path                  |
| <code>./papers/last.pdf</code>                           | same as above                        |
| <code>file:/ssh:myself@some.where:papers/last.pdf</code> | file, path on remote machine         |
| <code>/ssh:myself@some.where:papers/last.pdf</code>      | same as above                        |
| <code>file:sometextfile::NNN</code>                      | file, jump to line number            |
| <code>file:projects.org</code>                           | another Org file                     |
| <code>file:projects.org::some words</code>               | text search in Org file <sup>3</sup> |

<sup>2</sup> When targeting a `#+NAME` keyword, `#+CAPTION` keyword is mandatory in order to get proper numbering (see Section 11.4 [Images and tables], page 132).

<sup>3</sup> The actual behavior of the search will depend on the value of the option `org-link-search-must-match-exact-headline`. If its value is `nil`, then a fuzzy text search will be done. If it is `t`, then only the exact headline will be matched, ignoring spaces and cookies. If the value is `query-to-create`, then an exact headline will be searched; if it is not found, then the user will be queried to create it.

|  |   |
|--|---|
| <code>file:projects.org:*task title</code>             | heading search in Org file <sup>4</sup> |
| <code>docview:papers/last.pdf::NNN</code>              | open in doc-view mode at page           |
| <code>id:B7423F4D-2E8A-471B-8810-C40F074717E9</code>   | Link to heading by ID                   |
| <code>news:comp.emacs</code>                           | Usenet link                             |
| <code>mailto:adent@galaxy.net</code>                   | Mail link                               |
| <code>mhe:folder</code>                                | MH-E folder link                        |
| <code>mhe:folder#id</code>                             | MH-E message link                       |
| <code>rmail:folder</code>                              | RMAIL folder link                       |
| <code>rmail:folder#id</code>                           | RMAIL message link                      |
| <code>gnus:group</code>                                | Gnus group link                         |
| <code>gnus:group#id</code>                             | Gnus article link                       |
| <code>bbdb:R.*Stallman</code>                          | BBDB link (with regexp)                 |
| <code>irc:/irc.com/#emacs/bob</code>                   | IRC link                                |
| <code>info:org#External links</code>                   | Info node or index link                 |
| <code>shell:ls *.org</code>                            | A shell command                         |
| <code>elisp:org-agenda</code>                          | Interactive Elisp command               |
| <code>elisp:(find-file-other-frame "Elisp.org")</code> | Elisp form to evaluate                  |

On top of these built-in link types, some are available through the `contrib/` directory (see Section 1.2 [Installation], page 2). For example, these links to VM or Wanderlust messages are available when you load the corresponding libraries from the `contrib/` directory:

|   |                         |
|---|-------------------------|
| <code>vm:folder</code>                            | VM folder link          |
| <code>vm:folder#id</code>                         | VM message link         |
| <code>vm://myself@some.where.org/folder#id</code> | VM on remote machine    |
| <code>vm-imap:account:folder</code>               | VM IMAP folder link     |
| <code>vm-imap:account:folder#id</code>            | VM IMAP message link    |
| <code>wl:folder</code>                            | WANDERLUST folder link  |
| <code>wl:folder#id</code>                         | WANDERLUST message link |

For customizing Org to add new link types Section A.3 [Adding hyperlink types], page 242.

A link should be enclosed in double brackets and may contain a descriptive text to be displayed instead of the URL (see Section 4.1 [Link format], page 38), for example:

```
[[https://www.gnu.org/software/emacs/] [GNU Emacs]]
```

If the description is a file name or URL that points to an image, HTML export (see Section 12.9 [HTML export], page 150) will inline the image as a clickable button. If there is no description at all and the link points to an image, that image will be inlined into the exported HTML file.

Org also finds external links in the normal text and activates them as links. If spaces must be part of the link (for example in `'bbdb:Richard Stallman'`), or if you need to remove ambiguities about the end of the link, enclose them in square brackets.

<sup>4</sup> Headline searches always match the exact headline, ignoring spaces and cookies. If the headline is not found and the value of the option `org-link-search-must-match-exact-headline` is `query-to-create`, then the user will be queried to create it.

## 4.4 Handling links

Org provides methods to create a link in the correct syntax, to insert it into an Org file, and to follow the link.

**C-c l**

**org-store-link**

Store a link to the current location. This is a *global* command (you must create the key binding yourself) which can be used in any buffer to create a link. The link will be stored for later insertion into an Org buffer (see below). What kind of link will be created depends on the current buffer:

### Org mode buffers

For Org files, if there is a ‘<<target>>’ at the cursor, the link points to the target. Otherwise it points to the current headline, which will also be the description<sup>5</sup>.

If the headline has a `CUSTOM_ID` property, a link to this custom ID will be stored. In addition or alternatively (depending on the value of `org-id-link-to-org-use-id`), a globally unique ID property will be created and/or used to construct a link<sup>6</sup>. So using this command in Org buffers will potentially create two links: a human-readable from the custom ID, and one that is globally unique and works even if the entry is moved from file to file. Later, when inserting the link, you need to decide which one to use.

### Email/News clients: VM, Rmail, Wanderlust, MH-E, Gnus

Pretty much all Emacs mail clients are supported. The link will point to the current article, or, in some GNUS buffers, to the group. The description is constructed from the author and the subject.

### Web browsers: Eww, W3 and W3M

Here the link will be the current URL, with the page title as description.

### Contacts: BBDB

Links created in a BBDB buffer will point to the current entry.

### Chat: IRC

For IRC links, if you set the option `org-irc-link-to-logs` to `t`, a ‘file:’ style link to the relevant point in the logs for the current conversation is created. Otherwise an ‘irc:’ style link to the user/channel/server under the point will be stored.

### Other files

For any other files, the link will point to the file, with a search string (see Section 4.7 [Search options], page 45) pointing to the contents of the current line. If there is an active region, the selected words will form the basis of the search string. If the automatically created link is not working correctly or accurately enough, you can write custom functions to select the search string and to do the search for particular file types—see Section 4.8 [Custom searches], page 45. The key binding `C-c l` is only a suggestion—see Section 1.2 [Installation], page 2.

<sup>5</sup> If the headline contains a timestamp, it will be removed from the link and result in a wrong link—you should avoid putting timestamp in the headline.

<sup>6</sup> The library `org-id.el` must first be loaded, either through `org-customize` by enabling `org-id` in `org-modules`, or by adding `(require 'org-id)` in your Emacs init file.

**Agenda view**

When the cursor is in an agenda view, the created link points to the entry referenced by the current line.

**C-c C-l****org-insert-link**

Insert a link<sup>7</sup>. This prompts for a link to be inserted into the buffer. You can just type a link, using text for an internal link, or one of the link type prefixes mentioned in the examples above. The link will be inserted into the buffer<sup>8</sup>, along with a descriptive text. If some text was selected when this command is called, the selected text becomes the default description.

**Inserting stored links**

All links stored during the current session are part of the history for this prompt, so you can access them with UP and DOWN (or *M-p/n*).

**Completion support**

Completion with TAB will help you to insert valid link prefixes like ‘https:’, including the prefixes defined through link abbreviations (see Section 4.6 [Link abbreviations], page 44). If you press RET after inserting only the *prefix*, Org will offer specific completion support for some link types<sup>9</sup>. For example, if you type *file RET*, file name completion (alternative access: *C-u C-c C-l*, see below) will be offered, and after *bbdb RET* you can complete contact names.

**C-u C-c C-l**

When *C-c C-l* is called with a *C-u* prefix argument, a link to a file will be inserted and you may use file name completion to select the name of the file. The path to the file is inserted relative to the directory of the current Org file, if the linked file is in the current directory or in a sub-directory of it, or if the path is written relative to the current directory using ‘./’. Otherwise an absolute path is used, if possible with ‘~/’ for your home directory. You can force an absolute path with two *C-u* prefixes.

**C-c C-l** (with cursor on existing link)

When the cursor is on an existing link, *C-c C-l* allows you to edit the link and description parts of the link.

**C-c C-o****org-open-at-point**

Open link at point. This will launch a web browser for URLs (using *browse-url-at-point*), run VM/MH-E/Wanderlust/Rmail/Gnus/BBDB for the corresponding links, and execute the command in a shell link. When the cursor is on an internal link, this command runs the corresponding search. When the cursor is on a TAG list in a headline, it creates the corresponding TAGS view. If the cursor is on a timestamp, it compiles the agenda for that

<sup>7</sup> Note that you don’t have to use this command to insert a link. Links in Org are plain text, and you can type or paste them straight into the buffer. By using this command, the links are automatically enclosed in double brackets, and you will be asked for the optional descriptive text.

<sup>8</sup> After insertion of a stored link, the link will be removed from the list of stored links. To keep it in the list later use, use a triple *C-u* prefix argument to *C-c C-l*, or configure the option *org-keep-stored-link-after-insertion*.

<sup>9</sup> This works if a completion function is defined in the ‘:complete’ property of a link in *org-link-parameters*.

date. Furthermore, it will visit text and remote files in ‘file:’ links with Emacs and select a suitable application for local non-text files. Classification of files is based on file extension only. See option `org-file-apps`. If you want to override the default application and visit the file with Emacs, use a `C-u` prefix. If you want to avoid opening in Emacs, use a `C-u C-u` prefix.

If the cursor is on a headline, but not on a link, offer all links in the headline and entry text. If you want to setup the frame configuration for following links, customize `org-link-frame-setup`.

**RET** When `org-return-follows-link` is set, RET will also follow the link at point.

*mouse-2*

*mouse-1* On links, *mouse-1* and *mouse-2* will open the link just as `C-c C-o` would.

*mouse-3* Like *mouse-2*, but force file links to be opened with Emacs, and internal links to be displayed in another window<sup>10</sup>.

`C-c C-x C-v` `org-toggle-inline-images`

Toggle the inline display of linked images. Normally this will only inline images that have no description part in the link, i.e., images that will also be inlined during export. When called with a prefix argument, also display images that do have a link description. You can ask for inline images to be displayed at startup by configuring the variable `org-startup-with-inline-images`<sup>11</sup>.

`C-c %` `org-mark-ring-push`

Push the current position onto the mark ring, to be able to return easily. Commands following an internal link do this automatically.

`C-c &` `org-mark-ring-goto`

Jump back to a recorded position. A position is recorded by the commands following internal links, and by `C-c %`. Using this command several times in direct succession moves through a ring of previously recorded positions.

`C-c C-x C-n` `org-next-link`

`C-c C-x C-p` `org-previous-link`

Move forward/backward to the next link in the buffer. At the limit of the buffer, the search fails once, and then wraps around. The key bindings for this are really too long; you might want to bind this also to `C-n` and `C-p`

```
(add-hook 'org-load-hook
  (lambda ()
    (define-key org-mode-map "\C-n" 'org-next-link)
    (define-key org-mode-map "\C-p" 'org-previous-link)))
```

## 4.5 Using links outside Org

You can insert and follow links that have Org syntax not only in Org, but in any Emacs buffer. For this, you should create two global commands, like this (please select suitable global keys yourself):

<sup>10</sup> See the option `org-display-internal-link-with-indirect-buffer`

<sup>11</sup> with corresponding `#+STARTUP` keywords `inlineimages` and `noinlineimages`

```
(global-set-key "\C-c L" 'org-insert-link-global)
(global-set-key "\C-c o" 'org-open-at-point-global)
```

## 4.6 Link abbreviations

Long URLs can be cumbersome to type, and often many similar links are needed in a document. For this you can use link abbreviations. An abbreviated link looks like this

```
[[linkword:tag][description]]
```

where the tag is optional. The *linkword* must be a word, starting with a letter, followed by letters, numbers, ‘-’, and ‘\_’. Abbreviations are resolved according to the information in the variable `org-link-abbrev-alist` that relates the linkwords to replacement text. Here is an example:

```
(setq org-link-abbrev-alist
  '(("bugzilla" . "http://10.1.2.9/bugzilla/show_bug.cgi?id=")
    ("url-to-ja" . "http://translate.google.fr/translate?sl=en&tl=ja&u=%h")
    ("google" . "http://www.google.com/search?q=")
    ("gmap" . "http://maps.google.com/maps?q=%s")
    ("omap" . "http://nominatim.openstreetmap.org/search?q=%s&polygon=1")
    ("ads" . "http://adsabs.harvard.edu/cgi-bin/nph-abs_connect?author=%s&db_key=AST")))
```

If the replacement text contains the string ‘%s’, it will be replaced with the tag. Using ‘%h’ instead of ‘%s’ will url-encode the tag (see the example above, where we need to encode the URL parameter.) Using ‘%(my-function)’ will pass the tag to a custom function, and replace it by the resulting string.

If the replacement text doesn’t contain any specifier, the tag will simply be appended in order to create the link.

Instead of a string, you may also specify a function that will be called with the tag as the only argument to create the link.

With the above setting, you could link to a specific bug with `[[bugzilla:129]]`, search the web for ‘OrgMode’ with `[[google:OrgMode]]`, show the map location of the Free Software Foundation `[[gmap:51 Franklin Street, Boston]]` or of Carsten office `[[omap:Science Park 904, Amsterdam, The Netherlands]]` and find out what the Org author is doing besides Emacs hacking with `[[ads:Dominik,C]]`.

If you need special abbreviations just for a single Org buffer, you can define them in the file with

```
#+LINK: bugzilla http://10.1.2.9/bugzilla/show_bug.cgi?id=
#+LINK: google http://www.google.com/search?q=%s
```

In-buffer completion (see Section 15.1 [Completion], page 229) can be used after ‘[’ to complete link abbreviations. You may also define a function that implements special (e.g., completion) support for inserting such a link with `C-c C-l`. Such a function should not accept any arguments, and return the full link with prefix. You can add a completion function to a link like this:

```
(org-link-set-parameters ``type' :complete #'some-function)
```

## 4.7 Search options in file links

File links can contain additional information to make Emacs jump to a particular location in the file when following a link. This can be a line number or a search option after a double<sup>12</sup> colon. For example, when the command `C-c l` creates a link (see Section 4.4 [Handling links], page 41) to a file, it encodes the words in the current line as a search string that can be used to find this line back later when following the link with `C-c C-o`.

Here is the syntax of the different ways to attach a search to a file link, together with an explanation:

|               |  |
|---------------|--|
|               | <code>[[file:~/code/main.c::255]]</code>   |
|               | <code>[[file:~/xx.org::My Target]]</code>  |
|               | <code>[[file:~/xx.org::*My Target]]</code>   |
|               | <code>[[file:~/xx.org::#my-custom-id]]</code>  |
|               | <code>[[file:~/xx.org::/regexp/]]</code>   |
| 255           | Jump to line 255.  |
| My Target     | Search for a link target ‘<<My Target>>’, or do a text search for ‘my target’, similar to the search in internal links, see Section 4.2 [Internal links], page 38. In HTML export (see Section 12.9 [HTML export], page 150), such a file link will become an HTML reference to the corresponding named anchor in the linked file. |
| *My Target    | In an Org file, restrict search to headlines.  |
| #my-custom-id | Link to a heading with a CUSTOM_ID property  |
| /regexp/      | Do a regular expression search for <code>regexp</code> . This uses the Emacs command <code>occur</code> to list all matches in a separate window. If the target file is in Org mode, <code>org-occur</code> is used to create a sparse tree with the matches.  |

As a degenerate case, a file link with an empty file name can be used to search the current file. For example, `[[file::find me]]` does a search for ‘find me’ in the current file, just as ‘`[[find me]]`’ would.

## 4.8 Custom Searches

The default mechanism for creating search strings and for doing the actual search related to a file link may not work correctly in all cases. For example, BibTeX database files have many entries like ‘`year="1993"`’ which would not result in good search strings, because the only unique identification for a BibTeX entry is the citation key.

If you come across such a problem, you can write custom functions to set the right search string for a particular file type, and to do the search for the string in the file. Using `add-hook`, these functions need to be added to the hook variables `org-create-file-search-functions` and `org-execute-file-search-functions`. See the docstring for these variables for more information. Org actually uses this mechanism for BibTeX database files, and you can use the corresponding code as an implementation example. See the file `org-bibtex.el`.

---

<sup>12</sup> For backward compatibility, line numbers can also follow a single colon.



## 5 TODO items

Org mode does not maintain TODO lists as separate documents<sup>1</sup>. Instead, TODO items are an integral part of the notes file, because TODO items usually come up while taking notes! With Org mode, simply mark any entry in a tree as being a TODO item. In this way, information is not duplicated, and the entire context from which the TODO item emerged is always present.

Of course, this technique for managing TODO items scatters them throughout your notes file. Org mode compensates for this by providing methods to give you an overview of all the things that you have to do.

### 5.1 基本的な **TODO** の機能

どの見出しでも ‘TODO’ という言葉を前につけることで、TODO アイテムとみなします。例えば:

\*\*\* TODO サム フォーチュンに手紙を書く。

TODO 項目を入力するときの最も重要なコマンドは以下のとおりです。

**C-c C-t** **org-todo**

現在の TODO の状態を次のように切り替えます。

, -> (マーク無し) -> TODO -> DONE --.

'-----'

If TODO keywords have fast access keys (see Section 5.2.4 [Fast access to TODO states], page 49), you will be prompted for a TODO keyword through the fast selection interface; this is the default behavior when **org-use-fast-todo-selection** is non-nil.

The same rotation can also be done “remotely” from agenda buffers with the **t** command key (see Section 10.5 [Agenda commands], page 115).

**C-u C-c C-t**

When TODO keywords have no selection keys, select a specific keyword using completion; otherwise force cycling through TODO states with no prompt. When **org-use-fast-todo-selection** is set to **prefix**, use the fast selection interface.

**S-RIGHT** / **S-LEFT**

Select the following/preceding TODO state, similar to cycling. Useful mostly if more than two TODO states are possible (see Section 5.2 [TODO extensions], page 47). See also Section 15.10.2 [Conflicts], page 239, for a discussion of the interaction with **shift-selection-mode**. See also the variable **org-treat-S-cursor-todo-selection-as-state-change**.

**C-c / t** **org-show-todo-tree**

View TODO items in a *sparse tree* (see Section 2.6 [Sparse trees], page 11). Folds the entire buffer, but shows all TODO items (with not-DONE state) and the headings hierarchy above them. With a prefix argument (or by using **C-c**

<sup>1</sup> Of course, you can make a document that contains only long lists of TODO items, but this is not required.

/ T), search for a specific TODO. You will be prompted for the keyword, and you can also give a list of keywords like KWD1|KWD2|... to list entries that match any one of these keywords. With a numeric prefix argument N, show the tree for the Nth keyword in the option `org-todo-keywords`. With two prefix arguments, find all TODO states, both un-done and done.

**C-c a t**

`org-todo-list`

Show the global TODO list. Collects the TODO items (with not-DONE states) from all agenda files (see Chapter 10 [Agenda views], page 102) into a single buffer. The new buffer will be in `agenda-mode`, which provides commands to examine and manipulate the TODO entries from the new buffer (see Section 10.5 [Agenda commands], page 115). See Section 10.3.2 [Global TODO list], page 106, for more information.

**S-M-RET**

`org-insert-todo-heading`

Insert a new TODO entry below the current one.

Changing a TODO state can also trigger tag changes. See the docstring of the option `org-todo-state-tags-triggers` for details.

## 5.2 Extended use of TODO keywords

By default, marked TODO entries have one of only two states: TODO and DONE. Org mode allows you to classify TODO items in more complex ways with *TODO keywords* (stored in `org-todo-keywords`). With special setup, the TODO keyword system can work differently in different files.

注記、タグは見出しと特に TODO アイテムを分類するもう一つの方法です。(see `<undefined>` [タグ], page `<undefined>`).

### 5.2.1 ワークフローの状態としての TODO キーワード

You can use TODO keywords to indicate different *sequential* states in the process of working on an item, for example<sup>2</sup>:

```
(setq org-todo-keywords
  '((sequence "TODO" "FEEDBACK" "VERIFY" "|" "DONE" "DELEGATED")))
```

縦線は、TODO キーワード「アクションが必要な状態」と DONE 状態「アクションが必要ない状態」を分けます。縦線が指定されていない場合は、最後の状態が、DONE 状態として使用されます。With this setup, the command `C-c C-t` will cycle an entry from TODO to FEEDBACK, then to VERIFY, and finally to DONE and DELEGATED. You may also use a numeric prefix argument to quickly select a specific state. For example `C-3 C-c C-t` will change the state immediately to VERIFY. Or you can use `S-LEFT` to go backward through the sequence. If you define many keywords, you can use in-buffer completion (see Section 15.1 [Completion], page 229) or even a special one-key selection scheme (see Section 5.2.4 [Fast access to TODO states], page 49) to insert these words into the buffer. Changing a TODO state can be logged with a timestamp, see Section 5.3.2 [Tracking TODO state changes], page 52, for more information.

<sup>2</sup> Changing this variable only becomes effective after restarting Org mode in a buffer.

### 5.2.2 種類としての **TODO** キーワード

TODO キーワードの2つ目の使い方として、いろいろな種類のやるべき事として表す使用法です。例えば、項目を「仕事」または「家」を示すようにも使えます。もしくは、複数の人と同じプロジェクトに参加するとき、その中の何人かに彼らの名前を TODO キーワードとして使って直接やるべき事を割り当てたいかもしれません。これは、以下のように設定します。:

```
(setq org-todo-keywords '((type "Fred" "Sara" "Lucy" "|" "DONE")))
```

In this case, different keywords do not indicate a sequence, but rather different types. So the normal work flow would be to assign a task to a person, and later to mark it DONE. Org mode supports this style by adapting the workings of the command `C-c C-t`<sup>3</sup>. When used several times in succession, it will still cycle through all names, in order to first select the right type for a task. But when you return to the item after some time and execute `C-c C-t` again, it will switch from any name directly to DONE. Use prefix arguments or completion to quickly select a specific name. You can also review the items of a specific TODO type in a sparse tree by using a numeric prefix to `C-c / t`. For example, to see all things Lucy has to do, you would use `C-3 C-c / t`. To collect Lucy's items from all agenda files into a single buffer, you would use the numeric prefix argument as well when creating the global TODO list: `C-3 C-c a t`.

### 5.2.3 同一ファイル内での複数のキーワードセット

時には、異なるセットの TODO キーワードを同時に使いたい場合があるかもしれません。例えば、通常の TODO/DONE を使用しつつ、バグフィックスのワークフロー、さらにアイテムがキャンセルをされたことを表すもう一つの状態を使用したい場合などです「つまり DONE ではないが、次のアクションが必要ない場合」。その場合の設定は次のようになります:

```
(setq org-todo-keywords
  '((sequence "TODO" "|" "DONE")
    (sequence "REPORT" "BUG" "KNOWNCAUSE" "|" "FIXED")
    (sequence "|" "CANCELED")))
```

The keywords should all be different, this helps Org mode to keep track of which subsequence should be used for a given entry. In this setup, `C-c C-t` only operates within a subsequence, so it switches from DONE to (nothing) to TODO, and from FIXED to (nothing) to REPORT. Therefore you need a mechanism to initially select the correct sequence. Besides the obvious ways like typing a keyword or using completion, you may also apply the following commands:

`C-u C-u C-c C-t`

`C-S-RIGHT`

`C-S-LEFT` These keys jump from one TODO subset to the next. In the above example, `C-u C-u C-c C-t` or `C-S-RIGHT` would jump from TODO or DONE to REPORT, and any of the words in the second row to CANCELED. Note that the `C-S-` key binding conflict with `shift-selection-mode` (see Section 15.10.2 [Conflicts], page 239).

`S-RIGHT`

`S-LEFT` `S-LEFT` and `S-RIGHT` and walk through *all* keywords from all sets, so for example `S-RIGHT` would switch from DONE to REPORT in the example above. See also

<sup>3</sup> This is also true for the `t` command in the agenda buffers.

Section 15.10.2 [Conflicts], page 239, for a discussion of the interaction with `shift-selection-mode`.

### 5.2.4 Fast access to TODO states

If you would like to quickly change an entry to an arbitrary TODO state instead of cycling through the states, you can set up keys for single-letter access to the states. This is done by adding the selection character after each keyword, in parentheses<sup>4</sup>. For example:

```
(setq org-todo-keywords
  '(sequence "TODO(t)" "|" "DONE(d)")
    (sequence "REPORT(r)" "BUG(b)" "KNOWNCAUSE(k)" "|" "FIXED(f)")
    (sequence "|" "CANCELED(c)")))
```

If you then press `C-c C-t` followed by the selection key, the entry will be switched to this state. SPC can be used to remove any TODO keyword from an entry.<sup>5</sup>

### 5.2.5 Setting up keywords for individual files

It can be very useful to use different aspects of the TODO mechanism in different files. For file-local settings, you need to add special lines to the file which set the keywords and interpretation for that file only. For example, to set one of the two examples discussed above, you need one of the following lines anywhere in the file:

```
#+TODO: TODO FEEDBACK VERIFY | DONE CANCELED
```

(you may also write `#+SEQ_TODO` to be explicit about the interpretation, but it means the same as `#+TODO`), or

```
#+TYP_TODO: Fred Sara Lucy Mike | DONE
```

同時に複数のキーワードグループの設定には:

```
#+TODO: TODO | DONE
#+TODO: REPORT BUG KNOWNCAUSE | FIXED
#+TODO: | CANCELED
```

間違いなく正しいキーワードを使うため、そのバッファ内で `#+` をタイプして、*M-TAB* を使って補完してください。

Remember that the keywords after the vertical bar (or the last keyword if no bar is there) must always mean that the item is DONE (although you may use a different word). After changing one of these lines, use `C-c C-c` with the cursor still in the line to make the changes known to Org mode<sup>6</sup>.

### 5.2.6 Faces for TODO keywords

Org mode highlights TODO keywords with special faces: `org-todo` for keywords indicating that an item still has to be acted upon, and `org-done` for keywords indicating that an item is finished. If you are using more than 2 different states, you might want to use special

<sup>4</sup> All characters are allowed except `@^!`, which have a special meaning here.

<sup>5</sup> Check also the option `org-fast-tag-selection-include-todo`, it allows you to change the TODO state through the tags interface (see Section 6.2 [Setting tags], page 59), in case you like to mingle the two concepts. Note that this means you need to come up with unique keys across both sets of keywords.

<sup>6</sup> Org mode parses these lines only when Org mode is activated after visiting a file. `C-c C-c` with the cursor in a line starting with `#+` is simply restarting Org mode for the current buffer.

faces for some of them. This can be done using the option `org-todo-keyword-faces`. For example:

```
(setq org-todo-keyword-faces
      '(("TODO" . org-warning) ("STARTED" . "yellow")
        ("CANCELED" . (:foreground "blue" :weight bold))))
```

While using a list with face properties as shown for CANCELED *should* work, this does not always seem to be the case. If necessary, define a special face and use that. A string is interpreted as a color. The option `org-faces-easy-properties` determines if that color is interpreted as a foreground or a background color.

### 5.2.7 TODO dependencies

The structure of Org files (hierarchy and lists) makes it easy to define TODO dependencies. Usually, a parent TODO task should not be marked DONE until all subtasks (defined as children tasks) are marked as DONE. And sometimes there is a logical sequence to a number of (sub)tasks, so that one task cannot be acted upon before all siblings above it are done. If you customize the option `org-enforce-todo-dependencies`, Org will block entries from changing state to DONE while they have children that are not DONE. Furthermore, if an entry has a property `ORDERED`, each of its children will be blocked until all earlier siblings are marked DONE. Here is an example:

```
* TODO この TODO は、2 番が終るまで変更できない。
** DONE 1 番
** TODO 2 番

* Parent
:PROPERTIES:
:ORDERED: t
:END:
** TODO a
** TODO b, needs to wait for (a)
** TODO c, needs to wait for (a) and (b)
```

You can ensure an entry is never blocked by using the `NOBLOCKING` property:

```
* This entry is never blocked
:PROPERTIES:
:NOBLOCKING: t
:END:
```

**C-c C-x o** `org-toggle-ordered-property`  
Toggle the `ORDERED` property of the current entry. A property is used for this behavior because this should be local to the current entry, not inherited like a tag. However, if you would like to *track* the value of this property with a tag for better visibility, customize the option `org-track-ordered-property-with-tag`.

**C-u C-u C-u C-c C-t**  
Change TODO state, circumventing any state blocking.

If you set the option `org-agenda-dim-blocked-tasks`, TODO entries that cannot be closed because of such dependencies will be shown in a dimmed font or even made invisible in agenda views (see Chapter 10 [Agenda views], page 102).

You can also block changes of TODO states by looking at checkboxes (see Section 5.6 [Checkboxes], page 56). If you set the option `org-enforce-todo-checkbox-dependencies`, an entry that has unchecked checkboxes will be blocked from switching to DONE.

もし、より複雑な依存関係の構造が必要であれば、例えば、異なるツリーやファイルのエントリー同士の依存関係、付属モジュールの `org-depend.el` を参照してみてください。

## 5.3 Progress logging

Org mode can automatically record a timestamp and possibly a note when you mark a TODO item as DONE, or even each time you change the state of a TODO item. This system is highly configurable; settings can be on a per-keyword basis and can be localized to a file or even a subtree. For information on how to clock working time for a task, see Section 8.4 [Clocking work time], page 81.

### 5.3.1 Closing items

一番基本的な時間の記録機能は、いつ TODO アイテムが完了したかを記録することです。これは、次のようにしてください<sup>1</sup>

```
(setq org-log-done 'time)
```

Then each time you turn an entry from a TODO (not-done) state into any of the DONE states, a line ‘CLOSED: [timestamp]’ will be inserted just after the headline. If you turn the entry back into a TODO item through further state cycling, that line will be removed again. If you turn the entry back to a non-TODO state (by pressing *C-c C-t SPC* for example), that line will also be removed, unless you set `org-closed-keep-when-no-todo` to non-nil. If you want to record a note along with the timestamp, use<sup>2</sup>

```
(setq org-log-done 'note)
```

この設定によりメモの入力を聞いてきます。そのメモは ‘Closing Note’ という見出しの下に挿入されます。

### 5.3.2 Tracking TODO state changes

When TODO keywords are used as workflow states (see Section 5.2.1 [Workflow states], page 47), you might want to keep track of when a state change occurred and maybe take a note about this change. You can either record just a timestamp, or a time-stamped note for a change. These records will be inserted after the headline as an itemized list, newest first<sup>3</sup>. When taking a lot of notes, you might want to get the notes out of the way into a drawer (see Section 2.8 [Drawers], page 15). Customize `org-log-into-drawer` to get this behavior—the recommended drawer for this is called LOGBOOK<sup>4</sup>. You can also overrule the setting of this variable for a subtree by setting a `LOG_INTRO_DRAWER` property.

Since it is normally too much to record a note for every state, Org mode expects configuration on a per-keyword basis for this. This is achieved by adding special markers ‘!’ (for a timestamp) or ‘@’ (for a note with timestamp) in parentheses after each keyword. For example, with the setting

```
(setq org-todo-keywords
      '((sequence "TODO(t)" "WAIT(w@/!)" "|" "DONE(d!)" "CANCELED(c@)")))
```

To record a timestamp without a note for TODO keywords configured with ‘@’, just type *C-c C-c* to enter a blank note when prompted.

<sup>1</sup> これに対応するイン-バッファ定義は: `#+STARTUP: logdone`

<sup>2</sup> The corresponding in-buffer setting is: `#+STARTUP: lognotedone`.

<sup>3</sup> See the option `org-log-states-order-reversed`

<sup>4</sup> Note that the LOGBOOK drawer is unfolded when pressing SPC in the agenda to show an entry—use *C-u SPC* to keep it folded here

You not only define global TODO keywords and fast access keys, but also request that a time is recorded when the entry is set to DONE<sup>5</sup>, and that a note is recorded when switching to WAIT or CANCELED. The setting for WAIT is even more special: the ‘!’ after the slash means that in addition to the note taken when entering the state, a timestamp should be recorded when *leaving* the WAIT state, if and only if the *target* state does not configure logging for entering it. So it has no effect when switching from WAIT to DONE, because DONE is configured to record a timestamp only. But when switching from WAIT back to TODO, the ‘/!’ in the WAIT setting now triggers a timestamp even though TODO has no logging configured.

まったく同じ構文を使ってバッファ内のみ有効な設定を使用できます。

```
#+TODO: TODO(t) WAIT(w@/!) | DONE(d!) CANCELED(c@)
```

In order to define logging settings that are local to a subtree or a single item, define a **LOGGING** property in this entry. Any non-empty **LOGGING** property resets all logging settings to `nil`. You may then turn on logging for this specific tree using **#+STARTUP** keywords like `lognotedone` or `logrepeat`, as well as adding state specific settings like `TODO(!)`. For example

```
* TODO 各状態のタイムスタンプだけをログを取る
:PROPERTIES:
:LOGGING: TODO(!) WAIT(!) DONE(!) CANCELED(!)
:END:
* TODO WAIT に切り替えられ、さらに繰り返されたときだけログを取る
:PROPERTIES:
:LOGGING: WAIT(@) logrepeat
:END:
* TODO 何もログを取らない
:PROPERTIES:
:LOGGING: nil
:END:
```

### 5.3.3 Tracking your habits

Org-mode には、「habits(習慣)」と呼ばれる TODO の特別なカテゴリーの一貫性を記録するための機能があります。habit には、以下の性質があります:

1. You have enabled the **habits** module by customizing **org-modules**.
2. The habit is a TODO item, with a TODO keyword representing an open state.
3. プロパティ **STYLE** に、**habit** を値として定義してある。
4. この TODO は、スケジュールされた日付があり、通常 **.+スタイル** で繰り返される間隔を表します。**++.スタイル** は、時間制限があるような場合に有効でしょう。例えば、週末にしなければいけないことなどです。**+スタイル** は、遅れることがあるような通常の習慣ではないような場合「例: 週次報告書」に適しています。
5. この TODO は、最短から最長の期間を **‘.+2d/3d’** のようなシンタックスで指定できます。この例の場合、このタスクを少なくとも 3 日ごとか、多くて 2 日ごとにこなすと指定しています。

<sup>5</sup> It is possible that Org mode will record two timestamps when you are using both **org-log-done** and state change logging. However, it will never prompt for two notes—if you have configured both, the state change recording note will take precedence and cancel the ‘Closing Note’.



6. You must also have state logging for the DONE state enabled (see Section 5.3.2 [Tracking TODO state changes], page 52), in order for historical data to be represented in the consistency graph. If it is not enabled it is not an error, but the consistency graphs will be largely meaningless.

上記にある定義が実際にはどのようなことになるか分かってもらうために、ここに経過の記録情報と共に実際の habit があります。

```
** TODO Shave
SCHEDULED: <2009-10-17 Sat .+2d/4d>
:PROPERTIES:
:STYLE:      habit
:LAST_REPEAT: [2009-10-19 Mon 00:36]
:END:
- State "DONE"      from "TODO"      [2009-10-15 Thu]
- State "DONE"      from "TODO"      [2009-10-12 Mon]
- State "DONE"      from "TODO"      [2009-10-10 Sat]
- State "DONE"      from "TODO"      [2009-10-04 Sun]
- State "DONE"      from "TODO"      [2009-10-02 Fri]
- State "DONE"      from "TODO"      [2009-09-29 Tue]
- State "DONE"      from "TODO"      [2009-09-25 Fri]
- State "DONE"      from "TODO"      [2009-09-19 Sat]
- State "DONE"      from "TODO"      [2009-09-16 Wed]
- State "DONE"      from "TODO"      [2009-09-12 Sat]
```

この habit が表しているのは、髭剃りを多くて2日ごとか「SCHEDULEDにより指定された日付と繰り返される間隔」、少なくとも4日ごとにする。もし、今日が15日とすると、この habit は、二日後の10月17日にアジェンダに表示されて、4日後の19日には、期限切れとして表示されます。

habits の本当に使い易いところは、定期性グラフと表示されることです。これは、過去にどれぐらいタスクが定期的に完了したかを見るためのものです。このグラフは、毎日過去3週間のタスクが完了したかを色分けして表示します。各色は、以下を表します:

|   |                           |
|---|---------------------------|
| 青 | まだ、その日までにタスクが完了してなくていい場合。 |
| 緑 | その日に完了できたはずのタスクの場合。       |
| 黄 | 明日になると期限切れになるタスクの場合。      |
| 赤 | その日に期限切れのタスクの場合。          |

日ごとの色分けだけでなく、その日に終わったタスクに関してはアスタリスクでマークされ、感嘆符が、グラフ中の今日の日付の部分に付きます。

アジェンダ上で habits が表示方法を変える幾つかの設定変数があります。

#### org-habit-graph-column

The buffer column at which the consistency graph should be drawn. This will overwrite any text in that column, so it is a good idea to keep your habits' titles brief and to the point.

#### org-habit-preceding-days

今日より前に、定期的グラフに何日分の日付を表示するかの指定。

#### org-habit-following-days

今日より後に、定期的グラフに何日分の日付表示するかの指定。

**org-habit-show-habits-only-for-today**

If non-nil, only show habits in today's agenda view. This is set to true by default.

最後に、アジェンダバッファで *K* を押すことにより habits を一時的に使用不可にし、まったく表示させないようにできます。もう一度 *K* を押すと元にもどります。例えば、もし特定の前後の内容によるのみ必要な habits の場合は、タグのフィルタリングの影響も受けます。

## 5.4 Priorities

If you use Org mode extensively, you may end up with enough TODO items that it starts to make sense to prioritize them. Prioritizing can be done by placing a *priority cookie* into the headline of a TODO item, like this

\*\*\* TODO [#A] サム フォーチュンに手紙を書く。

By default, Org mode supports three priorities: 'A', 'B', and 'C'. 'A' is the highest priority. An entry without a cookie is treated just like priority 'B'. Priorities make a difference only for sorting in the agenda (see Section 10.3.1 [Weekly/daily agenda], page 104); outside the agenda, they have no inherent meaning to Org mode. The cookies can be highlighted with special faces by customizing **org-priority-faces**.

Priorities can be attached to any outline node; they do not need to be TODO items.

**C-c** , Set the priority of the current headline (**org-priority**). The command prompts for a priority character 'A', 'B' or 'C'. When you press SPC instead, the priority cookie is removed from the headline. The priorities can also be changed "remotely" from the agenda buffer with the **,** command (see Section 10.5 [Agenda commands], page 115).

**S-UP**

**org-priority-up**

**S-DOWN**

**org-priority-down**

現在の見出し優先度を上下する。<sup>6</sup> これらのキーはタイムスタンプを修正するのにも使うので注意してください。「see (undefined) [タイムスタンプの作成], page (undefined)」**shift-selection-mode**との相互利用に関しては次を参照してください。(undefined) [衝突], page (undefined)

You can change the range of allowed priorities by setting the options **org-highest-priority**, **org-lowest-priority**, and **org-default-priority**. For an individual buffer, you may set these values (highest, lowest, default) like this (please make sure that the highest priority is earlier in the alphabet than the lowest priority):

**#+PRIORITIES: A C B**

## 5.5 タスクをサブタスクに細分化する。

通常、大きなタスクは小さくて管理しやすいサブタスクに細分化することをお勧めします。これは TODO アイテムの下にアウトラインツリーをさらに詳細なサブタスクをそのツリーを付けて作成することで可能です。<sup>7</sup> すでに完了したサブタスクの進捗状況を表示してさせておくには、'[ / ]'か'[ % ]'

<sup>6</sup> 次のオプションも参照してください。 **org-priority-start-cycle-with-default**.

<sup>7</sup> サブタスクをグローバル TODO リストに含めないようにするには、**org-agenda-todo-list-sublevels**を参照してください。

ヘッドラインのどこかに挿入してください。これらのクッキーは、サブタスクの TODO の状態が変わるかクッキー上で *C-c C-c* を押すたびに更新されます。例えば:

```
* パーティーの準備をする [33%]
** TODO 出席者に電話する [1/2]
*** TODO ピーター
*** DONE サラ
** TODO 食べ物を買う
** DONE 近所の人と話す
```

もし、見出しがチェックボックスと子 TODO を両方持っていた場合、統計クッキーは、あいまいなものになります。この問題を解決するには、`COOKIE_DATA` プロパティを `'checkbox'` か `'todo'` に設定してください。

If you would like to have the statistics cookie count any TODO entries in the subtree (not just direct children), configure `org-hierarchical-todo-statistics`. To do this for a single subtree, include the word `'recursive'` into the value of the `COOKIE_DATA` property.

```
* 親キャプチャ統計 [2/20]
:PROPERTIES:
:COOKIE_DATA: todo recursive
:END:
```

もしすべての子タスクが終了後、TODO エントリーを自動的に DONE に切り替えたい場合は、以下の設定を行なってください:

```
(defun org-summary-todo (n-done n-not-done)
  "すべてのサブツリーが終了すると DONE に切り替え、その他の場合は、TODO になる。"
  (let (org-log-done org-log-states) ; 記録「logging」を終了
    (org-todo (if (= n-not-done 0) "DONE" "TODO"))))
(add-hook 'org-after-todo-statistics-hook 'org-summary-todo)
```

その他の方法としては、チェックボックスをつかって「階層化された」多量のサブタスクがいくつあるか調べることもできます。「see `<undefined>` [チェックボックス], page `<undefined>`」

## 5.6 Checkboxes

Every item in a plain list<sup>8</sup> (see Section 2.7 [Plain lists], page 12) can be made into a checkbox by starting it with the string `'[ ]'`. This feature is similar to TODO items (see Chapter 5 [TODO items], page 46), but is more lightweight. Checkboxes are not included in the global TODO list, so they are often great to split a task into a number of simple steps. Or you can use them in a shopping list. To toggle a checkbox, use *C-c C-c*, or use the mouse (thanks to Piotr Zielinski's `org-mouse.el`).

以下は、チェックボックスリストの例です:

```
* TODO パーティーの準備 [2/4]
- [-] みんなに連絡 [1/3]
- [ ] ピーター
- [X] サラ
- [ ] サム
```

<sup>8</sup> With the exception of description lists. But you can allow it by modifying `org-list-automatic-rules` accordingly.

- [X] 食べ物を注文
- [ ] どんな音楽を掛けるか考える
- [X] 近所の人と話す

チェックボックスは、階層化に対応しています。そのため、もしあるチェックボックス下の項目が複数のチェックボックスである場合、チェックボックスが全くチェックされていないか、いくつかされているか、全てされているかによりその内の一つのチェックボックスをチェックした状態にすると親チェックボックスに影響します。

The ‘[2/4]’ and ‘[1/3]’ in the first and second line are cookies indicating how many checkboxes present in this entry have been checked off, and the total number of checkboxes present. This can give you an idea on how many checkboxes remain, even without opening a folded entry. The cookies can be placed into a headline or into (the first line of) a plain list item. Each cookie covers checkboxes of direct children structurally below the headline/item on which the cookie appears<sup>9</sup>. You have to insert the cookie yourself by typing either ‘[/]’ or ‘[%]’. With ‘[/]’ you get an ‘n out of m’ result, as in the examples above. With ‘[%]’ you get information about the percentage of checkboxes checked (in the above example, this would be ‘[50%]’ and ‘[33%]’, respectively). In a headline, a cookie can count either checkboxes below the heading or TODO states of children, and it will display whatever was changed last. Set the property `COOKIE_DATA` to either ‘checkbox’ or ‘todo’ to resolve this issue.

アウトラインノードに `ORDERED` プロパティが設定されている場合、チェックボックスは、連続でチェックされていなければなりません。上部のチェックボックスがチェックされていない状態でその下部のチェックボックスをチェックしようとするとエラーがスローされます

以下のコマンドでチェックボックスを操作できます:

`C-c C-c` `org-toggle-checkbox`  
 Toggle checkbox status or (with prefix arg) checkbox presence at point. With a single prefix argument, add an empty checkbox or remove the current one<sup>10</sup>. With a double prefix argument, set it to ‘[-]’, which is considered to be an intermediate state.

`C-c C-x C-b` `org-toggle-checkbox`  
 Toggle checkbox status or (with prefix arg) checkbox presence at point. With double prefix argument, set it to ‘[-]’, which is considered to be an intermediate state.

- アクティブなリージョンがある場合は、そのリージョンの最初のチェックボックスをトグルします。そして残りのボックスを最初のボックスと同じ状態にします。前置引数と使用すると、リージョン内のすべてのチェックボックスを作成するか削除します。
- If the cursor is in a headline, toggle the state of the first checkbox in the region between this headline and the next—so *not* the entire subtree—and propagate this new state to all other checkboxes in the same area.
- アクティブなリージョンがなければ、その場所のチェックボックスをトグルします。

<sup>9</sup> Set the option `org-checkbox-hierarchical-statistics` if you want such cookies to count all checkboxes below the cookie, not just those belonging to direct children.

<sup>10</sup> `C-u C-c C-c` before the *first* bullet in a list with no checkbox will add checkboxes to the rest of the list.

- M-S-RET*** **org-insert-todo-heading**  
新しい項目をチェックボックスと共に挿入します。これは、プレーンなリスト (see [\(undefined\)](#) [エントリー中の階層構造], page [\(undefined\)](#)) 内にカーソルがすでにある場合にのみ動作します。
- C-c C-x o*** **org-toggle-ordered-property**  
Toggle the **ORDERED** property of the entry, to toggle if checkboxes must be checked off in sequence. A property is used for this behavior because this should be local to the current entry, not inherited like a tag. However, if you would like to *track* the value of this property with a tag for better visibility, customize **org-track-ordered-property-with-tag**.
- C-c #*** **org-update-statistics-cookies**  
Update the statistics cookie in the current outline entry. When called with a *C-u* prefix, update the entire file. Checkbox statistic cookies are updated automatically if you toggle checkboxes with *C-c C-c* and make new ones with *M-S-RET*. TODO statistics cookies update when changing TODO states. If you delete boxes/entries or add/change them by hand, use this command to get things back into sync.

## 6 Tags

An excellent way to implement labels and contexts for cross-correlating information is to assign *tags* to headlines. Org mode has extensive support for tags.

Every headline can contain a list of tags; they occur at the end of the headline. Tags are normal words containing letters, numbers, ‘\_’, and ‘@’. Tags must be preceded and followed by a single colon, e.g., ‘:work:’. Several tags can be specified, as in ‘:work:urgent:’. Tags will by default be in bold face with the same color as the headline. You may specify special faces for specific tags using the option `org-tag-faces`, in much the same way as you can for TODO keywords (see Section 5.2.6 [Faces for TODO keywords], page 49).

### 6.1 Tag inheritance

*Tags* make use of the hierarchical structure of outline trees. If a heading has a certain tag, all subheadings will inherit the tag as well. For example, in the list

```
* Meeting with the French group      :work:
** Summary by Frank                  :boss:notes:
*** TODO Prepare slides for him      :action:
```

the final heading will have the tags ‘:work:’, ‘:boss:’, ‘:notes:’, and ‘:action:’ even though the final heading is not explicitly marked with all those tags. You can also set tags that all entries in a file should inherit just as if these tags were defined in a hypothetical level zero that surrounds the entire file. Use a line like this<sup>1</sup>:

```
#+FILETAGS: :Peter:Boss:Secret:
```

To limit tag inheritance to specific tags, use `org-tags-exclude-from-inheritance`. To turn it off entirely, use `org-use-tag-inheritance`.

When a headline matches during a tags search while tag inheritance is turned on, all the sublevels in the same tree will (for a simple match form) match as well<sup>2</sup>. The list of matches may then become very long. If you only want to see the first tags match in a subtree, configure `org-tags-match-list-sublevels` (not recommended).

Tag inheritance is relevant when the agenda search tries to match a tag, either in the `tags` or `tags-todo` agenda types. In other agenda types, `org-use-tag-inheritance` has no effect. Still, you may want to have your tags correctly set in the agenda, so that tag filtering works fine, with inherited tags. Set `org-agenda-use-tag-inheritance` to control this: the default value includes all agenda types, but setting this to `nil` can really speed up agenda generation.

### 6.2 Setting tags

簡単に見出しの最後へタグは入力することができます。コロンの次に *M-TAB* でタグの補完をします。他にもタグの入力のための特別なコマンドが以下に続きます:

|   |                             |
|---|-----------------------------|
| <b>C-c C-q</b>  | <b>org-set-tags-command</b> |
| Enter new tags for the current headline. Org mode will either offer completion or a special single-key interface for setting tags, see below. After pressing RET, |                             |

<sup>1</sup> As with all these in-buffer settings, pressing `C-c C-c` activates any changes in the line.

<sup>2</sup> This is only true if the search does not involve more complex tests including properties (see Section 7.3 [Property searches], page 66).

the tags will be inserted and aligned to `org-tags-column`. When called with a `C-u` prefix, all tags in the current buffer will be aligned to that column, just to make things look nice. TAGS are automatically realigned after promotion, demotion, and TODO state changes (see Section 5.1 [TODO basics], page 46).

`C-c C-c`

`org-set-tags-command`

When the cursor is in a headline, this does the same as `C-c C-q`.

Org supports tag insertion based on a *list of tags*. By default this list is constructed dynamically, containing all tags currently used in the buffer. You may also globally specify a hard list of tags with the variable `org-tag-alist`. Finally you can set the default tags for a given file with lines like

```
#+TAGS: @work @home @tennisclub
```

```
#+TAGS: laptop car pc sailboat
```

もし変数 `org-tag-alist` にタグを設定することでタグリストがグローバルに定義されていたとして、そのリストよりもファイルから生成された動的なタグリストを利用したければ、次のような空の TAGS オプションをファイルに指定します

```
#+TAGS:
```

If you have a preferred set of tags that you would like to use in every file, in addition to those defined on a per-file basis by TAGS option lines, then you may specify a list of tags with the variable `org-tag-persistent-alist`. You may turn this off on a per-file basis by adding a `#+STARTUP` option line to that file:

```
#+STARTUP: noptag
```

By default Org mode uses the standard minibuffer completion facilities for entering tags. However, it also implements another, quicker, tag selection method called *fast tag selection*. This allows you to select and deselect tags with just a single key press. For this to work well you should assign unique, case-sensitive, letters to most of your commonly used tags. You can do this globally by configuring the variable `org-tag-alist` in your Emacs init file. For example, you may find the need to tag many items in different files with `:@home:`. In this case you can set something like:

```
(setq org-tag-alist '(("@work" . ?w) ("@home" . ?h) ("laptop" . ?l)))
```

もしそのタグが作業中のファイルにおいてのみふさわしいのであれば、かわりに次のように TAGS オプションを書くこともできます。

```
#+TAGS: @work(w) @home(h) @tennisclub(t) laptop(l) pc(p)
```

タグインタフェースはスプラッシュウィンドウにとりうるタグを表示します。もし特定のタグの後に新しい行から始めたいのであればタグリストに `\n` を挿入します。

```
#+TAGS: @work(w) @home(h) @tennisclub(t) \n laptop(l) pc(p)
```

もしくは2行に分けて書きます。

```
#+TAGS: @work(w) @home(h) @tennisclub(t)
```

```
#+TAGS: laptop(l) pc(p)
```

また次のようにブレース (波括弧 `{}`) を使うことで相互排除したグループにタグをまとめることもできます。

```
#+TAGS: { @work(w) @home(h) @tennisclub(t) } laptop(l) pc(p)
```

これは `@work`、`@home`、`@tennisclub` のうち少なくとも一つは選択されることを意図しています。そのようなグループを複数持つことも可能です。

これらの変更を有効にするために、その行にカーソルを置いて `C-c C-c` を押すことを忘れないようにしてください。

To set these mutually exclusive groups in the variable `org-tag-alist`, you must use the dummy tags `:startgroup` and `:endgroup` instead of the braces. Similarly, you can use `:newline` to indicate a line break. The previous example would be set globally by the following configuration:

```
(setq org-tag-alist '(:startgroup . nil)
                    ("@work" . ?w) ("@home" . ?h)
                    ("@tennisclub" . ?t)
                    (:endgroup . nil)
                    ("laptop" . ?l) ("pc" . ?p)))
```

If at least one tag has a selection key then pressing `C-c C-c` will automatically present you with a special interface, listing inherited tags, the tags of the current headline, and a list of all valid tags with corresponding keys<sup>3</sup>.

アサインされたキーを押すことで現在の行にタグが追加または削除されます。相互排除グループのタグを選択することでそのグループの他のタグは無効になります。

In this interface, you can also use the following special keys:

|                  |   |
|------------------|---|
| <code>TAB</code> | たとえあらかじめ定義されていないタグでも、ミニバッファでタグを入力します。バッファ内の全てのタグを補完することができます。コンマで区切ることで複数のタグを追加することも可能です。   |
| <code>SPC</code> | この行のタグを全てクリアします。  |
| <code>RET</code> | 変更された集合を確定します。  |
| <code>C-g</code> | 変更を破棄します。   |
| <code>q</code>   | もし <code>q</code> がタグにアサインされていなければ <code>C-g</code> のように変更を破棄します。   |
| <code>!</code>   | 相互排除しているタグのグループングを無効にします。これはそのグループ内のタグを(例外的に)複数個、タグづけるときに使います。  |
| <code>C-c</code> | Toggle auto-exit after the next change (see below). If you are using expert mode, the first <code>C-c</code> will display the selection window. |

このメソッド (特別なインタフェース) であなたはとても少ないタイプで見出しにタグをつけられます。上記の設定で、現在のタグをクリアして、`@home`、`laptop`、`pc` のタグをつけるには次のように入力します: `C-c C-c SPC h l p RET`。タグ `@home` を `@work` へ付け替えるには `C-c C-c w RET` とタイプするか代わりに `C-c C-c C-c w` とタイプします。定義されていないタグ `Sarah` を追加するには `C-c C-c TAB S a r a h RET RET` すればよいのです。

If you find that most of the time you need only a single key press to modify your list of tags, set `org-fast-tag-selection-single-key`. Then you no longer have to press `RET` to exit fast tag selection—it will immediately exit after the first change. If you then occasionally need more keys, press `C-c` to turn off auto-exit for the current tag selection process (in effect: start selection with `C-c C-c C-c` instead of `C-c C-c`). If you set the variable to the value `expert`, the special window is not even shown for single-key tag selection, it comes up only when you press an extra `C-c`.

<sup>3</sup> Keys will automatically be assigned to tags which have no configured keys.



### 6.3 Tag hierarchy

Tags can be defined in hierarchies. A tag can be defined as a *group tag* for a set of other tags. The group tag can be seen as the “broader term” for its set of tags. Defining multiple *group tags* and nesting them creates a tag hierarchy.

One use-case is to create a taxonomy of terms (tags) that can be used to classify nodes in a document or set of documents.

When you search for a group tag, it will return matches for all members in the group and its subgroups. In an agenda view, filtering by a group tag will display or hide headlines tagged with at least one of the members of the group or any of its subgroups. This makes tag searches and filters even more flexible.

You can set group tags by using brackets and inserting a colon between the group tag and its related tags—beware that all whitespaces are mandatory so that Org can parse this line correctly:

```
#+TAGS: [ GTD : Control Persp ]
```

In this example, ‘GTD’ is the *group tag* and it is related to two other tags: ‘Control’, ‘Persp’. Defining ‘Control’ and ‘Persp’ as group tags creates a hierarchy of tags:

```
#+TAGS: [ Control : Context Task ]
```

```
#+TAGS: [ Persp : Vision Goal AOF Project ]
```

That can conceptually be seen as a hierarchy of tags:

- GTD
  - Persp
    - Vision
    - Goal
    - AOF
    - Project
  - Control
    - Context
    - Task

You can use the `:startgrouptag`, `:grouptags` and `:endgrouptag` keyword directly when setting `org-tag-alist` directly:

```
(setq org-tag-alist '((:startgrouptag)
                      ("GTD")
                      (:grouptags)
                      ("Control")
                      ("Persp")
                      (:endgrouptag)
                      (:startgrouptag)
                      ("Control")
                      (:grouptags)
                      ("Context")
                      ("Task")
                      (:endgrouptag)))
```

The tags in a group can be mutually exclusive if using the same group syntax as is used for grouping mutually exclusive tags together; using curly brackets.

```
#+TAGS: { Context : @Home @Work @Call }
```

When setting `org-tag-alist` you can use `:startgroup & :endgroup` instead of `:startgroup tag & :endgroup tag` to make the tags mutually exclusive.

Furthermore, the members of a *group tag* can also be regular expressions, creating the possibility of a more dynamic and rule-based tag structure. The regular expressions in the group must be specified within `{ }`. Here is an expanded example:

```
#+TAGS: [ Vision : {V@.+} ]
#+TAGS: [ Goal : {G@.+} ]
#+TAGS: [ AOF : {AOF@.+} ]
#+TAGS: [ Project : {P@.+} ]
```

Searching for the tag ‘Project’ will now list all tags also including regular expression matches for ‘P@.+’, and similarly for tag searches on ‘Vision’, ‘Goal’ and ‘AOF’. For example, this would work well for a project tagged with a common project-identifier, e.g. ‘P@2014\_OrgTags’.

If you want to ignore group tags temporarily, toggle group tags support with `org-toggle-tags-groups`, bound to `C-c C-x q`. If you want to disable tag groups completely, set `org-group-tags` to `nil`.

## 6.4 Tag searches

一度タグシステムが設定されると、関連する情報を特殊なリストに集めるのに使われます。

`C-c / m` or `C-c \` `org-match-sparse-tree`  
 Create a sparse tree with all headlines matching a tags/property/TODO search. With a `C-u` prefix argument, ignore headlines that are not a TODO line. See Section 10.3.3 [Matching tags and properties], page 107.

`C-c a m` `org-tags-view`  
 Create a global list of tag matches from all agenda files. See Section 10.3.3 [Matching tags and properties], page 107.

`C-c a M` `org-tags-view`  
 Create a global list of tag matches from all agenda files, but check only TODO items and force checking subitems (see the option `org-tags-match-list-sublevels`).

These commands all prompt for a match string which allows basic Boolean logic like ‘+boss+urgent-project1’, to find entries with tags ‘boss’ and ‘urgent’, but not ‘project1’, or ‘Kathy|Sally’ to find entries tagged as ‘Kathy’ or ‘Sally’. The full syntax of the search string is rich and allows also matching against TODO keywords, entry levels and properties. For a complete description with many examples, see Section 10.3.3 [Matching tags and properties], page 107.

## 7 Properties and columns

A property is a key-value pair associated with an entry. Properties can be set so they are associated with a single entry, with every entry in a tree, or with every entry in an Org mode file.

There are two main applications for properties in Org mode. First, properties are like tags, but with a value. Imagine maintaining a file where you document bugs and plan releases for a piece of software. Instead of using tags like `:release_1:`, `:release_2:`, you can use a property, say `:Release:`, that in different subtrees has different values, such as 1.0 or 2.0. Second, you can use properties to implement (very basic) database capabilities in an Org buffer. Imagine keeping track of your music CDs, where properties could be things such as the album, artist, date of release, number of tracks, and so on.

プロパティはカラムビューで便利に編集、閲覧できます (see [〈undefined〉 \[カラムビュー\]](#), page [〈undefined〉](#)).

### 7.1 Property syntax

Properties are key-value pairs. When they are associated with a single entry or with a tree they need to be inserted into a special drawer (see Section 2.8 [Drawers], page 15) with the name `PROPERTIES`, which has to be located right below a headline, and its planning line (see Section 8.3 [Deadlines and scheduling], page 77) when applicable. Each property is specified on a single line, with the key (surrounded by colons) first, and the value after it. Keys are case-insensitive. Here is an example:

```
* CD collection
** Classic
*** Goldberg Variations
    :PROPERTIES:
    :Title:      Goldberg Variations
    :Composer:   J.S. Bach
    :Artist:     Glen Gould
    :Publisher:  Deutsche Grammophon
    :NDisks:     1
    :END:
```

Depending on the value of `org-use-property-inheritance`, a property set this way will either be associated with a single entry, or the subtree defined by the entry, see Section 7.4 [Property inheritance], page 67.

プロパティ '`:Xyz_ALL:`' のように設定する事で、特定のプロパティ '`:Xyz:`' の許容値を定義できます。この特別なプロパティは、もしレベル 1 のエントリに設定されたならば全てのツリーに適用されるように継承されます。許容値を定義すると、対応するプロパティの設定が簡単になり、入力ミスを防ぐ事ができます。CD コレクションの例では、以下のように 1 つのボックスの中に発売元とディスクの数をあらかじめ定義できます。

```
* CD collection
    :PROPERTIES:
    :NDisks_ALL:  1 2 3 4
    :Publisher_ALL: "Deutsche Grammophon" Philips EMI
    :END:
```

1つのファイル全体で継承されるプロパティを設定したいならば、以下の行のようにします。

```
#+PROPERTY: NDisks_ALL 1 2 3 4
```

Contrary to properties set from a special drawer, you have to refresh the buffer with `C-c C-c` to activate this change.

If you want to add to the value of an existing property, append a `+` to the property name. The following results in the property `var` having the value “foo=1 bar=2”.

```
#+PROPERTY: var foo=1
#+PROPERTY: var+ bar=2
```

It is also possible to add to the values of inherited properties. The following results in the `genres` property having the value “Classic Baroque” under the `Goldberg Variations` subtree.

```
* CD collection
** Classic
:PROPERTIES:
:GENRES: Classic
:END:
*** Goldberg Variations
:PROPERTIES:
:Title:      Goldberg Variations
:Composer:   J.S. Bach
:Artist:     Glen Gould
:Publisher:  Deutsche Grammophon
:NDisks:     1
:GENRES+:    Baroque
:END:
```

Note that a property can only have one entry per Drawer.

グローバル変数 `org-global-properties` に設定するプロパティの値は全ての Org-mode のファイルに継承されます。

以下のコマンドはプロパティを操作する助けとなります。

**M-TAB** pcomplete  
After an initial colon in a line, complete property keys. All keys used in the current file will be offered as possible completions.

**C-c C-x p** org-set-property  
Set a property. This prompts for a property name and a value. If necessary, the property drawer is created as well.

**C-u M-x org-insert-drawer RET**  
Insert a property drawer into the current entry. The drawer will be inserted early in the entry, but after the lines with planning information like deadlines.

**C-c C-c** org-property-action  
With the cursor in a property drawer, this executes property commands.

**C-c C-c s** org-set-property  
Set a property in the current entry. Both the property and the value can be inserted using completion.

|  |  |
|--|--|
| <i>S-RIGHT</i>   | <code>org-property-next-allowed-value</code>     |
| <i>S-LEFT</i>  | <code>org-property-previous-allowed-value</code> |
| Switch property at point to the next/previous allowed value.   |  |
| <i>C-c C-c d</i>   | <code>org-delete-property</code>                 |
| Remove a property from the current entry.  |  |
| <i>C-c C-c D</i>   | <code>org-delete-property-globally</code>        |
| Globally remove a property, from all entries in the current file.                                      |  |
| <i>C-c C-c c</i>   | <code>org-compute-property-at-point</code>       |
| Compute the property at point, using the operator and scope from the nearest column format definition. |  |

## 7.2 Special properties

Special properties provide an alternative access method to Org mode features, like the TODO state or the priority of an entry, discussed in the previous chapters. This interface exists so that you can include these states in a column view (see Section 7.5 [Column view], page 67), or to use them in queries. The following property names are special and should not be used as keys in the properties drawer:

|              |  |
|--------------|--|
| ALLTAGS      | All tags, including inherited ones.  |
| BLOCKED      | "t" if task is currently blocked by children or siblings.  |
| CLOCKSUM     | The sum of CLOCK intervals in the subtree. <code>org-clock-sum</code> must be run first to compute the values in the current buffer.                 |
| CLOCKSUM_T   | The sum of CLOCK intervals in the subtree for today. <code>org-clock-sum-today</code> must be run first to compute the values in the current buffer. |
| CLOSED       | When was this entry closed?  |
| DEADLINE     | The deadline time string, without the angular brackets.  |
| FILE         | The filename the entry is located in.  |
| ITEM         | The headline of the entry.   |
| PRIORITY     | The priority of the entry, a string with a single letter.  |
| SCHEDULED    | The scheduling timestamp, without the angular brackets.  |
| TAGS         | The tags defined directly in the headline.   |
| TIMESTAMP    | The first keyword-less timestamp in the entry.   |
| TIMESTAMP_IA | The first inactive timestamp in the entry.   |
| TODO         | The TODO keyword of the entry.   |

## 7.3 Property searches

プロパティに基づいて選択した特別なリストやツリーの抽出を作成するために、タグ検索 (see <undefined> [タグの検索], page <undefined>) の場合と同じコマンドが使えます。

|  |                                    |
|--|------------------------------------|
| <i>C-c / m</i> or <i>C-c \</i>   | <code>org-match-sparse-tree</code> |
| Create a sparse tree with all matching entries. With a <i>C-u</i> prefix argument, ignore headlines that are not a TODO line.    |                                    |
| <i>C-c a m</i>   | <code>org-tags-view</code>         |
| Create a global list of tag/property matches from all agenda files. See Section 10.3.3 [Matching tags and properties], page 107. |                                    |

**C-c a M****org-tags-view**

Create a global list of tag matches from all agenda files, but check only TODO items and force checking of subitems (see the option `org-tags-match-list-sublevels`).

検索文字列のための文法は、`<undefined>` [タグとプロパティとのマッチング], page `<undefined>` で説明されています。

1つのプロパティに基いて抽出したツリーを作成するための特別なコマンドもあります。

**C-c / p**

プロパティの値に基づいて抽出したツリーを作成します。これは最初にプロパティ名、次にその値の入力を促します。抽出したツリーは与えられた値でこのプロパティを定義する全てのエントリで作られます。その値を大括弧で括っていたならば正規表現として解釈され、プロパティ値に対してマッチします。

## 7.4 プロパティの継承

The outline structure of Org mode documents lends itself to an inheritance model of properties: if the parent in a tree has a certain property, the children can inherit this property. Org mode does not turn this on by default, because it can slow down property searches significantly and is often not needed. However, if you find inheritance useful, you can turn it on by setting the variable `org-use-property-inheritance`. It may be set to `t` to make all properties inherited from the parent, to a list of properties that should be inherited, or to a regular expression that matches inherited properties. If a property has the value `nil`, this is interpreted as an explicit undefine of the property, so that inheritance search will stop at this value and return `nil`.

Org mode has a few properties for which inheritance is hard-coded, at least for the special applications for which they are used:

**COLUMNS** :**COLUMNS**: プロパティは、カラムビュー (see `<undefined>` [カラムビュー], page `<undefined>`) のフォーマットを定義します。:**COLUMNS**: プロパティが定義されているレベルがカラムビューテーブルの開始地点として使われるという意味で継承されます。そして、列表示が有効である場所とサブツリーの場所とは無関係です。

**CATEGORY** アジェンダビュー用です。:**CATEGORY**: プロパティを通して設定されたカテゴリがサブツリー全体に適用されます。

**ARCHIVE** アーカイブ用です。:**ARCHIVE**: プロパティは、サブツリー全体のアーカイブ位置を定義します (see `<undefined>` [サブツリーを移動する], page `<undefined>`)。

**LOGGING** The **LOGGING** property may define logging settings for an entry or a subtree (see Section 5.3.2 [Tracking TODO state changes], page 52).

## 7.5 Column view

A great way to view and edit properties in an outline tree is *column view*. In column view, each outline node is turned into a table row. Columns in this table provide access to properties of the entries. Org mode implements columns by overlaying a tabular structure over the headline of each item. While the headlines have been turned into a table row, you can still change the visibility of the outline tree. For example, you get a compact table by switching to CONTENTS view (`S-TAB S-TAB`, or simply `c` while column view is active), but you can still open, read, and edit the entry below each headline. Or, you can switch to

column view after executing a sparse tree command and in this way get a table only for the selected items. Column view also works in agenda buffers (see Chapter 10 [Agenda views], page 102) where queries have collected selected items, possibly from a number of files.

### 7.5.1 Defining columns

最初にカラムビューを設定するために、カラムを定義する必要があります。これはカラムフォーマット行を定義する事によってなされます。

#### 7.5.1.1 Scope of column definitions

カラムフォーマットを定義するために、次のように行を記載します。

```
#+COLUMNS: %25ITEM %TAGS %PRIORITY %TODO
```

指定したツリーに適用するだけのフォーマットを指定するために、`:COLUMNS:` プロパティをそのツリーの一番上のノードに追加します。例えば、

```
** カラムビューの最初のノード
:PROPERTIES:
:COLUMNS: %25ITEM %TAGS %PRIORITY %TODO
:END:
```

もし `:COLUMNS:` プロパティがエントリに現れると、そのエントリ自身とそれ以下のサブツリー全体の列を定義します。列定義は文章の階層構造の一部なので、全てのサブレベルに十分一般的なレベル 1 の列を定義でき、また、ツリーのより深い部分を編集するとき、より下位の指定された列を定義できます。

#### 7.5.1.2 Column attributes

列定義は、列の属性の集りです。一般的な定義は以下のようになります。

```
%[width]property[(title)][{summary-type}]
```

Except for the percent sign and the property name, all items are optional. The individual parts have the following meaning:

|                       |   |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
|-----------------------|---|------------------|-----------------------------|-----------------------|--|--------------------|-------------------------------|--------------------|----------------------------|--------------------|-----------------|
| <i>width</i>          | An integer specifying the width of the column in characters.<br>If omitted, the width will be determined automatically.   |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <i>property</i>       | The property that should be edited in this column.<br>Special properties representing meta data are allowed here as well (see Section 7.2 [Special properties], page 66)  |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <i>title</i>          | The header text for the column. If omitted, the property name is used.  |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <i>{summary-type}</i> | The summary type. If specified, the column values for parent nodes are computed from the children <sup>1</sup> .<br>Supported summary types are:<br><table> <tr> <td><code>{+}</code></td><td>Sum numbers in this column.</td></tr> <tr> <td><code>{+;%.1f}</code></td><td>Like '+', but format result with '%.1f'.</td></tr> <tr> <td><code>{\${}</code></td><td>Currency, short for '+;%.2f'.</td></tr> <tr> <td><code>{min}</code></td><td>Smallest number in column.</td></tr> <tr> <td><code>{max}</code></td><td>Largest number.</td></tr> </table> | <code>{+}</code> | Sum numbers in this column. | <code>{+;%.1f}</code> | Like '+', but format result with '%.1f'. | <code>{\${}</code> | Currency, short for '+;%.2f'. | <code>{min}</code> | Smallest number in column. | <code>{max}</code> | Largest number. |
| <code>{+}</code>      | Sum numbers in this column.   |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <code>{+;%.1f}</code> | Like '+', but format result with '%.1f'.  |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <code>{\${}</code>    | Currency, short for '+;%.2f'.   |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <code>{min}</code>    | Smallest number in column.  |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |
| <code>{max}</code>    | Largest number.   |                  |                             |                       |  |                    |                               |                    |                            |                    |                 |

<sup>1</sup> If more than one summary type apply to the property, the parent values are computed according to the first of them.

|                      |  |
|----------------------|--|
| <code>{mean}</code>  | Arithmetic mean of numbers.                              |
| <code>{X}</code>     | Checkbox status, ‘[X]’ if all children are ‘[X]’.        |
| <code>{X/}</code>    | Checkbox status, ‘[n/m]’.                                |
| <code>{X%}</code>    | Checkbox status, ‘[n%]’.                                 |
| <code>{:}</code>     | Sum times, HH:MM, plain numbers are hours <sup>2</sup> . |
| <code>{:min}</code>  | Smallest time value in column.                           |
| <code>{:max}</code>  | Largest time value.                                      |
| <code>{:mean}</code> | Arithmetic mean of time values.                          |
| <code>{@min}</code>  | Minimum age <sup>3</sup> (in days/hours/mins/seconds).   |
| <code>{@max}</code>  | Maximum age (in days/hours/mins/seconds).                |
| <code>{@mean}</code> | Arithmetic mean of ages (in days/hours/mins/seconds).    |
| <code>{est+}</code>  | Add ‘low-high’ estimates.                                |

The `est+` summary type requires further explanation. It is used for combining estimates, expressed as ‘low-high’ ranges or plain numbers. For example, instead of estimating a particular task will take 5 days, you might estimate it as 5–6 days if you’re fairly confident you know how much work is required, or 1–10 days if you don’t really know what needs to be done. Both ranges average at 5.5 days, but the first represents a more predictable delivery.

When combining a set of such estimates, simply adding the lows and highs produces an unrealistically wide result. Instead, `est+` adds the statistical mean and variance of the sub-tasks, generating a final estimate from the sum. For example, suppose you had ten tasks, each of which was estimated at 0.5 to 2 days of work. Straight addition produces an estimate of 5 to 20 days, representing what to expect if everything goes either extremely well or extremely poorly. In contrast, `est+` estimates the full job more realistically, at 10–15 days.

Numbers are right-aligned when a format specifier with an explicit width like `%5d` or `%5.1f` is used.

You can also define custom summary types by setting `org-columns-summary-types`, which see.

以下は許可値にそって列定義を計算する例です。

```
:COLUMNS: %25ITEM %9Approved(Approved?){X} %Owner %11Status \
           %10Time_Estimate{:} %CLOCKSUM %CLOCKSUM_T
:Owner_ALL: Tammy Mark Karl Lisa Don
:Status_ALL: "In progress" "Not started yet" "Finished" ""
:Approved_ALL: "[ ]" "[X]"
```

The first column, ‘%25ITEM’, means the first 25 characters of the item itself, i.e., of the headline. You probably always should start the column definition with the ‘ITEM’ specifier. The other specifiers create columns ‘Owner’ with a list of names as allowed values, for ‘Status’

<sup>2</sup> A time can also be a duration, using effort modifiers defined in `org-effort-durations`, e.g., ‘3d 1h’. If any column is as such, the summary will also be an effort duration.

<sup>3</sup> An age is defined as a duration since a given time-stamp (see Section 8.1 [Timestamps], page 73). It can also be expressed as days, hours, minutes and seconds, identified by ‘d’, ‘h’, ‘m’ and ‘s’ suffixes, all mandatory, e.g., ‘0d 13h 0m 10s’.

<sup>4</sup> Please note that the COLUMNS definition must be on a single line—it is wrapped here only because of formatting constraints.



with four different possible values, and for a checkbox field ‘**Approved**’. When no width is given after the ‘%’ character, the column will be exactly as wide as it needs to be in order to fully display all values. The ‘**Approved**’ column does have a modified title (‘**Approved?**’, with a question mark). Summaries will be created for the ‘**Time\_Estimate**’ column by adding time duration expressions like HH:MM, and for the ‘**Approved**’ column, by providing an ‘[X]’ status if all children have been checked. The ‘**CLOCKSUM**’ and ‘**CLOCKSUM\_T**’ columns are special, they lists the sums of CLOCK intervals in the subtree, either for all clocks or just for today.

### 7.5.2 Using column view

カラムビューのオン・オフ

*C-c C-x C-c*

**org-columns**

Turn on column view. If the cursor is before the first headline in the file, or the function called with the universal prefix argument, column view is turned on for the entire file, using the **#+COLUMNS** definition. If the cursor is somewhere inside the outline, this command searches the hierarchy, up from point, for a **:COLUMNS:** property that defines a format. When one is found, the column view table is established for the tree starting at the entry that contains the **:COLUMNS:** property. If no such property is found, the format is taken from the **#+COLUMNS** line or from the variable **org-columns-default-format**, and column view is established for the current entry and its subtree.

*r* **org-columns-redo**  
Recreate the column view, to include recent changes made in the buffer.

*g* **org-columns-redo**  
Same as *r*.

*q* **org-columns-quit**  
Exit column view.

#### Editing values

*LEFT RIGHT UP DOWN*

フィールドからフィールドへカラムビューを通じて移動します。

*S-LEFT/RIGHT*

フィールドの次と前の許可値を切り替えます。このために、プロパティの指定された許可値を持つ必要があります。

*1..9,0* Directly select the Nth allowed value, 0 selects the 10th value.

*n* **org-columns-next-allowed-value**

*p* **org-columns-previous-allowed-value**

Same as *S-LEFT/RIGHT*

*e* **org-columns-edit-value**

Edit the property at point. For the special properties, this will invoke the same interface that you normally use to change that property. For example, when editing a TAGS property, the tag completion or fast selection interface will pop up.

|                                      |  |
|--------------------------------------|--|
| <i>C-c C-c</i>                       | <code>org-columns-set-tags-or-toggle</code>  |
|                                      | When there is a checkbox at point, toggle it.  |
| <i>v</i>                             | <code>org-columns-show-value</code>  |
|                                      | View the full value of this property. This is useful if the width of the column is smaller than that of the value.   |
| <i>a</i>                             | <code>org-columns-edit-allowed</code>  |
|                                      | Edit the list of allowed values for this property. If the list is found in the hierarchy, the modified value is stored there. If no list is found, the new value is stored in the first entry that is part of the current column view. |
| <b>Modifying the table structure</b> |  |
| <i>&lt;</i>                          | <code>org-columns-narrow</code>  |
| <i>&gt;</i>                          | <code>org-columns-widen</code>   |
|                                      | Make the column narrower/wider by one character.   |
| <i>S-M-RIGHT</i>                     | <code>org-columns-new</code>   |
|                                      | Insert a new column, to the left of the current column.  |
| <i>S-M-LEFT</i>                      | <code>org-columns-delete</code>  |
|                                      | Delete the current column.   |

### 7.5.3 Capturing column view

カラムビューはバッファへのオーバーレイのみなので、直接にエクスポートや印字ができません。カラムビューをキャプチャしたいならば、`columnview`ダイナミックビュー (see [\(undefined\)](#) [ダイナミックブロック], page [\(undefined\)](#)) を使って下さい。このブロックのフレームは以下のように見えます。

```
* The column view
#+BEGIN: columnview :hlines 1 :id "label"

#+END:
```

ダイナミックブロックを更新するには以下のコマンドを使用します。

|                                  |  |
|----------------------------------|--|
| <code>:id</code>                 | This is the most important parameter. Column view is a feature that is often localized to a certain (sub)tree, and the capture block might be at a different location in the file. To identify the tree whose view to capture, you can use 4 values: |
| <code>local</code>               | use the tree in which the capture block is located   |
| <code>global</code>              | make a global view, including all headings in the file   |
| <code>"file:path-to-file"</code> | run column view at the top of this file  |
| <code>"ID"</code>                | call column view in the tree that has an <code>:ID:</code> property with the value <i>label</i> . You can use <i>M-x org-id-copy RET</i> to create a globally unique ID for the current entry and copy it to the kill-ring.                          |
| <code>:hlines</code>             | <i>t</i> のとき、全ての行の後に横線を挿入します。数値 <i>N</i> のとき、レベル $\leq N$ を持つ各見出しの前に縦線を挿入します。  |
| <code>:vlines</code>             | When set to <i>t</i> , force column groups to get vertical lines.  |

**:maxlevel**

数値を設定すると、そのレベル以下のエントリをキャプチャしません。

**:skip-empty-rows**

tに設定すると、カラムビューの空でない指定子が ITEMのみである行をスキップします。

**:indent** When non-nil, indent each ITEM field according to its level.

次のコマンドはダイナミックブロックを挿入、更新します。

**C-c C-x i** **org-insert-columns-dblock**  
 Insert a dynamic block capturing a column view. You will be prompted for the scope or ID of the view.

**C-c C-c** or **C-c C-x C-u** **org-dblock-update**  
 Update dynamic block at point.

**C-u C-c C-x C-u** **org-update-all-dblocks**  
 Update all dynamic blocks (see Section A.7 [Dynamic blocks], page 248). This is useful if you have several clock table blocks, column-capturing blocks or other dynamic blocks in a buffer.

カラムビューテーブルに計算式を追加し、テーブルの前にプロットする命令を追加できます。これらはブロックの更新があっても生き残ります。テーブルの後に#+TBLFM:があるならば、テーブルは実際に更新の後に自動的に実際に再計算されます。

An alternative way to capture and process property values into a table is provided by Eric Schulte's `org-collector.el` which is a contributed package<sup>5</sup>. It provides a general API to collect properties from entries in a certain scope, and arbitrary Lisp expressions to process these values before inserting them into a table or a dynamic block.

## 7.6 プロパティAPI

プロパティにアクセス、変更するための完全な API があります。この API はプロパティと共に動作するために、また、それらを元とした機能を実装するために Emacs Lisp プログラムから使われます。詳細な情報は `<undefined>` [プロパティAPIを使う], page `<undefined>` を参照して下さい。

---

<sup>5</sup> Contributed packages are not part of Emacs, but are distributed with the main distribution of Org (visit <https://orgmode.org>).

## 8 Dates and times

To assist project planning, TODO items can be labeled with a date and/or a time. The specially formatted string carrying the date and time information is called a *timestamp* in Org mode. This may be a little confusing because timestamp is often used to indicate when something was created or last changed. However, in Org mode this term is used in a much wider sense.

### 8.1 タイムスタンプ、デッドラインおよびスケジューリング

A timestamp is a specification of a date (possibly with a time or a range of times) in a special format, either ‘<2003-09-16 Tue>’<sup>1</sup> or ‘<2003-09-16 Tue 09:39>’ or ‘<2003-09-16 Tue 12:00-12:30>’<sup>2</sup>. A timestamp can appear anywhere in the headline or body of an Org tree entry. Its presence causes entries to be shown on specific dates in the agenda (see Section 10.3.1 [Weekly/daily agenda], page 104). We distinguish:

プレーンなタイムスタンプ、イベント、アポイント

A simple timestamp just assigns a date/time to an item. This is just like writing down an appointment or event in a paper agenda. In the agenda display, the headline of an entry associated with a plain timestamp will be shown exactly on that date.

```
* Meet Peter at the movies
  <2006-11-01 Wed 19:15>
* Discussion on climate change
  <2006-11-02 Thu 20:00-22:00>
```

リピート間隔を指定したタイムスタンプ

タイムスタンプにはリピート間隔を含めることができます。すなわち単一の日時だけでなく、N日(d)、N週間(w)、Nヶ月(m)あるいはN年(y)といった一定の間隔で繰り返すようなケースに対応しています。例えば、毎週水曜日のアジェンダに表示する場合は以下ようになります。

```
* Pick up Sam at school
  <2007-05-16 Wed 12:30 +1w>
```

ダイアリー形式のS式項目

For more complex date specifications, Org mode supports using the special sexp diary entries implemented in the Emacs calendar/diary package<sup>3</sup>. For example with optional time

<sup>1</sup> In this simplest form, the day name is optional when you type the date yourself. However, any dates inserted or modified by Org will add that day name, for reading convenience.

<sup>2</sup> This is inspired by the standard ISO 8601 date/time format. To use an alternative format, see Section 8.2.2 [Custom time format], page 77.

<sup>3</sup> When working with the standard diary sexp functions, you need to be very careful with the order of the arguments. That order depends evily on the variable `calendar-date-style` (or, for older Emacs versions, `european-calendar-style`). For example, to specify a date December 1, 2005, the call might look like `(diary-date 12 1 2005)` or `(diary-date 1 12 2005)` or `(diary-date 2005 12 1)`, depending on the settings. This has been the source of much confusion. Org mode users can resort to special versions of these functions like `org-date` or `org-anniversary`. These work just like the corresponding `diary-` functions, but with stable ISO order of arguments (year, month, day) wherever applicable, independent of the value of `calendar-date-style`.

```
* 22:00-23:00 The nerd meeting on every 2nd Thursday of the month
<%(diary-float t 4 2)>
```

日付または時刻の間隔

2つのタイムスタンプを‘--’でつなげるにより、時間間隔を表現できます。時間間隔の指定されたヘッドラインは、間隔の始めと終わりの日、およびその間の現在表示されている日付の項目に表示されます。以下のような形です。

```
** アムステルダムでのミーティング
<2004-08-23 Mon>--<2004-08-26 Thu>
```

アクティブでないタイムスタンプ

プレーンなタイムスタンプと同様ですが、<>ではなく [] で囲むことによりアクティブでないタイムスタンプとなります。このようなタイムスタンプが指定されたエントリーは、アジェンダに表示されません。

```
* Gillian comes late for the fifth time
[2006-11-01 Wed]
```

## 8.2 Creating timestamps

For Org mode to recognize timestamps, they need to be in the specific format. All commands listed below produce timestamps in the correct format.

**C-c .** org-time-stamp  
日付を入力して、それに対応するタイムスタンプを挿入します。既にバッファ内に存在するタイムスタンプにカーソルが置かれている場合は、このコマンドは新たなタイムスタンプを挿入する代わりに、既にあるタイムスタンプを変更します。このコマンドを2回連続で使用すると、時間間隔を指定することができます。

**C-c !** org-time-stamp-inactive  
コマンド **C-c .** と同様ですが、アクティブでない（アジェンダのエントリーに反映されない）タイムスタンプを生成します。

**C-u C-c .**  
**C-u C-c !** Like **C-c .** and **C-c !**, but use the alternative format which contains date and time. The default time can be rounded to multiples of 5 minutes, see the option **org-time-stamp-rounding-minutes**.

**C-c C-c** Normalize timestamp, insert/fix day name if missing or wrong.

**C-c <** org-date-from-calendar  
Insert a timestamp corresponding to the cursor date in the Calendar.

**C-c >** org-goto-calendar  
Access the Emacs calendar for the current date. If there is a timestamp in the current line, go to the corresponding date instead.

**C-c C-o** org-open-at-point  
Access the agenda for the date given by the timestamp or -range at point (see Section 10.3.1 [Weekly/daily agenda], page 104).

**S-LEFT** org-timestamp-down-day  
**S-RIGHT** org-timestamp-up-day  
Change date at cursor by one day. These key bindings conflict with shift-selection and related modes (see Section 15.10.2 [Conflicts], page 239).

|   |                                      |
|---|--------------------------------------|
| <i>S-UP</i>   | <code>org-timestamp-up</code>        |
| <i>S-DOWN</i>   | <code>org-timestamp-down-down</code> |
| <p>Change the item under the cursor in a timestamp. The cursor can be on a year, month, day, hour or minute. When the timestamp contains a time range like ‘15:30-16:30’, modifying the first time will also shift the second, shifting the time block with constant length. To change the length, modify the second time. Note that if the cursor is in a headline and not at a timestamp, these same keys modify the priority of an item. (see Section 5.4 [Priorities], page 55). The key bindings also conflict with shift-selection and related modes (see Section 15.10.2 [Conflicts], page 239).</p> |                                      |
| <i>C-c C-y</i>  | <code>org-evaluate-time-range</code> |
| <p>開始日時と終了日時の差を計算することにより、時間間隔を計算します。前置引数を指定することにより、計算結果をタイムスタンプの後に挿入できます（テーブルの中では隣の列に挿入されます）。</p>   |                                      |

### 8.2.1 The date/time prompt

When Org mode prompts for a date/time, the default is shown in default date/time format, and the prompt therefore seems to ask for a specific format. But it will in fact accept date/time information in a variety of formats. Generally, the information should start at the beginning of the string. Org mode will find whatever information is in there and derive anything you have not specified from the *default date and time*. The default is usually the current date and time, but when modifying an existing timestamp, or when entering the second stamp of a range, it is taken from the stamp in the buffer. When filling in information, Org mode assumes that most of the time you will want to enter a date in the future: if you omit the month/year and the given day/month is *before* today, it will assume that you mean a future date<sup>4</sup>. If the date has been automatically shifted into the future, the time prompt will show this with ‘(=>F).’

For example, let’s assume that today is **June 13, 2006**. Here is how various inputs will be interpreted, the items filled in by Org mode are in **bold**.

|              |   |
|--------------|---|
| 3-2-5        | ⇒ 2003-02-05                                    |
| 2/5/3        | ⇒ 2003-02-05                                    |
| 14           | ⇒ <b>2006-06-14</b>                             |
| 12           | ⇒ <b>2006-07-12</b>                             |
| 2/5          | ⇒ <b>2007-02-05</b>                             |
| Fri          | ⇒ nearest Friday after the default date         |
| sep 15       | ⇒ <b>2006-09-15</b>                             |
| feb 15       | ⇒ <b>2007-02-15</b>                             |
| sep 12 9     | ⇒ 2009-09-12                                    |
| 12:45        | ⇒ <b>2006-06-13</b> 12:45                       |
| 22 sept 0:34 | ⇒ <b>2006-09-22</b> 00:34                       |
| w4           | ⇒ ISO week four of the current year <b>2006</b> |
| 2012 w4 fri  | ⇒ Friday of ISO week 4 in 2012                  |
| 2012-w04-5   | ⇒ Same as above                                 |

<sup>4</sup> See the variable `org-read-date-prefer-future`. You may set that variable to the symbol `time` to even make a time before now shift the date to tomorrow.

Furthermore you can specify a relative date by giving, as the *first* thing in the input: a plus/minus sign, a number and a letter ([hdwmy]) to indicate change in hours, days, weeks, months, or years. With a single plus or minus, the date is always relative to today. With a double plus or minus, it is relative to the default date. If instead of a single letter, you use the abbreviation of day name, the date will be the Nth such day, e.g.:

```
+0           ⇒ today
.           ⇒ today
+4d         ⇒ four days from today
+4          ⇒ same as above
+2w         ⇒ two weeks from today
++5         ⇒ five days from default date
+2tue       ⇒ second Tuesday from now
-wed        ⇒ last Wednesday
```

この機能では、英語の月および曜日の省略記法に対応しています。省略しない記法や他の言語の記法を使用したい場合には、変数 `parse-time-months` および `parse-time-weekdays` を変更して下さい。

Not all dates can be represented in a given Emacs implementation. By default Org mode forces dates into the compatibility range 1970–2037 which works on all Emacs implementations. If you want to use dates outside of this range, read the docstring of the variable `org-read-date-force-compatible-dates`.

You can specify a time range by giving start and end times or by giving a start time and a duration (in HH:MM format). Use one or two dash(es) as the separator in the former case and use '+' as the separator in the latter case, e.g.:

```
11am-1:15pm ⇒ 11:00-13:15
11am--1:15pm ⇒ 上と同様
11am+2:15   ⇒ 上と同様
```

ミニバッファのプロンプトと並行して、カレンダーがポップアップします<sup>5</sup>。カレンダー内の日付をクリックするか、RETを入力することにより日付のプロンプトを抜けると、カレンダーで選択した日付とプロンプトで入力された情報が組み合わせられます。カレンダーはミニバッファから自由に操作することができます。

```
RET          Choose date at cursor in calendar.
mouse-1      Select date by clicking on it.
S-RIGHT/LEFT One day forward/backward.
S-DOWN/UP    One week forward/backward.
M-S-RIGHT/LEFT One month forward/backward.
> / <        Scroll calendar forward/backward by one month.
M-v / C-v    Scroll calendar forward/backward by 3 months.
M-S-DOWN/UP  Scroll calendar forward/backward by one year.
```

The actions of the date/time prompt may seem complex, but I assure you they will grow on you, and you will start getting annoyed by pretty much any other way of entering a date/time out there. To help you understand what is going on, the current interpretation of your input will be displayed live in the minibuffer<sup>6</sup>.

<sup>5</sup> カレンダーの表示が不要の場合、変数 `org-popup-calendar-for-date-prompt` を変更して下さい。

<sup>6</sup> If you find this distracting, turn the display off with `org-read-date-display-live`.

### 8.2.2 Custom time format

Org mode uses the standard ISO notation for dates and times as it is defined in ISO 8601. If you cannot get used to this and require another representation of date and time to keep you happy, you can get it by customizing the options `org-display-custom-times` and `org-time-stamp-custom-formats`.

`C-c C-x C-t` `org-toggle-time-stamp-overlays`  
 カスタムフォーマットの日付および時間の表示をトグルします。

Org mode needs the default format for scanning, so the custom date/time format does not *replace* the default format—instead it is put *over* the default format using text properties. This has the following consequences:

- タイムスタンプの上にカーソルを置くことはできなくなり、その前後にしかカーソルが移動しなくなります。
- The *S-UP/DOWN* keys can no longer be used to adjust each component of a timestamp. If the cursor is at the beginning of the stamp, *S-UP/DOWN* will change the stamp by one day, just like *S-LEFT/RIGHT*. At the end of the stamp, the time will be changed by one minute.
- タイムスタンプが時間間隔や繰り返し時刻を含む場合、これらは上書きされずに元の形式のままバッファに表示されます。
- タイムスタンプを一文字ずつ消去した場合、(隠れた)ISO 標準フォーマット文字列の全てを削除した場合のみカスタムフォーマットのタイムスタンプが消去されます。
- もし、カスタムフォーマットのタイムスタンプが標準フォーマットより長く、テーブル内で用いられている場合、テーブルの整形が崩れます。標準フォーマットより短い場合には、期待通りに動作します。

## 8.3 Deadlines and scheduling

A timestamp may be preceded by special keywords to facilitate planning. Both the timestamp and the keyword have to be positioned immediately after the task they refer to.

### DEADLINE

意味:タスク (多くの場合は TODO アイテムですが、それに限りません) はタイムスタンプで示された日のうちに終了するものと見なされます。

On the deadline date, the task will be listed in the agenda. In addition, the agenda for *today* will carry a warning about the approaching or missed deadline, starting `org-deadline-warning-days` before the due date, and continuing until the entry is marked DONE. An example:

```
*** TODO write article about the Earth for the Guide
    DEADLINE: <2004-02-29 Sun>
    The editor in charge is [[bbdb:Ford Prefect]]
```

You can specify a different lead time for warnings for a specific deadline using the following syntax. Here is an example with a warning period of 5 days `DEADLINE: <2004-02-29 Sun -5d>`. This warning is deactivated if the task gets scheduled and you set `org-agenda-skip-deadline-prewarning-if-scheduled` to `t`.



**SCHEDULED**

意味:指定された日に、そのタスクに取りかかる予定であることを示します。

The headline will be listed under the given date<sup>7</sup>. In addition, a reminder that the scheduled date has passed will be present in the compilation for *today*, until the entry is marked DONE, i.e., the task will automatically be forwarded until completed.

\*\*\* TODO トリリアンに大晦日のデートについて電話する。

SCHEDULED: <2004-12-25 Sat>

If you want to *delay* the display of this task in the agenda, use **SCHEDULED: <2004-12-25 Sat -2d>**: the task is still scheduled on the 25th but will appear two days later. In case the task contains a repeater, the delay is considered to affect all occurrences; if you want the delay to only affect the first scheduled occurrence of the task, use **--2d** instead. See **org-scheduled-delay-days** and **org-agenda-skip-scheduled-delay-if-deadline** for details on how to control this globally or per agenda.

**Important:** Scheduling an item in Org mode should *not* be understood in the same way that we understand *scheduling a meeting*. Setting a date for a meeting is just a simple appointment, you should mark this entry with a simple plain timestamp, to get this item shown on the date where it applies. This is a frequent misunderstanding by Org users. In Org mode, *scheduling* means setting a date when you want to start working on an action item.

You may use timestamps with repeaters in scheduling and deadline entries. Org mode will issue early and late warnings based on the assumption that the timestamp represents the *nearest instance* of the repeater. However, the use of diary sexp entries like **<%(diary-float t 42)>** in scheduling and deadline timestamps is limited. Org mode does not know enough about the internals of each sexp function to issue early and late warnings. However, it will show the item on each day where the sexp entry matches.

**8.3.1 デッドラインおよびスケジュールの挿入**

The following commands allow you to quickly insert a deadline or to schedule an item:

**C-c C-d**

**org-deadline**

Insert **DEADLINE** keyword along with a stamp. Any **CLOSED** timestamp will be removed. When called with a prefix arg, an existing deadline will be removed from the entry. Depending on the variable **org-log-redeadline**<sup>8</sup>, a note will be taken when changing an existing deadline.

**C-c C-s**

**org-schedule**

Insert **SCHEDULED** keyword along with a stamp. Any **CLOSED** timestamp will be removed. When called with a prefix argument, remove the scheduling date from the entry. Depending on the variable **org-log-reschedule**<sup>9</sup>, a note will be taken when changing an existing scheduling time.

<sup>7</sup> It will still be listed on that date after it has been marked DONE. If you don't like this, set the variable **org-agenda-skip-scheduled-if-done**.

<sup>8</sup> with corresponding **#+STARTUP** keywords **logredeadline**, **lognoteredecline**, and **nologredeadline**

<sup>9</sup> with corresponding **#+STARTUP** keywords **logreschedule**, **lognotereschedule**, and **nologreschedule**

|                      |   |
|----------------------|---|
| <code>C-c / d</code> | <code>org-check-deadlines</code><br>全てのデッドラインのうち、既に過ぎているものと <code>org-deadline-warning-days</code> 以内に期限となるものを抽出したツリーを作成します。前置引数 <code>C-u</code> により、ファイル内の全てのデッドラインを表示します。前置引数で数値を指定すると、指定した分だけ先のデッドラインを表示します。例えば、 <code>C-1 C-c / d</code> とすると明日期限となる全てのデッドラインを表示します。 |
| <code>C-c / b</code> | <code>org-check-before-date</code><br>指定された日より前のデッドラインを抽出したツリーを作成します。   |
| <code>C-c / a</code> | <code>org-check-after-date</code><br>指定された日より後のデッドラインを抽出したツリーを作成します。  |

Note that `org-schedule` and `org-deadline` supports setting the date by indicating a relative time: e.g., `+1d` will set the date to the next day after today, and `-1w` will set the date to the previous week before any current timestamp.

### 8.3.2 Repeated tasks

Some tasks need to be repeated again and again. Org mode helps to organize such tasks using a so-called repeater in a `DEADLINE`, `SCHEDULED`, or plain timestamp. In the following example

```
** TODO 家賃の支払い
   DEADLINE: <2005-10-01 Sat +1m>
```

the `+1m` is a repeater; the intended interpretation is that the task has a deadline on `<2005-10-01>` and repeats itself every (one) month starting from that time. You can use yearly, monthly, weekly, daily and hourly repeat cookies by using the `y/w/m/d/h` letters. If you need both a repeater and a special warning period in a deadline entry, the repeater should come first and the warning period last: `DEADLINE: <2005-10-01 Sat +1m -3d>`.

Deadlines and scheduled items produce entries in the agenda when they are over-due, so it is important to be able to mark such an entry as completed once you have done so. When you mark a `DEADLINE` or a `SCHEDULED` with the `TODO` keyword `DONE`, it will no longer produce entries in the agenda. The problem with this is, however, that then also the *next* instance of the repeated entry will not be active. Org mode deals with this in the following way: When you try to mark such an entry `DONE` (using `C-c C-t`), it will shift the base date of the repeating timestamp by the repeater interval, and immediately set the entry state back to `TODO`<sup>10</sup>. In the example above, setting the state to `DONE` would actually switch the date like this:

```
** TODO 家賃の支払い
   DEADLINE: <2005-11-01 Tue +1m>
```

To mark a task with a repeater as `DONE`, use `C-- 1 C-c C-t` (i.e., `org-todo` with a numeric prefix argument of `-1`.)

<sup>10</sup> In fact, the target state is taken from, in this sequence, the `REPEAT_TO_STATE` property or the variable `org-todo-repeat-to-state`. If neither of these is specified, the target state defaults to the first state of the `TODO` state sequence.

デッドラインの下にタイムスタンプが追加され<sup>11</sup>、これにより前の時刻のデッドラインについて実際に行動したことが記録されます。

日付がシフトされた結果として、このエントリはアジェンダ上の過去の日付からは見えなくなりますが、将来の日付はアジェンダ上で確認することができます。

With the ‘+1m’ cookie, the date shift will always be exactly one month. So if you have not paid the rent for three months, marking this entry DONE will still keep it as an overdue deadline. Depending on the task, this may not be the best way to handle it. For example, if you forgot to call your father for 3 weeks, it does not make sense to call him 3 times in a single day to make up for it. Finally, there are tasks like changing batteries which should always repeat a certain time *after* the last time you did it. For these tasks, Org mode has special repeaters ‘++’ and ‘.+’. For example:

```
** TODO Call Father
  DEADLINE: <2008-02-10 Sun ++1w>
  Marking this DONE will shift the date by at least one week,
  but also by as many weeks as it takes to get this date into
  the future. However, it stays on a Sunday, even if you called
  and marked it done on Saturday.

** TODO Empty kitchen trash
  DEADLINE: <2008-02-08 Fri 20:00 ++1d>
  Marking this DONE will shift the date by at least one day, and
  also by as many days as it takes to get the timestamp into the
  future. Since there is a time in the timestamp, the next
  deadline in the future will be on today's date if you
  complete the task before 20:00.

** TODO Check the batteries in the smoke detectors
  DEADLINE: <2005-11-01 Tue .+1m>
  Marking this DONE will shift the date to one month after
  today.
```

You may have both scheduling and deadline information for a specific task. If the repeater is set for the scheduling information only, you probably want the repeater to be ignored after the deadline. If so, set the variable `org-agenda-skip-scheduled-if-deadline-is-shown` to `repeated-after-deadline`. However, any scheduling information without a repeater is no longer relevant once the task is done, and thus, removed upon repeating the task. If you want both scheduling and deadline information to repeat after the same interval, set the same repeater for both timestamps.

繰り返し演算子を使わない代替的な方法の一つとして、タスクサブツリーのコピーをいくつか作成し、それぞれのコピーに対してシフトされた時刻を指定する方法があります。そのために `C-c C-x c` コマンドがあります。この機能は `<undefined>` [見出し位置と階層の変更], page `<undefined>` で解説されています。

<sup>11</sup> この部分の動作は、オプション `org-log-repeat`、あるいは `#+STARTUP` オプションの `logrepeat`、`lognoterepeat`、`nologrepeat` により変更することができます。 `lognoterepeat` を指定した場合には、メモを入力するように促されます。

## 8.4 Clocking work time

Org mode allows you to clock the time you spend on specific tasks in a project. When you start working on an item, you can start the clock. When you stop working on that task, or when you mark the task done, the clock is stopped and the corresponding time interval is recorded. It also computes the total time spent on each subtree<sup>12</sup> of a project. And it remembers a history of tasks recently clocked, so that you can jump quickly between a number of tasks absorbing your time.

Emacs セッションでの経過時間の履歴を保存するためには、以下のコマンドを使います。

```
(setq org-clock-persist 'history)
(org-clock-persistence-insinuate)
```

Emacs の再開後に新しいタスクの計測を始めると、不完全な時計<sup>13</sup>が表示され (see `<undefined>` [空き時間の解決], page `<undefined>`)、それについて何をするかを入力するように促されます。

### 8.4.1 Clocking commands

**C-c C-x C-i** **org-clock-in**

!Start the clock on the current item (clock-in). This inserts the **CLOCK** keyword together with a timestamp. If this is not the first clocking of this item, the multiple **CLOCK** lines will be wrapped into a **:LOGBOOK:** drawer (see also the variable **org-clock-into-drawer**). You can also overrule the setting of this variable for a subtree by setting a **CLOCK\_INTO\_DRAWER** or **LOG\_INTO\_DRAWER** property. When called with a **C-u** prefix argument, select the task from a list of recently clocked tasks. With two **C-u C-u** prefixes, clock into the task at point and mark it as the default task; the default task will then always be available with letter **d** when selecting a clocking task. With three **C-u C-u C-u** prefixes, force continuous clocking by starting the clock when the last clock stopped. While the clock is running, the current clocking time is shown in the mode line, along with the title of the task. The clock time shown will be all time ever clocked for this task and its children. If the task has an effort estimate (see Section 8.5 [Effort estimates], page 87), the mode line displays the current clocking time against it<sup>14</sup>. If the task is a repeating one (see Section 8.3.2 [Repeated tasks], page 79), only the time since the last reset of the task<sup>15</sup> will be shown. More control over what time is shown can be exercised with the **CLOCK\_MODELINE\_TOTAL** property. It may have the values **current** to show only the current clocking instance, **today** to show all time clocked on this task today (see also the variable **org-extend-today-until**), **all** to include all time, or **auto** which is the default<sup>16</sup>. Clicking with **mouse-1** onto the mode line entry will pop up a menu with clocking options.

<sup>12</sup> Clocking only works if all headings are indented with less than 30 stars. This is a hardcoded limitation of **lmax** in **org-clock-sum**.

<sup>13</sup> Emacs の外でタスクに取りかかっていたという想定で時計を再開する場合は、`(setq org-clock-persist t)` を使用して下さい。

<sup>14</sup> To add an effort estimate “on the fly”, hook a function doing this to **org-clock-in-prepare-hook**.

<sup>15</sup> as recorded by the **LAST\_REPEAT** property

<sup>16</sup> See also the variable **org-clock-modeline-total**.

- C-c C-x C-o** **org-clock-out**  
 Stop the clock (clock-out). This inserts another timestamp at the same location where the clock was last started. It also directly computes the resulting time and inserts it after the time range as ‘=> HH:MM’. See the variable **org-log-note-clock-out** for the possibility to record an additional note together with the clock-out timestamp<sup>17</sup>.
- C-c C-x C-x** **org-clock-in-last**  
 Reclock the last clocked task. With one **C-u** prefix argument, select the task from the clock history. With two **C-u** prefixes, force continuous clocking by starting the clock when the last clock stopped.
- C-c C-x C-e** **org-clock-modify-effort-estimate**  
 Update the effort estimate for the current clock task.
- C-c C-c** or **C-c C-y** **org-evaluate-time-range**  
 Recompute the time interval after changing one of the timestamps. This is only necessary if you edit the timestamps directly. If you change them with **S-cursor** keys, the update is automatic.
- C-S-up/down** **org-clock-timestamps-up/down**  
 On **CLOCK** log lines, increase/decrease both timestamps so that the clock duration keeps the same.
- S-M-up/down** **org-timestamp-up/down**  
 On **CLOCK** log lines, increase/decrease the timestamp at point and the one of the previous (or the next clock) timestamp by the same duration. For example, if you hit **S-M-UP** to increase a clocked-out timestamp by five minutes, then the clocked-in timestamp of the next clock will be increased by five minutes.
- C-c C-t** **org-todo**  
 Changing the TODO state of an item to DONE automatically stops the clock if it is running in this same item.
- C-c C-x C-q** **org-clock-cancel**  
 Cancel the current clock. This is useful if a clock was started by mistake, or if you ended up working on something else.
- C-c C-x C-j** **org-clock-goto**  
 Jump to the headline of the currently clocked in task. With a **C-u** prefix arg, select the target task from a list of recently clocked tasks.
- C-c C-x C-d** **org-clock-display**  
 Display time summaries for each subtree in the current buffer. This puts overlays at the end of each headline, showing the total time recorded under that heading, including the time of any subheadings. You can use visibility cycling to study the tree, but the overlays disappear when you change the buffer (see variable **org-remove-highlights-with-change**) or press **C-c C-c**.

The **l** key may be used the agenda (see Section 10.3.1 [Weekly/daily agenda], page 104) to show which tasks have been worked on or closed during a day.

---

<sup>17</sup> The corresponding in-buffer setting is: **#+STARTUP: lognoteclock-out**

**Important:** note that both `org-clock-out` and `org-clock-in-last` can have a global key binding and will not modify the window disposition.

### 8.4.2 The clock table

Org-mode は、時間計測の情報をもとにかなり詳細なレポートを作成することができます。このようなレポートはクロックテーブルと呼ばれており、その名の通り Org-mode のテーブルまたはその組み合わせにより作成されます。

`C-c C-x C-r` `org-clock-report`  
 Insert a dynamic block (see Section A.7 [Dynamic blocks], page 248) containing a clock report as an Org mode table into the current file. When the cursor is at an existing clock table, just update it. When called with a prefix argument, jump to the first clock report in the current document and update it. The clock table always includes also trees with `:ARCHIVE:` tag.

`C-c C-c` or `C-c C-x C-u` `org-dblock-update`  
 Update dynamic block at point.

`C-u C-c C-x C-u`  
 Update all dynamic blocks (see Section A.7 [Dynamic blocks], page 248). This is useful if you have several clock table blocks in a buffer.

`S-LEFT`

`S-RIGHT` `org-clocktable-try-shift`  
 Shift the current `:block` interval and update the table. The cursor needs to be in the `#+BEGIN: clocktable` line for this command. If `:block` is `today`, it will be shifted to `today-1` etc.

以下に、`C-c C-x C-r` コマンドによりバッファに挿入されるクロックテーブルのフレームの例を示します。

```
#+BEGIN: clocktable :maxlevel 2 :emphasize nil :scope file
#+END: clocktable
```

The ‘BEGIN’ line specifies a number of options to define the scope, structure, and formatting of the report. Defaults for all these options can be configured in the variable `org-clocktable-defaults`.

まず、時間が計測されたエントリのうちどれが選択されるかを定めるオプションがあります。

|                                   |  |
|-----------------------------------|--|
| <code>:maxlevel</code>            | Maximum level depth to which times are listed in the table. Clocks at deeper levels will be summed into the upper level. |
| <code>:scope</code>               | The scope to consider. This can be any of the following:   |
| <code>nil</code>                  | the current buffer or narrowed region  |
| <code>file</code>                 | the full current buffer  |
| <code>subtree</code>              | the subtree where the clocktable is located  |
| <code>treeN</code>                | the surrounding level <i>N</i> tree, for example <code>tree3</code>  |
| <code>tree</code>                 | the surrounding level 1 tree   |
| <code>agenda</code>               | all agenda files   |
| <code>("file"..)</code>           | scan these files   |
| <code>function</code>             | the list of files returned by a function of no argument  |
| <code>file-with-archives</code>   | current file and its archives  |
| <code>agenda-with-archives</code> | all agenda files, including archives   |

|                         |   |
|-------------------------|---|
| <code>:block</code>     | The time block to consider. This block is specified either absolutely, or relative to the current time and may be any of these formats:<br>2007-12-31      New year eve 2007<br>2007-12          December 2007<br>2007-W50        ISO-week 50 in 2007<br>2007-Q2         2nd quarter in 2007<br>2007            the year 2007<br>today, yesterday, today- <i>N</i> a relative day<br>thisweek, lastweek, thisweek- <i>N</i> a relative week<br>thismonth, lastmonth, thismonth- <i>N</i> a relative month<br>thisyear, lastyear, thisyear- <i>N</i> a relative year<br>untilnow<br>Use <i>S-LEFT/RIGHT</i> keys to shift the time interval. |
| <code>:tstart</code>    | A time string specifying when to start considering times. Relative times like "<-2w>" can also be used. See Section 10.3.3 [Matching tags and properties], page 107 for relative time syntax.   |
| <code>:tend</code>      | A time string specifying when to stop considering times. Relative times like "<now>" can also be used. See Section 10.3.3 [Matching tags and properties], page 107 for relative time syntax.  |
| <code>:wstart</code>    | The starting day of the week. The default is 1 for monday.  |
| <code>:mstart</code>    | The starting day of the month. The default 1 is for the first day of the month.   |
| <code>:step</code>      | week or day, to split the table into chunks.<br>To use this, <code>:block</code> or <code>:tstart</code> , <code>:tend</code> are needed.   |
| <code>:stepskip0</code> | Do not show steps that have zero time.  |
| <code>:fileskip0</code> | Do not show table sections from files which did not contribute.   |
| <code>:tags</code>      | A tags match to select entries that should contribute. See Section 10.3.3 [Matching tags and properties], page 107 for the match syntax.  |

Then there are options which determine the formatting of the table. These options are interpreted by the function `org-clocktable-write-default`, but you can specify your own function using the `:formatter` parameter.

|                         |  |
|-------------------------|--|
| <code>:emphasize</code> | When <code>t</code> , emphasize level one and level two items.   |
| <code>:lang</code>      | Language <sup>18</sup> to use for descriptive cells like "Task".   |
| <code>:link</code>      | Link the item headlines in the table to their origins.   |
| <code>:narrow</code>    | An integer to limit the width of the headline column in the org table. If you write it like '50!', then the headline will also be shortened in export. |
| <code>:indent</code>    | Indent each headline field according to its level.   |
| <code>:tcolumns</code>  | Number of columns to be used for times. If this is smaller than <code>:maxlevel</code> , lower levels will be lumped into one column.                  |
| <code>:level</code>     | Should a level number column be included?  |
| <code>:sort</code>      | A cons cell like containing the column to sort and a sorting type. E.g., <code>:sort (1 . ?a)</code> sorts the first column alphabetically.            |
| <code>:compact</code>   | Abbreviation for <code>:level nil :indent t :narrow 40! :tcolumns 1</code>   |

<sup>18</sup> Language terms can be set through the variable `org-clock-clocktable-language-setup`.

All are overwritten except if there is an explicit `:narrow`

- `:timestamp` A timestamp for the entry, when available. Look for SCHEDULED, DEADLINE, TIMESTAMP and TIMESTAMP\_IA, in this order.
- `:properties` List of properties that should be shown in the table. Each property will get its own column.
- `:inherit-props` When this flag is t, the values for `:properties` will be inherited.
- `:formula` Content of a `#+TBLFM` line to be added and evaluated.  
As a special case, `:formula %` adds a column with % time.  
If you do not specify a formula here, any existing formula below the clock table will survive updates and be evaluated.
- `:formatter` A function to format clock data and insert it into the buffer.

現在のレベル1のツリーについて、当日分の時間サマリーを得る場合は以下のように指定します。

```
#+BEGIN: clocktable :maxlevel 2 :block today :scope tree1 :link t
#+END: clocktable
```

明示的に時間間隔を指定する場合には、以下のように記述します<sup>19</sup>。

```
#+BEGIN: clocktable :tstart "<2006-08-10 Thu 10:00>"
                  :tend "<2006-08-10 Thu 12:00>"
#+END: clocktable
```

A range starting a week ago and ending right now could be written as

```
#+BEGIN: clocktable :tstart "<-1w>" :tend "<now>"
#+END: clocktable
```

現在のサブツリーでの経過時間のまとめを%表示するには、以下のように記述します。

```
#+BEGIN: clocktable :scope subtree :link t :formula %
#+END: clocktable
```

ここ1週間で計測された時間をコンパクトな幅で表示するには、以下のようにします。

```
#+BEGIN: clocktable :scope agenda :block lastweek :compact t
#+END: clocktable
```

### 8.4.3 Resolving idle time and continuous clocking

#### Resolving idle time

ある項目について作業を開始したあとで、例えば電話を取る場合などで一時的にコンピュータの前を離れると、その時間について現在の経過時間から差し引いたり、他の項目に加えたりして「解決」する必要が生じます。

By customizing the variable `org-clock-idle-time` to some integer, such as 10 or 15, Emacs can alert you when you get back to your computer after being idle for that many minutes<sup>20</sup>, and ask what you want to do with the idle time. There will be a question waiting

<sup>19</sup> 全てのパラメーターは単一行で指定する必要があるので注意して下さい。この例ではマニュアルの文字幅の制約のために改行が入っています。

<sup>20</sup> On computers using macOS, idleness is based on actual user idleness, not just Emacs' idle time. For X11, you can install a utility program `x11idle.c`, available in the `contrib/scripts` directory of the Org git distribution, or install the `xprintidle` package and set it to the variable `org-clock-x11idle-program-name` if you are running Debian, to get the same general treatment of idleness. On other systems, idle time refers to Emacs idle time only.



for you when you get back, indicating how much idle time has passed (constantly updated with the current amount), as well as a set of choices to correct the discrepancy:

- k** To keep some or all of the minutes and stay clocked in, press **k**. Org will ask how many of the minutes to keep. Press **RET** to keep them all, effectively changing nothing, or enter a number to keep that many minutes.
- K** シフトキーと共に **K**を押した場合、入力された分数だけ時間を保持すると同時にただちに現在のタスクの計測を中止します。全ての休止時間を保持する場合、これは単にタスクの計測を中止したのと同じことになります。
- s** 休止時間を保持しない場合には、**s**を押すことにより計測時間から全ての休止時間が差し引かれ、戻ってきた時点から再開されます。
- S** 休止時間を保持せず、休止開始時の時刻で時間の計測を止める場合には、シフトキーと共に **S**を押して下さい。シフトキーを使うと、いずれのオプションでも時間の計測が中止されるということを覚えておいて下さい。
- C** 時間計測そのものをキャンセルする場合は、**C**を押して下さい。キャンセルしない場合でも、時間が引かれた結果の残り時間が1分未満である場合には、中身の無いエントリでログが見づらくなるのを防ぐため、時間計測はやはりキャンセルされますので注意して下さい。

空き時間について、現在の計測時間から差し引いたあとで別の計測項目に追加したい場合にはどうすれば良いでしょうか。その場合は、差し引いたあとに単純に次のタスクの計測を開始して下さい。Org-modeは差し引かれた時間があることを記憶していて、次の時間計測を始める際にその時間を足し込むかを尋ねます。

次のようなケースでも、時間解決機能が魔法のような働きをします。あなたがタスクの時間計測をしながらご機嫌に作業をすすめていると、突然飼い猫がネズミを追いかけて、それを見たハムスターが驚いてUPSの電源装置に衝突してしまったとしましょう!あなたは全てのバッファを失うこととなりますが、オートセーブ機能のおかげでOrg-modeで行った最近の変更は保持され、途中であった時間計測の時間も保持されています。

If you restart Emacs and clock into any task, Org will notice that you have a dangling clock which was never clocked out from your last session. Using that clock's starting time as the beginning of the unaccounted-for period, Org will ask how you want to resolve that time. The logic and behavior is identical to dealing with away time due to idleness; it is just happening due to a recovery event rather than a set amount of idle time.

You can also check all the files visited by your Org agenda for dangling clocks at any time using *M-x org-resolve-clocks RET* (or *C-c C-x C-z*).

## Continuous clocking

You may want to start clocking from the time when you clocked out the previous task. To enable this systematically, set `org-clock-continuously` to `t`. Each time you clock in, Org retrieves the clock-out time of the last clocked entry for this session, and start the new clock from there.

If you only want this from time to time, use three universal prefix arguments with `org-clock-in` and two *C-u C-u* with `org-clock-in-last`.

## 8.5 Effort estimates

If you want to plan your work in a very detailed way, or if you need to produce offers with quotations of the estimated work effort, you may want to assign effort estimates to entries. If you are also clocking your work, you may later want to compare the planned effort with the actual working time, a great way to improve planning estimates. Effort estimates are stored in a special property **EFFORT**. You can set the effort for an entry with the following commands:

**C-c C-x e** **org-set-effort**  
 現在のエントリについて工数の見積もりを行います。前置引数に数値を指定することにより、N 番目の数値に指定します (下記の例を参照)。このコマンドはアジェンダからも **e**キーを押すことによりアクセスできます。

**C-c C-x C-e** **org-clock-modify-effort-estimate**  
 現在時間が計測されている項目の工数見積もりを変更します。

明らかなように、工数見積もりを行う最善の方法はカラムビュー (see [\(undefined\)](#) [カラムビュー], page [\(undefined\)](#)) を用いることです。個別の項目についての工数見積もりから始めて、**COLUMNS** フォーマットによりこれらの値と実際の計測時間 (時間の計測を行いたい場合) を同時に表示します。例えば、あるバッファについて以下のように指定できます。

```
#+PROPERTY: Effort_ALL 0 0:10 0:30 1:00 2:00 3:00 4:00 5:00 6:00 7:00
#+COLUMNS: %40ITEM(Task) %17Effort(Estimated Effort){:} %CLOCKSUM
```

さらに良い方法としては、変数 **org-global-properties**あるいは **org-columns-default-format**をカスタマイズすることにより、これらの数値をグローバルに指定できます。特にこの指定をアジェンダで使いたい場合には、グローバルな指定を行うことが推奨されます。

The way to assign estimates to individual items is then to switch to column mode, and to use **S-RIGHT** and **S-LEFT** to change the value. The values you enter will immediately be summed up in the hierarchy. In the column next to it, any clocked time will be displayed.

日別あるいは週別のアジェンダでカラムビューに移行すると、工数の列は各日についての工数見積もりを足し合わせて表示され<sup>21</sup>、これを用いてスケジュールの空きを見つけることができます。その日の作業について全体像をつかみたい場合には、オプション **org-agenda-columns-add-appointments-to-effort-sum**を指定することができます。時間間隔が指定されているアポイントで、その日に発生するものについても負荷見積もりに加算して表示されます。

工数見積もりは、アジェンダ内で/を押すことによりアジェンダの2次的なフィルタリングに用いることができます (see [\(undefined\)](#) [アジェンダのコマンド], page [\(undefined\)](#))。このような見積もりを確実に行えば、2、3回キーを押すことにより空いている時間間隔に合うように項目を絞り込むことができます。

## 8.6 Taking notes with a timer

Org provides two types of timers. There is a relative timer that counts up, which can be useful when taking notes during, for example, a meeting or a video viewing. There is also a countdown timer.

The relative and countdown are started with separate commands.

<sup>21</sup> 単なるリストを階層的に足し合わせる際には落とし穴があります (see [\(undefined\)](#) [アジェンダのカラム表示], page [\(undefined\)](#))。

***C-c C-x 0*** **org-timer-start**  
 Start or reset the relative timer. By default, the timer is set to 0. When called with a *C-u* prefix, prompt the user for a starting offset. If there is a timer string at point, this is taken as the default, providing a convenient way to restart taking notes after a break in the process. When called with a double prefix argument *C-u C-u*, change all timer strings in the active region by a certain amount. This can be used to fix timer strings if the timer was not started at exactly the right moment.

***C-c C-x ;*** **org-timer-set-timer**  
 Start a countdown timer. The user is prompted for a duration. **org-timer-default-timer** sets the default countdown value. Giving a numeric prefix argument overrides this default value. This command is available as *;* in agenda buffers.

Once started, relative and countdown timers are controlled with the same commands.

***C-c C-x .*** **org-timer**  
 Insert the value of the current relative or countdown timer into the buffer. If no timer is running, the relative timer will be started. When called with a prefix argument, the relative timer is restarted.

***C-c C-x -*** **org-timer-item**  
 Insert a description list item with the value of the current relative or countdown timer. With a prefix argument, first reset the relative timer to 0.

***M-RET*** **org-insert-heading**  
 Once the timer list is started, you can also use *M-RET* to insert new timer items.

***C-c C-x ,*** **org-timer-pause-or-continue**  
 Pause the timer, or continue it if it is already paused.

***C-c C-x \_*** **org-timer-stop**  
 Stop the timer. After this, you can only start a new timer, not continue the old one. This command also removes the timer from the mode line.

## 9 Capture - Refile - Archive

管理システムにおける重要な点の一つとして、新しいアイデアやタスクを素早くキャプチャし、それらを参考資料と連携する能力があることです。Org-mode ではキャプチャと呼ばれるプロセスを用いて行います。また、タスクに関係するファイル (*attachments*) を特別なディレクトリに保存することも可能です。一度システムへ取り込むと、タスクとプロジェクトを移動させる必要があります。完了したプロジェクトツリーをアーカイブファイルへ移動することで、システムをコンパクトで速く保つことが可能です。

### 9.1 Capture

Capture lets you quickly store notes with little interruption of your work flow. Org's method for capturing new items is heavily inspired by John Wiegley excellent `remember.el` package. Up to version 6.36, Org used a special setup for `remember.el`, then replaced it with `org-remember.el`. As of version 8.0, `org-remember.el` has been completely replaced by `org-capture.el`.

If your configuration depends on `org-remember.el`, you need to update it and use the setup described below. To convert your `org-remember-templates`, run the command

```
M-x org-capture-import-remember-templates RET
```

and then customize the new variable with `M-x customize-variable org-capture-templates`, check the result, and save the customization.

#### 9.1.1 Setting up capture

以下のカスタマイズはノートを取るデフォルトのファイルと新しい素材をキャプチャするためのグローバルなキー<sup>1</sup>

```
(setq org-default-notes-file (concat org-directory "/notes.org"))
(define-key global-map "\C-cc" 'org-capture)
```

#### 9.1.2 Using capture

`C-c c` `org-capture`

Call the command `org-capture`. Note that this key binding is global and not active by default: you need to install it. If you have templates defined see Section 9.1.3 [Capture templates], page 90, it will offer these templates for selection or use a new Org outline node as the default template. It will insert the template into the target file and switch to an indirect buffer narrowed to this new node. You may then insert the information you want.

`C-c C-c` `org-capture-finalize`

Once you have finished entering information into the capture buffer, `C-c C-c` will return you to the window configuration before the capture process, so that you can resume your work without further distraction. When called with a prefix arg, finalize and then jump to the captured item.

`C-c C-w` `org-capture-refile`

Finalize the capture process by refileing (see Section 9.5 [Refile and copy], page 99) the note to a different place. Please realize that this is a normal

---

<sup>1</sup> 自分のキーを選択して下さい。C-c cはただの提案です。

refiling command that will be executed—so the cursor position at the moment you run this command is important. If you have inserted a tree with a parent and children, first move the cursor back to the parent. Any prefix argument given to this command will be passed on to the `org-refile` command.

**C-c C-k** **org-capture-kill**  
 Abort the capture process and return to the previous state.

`org-capture`をアジェンダから `k c` キーの組合せを用いた特別な方法で呼ぶこともできます。この方法では、選択されたキャプチャテンプレートに挿入されるタイムスタンプは、現在の日付ではなくアジェンダ内のカーソルがある位置の日付がデフォルトになります。

最後にキャプチャが保存された場所を探すには、`org-capture`をプレフィックスコマンドと一緒に使用します。

**C-u C-c c**  
 Visit the target location of a capture template. You get to select the template in the usual way.

**C-u C-u C-c c**  
 Visit the last stored capture item in its buffer.

You can also jump to the bookmark `org-capture-last-stored`, which will automatically be created unless you set `org-capture-bookmark` to `nil`.

To insert the capture at point in an Org buffer, call `org-capture` with a `C-0` prefix argument.

### 9.1.3 Capture templates

テンプレートは異なる種類のキャプチャアイテムや、異なる場所へ使用することができます。最も簡単にそのようなテンプレートを作る方法、カスタマイズインターフェースを通じて行うことです。

**C-c c C** Customize the variable `org-capture-templates`.

テンプレート定義の正式な説明の前に、例を挙げます。一般的な TODO エントリーを作成する場合を考えます。また、これらのエントリーは `~/org/gtd.org` の中にある見出し `'Tasks'` の下に作成され、`journal.org` 内のデートツリーはジャーナルエントリーがキャプチャされます。このような場合の設定は以下のようになります。

```
(setq org-capture-templates
  '(("t" "Todo" entry (file+headline "~/org/gtd.org" "Tasks")
    "* TODO %?\n %i\n %a")
    ("j" "Journal" entry (file+olp+datetree "~/org/journal.org")
    "* %?\nEntered on %U\n %i\n %a")))
```

ここで `C-c c t` を押すと、Org-mode はこのようなテンプレートを用意します。

```
* TODO
[[file:lint to where you initiated capture]]
```

During expansion of the template, `%a` has been replaced by a link to the location from where you called the capture command. This can be extremely useful for deriving tasks from emails, for example. You fill in the task definition, press `C-c C-c` and Org returns you to the same place where you started the capture process.

特定のテンプレートを対話的な選択なしに用いてキャプチャする特別なキーを定義するには、以下のようにキーバインドを作成できます。

```
(define-key global-map "\C-cx\n"
  (lambda () (interactive) (org-capture nil "x")))
```

### 9.1.3.1 Template elements

それではテンプレートを定義するための要素について述べています。`org-capture-templates`中にあるそれぞれのエントリーは以下の項目から構成されるリストです。

**keys** キーは文字列で示されるテンプレートを文字だけで選択します。例えば、`"a"`はaキーだけで選択するテンプレートであり、`"bt"`は2つのキーで選択します。幾つかのキーを使う場合、同じ接頭辞キーを持つキーはリストの中で連続している必要があり、以下に例をあげるような、接頭辞キーを表す2つの要素を持つエントリーをその前に置きます。

```
("b" "買い物リストのためのテンプレート")
```

もしCキーにテンプレートを定義していなければ、このキーでこの複雑な変数のカスタマイズバッファを開きます。

**description**

選択時に表示されるテンプレートの短い説明文。

**type**

エントリーの種類をシンボルで表します。正しい値は以下の通りです。

**entry** An Org mode node, with a headline. Will be filed as the child of the target entry or as a top-level entry. The target file should be an Org mode file.

**item** 対象の場所の最初にある単純なリストへ書き込まれる単純なリストの項目。対象のファイルはOrg-modeのファイルでなければなりません。

**checkitem**

チェックボックス項目。これは単純なリストとデフォルトのテンプレートが異なるだけです。

**table-line**

対象の場所にある最初のテーブルへ追加する新しい行。行が追加される場所は`:prepend`と`:table-line-pos`プロパティに依存します。(下記参照)

**plain** そのまま挿入される文章

**target**

Specification of where the captured item should be placed. In Org mode files, targets usually define a node. Entries will become children of this node. Other types will be added to the table or list in the body of this node. Most target specifications contain a file name. If that file name is the empty string, it defaults to `org-default-notes-file`. A file can also be given as a variable or as a function called with no argument. When an absolute path is not specified for a target, it is taken as relative to `org-directory`.

正しい値は以下の通りです。

```
(file "path/to/file")
```

文章はこのファイルの先頭か最後に挿入されます。

```
(id "現存する org エントリーの id")
```

このエントリーの子として、もしくはこのエントリの本文として記入します。

(file+headline "path/to/file" "node headline")

対象の見出しがファイル中で一つしか無い場合の速い設定。

(file+olp "path/to/file" "Level 1 heading" "Level 2" ...)

唯一でない見出しの場合、フルパスの方が安全です。

(file+regexp "path/to/file" "regexp to find location")

カーソルの位置に正規表現を使った場合。

(file+olp+datetree "path/to/file" [ "Level 1 heading" ....])

This target<sup>2</sup> will create a heading in a date tree<sup>3</sup> for today's date. If the optional outline path is given, the tree will be built under the node it is pointing to, instead of at top level. Check out the :time-prompt and :tree-type properties below for additional options.

(file+function "path/to/file" function-finding-location)

ファイルの中で正しい場所を見つける関数

(clock) ファイル中で現在時間を測っているエントリー。

(function function-finding-location)

Most general way: write your own function which both visits the file and moves point to the right location.

**template** キャプチャする項目を作るテンプレート。もし空にした場合、デフォルトのテンプレートを使用します。もしエスケープコードを付加した文字列の場合、キャプチャされた時の時間と状況に依存して置換されます。エスケープされた文字列はテンプレートファイルからロードされ、特別な構文 (file "path/to/template") を使用します。詳細をいかに示します。

## properties

エントリーの残りは追加オプションのプロパティリストです。理解できるプロパティは

**:prepend** 通常新しくキャプチャされた情報は対象の場所 (最後の子、最後のテーブル、最後のリスト項目...) に追加されます。このプロパティを設定することで変更します。

**:immediate-finish**

セットされると情報の編集を行わず、追加だけを直ちに行います。テンプレートが自動的に追加できる情報だけを必要とする場合だけ意味を成します。

**:empty-lines**

新しい項目の前後に挿入する行の数をこれに設定します。デフォルトは0で、通常の他の値は1です。

**:clock-in**

この項目の時計を開始します。

<sup>2</sup> Org used to offer four different targets for date/week tree capture. Now, Org automatically translates these to use file+olp+datetree, applying the :time-prompt and :tree-type properties. Please rewrite your date/week-tree targets using file+olp+datetree since the older targets are now deprecated.

<sup>3</sup> A date tree is an outline structure with years on the highest level, months or ISO-weeks as sublevels and then dates on the lowest level. Tags are allowed in the tree structure.

**:clock-keep**

キャプチャしたエントリーを追加しても時計を動かし続けます。

**:clock-resume**

もし時計を割り込むキャプチャを始めた場合、キャプチャ終了時に時計を再開します。**:clock-keep**は**:clock-resume**より優先されることに注意が必要です。もしどちらの設定も **t** に設定された場合、現在の時計が動き、一つ前の時計は再開されません。

**:time-prompt**

Prompt for a date/time to be used for date/week trees and when filling the template. Without this property, capture uses the current date and time. Even if this property has not been set, you can force the same behavior by calling **org-capture** with a **C-1** prefix argument.

**:tree-type**

When 'week', make a week tree instead of the month tree, i.e. place the headings for each day under a heading with the current iso week.

**:unnarrowed**

対象とするバッファの幅を狭めず、単純にフルバッファで表示します。デフォルトでは幅を狭くし、新しい内容だけが表示されます。

**:table-line-pos**

Specification of the location in the table where the new line should be inserted. It can be a string, a variable holding a string or a function returning a string. The string should look like "II-3" meaning that the new line should become the third line before the second horizontal separator line.

**:kill-buffer**

対象のファイルがキャプチャ時にまだ読み込まれていなかった場合、キャプチャ終了時に再びバッファを閉じます。

### 9.1.3.2 Template expansion

In the template itself, special %-escapes<sup>4</sup> allow dynamic insertion of content. The templates are expanded in the order given here:

|                     |  |
|---------------------|--|
| <b>%[file]</b>      | Insert the contents of the file given by <i>file</i> .   |
| <b>%(sexp)</b>      | Evaluate Emacs <i>sexp</i> and replace with the result.<br>For convenience, <b>:%keyword</b> (see below) placeholders within the expression will be expanded prior to this.<br>The <i>sexp</i> must return a string. |
| <b>%&lt;...&gt;</b> | The result of <code>format-time-string</code> on the ... format specification.   |
| <b>%t</b>           | Timestamp, date only.  |
| <b>%T</b>           | Timestamp, with date and time.   |
| <b>%u, %U</b>       | Like the above, but inactive timestamps.   |
| <b>%i</b>           | Initial content, the region when capture is called while the region is active.   |
|                     | The entire text will be indented like <b>%i</b> itself.  |
| <b>%a</b>           | Annotation, normally the link created with <b>org-store-link</b> .   |

<sup>4</sup> If you need one of these sequences literally, escape the % with a backslash.



|             |  |
|-------------|--|
| %A          | Like %a, but prompt for the description part.  |
| %l          | Like %a, but only insert the literal link.   |
| %c          | Current kill ring head.  |
| %x          | Content of the X clipboard.  |
| %k          | Title of the currently clocked task.   |
| %K          | Link to the currently clocked task.  |
| %n          | User name (taken from <code>user-full-name</code> ).   |
| %f          | File visited by current buffer when org-capture was called.  |
| %F          | Full path of the file or directory visited by current buffer.  |
| %:keyword   | Specific information for certain link types, see below.  |
| %^g         | Prompt for tags, with completion on tags in target file.   |
| %^G         | Prompt for tags, with completion all tags in all agenda files.   |
| %^t         | Like %t, but prompt for date. Similarly %^T, %^u, %^U.<br>You may define a prompt like %^{Birthday}t.  |
| %^C         | Interactive selection of which kill or clip to use.  |
| %^L         | Like %^C, but insert as link.  |
| %^{prop}p   | Prompt the user for a value for property <i>prop</i> .   |
| %^{prompt}  | prompt the user for a string and replace this sequence with it.<br>You may specify a default value and a completion table with<br>%^{prompt default completion2 completion3...}.<br>The arrow keys access a prompt-specific history. |
| %\1 ... %\N | Insert the text entered at the Nth %^{prompt}, where N is<br>a number, starting from 1. <sup>5</sup>   |
| %?          | After completing the template, position cursor here.   |

リンクのタイプを指定するため、以下のキーワードが定義<sup>6</sup>されています。

| Link type                        | Available keywords  |
|----------------------------------|---|
| bbdb                             | :name  :company   |
| irc                              | :server  :port  :nick   |
| vm, vm-imap, wl, mh, mew, rmail, | :type  :subject  :message-id  |
| gnus, notmuch                    | :from  :fromname  :fromaddress<br> :to  :toname  :toaddress<br> :date (message date header field)<br> :date-timestamp (date as active timestamp)<br> :date-timestamp-inactive (date as inactive timestamp)<br> :fromto (either "to NAME" or "from NAME") <sup>7</sup> |
| gnus                             | :group, for messages also all email fields  |
| eww, w3, w3m                     | :url  |
| info                             | :file  :node  |
| calendar                         | :date   |
| org-protocol                     | :link  :description  :annotation  |

テンプレート展開後のカーソル位置のために以下を用います。

%?           テンプレートを完了したあと、カーソル位置をここに移動します。

### 9.1.3.3 Templates in contexts

To control whether a capture template should be accessible from a specific context, you can customize `org-capture-templates-contexts`. Let's say for example that you have a

<sup>5</sup> As required in Emacs Lisp, it is necessary to escape any backslash character in a string with another backslash. So, in order to use '%\1' placeholder, you need to write '%\\1' in the template.

<sup>6</sup> もしあなたが自分のリンクタイプを定義しているならば (see (undefined) [ハイパーリンクのタイプの追加], page (undefined)), `org-store-link-props`に保存されているいずれのプロパティもキャプチャテンプレート中に同じ方法でアクセス可能です。

<sup>7</sup> This will always be the other, not the user. See the variable `org-from-is-user-regexp`.

capture template "p" for storing Gnus emails containing patches. Then you would configure this option like this:

```
(setq org-capture-templates-contexts
  '(("p" (in-mode . "message-mode"))))
```

You can also tell that the command key "p" should refer to another template. In that case, add this command key like this:

```
(setq org-capture-templates-contexts
  '(("p" "q" (in-mode . "message-mode"))))
```

See the docstring of the variable for more information.

## 9.2 Attachments

参照すべきものをアウトラインノートやタスクと連携させることは大抵役立ちます。小さなプレーンテキストの塊は単純にプロジェクトのサブツリーとして保存可能です。ハイパーリンク (see [\[ハイパーリンク\]](#), page [\[ハイパーリンク\]](#)) はあなたのコンピュータやクラウドなどにあるファイル、例えばプロジェクトに関連する email やソースコードファイルとの連携を確立します。その他の方法として、*attachments* があります。これはアウトラインノードの属するディレクトリにあるファイルです。これらのディレクトリは **data** ディレクトリの中に位置し、あなたの Org ファイルが保存されているディレクトリと同じディレクトリの中にある<sup>8</sup>。もしこのディレクトリを `git init` で初期化した場合、Org-mode は変更点を見つけた時に自動的にそれらをコミットします。添付システムは John Wiegley の貢献によって Org-mode へ追加されました。

もしそれがより良い方法に思える場合は、エントリーヘディレクトリを添付することも自身の選択で可能です。子エントリーは親から添付ディレクトリを受け継ぐため、サブツリー全体が同じ添付ディレクトリを使うことになります。

以下のコマンドはプロパティを操作する助けとなります。

|                |   |
|----------------|---|
| <b>C-c C-a</b> | <b>org-attach</b>   |
|                | 添付システムに関連するコマンドのコマンド選択画面。これらのキーのあとに、コマンドのリストが表示され、コマンドを選択するために更にキーを押す必要があります。   |
| <b>a</b>       | <b>org-attach-attach</b>  |
|                | ファイルを選択してそれをタスクの添付ディレクトリへ移動します。ファイルは <b>org-attach-method</b> によって複製、移動、リンクされます。ハードリンクはすべてのシステムでサポートされていないことに注意が必要です。 |
| <b>c/m/l</b>   | コピー/移動/リンクメソッドを使ってファイルを添付します。ハードリンクはすべてのシステムでサポートされていないことに注意が必要です。  |
| <b>u</b>       | <b>org-attach-url</b>   |
|                | Attach a file from URL  |
| <b>n</b>       | <b>org-attach-new</b>   |
|                | Emacs のバッファとして新しい添付を作成します。  |
| <b>z</b>       | <b>org-attach-sync</b>  |
|                | あなた自身で添付を追加した場合に、現在のタスクをその添付ディレクトリと同期します。   |

<sup>8</sup> もしエントリーや Org ファイルを他のディレクトリに移動した場合、`org-attach-directory` が絶対パスを含むように設定する必要があります。

|          |   |
|----------|---|
| <i>o</i> | <b>org-attach-open</b><br>現在のタスクに関する添付を開きます。もし 1 つ以上ある場合は、まずファイル名を入力させます。開き方は <b>org-file-apps</b> に従います。詳細は、ハイパーリンクを辿るための情報 (see <a href="#">〈undefined〉</a> [リンクの扱い], page <a href="#">〈undefined〉</a> ) を参照してください。 |
| <i>O</i> | <b>org-attach-open-in-emacs</b><br>これも添付を開きますが、強制的にファイルを Emacs を用います。   |
| <i>f</i> | <b>org-attach-reveal</b><br>現在のタスクの添付ディレクトリを開きます。   |
| <i>F</i> | <b>org-attach-reveal-in-emacs</b><br>これもディレクトリを開きますが、強制的に Emacs 内で dired を用います。   |
| <i>d</i> | <b>org-attach-delete-one</b><br>添付の 1 つを選択し削除します。   |
| <i>D</i> | <b>org-attach-delete-all</b><br>タスクに関連する添付をすべて削除します。安全な方法は dired を用いてディレクトリを開き、そこから削除する方法です。  |
| <i>s</i> | <b>org-attach-set-directory</b><br>特定のディレクトリをエントリーの添付ディレクトリに指定します。これはディレクトリのパスを <b>ATTACH_DIR</b> プロパティに代入することで動作します。   |
| <i>i</i> | <b>org-attach-set-inherit</b><br><b>ATTACH_DIR_INHERIT</b> プロパティをセットします。これによって子も親と同じディレクトリを添付として使用します。  |

### 9.3 RSS フィード

Org-mode はエントリーの追加や変更を RSS フィードと Atom フィードの情報を元に行うことができます。この機能を使ってプロダクトフィードのなかからそれぞれの新しいプロダクトに対してタスクを作ることも可能です。もしくは、携帯電話を対象とした Web 上のノート作成サービスを Org-mode のタスクにインポートできます。フィードにアクセスするには、**org-feed-alist** を設定します。この変数のドキュメント文字列に詳細があります。以下に例をします。

```
(setq org-feed-alist
  '(("Slashdot"
    "http://rss.slashdot.org/Slashdot/slashdot"
    "~/txt/org/feeds.org" "Slashdot Entries")))
```

この例は、以下のコマンドが使われるたびに、**rss.slashdot.org** で提供されているフィードの新しいアイテムから、**~/org/feeds.org** ファイル内に **'Slashdot Entries'** をヘッダーにもつ新しいエントリーを生成します。

**C-c C-x g** **org-feed-update-all**  
**C-c C-x g** Collect items from the feeds configured in **org-feed-alist** and act upon them.

**C-c C-x G** **org-feed-goto-inbox**  
Prompt for a feed name and go to the inbox configured for this feed.

Under the same headline, Org will create a drawer ‘FEEDSTATUS’ in which it will store information about the status of items in the feed, to avoid adding the same item several times.

Atom フィードの読み方などのより詳しい情報は、`org-feed.el`を御覧ください。また、`org-feed-alist`のドキュメント文字列もあります。

## 9.4 外部アクセスのためのプロトコル

Org protocol is a mean to trigger custom actions in Emacs from external applications. Any application that supports calling external programs with an URL as argument may be used with this functionality. For example, you can configure bookmarks in your web browser to send a link to the current page to Org and create a note from it using capture (see Section 9.1 [Capture], page 89). You can also create a bookmark that tells Emacs to open the local source file of a remote website you are browsing.

In order to use Org protocol from an application, you need to register ‘org-protocol://’ as a valid scheme-handler. External calls are passed to Emacs through the `emacsclient` command, so you also need to ensure an Emacs server is running. More precisely, when the application calls

```
emacsclient org-protocol://PROTOCOL?key1=val1&key2=val2
```

Emacs calls the handler associated to ‘PROTOCOL’ with argument ‘(:key1 val1 :key2 val2)’.

Org protocol comes with three predefined protocols, detailed in the following sections. Configure `org-protocol-protocol-alist` to define your own.

### 9.4.1 store-link protocol

Using `store-link` handler, you can copy links, insertable through `M-x org-insert-link` or yanking thereafter. More precisely, the command

```
emacsclient org-protocol://store-link?url=URL&title=TITLE
```

stores the following link:

```
[ [URL] [TITLE] ]
```

In addition, ‘URL’ is pushed on the kill-ring for yanking. You need to encode ‘URL’ and ‘TITLE’ if they contain slashes, and probably quote those for the shell.

To use this feature from a browser, add a bookmark with an arbitrary name, e.g., ‘Org: store-link’ and enter this as *Location*:

```
javascript:location.href='org-protocol://store-link?url='+
  encodeURIComponent(location.href);
```

### 9.4.2 capture protocol

Activating `capture` handler pops up a ‘Capture’ buffer and fills the capture template associated to the ‘X’ key with them.

```
emacsclient org-protocol://capture?template=X?url=URL?title=TITLE?body=BODY
```

To use this feature, add a bookmark with an arbitrary name, e.g. ‘Org: capture’ and enter this as ‘Location’:

```
javascript:location.href='org-protocol://capture?template=x'+
  '&url='+encodeURIComponent(window.location.href)+
  '&title='+encodeURIComponent(document.title)+
  '&body='+encodeURIComponent(window.getSelection());
```

The result depends on the capture template used, which is set in the bookmark itself, as in the example above, or in `org-protocol-default-template-key`.

The following template placeholders are available:

|                             |  |
|-----------------------------|--|
| <code>:%:link</code>        | The URL  |
| <code>:%:description</code> | The webpage title                                    |
| <code>:%:annotation</code>  | Equivalent to <code>[[[:link]][:description]]</code> |
| <code>%i</code>             | The selected text                                    |

### 9.4.3 open-source protocol

The `open-source` handler is designed to help with editing local sources when reading a document. To that effect, you can use a bookmark with the following location:

```
javascript:location.href='org-protocol://open-source?&url='+
  encodeURIComponent(location.href)
```

The variable `org-protocol-project-alist` maps URLs to local file names, by stripping URL parameters from the end and replacing the `:base-url` with `:working-directory` and `:online-suffix` with `:working-suffix`. For example, assuming you own a local copy of `https://orgmode.org/worg/` contents at `/home/user/worg`, you can set `org-protocol-project-alist` to the following

```
(setq org-protocol-project-alist
  '(("Worg"
    :base-url "https://orgmode.org/worg/"
    :working-directory "/home/user/worg/"
    :online-suffix ".html"
    :working-suffix ".org")))
```

If you are now browsing `https://orgmode.org/worg/org-contrib/org-protocol.html` and find a typo or have an idea about how to enhance the documentation, simply click the bookmark and start editing.

However, such mapping may not yield the desired results. Suppose you maintain an online store located at `http://example.com/`. The local sources reside in `/home/user/example/`. It is common practice to serve all products in such a store through one file and rewrite URLs that do not match an existing file on the server. That way, a request to `http://example.com/print/posters.html` might be rewritten on the server to something like `http://example.com/shop/products.php/posters.html.php`. The `open-source` handler probably cannot find a file named `/home/user/example/print/posters.html.php` and fails.

Such an entry in `org-protocol-project-alist` may hold an additional property `:rewrites`. This property is a list of cons cells, each of which maps a regular expression to a path relative to the `:working-directory`.

Now map the URL to the path `/home/user/example/products.php` by adding `:rewrites` rules like this:

```
(setq org-protocol-project-alist
  '(("example.com"
     :base-url "http://example.com/"
     :working-directory "/home/user/example/"
     :online-suffix ".php"
     :working-suffix ".php"
     :rewrites (("example.com/print/" . "products.php")
                ("example.com/$" . "index.php")))))
```

Since `'example.com/$'` is used as a regular expression, it maps `http://example.com/`, `https://example.com`, `http://www.example.com/` and similar to `/home/user/example/index.php`.

The `:rewrites` rules are searched as a last resort if and only if no existing file name is matched.

Two functions can help you filling `org-protocol-project-alist` with valid contents: `org-protocol-create` and `org-protocol-create-for-org`. The latter is of use if you're editing an Org file that is part of a publishing project.

## 9.5 Refile and copy

When reviewing the captured data, you may want to refile or to copy some of the entries into a different list, for example into a project. Cutting, finding the right location, and then pasting the note is cumbersome. To simplify this process, you can use the following special command:

**C-c M-w** **org-copy**  
Copying works like refiling, except that the original note is not deleted.

**C-c C-w** **org-refile**  
エンタリーや現在のリージョンをリファイルします。このコマンドはエンタリーをリファイルする場所を求め、補完とともに1つ選択させます。アイテム (またはリージョン内のアイテム) は対象の見出しの下にサブアイテムとして挿入されます。**org-reverse-note-order**に依存して、サブアイテムの先頭か最後のどちらかに挿入されます。デフォルトでは、現在のバッファのレベル1の見出しすべては対象と考えられますが、いくつかのファイルにまたがった複雑な定義もすることが可能です。詳細は**org-refile-targets**変数をご覧ください。もし場所をアウトラインのパスをファイルパスのような補完で選択したければ、**org-refile-use-outline-path**と**org-outline-path-complete-in-steps**変数をご覧ください。もしリファイルするノードの親ノードをその場で作れるようにしたいときは、**org-refile-allow-creating-parent-nodes**変数をご覧ください。**org-log-refile**変数<sup>9</sup>がセットされている場合、タイムスタンプやノートがエンタリーがリファイルされたときに記録されます。

**C-u C-c C-w**  
リファイルのインターフェースを見出しのジャンプに使用します。

**C-u C-u C-c C-w** **org-refile-goto-last-stored**  
**org-refile**が最後に木を移動させた場所に移動します。

---

<sup>9</sup> `#+STARTUP`キーワード、`logfile`、`lognoterefile`、そして`nologrefile`に対応している。

**C-2 C-c C-w**

現在計時しているアイテムの子としてリファイルします。

**C-3 C-c C-w**

Refile and keep the entry in place. Also see `org-refile-keep` to make this the default behavior, and beware that this may result in duplicated ID properties.

**C-0 C-c C-w** or **C-u C-u C-u C-c C-w** `org-refile-cache-clear`  
Clear the target cache. Caching of refile targets can be turned on by setting `org-refile-use-cache`. To make the command see new possible targets, you have to clear the cache with this command.

## 9.6 Archiving

(サブ) ツリーとして表現されたプロジェクトが終わった時、ツリーを外に移動し、それがアジェンダに現れない様にしたいでしょう。アーカイブは活動中のファイルをコンパクトにし、アジェンダビューを作るようなグローバルな検索を早くするために重要です。

**C-c C-x C-a** `org-archive-subtree-default`  
現在のエントリーを `org-archive-default-command` 変数で指定されたコマンドを使ってアーカイブします。

### 9.6.1 ツリーをアーカイブファイルへ移動

もっとも一般的なアーカイブアクションはプロジェクトを他のファイル、アーカイブファイル、へ移動させることです。

**C-c C-x C-s** or short **C-c \$** `org-archive-subtree`  
Archive the subtree starting at the cursor position to the location given by `org-archive-location`.

**C-u C-c C-x C-s**  
Check if any direct children of the current headline could be moved to the archive. To do this, each subtree is checked for open TODO entries. If none are found, the command offers to move it to the archive location. If the cursor is *not* on a headline when this command is invoked, the level 1 trees will be checked.

**C-u C-u C-c C-x C-s**  
As above, but check subtree for timestamps instead of TODO entries. The command will offer to archive the subtree if it *does* contain a timestamp, and that timestamp is in the past.

The default archive location is a file in the same directory as the current file, with the name derived by appending `_archive` to the current file name. You can also choose what heading to file archived items under, with the possibility to add them to a datetree in a file. For information and examples on how to specify the file and the heading, see the documentation string of the variable `org-archive-location`.

There is also an in-buffer option for setting this variable, for example:

```
#+ARCHIVE: %s_done::
```

If you would like to have a special ARCHIVE location for a single entry or a (sub)tree, give the entry an `:ARCHIVE:` property with the location as the value (see Chapter 7 [Properties and columns], page 64).

もしサブツリーが移動した場合、エントリーが移動してきたファイル、アーカイブしたときのアウトラインパスなどのコンテキスト情報が記録されます。`org-archive-save-context-info`変数を設定することで追加される情報の量を調整します。

### 9.6.2 Internal archiving

If you want to just switch off—for agenda views—certain subtrees without moving them to a different file, you can use the archive tag.

A headline that is marked with the `:ARCHIVE:` tag (see Chapter 6 [Tags], page 59) stays at its location in the outline tree, but behaves in the following way:

- 表示を切り替えるコマンド (see `<undefined>` [表示、非表示の切り替え], page `<undefined>`) では開くことはできません。アーカイブしたサブツリーを強制的に切り替えるには `C-TAB` を使うか、`org-cycle-open-archived-trees` オプションを設定します。また、`show-all` などの通常のアウトラインコマンドはアーカイブしたサブツリーも開きます。
- ツリーの抽出を行う過程で (see `<undefined>` [ツリーの抽出], page `<undefined>`)、アーカイブしたサブツリー内でマッチしたものは `org-suparse-tree-open-archived-trees` オプションを設定しない限り現れません。
- During agenda view construction (see Chapter 10 [Agenda views], page 102), the content of archived trees is ignored unless you configure the option `org-agenda-skip-archived-trees`, in which case these trees will always be included. In the agenda you can press `v a` to get archives temporarily included.
- Archived trees are not exported (see Chapter 12 [Exporting], page 138), only the headline is. Configure the details using the variable `org-export-with-archived-trees`.
- アーカイブしたツリーは `org-columns-skip-archived-trees` 変数が `nil` に設定されない限りカラムビューから除外されます。

以下に示すコマンドが ARCHIVE タグの編集に役立ちます。

`C-c C-x a` `org-toggle-archive-tag`  
Toggle the ARCHIVE tag for the current headline. When the tag is set, the headline changes to a shadowed face, and the subtree below it is hidden.

`C-u C-c C-x a`  
Check if any direct children of the current headline should be archived. To do this, each subtree is checked for open TODO entries. If none are found, the command offers to set the ARCHIVE tag for the child. If the cursor is *not* on a headline when this command is invoked, the level 1 trees will be checked.

`C-TAB` `org-force-cycle-archived`  
Cycle a tree even if it is tagged with ARCHIVE.

`C-c C-x A` `org-archive-to-archive-sibling`  
Move the current entry to the *Archive Sibling*. This is a sibling of the entry with the heading ‘Archive’ and the tag ‘ARCHIVE’. The entry becomes a child of that sibling and in this way retains a lot of its original context, including inherited tags and approximate position in the outline.



## 10 Agenda views

Due to the way Org works, TODO items, time-stamped items, and tagged headlines can be scattered throughout a file or even a number of files. To get an overview of open action items, or of events that are important for a particular date, this information must be collected, sorted and displayed in an organized way.

Org can select items based on various criteria and display them in a separate buffer. Six different view types are provided:

- アジェンダ カレンダーのように指定した日付の情報を表示します、
- *TODO* リスト 未完了のアクションアイテムをカバーします、
- マッチビュー 関連づけられているタグやプロパティ、TODO の状態に基づいて見出しを表示します、
- a *text search view* that shows all entries from multiple files that contain specified keywords,
- a *stuck projects view* showing projects that currently don't move along, and
- *custom views* that are special searches and combinations of different views.

抽出された情報は特別なアジェンダバッファに表示されます。このバッファはリードオンリーですが、オリジナルの Org-mode ファイルにジャンプしたり、オリジナルのファイルを間接的に編集することができます。

By default, the report ignores commented (see Section 12.6 [Comment lines], page 145) and archived (see Section 9.6.2 [Internal archiving], page 101) entries. You can override this by setting `org-agenda-skip-comment-trees` and `org-agenda-skip-archived-trees` to `nil`.

Two variables control how the agenda buffer is displayed and whether the window configuration is restored when the agenda exits: `org-agenda-window-setup` and `org-agenda-restore-windows-after-quit`.

### 10.1 Agenda files

The information to be shown is normally collected from all *agenda files*, the files listed in the variable `org-agenda-files`<sup>1</sup>. If a directory is part of this list, all files with the extension `.org` in this directory will be part of the list.

Thus, even if you only work with a single Org file, that file should be put into the list<sup>2</sup>. You can customize `org-agenda-files`, but the easiest way to maintain it is through the following commands

```
C-c [                                org-agenda-file-to-front
      アジェンダファイルのリストに編集中のファイルを追加する。そのファイルは、リス
      トの先頭に追加される。もしも既にリストに存在していたら、先頭に移動する。前置引
      数をつけることで、リストの最後に追加／移動する。
```

<sup>1</sup> If the value of that variable is not a list, but a single file name, then the list of agenda files will be maintained in that external file.

<sup>2</sup> When using the dispatcher, pressing < before selecting a command will actually limit the command to the current file, and ignore `org-agenda-files` until the next dispatcher command.

**C-c ]** **org-remove-file**  
 編集集中のファイルをアジェンダファイルのリストから削除する。

**C-'** **org-cycle-agenda-files**  
**C-,** アジェンダファイルのリストに従って、1つのファイルから次のファイルへと切り替える。

**M-x org-switchb RET**  
**iswitchb**と似たようなインターフェースで Org-mode のバッファの間を切り替えるコマンド。

The Org menu contains the current list of files and can be used to visit any of them.

If you would like to focus the agenda temporarily on a file not in this list, or on just one file in the list, or even on only a subtree in a file, then this can be done in different ways. For a single agenda command, you may press < once or several times in the dispatcher (see Section 10.2 [Agenda dispatcher], page 103). To restrict the agenda scope for an extended period, use the following commands:

**C-c C-x <** **org-agenda-set-restriction-lock**  
 Permanently restrict the agenda to the current subtree. When with a prefix argument, or with the cursor before the first headline in a file, the agenda scope is set to the entire file. This restriction remains in effect until removed with **C-c C-x >**, or by typing either < or > in the agenda dispatcher. If there is a window displaying an agenda view, the new restriction takes effect immediately.

**C-c C-x >** **org-agenda-remove-restriction-lock**  
 Remove the permanent restriction created by **C-c C-x <**.

**speedbar.el**を併用しているときは、Speedbar のフレームの中で以下のコマンドを使用することができます。

**< in the speedbar frame** **org-speedbar-set-agenda-restriction**  
 Permanently restrict the agenda to the item—either an Org file or a subtree in such a file—at the cursor in the Speedbar frame. If there is a window displaying an agenda view, the new restriction takes effect immediately.

**> in the speedbar frame** **org-agenda-remove-restriction-lock**  
 Lift the restriction.

## 10.2 アジェンダのコマンド選択画面

The views are created through a dispatcher, which should be bound to a global key—for example **C-c a** (see Section 1.3 [Activation], page 3). In the following we will assume that **C-c a** is indeed how the dispatcher is accessed and list keyboard access to commands accordingly. After pressing **C-c a**, an additional letter is required to execute a command. The dispatcher offers the following default commands:

**a** カレンダーのようなアジェンダを作成します。(see [\[毎週／毎日のアジェンダ\]](#), page [\[undefined\]](#))

**t / T** すべての TODO アイテムのリストを作成します。(see [\[グローバルな TODO リスト\]](#), page [\[undefined\]](#))

**m / M** タグの表記にマッチした見出しのリストを作成します。(see [\[タグとプロパティとのマッチング\]](#), page [\[undefined\]](#))

- s** Create a list of entries selected by a boolean expression of keywords and/or regular expressions that must or must not occur in the entry.
- /** Search for a regular expression in all agenda files and additionally in the files listed in `org-agenda-text-search-extra-files`. This uses the Emacs command `multi-occur`. A prefix argument can be used to specify the number of context lines for each match, default is 1.
- # / !** Create a list of stuck projects (see Section 10.3.5 [Stuck projects], page 110).
- <** Restrict an agenda command to the current buffer<sup>3</sup>. After pressing **<**, you still need to press the character selecting the command.
- < <** If there is an active region, restrict the following agenda command to the region. Otherwise, restrict it to the current subtree<sup>4</sup>. After pressing **< <**, you still need to press the character selecting the command.
- \*** Toggle sticky agenda views. By default, Org maintains only a single agenda buffer and rebuilds it each time you change the view, to make sure everything is always up to date. If you often switch between agenda views and the build time bothers you, you can turn on sticky agenda buffers or make this the default by customizing the variable `org-agenda-sticky`. With sticky agendas, the agenda dispatcher will not recreate agenda views from scratch, it will only switch to the selected one, and you need to update the agenda by hand with **r** or **g** when needed. You can toggle sticky agenda view any time with `org-toggle-sticky-agenda`.

You can also define custom commands that will be accessible through the dispatcher, just like the default commands. This includes the possibility to create extended agenda buffers that contain several blocks together, for example the weekly agenda, the global TODO list and a number of special tags matches. See Section 10.6 [Custom agenda views], page 124.

## 10.3 The built-in agenda views

このセクションではビルトインビューについて説明します。

### 10.3.1 The weekly/daily agenda

The purpose of the weekly/daily *agenda* is to act like a page of a paper agenda, showing all the tasks for the current week or day.

**C-c a a** **org-agenda-list**  
 Compile an agenda for the current week from a list of Org files. The agenda shows the entries for each day. With a numeric prefix<sup>5</sup> (like **C-u 2 1 C-c a a**) you may set the number of days to be displayed.

The default number of days displayed in the agenda is set by the variable `org-agenda-span` (or the obsolete `org-agenda-ndays`). This variable can be set to any number of

<sup>3</sup> For backward compatibility, you can also press **1** to restrict to the current buffer.

<sup>4</sup> For backward compatibility, you can also press **0** to restrict to the current region/subtree.

<sup>5</sup> For backward compatibility, the universal prefix **C-u** causes all TODO entries to be listed before the agenda. This feature is deprecated, use the dedicated TODO list, or a block agenda instead (see Section 10.6.2 [Block agenda], page 125).

days you want to see by default in the agenda, or to a span name, such as `day`, `week`, `month` or `year`. For weekly agendas, the default is to start on the previous monday (see `org-agenda-start-on-weekday`). You can also set the start date using a date shift: (`setq org-agenda-start-day "+10d"`) will start the agenda ten days from today in the future.

Remote editing from the agenda buffer means, for example, that you can change the dates of deadlines and appointments from the agenda buffer. The commands available in the Agenda buffer are listed in Section 10.5 [Agenda commands], page 115.

### カレンダー／日記の統合

Emacs contains the calendar and diary by Edward M. Reingold. The calendar displays a three-month calendar with holidays from different countries and cultures. The diary allows you to keep track of anniversaries, lunar phases, sunrise/set, recurrent appointments (weekly, monthly) and more. In this way, it is quite complementary to Org. It can be very useful to combine output from Org with the diary.

In order to include entries from the Emacs diary into Org mode's agenda, you only need to customize the variable

```
(setq org-agenda-include-diary t)
```

After that, everything will happen automatically. All diary entries including holidays, anniversaries, etc., will be included in the agenda buffer created by Org mode. `SPC`, `TAB`, and `RET` can be used from the agenda buffer to jump to the diary file in order to edit existing diary entries. The `i` command to insert new entries for the current date works in the agenda buffer, as well as the commands `S`, `M`, and `C` to display Sunrise/Sunset times, show lunar phases and to convert to other calendars, respectively. `c` can be used to switch back and forth between calendar and agenda.

If you are using the diary only for sexp entries and holidays, it is faster to not use the above setting, but instead to copy or even move the entries into an Org file. Org mode evaluates diary-style sexp entries, and does it faster because there is no overhead for first creating the diary display. Note that the sexp entries must start at the left margin, no whitespace is allowed before them. For example, the following segment of an Org file will be processed and entries will be made in the agenda:

```
* Holidays
:PROPERTIES:
:CATEGORY: Holiday
:END:
%%(org-calendar-holiday)    ; special function for holiday names

* Birthdays
:PROPERTIES:
:CATEGORY: Ann
:END:
%%(org-anniversary 1956  5 14)6 Arthur Dent is %d years old
%%(org-anniversary 1869 10  2) Mahatma Gandhi would be %d years old
```

---

<sup>6</sup> `org-anniversary` is just like `diary-anniversary`, but the argument order is always according to ISO and therefore independent of the value of `calendar-date-style`.

## Anniversaries from BBDB

If you are using the Big Brothers Database to store your contacts, you will very likely prefer to store anniversaries in BBDB rather than in a separate Org or diary file. Org supports this and will show BBDB anniversaries as part of the agenda. All you need to do is to add the following to one of your agenda files:

```
* Anniversaries
:PROPERTIES:
:CATEGORY: Anniv
:END:
%%(org-bbdb-anniversaries)
```

You can then go ahead and define anniversaries for a BBDB record. Basically, you need to press *C-o anniversary RET* with the cursor in a BBDB record and then add the date in the format YYYY-MM-DD or MM-DD, followed by a space and the class of the anniversary ('birthday' or 'wedding', or a format string). If you omit the class, it will default to 'birthday'. Here are a few examples, the header for the file `org-bbdb.el` contains more detailed information.

```
1973-06-22
06-22
1955-08-02 wedding
2008-04-14 %s released version 6.01 of org mode, %d years ago
```

After a change to BBDB, or for the first agenda display during an Emacs session, the agenda display will suffer a short delay as Org updates its hash with anniversaries. However, from then on things will be very fast—much faster in fact than a long list of '%%(diary-anniversary)' entries in an Org or Diary file.

If you would like to see upcoming anniversaries with a bit of forewarning, you can use the following instead:

```
* Anniversaries
:PROPERTIES:
:CATEGORY: Anniv
:END:
%%(org-bbdb-anniversaries-future 3)
```

That will give you three days' warning: on the anniversary date itself and the two days prior. The argument is optional: if omitted, it defaults to 7.

## Appointment reminders

Org can interact with Emacs appointments notification facility. To add the appointments of your agenda files, use the command `org-agenda-to-appt`. This command lets you filter through the list of your appointments and add only those belonging to a specific category or matching a regular expression. It also reads a `APPT_WARNTIME` property which will then override the value of `appt-message-warning-time` for this appointment. See the docstring for details.

### 10.3.2 The global TODO list

グローバルな TODO リストには、形式を整えられ、1つの場所に集められたすべての未完了の TODO アイテムが含まれています。

**C-c a t** **org-todo-list**  
 Show the global TODO list. This collects the TODO items from all agenda files (see Chapter 10 [Agenda views], page 102) into a single buffer. By default, this lists items with a state the is not a DONE state. The buffer is in **agenda-mode**, so there are commands to examine and manipulate the TODO entries directly from that buffer (see Section 10.5 [Agenda commands], page 115).

**C-c a T** **org-todo-list**  
 Like the above, but allows selection of a specific TODO keyword. You can also do this by specifying a prefix argument to **C-c a t**. You are prompted for a keyword, and you may also specify several keywords by separating them with ‘|’ as the boolean OR operator. With a numeric prefix, the Nth keyword in **org-todo-keywords** is selected. **r**キーをアジェンダバッファで使用するとバッファの再構成が行われます。たとえば **3 r** というように、前置引数をつけてこのコマンドを実行すると選択した TODO キーワードが変更することができます。もしも特定のキーワードを使って検索することが多い場合は、カスタムコマンドを定義することもできます (see [\[アジェンダのコマンド選択\]](#), page [\[undefined\]](#))。特定の TODO キーワードと合致するものを検索するのは、タグ検索の 1 機能として行うこともできます (see [\[タグの検索\]](#), page [\[undefined\]](#))。)

リモートで TODO アイテムを編集するという意味は、1 つのキーを入力することで TODO エントリーの状態を変更できるということです。TODO リストの中で利用できるコマンドは [\[アジェンダのコマンド\]](#), page [\[undefined\]](#) の記述を参考にしてください。

Normally the global TODO list simply shows all headlines with TODO keywords. This list can become very long. There are two ways to keep it more compact:

- Some people view a TODO item that has been *scheduled* for execution or have a *deadline* (see Section 8.1 [Timestamps], page 73) as no longer *open*. Configure the variables **org-agenda-todo-ignore-scheduled**, **org-agenda-todo-ignore-deadlines**, **org-agenda-todo-ignore-timestamp** and/or **org-agenda-todo-ignore-with-date** to exclude such items from the global TODO list.
- TODO items may have sublevels to break up the task into subtasks. In such cases it may be enough to list only the highest level TODO headline and omit the sublevels from the global list. Configure the variable **org-agenda-todo-list-sublevels** to get this behavior.

### 10.3.3 Matching tags and properties

If headlines in the agenda files are marked with *tags* (see Chapter 6 [Tags], page 59), or have properties (see Chapter 7 [Properties and columns], page 64), you can select headlines based on this metadata and collect them into an agenda buffer. The match syntax described here also applies when creating sparse trees with **C-c / m**.

**C-c a m** **org-tags-view**  
 Produce a list of all headlines that match a given set of tags. The command prompts for a selection criterion, which is a boolean logic expression with tags, like **+work+urgent-withboss** or **work|home** (see Chapter 6 [Tags], page 59). If you often need a specific search, define a custom command for it (see Section 10.2 [Agenda dispatcher], page 103).

**C-c a M****org-tags-view**

Like **C-c a m**, but only select headlines that are also TODO items in a not-DONE state and force checking subitems (see variable **org-tags-match-list-sublevels**). To exclude scheduled/deadline items, see the variable **org-agenda-tags-todo-honor-ignore-options**. Matching specific TODO keywords together with a tags match is also possible, see Section 6.4 [Tag searches], page 63.

タグのリストで利用できるコマンドは `<undefined>` [アジェンダのコマンド], page `<undefined>` のところで説明しています。

## Match syntax

A search string can use Boolean operators **&** for AND and **|** for OR. **&** binds more strongly than **|**. Parentheses are not implemented. Each element in the search is either a tag, a regular expression matching tags, or an expression like **PROPERTY OPERATOR VALUE** with a comparison operator, accessing a property value. Each element may be preceded by **-**, to select against it, and **+** is syntactic sugar for positive selection. The AND operator **&** is optional when **+** or **-** is present. Here are some examples, using only tags.

**'work'**        Select headlines tagged **:work:**.

**'work&boss'**  
Select headlines tagged **:work:** and **:boss:**.

**'+work-boss'**  
Select headlines tagged **:work:**, but discard those also tagged **:boss:**.

**'work|laptop'**  
Selects lines tagged **:work:** or **:laptop:**.

**'work|laptop+night'**  
Like before, but require the **:laptop:** lines to be tagged also **:night:**.

Instead of a tag, you may also specify a regular expression enclosed in curly braces. For example, **'work+{^boss.\*}'** matches headlines that contain the tag **:work:** and any tag *starting* with **boss**.

Group tags (see Section 6.3 [Tag hierarchy], page 62) are expanded as regular expressions. E.g., if **:work:** is a group tag for the group **:work:lab:conf:**, then searching for **'work'** will search for **'{\(?:work\|lab\|conf\)}'** and searching for **'-work'** will search for all headlines but those with one of the tags in the group (i.e., **'-{\(?:work\|lab\|conf\)}'**).

You may also test for properties (see Chapter 7 [Properties and columns], page 64) at the same time as matching tags. The properties may be real properties, or special properties that represent other metadata (see Section 7.2 [Special properties], page 66). For example, the “property” **TODO** represents the TODO keyword of the entry and the “property” **PRIORITY** represents the PRIORITY keyword of the entry.

In addition to the properties mentioned above, **LEVEL** represents the level of an entry. So a search **'+LEVEL=3+boss-TODO="DONE"'** lists all level three headlines that have the tag **boss** and are *not* marked with the TODO keyword DONE. In buffers with **org-odd-levels-only** set, **LEVEL** does not count the number of stars, but **LEVEL=2** will correspond to 3 stars etc.

Here are more examples:

```
'work+TODO="WAITING"'
```

Select `:work:`-tagged TODO lines with the specific TODO keyword `'WAITING'`.

```
'work+TODO="WAITING"|home+TODO="WAITING"'
```

Waiting tasks both at work and at home.

When matching properties, a number of different operators can be used to test the value of a property. Here is a complex example:

```
+work-boss+PRIORITY="A"+Coffee="unlimited"+Effort<2 \
+With={Sarah\Denny}+SCHEDULED>="<2008-10-11>"
```

比較のタイプは比較の値がどのように書かれているかによります。

- If the comparison value is a plain number, a numerical comparison is done, and the allowed operators are `'<'`, `'='`, `'>'`, `'<='`, `'>='`, and `'<>'`.
- If the comparison value is enclosed in double-quotes, a string comparison is done, and the same operators are allowed.
- If the comparison value is enclosed in double-quotes *and* angular brackets (like `'DEADLINE<="<2008-12-24 18:30>"'`), both values are assumed to be date/time specifications in the standard Org way, and the comparison will be done accordingly. Special values that will be recognized are `"<now>"` for now (including time), and `"<today>"`, and `"<tomorrow>"` for these days at 00:00 hours, i.e., without a time specification. Also strings like `"<+5d>"` or `"<-2m>"` with units `d`, `w`, `m`, and `y` for day, week, month, and year, respectively, can be used.
- If the comparison value is enclosed in curly braces, a regexp match is performed, with `'='` meaning that the regexp matches the property value, and `'<>'` meaning that it does not match.

So the search string in the example finds entries tagged `:work:` but not `:boss:`, which also have a priority value `'A'`, a `:Coffee:` property with the value `'unlimited'`, an `'Effort'` property that is numerically smaller than 2, a `:With:` property that is matched by the regular expression `'Sarah\Denny'`, and that are scheduled on or after October 11, 2008.

You can configure Org mode to use property inheritance during a search, but beware that this can slow down searches considerably. See Section 7.4 [Property inheritance], page 67, for details.

For backward compatibility, and also for typing speed, there is also a different way to test TODO states in a search. For this, terminate the tags/property part of the search string (which may include several terms connected with `'|'`) with a `'/'` and then specify a Boolean expression just for TODO keywords. The syntax is then similar to that for tags, but should be applied with care: for example, a positive selection on several TODO keywords cannot meaningfully be combined with boolean AND. However, *negative selection* combined with AND can be meaningful. To make sure that only lines are checked that actually have any TODO keyword (resulting in a speed-up), use `C-c a M`, or equivalently start the TODO part after the slash with `'!'`. Using `C-c a M` or `'/!'` will not match TODO keywords in a DONE state. Examples:

```
'work/WAITING'
```

`'work+TODO="WAITING"'`と同じ



`'work/!-WAITING-NEXT'`

‘:work:’を選択。ただし TODO 行では ‘WAITING’ と ‘NEXT’ のどちらのタグもついていないもの

`'work/!+WAITING|+NEXT'`

Select ‘:work:’-tagged TODO lines that are either ‘WAITING’ or ‘NEXT’.

### 10.3.4 Search view

This agenda view is a general text search facility for Org mode entries. It is particularly useful to find notes.

`C-c a s`

`org-search-view`

This is a special search that lets you select entries by matching a substring or specific words using a boolean logic.

For example, the search string ‘computer equipment’ will find entries that contain ‘computer equipment’ as a substring. If the two words are separated by more space or a line break, the search will still match. Search view can also search for specific keywords in the entry, using Boolean logic. The search string ‘+computer +wifi -ethernet -{8\11[bg]}’ will search for note entries that contain the keywords `computer` and `wifi`, but not the keyword `ethernet`, and which are also not matched by the regular expression `8\11[bg]`, meaning to exclude both 8.11b and 8.11g. The first ‘+’ is necessary to turn on word search, other ‘+’ characters are optional. For more details, see the docstring of the command `org-search-view`.

アジェンダファイルに加えて、このコマンドは `org-agenda-text-search-extra-files` の中で一覧になっているファイルもまた検索するということに注意してください。

### 10.3.5 Stuck projects

If you are following a system like David Allen’s GTD to organize your work, one of the “duties” you have is a regular review to make sure that all projects move along. A *stuck* project is a project that has no defined next actions, so it will never show up in the TODO lists Org mode produces. During the review, you need to identify such projects and define next actions for them.

`C-c a #`

`org-agenda-list-stuck-projects`

List projects that are stuck.

`C-c a !`

Customize the variable `org-stuck-projects` to define what a stuck project is and how to find it.

You almost certainly will have to configure this view before it will work for you. The built-in default assumes that all your projects are level-2 headlines, and that a project is not stuck if it has at least one entry marked with a TODO keyword `TODO` or `NEXT` or `NEXTACTION`.

Let’s assume that you, in your own way of using Org mode, identify projects with a tag `PROJECT`, and that you use a TODO keyword `MAYBE` to indicate a project that should not be considered yet. Let’s further assume that the TODO keyword `DONE` marks finished projects, and that `NEXT` and `TODO` indicate next actions. The tag `@SHOP` indicates shopping and is a next action even without the `NEXT` tag. Finally, if the project contains

the special word `IGNORE` anywhere, it should not be listed either. In this case you would start by identifying eligible projects with a tags/todo match<sup>7</sup> `'+PROJECT/-MAYBE-DONE'`, and then check for `TODO`, `NEXT`, `@SHOP`, and `IGNORE` in the subtree to identify projects that are not stuck. The correct customization for this is

```
(setq org-stuck-projects
  '("+PROJECT/-MAYBE-DONE" ("NEXT" "TODO") ("@SHOP")
    "\\<IGNORE\\>"))
```

Note that if a project is identified as non-stuck, the subtree of this entry will still be searched for stuck projects.

## 10.4 Presentation and sorting

Before displaying items in an agenda view, Org mode visually prepares the items and sorts them. Each item occupies a single line. The line starts with a *prefix* that contains the *category* (see Section 10.4.1 [Categories], page 111) of the item and other important information. You can customize in which column tags will be displayed through `org-agenda-tags-column`. You can also customize the prefix using the option `org-agenda-prefix-format`. This prefix is followed by a cleaned-up version of the outline headline associated with the item.

### 10.4.1 Categories

The category is a broad label assigned to each agenda item. By default, the category is simply derived from the file name, but you can also specify it with a special line in the buffer, like this:

```
#+CATEGORY: Thesis
```

If you would like to have a special `CATEGORY` for a single entry or a (sub)tree, give the entry a `:CATEGORY:` property with the special category you want to apply as the value.

The display in the agenda buffer looks best if the category is not longer than 10 characters.

あなたは `org-agenda-category-icon-alist` 変数をカスタマイズすることで、カテゴリーにアイコンを設定することができます。

### 10.4.2 Time-of-day specifications

Org mode checks each agenda item for a time-of-day specification. The time can be part of the timestamp that triggered inclusion into the agenda, for example as in `'<2005-05-10 Tue 19:00>'`. Time ranges can be specified with two timestamps, like `'<2005-05-10 Tue 20:30>--<2005-05-10 Tue 22:15>'`.

In the headline of the entry itself, a time(range) may also appear as plain text (like `'12:45'` or a `'8:30-1pm'`)<sup>8</sup>.

If the agenda integrates the Emacs diary (see Section 10.3.1 [Weekly/daily agenda], page 104), time specifications in diary entries are recognized as well.

<sup>7</sup> See Section 6.4 [Tag searches], page 63.

<sup>8</sup> You can, however, disable this by setting `org-agenda-search-headline-for-time` variable to a `nil` value.

For agenda display, Org mode extracts the time and displays it in a standard 24 hour format as part of the prefix. The example times in the previous paragraphs would end up in the agenda like this:

```
8:30-13:00 Arthur Dent lies in front of the bulldozer
12:45..... Ford Prefect arrives and takes Arthur to the pub
19:00..... The Vagon reads his poem
20:30-22:15 Marvin escorts the Hitchhikers to the bridge
```

もしもアジェンダが一日モードであるならば、あるいは今日を表示しているならば、時間設定されたエントリーは、次のような時間のグリッドに埋め込まれます。

```
8:00..... -----
8:30-13:00 Arthur Dent lies in front of the bulldozer
10:00..... -----
12:00..... -----
12:45..... Ford Prefect arrives and takes Arthur to the pub
14:00..... -----
16:00..... -----
18:00..... -----
19:00..... The Vagon reads his poem
20:00..... -----
20:30-22:15 Marvin escorts the Hitchhikers to the bridge
```

時間のグリッドは、`org-agenda-use-time-grid`変数で表示したりしなかったさせることができます。そしてまた `org-agenda-time-grid`で設定をすることができます。

### 10.4.3 Sorting agenda items

Before being inserted into a view, the items are sorted. How this is done depends on the type of view.

- For the daily/weekly agenda, the items for each day are sorted. The default order is to first collect all items containing an explicit time-of-day specification. These entries will be shown at the beginning of the list, as a *schedule* for the day. After that, items remain grouped in categories, in the sequence given by `org-agenda-files`. Within each category, items are sorted by priority (see Section 5.4 [Priorities], page 55), which is composed of the base priority (2000 for priority ‘A’, 1000 for ‘B’, and 0 for ‘C’), plus additional increments for overdue scheduled or deadline items.
- For the TODO list, items remain in the order of categories, but within each category, sorting takes place according to priority (see Section 5.4 [Priorities], page 55). The priority used for sorting derives from the priority cookie, with additions depending on how close an item is to its due or scheduled date.
- For tags matches, items are not sorted at all, but just appear in the sequence in which they are found in the agenda files.

Sorting can be customized using the variable `org-agenda-sorting-strategy`, and may also include criteria based on the estimated effort of an entry (see Section 8.5 [Effort estimates], page 87).

### 10.4.4 Filtering/limiting agenda items

Agenda built-in or customized commands are statically defined. Agenda filters and limits provide two ways of dynamically narrowing down the list of agenda entries: *filters* and *limits*. Filters only act on the display of the items, while limits take effect before the list of agenda entries is built. Filters are more often used interactively, while limits are mostly useful when defined as local variables within custom agenda commands.

### Filtering in the agenda

`/` **org-agenda-filter-by-tag**  
Filter the agenda view with respect to a tag and/or effort estimates. The difference between this and a custom agenda command is that filtering is very fast, so that you can switch quickly between different filters without having to recreate the agenda.<sup>9</sup>

You will be prompted for a tag selection letter; **SPC** will mean any tag at all. Pressing **TAB** at that prompt will offer use completion to select a tag (including any tags that do not have a selection character). The command then hides all entries that do not contain or inherit this tag. When called with prefix arg, remove the entries that *do* have the tag. A second `/` at the prompt will turn off the filter and unhide any hidden entries. Pressing `+` or `-` switches between filtering and excluding the next tag.

Org also supports automatic, context-aware tag filtering. If the variable **org-agenda-auto-exclude-function** is set to a user-defined function, that function can decide which tags should be excluded from the agenda automatically. Once this is set, the `/` command then accepts **RET** as a sub-option key and runs the auto exclusion logic. For example, let's say you use a **Net** tag to identify tasks which need network access, an **Errand** tag for errands in town, and a **Call** tag for making phone calls. You could auto-exclude these tags based on the availability of the Internet, and outside of business hours, with something like this:

```
(defun org-my-auto-exclude-function (tag)
  (and (cond
        ((string= tag "Net")
         (/= 0 (call-process "/sbin/ping" nil nil nil
                             "-c1" "-q" "-t1" "mail.gnu.org")))
        ((or (string= tag "Errand") (string= tag "Call"))
         (let ((hour (nth 2 (decode-time))))
           (or (< hour 8) (> hour 21))))))
    (concat "-" tag)))

(setq org-agenda-auto-exclude-function 'org-my-auto-exclude-function)
```

`[] {}`

---

<sup>9</sup> Custom commands can preset a filter by binding the variable **org-agenda-tag-filter-preset** as an option. This filter will then be applied to the view and persist as a basic filter through refreshes and more secondary filtering. The filter is a global property of the entire agenda view—in a block agenda, you should only set this in the global options section, not in the section of an individual block.

in *search view*

add new search words (`[` and `]`) or new regular expressions (`{` and `}`) to the query string. The opening bracket/brace will add a positive search term prefixed by `+`, indicating that this search term *must* occur/match in the entry. The closing bracket/brace will add a negative search term which *must not* occur/match in the entry for it to be selected.

**<** **org-agenda-filter-by-category**  
Filter the current agenda view with respect to the category of the item at point. Pressing `<` another time will remove this filter. When called with a prefix argument exclude the category of the item at point from the agenda.

You can add a filter preset in custom agenda commands through the option **org-agenda-category-filter-preset**. See Section 10.6.3 [Setting options], page 125.

**^** **org-agenda-filter-by-top-headline**  
Filter the current agenda view and only display the siblings and the parent headline of the one at point.

**=** **org-agenda-filter-by-regexp**  
Filter the agenda view by a regular expression: only show agenda entries matching the regular expression the user entered. When called with a prefix argument, it will filter *out* entries matching the regexp. With two universal prefix arguments, it will remove all the regexp filters, which can be accumulated. You can add a filter preset in custom agenda commands through the option **org-agenda-regexp-filter-preset**. See Section 10.6.3 [Setting options], page 125.

**-** **org-agenda-filter-by-effort**  
Filter the agenda view with respect to effort estimates. You first need to set up allowed efforts globally, for example

```
(setq org-global-properties
      '(("Effort_ALL". "0 0:10 0:30 1:00 2:00 3:00 4:00")))
```

You can then filter for an effort by first typing an operator, one of `<`, `>`, and `=`, and then the one-digit index of an effort estimate in your array of allowed values, where `0` means the 10th value. The filter will then restrict to entries with effort smaller-or-equal, equal, or larger-or-equal than the selected value. For application of the operator, entries without a defined effort will be treated according to the value of **org-sort-agenda-noeffort-is-high**.

When called with a prefix argument, it will remove entries matching the condition. With two universal prefix arguments, it will clear effort filters, which can be accumulated.

You can add a filter preset in custom agenda commands through the option **org-agenda-effort-filter-preset**. See Section 10.6.3 [Setting options], page 125.

**|** **org-agenda-filter-remove-all**  
Remove all filters in the current agenda view.

## Setting limits for the agenda

Here is a list of options that you can set, either globally, or locally in your custom agenda views (see Section 10.6 [Custom agenda views], page 124).

**org-agenda-max-entries**

Limit the number of entries.

**org-agenda-max-effort**

Limit the duration of accumulated efforts (as minutes).

**org-agenda-max-todos**

Limit the number of entries with TODO keywords.

**org-agenda-max-tags**

Limit the number of tagged entries.

When set to a positive integer, each option will exclude entries from other categories: for example, `(setq org-agenda-max-effort 100)` will limit the agenda to 100 minutes of effort and exclude any entry that has no effort property. If you want to include entries with no effort property, use a negative value for **org-agenda-max-effort**.

One useful setup is to use **org-agenda-max-entries** locally in a custom command. For example, this custom command will display the next five entries with a NEXT TODO keyword.

```
(setq org-agenda-custom-commands
      '(("n" todo "NEXT"
         ((org-agenda-max-entries 5)))))
```

Once you mark one of these five entry as DONE, rebuilding the agenda will again the next five entries again, including the first entry that was excluded so far.

You can also dynamically set temporary limits, which will be lost when rebuilding the agenda:

```
~                                     org-agenda-limit-interactively
                                     This prompts for the type of limit to apply and its value.
```

## 10.5 Commands in the agenda buffer

Entries in the agenda buffer are linked back to the Org file or diary file where they originate. You are not allowed to edit the agenda buffer itself, but commands are provided to show and jump to the original entry location, and to edit the Org files “remotely” from the agenda buffer. In this way, all information is stored only once, removing the risk that your agenda and note files may diverge.

いくつかのコマンドはアジェンダの行上でマウスをクリックすることで実行されます。それ以外のコマンドは、必要とされる行の中にカーソルが置かれている必要があります。

### Motion

|          |                                     |                                 |
|----------|-------------------------------------|---------------------------------|
| <b>n</b> | Next line (same as DOWN and C-n).   | <b>org-agenda-next-line</b>     |
| <b>p</b> | Previous line (same as UP and C-p). | <b>org-agenda-previous-line</b> |

*N* org-agenda-next-item  
Next item: same as next line, but only consider items.

*P* org-agenda-previous-item  
Previous item: same as previous line, but only consider items.

### View/Go to Org file

*SPC* or *mouse-3* org-agenda-show-and-scroll-up  
Display the original location of the item in another window. With prefix arg, make sure that drawers stay folded.

*L* org-agenda-recenter  
Display original location and recenter that window.

*TAB* or *mouse-2* org-agenda-goto  
Go to the original location of the item in another window.

*RET* org-agenda-switch-to  
Go to the original location of the item and delete other windows.

*F* org-agenda-follow-mode  
Toggle Follow mode. In Follow mode, as you move the cursor through the agenda buffer, the other window always shows the corresponding location in the Org file. The initial setting for this mode in new agenda buffers can be set with the variable `org-agenda-start-with-follow-mode`.

*C-c C-x b* org-agenda-tree-to-indirect-buffer  
Display the entire subtree of the current item in an indirect buffer. With a numeric prefix argument *N*, go up to level *N* and then take that tree. If *N* is negative, go up that many levels. With a *C-u* prefix, do not remove the previously used indirect buffer.

*C-c C-o* org-agenda-open-link  
Follow a link in the entry. This will offer a selection of any links in the text belonging to the referenced Org node. If there is only one link, it will be followed without a selection prompt.

### Change display

*A* Interactively select another agenda view and append it to the current view.

*o* Delete other windows.

|                              |                           |
|------------------------------|---------------------------|
| <i>v d</i> or short <i>d</i> | org-agenda-day-view       |
| <i>v w</i> or short <i>w</i> | org-agenda-week-view      |
| <i>v t</i>                   | org-agenda-fortnight-view |
| <i>v m</i>                   | org-agenda-month-view     |
| <i>v y</i>                   | org-agenda-year-view      |
| <i>v SPC</i>                 | org-agenda-reset-view     |

Switch to day/week/month/year view. When switching to day or week view, this setting becomes the default for subsequent agenda refreshes. Since month and year views are slow to create, they do not become the default. A numeric prefix argument may be used to jump directly to a specific day of the year, ISO week, month, or year, respectively. For example, *32 d* jumps to February 1st, *9 w* to ISO week number 9. When setting day, week, or month view, a year may

be encoded in the prefix argument as well. For example, `200712 w` will jump to week 12 in 2007. If such a year specification has only one or two digits, it will be mapped to the interval 1938–2037. `v SPC` will reset to what is set in `org-agenda-span`.

- |   |   |
|---|---|
| <code>f</code>  | <code>org-agenda-later</code>   |
| Go forward in time to display the following <code>org-agenda-current-span</code> days. For example, if the display covers a week, switch to the following week. With prefix arg, go forward that many times <code>org-agenda-current-span</code> days.  |   |
| <code>b</code>  | <code>org-agenda-earlier</code>   |
| Go backward in time to display earlier dates.   |   |
| <code>.</code>  | <code>org-agenda-goto-today</code>  |
| Go to today.  |   |
| <code>j</code>  | <code>org-agenda-goto-date</code>   |
| Prompt for a date and go there.   |   |
| <code>J</code>  | <code>org-agenda-clock-goto</code>  |
| Go to the currently clocked-in task <i>in the agenda buffer</i> .   |   |
| <code>D</code>  | <code>org-agenda-toggle-diary</code>  |
| Toggle the inclusion of diary entries. See Section 10.3.1 [Weekly/daily agenda], page 104.  |   |
| <code>v l</code> or short <code>l</code>  | <code>org-agenda-log-mode</code>  |
| Toggle Logbook mode. In Logbook mode, entries that were marked DONE while logging was on (variable <code>org-log-done</code> ) are shown in the agenda, as are entries that have been clocked on that day. You can configure the entry types that should be included in log mode using the variable <code>org-agenda-log-mode-items</code> . When called with a <code>C-u</code> prefix, show all possible logbook entries, including state changes. When called with two prefix arguments <code>C-u C-u</code> , show only logging information, nothing else. <code>v L</code> is equivalent to <code>C-u v l</code> . |   |
| <code>v [</code> or short <code>[</code>  | <code>org-agenda-manipulate-query-add</code>  |
| Include inactive timestamps into the current view. Only for weekly/daily agenda.  |   |
| <code>v a</code><br><code>v A</code>  | <code>org-agenda-archives-mode</code><br><code>org-agenda-archives-mode 'files</code> |
| Toggle Archives mode. In Archives mode, trees that are marked ARCHIVED are also scanned when producing the agenda. When you use the capital A, even all archive files are included. To exit archives mode, press <code>v a</code> again.  |   |
| <code>v R</code> or short <code>R</code>  | <code>org-agenda-clockreport-mode</code>  |
| Toggle Clockreport mode. In Clockreport mode, the daily/weekly agenda will always show a table with the clocked times for the time span and file scope covered by the current agenda view. The initial setting for this mode in new agenda buffers can be set with the variable <code>org-agenda-start-with-clockreport-mode</code> . By using a prefix argument when toggling this mode (i.e., <code>C-u R</code> ), the clock table will not show contributions from entries that are hidden by agenda  |   |



filtering<sup>10</sup>. See also the variable `org-clock-report-include-clocking-task`.

**v c** Show overlapping clock entries, clocking gaps, and other clocking problems in the current agenda range. You can then visit clocking lines and fix them manually. See the variable `org-agenda-clock-consistency-checks` for information on how to customize the definition of what constituted a clocking problem. To return to normal agenda display, press `l` to exit Logbook mode.

**v E** or short **E** `org-agenda-entry-text-mode`  
Toggle entry text mode. In entry text mode, a number of lines from the Org outline node referenced by an agenda line will be displayed below the line. The maximum number of lines is given by the variable `org-agenda-entry-text-maxlines`. Calling this command with a numeric prefix argument will temporarily modify that number to the prefix value.

**G** `org-agenda-toggle-time-grid`  
Toggle the time grid on and off. See also the variables `org-agenda-use-time-grid` and `org-agenda-time-grid`.

**r** `org-agenda-redo`  
Recreate the agenda buffer, for example to reflect the changes after modification of the timestamps of items with *S-LEFT* and *S-RIGHT*. When the buffer is the global TODO list, a prefix argument is interpreted to create a selective list for a specific TODO keyword.

**g** `org-agenda-redo`  
Same as **r**.

**C-x C-s** or short **s** `org-save-all-org-buffers`  
Save all Org buffers in the current Emacs session, and also the locations of IDs.

**C-c C-x C-c** `org-agenda-columns`  
Invoke column view (see Section 7.5 [Column view], page 67) in the agenda buffer. The column view format is taken from the entry at point, or (if there is no entry at point), from the first entry in the agenda view. So whatever the format for that entry would be in the original buffer (taken from a property, from a `#+COLUMNS` line, or from the default variable `org-columns-default-format`), will be used in the agenda.

**C-c C-x >** `org-agenda-remove-restriction-lock`  
Remove the restriction lock on the agenda, if it is currently restricted to a file or subtree (see Section 10.1 [Agenda files], page 102).

### Secondary filtering and query editing

For a detailed description of these commands, see Section 10.4.4 [Filtering/limiting agenda items], page 113.

**/** `org-agenda-filter-by-tag`  
Filter the agenda view with respect to a tag and/or effort estimates.

---

<sup>10</sup> Only tags filtering will be respected here, effort filtering is ignored.

|   |   |
|---|---|
| < | <code>org-agenda-filter-by-category</code>  |
|   | Filter the current agenda view with respect to the category of the item at point.                         |
| ^ | <code>org-agenda-filter-by-top-headline</code>  |
|   | Filter the current agenda view and only display the siblings and the parent headline of the one at point. |
| = | <code>org-agenda-filter-by-regexp</code>  |
|   | Filter the agenda view by a regular expression.   |
| - | <code>org-agenda-filter-by-effort</code>  |
|   | Filter the agenda view with respect to effort estimates.  |
|   | <code>org-agenda-filter-remove-all</code>   |
|   | Remove all filters in the current agenda view.  |

### Remote editing

|                         |  |
|-------------------------|--|
| 0--9                    | Digit argument.  |
| C-_                     | <code>org-agenda-undo</code>   |
|                         | Undo a change due to a remote editing command. The change is undone both in the agenda buffer and in the remote buffer.  |
| t                       | <code>org-agenda-todo</code>   |
|                         | Change the TODO state of the item, both in the agenda and in the original org file.  |
| C-S-RIGHT               | <code>org-agenda-todo-nextset</code>   |
| C-S-LEFT                | <code>org-agenda-todo-previousset</code>   |
|                         | Switch to the next/previous set of TODO keywords.  |
| C-k                     | <code>org-agenda-kill</code>   |
|                         | Delete the current agenda item along with the entire subtree belonging to it in the original Org file. If the text to be deleted remotely is longer than one line, the kill needs to be confirmed by the user. See variable <code>org-agenda-confirm-kill</code> . |
| C-c C-w                 | <code>org-agenda-refile</code>   |
|                         | Refile the entry at point.   |
| C-c C-x C-a or short a  | <code>org-agenda-archive-default-with-confirmation</code>  |
|                         | Archive the subtree corresponding to the entry at point using the default archiving command set in <code>org-archive-default-command</code> . When using the a key, confirmation will be required.   |
| C-c C-x a               | <code>org-agenda-toggle-archive-tag</code>   |
|                         | Toggle the ARCHIVE tag for the current headline.   |
| C-c C-x A               | <code>org-agenda-archive-to-archive-sibling</code>   |
|                         | Move the subtree corresponding to the current entry to its <i>archive sibling</i> .  |
| C-c C-x C-s or short \$ | <code>org-agenda-archive</code>  |
|                         | Archive the subtree corresponding to the current headline. This means the entry will be moved to the configured archive location, most likely a different file.  |

|                     |  |
|---------------------|--|
| <i>T</i>            | <code>org-agenda-show-tags</code>  |
|                     | Show all tags associated with the current item. This is useful if you have turned off <code>org-agenda-show-inherited-tags</code> , but still want to see all tags of a headline occasionally.   |
| <i>:</i>            | <code>org-agenda-set-tags</code>   |
|                     | Set tags for the current headline. If there is an active region in the agenda, change a tag for all headings in the region.  |
| <i>,</i>            |  |
|                     | Set the priority for the current item ( <code>org-agenda-priority</code> ). Org mode prompts for the priority character. If you reply with <code>SPC</code> , the priority cookie is removed from the entry.   |
| <i>P</i>            | <code>org-agenda-show-priority</code>  |
|                     | Display weighted priority of current item.   |
| <i>+ or S-UP</i>    | <code>org-agenda-priority-up</code>  |
|                     | Increase the priority of the current item. The priority is changed in the original buffer, but the agenda is not resorted. Use the <code>r</code> key for this.  |
| <i>- or S-DOWN</i>  | <code>org-agenda-priority-down</code>  |
|                     | Decrease the priority of the current item.   |
| <i>z or C-c C-z</i> | <code>org-agenda-add-note</code>   |
|                     | Add a note to the entry. This note will be recorded, and then filed to the same location where state change notes are put. Depending on <code>org-log-into-drawer</code> , this may be inside a drawer.  |
| <i>C-c C-a</i>      | <code>org-attach</code>  |
|                     | Dispatcher for all command related to attachments.   |
| <i>C-c C-s</i>      | <code>org-agenda-schedule</code>   |
|                     | Schedule this item. With prefix arg remove the scheduling timestamp  |
| <i>C-c C-d</i>      | <code>org-agenda-deadline</code>   |
|                     | Set a deadline for this item. With prefix arg remove the deadline.   |
| <i>S-RIGHT</i>      | <code>org-agenda-do-date-later</code>  |
|                     | Change the timestamp associated with the current line by one day into the future. If the date is in the past, the first call to this command will move it to today.<br>With a numeric prefix argument, change it by that many days. For example, <code>3 6 5 S-RIGHT</code> will change it by a year. With a <code>C-u</code> prefix, change the time by one hour. If you immediately repeat the command, it will continue to change hours even without the prefix arg. With a double <code>C-u C-u</code> prefix, do the same for changing minutes.<br>The stamp is changed in the original Org file, but the change is not directly reflected in the agenda buffer. Use <code>r</code> or <code>g</code> to update the buffer. |
| <i>S-LEFT</i>       | <code>org-agenda-do-date-earlier</code>  |
|                     | Change the timestamp associated with the current line by one day into the past.  |

|             |   |
|-------------|---|
| <b>&gt;</b> | <b>org-agenda-date-prompt</b><br>Change the timestamp associated with the current line. The key > has been chosen, because it is the same as <i>S-</i> . on my keyboard.  |
| <b>I</b>    | <b>org-agenda-clock-in</b><br>Start the clock on the current item. If a clock is running already, it is stopped first.  |
| <b>O</b>    | <b>org-agenda-clock-out</b><br>Stop the previously started clock.   |
| <b>X</b>    | <b>org-agenda-clock-cancel</b><br>Cancel the currently running clock.   |
| <b>J</b>    | <b>org-agenda-clock-goto</b><br>Jump to the running clock in another window.  |
| <b>k</b>    | <b>org-agenda-capture</b><br>Like <b>org-capture</b> , but use the date at point as the default date for the capture template. See <b>org-capture-use-agenda-date</b> to make this the default behavior of <b>org-capture</b> . |

#### Dragging agenda lines forward/backward

|                       |  |
|-----------------------|--|
| <b>M-&lt;up&gt;</b>   | <b>org-agenda-drag-line-backward</b><br>Drag the line at point backward one line <sup>11</sup> . With a numeric prefix argument, drag backward by that many lines. |
| <b>M-&lt;down&gt;</b> | <b>org-agenda-drag-line-forward</b><br>Drag the line at point forward one line. With a numeric prefix argument, drag forward by that many lines.                   |

#### Bulk remote editing selected entries

|            |  |
|------------|--|
| <b>m</b>   | <b>org-agenda-bulk-mark</b><br>Mark the entry at point for bulk action. With numeric prefix argument, mark that many successive entries. |
| <b>*</b>   | <b>org-agenda-bulk-mark-all</b><br>Mark all visible agenda entries for bulk action.  |
| <b>u</b>   | <b>org-agenda-bulk-unmark</b><br>Unmark entry at point for bulk action.  |
| <b>U</b>   | <b>org-agenda-bulk-remove-all-marks</b><br>Unmark all marked entries for bulk action.  |
| <b>M-m</b> | <b>org-agenda-bulk-toggle</b><br>Toggle mark of the entry at point for bulk action.  |
| <b>M-*</b> | <b>org-agenda-bulk-toggle-all</b><br>Toggle marks of all visible entries for bulk action.  |
| <b>%</b>   | <b>org-agenda-bulk-mark-regexp</b><br>Mark entries matching a regular expression for bulk action.  |

---

<sup>11</sup> Moving agenda lines does not persist after an agenda refresh and does not modify the contributing .org files

*B***org-agenda-bulk-action**

Bulk action: act on all marked entries in the agenda. This will prompt for another key to select the action to be applied. The prefix arg to *B* will be passed through to the *s* and *d* commands, to bulk-remove these special timestamps. By default, marks are removed after the bulk. If you want them to persist, set **org-agenda-persistent-marks** to *t* or hit *p* at the prompt.

- \** Toggle persistent marks.
- \$* Archive all selected entries.
- A* Archive entries by moving them to their respective archive siblings.
- t* Change TODO state. This prompts for a single TODO keyword and changes the state of all selected entries, bypassing blocking and suppressing logging notes (but not timestamps).
- +* Add a tag to all selected entries.
- Remove a tag from all selected entries.
- s* Schedule all items to a new date. To shift existing schedule dates by a fixed number of days, use something starting with double plus at the prompt, for example ‘++8d’ or ‘++2w’.
- d* Set deadline to a specific date.
- r* Prompt for a single refile target and move all entries. The entries will no longer be in the agenda; refresh (*g*) to bring them back.
- S* Reschedule randomly into the coming N days. N will be prompted for. With prefix arg (*C-u B S*), scatter only across weekdays.
- f* Apply a function<sup>12</sup> to marked entries. For example, the function below sets the CATEGORY property of the entries to web.

```
(defun set-category ()
  (interactive "P")
  (let* ((marker (or (org-get-at-bol 'org-hd-marker)
                    (org-agenda-error)))
        (buffer (marker-buffer marker)))
    (with-current-buffer buffer
      (save-excursion
        (save-restriction
          (widen)
          (goto-char marker)
          (org-back-to-heading t)
          (org-set-property "CATEGORY" "web"))))))
```

### Calendar commands

*c***org-agenda-goto-calendar**

Open the Emacs calendar and move to the date at the agenda cursor.

---

<sup>12</sup> You can also create persistent custom functions through **org-agenda-bulk-custom-functions**.

- c** **org-calendar-goto-agenda**  
When in the calendar, compute and show the Org mode agenda for the date at the cursor.
- i** **org-agenda-diary-entry**  
Insert a new entry into the diary, using the date at the cursor and (for block entries) the date at the mark. This will add to the Emacs diary file<sup>13</sup>, in a way similar to the *i* command in the calendar. The diary file will pop up in another window, where you can add the entry.  
If you configure **org-agenda-diary-file** to point to an Org mode file, Org will create entries (in Org mode syntax) in that file instead. Most entries will be stored in a date-based outline tree that will later make it easy to archive appointments from previous months/years. The tree will be built under an entry with a **DATE\_TREE** property, or else with years as top-level entries. Emacs will prompt you for the entry text—if you specify it, the entry will be created in **org-agenda-diary-file** without further interaction. If you directly press RET at the prompt without typing text, the target file will be shown in another window for you to finish the entry there. See also the *k r* command.
- M** **org-agenda-phases-of-moon**  
Show the phases of the moon for the three months around current date.
- S** **org-agenda-sunrise-sunset**  
Show sunrise and sunset times. The geographical location must be set with calendar variables, see the documentation for the Emacs calendar.
- C** **org-agenda-convert-date**  
Convert the date at cursor into many other cultural and historic calendars.
- H** **org-agenda-holidays**  
Show holidays for three months around the cursor date.
- M-x org-icalendar-combine-agenda-files RET**  
Export a single iCalendar file containing entries from all agenda files. This is a globally available command, and also available in the agenda menu.

### Exporting to a file

- C-x C-w** **org-agenda-write**  
Write the agenda view to a file. Depending on the extension of the selected file name, the view will be exported as HTML (**.html** or **.htm**), Postscript (**.ps**), PDF (**.pdf**), Org (**.org**) and plain text (any other extension). When exporting to Org, only the body of original headlines are exported, not subtrees or inherited tags. When called with a **C-u** prefix argument, immediately open the newly created file. Use the variable **org-agenda-exporter-settings** to set options for **ps-print** and for **htmlize** to be used during export.

### Quit and Exit

- q** **org-agenda-quit**  
アジェンダを終了し、アジェンダバッファを削除します s。

<sup>13</sup> This file is parsed for the agenda when **org-agenda-include-diary** is set.

`x` `org-agenda-exit`  
 アジェンダを終了し、アジェンダバッファとアジェンダを編集するために Emacs で読み込まれたすべてのバッファを削除する。Org-mode ファイルを読み込むためにユーザーによって作成されたバッファは削除されない。

## 10.6 Custom agenda views

Custom agenda commands serve two purposes: to store and quickly access frequently used TODO and tags searches, and to create special composite agenda buffers. Custom agenda commands will be accessible through the dispatcher (see Section 10.2 [Agenda dispatcher], page 103), just like the default commands.

### 10.6.1 Storing searches

The first application of custom searches is the definition of keyboard shortcuts for frequently used searches, either creating an agenda buffer, or a sparse tree (the latter covering of course only the current buffer).

Custom commands are configured in the variable `org-agenda-custom-commands`. You can customize this variable, for example by pressing `C-c a C`. You can also directly set it with Emacs Lisp in the Emacs init file. The following example contains all valid agenda views:

```
(setq org-agenda-custom-commands
      '(("x" agenda)
        ("y" agenda*)
        ("w" todo "WAITING")
        ("W" todo-tree "WAITING")
        ("u" tags "+boss-urgent")
        ("v" tags-todo "+boss-urgent")
        ("U" tags-tree "+boss-urgent")
        ("f" occur-tree "\\<FIXME\\>")
        ("h" . "HOME+Name tags searches") ; description for "h" prefix
        ("hl" tags "+home+Lisa")
        ("hp" tags "+home+Peter")
        ("hk" tags "+home+Kim")))
```

The initial string in each entry defines the keys you have to press after the dispatcher command `C-c a` in order to access the command. Usually this will be just a single character, but if you have many similar commands, you can also define two-letter combinations where the first character is the same in several combinations and serves as a prefix key<sup>14</sup>. The second parameter is the search type, followed by the string or regular expression to be used for the matching. The example above will therefore define:

`C-c a x` as a global search for agenda entries planned<sup>15</sup> this week/day.

`C-c a y` as a global search for agenda entries planned this week/day, but only those with an hour specification like `[h]h:mm`—think of them as appointments.

<sup>14</sup> You can provide a description for a prefix key by inserting a cons cell with the prefix and the description.

<sup>15</sup> *Planned* means here that these entries have some planning information attached to them, like a timestamp, a scheduled or a deadline string. See `org-agenda-entry-types` on how to set what planning information will be taken into account.

|                      |   |
|----------------------|---|
| <code>C-c a w</code> | as a global search for TODO entries with ‘WAITING’ as the TODO keyword  |
| <code>C-c a W</code> | as the same search, but only in the current buffer and displaying the results as a sparse tree  |
| <code>C-c a u</code> | as a global tags search for headlines marked ‘:boss:’ but not ‘:urgent:’  |
| <code>C-c a v</code> | as the same search as <code>C-c a u</code> , but limiting the search to headlines that are also TODO items  |
| <code>C-c a U</code> | as the same search as <code>C-c a u</code> , but only in the current buffer and displaying the result as a sparse tree  |
| <code>C-c a f</code> | すべてのエントリーのうちで‘FIXME’という言葉を含んでいるものを検索してツリーの抽出を行う（くどいかもしれませんが、カレントバッファだけが対象です）。   |
| <code>C-c a h</code> | as a prefix command for a HOME tags search where you have to press an additional key ( <i>l</i> , <i>p</i> or <i>k</i> ) to select a name (Lisa, Peter, or Kim) as additional tag to match. |

Note that the `*-tree` agenda views need to be called from an Org buffer as they operate on the current buffer only.

### 10.6.2 Block agenda

Another possibility is the construction of agenda views that comprise the results of *several* commands, each of which creates a block in the agenda buffer. The available commands include `agenda` for the daily or weekly agenda (as created with `C-c a a`), `alltodo` for the global TODO list (as constructed with `C-c a t`), and the matching commands discussed above: `todo`, `tags`, and `tags-todo`. Here are two examples:

```
(setq org-agenda-custom-commands
  '(("h" "Agenda and Home-related tasks"
    ((agenda "")
     (tags-todo "home")
     (tags "garden"))))
  ("o" "Agenda and Office-related tasks"
    ((agenda "")
     (tags-todo "work")
     (tags "office")))))
```

This will define `C-c a h` to create a multi-block view for stuff you need to attend to at home. The resulting agenda buffer will contain your agenda for the current week, all TODO items that carry the tag ‘home’, and also all lines tagged with ‘garden’. Finally the command `C-c a o` provides a similar view for office tasks.

### 10.6.3 Setting options for custom commands

Org mode contains a number of variables regulating agenda construction and display. The global variables define the behavior for all agenda commands, including the custom commands. However, if you want to change some settings just for a single custom view, you can do so. Setting options requires inserting a list of variable names and values at the right spot in `org-agenda-custom-commands`. For example:



```
(setq org-agenda-custom-commands
  '(("w" todo "WAITING"
    ((org-agenda-sorting-strategy '(priority-down))
     (org-agenda-prefix-format " Mixed: ")))
    ("U" tags-tree "+boss-urgent"
     ((org-show-context-detail 'minimal)))
    ("N" search ""
     ((org-agenda-files '("~/org/notes.org"))
      (org-agenda-text-search-extra-files nil)))))
```

Now the `C-c a w` command will sort the collected entries only by priority, and the prefix format is modified to just say ‘Mixed:’ instead of giving the category of the entry. The sparse tags tree of `C-c a U` will now turn out ultra-compact, because neither the headline hierarchy above the match, nor the headline following the match will be shown. The command `C-c a N` will do a text search limited to only a single file.

For command sets creating a block agenda, `org-agenda-custom-commands` has two separate spots for setting options. You can add options that should be valid for just a single command in the set, and options that should be valid for all commands in the set. The former are just added to the command entry; the latter must come after the list of command entries. Going back to the block agenda example (see Section 10.6.2 [Block agenda], page 125), let’s change the sorting strategy for the `C-c a h` commands to `priority-down`, but let’s sort the results for GARDEN tags query in the opposite order, `priority-up`. This would look like this:

```
(setq org-agenda-custom-commands
  '(("h" "Agenda and Home-related tasks"
    ((agenda)
     (tags-todo "home")
     (tags "garden"
      ((org-agenda-sorting-strategy '(priority-up)))))
     ((org-agenda-sorting-strategy '(priority-down)))))
    ("o" "Agenda and Office-related tasks"
    ((agenda)
     (tags-todo "work")
     (tags "office")))))
```

As you see, the values and parentheses setting is a little complex. When in doubt, use the customize interface to set this variable—it fully supports its structure. Just one caveat: when setting options in this interface, the *values* are just Lisp expressions. So if the value is a string, you need to add the double-quotes around the value yourself.

To control whether an agenda command should be accessible from a specific context, you can customize `org-agenda-custom-commands-contexts`. Let’s say for example that you have an agenda command “o” displaying a view that you only need when reading emails. Then you would configure this option like this:

```
(setq org-agenda-custom-commands-contexts
  '(("o" (in-mode . "message-mode"))))
```

You can also tell that the command key “o” should refer to another command key “r”. In that case, add this command key like this:

```
(setq org-agenda-custom-commands-contexts
      '(("o" "r" (in-mode . "message-mode"))))
```

See the docstring of the variable for more information.

## 10.7 Exporting agenda views

If you are away from your computer, it can be very useful to have a printed version of some agenda views to carry around. Org mode can export custom agenda views as plain text, HTML<sup>16</sup>, Postscript, PDF<sup>17</sup>, and iCalendar files. If you want to do this only occasionally, use the command

**C-x C-w** **org-agenda-write**

Write the agenda view to a file. Depending on the extension of the selected file name, the view will be exported as HTML (extension `.html` or `.htm`), Postscript (extension `.ps`), iCalendar (extension `.ics`), or plain text (any other extension). Use the variable `org-agenda-exporter-settings` to set options for `ps-print` and for `htmlize` to be used during export, for example

```
(setq org-agenda-exporter-settings
      '((ps-number-of-columns 2)
        (ps-landscape-mode t)
        (org-agenda-add-entry-text-maxlines 5)
        (htmlize-output-type 'css)))
```

If you need to export certain agenda views frequently, you can associate any custom agenda command with a list of output file names<sup>18</sup>. Here is an example that first defines custom commands for the agenda and the global TODO list, together with a number of files to which to export them. Then we define two block agenda commands and specify file names for them as well. File names can be relative to the current working directory, or absolute.

<sup>16</sup> You need to install `htmlize.el` from Hrvoje Niksic's repository. (<https://github.com/hniksic/emacs-htmlize>)

<sup>17</sup> To create PDF output, the ghostscript `ps2pdf` utility must be installed on the system. Selecting a PDF file will also create the postscript file.

<sup>18</sup> If you want to store standard views like the weekly agenda or the global TODO list as well, you need to define custom commands for them in order to be able to specify file names.

```
(setq org-agenda-custom-commands
  '(("X" agenda "" nil ("agenda.html" "agenda.ps"))
    ("Y" alltodo "" nil ("todo.html" "todo.txt" "todo.ps"))
    ("h" "Agenda and Home-related tasks"
      ((agenda "")
        (tags-todo "home")
        (tags "garden")))
      nil
      ("~/views/home.html"))
    ("o" "Agenda and Office-related tasks"
      ((agenda)
        (tags-todo "work")
        (tags "office")))
      nil
      ("~/views/office.ps" "~/calendars/office.ics"))))
```

The extension of the file name determines the type of export. If it is `.html`, Org mode will try to use the `htmlize.el` package to convert the buffer to HTML and save it to this file name. If the extension is `.ps`, `ps-print-buffer-with-faces` is used to produce Postscript output. If the extension is `.ics`, iCalendar export is run export over all files that were used to construct the agenda, and limit the export to entries listed in the agenda. Any other extension produces a plain ASCII file.

The export files are *not* created when you use one of those commands interactively because this might use too much overhead. Instead, there is a special command to produce *all* specified files in one step:

**C-c a e** **org-store-agenda-views**  
 アジェンダに関連するエクスポートファイル名を持つすべてのアジェンダビューをエクスポートします。

You can use the options section of the custom agenda commands to also set options for the export commands. For example:

```
(setq org-agenda-custom-commands
  '(("X" agenda ""
    ((ps-number-of-columns 2)
      (ps-landscape-mode t)
      (org-agenda-prefix-format " [ ] ")
      (org-agenda-with-colors nil)
      (org-agenda-remove-tags t))
    ("theagenda.ps"))))
```

This command sets two options for the Postscript exporter, to make it print in two columns in landscape format—the resulting page can be cut in two and then used in a paper agenda. The remaining settings modify the agenda prefix to omit category and scheduling information, and instead include a checkbox to check off items. We also remove the tags to make the lines compact, and we don't want to use colors for the black-and-white printer. Settings specified in `org-agenda-exporter-settings` will also apply, but the settings in `org-agenda-custom-commands` take precedence.

From the command line you may also use

`emacs -eval (org-batch-store-agenda-views) -kill`  
 or, if you need to modify some parameters<sup>19</sup>

```
emacs -eval '(org-batch-store-agenda-views           \
               org-agenda-span (quote month)         \
               org-agenda-start-day "2007-11-01"     \
               org-agenda-include-diary nil          \
               org-agenda-files (quote ("~/org/project.org")))' \
               -kill
```

which will create the agenda views restricted to the file `~/org/project.org`, without diary entries and with a 30-day extent.

You can also extract agenda information in a way that allows further processing by other programs. See Section A.10 [Extracting agenda information], page 251, for more information.

## 10.8 Using column view in the agenda

カラムビュー (see `<undefined>` [カラムビュー], page `<undefined>`) は、Org-mode ファイルの階層構造の中に組み込まれている属性を見たり編集したりするために通常は使われます。エントリーがある評価基準で収集されているアジェンダから、カラムビューを使用することは大変便利です。

`C-c C-x C-c` `org-agenda-columns`  
 Turn on column view in the agenda.

To understand how to use this properly, it is important to realize that the entries in the agenda are no longer in their proper outline environment. This causes the following issues:

1. Org needs to make a decision which **COLUMNS** format to use. Since the entries in the agenda are collected from different files, and different files may have different **COLUMNS** formats, this is a non-trivial problem. Org first checks if the variable `org-agenda-overriding-columns-format` is currently set, and if so, takes the format from there. Otherwise it takes the format associated with the first item in the agenda, or, if that item does not have a specific format—defined in a property, or in its file—it uses `org-columns-default-format`.
2. If any of the columns has a summary type defined (see Section 7.5.1.2 [Column attributes], page 68), turning on column view in the agenda will visit all relevant agenda files and make sure that the computations of this property are up to date. This is also true for the special **CLOCKSUM** property. Org will then sum the values displayed in the agenda. In the daily/weekly agenda, the sums will cover a single day; in all other views they cover the entire block. It is vital to realize that the agenda may show the same entry *twice*—for example as scheduled and as a deadline—and it may show two entries from the same hierarchy—for example a *parent* and its *child*. In these cases, the summation in the agenda will lead to incorrect results because some values will count double.
3. When the column view in the agenda shows the **CLOCKSUM**, that is always the entire clocked time for this item. So even in the daily/weekly agenda, the clocksum listed in column view may originate from times outside the current view. This has the advantage

<sup>19</sup> Quoting depends on the system you use, please check the FAQ for examples.

that you can compare these values with a column listing the planned total effort for a task—one of the major applications for column view in the agenda. If you want information about clocked time in the displayed period use clock table mode (press **R** in the agenda).

4. When the column view in the agenda shows the **CLOCKSUM\_T**, that is always today's clocked time for this item. So even in the weekly agenda, the clocksum listed in column view only originates from today. This lets you compare the time you spent on a task for today, with the time already spent —via **CLOCKSUM**—and with the planned total effort for it.

## 11 Markup for rich export

When exporting Org mode documents, the exporter tries to reflect the structure of the document as accurately as possible in the back-end. Since export targets like HTML and L<sup>A</sup>T<sub>E</sub>X allow much richer formatting, Org mode has rules on how to prepare text for rich export. This section summarizes the markup rules used in an Org mode buffer.

### 11.1 段落、改行、引用

段落は最低 1 つの空白行で区切られます。もしあなたが強制的に段落の中で改行しないなら、‘\\’を行の末尾に書いてください。

To preserve the line breaks, indentation and blank lines in a region, but otherwise use normal formatting, you can use this construct, which can also be used to format poetry.

```
#+BEGIN_VERSE
Great clouds overhead
Tiny black birds rise and fall
Snow covers Emacs

-- AlexSchroeder
#+END_VERSE
```

When quoting a passage from another document, it is customary to format this as a paragraph that is indented on both the left and the right margin. You can include quotations in Org mode documents like this:

```
#+BEGIN_QUOTE
Everything should be made as simple as possible,
but not any simpler -- Albert Einstein
#+END_QUOTE
```

もしあなたがテキストを中央寄せにしたいなら、以下を使うことができます:

```
#+BEGIN_CENTER
Everything should be made as simple as possible, \\
but not any simpler
#+END_CENTER
```

### 11.2 Emphasis and monospace

You can make words **\*bold\***, */italic/*, underlined, =verbatim= and ~code~, and, if you must, ~~+strike-through+~~. Text in the code and verbatim string is not processed for Org mode specific syntax, it is exported verbatim.

To turn off fontification for marked up text, you can set `org-fontify-emphasized-text` to `nil`. To narrow down the list of available markup syntax, you can customize `org-emphasis-alist`. To fine tune what characters are allowed before and after the markup characters, you can tweak `org-emphasis-regexp-components`. Beware that changing one of the above variables will not take effect until you reload Org, for which you may need to restart Emacs.

### 11.3 Horizontal rules

A line consisting of only dashes, and at least 5 of them, will be exported as a horizontal line.

### 11.4 画像と表

Both the native Org mode tables (see Chapter 3 [Tables], page 19) and tables formatted with the `table.el` package will be exported properly. For Org mode tables, the lines before the first horizontal separator line will become table header lines. You can use the following lines somewhere before the table to assign a caption and a label for cross references, and in the text you can refer to the object with `[[tab:basic-data]]` (see Section 4.2 [Internal links], page 38):

```
#+CAPTION: This is the caption for the next table (or link)
#+NAME:    tab:basic-data
| ... | ... |
|-----|-----|
```

Optionally, the caption can take the form:

```
#+CAPTION[Caption for list of tables]: Caption for table.
```

Some back-ends allow you to directly include images into the exported document. Org does this, if a link to an image files does not have a description part, for example `[[./img/a.jpg]]`. If you wish to define a caption for the image and maybe a label for internal cross references, make sure that the link is on a line by itself and precede it with `#+CAPTION` and `#+NAME` as follows:

```
#+CAPTION: This is the caption for the next figure link (or table)
#+NAME:    fig:SED-HR4049
[[./img/a.jpg]]
```

Such images can be displayed within the buffer. See Section 4.4 [Handling links], page 41.

Even though images and tables are prominent examples of captioned structures, the same caption mechanism can apply to many others (e.g.,  $\text{\LaTeX}$  equations, source code blocks). Depending on the export back-end, those may or may not be handled.

### 11.5 Literal examples

あなたはマークアップに依存しないリテラルの例を含めることができます。そのような例に等幅のタイプセットがあり、それはソースコードやそれに似た例向きです。

```
#+BEGIN_EXAMPLE
テキストファイルからの例。
#+END_EXAMPLE
```

そのようなブロックは、インデントされたテキストや、特にプレーンリスト構造 (See `<undefined>` [エントリー中の階層構造], page `<undefined>`.) をうまく整列させるためにインデントされるでしょう。小さな例を使う時、それを簡単にするために、あなたはコロンとそれに続く空白からなる例の行を使うことができます。それらはコロンの前に空白を追加することもできます。

```
Here is an example
: Some example from a text file.
```

If the example is source code from a programming language, or any other text that can be marked up by font-lock in Emacs, you can ask for the example to look like the fontified Emacs buffer<sup>1</sup>. This is done with the ‘src’ block, where you also need to specify the name of the major mode that should be used to fontify the example<sup>2</sup>, see Section 15.2 [Easy templates], page 229 for shortcuts to easily insert code blocks.

```
#+BEGIN_SRC emacs-lisp
  (defun org-xor (a b)
    "Exclusive or."
    (if a (not b) b))
#+END_SRC
```

Both in `example` and in `src` snippets, you can add a `-n` switch to the end of the `BEGIN` line, to get the lines of the example numbered. The `-n` takes an optional numeric argument specifying the starting line number of the block. If you use a `+n` switch, the numbering from the previous numbered snippet will be continued in the current one. The `+n` can also take a numeric argument. The value of the argument will be added to the last line of the previous block to determine the starting line number.

```
#+BEGIN_SRC emacs-lisp -n 20
  ;; this will export with line number 20
  (message "This is line 21")
#+END_SRC
#+BEGIN_SRC emacs-lisp +n 10
  ;; This will be listed as line 31
  (message "This is line 32")
#+END_SRC
```

In literal examples, Org will interpret strings like ‘(ref:name)’ as labels, and use them as targets for special hyperlinks like `[[name]]` (i.e., the reference name enclosed in single parenthesis). In HTML, hovering the mouse over such a link will remote-highlight the corresponding code line, which is kind of cool.

You can also add a `-r` switch which *removes* the labels from the source code<sup>3</sup>. With the `-n` switch, links to these references will be labeled by the line numbers from the code listing, otherwise links will use the labels with no parentheses. Here is an example:

```
#+BEGIN_SRC emacs-lisp -n -r
  (save-excursion                (ref:sc)
    (goto-char (point-min)))    (ref:jump)
#+END_SRC
In line [[(sc)]] we remember the current position. [[(jump)][Line (jump)]]
```

<sup>1</sup> This works automatically for the HTML back-end (it requires version 1.34 of the `htmlize.el` package, which you need to install). Fontified code chunks in L<sup>A</sup>T<sub>E</sub>X can be achieved using either the listings (<https://www.ctan.org/tex-archive/macros/latex/contrib/listings/?lang=en>) or the minted (<https://github.com/gpoore/minted>) package. If you use minted or listing, you must load the packages manually, for example by adding the desired package to `org-latex-packages-alist`. Refer to `org-latex-listings` for details.

<sup>2</sup> Code in ‘src’ blocks may also be evaluated either interactively or on export. See Chapter 14 [Working with source code], page 199, for more information on evaluating code blocks.

<sup>3</sup> Adding `-k` to `-n -r` will *keep* the labels in the source code while using line numbers for the links, which might be useful to explain those in an Org mode example code.



jumps to point-min.

Finally, you can use `-i` to preserve the indentation of a specific code block (see Section 14.2 [Editing source code], page 201).

もし、ラベルの構文が言語の構文と衝突した場合、`-l`スイッチを例えば `'#+BEGIN_SRC pascal -n -r -l "((%s))"'` と使ってフォーマットを変更できます。変数 `org-coderef-label-format` も見てください。

HTML export also allows examples to be published as text areas (see Section 12.9.10 [Text areas in HTML export], page 155).

Because the `#+BEGIN_...` and `#+END_...` patterns need to be added so often, shortcuts are provided using the Easy templates facility (see Section 15.2 [Easy templates], page 229).

**C-c '** Edit the source code example at point in its native mode. This works by switching to a temporary buffer with the source code. You need to exit by pressing **C-c '** again<sup>4</sup>. The edited version will then replace the old version in the Org buffer. Fixed-width regions (where each line starts with a colon followed by a space) will be edited using `artist-mode`<sup>5</sup> to allow creating ASCII drawings easily. Using this command in an empty line will create a new fixed-width region.

**C-c l** Calling `org-store-link` while editing a source code example in a temporary buffer created with **C-c '** will prompt for a label. Make sure that it is unique in the current buffer, and insert it with the proper formatting like `'(ref:label)'` at the end of the current line. Then the label is stored as a link `'(label)'`, for retrieval with **C-c C-l**.

## 11.6 Special symbols

You can use L<sup>A</sup>T<sub>E</sub>X-like syntax to insert special symbols—named entities—like `'\alpha'` to indicate the Greek letter, or `'\to'` to indicate an arrow. Completion for these symbols is available, just type `'\` and maybe a few letters, and press **M-TAB** to see possible completions. If you need such a symbol inside a word, terminate it with a pair of curly brackets. For example

Pro tip: Given a circle `\Gamma` of diameter `d`, the length of its circumference is `\pi{d}`.

A large number of entities is provided, with names taken from both HTML and L<sup>A</sup>T<sub>E</sub>X; you can comfortably browse the complete list from a dedicated buffer using the command `org-entities-help`. It is also possible to provide your own special symbols in the variable `org-entities-user`.

During export, these symbols are transformed into the native format of the exporter back-end. Strings like `\alpha` are exported as `&alpha;` in the HTML output, and as `\(\alpha\)` in the L<sup>A</sup>T<sub>E</sub>X output. Similarly, `\nbsp` becomes `&nbsp;` in HTML and `~` in L<sup>A</sup>T<sub>E</sub>X.

<sup>4</sup> Upon exit, lines starting with `'*`, `'*,`, `'#+` and `' ,#+` will get a comma prepended, to keep them from being interpreted by Org as outline nodes or special syntax. These commas will be stripped for editing with **C-c '**, and also for export.

<sup>5</sup> You may select a different mode with the variable `org-edit-fixed-width-region-mode`.

Entities may also be used as a way to escape markup in an Org document, e.g., ‘`\under{}not underlined\under`’ exports as ‘`_not underlined_`’.

If you would like to see entities displayed as UTF-8 characters, use the following command<sup>6</sup>:

**C-c C-x \** エンティティの UTF-8 での表示をトグルします。これは単純な ASCII 文字のままであるバッファの内容を変更せず、表示のみを目的として UTF-8 の文字で重ね書き (overlay) します。

In addition to regular entities defined above, Org exports in a special way<sup>7</sup> the following commonly used character combinations: ‘`\-`’ is treated as a shy hyphen, ‘`--`’ and ‘`---`’ are converted into dashes, and ‘`...`’ becomes a compact set of dots.

## 11.7 Subscripts and superscripts

‘`^`’ and ‘`_`’ are used to indicate super- and subscripts. To increase the readability of ASCII text, it is not necessary—but OK—to surround multi-character sub- and superscripts with curly braces. Those are, however, mandatory, when more than one word is involved. For example

The radius of the sun is `R_sun = 6.96 x 10^8 m`. On the other hand, the radius of Alpha Centauri is `R_{Alpha Centauri} = 1.28 x R_{sun}`.

If you write a text where the underscore is often used in a different context, Org’s convention to always interpret these as subscripts can get in your way. Configure the variable `org-use-sub-superscripts` to change this convention. For example, when setting this variable to `{}`, ‘`a_b`’ will not be interpreted as a subscript, but ‘`a_{b}`’ will.

**C-c C-x \** エンティティを UTF-8 の文字として表示することに加えて、このコマンドは下付き文字と上付き文字を WYSIWYM 方式で書式を整えます。

## 11.8 Embedded L<sup>A</sup>T<sub>E</sub>X

Plain ASCII is normally sufficient for almost all note taking. Exceptions include scientific notes, which often require mathematical symbols and the occasional formula. L<sup>A</sup>T<sub>E</sub>X<sup>8</sup> is widely used to typeset scientific documents. Org mode supports embedding L<sup>A</sup>T<sub>E</sub>X code into its files, because many academics are used to writing and reading L<sup>A</sup>T<sub>E</sub>X source code, and because it can be readily processed to produce pretty output for a number of export back-ends.

### 11.8.1 L<sup>A</sup>T<sub>E</sub>X の断片的なコード

Org mode can contain L<sup>A</sup>T<sub>E</sub>X math fragments, and it supports ways to process these for several export back-ends. When exporting to L<sup>A</sup>T<sub>E</sub>X, the code is left as it is. When exporting to HTML, Org can use either MathJax (<http://www.mathjax.org>) (see Section 12.9.9 [Math formatting in HTML export], page 155) or transcode the math into images (see see Section 11.8.2 [Previewing L<sup>A</sup>T<sub>E</sub>X fragments], page 136).

<sup>6</sup> You can turn this on by default by setting the variable `org-pretty-entities`, or on a per-file base with the `#+STARTUP` option `entitiespretty`.

<sup>7</sup> This behavior can be disabled with `- export` setting (see Section 12.2 [Export settings], page 139).

<sup>8</sup> L<sup>A</sup>T<sub>E</sub>X is a macro system based on Donald E. Knuth’s T<sub>E</sub>X system. Many of the features described here as “L<sup>A</sup>T<sub>E</sub>X” are really from T<sub>E</sub>X, but for simplicity I am blurring this distinction.

L<sup>A</sup>T<sub>E</sub>X のコード片は、特別なマークは全く必要ありません。以下のコードは L<sup>A</sup>T<sub>E</sub>X のソースコードと見なされます。

- Environments of any kind<sup>9</sup>. The only requirement is that the `\begin` statement appears on a new line, at the beginning of the line or after whitespaces only.
- Text within the usual L<sup>A</sup>T<sub>E</sub>X math delimiters. To avoid conflicts with currency specifications, single ‘\$’ characters are only recognized as math delimiters if the enclosed text contains at most two line breaks, is directly attached to the ‘\$’ characters with no whitespace in between, and if the closing ‘\$’ is followed by whitespace or punctuation (parentheses and quotes are considered to be punctuation in this context). For the other delimiters, there is no such restriction, so when in doubt, use ‘`\(...\)`’ as inline math delimiters.

例：

```
\begin{equation}
x=\sqrt{b}
\end{equation}
```

If  $a^2=b$  and  $(b=2)$ , then the solution must be either  $a=+\sqrt{2}$  or  $a=-\sqrt{2}$ .

L<sup>A</sup>T<sub>E</sub>X processing can be configured with the variable `org-export-with-latex`. The default setting is `t` which means MathJax for HTML, and no processing for ASCII and L<sup>A</sup>T<sub>E</sub>X back-ends. You can also set this variable on a per-file basis using one of these lines:

```
#+OPTIONS: tex:t           Do the right thing automatically (MathJax)
#+OPTIONS: tex:nil         Do not process LATEX fragments at all
#+OPTIONS: tex:verbatim   Verbatim export, for jsMath or so
```

### 11.8.2 Previewing L<sup>A</sup>T<sub>E</sub>X fragments

If you have a working L<sup>A</sup>T<sub>E</sub>X installation and `dvipng`, `dvisvgm` or `convert` installed<sup>10</sup>, L<sup>A</sup>T<sub>E</sub>X fragments can be processed to produce images of the typeset expressions to be used for inclusion while exporting to HTML (see see `[LATEX fragments]`, page `[undefined]`), or for inline previewing within Org mode.

You can customize the variables `org-format-latex-options` and `org-format-latex-header` to influence some aspects of the preview. In particular, the `:scale` (and for HTML export, `:html-scale`) property of the former can be used to adjust the size of the preview images.

`C-c C-x C-l`

ポイント位置の L<sup>A</sup>T<sub>E</sub>X コード片のプレビュー画像を作成し、ソースコード上に重ねて表示します。もしポイント位置にコード片がないのであれば、現在のエントリ (2つの見出しの間) の全てのコード片を処理します。前置引数を付けて呼ばれた時は、サブツリー

<sup>9</sup> When MathJax is used, only the environments recognized by MathJax will be processed. When `dvipng` program, `dvisvgm` program or `imagemagick` suite is used to create images, any L<sup>A</sup>T<sub>E</sub>X environment will be handled.

<sup>10</sup> These are respectively available at <http://sourceforge.net/projects/dvipng/>, <http://dvisvgm.bplaced.net/> and from the `imagemagick` suite. Choose the converter by setting the variable `org-preview-latex-default-process` accordingly.

全体を処理します。前置引数を 2 つ付けて呼ばれた時、またはカーソルが最初の見出しの前にある時は、バッファ全体を処理します。

**C-c C-c** オーバレイされたプレビュー画像を消去します。

You can turn on the previewing of all  $\text{\LaTeX}$  fragments in a file with

```
#+STARTUP: latexpreview
```

To disable it, simply use

```
#+STARTUP: nolatexpreview
```

### 11.8.3 Using $\text{CD}\text{\LaTeX}$ to enter math

$\text{CD}\text{\LaTeX}$  mode is a minor mode that is normally used in combination with a major  $\text{\LaTeX}$  mode like  $\text{AUCT}\text{\LaTeX}$  in order to speed-up insertion of environments and math templates. Inside Org mode, you can make use of some of the features of  $\text{CD}\text{\LaTeX}$  mode. You need to install `cdlatex.el` and `texmathp.el` (the latter comes also with  $\text{AUCT}\text{\LaTeX}$ ) from <https://staff.fnwi.uva.nl/c.dominik/Tools/cdlatex>. Don't use  $\text{CD}\text{\LaTeX}$  mode itself under Org mode, but use the light version `org-cdlatex-mode` that comes as part of Org mode. Turn it on for the current buffer with *M-x org-cdlatex-mode RET*, or for all Org files with

```
(add-hook 'org-mode-hook 'turn-on-org-cdlatex)
```

When this mode is enabled, the following features are present (for more details see the documentation of  $\text{CD}\text{\LaTeX}$  mode):

- **C-c {**により環境のテンプレートを挿入できます。
- The **TAB** key will do template expansion if the cursor is inside a  $\text{\LaTeX}$  fragment<sup>11</sup>. For example, **TAB** will expand `fr` to `\frac{}{}` and position the cursor correctly inside the first brace. Another **TAB** will get you into the second brace. Even outside fragments, **TAB** will expand environment abbreviations at the beginning of a line. For example, if you write `'equ` at the beginning of a line and press **TAB**, this abbreviation will be expanded to an `equation` environment. To get a list of all abbreviations, type *M-x cdlatex-command-help RET*.
- $\text{\LaTeX}$  コード片の中で`_`や`^`を押すと、それらの文字が一对の波括弧と一緒に挿入されます。もしあなたが`TAB`を括弧から抜け出すために使うなら、また括弧が1文字の文字かマクロのみを囲っているなら、それらは再び消去されます(`cdlatex-simplify-sub-super-script`変数に依存します)。
- Pressing the grave accent ``` followed by a character inserts math macros, also outside  $\text{\LaTeX}$  fragments. If you wait more than 1.5 seconds after the grave accent, a help window will pop up.
- Pressing the apostrophe `'` followed by another character modifies the symbol before point with an accent or a font. If you wait more than 1.5 seconds after the apostrophe, a help window will pop up. Character modification will work only inside  $\text{\LaTeX}$  fragments; outside the quote is normal.

<sup>11</sup> Org mode has a method to test if the cursor is inside such a fragment, see the documentation of the function `org-inside-LaTeX-fragment-p`.

## 12 Exporting

Sometimes, you may want to pretty print your notes, publish them on the web or even share them with people not using Org. In these cases, the Org export facilities can be used to convert your documents to a variety of other formats, while retaining as much structure (see Chapter 2 [Document structure], page 6) and markup (see Chapter 11 [Markup], page 131) as possible.

Libraries responsible for such translation are called back-ends. Org ships with the following ones

- `ascii` (ASCII format)
- `beamer` (L<sup>A</sup>T<sub>E</sub>X Beamer format)
- `html` (HTML format)
- `icalendar` (iCalendar format)
- `latex` (L<sup>A</sup>T<sub>E</sub>X format)
- `md` (Markdown format)
- `odt` (OpenDocument Text format)
- `org` (Org format)
- `texinfo` (Texinfo format)
- `man` (Man page format)

Org also uses additional libraries located in `contrib/` directory (see Section 1.2 [Installation], page 2). Users can install additional export libraries for additional formats from the Emacs packaging system. For easy discovery, these packages have a common naming scheme: `ox-NAME`, where NAME is one of the formats. For example, `ox-koma-letter` for `koma-letter` back-end.

Org loads back-ends for the following formats by default: `ascii`, `html`, `icalendar`, `latex` and `odt`.

Org can load additional back-ends either of two ways: through the `org-export-backends` variable configuration; or, by requiring the library in the Emacs init file like this:

```
(require 'ox-md)
```

### 12.1 The export dispatcher

The export dispatcher is the main interface for Org's exports. A hierarchical menu presents the currently configured export formats. Options are shown as easy toggle switches on the same screen.

Org also has a minimal prompt interface for the export dispatcher. When the variable `org-export-dispatch-use-expert-ui` is set to a non-`nil` value, Org prompts in the minibuffer. To switch back to the hierarchical menu, press `?`.

**C-c C-e**

**org-export-dispatch**

Invokes the export dispatcher interface. The options show default settings. The **C-u** prefix argument preserves options from the previous export, including any sub-tree selections.

Org exports the entire buffer by default. If the Org buffer has an active region, then Org exports just that region.

These are the export options, the key combinations that toggle them (see Section 12.2 [Export settings], page 139):

- C-a**      Toggles asynchronous export. Asynchronous export uses an external Emacs process with a specially configured initialization file to complete the exporting process in the background thereby releasing the current interface. This is particularly useful when exporting long documents.  
              Output from an asynchronous export is saved on the “the export stack”. To view this stack, call the export dispatcher with a double **C-u** prefix argument. If already in the export dispatcher menu, **&** displays the stack.  
              To make the background export process the default, customize the variable, `org-export-in-background`.
- C-b**      Toggle body-only export. Useful for excluding headers and footers in the export. Affects only those back-end formats that have such sections—like `<head>...</head>` in HTML.
- C-s**      Toggle sub-tree export. When turned on, Org exports only the sub-tree starting from the cursor position at the time the export dispatcher was invoked. Org uses the top heading of this sub-tree as the document’s title. If the cursor is not on a heading, Org uses the nearest enclosing header. If the cursor is in the document preamble, Org signals an error and aborts export.  
              To make the sub-tree export the default, customize the variable, `org-export-initial-scope`.
- C-v**      Toggle visible-only export. Useful for exporting only visible parts of an Org document by adjusting outline visibility settings.

## 12.2 Export settings

Export options can be set: globally with variables; for an individual file by making variables buffer-local with in-buffer settings (see Section 15.6 [In-buffer settings], page 231), by setting individual keywords, or by specifying them in a compact form with the `#+OPTIONS` keyword; or for a tree by setting properties (see Chapter 7 [Properties and columns], page 64). Options set at a specific level override options set at a more general level.

In-buffer settings may appear anywhere in the file, either directly or indirectly through a file included using `'#+SETUPFILE: filename or URL'` syntax. Option keyword sets tailored to a particular back-end can be inserted from the export dispatcher (see Section 12.1 [The export dispatcher], page 138) using the **Insert template** command by pressing **#**. To insert keywords individually, a good way to make sure the keyword is correct is to type **#+** and then to use *M-TAB*<sup>1</sup> for completion.

The export keywords available for every back-end, and their equivalent global variables, include:

**'AUTHOR'**    The document author (`user-full-name`).

---

<sup>1</sup> Many desktops intercept *M-TAB* to switch windows. Use *C-M-i* or *ESC TAB* instead.

- ‘CREATOR’ Entity responsible for output generation (`org-export-creator-string`).
- ‘DATE’ A date or a time-stamp<sup>2</sup>.
- ‘EMAIL’ The email address (`user-mail-address`).
- ‘LANGUAGE’  
Language to use for translating certain strings (`org-export-default-language`). With ‘`#+LANGUAGE: fr`’, for example, Org translates *Table of contents* to the French *Table des mati<sup>e</sup>res*.
- ‘SELECT\_TAGS’  
The default value is `:export:`. When a tree is tagged with `:export:` (`org-export-select-tags`), Org selects that tree and its sub-trees for export. Org excludes trees with `:noexport:` tags, see below. When selectively exporting files with `:export:` tags set, Org does not export any text that appears before the first headline.
- ‘EXCLUDE\_TAGS’  
The default value is `:noexport:`. When a tree is tagged with `:noexport:` (`org-export-exclude-tags`), Org excludes that tree and its sub-trees from export. Entries tagged with `:noexport:` will be unconditionally excluded from the export, even if they have an `:export:` tag. Even if a sub-tree is not exported, Org will execute any code blocks contained in them.
- ‘TITLE’ Org displays this title. For long titles, use multiple `#+TITLE` lines.
- ‘EXPORT\_FILE\_NAME’  
The name of the output file to be generated. Otherwise, Org generates the file name based on the buffer name and the extension based on the back-end format.
- The `#+OPTIONS` keyword is a compact form. To configure multiple options, use several `#+OPTIONS` lines. `#+OPTIONS` recognizes the following arguments.
- ‘:’ Toggle smart quotes (`org-export-with-smart-quotes`). Depending on the language used, when activated, Org treats pairs of double quotes as primary quotes, pairs of single quotes as secondary quotes, and single quote marks as apostrophes.
  - \*: Toggle emphasized text (`org-export-with-emphasize`).
  - : Toggle conversion of special strings (`org-export-with-special-strings`).
  - :: Toggle fixed-width sections (`org-export-with-fixed-width`).
  - <: Toggle inclusion of time/date active/inactive stamps (`org-export-with-timestamps`).
  - \n: Toggles whether to preserve line breaks (`org-export-preserve-breaks`).
  - ^: Toggle T<sub>E</sub>X-like syntax for sub- and superscripts. If you write “`^:{ }`”, ‘`a_{b}`’ will be interpreted, but the simple ‘`a_b`’ will be left as it is (`org-export-with-sub-superscripts`).

---

<sup>2</sup> The variable `org-export-date-timestamp-format` defines how this time-stamp will be exported.

- arch:** Configure how archived trees are exported. When set to **headline**, the export process skips the contents and processes only the headlines (**org-export-with-archived-trees**).
- author:** Toggle inclusion of author name into exported file (**org-export-with-author**).
- broken-links:** Toggles if Org should continue exporting upon finding a broken internal link. When set to **mark**, Org clearly marks the problem link in the output (**org-export-with-broken-links**).
- c:** Toggle inclusion of CLOCK keywords (**org-export-with-clocks**).
- creator:** Toggle inclusion of creator information in the exported file (**org-export-with-creator**).
- d:** Toggles inclusion of drawers, or list of drawers to include, or list of drawers to exclude (**org-export-with-drawers**).
- date:** Toggle inclusion of a date into exported file (**org-export-with-date**).
- e:** Toggle inclusion of entities (**org-export-with-entities**).
- email:** Toggle inclusion of the author's e-mail into exported file (**org-export-with-email**).
- f:** Toggle the inclusion of footnotes (**org-export-with-footnotes**).
- H:** Set the number of headline levels for export (**org-export-headline-levels**). Below that level, headlines are treated differently. In most back-ends, they become list items.
- inline:** Toggle inclusion of inlinetasks (**org-export-with-inlinetasks**).
- num:** Toggle section-numbers (**org-export-with-section-numbers**). When set to number 'n', Org numbers only those headlines at level 'n' or above. Setting **UNNUMBERED** property to non-nil disables numbering of a heading. Since sub-headings inherit from this property, it affects their numbering, too.
- p:** Toggle export of planning information (**org-export-with-planning**). "Planning information" comes from lines located right after the headline and contain any combination of these cookies: **SCHEDULED:**, **DEADLINE:**, or **CLOSED:**.
- pri:** Toggle inclusion of priority cookies (**org-export-with-priority**).
- prop:** Toggle inclusion of property drawers, or list the properties to include (**org-export-with-properties**).
- stat:** Toggle inclusion of statistics cookies (**org-export-with-statistics-cookies**).
- tags:** Toggle inclusion of tags, may also be **not-in-toc** (**org-export-with-tags**).
- tasks:** Toggle inclusion of tasks (TODO items); or **nil** to remove all tasks; or **todo** to remove DONE tasks; or list the keywords to keep (**org-export-with-tasks**).
- tex:** **nil** does not export; **t** exports; **verbatim** keeps everything in verbatim (**org-export-with-latex**).



**timestamp:** Toggle inclusion of the creation time in the exported file (`org-export-time-stamp-file`).

**title:** Toggle inclusion of title (`org-export-with-title`).

**toc:** Toggle inclusion of the table of contents, or set the level limit (`org-export-with-toc`).

**todo:** Toggle inclusion of TODO keywords into exported text (`org-export-with-todo-keywords`).

**|:** Toggle inclusion of tables (`org-export-with-tables`).

When exporting sub-trees, special node properties in them can override the above keywords. They are special because they have an ‘EXPORT\_’ prefix. For example, ‘DATE’ and ‘EXPORT\_FILE\_NAME’ keywords become, respectively, ‘EXPORT\_DATE’ and ‘EXPORT\_FILE\_NAME’. Except for ‘SETUPFILE’, all other keywords listed above have an ‘EXPORT\_’ equivalent.

If `org-export-allow-bind-keywords` is non-`nil`, Emacs variables can become buffer-local during export by using the BIND keyword. Its syntax is ‘`#+BIND: variable value`’. This is particularly useful for in-buffer settings that cannot be changed using keywords.

## 12.3 Table of contents

Org normally inserts the table of contents directly before the first headline of the file. Org sets the TOC depth the same as the headline levels in the file. Use a lower number for lower TOC depth. To turn off TOC entirely, use `nil`. This is configured in the `org-export-with-toc` variable or as keywords in an Org file as:

```
#+OPTIONS: toc:2          only include two levels in TOC
#+OPTIONS: toc:nil        no default TOC at all
```

To move the table of contents to a different location, first turn off the default with `org-export-with-toc` variable or with `#+OPTIONS: toc:nil`. Then insert `#+TOC: headlines N` at the desired location(s).

```
#+OPTIONS: toc:nil        no default TOC
...
#+TOC: headlines 2        insert TOC here, with two headline levels
```

To adjust the TOC depth for a specific section of the Org document, append an additional ‘local’ parameter. This parameter becomes a relative depth for the current level.

Note that for this feature to work properly in L<sup>A</sup>T<sub>E</sub>X export, the Org file requires the inclusion of the `titletoc` package. Because of compatibility issues, `titletoc` has to be loaded *before* `hyperref`. Customize the `org-latex-default-packages-alist` variable.

\* Section `#+TOC: headlines 1 local` insert local TOC, with direct children only

Use the TOC keyword to generate list of tables (resp. all listings) with captions.

```
#+TOC: listings          build a list of listings
#+TOC: tables            build a list of tables
```

Normally Org uses the headline for its entry in the table of contents. But with `ALT_TITLE` property, a different entry can be specified for the table of contents.

## 12.4 Include files

Include other files during export. For example, to include your `.emacs` file, you could use:

```
#+INCLUDE: "~/emacs" src emacs-lisp
```

The first parameter is the file name to include. The optional second parameter specifies the block type: `example`, `export` or `src`. The optional third parameter specifies the source code language to use for formatting the contents. This is relevant to both `export` and `src` block types.

If an include file is specified as having a markup language, Org neither checks for valid syntax nor changes the contents in any way. For `example` and `src` blocks, Org code-escapes the contents before inclusion.

If an include file is not specified as having any markup language, Org assumes it be in Org format and proceeds as usual with a few exceptions. Org makes the footnote labels (see Section 2.10 [Footnotes], page 16) in the included file local to that file. The contents of the included file will belong to the same structure—headline, item—containing the `INCLUDE` keyword. In particular, headlines within the file will become children of the current section. That behavior can be changed by providing an additional keyword parameter, `:minlevel`. It shifts the headlines in the included file to become the lowest level. For example, this syntax makes the included file a sibling of the current top-level headline:

```
#+INCLUDE: "~/my-book/chapter2.org" :minlevel 1
```

Inclusion of only portions of files are specified using ranges parameter with `:lines` keyword. The line at the upper end of the range will not be included. The start and/or the end of the range may be omitted to use the obvious defaults.

```
#+INCLUDE: "~/emacs" :lines "5-10"    5 行目から 10 行目までを含みます。10 行目
を
除きます。
```

```
#+INCLUDE: "~/emacs" :lines "-10"    1 行目から 10 行目までを含みます。10 行目
を
除きます。
```

```
#+INCLUDE: "~/emacs" :lines "10-"    10 行目から EOF までを含みます。
```

Inclusions may specify a file-link to extract an object matched by `org-link-search`<sup>3</sup> (see Section 4.7 [Search options], page 45).

To extract only the contents of the matched object, set `:only-contents` property to non-`nil`. This will omit any planning lines or property drawers. The ranges for `:lines` keyword are relative to the requested element. Some examples:

```
#+INCLUDE: "./paper.org::#theory" :only-contents t
  Include the body of the heading with the custom id 'theory'
#+INCLUDE: "./paper.org::mytable"  Include named element.
#+INCLUDE: "./paper.org::*conclusion" :lines 1-20
  Include the first 20 lines of the headline named 'conclusion'.
```

`C-c '` Visit the include file at point.

<sup>3</sup> Note that `org-link-search-must-match-exact-headline` is locally bound to non-`nil`. Therefore, `org-link-search` only matches headlines and named elements.

## 12.5 Macro replacement

Macros replace text snippets during export. Macros are defined globally in `org-export-global-macros`, or document-wise with the following syntax:

```
#+MACRO: name replacement text $1, $2 are arguments
```

which can be referenced using `{{{name(arg1, arg2)}}}`<sup>4</sup>.

Org recognizes macro references in following Org markup areas: paragraphs, headlines, verse blocks, tables cells and lists. Org also recognizes macro references in keywords, such as `#+CAPTION`, `#+TITLE`, `#+AUTHOR`, `#+DATE`, and for some back-end specific export options.

Org comes with following pre-defined macros:

```
{{{title}}}
{{{author}}}
{{{email}}}
```

Org replaces these macro references with available information at the time of export.

```
{{{date}}}
{{{date(FORMAT)}}}
```

This macro refers to the `#+DATE` keyword. *FORMAT* is an optional argument to the `{{{date}}}` macro that will be used only if `#+DATE` is a single timestamp. *FORMAT* should be a format string understood by `format-time-string`.

```
{{{time(FORMAT)}}}
{{{modification-time(FORMAT, VC)}}}
```

These macros refer to the document's date and time of export and date and time of modification. *FORMAT* is a string understood by `format-time-string`. If the second argument to the `modification-time` macro is non-`nil`, Org uses `vc.el` to retrieve the document's modification time from the version control system. Otherwise Org reads the file attributes.

```
{{{input-file}}}
```

This macro refers to the filename of the exported file.

```
{{{property(PROPERTY-NAME)}}}
{{{property(PROPERTY-NAME, SEARCH-OPTION)}}}
```

This macro returns the value of property *PROPERTY-NAME* in the current entry. If *SEARCH-OPTION* (see Section 4.7 [Search options], page 45) refers to a remote entry, that will be used instead.

```
{{{n}}}
{{{n(NAME)}}}
{{{n(NAME, ACTION)}}}
```

This macro implements custom counters by returning the number of times the macro has been expanded so far while exporting the buffer. You can create more than one counter using different *NAME* values. If *ACTION* is `-`, previous value

---

<sup>4</sup> Since commas separate the arguments, commas within arguments have to be escaped with the backslash character. So only those backslash characters before a comma need escaping with another backslash character.

of the counter is held, i.e. the specified counter is not incremented. If the value is a number, the specified counter is set to that value. If it is any other non-empty string, the specified counter is reset to 1. You may leave *NAME* empty to reset the default counter.

The surrounding brackets can be made invisible by setting `org-hide-macro-markers` `non-nil`.

Org expands macros at the very beginning of the export process.

## 12.6 Comment lines

Lines starting with zero or more whitespace characters followed by one ‘#’ and a whitespace are treated as comments and, as such, are not exported.

Likewise, regions surrounded by `#+BEGIN_COMMENT ... #+END_COMMENT` are not exported.

Finally, a ‘COMMENT’ keyword at the beginning of an entry, but after any other keyword or priority cookie, comments out the entire subtree. In this case, the subtree is not exported and no code block within it is executed either<sup>5</sup>. The command below helps changing the comment status of a headline.

`C-c ;`      Toggle the ‘COMMENT’ keyword at the beginning of an entry.

## 12.7 ASCII/Latin-1/UTF-8 export

ASCII export produces an output file containing only plain ASCII characters. This is the most simplest and direct text output. It does not contain any Org markup either. Latin-1 and UTF-8 export use additional characters and symbols available in these encoding standards. All three of these export formats offer the most basic of text output for maximum portability.

On export, Org fills and justifies text according to the text width set in `org-ascii-text-width`.

Org exports links using a footnote-like style where the descriptive part is in the text and the link is in a note before the next heading. See the variable `org-ascii-links-to-notes` for details.

### ASCII export commands

`C-c C-e t a/l/u`      `org-ascii-export-to-ascii`  
Export as an ASCII file with a `.txt` extension. For `myfile.org`, Org exports to `myfile.txt`, overwriting without warning. For `myfile.txt`, Org exports to `myfile.txt.txt` in order to prevent data loss.

`C-c C-e t A/L/U`      `org-ascii-export-as-ascii`  
Export to a temporary buffer. Does not create a file.

---

<sup>5</sup> For a less drastic behavior, consider using a select tag (see Section 12.2 [Export settings], page 139) instead.

## ASCII specific export settings

The ASCII export back-end has one extra keyword for customizing ASCII output. Setting this keyword works similar to the general options (see Section 12.2 [Export settings], page 139).

‘SUBTITLE’

The document subtitle. For long subtitles, use multiple `#+SUBTITLE` lines in the Org file. Org prints them on one continuous line, wrapping into multiple lines if necessary.

## Header and sectioning structure

Org converts the first three outline levels into headlines for ASCII export. The remaining levels are turned into lists. To change this cut-off point where levels become lists, see Section 12.2 [Export settings], page 139.

## Quoting ASCII text

To insert text within the Org file by the ASCII back-end, use one the following constructs, inline, keyword, or export block:

Inline text `@@ascii:and additional text@@` within a paragraph.

`#+ASCII: Some text`

`#+BEGIN_EXPORT ascii`

Org exports text in this block only when using ASCII back-end.

`#+END_EXPORT`

## ASCII specific attributes

ASCII back-end recognizes only one attribute, `:width`, which specifies the width of a horizontal rule in number of characters. The keyword and syntax for specifying widths is:

`#+ATTR_ASCII: :width 10`

-----

## ASCII special blocks

Besides `#+BEGIN_CENTER` blocks (see Section 11.1 [Paragraphs], page 131), ASCII back-end has these two left and right justification blocks:

`#+BEGIN_JUSTIFYLEFT`

It's just a jump to the left...

`#+END_JUSTIFYLEFT`

`#+BEGIN_JUSTIFYRIGHT`

...and then a step to the right.

`#+END_JUSTIFYRIGHT`

## 12.8 Beamer export

Org uses *Beamer* export to convert an Org file tree structure into a high-quality interactive slides for presentations. *Beamer* is a  $\text{\LaTeX}$  document class for creating presentations in PDF, HTML, and other popular display formats.

### 12.8.1 Beamer export commands

`C-c C-e l b` `org-beamer-export-to-latex`  
Export as  $\text{\LaTeX}$  file with a `.tex` extension. For `myfile.org`, Org exports to `myfile.tex`, overwriting without warning.

`C-c C-e l B` `org-beamer-export-as-latex`  
Export to a temporary buffer. Does not create a file.

`C-c C-e l P` `org-beamer-export-to-pdf`  
Export as  $\text{\LaTeX}$  file and then convert it to PDF format.

`C-c C-e l O`  
Export as  $\text{\LaTeX}$  file, convert it to PDF format, and then open the PDF file.

### 12.8.2 Beamer specific export settings

Beamer export back-end has several additional keywords for customizing Beamer output. These keywords work similar to the general options settings (see Section 12.2 [Export settings], page 139).

`'BEAMER_THEME'`  
The Beamer layout theme (`org-beamer-theme`). Use square brackets for options. For example:  
`#+BEAMER_THEME: Rochester [height=20pt]`

`'BEAMER_FONT_THEME'`  
The Beamer font theme.

`'BEAMER_INNER_THEME'`  
The Beamer inner theme.

`'BEAMER_OUTER_THEME'`  
The Beamer outer theme.

`'BEAMER_HEADER'`  
Arbitrary lines inserted in the preamble, just before the `'hyperref'` settings.

`'DESCRIPTION'`  
The document description. For long descriptions, use multiple `#+DESCRIPTION` keywords. By default, `'hyperref'` inserts `#+DESCRIPTION` as metadata. Use `org-latex-hyperref-template` to configure document metadata. Use `org-latex-title-command` to configure typesetting of description as part of front matter.

`'KEYWORDS'`  
The keywords for defining the contents of the document. Use multiple `#+KEYWORDS` lines if necessary. By default, `'hyperref'` inserts `#+KEYWORDS`

as metadata. Use `org-latex-hyperref-template` to configure document metadata. Use `org-latex-title-command` to configure typesetting of keywords as part of front matter.

#### ‘SUBTITLE’

Document’s subtitle. For typesetting, use `org-beamer-subtitle-format` string. Use `org-latex-hyperref-template` to configure document metadata. Use `org-latex-title-command` to configure typesetting of subtitle as part of front matter.

### 12.8.3 Sectioning, Frames and Blocks in Beamer

Org transforms heading levels into Beamer’s sectioning elements, frames and blocks. Any Org tree with a not-too-deep-level nesting should in principle be exportable as a Beamer presentation.

- Org headlines become Beamer frames when the heading level in Org is equal to `org-beamer-frame-level` or H value in an `OPTIONS` line (see Section 12.2 [Export settings], page 139).

Org overrides headlines to frames conversion for the current tree of an Org file if it encounters the `BEAMER_ENV` property set to `frame` or `fullframe`. Org ignores whatever `org-beamer-frame-level` happens to be for that headline level in the Org tree. In Beamer terminology, a `fullframe` is a frame without its title.

- Org exports a Beamer frame’s objects as `block` environments. Org can enforce wrapping in special block types when `BEAMER_ENV` property is set<sup>6</sup>. For valid values see `org-beamer-environments-default`. To add more values, see `org-beamer-environments-extra`.
- If `BEAMER_ENV` is set to `appendix`, Org exports the entry as an appendix. When set to `note`, Org exports the entry as a note within the frame or between frames, depending on the entry’s heading level. When set to `noteNH`, Org exports the entry as a note without its title. When set to `againframe`, Org exports the entry with `\againframe` command, which makes setting the `BEAMER_REF` property mandatory because `\againframe` needs frame to resume.

When `ignoreheading` is set, Org export ignores the entry’s headline but not its content. This is useful for inserting content between frames. It is also useful for properly closing a `column` environment.

When `BEAMER_ACT` is set for a headline, Org export translates that headline as an overlay or action specification. When enclosed in square brackets, Org export makes the overlay specification a default. Use `BEAMER_OPT` to set any options applicable to the current Beamer frame or block. The Beamer export back-end wraps with appropriate angular or square brackets. It also adds the `fragile` option for any code that may require a verbatim block.

To create a column on the Beamer slide, use the `BEAMER_COL` property for its headline in the Org file. Set the value of `BEAMER_COL` to a decimal number representing the fraction of the total text width. Beamer export uses this value to set the column’s width and fills the column with the contents of the Org entry. If the Org entry has no specific environment

---

<sup>6</sup> If `BEAMER_ENV` is set, Org export adds `:B_environment:` tag to make it visible. The tag serves as a visual aid and has no semantic relevance.

defined, Beamer export ignores the heading. If the Org entry has a defined environment, Beamer export uses the heading as title. Behind the scenes, Beamer export automatically handles L<sup>A</sup>T<sub>E</sub>X column separations for contiguous headlines. To manually adjust them for any unique configurations needs, use the `BEAMER_ENV` property.

### 12.8.4 Beamer specific syntax

Since Org's Beamer export back-end is an extension of the L<sup>A</sup>T<sub>E</sub>X back-end, it recognizes other L<sup>A</sup>T<sub>E</sub>X specific syntax—for example, `#+LATEX:` or `#+ATTR_LATEX:`. See [L<sup>A</sup>T<sub>E</sub>X export], page [undefined], for details.

Beamer export wraps the table of contents generated with `toc:t` OPTION keyword in a `frame` environment. Beamer export does not wrap the table of contents generated with `TOC` keyword (see Section 12.3 [Table of contents], page 142). Use square brackets for specifying options.

```
#+TOC: headlines [currentsection]
```

Insert Beamer-specific code using the following constructs:

```
#+BEAMER: \pause
```

```
#+BEGIN_EXPORT beamer
```

```
Only Beamer export back-end will export this line.
```

```
#+END_BEAMER
```

```
Text @@beamer:some code@@ within a paragraph.
```

Inline constructs, such as the last one above, are useful for adding overlay specifications to objects with `bold`, `item`, `link`, `radio-target` and `target` types. Enclose the value in angular brackets and place the specification at the beginning the object as shown in this example:

```
A *@@beamer:<2->@@useful* feature
```

Beamer export recognizes the `ATTR_BEAMER` keyword with the following attributes from Beamer configurations: `:environment` for changing local Beamer environment, `:overlay` for specifying Beamer overlays in angular or square brackets, and `:options` for inserting optional arguments.

```
#+ATTR_BEAMER: :environment nonindentlist
```

```
- item 1, not indented
```

```
- item 2, not indented
```

```
- item 3, not indented
```

```
#+ATTR_BEAMER: :overlay <+>
```

```
- item 1
```

```
- item 2
```

```
#+ATTR_BEAMER: :options [Lagrange]
```

```
Let  $G$  be a finite group, and let  $H$  be
```

```
a subgroup of  $G$ . Then the order of  $H$  divides the order of  $G$ .
```

### 12.8.5 Editing support

The `org-beamer-mode` is a special minor mode for faster editing of Beamer documents.



```
#+STARTUP: beamer
```

*C-c C-b*

`org-beamer-select-environment`

The `org-beamer-mode` provides this key for quicker selections in Beamer normal environments, and for selecting the `BEAMER_COL` property.

### 12.8.6 A Beamer example

Here is an example of an Org document ready for Beamer export.

```
#+TITLE: Example Presentation
#+AUTHOR: Carsten Dominik
#+OPTIONS: H:2 toc:t num:t
#+LATEX_CLASS: beamer
#+LATEX_CLASS_OPTIONS: [presentation]
#+BEAMER_THEME: Madrid
#+COLUMNS: %45ITEM %10BEAMER_ENV(Env) %10BEAMER_ACT(Act) %4BEAMER_COL(Col) %8BEAMER

* This is the first structural section

** Frame 1
*** Thanks to Eric Fraga                                     :B_block:
    :PROPERTIES:
    :BEAMER_COL: 0.48
    :BEAMER_ENV: block
    :END:
    for the first viable Beamer setup in Org
*** Thanks to everyone else                                   :B_block:
    :PROPERTIES:
    :BEAMER_COL: 0.48
    :BEAMER_ACT: <2->
    :BEAMER_ENV: block
    :END:
    for contributing to the discussion
**** This will be formatted as a beamer note                  :B_note:
    :PROPERTIES:
    :BEAMER_env: note
    :END:
** Frame 2 (where we will not use columns)
*** Request
    Please test this stuff!
```

## 12.9 HTML export

Org mode contains an HTML exporter with extensive HTML formatting compatible with XHTML 1.0 strict standard.

### 12.9.1 HTML エクスポートのコマンド

`C-c C-e h h` `org-html-export-to-html`  
 Export as HTML file with a `.html` extension. For `myfile.org`, Org exports to `myfile.html`, overwriting without warning. `C-c C-e h o` Exports to HTML and opens it in a web browser.

`C-c C-e h H` `org-html-export-as-html`  
 Exports to a temporary buffer. Does not create a file.

### 12.9.2 HTML Specific export settings

HTML export has a number of keywords, similar to the general options settings described in Section 12.2 [Export settings], page 139.

**‘DESCRIPTION’**

This is the document’s description, which the HTML exporter inserts it as a HTML meta tag in the HTML file. For long descriptions, use multiple `#+DESCRIPTION` lines. The exporter takes care of wrapping the lines properly.

**‘HTML\_DOCTYPE’**

Specify the document type, for example: HTML5 (`org-html-doctype`).

**‘HTML\_CONTAINER’**

Specify the HTML container, such as ‘div’, for wrapping sections and elements (`org-html-container-element`).

**‘HTML\_LINK\_HOME’**

The URL for home link (`org-html-link-home`).

**‘HTML\_LINK\_UP’**

The URL for the up link of exported HTML pages (`org-html-link-up`).

**‘HTML\_MATHJAX’**

Options for MathJax (`org-html-mathjax-options`). MathJax is used to typeset  $\text{\LaTeX}$  math in HTML documents. See Section 12.9.9 [Math formatting in HTML export], page 155, for an example.

**‘HTML\_HEAD’**

Arbitrary lines for appending to the HTML document’s head (`org-html-head`).

**‘HTML\_HEAD\_EXTRA’**

More arbitrary lines for appending to the HTML document’s head (`org-html-head-extra`).

**‘KEYWORDS’**

Keywords to describe the document’s content. HTML exporter inserts these keywords as HTML meta tags. For long keywords, use multiple `#+KEYWORDS` lines.

**‘LATEX\_HEADER’**

Arbitrary lines for appending to the preamble; HTML exporter appends when transcoding  $\text{\LaTeX}$  fragments to images (see Section 12.9.9 [Math formatting in HTML export], page 155).

‘SUBTITLE’

The document’s subtitle. HTML exporter formats subtitle if document type is ‘HTML5’ and the CSS has a ‘subtitle’ class.

Some of these keywords are explained in more detail in the following sections of the manual.

### 12.9.3 HTML doctypes

Org can export to various (X)HTML flavors.

Set the `org-html-doctype` variable for different (X)HTML variants. Depending on the variant, the HTML exporter adjusts the syntax of HTML conversion accordingly. Org includes the following ready-made variants:

- “html4-strict”
- “html4-transitional”
- “html4-frameset”
- “xhtml-strict”
- “xhtml-transitional”
- “xhtml-frameset”
- “xhtml-11”
- “html5”
- “xhtml5”

See the variable `org-html-doctype-alist` for details. The default is “xhtml-strict”.

Org’s HTML exporter does not by default enable new block elements introduced with the HTML5 standard. To enable them, set `org-html-html5-fancy` to non-nil. Or use an `OPTIONS` line in the file to set `html5-fancy`. HTML5 documents can now have arbitrary `#+BEGIN` and `#+END` blocks. For example:

```
#+BEGIN_aside
Lorem ipsum
#+END_aside
```

Will export to:

```
<aside>
  <p>Lorem ipsum</p>
</aside>
```

While this:

```
#+ATTR_HTML: :controls controls :width 350
#+BEGIN_video
#+HTML: <source src="movie.mp4" type="video/mp4">
#+HTML: <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
#+END_video
```

Exports to:

```
<video controls="controls" width="350">
  <source src="movie.mp4" type="video/mp4">
```

```
<source src="movie.ogg" type="video/ogg">
<p>Your browser does not support the video tag.</p>
</video>
```

When special blocks do not have a corresponding HTML5 element, the HTML exporter reverts to standard translation (see `org-html-html5-elements`). For example, `#+BEGIN_lederhosen` exports to `<div class="lederhosen">`.

Special blocks cannot have headlines. For the HTML exporter to wrap the headline and its contents in `<section>` or `<article>` tags, set the `HTML_CONTAINER` property for the headline.

### 12.9.4 HTML preamble and postamble

The HTML exporter has delineations for preamble and postamble. The default value for `org-html-preamble` is `t`, which makes the HTML exporter insert the preamble. See the variable `org-html-preamble-format` for the format string.

Set `org-html-preamble` to a string to override the default format string. If the string is a function, the HTML exporter expects the function to return a string upon execution. The HTML exporter inserts this string in the preamble. The HTML exporter will not insert a preamble if `org-html-preamble` is set `nil`.

The default value for `org-html-postamble` is `auto`, which makes the HTML exporter build a postamble from looking up author's name, email address, creator's name, and date. Set `org-html-postamble` to `t` to insert the postamble in the format specified in the `org-html-postamble-format` variable. The HTML exporter will not insert a postamble if `org-html-postamble` is set to `nil`.

### 12.9.5 Quoting HTML tags

The HTML export back-end transforms `<` and `>` to `&lt;` and `&gt;`. To include raw HTML code in the Org file so the HTML export back-end can insert that HTML code in the output, use this inline syntax: `@@html:`. For example: `@@html:<b>@@bold text@@html:</b>@@`. For larger raw HTML code blocks, use these HTML export code blocks:

```
#+HTML: Literal HTML code for export
```

or

```
#+BEGIN_EXPORT html
All lines between these markers are exported literally
#+END_EXPORT
```

### 12.9.6 Links in HTML export

The HTML export back-end transforms Org's internal links (see Section 4.2 [Internal links], page 38) to equivalent HTML links in the output. The back-end similarly handles Org's automatic links created by radio targets (see Section 4.2.1 [Radio targets], page 39) similarly. For Org links to external files, the back-end transforms the links to *relative* paths.

For Org links to other `.org` files, the back-end automatically changes the file extension to `.html` and makes file paths relative. If the `.org` files have an equivalent `.html` version at the same location, then the converted links should work without any further manual

intervention. However, to disable this automatic path translation, set `org-html-link-org-files-as-html` to `nil`. When disabled, the HTML export back-end substitutes the ‘id:’-based links in the HTML output. For more about linking files when publishing to a directory, see Section 13.1.6 [Publishing links], page 194.

Org files can also have special directives to the HTML export back-end. For example, by using `#+ATTR_HTML` lines to specify new format attributes to `<a>` or `<img>` tags. This example shows changing the link’s `title` and `style`:

```
#+ATTR_HTML: :title The Org mode homepage :style color:red;
[[https://orgmode.org]]
```

### 12.9.7 Tables in HTML export

The HTML export back-end uses `org-html-table-default-attributes` when exporting Org tables to HTML. By default, the exporter does not draw frames and cell borders. To change for this for a table, use the following lines before the table in the Org file:

```
#+CAPTION: This is a table with lines around and between cells
#+ATTR_HTML: :border 2 :rules all :frame border
```

The HTML export back-end preserves column groupings in Org tables (see Section 3.3 [Column groups], page 23) when exporting to HTML.

Additional options for customizing tables for HTML export.

`org-html-table-align-individual-fields`

Non-`nil` attaches style attributes for alignment to each table field.

`org-html-table-caption-above`

Non-`nil` places caption string at the beginning of the table.

`org-html-table-data-tags`

Opening and ending tags for table data fields.

`org-html-table-default-attributes`

Default attributes and values for table tags.

`org-html-table-header-tags`

Opening and ending tags for table’s header fields.

`org-html-table-row-tags`

Opening and ending tags for table rows.

`org-html-table-use-header-tags-for-first-column`

Non-`nil` formats column one in tables with header tags.

### 12.9.8 Images in HTML export

The HTML export back-end has features to convert Org image links to HTML inline images and HTML clickable image links.

When the link in the Org file has no description, the HTML export back-end by default in-lines that image. For example: ‘[[file:myimg.jpg]]’ is in-lined, while ‘[[file:myimg.jpg][the image]]’ links to the text, ‘the image’.

For more details, see the variable `org-html-inline-images`.

On the other hand, if the description part of the Org link is itself another link, such as `file:` or `http:` URL pointing to an image, the HTML export back-end in-lines this image and links to the main image. This Org syntax enables the back-end to link low-resolution thumbnail to the high-resolution version of the image, as shown in this example:

```
[[file:highres.jpg][file:thumb.jpg]]
```

To change attributes of in-lined images, use `#+ATTR_HTML` lines in the Org file. This example shows realignment to right, and adds `alt` and `title` attributes in support of text viewers and modern web accessibility standards.

```
#+CAPTION: A black cat stalking a spider
#+ATTR_HTML: :alt cat/spider image :title Action! :align right
[[./img/a.jpg]]
```

The HTML export back-end copies the `http` links from the Org file as is.

### 12.9.9 Math formatting in HTML export

$\LaTeX$  math snippets (see `<undefined> [LATEX fragments], page <undefined>`) can be displayed in two different ways on HTML pages. The default is to use MathJax (<http://www.mathjax.org>) which should work out of the box with Org<sup>7</sup>. Some MathJax display options can be configured via `org-html-mathjax-options`, or in the buffer. For example, with the following settings,

```
#+HTML_MATHJAX: align: left indent: 5em tagside: left font: Neo-Euler
```

equation labels will be displayed on the left margin and equations will be five ems from the left margin.

See the docstring of `org-html-mathjax-options` for all supported variables. The MathJax template can be configure via `org-html-mathjax-template`.

If you prefer, you can also request that  $\LaTeX$  fragments are processed into small images that will be inserted into the browser page. Before the availability of MathJax, this was the default method for Org files. This method requires that the `dvipng` program, `dvisvgm` or `imagemagick` suite is available on your system. You can still get this processing with

```
#+OPTIONS: tex:dvipng
#+OPTIONS: tex:dvisvgm
```

or:

```
#+OPTIONS: tex:imagemagick
```

### 12.9.10 Text areas in HTML export

Before Org mode's Babel, one popular approach to publishing code in HTML was by using `:textarea`. The advantage of this approach was that copying and pasting was built into browsers with simple JavaScript commands. Even editing before pasting was made simple.

The HTML export back-end can create such text areas. It requires an `#+ATTR_HTML:` line as shown in the example below with the `:textarea` option. This must be followed by either an `example` or a `src` code block. Other Org block types will not honor the `:textarea` option.

---

<sup>7</sup> By default Org loads MathJax from [cdnjs.com](https://cdnjs.com) (<https://cdnjs.com>) as recommended by MathJax (<http://www.mathjax.org>).

By default, the HTML export back-end creates a text area 80 characters wide and height just enough to fit the content. Override these defaults with `:width` and `:height` options on the `#+ATTR_HTML:` line.

```
#+ATTR_HTML: :textarea t :width 40
#+BEGIN_EXAMPLE
  (defun org-xor (a b)
    "Exclusive or."
    (if a (not b) b))
#+END_EXAMPLE
```

### 12.9.11 CSS support

You can modify the CSS style definitions for the exported file. The HTML exporter assigns the following special CSS classes<sup>8</sup> to appropriate parts of the document—your style specifications may change these, in addition to any of the standard classes like for headlines, tables, etc.

|                                 |  |
|---------------------------------|--|
| <code>p.author</code>           | author information, including email                    |
| <code>p.date</code>             | publishing date  |
| <code>p.creator</code>          | creator info, about org mode version                   |
| <code>.title</code>             | document title   |
| <code>.subtitle</code>          | document subtitle                                      |
| <code>.todo</code>              | TODO keywords, all not-done states                     |
| <code>.done</code>              | the DONE keywords, all states that count as done       |
| <code>.WAITING</code>           | each TODO keyword also uses a class named after itself |
| <code>.timestamp</code>         | timestamp  |
| <code>.timestamp-kwd</code>     | keyword associated with a timestamp, like SCHEDULED    |
| <code>.timestamp-wrapper</code> | span around keyword plus timestamp                     |
| <code>.tag</code>               | tag in a headline                                      |
| <code>._HOME</code>             | each tag uses itself as a class, "@" replaced by "_"   |
| <code>.target</code>            | target for links                                       |
| <code>.linenr</code>            | the line number in a code example                      |
| <code>.code-highlighted</code>  | for highlighting referenced code lines                 |
| <code>div.outline-N</code>      | div for outline level N (headline plus text))          |
| <code>div.outline-text-N</code> | extra div for text at outline level N                  |
| <code>.section-number-N</code>  | section number in headlines, different for each level  |
| <code>.figure-number</code>     | label like "Figure 1:"                                 |
| <code>.table-number</code>      | label like "Table 1:"                                  |
| <code>.listing-number</code>    | label like "Listing 1:"                                |
| <code>div.figure</code>         | how to format an in-lined image                        |
| <code>pre.src</code>            | formatted source code                                  |
| <code>pre.example</code>        | normal example   |
| <code>p.verse</code>            | verse paragraph  |
| <code>div.footnotes</code>      | footnote section headline                              |
| <code>p.footnote</code>         | footnote definition paragraph, containing a footnote   |
| <code>.footref</code>           | a footnote reference number (always a <sup>)</sup>     |

<sup>8</sup> If the classes on TODO keywords and tags lead to conflicts, use the variables `org-html-todo-kwd-class-prefix` and `org-html-tag-class-prefix` to make them unique.

`.footnum`                      footnote number in footnote definition (always `<sup>`)  
`.org-svg`                      default class for a linked `.svg` image

The HTML export back-end includes a compact default style in each exported HTML file. To override the default style with another style, use these keywords in the Org file. They will replace the global defaults the HTML exporter uses.

```
#+HTML_HEAD: <link rel="stylesheet" type="text/css" href="style1.css" />
#+HTML_HEAD_EXTRA: <link rel="alternate stylesheet" type="text/css" href="style2.css" />
```

To just turn off the default style, customize `org-html-head-include-default-style` variable, or use this option line in the Org file.

```
#+OPTIONS: html-style:nil
```

For longer style definitions, either use several `#+HTML_HEAD` and `#+HTML_HEAD_EXTRA` lines, or use `<style>` `</style>` blocks around them. Both of these approaches can avoid referring to an external file.

In order to add styles to a sub-tree, use the `:HTML_CONTAINER_CLASS:` property to assign a class to the tree. In order to specify CSS styles for a particular headline, you can use the id specified in a `:CUSTOM_ID:` property.

Never change the `org-html-style-default` constant. Instead use other simpler ways of customizing as described above.

### 12.9.12 ウェブページの表示に関する JavaScript のサポート

Sebastian Rose has written a JavaScript program especially designed to enhance the web viewing experience of HTML files created with Org. This program enhances large files in two different ways of viewing. One is an *Info*-like mode where each section is displayed separately and navigation can be done with the `n` and `p` keys (and some other keys as well, press `?` for an overview of the available keys). The second one has a *folding* view, much like Org provides inside Emacs. The script is available at <https://orgmode.org/org-info.js> and the documentation at <https://orgmode.org/worg/code/org-info-js/>. The script is hosted on <https://orgmode.org>, but for reliability, prefer installing it on your own web server.

To use this program, just add this line to the Org file:

```
#+INFOJS_OPT: view:info toc:nil
```

The HTML header now has the code needed to automatically invoke the script. For setting options, use the syntax from the above line for options described below:

**path:**        The path to the script. The default grabs the script from <https://orgmode.org/org-info.js>, but you might want to have a local copy and use a path like `../scripts/org-info.js`.

**view:**        Initial view when the website is first shown. Possible values are:

**info**        Info-like interface with one section per page.

**overview**    Folding interface, initially showing only top-level.

**content**    Folding interface, starting with all headlines visible.

**showall**    Folding interface, all headlines and text visible.

**sdepth:**    Maximum headline level that will still become an independent section for info and folding modes. The default is taken from `org-export-headline-levels` (= the `H` switch in `#+OPTIONS`).



- If this is smaller than in `org-export-headline-levels`, each info/folding section can still contain child headlines.
- toc:** Should the table of contents *initially* be visible? Even when `nil`, you can always get to the "toc" with *i*.
- tddepth:** The depth of the table of contents. The defaults are taken from the variables `org-export-headline-levels` and `org-export-with-toc`.
- ftoc:** Does the CSS of the page specify a fixed position for the "toc"? If yes, the toc will never be displayed as a section.
- ltoc:** Should there be short contents (children) in each section? Make this **above** if the section should be above initial text.
- mouse:** Headings are highlighted when the mouse is over them. Should be ‘underline’ (default) or a background color like ‘#cccccc’.
- buttons:** Should view-toggle buttons be everywhere? When `nil` (the default), only one such button will be present.

You can choose default values for these options by customizing the variable `org-html-infojs-options`. If you want the script to always apply to your pages, configure the variable `org-html-use-infojs`.

## 12.10 L<sup>A</sup>T<sub>E</sub>X のエクスポート

The L<sup>A</sup>T<sub>E</sub>X export back-end can handle complex documents, incorporate standard or custom L<sup>A</sup>T<sub>E</sub>X document classes, generate documents using alternate L<sup>A</sup>T<sub>E</sub>X engines, and produce fully linked PDF files with indexes, bibliographies, and tables of contents, destined for interactive online viewing or high-quality print publication.

While the details are covered in-depth in this section, here are some quick references to variables for the impatient: for engines, see `org-latex-compiler`; for build sequences, see `org-latex-pdf-process`; for packages, see `org-latex-default-packages-alist` and `org-latex-packages-alist`.

An important note about the L<sup>A</sup>T<sub>E</sub>X export back-end: it is sensitive to blank lines in the Org document. That’s because L<sup>A</sup>T<sub>E</sub>X itself depends on blank lines to tell apart syntactical elements, such as paragraphs.

### 12.10.1 L<sup>A</sup>T<sub>E</sub>X エクスポートのコマンド

- `C-c C-e l l` `org-latex-export-to-latex`  
Export as L<sup>A</sup>T<sub>E</sub>X file with a `.tex` extension. For `myfile.org`, Org exports to `myfile.tex`, overwriting without warning. `C-c C-e l l` Exports to L<sup>A</sup>T<sub>E</sub>X file.
- `C-c C-e l L` `org-latex-export-as-latex`  
Export to a temporary buffer. Do not create a file.
- `C-c C-e l p` `org-latex-export-to-pdf`  
Export as L<sup>A</sup>T<sub>E</sub>X file and convert it to PDF file.
- `C-c C-e l o`  
Export as L<sup>A</sup>T<sub>E</sub>X file and convert it to PDF, then open the PDF using the default viewer.

The L<sup>A</sup>T<sub>E</sub>X export back-end can use any of these L<sup>A</sup>T<sub>E</sub>X engines: ‘`pdflatex`’, ‘`xelatex`’, and ‘`lualatex`’. These engines compile L<sup>A</sup>T<sub>E</sub>X files with different compilers, packages,

and output options. The  $\text{\LaTeX}$  export back-end finds the compiler version to use from `org-latex-compiler` variable or the `#+LATEX_COMPILER` keyword in the Org file. See the docstring for the `org-latex-default-packages-alist` for loading packages with certain compilers. Also see `org-latex-bibtex-compiler` to set the bibliography compiler<sup>9</sup>.

### 12.10.2 $\text{\LaTeX}$ specific export settings

The  $\text{\LaTeX}$  export back-end has several additional keywords for customizing  $\text{\LaTeX}$  output. Setting these keywords works similar to the general options (see Section 12.2 [Export settings], page 139).

**‘DESCRIPTION’**

The document’s description. The description along with author name, keywords, and related file metadata are inserted in the output file by the ‘`hyperref`’ package. See `org-latex-hyperref-template` for customizing metadata items. See `org-latex-title-command` for typesetting description into the document’s front matter. Use multiple `#+DESCRIPTION` lines for long descriptions.

**‘LATEX\_CLASS’**

This is  $\text{\LaTeX}$  document class, such as `article`, `report`, `book`, and so on, which contain predefined preamble and headline level mapping that the  $\text{\LaTeX}$  export back-end needs. The back-end reads the default class name from the `org-latex-default-class` variable. Org has `article` as the default class. A valid default class must be an element of `org-latex-classes`.

**‘LATEX\_CLASS\_OPTIONS’**

Options the  $\text{\LaTeX}$  export back-end uses when calling the  $\text{\LaTeX}$  document class.

**‘LATEX\_COMPILER’**

The compiler, such as ‘`pdflatex`’, ‘`xelatex`’, ‘`lualatex`’, for producing the PDF (`org-latex-compiler`).

**‘LATEX\_HEADER’**

Arbitrary lines to add to the document’s preamble, before the ‘`hyperref`’ settings. See `org-latex-classes` for adjusting the structure and order of the  $\text{\LaTeX}$  headers.

**‘LATEX\_HEADER\_EXTRA’**

Arbitrary lines to add to the document’s preamble, before the ‘`hyperref`’ settings. See `org-latex-classes` for adjusting the structure and order of the  $\text{\LaTeX}$  headers.

**‘KEYWORDS’**

The keywords for the document. The description along with author name, keywords, and related file metadata are inserted in the output file by the ‘`hyperref`’ package. See `org-latex-hyperref-template` for customizing metadata items. See `org-latex-title-command` for typesetting description into the document’s front matter. Use multiple `#+KEYWORDS` lines if necessary.

---

<sup>9</sup> This does not allow setting different bibliography compilers for different files. However, “smart”  $\text{\LaTeX}$  compilation systems, such as ‘`latexmk`’, can select the correct bibliography compiler.

**‘SUBTITLE’**

The document’s subtitle. It is typeset as per `org-latex-subtitle-format`. If `org-latex-subtitle-separate` is non-nil, it is typed as part of the `\title`-macro. See `org-latex-hyperref-template` for customizing metadata items. See `org-latex-title-command` for typesetting description into the document’s front matter.

The following sections have further details.

### 12.10.3 L<sup>A</sup>T<sub>E</sub>X header and sectioning structure

The L<sup>A</sup>T<sub>E</sub>X export back-end converts the first three of Org’s outline levels into L<sup>A</sup>T<sub>E</sub>X headlines. The remaining Org levels are exported as `itemize` or `enumerate` lists. To change this globally for the cut-off point between levels and lists, (see Section 12.2 [Export settings], page 139).

By default, the L<sup>A</sup>T<sub>E</sub>X export back-end uses the `article` class.

To change the default class globally, edit `org-latex-default-class`. To change the default class locally in an Org file, add option lines `#+LATEX_CLASS: myclass`. To change the default class for just a part of the Org file, set a sub-tree property, `EXPORT_LATEX_CLASS`. The class name entered here must be valid member of `org-latex-classes`. This variable defines a header template for each class into which the exporter splices the values of `org-latex-default-packages-alist` and `org-latex-packages-alist`. Use the same three variables to define custom sectioning or custom classes.

The L<sup>A</sup>T<sub>E</sub>X export back-end sends the `LATEX_CLASS_OPTIONS` keyword and `EXPORT_LATEX_CLASS_OPTIONS` property as options to the L<sup>A</sup>T<sub>E</sub>X `\documentclass` macro. The options and the syntax for specifying them, including enclosing them in square brackets, follow L<sup>A</sup>T<sub>E</sub>X conventions.

```
#+LATEX_CLASS_OPTIONS: [a4paper,11pt,twoside,twocolumn]
```

The L<sup>A</sup>T<sub>E</sub>X export back-end appends values from `LATEX_HEADER` and `LATEX_HEADER_EXTRA` keywords to the L<sup>A</sup>T<sub>E</sub>X header. The docstring for `org-latex-classes` explains in more detail. Also note that L<sup>A</sup>T<sub>E</sub>X export back-end does not append `LATEX_HEADER_EXTRA` to the header when previewing L<sup>A</sup>T<sub>E</sub>X snippets (see Section 11.8.2 [Previewing L<sup>A</sup>T<sub>E</sub>X fragments], page 136).

A sample Org file with the above headers:

```
#+LATEX_CLASS: article
#+LATEX_CLASS_OPTIONS: [a4paper]
#+LATEX_HEADER: \usepackage{xyz}

* Headline 1
  some text
* Headline 2
  some more text
```

### 12.10.4 Quoting L<sup>A</sup>T<sub>E</sub>X code

The L<sup>A</sup>T<sub>E</sub>X export back-end can insert any arbitrary L<sup>A</sup>T<sub>E</sub>X code, see Section 11.8 [Embedded L<sup>A</sup>T<sub>E</sub>X], page 135. There are three ways to embed such code in the Org file and they all use different quoting syntax.

Inserting in-line quoted with symbols:

Code embedded in-line `@@latex: any arbitrary LaTeX code@@` in a paragraph.

Inserting as one or more keyword lines in the Org file:

```
#+LATEX: any arbitrary LaTeX code
```

Inserting as an export block in the Org file, where the back-end exports any code between begin and end markers:

```
#+BEGIN_EXPORT latex
any arbitrary LaTeX code
#+END_EXPORT
```

### 12.10.5 $\text{\LaTeX}$ エクスポートにおけるテーブル

The  $\text{\LaTeX}$  export back-end can pass several  $\text{\LaTeX}$  attributes for table contents and layout. Besides specifying label and caption (see Section 11.4 [Images and tables], page 132), the other valid  $\text{\LaTeX}$  attributes include:

**:mode** The  $\text{\LaTeX}$  export back-end wraps the table differently depending on the mode for accurate rendering of math symbols. Mode is either `table`, `math`, `inline-math` or `verbatim`. For `math` or `inline-math` mode,  $\text{\LaTeX}$  export back-end wraps the table in a math environment, but every cell in it is exported as-is. The  $\text{\LaTeX}$  export back-end determines the default mode from `org-latex-default-table-mode`. For `table`, The  $\text{\LaTeX}$  export back-end merges contiguous tables in the same mode into a single environment.

**:environment**

Set the default  $\text{\LaTeX}$  table environment for the  $\text{\LaTeX}$  export back-end to use when exporting Org tables. Common  $\text{\LaTeX}$  table environments are provided by these packages: `tabularx`, `longtable`, `array`, `tabu`, and `bmatrix`. For packages, such as `tabularx` and `tabu`, or any newer replacements, include them in the `org-latex-packages-alist` variable so the  $\text{\LaTeX}$  export back-end can insert the appropriate load package headers in the converted  $\text{\LaTeX}$  file. Look in the docstring for the `org-latex-packages-alist` variable for configuring these packages for  $\text{\LaTeX}$  snippet previews, if any.

**:caption** Use `#+CAPTION` keyword to set a simple caption for a table (see Section 11.4 [Images and tables], page 132). For custom captions, use `:caption` attribute, which accepts raw  $\text{\LaTeX}$  code. `:caption` value overrides `#+CAPTION` value.

**:float**

**:placement**

The table environments by default are not floats in  $\text{\LaTeX}$ . To make them floating objects use `:float` with one of the following options: `sideways`, `multicolumn`, `t`, and `nil`. Note that `sidewaystable` has been deprecated since Org 8.3.  $\text{\LaTeX}$  floats can also have additional layout `:placement` attributes. These are the usual `[h t b p ! H]` permissions specified in square brackets. Note that for `:float sideways` tables, the  $\text{\LaTeX}$  export back-end ignores `:placement` attributes.

`:align`  
`:font`  
`:width` The  $\text{\LaTeX}$  export back-end uses these attributes for regular tables to set their alignments, fonts, and widths.  
`:spread` When `:spread` is non-nil, the  $\text{\LaTeX}$  export back-end spreads or shrinks the table by the `:width` for `tabu` and `longtabu` environments. `:spread` has no effect if `:width` is not set.  
`:booktabs`  
`:center`  
`:rmlines` All three commands are toggles. `:booktabs` brings in modern typesetting enhancements to regular tables. The `booktabs` package has to be loaded through `org-latex-packages-alist`. `:center` is for centering the table. `:rmlines` removes all but the very first horizontal line made of ASCII characters from "table.el" tables only.  
`:math-prefix`  
`:math-suffix`  
`:math-arguments`  
 The  $\text{\LaTeX}$  export back-end inserts `:math-prefix` string value in a math environment before the table. The  $\text{\LaTeX}$  export back-end inserts `:math-suffix` string value in a math environment after the table. The  $\text{\LaTeX}$  export back-end inserts `:math-arguments` string value between the macro name and the table's contents. `:math-arguments` comes in use for matrix macros that require more than one argument, such as `qbordermatrix`.

$\text{\LaTeX}$  table attributes help formatting tables for a wide range of situations, such as matrix product or spanning multiple pages:

```

#+ATTR_LATEX: :environment longtable :align l|lp{3cm}r|l
| ..... | ..... |
| ..... | ..... |

#+ATTR_LATEX: :mode math :environment bmatrix :math-suffix \times
| a | b |
| c | d |
#+ATTR_LATEX: :mode math :environment bmatrix
| 1 | 2 |
| 3 | 4 |

```

Set the caption with the  $\text{\LaTeX}$  command `\bicaption{HeadingA}{HeadingB}`:

```

#+ATTR_LATEX: :caption \bicaption{HeadingA}{HeadingB}
| ..... | ..... |
| ..... | ..... |

```

### 12.10.6 $\text{\LaTeX}$ エクスポートにおける画像

The  $\text{\LaTeX}$  export back-end processes image links in Org files that do not have descriptions, such as these links `'[[file:img.jpg]]'` or `'[./img.jpg]'`, as direct image insertions in the final PDF output. In the PDF, they are no longer links but actual images embedded on

the page. The  $\text{\LaTeX}$  export back-end uses `\includegraphics` macro to insert the image. But for TikZ<sup>10</sup> images, the back-end uses an `\input` macro wrapped within a `tikzpicture` environment.

For specifying image `:width`, `:height`, and other `:options`, use this syntax:

```
#+ATTR_LATEX: :width 5cm :options angle=90
[[./img/sed-hr4049.pdf]]
```

For custom commands for captions, use the `:caption` attribute. It will override the default `#+CAPTION` value:

```
#+ATTR_LATEX: :caption \bicaption{HeadingA}{HeadingB}
[[./img/sed-hr4049.pdf]]
```

When captions follow the method as described in Section 11.4 [Images and tables], page 132, the  $\text{\LaTeX}$  export back-end wraps the picture in a floating `figure` environment. To float an image without specifying a caption, set the `:float` attribute to one of the following:

- `t`: for a standard ‘`figure`’ environment; used by default whenever an image has a caption.
- `multicolumn`: to span the image across multiple columns of a page; the back-end wraps the image in a `figure*` environment.
- `wrap`: for text to flow around the image on the right; the figure occupies the left half of the page.
- `sideways`: for a new page with the image sideways, rotated ninety degrees, in a `sidewaysfigure` environment; overrides `:placement` setting.
- `nil`: to avoid a `:float` even if using a caption.

Use the `placement` attribute to modify a floating environment’s placement.

```
#+ATTR_LATEX: :float wrap :width 0.38\textwidth :placement
{r}{0.4\textwidth} [[./img/hst.png]]
```

The  $\text{\LaTeX}$  export back-end centers all images by default. Setting `:center` attribute to `nil` disables centering. To disable centering globally, set `org-latex-images-centered` to `t`.

Set the `:comment-include` attribute to non-`nil` value for the  $\text{\LaTeX}$  export back-end to comment out the `\includegraphics` macro.

### 12.10.7 Plain lists in $\text{\LaTeX}$ export

The  $\text{\LaTeX}$  export back-end accepts the `:environment` and `:options` attributes for plain lists. Both attributes work together for customizing lists, as shown in the examples:

```
#+LATEX_HEADER: \usepackage[inline]{enumitem}
Some ways to say "Hello":
#+ATTR_LATEX: :environment itemize*
#+ATTR_LATEX: :options [label={}, itemjoin={,}, itemjoin*={, and}]
- Hola
- Bonjour
```

---

<sup>10</sup> <http://sourceforge.net/projects/pgf/>

- Guten Tag.

Since  $\text{\LaTeX}$  supports only four levels of nesting for lists, use an external package, such as ‘`enumitem`’ in  $\text{\LaTeX}$ , for levels deeper than four:

```
#+LATEX_HEADER: \usepackage{enumitem}
#+LATEX_HEADER: \renewlist{itemize}{itemize}{9}
#+LATEX_HEADER: \setlist[itemize]{label=$\circ$}
- One
  - Two
    - Three
      - Four
        - Five
```

### 12.10.8 Source blocks in $\text{\LaTeX}$ export

The  $\text{\LaTeX}$  export back-end can make source code blocks into floating objects through the attributes `:float` and `:options`. For `:float`:

- `t`: makes a source block float; by default floats any source block with a caption.
- `multicolumn`: spans the source block across multiple columns of a page.
- `nil`: avoids a `:float` even if using a caption; useful for source code blocks that may not fit on a page.

```
#+ATTR_LATEX: :float nil
#+BEGIN_SRC emacs-lisp
Lisp code that may not fit in a single page.
#+END_SRC
```

The  $\text{\LaTeX}$  export back-end passes string values in `:options` to  $\text{\LaTeX}$  packages for customization of that specific source block. In the example below, the `:options` are set for `Minted`. `Minted` is a source code highlighting  $\text{\LaTeX}$  package with many configurable options.

```
#+ATTR_LATEX: :options commentstyle=\bfseries
#+BEGIN_SRC emacs-lisp
(defun Fib (n)
  (if (< n 2) n (+ (Fib (- n 1)) (Fib (- n 2)))))
#+END_SRC
```

To apply similar configuration options for all source blocks in a file, use the `org-latex-listings-options` and `org-latex-minted-options` variables.

### 12.10.9 Example blocks in $\text{\LaTeX}$ export

The  $\text{\LaTeX}$  export back-end wraps the contents of example blocks in a ‘`verbatim`’ environment. To change this behavior to use another environment globally, specify an appropriate export filter (see Section 12.17 [Advanced configuration], page 184). To change this behavior to use another environment for each block, use the `:environment` parameter to specify a custom environment.

```
#+ATTR_LATEX: :environment myverbatim
#+BEGIN_EXAMPLE
This sentence is false.
#+END_EXAMPLE
```

### 12.10.10 Special blocks in $\text{\LaTeX}$ export

For other special blocks in the Org file, the  $\text{\LaTeX}$  export back-end makes a special environment of the same name. The back-end also takes `:options`, if any, and appends as-is to that environment's opening string. For example:

```
#+BEGIN_abstract
We demonstrate how to solve the Syracuse problem.
#+END_abstract

#+ATTR_LATEX: :options [Proof of important theorem]
#+BEGIN_proof
...
Therefore, any even number greater than 2 is the sum of two primes.
#+END_proof
```

exports to

```
\begin{abstract}
We demonstrate how to solve the Syracuse problem.
\end{abstract}

\begin{proof}[Proof of important theorem]
...
Therefore, any even number greater than 2 is the sum of two primes.
\end{proof}
```

If you need to insert a specific caption command, use `:caption` attribute. It will override standard `#+CAPTION` value, if any. For example:

```
#+ATTR_LATEX: :caption \MyCaption{HeadingA}
#+BEGIN_proof
...
#+END_proof
```

### 12.10.11 Horizontal rules in $\text{\LaTeX}$ export

The  $\text{\LaTeX}$  export back-end converts horizontal rules by the specified `:width` and `:thickness` attributes. For example:

```
#+ATTR_LATEX: :width .6\textwidth :thickness 0.8pt
-----
```

## 12.11 Markdown export

The Markdown export back-end, `md`, converts an Org file to Markdown format, as defined at <http://daringfireball.net/projects/markdown/>.

The `md` back-end is built on top of the HTML back-end (see Section 12.9 [HTML export], page 150). As a consequence, it converts every Org construct not supported by Markdown syntax, such as tables, to HTML.



## Markdown export commands

- C-c C-e m m** **org-md-export-to-markdown**  
 Export to a text file with Markdown syntax. For `myfile.org`, Org exports to `myfile.md`, overwritten without warning.
- C-c C-e m M** **org-md-export-as-markdown**  
 Export to a temporary buffer. Does not create a file.
- C-c C-e m o**  
 Export as a text file with Markdown syntax, then open it.

## Header and sectioning structure

Based on `org-md-headline-style`, markdown export can generate headlines of both `atx` and `setext` types. `atx` limits headline levels to two. `setext` limits headline levels to six. Beyond these limits, the export back-end converts headlines to lists. To set a limit to a level before the absolute limit (see Section 12.2 [Export settings], page 139).

## 12.12 OpenDocument Text export

The ODT export back-end handles creating of OpenDocument Text (ODT) format files. The format complies with *OpenDocument-v1.2 specification*<sup>11</sup> and is compatible with LibreOffice 3.4.

### 12.12.1 Pre-requisites for ODT export

The ODT export back-end relies on the `zip` program to create the final compressed ODT output. Check if `zip` is locally available and executable. Without `zip`, export cannot finish.

### 12.12.2 ODT export commands

- C-c C-e o o** **org-odt-export-to-odt**  
 Export as OpenDocument Text file.
- If `org-odt-preferred-output-format` is specified, the ODT export back-end automatically converts the exported file to that format. See [Automatically exporting to other formats], page 167.
- For `myfile.org`, Org exports to `myfile.odt`, overwriting without warning. The ODT export back-end exports a region only if a region was active. Note for exporting active regions, the `transient-mark-mode` has to be turned on.
- If the selected region is a single tree, the ODT export back-end makes the tree head the document title. Incidentally, **C-c @** selects the current sub-tree. If the tree head entry has, or inherits, an `EXPORT_FILE_NAME` property, the ODT export back-end uses that for file name.
- C-c C-e o O** Export to an OpenDocument Text file format and open it.
- When `org-odt-preferred-output-format` is specified, open the converted file instead. See [Automatically exporting to other formats], page 167.

<sup>11</sup> Open Document Format for Office Applications (OpenDocument) Version 1.2 (<http://docs.oasis-open.org/office/v1.2/OpenDocument-v1.2.html>)

### 12.12.3 ODT specific export settings

The ODT export back-end has several additional keywords for customizing ODT output. Setting these keywords works similar to the general options (see Section 12.2 [Export settings], page 139).

**‘DESCRIPTION’**

This is the document’s description, which the ODT export back-end inserts as document metadata. For long descriptions, use multiple **#+DESCRIPTION** lines.

**‘KEYWORDS’**

The keywords for the document. The ODT export back-end inserts the description along with author name, keywords, and related file metadata as metadata in the output file. Use multiple **#+KEYWORDS** lines if necessary.

**‘ODT\_STYLES\_FILE’**

The ODT export back-end uses the **org-odt-styles-file** by default. See Section 12.12.5 [Applying custom styles], page 168 for details.

**‘SUBTITLE’**

The document subtitle.

### 12.12.4 Extending ODT export

The ODT export back-end can produce documents in other formats besides ODT using a specialized ODT converter process. Its common interface works with popular converters to produce formats such as **‘doc’**, or convert a document from one format, say **‘csv’**, to another format, say **‘xls’**.

Customize **org-odt-convert-process** variable to point to **unoconv**, which is the ODT’s preferred converter. Working installations of LibreOffice would already have **unoconv** installed. Alternatively, other converters may be substituted here. See [Configuring a document converter], page 172.

### Automatically exporting to other formats

If ODT format is just an intermediate step to get to other formats, such as **‘doc’**, **‘docx’**, **‘rtf’**, or **‘pdf’**, etc., then extend the ODT export back-end to directly produce that format. Specify the final format in the **org-odt-preferred-output-format** variable. This is one way to extend (see [Exporting to ODT], page 166).

### Converting between document formats

The Org export back-end is made to be inter-operable with a wide range of text document format converters. Newer generation converters, such as LibreOffice and Pandoc, can handle hundreds of formats at once. Org provides a consistent interaction with whatever converter is installed. Here are some generic commands:

***M-x org-odt-convert RET***

Convert an existing document from one format to another. With a prefix argument, opens the newly produced file.

### 12.12.5 Applying custom styles

The ODT export back-end comes with many OpenDocument styles (see [Working with OpenDocument style files], page 173). To expand or further customize these built-in style sheets, either edit the style sheets directly or generate them using an application such as LibreOffice. The example here shows creating a style using LibreOffice.

#### Applying custom styles: the easy way

1. Create a sample `example.org` file with settings as shown below, and export it to ODT format.

```
#+OPTIONS: H:10 num:t
```

2. Open the above `example.odt` using LibreOffice. Use the **Stylist** to locate the target styles, which typically have the ‘`Org`’ prefix. Open one, modify, and save as either OpenDocument Text (`.odt`) or OpenDocument Template (`.ott`) file.
3. Customize the variable `org-odt-styles-file` and point it to the newly created file. For additional configuration options see [Overriding factory styles], page 173.

To apply an ODT style to a particular file, use the `#+ODT_STYLES_FILE` option as shown in the example below:

```
#+ODT_STYLES_FILE: "/path/to/example.ott"
```

もしくは

```
#+ODT_STYLES_FILE: ("/path/to/file.ott" ("styles.xml" "image/hdr.png"))
```

#### Using third-party styles and templates

The ODT export back-end relies on many templates and style names. Using third-party styles and templates can lead to mismatches. Templates derived from built-in ODT templates and styles seem to have fewer problems.

### 12.12.6 Links in ODT export

ODT export back-end creates native cross-references for internal links and Internet-style links for all other link types.

A link with no description and pointing to a regular—un-itemized—outline heading is replaced with a cross-reference and section number of the heading.

A ‘`\ref{label}`’-style reference to an image, table etc. is replaced with a cross-reference and sequence number of the labeled entity. See Section 12.12.10 [Labels and captions in ODT export], page 171.

### 12.12.7 Tables in ODT export

The ODT export back-end handles native Org mode tables (see Chapter 3 [Tables], page 19) and simple `table.el` tables. Complex `table.el` tables having column or row spans are not supported. Such tables are stripped from the exported document.

By default, the ODT export back-end exports a table with top and bottom frames and with ruled lines separating row and column groups (see Section 3.3 [Column groups], page 23). All tables are typeset to occupy the same width. The ODT export back-end honors any table alignments and relative widths for columns (see Section 3.2 [Column width and alignment], page 22).

Note that the ODT export back-end interprets column widths as weighted ratios, the default weight being 1.

Specifying `:rel-width` property on an `#+ATTR_ODT` line controls the width of the table. For example:

```
#+ATTR_ODT: :rel-width 50
| Area/Month | Jan | Feb | Mar | Sum |
|-----+-----+-----+-----+-----|
| /          | <  |   |   |   | <  |
| <l13>      | <r5> | <r5> | <r5> | <r6> |
| North America | 1 | 21 | 926 | 948 |
| Middle East   | 6 | 75 | 844 | 925 |
| Asia Pacific  | 9 | 27 | 790 | 826 |
|-----+-----+-----+-----+-----|
| Sum          | 16 | 123 | 2560 | 2699 |
```

On export, the above table takes 50% of text width area. The exporter sizes the columns in the ratio: 13:5:5:5:6. The first column is left-aligned and rest of the columns, right-aligned. Vertical rules separate the header and the last column. Horizontal rules separate the header and the last row.

For even more customization, create custom table styles and associate them with a table using the `#+ATTR_ODT` line. See [Customizing tables in ODT export], page 174.

## 12.12.8 Images in ODT export

### Embedding images

The ODT export back-end processes image links in Org files that do not have descriptions, such as these links `[[file:img.jpg]]` or `[[./img.jpg]]`, as direct image insertions in the final output. Either of these examples works:

```
[[file:img.png]]
[[./img.png]]
```

### Embedding clickable images

For clickable images, provide a link whose description is another link to an image file. For example, to embed an image `org-mode-unicorn.png` which when clicked jumps to `https://orgmode.org` website, do the following

```
[[https://orgmode.org][./org-mode-unicorn.png]]
```

### Sizing and scaling of embedded images

Control the size and scale of the embedded images with the `#+ATTR_ODT` attribute.

The ODT export back-end starts with establishing the size of the image in the final document. The dimensions of this size is measured in centimeters. The back-end then queries the image file for its dimensions measured in pixels. For this measurement, the back-end relies on ImageMagick's `identify` program or Emacs `create-image` and `image-size` API. ImageMagick is the preferred choice for large file sizes or frequent batch operations. The back-end then converts the pixel dimensions using `org-odt-pixels-per-inch` into the familiar 72 dpi or 96 dpi. The default value for this is in `display-pixels-per-inch`, which

can be tweaked for better results based on the capabilities of the output device. Here are some common image scaling operations:

Explicitly size the image

To embed `img.png` as a 10 cm x 10 cm image, do the following:

```
#+ATTR_ODT: :width 10 :height 10
[[./img.png]]
```

Scale the image

To embed `img.png` at half its size, do the following:

```
#+ATTR_ODT: :scale 0.5
[[./img.png]]
```

Scale the image to a specific width

To embed `img.png` with a width of 10 cm while retaining the original height:width ratio, do the following:

```
#+ATTR_ODT: :width 10
[[./img.png]]
```

Scale the image to a specific height

To embed `img.png` with a height of 10 cm while retaining the original height:width ratio, do the following

```
#+ATTR_ODT: :height 10
[[./img.png]]
```

## Anchoring of images

The ODT export back-end can anchor images to “`as-char`”, “`paragraph`”, or “`page`”. Set the preferred anchor using the `:anchor` property of the `#+ATTR_ODT` line.

To create an image that is anchored to a page:

```
#+ATTR_ODT: :anchor "page"
[[./img.png]]
```

### 12.12.9 Math formatting in ODT export

The ODT export back-end has special support built-in for handling math.

#### Working with $\text{\LaTeX}$ math snippets

$\text{\LaTeX}$  math snippets (see [\[ \$\text{\LaTeX}\$  fragments\]](#), page [\(undefined\)](#)) can be embedded in an ODT document in one of the following ways:

1. MathML

Add this line to the Org file. This option is activated on a per-file basis.

```
#+OPTIONS: LaTeX:t
```

With this option,  $\text{\LaTeX}$  fragments are first converted into MathML fragments using an external  $\text{\LaTeX}$ -to-MathML converter program. The resulting MathML fragments are then embedded as an OpenDocument Formula in the exported document.

To specify the  $\text{\LaTeX}$ -to-MathML converter, customize the variables `org-latex-to-mathml-convert-command` and `org-latex-to-mathml-jar-file`.

To use MathToWeb<sup>12</sup> as the preferred converter, configure the above variables as

```
(setq org-latex-to-mathml-convert-command
      "java -jar %j -unicode -force -df %o %I"
      org-latex-to-mathml-jar-file
      "/path/to/mathtoweb.jar")
```

To use L<sup>A</sup>T<sub>E</sub>X<sup>13</sup> use

```
(setq org-latex-to-mathml-convert-command
      "latexmlmath \"%i\" --presentationmathml=%o")
```

To quickly verify the reliability of the L<sup>A</sup>T<sub>E</sub>X-to-MathML converter, use the following commands:

*M-x org-odt-export-as-odf RET*

Convert a L<sup>A</sup>T<sub>E</sub>X math snippet to an OpenDocument formula (.odf) file.

*M-x org-odt-export-as-odf-and-open RET*

Convert a L<sup>A</sup>T<sub>E</sub>X math snippet to an OpenDocument formula (.odf) file and open the formula file with the system-registered application.

## 2. PNG images

Add this line to the Org file. This option is activated on a per-file basis.

```
#+OPTIONS: tex:dvipng
#+OPTIONS: tex:dvisvgm
```

or:

```
#+OPTIONS: tex:imagemagick
```

Under this option, L<sup>A</sup>T<sub>E</sub>X fragments are processed into PNG or SVG images and the resulting images are embedded in the exported document. This method requires dvipng program, dvisvgm or imagemagick programs.

## Working with MathML or OpenDocument formula files

When embedding L<sup>A</sup>T<sub>E</sub>X math snippets in ODT documents is not reliable, there is one more option to try. Embed an equation by linking to its MathML (.mml) source or its OpenDocument formula (.odf) file as shown below:

```
[[./equation.mml]]
もしくは
[[./equation.odf]]
```

### 12.12.10 Labels and captions in ODT export

ODT format handles labeling and captioning of objects based on their types. Inline images, tables, L<sup>A</sup>T<sub>E</sub>X fragments, and Math formulas are numbered and captioned separately. Each object also gets a unique sequence number based on its order of first appearance in the Org file. Each category has its own sequence. A caption is just a label applied to these objects.

```
#+CAPTION: Bell curve
#+LABEL:    fig:SED-HR4049
```

<sup>12</sup> See MathToWeb ([http://www.mathtoweb.com/cgi-bin/mathtoweb\\_home.pl](http://www.mathtoweb.com/cgi-bin/mathtoweb_home.pl)).

<sup>13</sup> See <http://dlmf.nist.gov/LaTeXML/>.

```
[[./img/a.png]]
```

When rendered, it may show as follows in the exported document:

Figure 2: Bell curve

To modify the category component of the caption, customize the option `org-odt-category-map-alist`. For example, to tag embedded images with the string ‘Illustration’ instead of the default string ‘Figure’, use the following setting:

```
(setq org-odt-category-map-alist
      '(("__Figure__" "Illustration" "value" "Figure" org-odt--enumerable-image-p))
```

With the above modification, the previous example changes to:

Illustration 2: Bell curve

### 12.12.11 Literal examples in ODT export

The ODT export back-end supports literal examples (see Section 11.5 [Literal examples], page 132) with full fontification. Internally, the ODT export back-end relies on `htmlfontify.el` to generate the style definitions needed for fancy listings. The auto-generated styles get ‘OrgSrc’ prefix and inherit colors from the faces used by Emacs `font-lock` library for that source language.

For custom fontification styles, customize the `org-odt-create-custom-styles-for-srcblocks` option.

To turn off fontification of literal examples, customize the `org-odt-fontify-srcblocks` option.

### 12.12.12 Advanced topics in ODT export

The ODT export back-end has extensive features useful for power users and frequent uses of ODT formats.

## Configuring a document converter

The ODT export back-end works with popular converters with little or no extra configuration. See Section 12.12.4 [Extending ODT export], page 167. The following is for unsupported converters or tweaking existing defaults.

#### 1. Register the converter

Add the name of the converter to the `org-odt-convert-processes` variable. Note that it also requires how the converter is invoked on the command line. See the variable’s docstring for details.

#### 2. Configure its capabilities

Specify which formats the converter can handle by customizing the variable `org-odt-convert-capabilities`. Use the entry for the default values in this variable for configuring the new converter. Also see its docstring for details.

#### 3. Choose the converter

Select the newly added converter as the preferred one by customizing the option `org-odt-convert-process`.

## Working with OpenDocument style files

This section explores the internals of the ODT exporter; the means by which it produces styled documents; the use of automatic and custom OpenDocument styles.

### a) Factory styles

The ODT exporter relies on two files for generating its output. These files are bundled with the distribution under the directory pointed to by the variable `org-odt-styles-dir`. The two files are:

- `OrgOdtStyles.xml`

This file contributes to the `styles.xml` file of the final ‘ODT’ document. This file gets modified for the following purposes:

1. To control outline numbering based on user settings.
2. To add styles generated by `htmlfontify.el` for fontification of code blocks.

- `OrgOdtContentTemplate.xml`

This file contributes to the `content.xml` file of the final ‘ODT’ document. The contents of the Org outline are inserted between the `<office:text>`...`</office:text>` elements of this file.

Apart from serving as a template file for the final `content.xml`, the file serves the following purposes:

1. It contains automatic styles for formatting of tables which are referenced by the exporter.
2. It contains `<text:sequence-decl>`...`</text:sequence-decl>` elements that control numbering of tables, images, equations, and similar entities.

### b) Overriding factory styles

The following two variables control the location from where the ODT exporter picks up the custom styles and content template files. Customize these variables to override the factory styles used by the exporter.

- `org-odt-styles-file`

The ODT export back-end uses the file pointed to by this variable, such as `styles.xml`, for the final output. It can take one of the following values:

1. A `styles.xml` file  
Use this file instead of the default `styles.xml`
2. A `.odt` or `.ott` file  
Use the `styles.xml` contained in the specified OpenDocument Text or Template file
3. A `.odt` or `.ott` file and a subset of files contained within them  
Use the `styles.xml` contained in the specified OpenDocument Text or Template file. Additionally extract the specified member files and embed those within the final ‘ODT’ document.  
Use this option if the `styles.xml` file references additional files like header and footer images.



4. `nil`

Use the default `styles.xml`

- `org-odt-content-template-file`

Use this variable to specify the blank `content.xml` that will be used in the final output.

## Creating one-off styles

The ODT export back-end can read embedded raw OpenDocument XML from the Org file. Such direct formatting are useful for one-off instances.

### 1. Embedding ODT tags as part of regular text

Enclose OpenDocument syntax in ‘`@@odt:...@@`’ for inline markup. For example, to highlight a region of text do the following:

```
@@odt:<text:span text:style-name="Highlight">This is highlighted
text</text:span>@@. But this is regular text.
```

**Hint:** To see the above example in action, edit the `styles.xml` (see [Factory styles], page 173) and add a custom ‘Highlight’ style as shown below:

```
<style:style style:name="Highlight" style:family="text">
  <style:text-properties fo:background-color="#ff0000"/>
</style:style>
```

### 2. Embedding a one-line OpenDocument XML

The ODT export back-end can read one-liner options with `#+ODT:` in the Org file. For example, to force a page break:

```
#+ODT: <text:p text:style-name="PageBreak"/>
```

**Hint:** To see the above example in action, edit your `styles.xml` (see [Factory styles], page 173) and add a custom ‘PageBreak’ style as shown below.

```
<style:style style:name="PageBreak" style:family="paragraph"
  style:parent-style-name="Text_20_body">
  <style:paragraph-properties fo:break-before="page"/>
</style:style>
```

### 3. Embedding a block of OpenDocument XML

The ODT export back-end can also read ODT export blocks for OpenDocument XML. Such blocks use the `#+BEGIN_EXPORT odt...#+END_EXPORT` constructs.

For example, to create a one-off paragraph that uses bold text, do the following:

```
#+BEGIN_EXPORT odt
<text:p text:style-name="Text_20_body_20_bold">
This paragraph is specially formatted and uses bold text.
</text:p>
#+END_EXPORT
```

## Customizing tables in ODT export

Override the default table format by specifying a custom table style with the `#+ATTR_ODT` line. For a discussion on default formatting of tables see Section 12.12.7 [Tables in ODT export], page 168.

This feature closely mimics the way table templates are defined in the OpenDocument-v1.2 specification.<sup>14</sup>

For quick preview of this feature, install the settings below and export the table that follows:

```
(setq org-odt-table-styles
  (append org-odt-table-styles
    '(("TableWithHeaderRowAndColumn" "Custom"
      ((use-first-row-styles . t)
       (use-first-column-styles . t)))
      ("TableWithFirstRowandLastRow" "Custom"
      ((use-first-row-styles . t)
       (use-last-row-styles . t))))))

#+ATTR_ODT: :style TableWithHeaderRowAndColumn
| Name | Phone | Age |
| Peter | 1234 | 17 |
| Anna | 4321 | 25 |
```

The example above used ‘Custom’ template and installed two table styles ‘TableWithHeaderRowAndColumn’ and ‘TableWithFirstRowandLastRow’. **Important:** The OpenDocument styles needed for producing the above template were pre-defined. They are available in the section marked ‘Custom Table Template’ in `OrgOdtContentTemplate.xml` (see [Factory styles], page 173. For adding new templates, define new styles here.

To use this feature proceed as follows:

1. Create a table template<sup>15</sup>

A table template is set of ‘table-cell’ and ‘paragraph’ styles for each of the following table cell categories:

- Body
- First column
- Last column
- First row
- Last row
- Even row
- Odd row
- Even column
- Odd Column

The names for the above styles must be chosen based on the name of the table template using a well-defined convention.

The naming convention is better illustrated with an example. For a table template with the name ‘Custom’, the needed style names are listed in the following table.

<sup>14</sup> OpenDocument-v1.2 Specification (<http://docs.oasis-open.org/office/v1.2/OpenDocument-v1.2.html>)

<sup>15</sup> See the `<table:table-template>` element of the OpenDocument-v1.2 specification

| Table cell type | table-cell style             | paragraph style                   |
|-----------------|------------------------------|-----------------------------------|
| Body            | 'CustomTableCell'            | 'CustomTableParagraph'            |
| First column    | 'CustomFirstColumnTableCell' | 'CustomFirstColumnTableParagraph' |
| Last column     | 'CustomLastColumnTableCell'  | 'CustomLastColumnTableParagraph'  |
| First row       | 'CustomFirstRowTableCell'    | 'CustomFirstRowTableParagraph'    |
| Last row        | 'CustomLastRowTableCell'     | 'CustomLastRowTableParagraph'     |
| Even row        | 'CustomEvenRowTableCell'     | 'CustomEvenRowTableParagraph'     |
| Odd row         | 'CustomOddRowTableCell'      | 'CustomOddRowTableParagraph'      |
| Even column     | 'CustomEvenColumnTableCell'  | 'CustomEvenColumnTableParagraph'  |
| Odd column      | 'CustomOddColumnTableCell'   | 'CustomOddColumnTableParagraph'   |

To create a table template with the name 'Custom', define the above styles in the `<office:automatic-styles>...</office:automatic-styles>` element of the content template file (see [Factory styles], page 173).

## 2. Define a table style<sup>16</sup>

To define a table style, create an entry for the style in the variable `org-odt-table-styles` and specify the following:

- the name of the table template created in step (1)
- the set of cell styles in that template that are to be activated

For example, the entry below defines two different table styles 'TableWithHeaderRowAndColumn' and 'TableWithFirstRowandLastRow' based on the same template 'Custom'. The styles achieve their intended effect by selectively activating the individual cell styles in that template.

```
(setq org-odt-table-styles
  (append org-odt-table-styles
    '(("TableWithHeaderRowAndColumn" "Custom"
      ((use-first-row-styles . t)
       (use-first-column-styles . t))))
    ("TableWithFirstRowandLastRow" "Custom"
      ((use-first-row-styles . t)
       (use-last-row-styles . t))))))
```

## 3. Associate a table with the table style

To do this, specify the table style created in step (2) as part of the `ATTR_ODT` line as shown below.

```
#+ATTR_ODT: :style "TableWithHeaderRowAndColumn"
| Name | Phone | Age |
| Peter | 1234 | 17 |
| Anna | 4321 | 25 |
```

<sup>16</sup> See the attributes `table:template-name`, `table:use-first-row-styles`, `table:use-last-row-styles`, `table:use-first-column-styles`, `table:use-last-column-styles`, `table:use-banding-rows-styles`, and `table:use-banding-column-styles` of the `<table:table>` element in the OpenDocument-v1.2 specification

## Validating OpenDocument XML

Sometimes ODT format files may not open due to `.odt` file corruption. To verify if the `.odt` file is corrupt, validate it against the OpenDocument RELAX NG Compact Syntax—RNC—schema. But first the `.odt` files have to be decompressed using ‘`zip`’. Note that `.odt` files are ‘`zip`’ archives: See Info file `emacs`, node ‘`File Archives`’. The contents of `.odt` files are in `.xml`. For general help with validation—and schema-sensitive editing—of XML files: See Info file `nxml-mode`, node ‘`Introduction`’.

Customize `org-odt-schema-dir` to point to a directory with OpenDocument `.rnc` files and the needed schema-locating rules. The ODT export back-end takes care of updating the `rng-schema-locating-files`.

### 12.13 Org export

`org` export back-end creates a normalized version of the Org document in current buffer. The exporter evaluates Babel code (see Section 14.5 [Evaluating code blocks], page 204) and removes content specific to other back-ends.

#### Org export commands

`C-c C-e O o` `org-org-export-to-org`  
 Export as an Org file with a `.org` extension. For `myfile.org`, Org exports to `myfile.org.org`, overwriting without warning.

`C-c C-e O O` `org-org-export-as-org`  
 Export to a temporary buffer. Does not create a file.

`C-c C-e O v`  
 Export to an Org file, then open it.

### 12.14 Texinfo export

The ‘`texinfo`’ export back-end generates documents with Texinfo code that can compile to Info format.

#### 12.14.1 Texinfo export commands

`C-c C-e i t` `org-texinfo-export-to-texinfo`  
 Export as a Texinfo file with `.texi` extension. For `myfile.org`, Org exports to `myfile.texi`, overwriting without warning.

`C-c C-e i i` `org-texinfo-export-to-info`  
 Export to Texinfo format first and then process it to make an Info file. To generate other formats, such as DocBook, customize the `org-texinfo-info-process` variable.

#### 12.14.2 Texinfo specific export settings

The Texinfo export back-end has several additional keywords for customizing Texinfo output. Setting these keywords works similar to the general options (see Section 12.2 [Export settings], page 139).

|                                      |   |
|--------------------------------------|---|
| <code>'SUBTITLE'</code>              | The document subtitle.  |
| <code>'SUBAUTHOR'</code>             | The document subauthor.   |
| <code>'TEXINFO_FILENAME'</code>      | The Texinfo filename.   |
| <code>'TEXINFO_CLASS'</code>         | The default document class ( <code>org-texinfo-default-class</code> ), which must be a member of <code>org-texinfo-classes</code> . |
| <code>'TEXINFO_HEADER'</code>        | Arbitrary lines inserted at the end of the header.  |
| <code>'TEXINFO_POST_HEADER'</code>   | Arbitrary lines inserted after the end of the header.   |
| <code>'TEXINFO_DIR_CATEGORY'</code>  | The directory category of the document.   |
| <code>'TEXINFO_DIR_TITLE'</code>     | The directory title of the document.  |
| <code>'TEXINFO_DIR_DESC'</code>      | The directory description of the document.  |
| <code>'TEXINFO_PRINTED_TITLE'</code> | The printed title of the document.  |

### 12.14.3 Texinfo file header

After creating the header for a Texinfo file, the Texinfo back-end automatically generates a name and destination path for the Info file. To override this default with a more sensible path and name, specify the `#+TEXINFO_FILENAME` keyword.

Along with the output's file name, the Texinfo header also contains language details (see Section 12.2 [Export settings], page 139) and encoding system as set in the `org-texinfo-coding-system` variable. Insert `#+TEXINFO_HEADER` keywords for each additional command in the header, for example: `@code{@synindex}`.

Instead of repeatedly installing the same set of commands, define a class in `org-texinfo-classes` once, and then activate it in the document by setting the `#+TEXINFO_CLASS` keyword to that class.

### 12.14.4 Texinfo title and copyright page

The default template for hard copy output has a title page with `#+TITLE` and `#+AUTHOR` (see Section 12.2 [Export settings], page 139). To replace the regular `#+TITLE` with something different for the printed version, use the `#+TEXINFO_PRINTED_TITLE` and `#+SUBTITLE` keywords. Both expect raw Texinfo code for setting their values.

If one `#+AUTHOR` is not sufficient, add multiple `#+SUBAUTHOR` keywords. They have to be set in raw Texinfo code.

```
#+AUTHOR: Jane Smith
#+SUBAUTHOR: John Doe
#+TEXINFO_PRINTED_TITLE: This Long Title@inlinefmt{tex,*} Is Broken in @TeX{}
```

Copying material is defined in a dedicated headline with a non-nil `:COPYING:` property. The back-end inserts the contents within a `@copying` command at the beginning of the document. The heading itself does not appear in the structure of the document.

Copyright information is printed on the back of the title page.

```
* Legalese
:PROPERTIES:
:COPYING: t
:END:
```

This is a short example of a complete Texinfo file, version 1.0.

Copyright \copy 2018 Free Software Foundation, Inc.

### 12.14.5 Info directory file

The end result of the Texinfo export process is the creation of an Info file. This Info file's metadata has variables for category, title, and description: `#+TEXINFO_DIR_CATEGORY`, `#+TEXINFO_DIR_TITLE`, and `#+TEXINFO_DIR_DESC` that establish where in the Info hierarchy the file fits.

Here is an example that writes to the Info directory file:

```
#+TEXINFO_DIR_CATEGORY: Emacs
#+TEXINFO_DIR_TITLE: Org Mode: (org)
#+TEXINFO_DIR_DESC: Outline-based notes management and organizer
```

### 12.14.6 Headings and sectioning structure

The Texinfo export back-end uses a pre-defined scheme to convert Org headlines to an equivalent Texinfo structuring commands. A scheme like this maps top-level headlines to numbered chapters tagged as `@chapter` and lower-level headlines to unnumbered chapters tagged as `@unnumbered`. To override such mappings to introduce `@part` or other Texinfo structuring commands, define a new class in `org-texinfo-classes`. Activate the new class with the `#+TEXINFO_CLASS` keyword. When no new class is defined and activated, the Texinfo export back-end defaults to the `org-texinfo-default-class`.

If an Org headline's level has no associated Texinfo structuring command, or is below a certain threshold (see Section 12.2 [Export settings], page 139), then the Texinfo export back-end makes it into a list item.

The Texinfo export back-end makes any headline with a non-nil `:APPENDIX:` property into an appendix. This happens independent of the Org headline level or the `#+TEXINFO_CLASS`.

The Texinfo export back-end creates a menu entry after the Org headline for each regular sectioning structure. To override this with a shorter menu entry, use the `:ALT_TITLE:` property (see Section 12.3 [Table of contents], page 142). Texinfo menu entries also have an option for a longer `:DESCRIPTION:` property. Here's an example that uses both to override the default menu entry:

```
* Controlling Screen Display
:PROPERTIES:
:ALT_TITLE: Display
:DESCRIPTION: Controlling Screen Display
:END:
```

The text before the first headline belongs to the ‘Top’ node, i.e., the node in which a reader enters an Info manual. As such, it is expected not to appear in printed output generated from the .texi file. See Info file `texinfo`, node ‘The Top Node’, for more information.

### 12.14.7 Indices

The Texinfo export back-end recognizes these indexing keywords if used in the Org file: `#+CINDEX`, `#+FINDEX`, `#+KINDEX`, `#+PINDEX`, `#+TINDEX`, and `#+VININDEX`. Write their value as verbatim Texinfo code; in particular, ‘{’, ‘}’ and ‘@’ characters need to be escaped with ‘@’ if they not belong to a Texinfo command.

```
#+CINDEX: Defining indexing entries
```

For the back-end to generate an index entry for a headline, set the `:INDEX:` property to ‘cp’ or ‘vr’. These abbreviations come from Texinfo that stand for concept index and variable index. The Texinfo manual has abbreviations for all other kinds of indexes. The back-end exports the headline as an unnumbered chapter or section command, and then inserts the index after its contents.

```
* Concept Index
:PROPERTIES:
:INDEX: cp
:END:
```

### 12.14.8 Quoting Texinfo code

Use any of the following three methods to insert or escape raw Texinfo code:

```
Richard @@texinfo:@sc{@@Stallman@@texinfo:}@@ commence' GNU.
```

```
#+TEXINFO: @need800
This paragraph is preceded by...
```

```
#+BEGIN_EXPORT texinfo
@auindex Johnson, Mark
@auindex Lakoff, George
#+END_EXPORT
```

### 12.14.9 Plain lists in Texinfo export

The Texinfo export back-end by default converts description lists in the Org file using the default command `@table`, which results in a table with two columns. To change this behavior, specify `:table-type` with `f`table or `v`table attributes. For more information, See Info file `texinfo`, node ‘Two-column Tables’.

The Texinfo export back-end by default also applies a text highlight based on the defaults stored in `org-texinfo-table-default-markup`. To override the default highlight command, specify another one with the `:indic` attribute.

Org syntax is limited to one entry per list item. Nevertheless, the Texinfo export back-end can split that entry according to any text provided through the `:sep` attribute. Each part then becomes a new entry in the first column of the table.

The following example illustrates all the attributes above:

```
#+ATTR_TEXINFO: :table-type vtable :sep , :indic asis
- foo, bar :: This is the common text for variables foo and bar.
```

becomes

```
@vtable @asis
@item foo
@itemx bar
This is the common text for variables foo and bar.
@end table
```

#### 12.14.10 Tables in Texinfo export

When exporting tables, the Texinfo export back-end uses the widest cell width in each column. To override this and instead specify as fractions of line length, use the `:columns` attribute. See example below.

```
#+ATTR_TEXINFO: :columns .5 .5
| a cell | another cell |
```

#### 12.14.11 Images in Texinfo export

Insert a file link to the image in the Org file, and the Texinfo export back-end inserts the image. These links must have the usual supported image extensions and no descriptions. To scale the image, use `:width` and `:height` attributes. For alternate text, use `:alt` and specify the text using Texinfo code, as shown in the example:

```
#+ATTR_TEXINFO: :width 1in :alt Alternate @i{text}
[[ridt.pdf]]
```

#### 12.14.12 Special blocks

The Texinfo export back-end converts special blocks to commands with the same name. It also adds any `:options` attributes to the end of the command, as shown in this example:

```
#+ATTR_TEXINFO: :options org-org-export-to-org ...
#+begin_defun
A somewhat obsessive function.
#+end_defun
```

becomes

```
@defun org-org-export-to-org ...
A somewhat obsessive function.
@end defun
```

#### 12.14.13 A Texinfo example

Here is a more detailed example Org file. See Section “GNU Sample Texts” in *GNU Texinfo Manual* for an equivalent example using Texinfo code.



```

#+TITLE: GNU Sample {{{version}}}
#+SUBTITLE: for version {{{version}}}, {{{updated}}}
#+AUTHOR: A.U. Thor
#+EMAIL: bug-sample@gnu.org

#+OPTIONS: ':t toc:t author:t email:t
#+LANGUAGE: en

#+MACRO: version 2.0
#+MACRO: updated last updated 4 March 2014

#+TEXINFO_FILENAME: sample.info
#+TEXINFO_HEADER: @syncodeindex pg cp

#+TEXINFO_DIR_CATEGORY: Texinfo documentation system
#+TEXINFO_DIR_TITLE: sample: (sample)
#+TEXINFO_DIR_DESC: Invoking sample

#+TEXINFO_PRINTED_TITLE: GNU Sample

This manual is for GNU Sample (version {{{version}}},
{{{updated}}}).

* Copying
  :PROPERTIES:
  :COPYING:   t
  :END:

This manual is for GNU Sample (version {{{version}}},
{{{updated}}}), which is an example in the Texinfo documentation.

Copyright \copy 2018 Free Software Foundation, Inc.

#+BEGIN_QUOTE
Permission is granted to copy, distribute and/or modify this
document under the terms of the GNU Free Documentation License,
Version 1.3 or any later version published by the Free Software
Foundation; with no Invariant Sections, with no Front-Cover Texts,
and with no Back-Cover Texts. A copy of the license is included in
the section entitled "GNU Free Documentation License".
#+END_QUOTE

* Invoking sample

#+PINDEX: sample
#+CINDEX: invoking @command{sample}

```

This is a sample manual. There is no sample program to invoke, but if there were, you could see its basic usage and command line options here.

```
* GNU Free Documentation License
:PROPERTIES:
:APPENDIX: t
:END:

#+TEXINFO: @include fdl.texi

* Index
:PROPERTIES:
:INDEX:      cp
:END:
```

## 12.15 iCalendar export

A large part of Org mode's inter-operability success is its ability to easily export to or import from external applications. The iCalendar export back-end takes calendar data from Org files and exports to the standard iCalendar format.

The iCalendar export back-end can also incorporate TODO entries based on the configuration of the `org-icalendar-include-todo` variable. The back-end exports plain timestamps as VEVENT, TODO items as VTODO, and also create events from deadlines that are in non-TODO items. The back-end uses the deadlines and scheduling dates in Org TODO items for setting the start and due dates for the iCalendar TODO entry. Consult the `org-icalendar-use-deadline` and `org-icalendar-use-scheduled` variables for more details.

For tags on the headline, the iCalendar export back-end makes them into iCalendar categories. To tweak the inheritance of tags and TODO states, configure the variable `org-icalendar-categories`. To assign clock alarms based on time, configure the `org-icalendar-alarm-time` variable.

The iCalendar format standard requires globally unique identifier—UID—for each entry. The iCalendar export back-end creates UIDs during export. To save a copy of the UID in the Org file set the variable `org-icalendar-store-UID`. The back-end looks for the `:ID:` property of the entry for re-using the same UID for subsequent exports.

Since a single Org entry can result in multiple iCalendar entries—as timestamp, deadline, scheduled item, or TODO item—Org adds prefixes to the UID, depending on which part of the Org entry triggered the creation of the iCalendar entry. Prefixing ensures UIDs remains unique, yet enable synchronization programs trace the connections.

**C-c C-e c f** **org-icalendar-export-to-ics**  
Create iCalendar entries from the current Org buffer and store them in the same directory, using a file extension `.ics`.

**C-c C-e c a** **org-icalendar-export-agenda-files**  
Create iCalendar entries from Org files in `org-agenda-files` and store in a separate iCalendar file for each Org file.

**C-c C-e c c** `org-icalendar-combine-agenda-files`  
 Create a combined iCalendar file from Org files in `org-agenda-files` and write it to `org-icalendar-combined-agenda-file` file name.

The iCalendar export back-end includes `SUMMARY`, `DESCRIPTION`, `LOCATION` and `TIMEZONE` properties from the Org entries when exporting. To force the back-end to inherit the `LOCATION` and `TIMEZONE` properties, configure the `org-use-property-inheritance` variable.

When Org entries do not have `SUMMARY`, `DESCRIPTION` and `LOCATION` properties, the iCalendar export back-end derives the summary from the headline, and derives the description from the body of the Org item. The `org-icalendar-include-body` variable limits the maximum number of characters of the content are turned into its description.

The `TIMEZONE` property can be used to specify a per-entry time zone, and will be applied to any entry with timestamp information. Time zones should be specified as per the IANA time zone database format, e.g. “Asia/Almaty”. Alternately, the property value can be “UTC”, to force UTC time for this entry only.

Exporting to iCalendar format depends in large part on the capabilities of the destination application. Some are more lenient than others. Consult the Org mode FAQ for advice on specific applications.

## 12.16 Other built-in back-ends

Other export back-ends included with Org are:

- `ox-man.el`: export to a man page.

To activate such back-ends, either customize `org-export-backends` or load directly with `(require 'ox-man)`. On successful load, the back-end adds new keys in the export dispatcher (see Section 12.1 [The export dispatcher], page 138).

Follow the comment section of such files, for example, `ox-man.el`, for usage and configuration details.

## 12.17 Advanced configuration

### Hooks

The export process executes two hooks before the actual exporting begins. The first hook, `org-export-before-processing-hook`, runs before any expansions of macros, Babel code, and include keywords in the buffer. The second hook, `org-export-before-parsing-hook`, runs before the buffer is parsed. Both hooks are specified as functions, see example below. Their main use is for heavy duty structural modifications of the Org content. For example, removing every headline in the buffer during export:

```
(defun my-headline-removal (backend)
  "Remove all headlines in the current buffer.
BACKEND is the export back-end being used, as a symbol."
  (org-map-entries
    (lambda () (delete-region (point) (progn (forward-line) (point))))))

(add-hook 'org-export-before-parsing-hook 'my-headline-removal)
```

Note that the hook function must have a mandatory argument that is a symbol for the back-end.

## Filters

The Org export process relies on filters to process specific parts of conversion process. Filters are just lists of functions to be applied to certain parts for a given back-end. The output from the first function in the filter is passed on to the next function in the filter. The final output is the output from the final function in the filter.

The Org export process has many filter sets applicable to different types of objects, plain text, parse trees, export options, and final output formats. The filters are named after the element type or object type: `org-export-filter-TYPE-functions`, where TYPE is the type targeted by the filter. Valid types are:

|                     |                    |                  |
|---------------------|--------------------|------------------|
| body                | bold               | babel-call       |
| center-block        | clock              | code             |
| diary-sexp          | drawer             | dynamic-block    |
| entity              | example-block      | export-block     |
| export-snippet      | final-output       | fixed-width      |
| footnote-definition | footnote-reference | headline         |
| horizontal-rule     | inline-babel-call  | inline-src-block |
| inlinetask          | italic             | item             |
| keyword             | latex-environment  | latex-fragment   |
| line-break          | link               | node-property    |
| options             | paragraph          | parse-tree       |
| plain-list          | plain-text         | planning         |
| property-drawer     | quote-block        | radio-target     |
| section             | special-block      | src-block        |
| statistics-cookie   | strike-through     | 下付き文字            |
| 上付き文字               | table              | table-cell       |
| table-row           | target             | timestamp        |
| underline           | verbatim           | verse-block      |

Here is an example filter that replaces non-breaking spaces ~ in the Org buffer with \_ for the L<sup>A</sup>T<sub>E</sub>X back-end.

```
(defun my-latex-filter-nobreaks (text backend info)
  "Ensure \"_\" are properly handled in LaTeX export."
  (when (org-export-derived-backend-p backend 'latex)
    (replace-regexp-in-string "~" "_" text)))

(add-to-list 'org-export-filter-plain-text-functions
  'my-latex-filter-nobreaks)
```

A filter requires three arguments: the code to be transformed, the name of the back-end, and some optional information about the export process. The third argument can be safely ignored. Note the use of `org-export-derived-backend-p` predicate that tests for `latex` back-end or any other back-end, such as `beamer`, derived from `latex`.

## Defining filters for individual files

The Org export can filter not just for back-ends, but also for specific files through the `#+BIND` keyword. Here is an example with two filters; one removes brackets from time stamps, and the other removes strike-through text. The filter functions are defined in a `'src'` code block in the same Org file, which is a handy location for debugging.

```
#+BIND: org-export-filter-timestamp-functions (tmp-f-timestamp)
#+BIND: org-export-filter-strike-through-functions (tmp-f-strike-through)
#+begin_src emacs-lisp :exports results :results none
  (defun tmp-f-timestamp (s backend info)
    (replace-regexp-in-string "&[lg]t;\\|[] []" "" s))
  (defun tmp-f-strike-through (s backend info) "")
#+end_src
```

## Extending an existing back-end

Some parts of the conversion process can be extended for certain elements so as to introduce a new or revised translation. That is how the HTML export back-end was extended to handle Markdown format. The extensions work seamlessly so any aspect of filtering not done by the extended back-end is handled by the original back-end. Of all the export customization in Org, extending is very powerful as it operates at the parser level.

For this example, make the `ascii` back-end display the language used in a source code block. Also make it display only when some attribute is non-nil, like the following:

```
#+ATTR_ASCII: :language t
```

Then extend `ascii` back-end with a custom `my-ascii` back-end.

```
(defun my-ascii-src-block (src-block contents info)
  "Transcode a SRC-BLOCK element from Org to ASCII.
  CONTENTS is nil. INFO is a plist used as a communication
  channel."
  (if (not (org-export-read-attribute :attr_ascii src-block :language))
      (org-export-with-backend 'ascii src-block contents info)
      (concat
        (format ",--[ %s ]--\n%s`----"
          (org-element-property :language src-block)
          (replace-regexp-in-string
            "^" "| "
            (org-element-normalize-string
              (org-export-format-code-default src-block info))))))

  (org-export-define-derived-backend 'my-ascii 'ascii
    :translate-alist '((src-block . my-ascii-src-block)))
```

The `my-ascii-src-block` function looks at the attribute above the current element. If not true, hands over to `ascii` back-end. If true, which it is in this example, it creates a box around the code and leaves room for the inserting a string for language. The last form creates the new back-end that springs to action only when translating `src-block` type elements.

To use the newly defined back-end, call the following from an Org buffer:

```
(org-export-to-buffer 'my-ascii "*Org MY-ASCII Export*")
```

Further steps to consider would be an interactive function, self-installing an item in the export dispatcher menu, and other user-friendly improvements.

## 12.18 Export in foreign buffers

The export back-ends in Org often include commands to convert selected regions. A convenient feature of this in-place conversion is that the exported output replaces the original source. Here are such functions:

```
org-html-convert-region-to-html  
    Convert the selected region into HTML.
```

```
org-latex-convert-region-to-latex  
    Convert the selected region into LATEX.
```

```
org-texinfo-convert-region-to-texinfo  
    Convert the selected region into Texinfo.
```

```
org-md-convert-region-to-md  
    Convert the selected region into Markdown.
```

In-place conversions are particularly handy for quick conversion of tables and lists in foreign buffers. For example, turn on the minor mode `M-x orgstruct-mode` in an HTML buffer, then use the convenient Org keyboard commands to create a list, select it, and convert it to HTML with `M-x org-html-convert-region-to-html RET`.

## 13 Publishing

Org includes a publishing management system that allows you to configure automatic HTML conversion of *projects* composed of interlinked org files. You can also configure Org to automatically upload your exported HTML pages and related attachments, such as images and source code files, to a web server.

あなたはまた、Org-mode から PDF へ変換することもでき、さらに、サーバ上で両方のファイルを利用できるように HTML と PDF への変換を組み合わせることも可能です。

公開機能は David O'Toole によって Org-mode にコントリビュートされました。

### 13.1 Configuration

公開機能の利用にはプロジェクトのはファイルや公開先、その他多くのプロパティを指定する重要な設定を必要とします。

#### 13.1.1 org-publishing-alist変数

Publishing is configured almost entirely through setting the value of one variable, called `org-publish-project-alist`. Each element of the list configures one project, and may be in one of the two following forms:

```
("project-name" :property value :property value ...)
    i.e., a well-formed property list with alternating keys and values
or
("project-name" :components ("project-name" "project-name" ...))
```

どちらの場合もプロパティと値の指定によって設定されます。プロジェクトは公開されるファイルのセットと、それらが公開される時に使用する設定を定義します。上記の 2 番目の形式を取る場合、`:components` プロパティの個々のメンバは異なる公開設定を持つファイルをまとめたサブプロジェクトとして扱われます。あなたが“メタプロジェクト”を公開する時、全てのコンポーネントは特定の順序で公開されます。

#### 13.1.2 ファイルの送り元と送り先

Most properties are optional, but some should always be set. In particular, Org needs to know where to look for source files, and where to put published files.

|                                    |   |
|------------------------------------|---|
| <code>:base-directory</code>       | 公開するソースファイルを含むディレクトリ  |
| <code>:publishing-directory</code> | Directory where output files will be published. You can directly publish to a web server using a file name syntax appropriate for the Emacs <code>tramp</code> package. Or you can publish to a local directory and use external tools to upload your website (see Section 13.2 [Uploading files], page 196). |
| <code>:preparation-function</code> | Function or list of functions to be called before starting the publishing process, for example, to run <code>make</code> for updating files to be published. Each preparation function is called with a single argument, the project property list.   |

**:completion-function**      Function or list of functions called after finishing the publishing process, for example, to change permissions of the resulting files. Each completion function is called with a single argument, the project property list.

### 13.1.3 Selecting files

デフォルトでは、ベースディレクトリの中の拡張子が`.org`である全てのファイルはプロジェクトの一部であるとみなされます。これはプロパティの設定により変更することができます:

**:base-extension**      ソースファイルの拡張子 (`.`を除きます!) です。これは実際は正規表現です。もしあなたが**:base-directory**の中の全てのファイルを取得したいのであれば、**any**シンボルをセットします。

**:exclude**      そのファイルの拡張子が選択されていたとしても、公開したくないファイル名にマッチする正規表現。

**:include**      **:base-extension**と**:exclude**にかかわらずインクルードするファイルのリスト。

**:recursive**      `non-nil` means, check base-directory recursively for files to publish.

### 13.1.4 Publishing action

Publishing means that a file is copied to the destination directory and possibly transformed in the process. The default transformation is to export Org files as HTML files, and this is done by the function `org-html-publish-to-html`, which calls the HTML exporter (see Section 12.9 [HTML export], page 150). But you also can publish your content as PDF files using `org-latex-publish-to-pdf` or as `ascii`, `Texinfo`, etc., using the corresponding functions.

If you want to publish the Org file as an `.org` file but with the *archived*, *commented* and *tag-excluded* trees removed, use the function `org-org-publish-to-org`. This will produce `file.org` and put it in the publishing directory. If you want a htmlized version of this file, set the parameter **:htmlized-source** to `t`, it will produce `file.org.html` in the publishing directory<sup>1</sup>.

Other files like images only need to be copied to the publishing destination. For this you can use `org-publish-attachment`. For non-org files, you always need to specify the publishing function:

**:publishing-function**      ファイルの公開を実行する関数です。これは、順番に呼び出される関数のリストにすることもできます。

**:htmlized-source**      `non-nil` means, publish htmlized source.

The function must accept three arguments: a property list containing at least a **:publishing-directory** property, the name of the file to be published and the path to the publishing directory of the output file. It should take the specified file, make the necessary transformation (if any) and place the result into the destination folder.

---

<sup>1</sup> If the publishing directory is the same than the source directory, `file.org` will be exported as `file.org.org`, so probably don't want to do this.



### 13.1.5 Options for the exporters

The property list can be used to set export options during the publishing process. In most cases, these properties correspond to user variables in Org. While some properties are available for all export back-ends, most of them are back-end specific. The following sections list properties along with the variable they belong to. See the documentation string of these options for details.

When a property is given a value in `org-publish-project-alist`, its setting overrides the value of the corresponding user variable (if any) during publishing. Options set within a file (see Section 12.2 [Export settings], page 139), however, override everything.

#### Generic properties

|                                    |   |
|------------------------------------|---|
| <code>:archived-trees</code>       | <code>org-export-with-archived-trees</code>   |
| <code>:exclude-tags</code>         | <code>org-export-exclude-tags</code>          |
| <code>:headline-levels</code>      | <code>org-export-headline-levels</code>       |
| <code>:language</code>             | <code>org-export-default-language</code>      |
| <code>:preserve-breaks</code>      | <code>org-export-preserve-breaks</code>       |
| <code>:section-numbers</code>      | <code>org-export-with-section-numbers</code>  |
| <code>:select-tags</code>          | <code>org-export-select-tags</code>           |
| <code>:with-author</code>          | <code>org-export-with-author</code>           |
| <code>:with-broken-links</code>    | <code>org-export-with-broken-links</code>     |
| <code>:with-clocks</code>          | <code>org-export-with-clocks</code>           |
| <code>:with-creator</code>         | <code>org-export-with-creator</code>          |
| <code>:with-date</code>            | <code>org-export-with-date</code>             |
| <code>:with-drawers</code>         | <code>org-export-with-drawers</code>          |
| <code>:with-email</code>           | <code>org-export-with-email</code>            |
| <code>:with-emphasize</code>       | <code>org-export-with-emphasize</code>        |
| <code>:with-fixed-width</code>     | <code>org-export-with-fixed-width</code>      |
| <code>:with-footnotes</code>       | <code>org-export-with-footnotes</code>        |
| <code>:with-latex</code>           | <code>org-export-with-latex</code>            |
| <code>:with-planning</code>        | <code>org-export-with-planning</code>         |
| <code>:with-priority</code>        | <code>org-export-with-priority</code>         |
| <code>:with-properties</code>      | <code>org-export-with-properties</code>       |
| <code>:with-special-strings</code> | <code>org-export-with-special-strings</code>  |
| <code>:with-sub-superscript</code> | <code>org-export-with-sub-superscripts</code> |
| <code>:with-tables</code>          | <code>org-export-with-tables</code>           |
| <code>:with-tags</code>            | <code>org-export-with-tags</code>             |
| <code>:with-tasks</code>           | <code>org-export-with-tasks</code>            |
| <code>:with-timestamps</code>      | <code>org-export-with-timestamps</code>       |
| <code>:with-title</code>           | <code>org-export-with-title</code>            |
| <code>:with-toc</code>             | <code>org-export-with-toc</code>              |
| <code>:with-todo-keywords</code>   | <code>org-export-with-todo-keywords</code>    |

#### ASCII specific properties

|                                   |                                      |
|-----------------------------------|--------------------------------------|
| <code>:ascii-bullets</code>       | <code>org-ascii-bullets</code>       |
| <code>:ascii-caption-above</code> | <code>org-ascii-caption-above</code> |
| <code>:ascii-charset</code>       | <code>org-ascii-charset</code>       |

|                                      |   |
|--------------------------------------|---|
| :ascii-global-margin                 | org-ascii-global-margin                 |
| :ascii-format-drawer-function        | org-ascii-format-drawer-function        |
| :ascii-format-inlinetask-function    | org-ascii-format-inlinetask-function    |
| :ascii-headline-spacing              | org-ascii-headline-spacing              |
| :ascii-indented-line-width           | org-ascii-indented-line-width           |
| :ascii-inlinetask-width              | org-ascii-inlinetask-width              |
| :ascii-inner-margin                  | org-ascii-inner-margin                  |
| :ascii-links-to-notes                | org-ascii-links-to-notes                |
| :ascii-list-margin                   | org-ascii-list-margin                   |
| :ascii-paragraph-spacing             | org-ascii-paragraph-spacing             |
| :ascii-quote-margin                  | org-ascii-quote-margin                  |
| :ascii-table-keep-all-vertical-lines | org-ascii-table-keep-all-vertical-lines |
| :ascii-table-use-ascii-art           | org-ascii-table-use-ascii-art           |
| :ascii-table-widen-columns           | org-ascii-table-widen-columns           |
| :ascii-text-width                    | org-ascii-text-width                    |
| :ascii-underline                     | org-ascii-underline                     |
| :ascii-verbatim-format               | org-ascii-verbatim-format               |

### Beamer specific properties

|                               |                                  |
|-------------------------------|----------------------------------|
| :beamer-theme                 | org-beamer-theme                 |
| :beamer-column-view-format    | org-beamer-column-view-format    |
| :beamer-environments-extra    | org-beamer-environments-extra    |
| :beamer-frame-default-options | org-beamer-frame-default-options |
| :beamer-outline-frame-options | org-beamer-outline-frame-options |
| :beamer-outline-frame-title   | org-beamer-outline-frame-title   |
| :beamer-subtitle-format       | org-beamer-subtitle-format       |

### HTML specific properties

|                                       |  |
|---------------------------------------|--|
| :html-allow-name-attribute-in-anchors | org-html-allow-name-attribute-in-anchors |
| :html-checkbox-type                   | org-html-checkbox-type                   |
| :html-container                       | org-html-container-element               |
| :html-divs                            | org-html-divs                            |
| :html-doctype                         | org-html-doctype                         |
| :html-extension                       | org-html-extension                       |
| :html-footnote-format                 | org-html-footnote-format                 |
| :html-footnote-separator              | org-html-footnote-separator              |
| :html-footnotes-section               | org-html-footnotes-section               |
| :html-format-drawer-function          | org-html-format-drawer-function          |
| :html-format-headline-function        | org-html-format-headline-function        |
| :html-format-inlinetask-function      | org-html-format-inlinetask-function      |
| :html-head-extra                      | org-html-head-extra                      |
| :html-head-include-default-style      | org-html-head-include-default-style      |
| :html-head-include-scripts            | org-html-head-include-scripts            |
| :html-head                            | org-html-head                            |
| :html-home/up-format                  | org-html-home/up-format                  |
| :html-html5-fancy                     | org-html-html5-fancy                     |

|  |   |
|--|---|
| :html-indent                                 | org-html-indent                                 |
| :html-infojs-options                         | org-html-infojs-options                         |
| :html-infojs-template                        | org-html-infojs-template                        |
| :html-inline-image-rules                     | org-html-inline-image-rules                     |
| :html-inline-images                          | org-html-inline-images                          |
| :html-link-home                              | org-html-link-home                              |
| :html-link-org-files-as-html                 | org-html-link-org-files-as-html                 |
| :html-link-up                                | org-html-link-up                                |
| :html-link-use-abs-url                       | org-html-link-use-abs-url                       |
| :html-mathjax-options                        | org-html-mathjax-options                        |
| :html-mathjax-template                       | org-html-mathjax-template                       |
| :html-metadata-timestamp-format              | org-html-metadata-timestamp-format              |
| :html-postamble-format                       | org-html-postamble-format                       |
| :html-postamble                              | org-html-postamble                              |
| :html-preamble-format                        | org-html-preamble-format                        |
| :html-preamble                               | org-html-preamble                               |
| :html-table-align-individual-fields          | org-html-table-align-individual-fields          |
| :html-table-attributes                       | org-html-table-default-attributes               |
| :html-table-caption-above                    | org-html-table-caption-above                    |
| :html-table-data-tags                        | org-html-table-data-tags                        |
| :html-table-header-tags                      | org-html-table-header-tags                      |
| :html-table-row-tags                         | org-html-table-row-tags                         |
| :html-table-use-header-tags-for-first-column | org-html-table-use-header-tags-for-first-column |
| :html-tag-class-prefix                       | org-html-tag-class-prefix                       |
| :html-text-markup-alist                      | org-html-text-markup-alist                      |
| :html-todo-kwd-class-prefix                  | org-html-todo-kwd-class-prefix                  |
| :html-toplevel-hlevel                        | org-html-toplevel-hlevel                        |
| :html-use-infojs                             | org-html-use-infojs                             |
| :html-validation-link                        | org-html-validation-link                        |
| :html-viewport                               | org-html-viewport                               |
| :html-xml-declaration                        | org-html-xml-declaration                        |

## **L<sup>A</sup>T<sub>E</sub>X** specific properties

|                                  |                                     |
|----------------------------------|-------------------------------------|
| :latex-active-timestamp-format   | org-latex-active-timestamp-format   |
| :latex-caption-above             | org-latex-caption-above             |
| :latex-classes                   | org-latex-classes                   |
| :latex-class                     | org-latex-default-class             |
| :latex-compiler                  | org-latex-compiler                  |
| :latex-default-figure-position   | org-latex-default-figure-position   |
| :latex-default-table-environment | org-latex-default-table-environment |
| :latex-default-table-mode        | org-latex-default-table-mode        |
| :latex-diary-timestamp-format    | org-latex-diary-timestamp-format    |
| :latex-footnote-defined-format   | org-latex-footnote-defined-format   |
| :latex-footnote-separator        | org-latex-footnote-separator        |
| :latex-format-drawer-function    | org-latex-format-drawer-function    |
| :latex-format-headline-function  | org-latex-format-headline-function  |

|   |  |
|---|--|
| <code>:latex-format-inlinetask-function</code>    | <code>org-latex-format-inlinetask-function</code>    |
| <code>:latex-hyperref-template</code>             | <code>org-latex-hyperref-template</code>             |
| <code>:latex-image-default-height</code>          | <code>org-latex-image-default-height</code>          |
| <code>:latex-image-default-option</code>          | <code>org-latex-image-default-option</code>          |
| <code>:latex-image-default-width</code>           | <code>org-latex-image-default-width</code>           |
| <code>:latex-images-centered</code>               | <code>org-latex-images-centered</code>               |
| <code>:latex-inactive-timestamp-format</code>     | <code>org-latex-inactive-timestamp-format</code>     |
| <code>:latex-inline-image-rules</code>            | <code>org-latex-inline-image-rules</code>            |
| <code>:latex-link-with-unknown-path-format</code> | <code>org-latex-link-with-unknown-path-format</code> |
| <code>:latex-listings-langs</code>                | <code>org-latex-listings-langs</code>                |
| <code>:latex-listings-options</code>              | <code>org-latex-listings-options</code>              |
| <code>:latex-listings</code>                      | <code>org-latex-listings</code>                      |
| <code>:latex-minted-langs</code>                  | <code>org-latex-minted-langs</code>                  |
| <code>:latex-minted-options</code>                | <code>org-latex-minted-options</code>                |
| <code>:latex-prefer-user-labels</code>            | <code>org-latex-prefer-user-labels</code>            |
| <code>:latex-subtitle-format</code>               | <code>org-latex-subtitle-format</code>               |
| <code>:latex-subtitle-separate</code>             | <code>org-latex-subtitle-separate</code>             |
| <code>:latex-table-scientific-notation</code>     | <code>org-latex-table-scientific-notation</code>     |
| <code>:latex-tables-booktabs</code>               | <code>org-latex-tables-booktabs</code>               |
| <code>:latex-tables-centered</code>               | <code>org-latex-tables-centered</code>               |
| <code>:latex-text-markup-alist</code>             | <code>org-latex-text-markup-alist</code>             |
| <code>:latex-title-command</code>                 | <code>org-latex-title-command</code>                 |
| <code>:latex-toc-command</code>                   | <code>org-latex-toc-command</code>                   |

## Markdown specific properties

|                                    |                                       |
|------------------------------------|---------------------------------------|
| <code>:md-footnote-format</code>   | <code>org-md-footnote-format</code>   |
| <code>:md-footnotes-section</code> | <code>org-md-footnotes-section</code> |
| <code>:md-headline-style</code>    | <code>org-md-headline-style</code>    |

## ODT specific properties

|  |   |
|--|---|
| <code>:odt-content-template-file</code>      | <code>org-odt-content-template-file</code>      |
| <code>:odt-display-outline-level</code>      | <code>org-odt-display-outline-level</code>      |
| <code>:odt-fontify-srcblocks</code>          | <code>org-odt-fontify-srcblocks</code>          |
| <code>:odt-format-drawer-function</code>     | <code>org-odt-format-drawer-function</code>     |
| <code>:odt-format-headline-function</code>   | <code>org-odt-format-headline-function</code>   |
| <code>:odt-format-inlinetask-function</code> | <code>org-odt-format-inlinetask-function</code> |
| <code>:odt-inline-formula-rules</code>       | <code>org-odt-inline-formula-rules</code>       |
| <code>:odt-inline-image-rules</code>         | <code>org-odt-inline-image-rules</code>         |
| <code>:odt-pixels-per-inch</code>            | <code>org-odt-pixels-per-inch</code>            |
| <code>:odt-styles-file</code>                | <code>org-odt-styles-file</code>                |
| <code>:odt-table-styles</code>               | <code>org-odt-table-styles</code>               |
| <code>:odt-use-date-fields</code>            | <code>org-odt-use-date-fields</code>            |

## Texinfo specific properties

|   |  |
|---|--|
| <code>:texinfo-active-timestamp-format</code> | <code>org-texinfo-active-timestamp-format</code> |
| <code>:texinfo-classes</code>                 | <code>org-texinfo-classes</code>                 |

|   |  |
|---|--|
| <code>:texinfo-class</code>                         | <code>org-texinfo-default-class</code>                 |
| <code>:texinfo-table-default-markup</code>          | <code>org-texinfo-table-default-markup</code>          |
| <code>:texinfo-diary-timestamp-format</code>        | <code>org-texinfo-diary-timestamp-format</code>        |
| <code>:texinfo-filename</code>                      | <code>org-texinfo-filename</code>                      |
| <code>:texinfo-format-drawer-function</code>        | <code>org-texinfo-format-drawer-function</code>        |
| <code>:texinfo-format-headline-function</code>      | <code>org-texinfo-format-headline-function</code>      |
| <code>:texinfo-format-inlinetask-function</code>    | <code>org-texinfo-format-inlinetask-function</code>    |
| <code>:texinfo-inactive-timestamp-format</code>     | <code>org-texinfo-inactive-timestamp-format</code>     |
| <code>:texinfo-link-with-unknown-path-format</code> | <code>org-texinfo-link-with-unknown-path-format</code> |
| <code>:texinfo-node-description-column</code>       | <code>org-texinfo-node-description-column</code>       |
| <code>:texinfo-table-scientific-notation</code>     | <code>org-texinfo-table-scientific-notation</code>     |
| <code>:texinfo-tables-verbatim</code>               | <code>org-texinfo-tables-verbatim</code>               |
| <code>:texinfo-text-markup-alist</code>             | <code>org-texinfo-text-markup-alist</code>             |

### 13.1.6 公開ファイル間のリンク

To create a link from one Org file to another, you would use something like ‘[[file:foo.org] [The foo]]’ or simply ‘file:foo.org’ (see Section 4.3 [External links], page 39). When published, this link becomes a link to `foo.html`. You can thus interlink the pages of your “org web” project and the links will work as expected when you publish them to HTML. If you also publish the Org source file and want to link to it, use an `http:` link instead of a `file:` link, because `file:` links are converted to link to the corresponding `html` file.

You may also link to related files, such as images. Provided you are careful with relative file names, and provided you have also configured Org to upload the related files, these links will work too. See Section 13.3.2 [Complex example], page 197, for an example of this usage.

Eventually, links between published documents can contain some search options (see Section 4.7 [Search options], page 45), which will be resolved to the appropriate location in the linked file. For example, once published to HTML, the following links all point to a dedicated anchor in `foo.html`.

```
[[file:foo.org::*heading]]
[[file:foo.org::*custom-id]]
[[file:foo.org::*target]]
```

### 13.1.7 サイトマップの生成

以下のプロパティはプロジェクトのファイルのマップの公開を制御するために使うことができます。

|                                |   |
|--------------------------------|---|
| <code>:auto-sitemap</code>     | When non-nil, publish a sitemap during <code>org-publish-current-project</code> or <code>org-publish-all</code> . |
| <code>:sitemap-filename</code> | Filename for output of sitemap. Defaults to <code>sitemap.org</code> (which becomes <code>sitemap.html</code> ).  |
| <code>:sitemap-title</code>    | Title of sitemap page. Defaults to name of file.  |

|                                    |   |
|------------------------------------|---|
| <code>:sitemap-format-entry</code> | With this option one can tell how a site-map entry is formatted in the site-map. It is a function called with three arguments: the file or directory name relative to base directory of the project, the site-map style and the current project. It is expected to return a string. Default value turns file names into links and use document titles as descriptions. For specific formatting needs, one can use <code>org-publish-find-date</code> , <code>org-publish-find-title</code> and <code>org-publish-find-property</code> , to retrieve additional information about published documents. |
| <code>:sitemap-function</code>     | Plug-in function to use for generation of the sitemap. It is called with two arguments: the title of the site-map and a representation of the files and directories involved in the project as a radio list (see Section A.6.4 [Radio lists], page 247). The latter can further be transformed using <code>org-list-to-generic</code> , <code>org-list-to-subtree</code> and alike. Default value generates a plain list of links to all files in the project.  |
| <code>:sitemap-sort-folders</code> | Where folders should appear in the sitemap. Set this to <code>first</code> (default) or <code>last</code> to display folders first or last, respectively. When set to <code>ignore</code> , folders are ignored altogether. Any other value will mix files and folders. This variable has no effect when site-map style is <code>tree</code> .  |
| <code>:sitemap-sort-files</code>   | How the files are sorted in the site map. Set this to <code>alphabetically</code> (default), <code>chronologically</code> or <code>anti-chronologically</code> . <code>chronologically</code> sorts the files with older date first while <code>anti-chronologically</code> sorts the files with newer date first. <code>alphabetically</code> sorts the files alphabetically. The date of a file is retrieved with <code>org-publish-find-date</code> .  |
| <code>:sitemap-ignore-case</code>  | ソート時に大文字と小文字を区別するかどうかです。デフォルトでは <code>nil</code> です。  |
| <code>:sitemap-date-format</code>  | Format string for the <code>format-time-string</code> function that tells how a sitemap entry's date is to be formatted. This property bypasses <code>org-publish-sitemap-date-format</code> which defaults to <code>%Y-%m-%d</code> .  |

### 13.1.8 Generating an index

Org mode can generate an index across the files of a publishing project.

|                         |   |
|-------------------------|---|
| <code>:makeindex</code> | When non- <code>nil</code> , generate an index in the file <code>theindex.org</code> and publish it as <code>theindex.html</code> . |
|-------------------------|---|

The file will be created when first publishing a project with the `:makeindex` set. The file only contains a statement `#+INCLUDE: "theindex.inc"`. You can then build around this include statement by adding a title, style information, etc.

Index entries are specified with `#+INDEX` keyword. An entry that contains an exclamation mark will create a sub item.

```
* Curriculum Vitae
#+INDEX: CV
#+INDEX: Application!CV
```

## 13.2 Uploading files

For those people already utilizing third party sync tools such as `rsync` or `unison`, it might be preferable not to use the built in *remote* publishing facilities of Org mode which rely heavily on Tramp. Tramp, while very useful and powerful, tends not to be so efficient for multiple file transfer and has been known to cause problems under heavy usage.

Specialized synchronization utilities offer several advantages. In addition to timestamp comparison, they also do content and permissions/attribute checks. For this reason you might prefer to publish your web to a local directory (possibly even *in place* with your Org files) and then use `unison` or `rsync` to do the synchronization with the remote host.

Since Unison (for example) can be configured as to which files to transfer to a certain remote destination, it can greatly simplify the project publishing definition. Simply keep all files in the correct location, process your Org files with `org-publish` and let the synchronization tool do the rest. You do not need, in this scenario, to include attachments such as `jpg`, `css` or `gif` files in the project definition since the 3rd party tool syncs them.

Publishing to a local directory is also much faster than to a remote one, so that you can afford more easily to republish entire projects. If you set `org-publish-use-timestamps-flag` to `nil`, you gain the main benefit of re-including any changed external files such as source example files you might include with `#+INCLUDE:.`. The timestamp mechanism in Org is not smart enough to detect if included files have been modified.

## 13.3 Sample configuration

私達は2つの設定の例を以下に示します。1つ目の設定はOrg-mode ファイルのみのシンプルなプロジェクトの公開です。2つ目の設定は複数のコンポーネントを持つプロジェクトで、少し複雑な例です。

### 13.3.1 例：シンプルな公開用の設定

This example publishes a set of Org files to the `public_html` directory on the local machine.

```
(setq org-publish-project-alist
  '(("org"
     :base-directory "~/org/"
     :publishing-directory "~/public_html"
     :publishing-function org-html-publish-to-html
     :section-numbers nil
     :with-toc nil
     :html-head "<link rel=\"stylesheet\"
                href=\"../other/mystyle.css\"
                type=\"text/css\"/>"))))
```

### 13.3.2 例：複雑な公開用の設定

This more complicated example publishes an entire website, including Org files converted to HTML, image files, Emacs Lisp source code, and style sheets. The publishing directory is remote and private files are excluded.

To ensure that links are preserved, care should be taken to replicate your directory structure on the web server, and to use relative file paths. For example, if your Org files are kept in `~/org` and your publishable images in `~/images`, you would link to an image with

```
file:../images/myimage.png
```

On the web server, the relative path to the image should be the same. You can accomplish this by setting up an "images" folder in the right place on the web server, and publishing images to it.

```
(setq org-publish-project-alist
  '(("orgfiles"
    :base-directory "~/org/"
    :base-extension "org"
    :publishing-directory "/ssh:user@host:~/html/notebook/"
    :publishing-function org-html-publish-to-html
    :exclude "PrivatePage.org" ;; regexp
    :headline-levels 3
    :section-numbers nil
    :with-toc nil
    :html-head "<link rel=\"stylesheet\"
              href=\"../other/mystyle.css\" type=\"text/css\"/>"
    :html-preamble t)

    ("images"
     :base-directory "~/images/"
     :base-extension "jpg\\|gif\\|png"
     :publishing-directory "/ssh:user@host:~/html/images/"
     :publishing-function org-publish-attachment)

    ("other"
     :base-directory "~/other/"
     :base-extension "css\\|el"
     :publishing-directory "/ssh:user@host:~/html/other/"
     :publishing-function org-publish-attachment)

    ("website" :components ("orgfiles" "images" "other"))))
```

## 13.4 Triggering publication

一度適切に設定すると、Org-mode は以下のコマンドを使うことで公開することができます。

```
C-c C-e P x                                org-publish
      Prompt for a specific project and publish all files that belong to it.

C-c C-e P p                                org-publish-current-project
      Publish the project containing the current file.
```



`C-c C-e P f` `org-publish-current-file`  
Publish only the current file.

`C-c C-e P a` `org-publish-all`  
Publish every project.

Org uses timestamps to track when a file has changed. The above functions normally only publish changed files. You can override this and force publishing of all files by giving a prefix argument to any of the commands above, or by customizing the variable `org-publish-use-timestamps-flag`. This may be necessary in particular if files include other files via `#+SETUPFILE:` or `#+INCLUDE:`.

## 14 Working with source code

Source code here refers to any code typed in Org mode documents. Org can manage source code in any Org file once such code is tagged with begin and end markers. Working with source code begins with tagging source code blocks. Tagged ‘src’ code blocks are not restricted to the preamble or the end of an Org document; they can go anywhere—with a few exceptions, such as not inside comments and fixed width areas. Here’s a sample ‘src’ code block in emacs-lisp:

```
#+BEGIN_SRC emacs-lisp
  (defun org-xor (a b)
    "Exclusive or."
    (if a (not b) b))
#+END_SRC
```

Org can take the code in the block between the `#+BEGIN_SRC` and `#+END_SRC` tags, and format, compile, execute, and show the results. Org can simplify many housekeeping tasks essential to modern code maintenance. That’s why these blocks in Org mode literature are sometimes referred to as ‘live code’ blocks (as compared to the static text and documentation around it). Users can control how ‘live’ they want each block by tweaking the headers for compiling, execution, extraction.

Org’s ‘src’ code block type is one of many block types, such as quote, export, verse, latex, example, and verbatim. This section pertains to `src` code blocks between `#+BEGIN_SRC` and `#+END_SRC`

For editing ‘src’ code blocks, Org provides native Emacs major-modes. That leverages the latest Emacs features for that source code language mode.

For exporting, Org can then extract ‘src’ code blocks into compilable source files (in a conversion process known as *tangling* in literate programming terminology).

For publishing, Org’s back-ends can handle the ‘src’ code blocks and the text for output to a variety of formats with native syntax highlighting.

For executing the source code in the ‘src’ code blocks, Org provides facilities that glue the tasks of compiling, collecting the results of the execution, and inserting them back to the Org file. Besides text output, results may include links to other data types that Emacs can handle: audio, video, and graphics.

An important feature of Org’s execution of the ‘src’ code blocks is passing variables, functions, and results between ‘src’ blocks. Such interoperability uses a common syntax even if these ‘src’ blocks are in different source code languages. The integration extends to linking the debugger’s error messages to the line in the ‘src’ code block in the Org file. That should partly explain why this functionality by the original contributors, Eric Schulte and Dan Davison, was called ‘Org Babel’.

In literate programming, the main appeal is code and documentation co-existing in one file. Org mode takes this several steps further. First by enabling execution, and then by inserting results of that execution back into the Org file. Along the way, Org provides extensive formatting features, including handling tables. Org handles multiple source code languages in one file, and provides a common syntax for passing variables, functions, and results between ‘src’ code blocks.

Org mode fulfills the promise of easy verification and maintenance of publishing reproducible research by keeping all these in the same file: text, data, code, configuration settings of the execution environment, the results of the execution, and associated narratives, claims, references, and internal and external links.

Details of Org’s facilities for working with source code are shown next.

## 14.1 Structure of code blocks

Org offers two ways to structure source code in Org documents: in a ‘src’ block, and directly inline. Both specifications are shown below.

A ‘src’ block conforms to this structure:

```
#+NAME: <name>
#+BEGIN_SRC <language> <switches> <header arguments>
  <body>
#+END_SRC
```

Org mode’s templates system (see Section 15.2 [Easy templates], page 229) speeds up creating ‘src’ code blocks with just three keystrokes. Do not be put-off by having to remember the source block syntax. Org also works with other completion systems in Emacs, some of which predate Org and have custom domain-specific languages for defining templates. Regular use of templates reduces errors, increases accuracy, and maintains consistency.

An inline code block conforms to this structure:

```
src_<language>{<body>}
もしくは
src_<language>[<header arguments>]{<body>}
```

**#+NAME: <name>**

Optional. Names the ‘src’ block so it can be called, like a function, from other ‘src’ blocks or inline blocks to evaluate or to capture the results. Code from other blocks, other files, and from table formulas (see Section 3.5 [The spreadsheet], page 24) can use the name to reference a ‘src’ block. This naming serves the same purpose as naming Org tables. Org mode requires unique names. For duplicate names, Org mode’s behavior is undefined.

**#+BEGIN\_SRC**

**#+END\_SRC**

Mandatory. They mark the start and end of a block that Org requires. The **#+BEGIN\_SRC** line takes additional arguments, as described next.

**<language>**

Mandatory for live code blocks. It is the identifier of the source code language in the block. See Section 14.7 [Languages], page 205, for identifiers of supported languages.

**<switches>**

Optional. Switches provide finer control of the code execution, export, and format (see the discussion of switches in Section 11.5 [Literal examples], page 132)

**<header arguments>**

Optional. Heading arguments control many aspects of evaluation, export and tangling of code blocks (see Section 14.8 [Header arguments], page 206). Using Org’s properties feature, header arguments can be selectively applied to the entire buffer or specific sub-trees of the Org document.

**source code, header arguments**

**<body>** Source code in the dialect of the specified language identifier.

## 14.2 Editing source code

**C-c '** for editing the current code block. It opens a new major-mode edit buffer containing the body of the ‘src’ code block, ready for any edits. **C-c '** again to close the buffer and return to the Org buffer.

**C-x C-s** saves the buffer and updates the contents of the Org buffer.

Set **org-edit-src-auto-save-idle-delay** to save the base buffer after a certain idle delay time.

Set **org-edit-src-turn-on-auto-save** to auto-save this buffer into a separate file using **auto-save-mode**.

**C-c '** to close the major-mode buffer and return back to the Org buffer.

While editing the source code in the major-mode, the **org-src-mode** minor mode remains active. It provides these customization variables as described below. For even more variables, look in the customization group **org-edit-structure**.

**org-src-lang-modes**

If an Emacs major-mode named **<lang>-mode** exists, where **<lang>** is the language identifier from code block’s header line, then the edit buffer uses that major-mode. Use this variable to arbitrarily map language identifiers to major modes.

**org-src-window-setup**

For specifying Emacs window arrangement when the new edit buffer is created.

**org-src-preserve-indentation**

Default is **nil**. Source code is indented. This indentation applies during export or tangling, and depending on the context, may alter leading spaces and tabs. When non-**nil**, source code is aligned with the leftmost column. No lines are modified during export or tangling, which is very useful for white-space sensitive languages, such as Python.

**org-src-ask-before-returning-to-edit-buffer**

When **nil**, Org returns to the edit buffer without further prompts. The default prompts for a confirmation.

Set **org-src-fontify-natively** to non-**nil** to turn on native code fontification in the Org buffer. Fontification of ‘src’ code blocks can give visual separation of text and code on the display page. To further customize the appearance of **org-block** for specific languages, customize **org-src-block-faces**. The following example shades the background of regular blocks, and colors source blocks only for Python and Emacs-Lisp languages.

```
(require 'color)
(set-face-attribute 'org-block nil :background
  (color-darken-name
    (face-attribute 'default :background) 3))

(setq org-src-block-faces '(("emacs-lisp" (:background "#EEE2FF"))
  ("python" (:background "#E5FFB8"))))
```

### 14.3 Exporting code blocks

Org can flexibly export just the *code* from the code blocks, just the *results* of evaluation of the code block, *both* the code and the results of the code block evaluation, or *none*. Org defaults to exporting *code* for most languages. For some languages, such as *ditaa*, Org defaults to *results*. To export just the body of code blocks, see Section 11.5 [Literal examples], page 132. To selectively export sub-trees of an Org document, see Chapter 12 [Exporting], page 138.

The `:exports` header arguments control exporting code blocks only and not inline code:

ヘッダー引数:

`:exports code`

This is the default for most languages where the body of the code block is exported. See Section 11.5 [Literal examples], page 132 for more.

`:exports results`

On export, Org includes only the results and not the code block. After each evaluation, Org inserts the results after the end of code block in the Org buffer. By default, Org replaces any previous results. Org can also append results.

`:exports both`

Org exports both the code block and the results.

`:exports none`

Org does not export the code block nor the results.

To stop Org from evaluating code blocks to speed exports, use the header argument `:eval never-export` (see Section 14.8.2.25 [eval], page 223). To stop Org from evaluating code blocks for greater security, set the `org-export-use-babel` variable to `nil`, but understand that header arguments will have no effect.

Turning off evaluation comes in handy when batch processing. For example, markup languages for wikis, which have a high risk of untrusted code. Stopping code block evaluation also stops evaluation of all header arguments of the code block. This may not be desirable in some circumstances. So during export, to allow evaluation of just the header arguments but not any code evaluation in the source block, set `:eval never-export` (see Section 14.8.2.25 [eval], page 223).

Org never evaluates code blocks in commented sub-trees when exporting (see Section 12.6 [Comment lines], page 145). On the other hand, Org does evaluate code blocks in sub-trees excluded from export (see Section 12.2 [Export settings], page 139).

## 14.4 Extracting source code

Extracting source code from code blocks is a basic task in literate programming. Org has features to make this easy. In literate programming parlance, documents on creation are *woven* with code and documentation, and on export, the code is *tangled* for execution by a computer. Org facilitates weaving and tangling for producing, maintaining, sharing, and exporting literate programming documents. Org provides extensive customization options for extracting source code.

When Org tangles ‘src’ code blocks, it expands, merges, and transforms them. Then Org recomposes them into one or more separate files, as configured through the options. During this *tangling* process, Org expands variables in the source code, and resolves any Noweb style references (see Section 14.10 [Noweb reference syntax], page 226).

### Header arguments

**:tangle no**

By default, Org does not tangle the ‘src’ code block on export.

**:tangle yes**

Org extracts the contents of the code block for the tangled output. By default, the output file name is the same as the Org file but with a file extension derived from the language identifier of the ‘src’ code block.

**:tangle filename**

Override the default file name with this one for the tangled output.

関数

**org-babel-tangle**

現在のファイルを tangle します。キーバインドは *C-c C-v t* です。

With prefix argument only tangle the current ‘src’ code block.

**org-babel-tangle-file**

tangle するファイルを選びます。キーバインドは *C-c C-v f* です。

### Hooks

**org-babel-post-tangle-hook**

This hook runs from within code tangled by **org-babel-tangle**, making it suitable for post-processing, compilation, and evaluation of code in the tangled files.

### Jumping between code and Org

Debuggers normally link errors and messages back to the source code. But for tangled files, we want to link back to the Org file, not to the tangled source file. To make this extra jump, Org uses **org-babel-tangle-jump-to-org** function with two additional source code block header arguments: One, set **padline** (see Section 14.8.2.12 [padline], page 216) to true (the default setting). Two, set **comments** (see Section 14.8.2.11 [comments], page 216) to **link**, which makes Org insert links to the Org file.

## 14.5 Evaluating code blocks

A note about security: With code evaluation comes the risk of harm. Org safeguards by prompting for user's permission before executing any code in the source block. To customize this safeguard (or disable it) see Section 15.4 [Code evaluation security], page 230.

Org captures the results of the 'src' code block evaluation and inserts them in the Org file, right after the 'src' code block. The insertion point is after a newline and the `#+RESULTS` label. Org creates the `#+RESULTS` label if one is not already there.

By default, Org enables only `emacs-lisp` 'src' code blocks for execution. See Section 14.7 [Languages], page 205 for identifiers to enable other languages.

Org provides many ways to execute 'src' code blocks. `C-c C-c` or `C-c C-v e` with the point on a 'src' code block<sup>1</sup> calls the `org-babel-execute-src-block` function, which executes the code in the block, collects the results, and inserts them in the buffer.

By calling a named code block<sup>2</sup> from an Org mode buffer or a table. Org can call the named 'src' code blocks from the current Org mode buffer or from the "Library of Babel" (see Section 14.6 [Library of Babel], page 205). Whether inline syntax or the `#+CALL:` syntax is used, the result is wrapped based on the variable `org-babel-inline-result-wrap`, which by default is set to `"= %s ="` to produce verbatim text suitable for markup.

The syntax for `#+CALL:` is

```
#+CALL: <name>(<arguments>)
#+CALL: <name>[<inside header arguments>](<arguments>) <end header arguments>
```

The syntax for inline named code block is

```
... call_<name>(<arguments>) ...
... call_<name>[<inside header arguments>](<arguments>)[<end header arguments>] ...
```

**<name>** This is the name of the code block to be evaluated (see Section 14.1 [Structure of code blocks], page 200).

**<arguments>**

Org passes arguments to the code block using standard function call syntax. For example, a `#+CALL:` line that passes '4' to a code block named `double`, which declares the header argument `:var n=2`, would be written as `#+CALL: double(n=4)`. Note how this function call syntax is different from the header argument syntax.

**<inside header arguments>**

Org passes inside header arguments to the named 'src' code block using the header argument syntax. Inside header arguments apply to code block evaluation. For example, `[:results output]` collects results printed to `STDOUT` during code execution of that block. Note how this header argument syntax is different from the function call syntax.

<sup>1</sup> The option `org-babel-no-eval-on-ctrl-c-ctrl-c` can be used to remove code evaluation from the `C-c C-c` key binding.

<sup>2</sup> Actually, the constructs `call_<name>()` and `src_<lang>{}` are not evaluated when they appear in a keyword line (i.e. lines starting with `#+KEYWORD:`, see Section 15.6 [In-buffer settings], page 231).

**<end header arguments>**

End header arguments affect the results returned by the code block. For example, `:results html` wraps the results in a `BEGIN_EXPORT html` block before inserting the results in the Org buffer.

For more examples of header arguments for `#+CALL:` lines, see [Arguments in function calls], page 208.

## 14.6 Library of Babel

The “Library of Babel” is a collection of code blocks. Like a function library, these code blocks can be called from other Org files. A collection of useful code blocks is available on Worg (<https://orgmode.org/worg/library-of-babel.html>). For remote code block evaluation syntax, see Section 14.5 [Evaluating code blocks], page 204.

For any user to add code to the library, first save the code in regular ‘src’ code blocks of an Org file, and then load the Org file with `org-babel-load-ingest`, which is bound to `C-c C-v i`.

## 14.7 Languages

Org supports the following languages for the ‘src’ code blocks:

| Language   | Identifier | Language       | Identifier |
|------------|------------|----------------|------------|
| Asymptote  | asymptote  | Awk            | awk        |
| C          | C          | C++            | C++        |
| Clojure    | clojure    | CSS            | css        |
| D          | d          | ditaa          | ditaa      |
| Graphviz   | dot        | Emacs Calc     | calc       |
| Emacs Lisp | emacs-lisp | Fortran        | fortran    |
| gnuplot    | gnuplot    | Haskell        | haskell    |
| Java       | java       | Javascript     | js         |
| LaTeX      | latex      | Ledger         | ledger     |
| Lisp       | lisp       | Lilypond       | lilypond   |
| Lua        | lua        | MATLAB         | matlab     |
| Mscgen     | mscgen     | Objective Caml | ocaml      |
| Octave     | octave     | Org mode       | org        |
| Oz         | oz         | Perl           | perl       |
| Plantuml   | plantuml   | Processing.js  | processing |
| Python     | python     | R              | R          |
| Ruby       | ruby       | Sass           | sass       |
| Scheme     | scheme     | GNU Screen     | screen     |
| Sed        | sed        | shell          | sh         |
| SQL        | sql        | SQLite         | sqlite     |
| Vala       | vala       |                |            |

Additional documentation for some languages are at <https://orgmode.org/worg/org-contrib/babel/languages.html>.



By default, only `emacs-lisp` is enabled for evaluation. To enable or disable other languages, customize the `org-babel-load-languages` variable either through the Emacs customization interface, or by adding code to the init file as shown next:

In this example, evaluation is disabled for `emacs-lisp`, and enabled for R.

```
(org-babel-do-load-languages
 'org-babel-load-languages
 '((emacs-lisp . nil)
  (R . t)))
```

Note that this is not the only way to enable a language. Org also enables languages when loaded with `require` statement. For example, the following enables execution of `clojure` code blocks:

```
(require 'ob-clojure)
```

## 14.8 Header arguments

Details of configuring header arguments are shown here.

### 14.8.1 Using header arguments

Since header arguments can be set in several ways, Org prioritizes them in case of overlaps or conflicts by giving local settings a higher priority. Header values in function calls, for example, override header values from global defaults.

#### System-wide header arguments

System-wide values of header arguments can be specified by adapting the `org-babel-default-header-args` variable:

```
:session    => "none"
:results    => "replace"
:exports    => "code"
:cache      => "no"
:noweb      => "no"
```

This example sets `:noweb` header arguments to `yes`, which makes Org expand `:noweb` references by default.

```
(setq org-babel-default-header-args
      (cons '(:noweb . "yes")
            (assq-delete-all :noweb org-babel-default-header-args)))
```

#### Language-specific header arguments

Each language can have separate default header arguments by customizing the variable `org-babel-default-header-args:<lang>`, where `<lang>` is the name of the language. For details, see the language-specific online documentation at <https://orgmode.org/worg/org-contrib/babel/>.

#### Header arguments in Org mode properties

For header arguments applicable to the buffer, use `#+PROPERTY:` lines anywhere in the Org mode file (see Section 7.1 [Property syntax], page 64).

The following example sets only for ‘R’ code blocks to `session`, making all the ‘R’ code blocks execute in the same session. Setting `results` to `silent` ignores the results of executions for all blocks, not just ‘R’ code blocks; no results inserted for any block.

```
#+PROPERTY: header-args:R :session *R*
#+PROPERTY: header-args :results silent
```

Header arguments set through Org’s property drawers (see Section 7.1 [Property syntax], page 64) apply at the sub-tree level on down. Since these property drawers can appear anywhere in the file hierarchy, Org uses outermost call or source block to resolve the values. Org ignores `org-use-property-inheritance` setting.

In this example, `:cache` defaults to `yes` for all code blocks in the sub-tree starting with ‘sample header’.

```
* sample header
:PROPERTIES:
:header-args: :cache yes
:END:
```

Properties defined through `org-set-property` function, bound to `C-c C-x p`, apply to all active languages. They override properties set in `org-babel-default-header-args`.

## Language-specific mode properties

Language-specific header arguments are also read from properties `header-args:<lang>` where `<lang>` is the language identifier. For example,

```
* Heading
:PROPERTIES:
:header-args:clojure: :session *clojure-1*
:header-args:R: :session *R*
:END:

** Subheading
:PROPERTIES:
:header-args:clojure: :session *clojure-2*
:END:
```

would force separate sessions for clojure blocks in Heading and Subheading, but use the same session for all ‘R’ blocks. Blocks in Subheading inherit settings from Heading.

## Code block specific header arguments

Header arguments are most commonly set at the ‘src’ code block level, on the `#+BEGIN_SRC` line. Arguments set at this level take precedence over those set in the `org-babel-default-header-args` variable, and also those set as header properties.

In the following example, setting `results` to `silent` makes it ignore results of the code execution. Setting `:exports` to `code` exports only the body of the ‘src’ code block to HTML or  $\text{\LaTeX}$ :

```
#+NAME: factorial
#+BEGIN_SRC haskell :results silent :exports code :var n=0
fac 0 = 1
fac n = n * fac (n-1)
#+END_SRC
```

The same header arguments in an inline ‘src’ code block:

```
src_haskell[:exports both]{fac 5}
```

Code block header arguments can span multiple lines using `#+HEADER:` on each line. Note that Org currently accepts the plural spelling of `#+HEADER:` only as a convenience for backward-compatibility. It may be removed at some point.

Multi-line header arguments on an unnamed ‘src’ code block:

```
#+HEADER: :var data1=1
#+BEGIN_SRC emacs-lisp :var data2=2
  (message "data1:%S, data2:%S" data1 data2)
#+END_SRC

#+RESULTS:
: data1:1, data2:2
```

Multi-line header arguments on a named ‘src’ code block:

```
#+NAME: named-block
#+HEADER: :var data=2
#+BEGIN_SRC emacs-lisp
  (message "data:%S" data)
#+END_SRC

#+RESULTS: named-block
: data:2
```

## Arguments in function calls

Header arguments in function calls are the most specific and override all other settings in case of an overlap. They get the highest priority. Two `#+CALL:` examples are shown below. For the complete syntax of `#+CALL:` lines, see Section 14.5 [Evaluating code blocks], page 204.

In this example, `:exports results` header argument is applied to the evaluation of the `#+CALL:` line.

```
#+CALL: factorial(n=5) :exports results
```

In this example, `:session special` header argument is applied to the evaluation of `factorial` code block.

```
#+CALL: factorial[:session special](n=5)
```

### 14.8.2 Specific header arguments

Org comes with many header arguments common to all languages. New header arguments are added for specific languages as they become available for use in ‘src’ code blocks. A header argument is specified with an initial colon followed by the argument’s name in lowercase. Common header arguments are:

For language-specific header arguments, see Section 14.7 [Languages], page 205.

### 14.8.2.1 `:var`

Use `:var` for passing arguments to ‘`src`’ code blocks. The specifics of variables in ‘`src`’ code blocks vary by the source language and are covered in the language-specific documentation. The syntax for `:var`, however, is the same for all languages. This includes declaring a variable, and assigning a default value.

Arguments can take values as literals, or as references, or even as Emacs Lisp code (see Section 14.8.2.1 [var], page 209). References are names from the Org file from the lines `#+NAME:` or `#+RESULTS:`. References can also refer to tables, lists, `#+BEGIN_EXAMPLE` blocks, other types of ‘`src`’ code blocks, or the results of execution of ‘`src`’ code blocks.

For better performance, Org can cache results of evaluations. But caching comes with severe limitations (see Section 14.8.2.18 [cache], page 219).

Argument values are indexed like arrays (see Section 14.8.2.1 [var], page 209).

The following syntax is used to pass arguments to ‘`src`’ code blocks using the `:var` header argument.

```
:var name=assign
```

The `assign` is a literal value, such as a string “`string`”, a number ‘9’, a reference to a table, a list, a literal example, another code block (with or without arguments), or the results from evaluating a code block.

Here are examples of passing values by reference:

*table*        an Org mode table named with either a `#+NAME:` line

```
#+NAME: example-table
| 1 |
| 2 |
| 3 |
| 4 |

#+NAME: table-length
#+BEGIN_SRC emacs-lisp :var table=example-table
(length table)
#+END_SRC

#+RESULTS: table-length
: 4
```

*list*        a simple list named with a `#+NAME:` line. Note that only the top level list items are passed along. Nested list items are ignored.

```
#+NAME: example-list
- simple
- not
- nested
- list

#+BEGIN_SRC emacs-lisp :var x=example-list
(print x)
#+END_SRC
```

```
#+RESULTS:
| simple | list |
```

*code block without arguments*

a code block name (from the example above), as assigned by `#+NAME:`, optionally followed by parentheses

```
#+BEGIN_SRC emacs-lisp :var length=table-length()
(* 2 length)
#+END_SRC
```

```
#+RESULTS:
: 8
```

*code block with arguments*

a ‘src’ code block name, as assigned by `#+NAME:`, followed by parentheses and optional arguments passed within the parentheses following the ‘src’ code block name using standard function call syntax

```
#+NAME: double
#+BEGIN_SRC emacs-lisp :var input=8
(* 2 input)
#+END_SRC
```

```
#+RESULTS: double
: 16
```

```
#+NAME: squared
#+BEGIN_SRC emacs-lisp :var input=double(input=2)
(* input input)
#+END_SRC
```

```
#+RESULTS: squared
: 4
```

*literal example*

a literal example block named with a `#+NAME:` line

```
#+NAME: literal-example
#+BEGIN_EXAMPLE
A literal example
on two lines
#+END_EXAMPLE
```

```
#+NAME: read-literal-example
#+BEGIN_SRC emacs-lisp :var x=literal-example
  (concatenate 'string x " for you.")
#+END_SRC
```

```
#+RESULTS: read-literal-example
```

```
: A literal example
: on two lines for you.
```

### インデックスを用いた変数の指定

Indexing variable values enables referencing portions of a variable. Indexes are 0 based with negative values counting backwards from the end. If an index is separated by ,s then each subsequent section will index as the next dimension. Note that this indexing occurs *before* other table-related header arguments are applied, such as `:hlines`, `:colnames` and `:rownames`. The following example assigns the last cell of the first row the table `example-table` to the variable `data`:

```
#+NAME: example-table
| 1 | a |
| 2 | b |
| 3 | c |
| 4 | d |

#+BEGIN_SRC emacs-lisp :var data=example-table[0,-1]
data
#+END_SRC

#+RESULTS:
: a
```

:で区切られた2つの整数を使うことで、変数の範囲を参照できます。この場合、包括的な範囲を参照します。例えば、以下では `example-table` の中央3行を `data` に割り当てています。

```
#+NAME: example-table
| 1 | a |
| 2 | b |
| 3 | c |
| 4 | d |
| 5 | 3 |

#+BEGIN_SRC emacs-lisp :var data=example-table[1:3]
data
#+END_SRC

#+RESULTS:
| 2 | b |
| 3 | c |
| 4 | d |
```

To pick the entire range, use an empty index, or the single character `*`. `0:-1` does the same thing. Example below shows how to reference the first column only.

```
#+NAME: example-table
| 1 | a |
| 2 | b |
| 3 | c |
```

```
| 4 | d |

#+BEGIN_SRC emacs-lisp :var data=example-table[,0]
  data
#+END_SRC

#+RESULTS:
| 1 | 2 | 3 | 4 |
```

Index referencing can be used for tables and code blocks. Index referencing can handle any number of dimensions. Commas delimit multiple dimensions, as shown below.

```
#+NAME: 3D
#+BEGIN_SRC emacs-lisp
  '(((1 2 3) (4 5 6) (7 8 9))
    ((10 11 12) (13 14 15) (16 17 18))
    ((19 20 21) (22 23 24) (25 26 27)))
#+END_SRC

#+BEGIN_SRC emacs-lisp :var data=3D[1,,1]
  data
#+END_SRC

#+RESULTS:
| 11 | 14 | 17 |
```

## Emacs Lisp による変数の評価

Emacs lisp code can set the values for variables. To differentiate a value from lisp code, Org interprets any value starting with (, [, ' or ` as Emacs Lisp code. The result of evaluating that code is then assigned to the value of that variable. The following example shows how to reliably query and pass file name of the Org mode buffer to a code block using headers. We need reliability here because the file's name could change once the code in the block starts executing.

```
#+BEGIN_SRC sh :var filename=(buffer-file-name) :exports both
  wc -w $filename
#+END_SRC
```

Note that values read from tables and lists will not be mistakenly evaluated as Emacs Lisp code, as illustrated in the following example.

```
#+NAME: table
| (a b c) |

#+HEADER: :var data=table[0,0]
#+BEGIN_SRC perl
  $data
#+END_SRC

#+RESULTS:
: (a b c)
```

### 14.8.2.2 :results

There are four classes of `:results` header arguments. Each ‘src’ code block can take only one option per class.

- **collection** for how the results should be collected from the ‘src’ code block
- **type** for which type of result the code block will return; affects how Org processes and inserts results in the Org buffer
- **format** for the result; affects how Org processes and inserts results in the Org buffer
- **handling** for processing results after evaluation of the ‘src’ code block

#### Collection

Collection options specify the results. Choose one of the options; they are mutually exclusive.

- **value** Default. Functional mode. Result is the value returned by the last statement in the ‘src’ code block. Languages like Python may require an explicit `return` statement in the ‘src’ code block. Usage example: `:results value`.
- **output** Scripting mode. Result is collected from STDOUT during execution of the code in the ‘src’ code block. Usage example: `:results output`.

#### Type

Type tells what result types to expect from the execution of the code block. Choose one of the options; they are mutually exclusive. The default behavior is to automatically determine the result type.

- **table**, **vector** Interpret the results as an Org table. If the result is a single value, create a table with one row and one column. Usage example: `:results value table`.
- **list** Interpret the results as an Org list. If the result is a single value, create a list of one element.
- **scalar**, **verbatim** Interpret literally and insert as quoted text. Do not create a table. Usage example: `:results value verbatim`.
- **file** Interpret as path to a file. Inserts a link to the file. Usage example: `:results value file`.

#### Format

Format pertains to the type of the result returned by the ‘src’ code block. Choose one of the options; they are mutually exclusive. The default follows from the type specified above.

- **raw** Interpreted as raw Org mode. Inserted directly into the buffer. Aligned if it is a table. Usage example: `:results value raw`.
- **org** Results enclosed in a `BEGIN_SRC org` block. For comma-escape, either `TAB` in the block, or export the file. Usage example: `:results value org`.
- **html** Results enclosed in a `BEGIN_EXPORT html` block. Usage example: `:results value html`.
- **latex** Results enclosed in a `BEGIN_EXPORT latex` block. Usage example: `:results value latex`.



- **code** Result enclosed in a ‘**src**’ code block. Useful for parsing. Usage example: `:results value code`.
- **pp** Result converted to pretty-print source code. Enclosed in a ‘**src**’ code block. Languages supported: Emacs Lisp, Python, and Ruby. Usage example: `:results value pp`.
- **drawer** Result wrapped in a RESULTS drawer. Useful for containing **raw** or **org** results for later scripting and automated processing. Usage example: `:results value drawer`.

## Handling

Handling options after collecting the results.

- **silent** Do not insert results in the Org mode buffer, but echo them in the minibuffer. Usage example: `:results output silent`.
- **replace** Default. Insert results in the Org buffer. Remove previous results. Usage example: `:results output replace`.
- **append** Append results to the Org buffer. Latest results are at the bottom. Does not remove previous results. Usage example: `:results output append`.
- **prepend** Prepend results to the Org buffer. Latest results are at the top. Does not remove previous results. Usage example: `:results output prepend`.

### 14.8.2.3 :file

An external **:file** that saves the results of execution of the code block. The **:file** is either a file name or two strings, where the first is the file name and the second is the description. A link to the file is inserted. It uses an Org mode style `[[file:]]` link (see Section 4.1 [Link format], page 38). Some languages, such as ‘R’, ‘dot’, ‘ditaa’, and ‘gnuplot’, automatically wrap the source code in additional boilerplate code. Such code wrapping helps recreate the output, especially graphics output, by executing just the **:file** contents.

### 14.8.2.4 :file-desc

A description of the results file. Org uses this description for the link (see Section 4.1 [Link format], page 38) it inserts in the Org file. If the **:file-desc** has no value, Org will use file name for both the “link” and the “description” portion of the Org mode link.

### 14.8.2.5 :file-ext

File name extension for the output file. Org generates the file’s complete name, and extension by combining **:file-ext**, **#+NAME:** of the source block, and the Section 14.8.2.6 [output-dir], page 214 header argument. To override this auto generated file name, use the **:file** header argument.

### 14.8.2.6 :output-dir

Specifies the **:output-dir** for the results file. Org accepts an absolute path (beginning with /) or a relative directory (without /). The value can be combined with **#+NAME:** of the source block and Section 14.8.2.3 [file], page 214 or Section 14.8.2.5 [file-ext], page 214 header arguments.

### 14.8.2.7 :dirとリモートでの実行

While the `:file` header argument can be used to specify the path to the output file, `:dir` specifies the default directory during ‘`src`’ code block execution. If it is absent, then the directory associated with the current buffer is used. In other words, supplying `:dir path` temporarily has the same effect as changing the current directory with `M-x cd path RET`, and then not supplying `:dir`. Under the surface, `:dir` simply sets the value of the Emacs variable `default-directory`.

When using `:dir`, relative paths (for example, `:file myfile.jpg` or `:file results/myfile.jpg`) become relative to the default directory.

For example, to save the plot file in the ‘`Work`’ folder of the home directory (notice tilde is expanded):

```
#+BEGIN_SRC R :file myplot.png :dir ~/Work
matplot(matrix(rnorm(100), 10), type="l")
#+END_SRC
```

リモートでの実行

To evaluate the ‘`src`’ code block on a remote machine, supply a remote `s` directory name using ‘`Tramp`’ syntax. For example:

```
#+BEGIN_SRC R :file plot.png :dir /scp:dand@yakuba.princeton.edu:
plot(1:10, main=system("hostname", intern=TRUE))
#+END_SRC
```

Org first captures the text results as usual for insertion in the Org file. Then Org also inserts a link to the remote file, thanks to Emacs ‘`Tramp`’. Org constructs the remote path to the file name from `:dir` and `default-directory`, as illustrated here:

```
[[file:/scp:dand@yakuba.princeton.edu:/home/dand/plot.png][plot.png]]
```

### Some more warnings

- When `:dir` is used with `:session`, Org sets the starting directory for a new session. But Org will not alter the directory of an already existing session.
- Do not use `:dir` with `:exports results` or with `:exports both` to avoid Org inserting incorrect links to remote files. That is because Org does not expand `default-directory` to avoid some underlying portability issues.

### 14.8.2.8 :exports

The `:exports` header argument is to specify if that part of the Org file is exported to, say, HTML or  $\text{\LaTeX}$  formats. Note that `:exports` affects only ‘`src`’ code blocks and not inline code.

- `code` The default. The body of code is included into the exported file. Example: `:exports code`.
- `results` The results of evaluation of the code is included in the exported file. Example: `:exports results`.
- `both` Both the code and results of evaluation are included in the exported file. Example: `:exports both`.

- **none** Neither the code nor the results of evaluation is included in the exported file. Whether the code is evaluated at all depends on other options. Example: `:exports none`.

#### 14.8.2.9 `:tangle`

The `:tangle` header argument specifies if the ‘`src`’ code block is exported to source file(s).

- **tangle** Export the ‘`src`’ code block to source file. The file name for the source file is derived from the name of the Org file, and the file extension is derived from the source code language identifier. Example: `:tangle yes`.
- **no** The default. Do not extract the code a source code file. Example: `:tangle no`.
- **other** Export the ‘`src`’ code block to source file whose file name is derived from any string passed to the `:tangle` header argument. Org derives the file name as being relative to the directory of the Org file’s location. Example: `:tangle path`.

#### 14.8.2.10 `:mkdirp`

The `:mkdirp` header argument creates parent directories for tangled files if the directory does not exist. **yes** enables directory creation and **no** inhibits directory creation.

#### 14.8.2.11 `:comments`

Controls inserting comments into tangled files. These are above and beyond whatever comments may already exist in the ‘`src`’ code block.

- **no** The default. Do not insert any extra comments during tangling.
- **link** Wrap the ‘`src`’ code block in comments. Include links pointing back to the place in the Org file from where the code was tangled.
- **yes** Kept for backward compatibility; same as “link”.
- **org** Nearest headline text from Org file is inserted as comment. The exact text that is inserted is picked from the leading context of the source block.
- **both** Includes both “link” and “org” comment options.
- **noweb** Includes “link” comment option, expands noweb references, and wraps them in link comments inside the body of the ‘`src`’ code block.

#### 14.8.2.12 `:padline`

Control insertion of newlines to pad ‘`src`’ code blocks in the tangled file.

- **yes** Default. Insert a newline before and after each ‘`src`’ code block in the tangled file.
- **no** Do not insert newlines to pad the tangled ‘`src`’ code blocks.

#### 14.8.2.13 `:no-expand`

By default Org expands ‘`src`’ code blocks during tangling. The `:no-expand` header argument turns off such expansions. Note that one side-effect of expansion by `org-babel-expand-src-block` also assigns values to `:var` (see Section 14.8.2.1 [var], page 209) variables. Expansions also replace Noweb references with their targets (see Section 14.10 [Noweb reference syntax], page 226). Some of these expansions may cause premature assignment, hence this option. This option makes a difference only for tangling. It has no effect when exporting since ‘`src`’ code blocks for execution have to be expanded anyway.

#### 14.8.2.14 :session

The `:session` header argument is for running multiple source code blocks under one session. Org runs ‘src’ code blocks with the same session name in the same interpreter process.

- **none** Default. Each ‘src’ code block gets a new interpreter process to execute. The process terminates once the block is evaluated.
- **other** Any string besides **none** turns that string into the name of that session. For example, `:session mysession` names it ‘mysession’. If `:session` has no argument, then the session name is derived from the source language identifier. Subsequent blocks with the same source code language use the same session. Depending on the language, state variables, code from other blocks, and the overall interpreted environment may be shared. Some interpreted languages support concurrent sessions when subsequent source code language blocks change session names.

#### 14.8.2.15 :noweb

The `:noweb` header argument controls expansion of Noweb syntax references (see Section 14.10 [Noweb reference syntax], page 226). Expansions occur when source code blocks are evaluated, tangled, or exported.

- **no** Default. No expansion of Noweb syntax references in the body of the code when evaluating, tangling, or exporting.
- **yes** Expansion of Noweb syntax references in the body of the ‘src’ code block when evaluating, tangling, or exporting.
- **tangle** Expansion of Noweb syntax references in the body of the ‘src’ code block when tangling. No expansion when evaluating or exporting.
- **no-export** Expansion of Noweb syntax references in the body of the ‘src’ code block when evaluating or tangling. No expansion when exporting.
- **strip-export** Expansion of Noweb syntax references in the body of the ‘src’ code block when expanding prior to evaluating or tangling. Removes Noweb syntax references when exporting.
- **eval** Expansion of Noweb syntax references in the body of the ‘src’ code block only before evaluating.

#### Noweb の行頭部分

Noweb insertions now honor prefix characters that appear before the Noweb syntax reference.

This behavior is illustrated in the following example. Because the `<<example>>` noweb reference appears behind the SQL comment syntax, each line of the expanded noweb reference will be commented.

With:

```
#+NAME: example
#+BEGIN_SRC text
this is the
multi-line body of example
#+END_SRC
```

this ‘src’ code block:

```
#+BEGIN_SRC sql :noweb yes
-- <<example>>
#+END_SRC
```

次のように展開されます:

```
-- これは
-- example の複数行の本文です
```

Since this change will not affect noweb replacement text without newlines in them, inline noweb references are acceptable.

This feature can also be used for management of indentation in exported code snippets.

With:

```
#+NAME: if-true
#+BEGIN_SRC python :exports none
print('Do things when True')
#+END_SRC
```

```
#+NAME: if-false
#+BEGIN_SRC python :exports none
print('Do things when False')
#+END_SRC
```

this ‘src’ code block:

```
#+BEGIN_SRC python :noweb yes :results output
if True:
    <<if-true>>
else:
    <<if-false>>
#+END_SRC
```

次のように展開されます:

```
if True:
    print('Do things when True')
else:
    print('Do things when False')
```

and evaluates to:

```
Do things when True
```

#### 14.8.2.16 :noweb-ref

When expanding Noweb style references, Org concatenates ‘src’ code blocks by matching the reference name to either the code block name or the `:noweb-ref` header argument.

For simple concatenation, set this `:noweb-ref` header argument at the sub-tree or file level. In the example Org file shown next, the body of the source code in each block is extracted for concatenation to a pure code file when tangled.

```
#+BEGIN_SRC sh :tangle yes :noweb yes :shebang #!/bin/sh
<<fullest-disk>>
#+END_SRC
* the mount point of the fullest disk
```

```

:PROPERTIES:
:header-args: :noweb-ref fullest-disk
:END:

** query all mounted disks
#+BEGIN_SRC sh
df \
#+END_SRC

** strip the header row
#+BEGIN_SRC sh
|sed '1d' \
#+END_SRC

** output mount point of fullest disk
#+BEGIN_SRC sh
|awk '{if (u < +$5) {u = +$5; m = $6}} END {print m}'
#+END_SRC

```

#### 14.8.2.17 :noweb-sep

By default a newline separates each noweb reference concatenation. To change this newline separator, edit the `:noweb-sep` (see Section 14.8.2.17 [noweb-sep], page 219) header argument.

#### 14.8.2.18 :cache

The `:cache` header argument is for caching results of evaluating code blocks. Caching results can avoid re-evaluating ‘src’ code blocks that have not changed since the previous run. To benefit from the cache and avoid redundant evaluations, the source block must have a result already present in the buffer, and neither the header arguments (including the value of `:var` references) nor the text of the block itself has changed since the result was last computed. This feature greatly helps avoid long-running calculations. For some edge cases, however, the cached results may not be reliable.

The caching feature is best for when ‘src’ blocks are pure functions, that is functions that return the same value for the same input arguments (see Section 14.8.2.1 [var], page 209), and that do not have side effects, and do not rely on external variables other than the input arguments. Functions that depend on a timer, file system objects, and random number generators are clearly unsuitable for caching.

A note of warning: when `:cache` is used for a `:session`, caching may cause unexpected results.

When the caching mechanism tests for any source code changes, it will not expand Noweb style references (see Section 14.10 [Noweb reference syntax], page 226). For reasons why, see <http://thread.gmane.org/gmane.emacs.orgmode/79046>.

The `:cache` header argument can have one of two values: **yes** or **no**.

- **no** Default. No caching of results; ‘src’ code block evaluated every time.
- **yes** Whether to run the code or return the cached results is determined by comparing the SHA1 hash value of the combined ‘src’ code block and arguments passed to it.

This hash value is packed on the `#+RESULTS:` line from previous evaluation. When hash values match, Org does not evaluate the `'src'` code block. When hash values mismatch, Org evaluates the `'src'` code block, inserts the results, recalculates the hash value, and updates `#+RESULTS:` line.

In this example, both functions are cached. But `caller` runs only if the result from `random` has changed since the last run.

```
#+NAME: random
#+BEGIN_SRC R :cache yes
runif(1)
#+END_SRC

#+RESULTS[a2a72cd647ad44515fab62e144796432793d68e1]: random
0.4659510825295

#+NAME: caller
#+BEGIN_SRC emacs-lisp :var x=random :cache yes
x
#+END_SRC

#+RESULTS[bec9c8724e397d5df3b696502df3ed7892fc4f5f]: caller
0.254227238707244
```

#### 14.8.2.19 :sep

The `:sep` header argument is the delimiter for saving results as tables to files (see Section 14.8.2.3 [file], page 214) external to Org mode. Org defaults to tab delimited output. The function, `org-open-at-point`, which is bound to `C-c C-o`, also uses `:sep` for opening tabular results.

#### 14.8.2.20 :hlines

In-between each table row or below the table headings, sometimes results have horizontal lines, which are also known as hlines. The `:hlines` argument with the value `yes` accepts such lines. The default is `no`.

- `no` Strips horizontal lines from the input table. For most code, this is desirable, or else those `hline` symbols raise unbound variable errors.

The default is `:hlines no`. The example shows hlines removed from the input table.

```
#+NAME: many-cols
| a | b | c |
|---+---+---|
| d | e | f |
|---+---+---|
| g | h | i |

#+NAME: echo-table
#+BEGIN_SRC python :var tab=many-cols
return tab
```

```
#+END_SRC
```

```
#+RESULTS: echo-table
```

```
| a | b | c |
| d | e | f |
| g | h | i |
```

- **yes** For `:hlines` **yes**, the example shows `hlines` unchanged.

```
#+NAME: many-cols
```

```
| a | b | c |
|---+---+---|
| d | e | f |
|---+---+---|
| g | h | i |
```

```
#+NAME: echo-table
```

```
#+BEGIN_SRC python :var tab=many-cols :hlines yes
```

```
    return tab
```

```
#+END_SRC
```

```
#+RESULTS: echo-table
```

```
| a | b | c |
|---+---+---|
| d | e | f |
|---+---+---|
| g | h | i |
```

#### 14.8.2.21 `:colnames`

The `:colnames` header argument accepts **yes**, **no**, or **nil** values. The default value is **nil**, which is unassigned. But this header argument behaves differently depending on the source code language.

- **nil** If an input table has column names (because the second row is an `hline`), then Org removes the column names, processes the table, puts back the column names, and then writes the table to the results block.

```
#+NAME: less-cols
```

```
| a |
|---|
| b |
| c |
```

```
#+NAME: echo-table-again
```

```
#+BEGIN_SRC python :var tab=less-cols
```

```
    return [[val + '*' for val in row] for row in tab]
```

```
#+END_SRC
```

```
#+RESULTS: echo-table-again
```

```
| a |
```



```
|---|
| b* |
| c* |
```

Note that column names have to be accounted for when using variable indexing (see Section 14.8.2.1 [var], page 209) because column names are not removed for indexing.

- **no** Do not pre-process column names.
- **yes** For an input table that has no hlines, process it like the `nil` value. That is, Org removes the column names, processes the table, puts back the column names, and then writes the table to the results block.

#### 14.8.2.22 :rownames

The `:rownames` header argument can take on values `yes` or `no` values. The default is `no`. Note that `emacs-lisp` code blocks ignore `:rownames` header argument because of the ease of table-handling in Emacs.

- **no** Org will not pre-process row names.
- **yes** If an input table has row names, then Org removes the row names, processes the table, puts back the row names, and then writes the table to the results block.

```
#+NAME: with-rownames
| one | 1 | 2 | 3 | 4 | 5 |
| two | 6 | 7 | 8 | 9 | 10 |

#+NAME: echo-table-once-again
#+BEGIN_SRC python :var tab=with-rownames :rownames yes
    return [[val + 10 for val in row] for row in tab]
#+END_SRC

#+RESULTS: echo-table-once-again
| one | 11 | 12 | 13 | 14 | 15 |
| two | 16 | 17 | 18 | 19 | 20 |
```

Note that row names have to be accounted for when using variable indexing (see Section 14.8.2.1 [var], page 209) because row names are not removed for indexing.

#### 14.8.2.23 :shebang

This header argument can turn results into executable script files. By setting the `:shebang` header argument to a string value (for example, `:shebang "#!/bin/bash"`), Org inserts that string as the first line of the tangled file that the `'src'` code block is extracted to. Org then turns on the tangled file's executable permission.

#### 14.8.2.24 :tangle-mode

The `tangle-mode` header argument specifies what permissions to set for tangled files by `set-file-modes`. For example, to make read-only tangled file, use `:tangle-mode (identity #o444)`. To make it executable, use `:tangle-mode (identity #o755)`.

On `'src'` code blocks with `shebang` (see Section 14.8.2.23 [shebang], page 222) header argument, Org will automatically set the tangled file to executable permissions. But this can be overridden with custom permissions using `tangle-mode` header argument.

When multiple ‘src’ code blocks tangle to a single file with different and conflicting `tangle-mode` header arguments, Org’s behavior is undefined.

#### 14.8.2.25 :eval

The `:eval` header argument can limit evaluation of specific code blocks. It is useful for protection against evaluating untrusted ‘src’ code blocks by prompting for a confirmation. This protection is independent of the `org-confirm-babel-evaluate` setting.

**never or no**

Org will never evaluate this ‘src’ code block.

**query**      Org prompts the user for permission to evaluate this ‘src’ code block.

**never-export or no-export**

Org will not evaluate this ‘src’ code block when exporting, yet the user can evaluate this source block interactively.

**query-export**

Org prompts the user for permission to export this ‘src’ code block.

If `:eval` header argument is not set for a source block, then Org determines whether to evaluate from the `org-confirm-babel-evaluate` variable (see Section 15.4 [Code evaluation security], page 230).

#### 14.8.2.26 :wrap

The `:wrap` header argument marks the results block by appending strings to `#+BEGIN_` and `#+END_`. If no string is specified, Org wraps the results in a `#+BEGIN/END_RESULTS` block.

#### 14.8.2.27 :post

The `:post` header argument is for post-processing results from ‘src’ block evaluation. When `:post` has any value, Org binds the results to `*this*` variable for easy passing to Section 14.8.2.1 [var], page 209 header argument specifications. That makes results available to other ‘src’ code blocks, or for even direct Emacs Lisp code execution.

The following two examples illustrate `:post` header argument in action. The first one shows how to attach `#+ATTR_LATEX:` line using `:post`.

```
#+name: attr_wrap
#+begin_src sh :var data="" :var width="\textwidth" :results output
  echo "#+ATTR_LATEX: :width $width"
  echo "$data"
#+end_src

#+header: :file /tmp/it.png
#+begin_src dot :post attr_wrap(width="5cm", data=*this*) :results drawer
  digraph{
    a -> b;
    b -> c;
    c -> a;
  }
#+end_src
```

```
#+RESULTS:
:RESULTS:
#+ATTR_LATEX :width 5cm
[[file:/tmp/it.png]]
:END:
```

The second example shows use of `:colnames` in `:post` to pass data between ‘src’ code blocks.

```
#+name: round-tbl
#+begin_src emacs-lisp :var tbl="" fmt="%.3f"
  (mapcar (lambda (row)
            (mapcar (lambda (cell)
                      (if (numberp cell)
                          (format fmt cell)
                          cell))
                    row))
          tbl)
#+end_src

#+begin_src R :colnames yes :post round-tbl[:colnames yes](*this*)
set.seed(42)
data.frame(foo=rnorm(1))
#+end_src

#+RESULTS:
|   foo |
|-----|
| 1.371 |
```

### 14.8.2.28 :prologue

The `prologue` header argument is for appending to the top of the code block for execution. For example, a clear or reset code at the start of new execution of a ‘src’ code block. A reset for ‘gnuplot’: `:prologue "reset"`. See also Section 14.8.2.29 [epilogue], page 224.

```
(add-to-list 'org-babel-default-header-args:gnuplot
  '(:prologue . "reset"))
```

### 14.8.2.29 :epilogue

The value of the `epilogue` header argument is for appending to the end of the code block for execution. See also Section 14.8.2.28 [prologue], page 224.

## 14.9 Results of evaluation

How Org handles results of a code block execution depends on many header arguments working together. Here is only a summary of these. For an enumeration of all the header arguments that affect results, see Section 14.8.2.2 [results], page 213.

The primary determinant is the execution context. Is it in a `:session` or not? Orthogonal to that is if the expected result is a `:results value` or `:results output`, which is a concatenation of output from start to finish of the ‘`src`’ code block’s evaluation.

|                              | Non-session | Session          |
|------------------------------|-------------|------------------|
| <code>:results value</code>  | 最後の式の値      | 最後の式の値           |
| <code>:results output</code> | STDOUT の内容  | インタプリタの出力をつなげたもの |

For `:session` and non-session, the `:results value` turns the results into an Org mode table format. Single values are wrapped in a one dimensional vector. Rows and columns of a table are wrapped in a two-dimensional vector.

## 14.9.1 Non-session

### 14.9.1.1 `:results value`

Default. Org gets the value by wrapping the code in a function definition in the language of the ‘`src`’ block. That is why when using `:results value`, code should execute like a function and return a value. For languages like Python, an explicit `return` statement is mandatory when using `:results value`.

This is one of four evaluation contexts where Org automatically wraps the code in a function definition.

### 14.9.1.2 `:results output`

For `:results output`, the code is passed to an external process running the interpreter. Org returns the contents of the standard output stream as text results.

## 14.9.2 Session

### 14.9.2.1 `:results value`

For `:results value` from a `:session`, Org passes the code to an interpreter running as an interactive Emacs inferior process. So only languages that provide interactive evaluation can have session support. Not all languages provide this support, such as ‘`C`’ and ‘`ditaa`’. Even those that do support, such as ‘`Python`’ and ‘`Haskell`’, they impose limitations on allowable language constructs that can run interactively. Org inherits those limitations for those ‘`src`’ code blocks running in a `:session`.

Org gets the value from the source code interpreter’s last statement output. Org has to use language-specific methods to obtain the value. For example, from the variable `_` in ‘`Python`’ and ‘`Ruby`’, and the value of `.Last.value` in ‘`R`’).

### 14.9.2.2 `:results output`

For `:results output`, Org passes the code to the interpreter running as an interactive Emacs inferior process. Org concatenates whatever text output emitted by the interpreter to return the collection as a result. Note that this collection is not the same as collected from `STDOUT` of a non-interactive interpreter running as an external process. Compare for example these two blocks:

```
#+BEGIN_SRC python :results output
print "hello"
```

```

2
print "bye"
#+END_SRC

```

```

#+RESULTS:
: hello
: bye

```

In the above non-session mode, the “2” is not printed; so does not appear in results.

```

#+BEGIN_SRC python :results output :session
print "hello"
2
print "bye"
#+END_SRC

```

```

#+RESULTS:
: hello
: 2
: bye

```

In the above `:session` mode, the interactive interpreter receives and prints “2”. Results show that.

## 14.10 Noweb reference syntax

Org supports named blocks in Noweb style syntax. For Noweb literate programming details, see <http://www.cs.tufts.edu/~nr/noweb/>.

```
<<code-block-name>>
```

For the header argument `:noweb yes`, Org expands Noweb style references in the ‘`src`’ code block before evaluation.

For the header argument `:noweb no`, Org does not expand Noweb style references in the ‘`src`’ code block before evaluation.

The default is `:noweb no`. Org defaults to `:noweb no` so as not to cause errors in languages where Noweb syntax is ambiguous. Change Org’s default to `:noweb yes` for languages where there is no risk of confusion.

Org offers a more flexible way to resolve Noweb style references (see Section 14.8.2.16 [noweb-ref], page 218).

Org can include the *results* of a code block rather than its body. To that effect, append parentheses, possibly including arguments, to the code block name, as show below.

```
<<code-block-name(optional arguments)>>
```

Note that when using the above approach to a code block’s results, the code block name set by `#+NAME` keyword is required; the reference set by `:noweb-ref` will not work.

Here is an example that demonstrates how the exported content changes when Noweb style references are used with parentheses versus without.

With:

```
#+NAME: some-code
#+BEGIN_SRC python :var num=0 :results output :exports none
print(num*10)
#+END_SRC
```

this code block:

```
#+BEGIN_SRC text :noweb yes
<<some-code>>
#+END_SRC
```

次のように展開されます:

```
print(num*10)
```

Below, a similar Noweb style reference is used, but with parentheses, while setting a variable `num` to 10:

```
#+BEGIN_SRC text :noweb yes
<<some-code(num=10)>>
#+END_SRC
```

Note that now the expansion contains the *results* of the code block `some-code`, not the code block itself:

```
100
```

For faster tangling of large Org mode files, set `org-babel-use-quick-and-dirty-noweb-expansion` variable to `t`. The speedup comes at the expense of not correctly resolving inherited values of the `:noweb-ref` header argument.

## 14.11 Key bindings and useful functions

Many common Org mode key sequences are re-bound depending on the context.

Active key bindings in code blocks:

|                      |  |
|----------------------|--|
| <code>C-c C-c</code> | <code>org-babel-execute-src-block</code>     |
| <code>C-c C-o</code> | <code>org-babel-open-src-block-result</code> |
| <code>M-UP</code>    | <code>org-babel-load-in-session</code>       |
| <code>M-DOWN</code>  | <code>org-babel-switch-to-session</code>     |

Active key bindings in Org mode buffer:

|                        |      |                          |  |
|------------------------|------|--------------------------|--|
| <code>C-c C-v p</code> | or   | <code>C-c C-v C-p</code> | <code>org-babel-previous-src-block</code>    |
| <code>C-c C-v n</code> | or   | <code>C-c C-v C-n</code> | <code>org-babel-next-src-block</code>        |
| <code>C-c C-v e</code> | or   | <code>C-c C-v C-e</code> | <code>org-babel-execute-maybe</code>         |
| <code>C-c C-v o</code> | or   | <code>C-c C-v C-o</code> | <code>org-babel-open-src-block-result</code> |
| <code>C-c C-v v</code> | or   | <code>C-c C-v C-v</code> | <code>org-babel-expand-src-block</code>      |
| <code>C-c C-v u</code> | or   | <code>C-c C-v C-u</code> | <code>org-babel-goto-src-block-head</code>   |
| <code>C-c C-v g</code> | or   | <code>C-c C-v C-g</code> | <code>org-babel-goto-named-src-block</code>  |
| <code>C-c C-v r</code> | or   | <code>C-c C-v C-r</code> | <code>org-babel-goto-named-result</code>     |
| <code>C-c C-v b</code> | もしくは | <code>C-c C-v C-b</code> | <code>org-babel-execute-buffer</code>        |
| <code>C-c C-v s</code> | もしくは | <code>C-c C-v C-s</code> | <code>org-babel-execute-subtree</code>       |
| <code>C-c C-v d</code> | or   | <code>C-c C-v C-d</code> | <code>org-babel-demarcate-block</code>       |

|                  |      |                    |   |
|------------------|------|--------------------|---|
| <i>C-c C-v t</i> | もしくは | <i>C-c C-v C-t</i> | <code>org-babel-tangle</code>                         |
| <i>C-c C-v f</i> | もしくは | <i>C-c C-v C-f</i> | <code>org-babel-tangle-file</code>                    |
| <i>C-c C-v c</i> | or   | <i>C-c C-v C-c</i> | <code>org-babel-check-src-block</code>                |
| <i>C-c C-v j</i> | or   | <i>C-c C-v C-j</i> | <code>org-babel-insert-header-arg</code>              |
| <i>C-c C-v l</i> | or   | <i>C-c C-v C-l</i> | <code>org-babel-load-in-session</code>                |
| <i>C-c C-v i</i> | or   | <i>C-c C-v C-i</i> | <code>org-babel-lob-ingest</code>                     |
| <i>C-c C-v I</i> | or   | <i>C-c C-v C-I</i> | <code>org-babel-view-src-block-info</code>            |
| <i>C-c C-v z</i> | or   | <i>C-c C-v C-z</i> | <code>org-babel-switch-to-session-with-code</code>    |
| <i>C-c C-v a</i> | もしくは | <i>C-c C-v C-a</i> | <code>org-babel-sha1-hash</code>                      |
| <i>C-c C-v h</i> | or   | <i>C-c C-v C-h</i> | <code>org-babel-describe-bindings</code>              |
| <i>C-c C-v x</i> | or   | <i>C-c C-v C-x</i> | <code>org-babel-do-key-sequence-in-edit-buffer</code> |

## 14.12 Batch execution

Org mode features, including working with source code facilities can be invoked from the command line. This enables building shell scripts for batch processing, running automated system tasks, and expanding Org mode's usefulness.

The sample script shows batch processing of multiple files using `org-babel-tangle`.

```
#!/bin/sh
# tangle files with org-mode
#
emacs -Q --batch --eval "
  (progn
    (require 'ob-tangle)
    (dolist (file command-line-args-left)
      (with-current-buffer (find-file-noselect file)
        (org-babel-tangle))))
" "$@"
```

## 15 Miscellaneous

### 15.1 Completion

Org has in-buffer completions. Unlike minibuffer completions, which are useful for quick command interactions, Org’s in-buffer completions are more suitable for content creation in Org documents. Type one or more letters and invoke the hot key to complete the text in-place. Depending on the context and the keys, Org will offer different types of completions. No minibuffer is involved. Such mode-specific hot keys have become an integral part of Emacs and Org provides several shortcuts.

*M-TAB*      ポイント位置での補完

- 見出しの先頭では、TODO キーワードを補完します。
- ‘\’の後では、エクスポート機能によりサポートされる  $\text{\TeX}$  のシンボルを補完します。
- ‘\*’の後では、‘`[[*find this headline]]`’のようにリンクを検索できるように、カレントバッファで見出しを補完します。
- After ‘:’ in a headline, complete tags. The list of tags is taken from the variable `org-tag-alist` (possibly set through the `#+TAGS` in-buffer option, see Section 6.2 [Setting tags], page 59), or it is created dynamically from all tags used in the current buffer.
- 見出しの外にある ‘:’の後では、プロパティキーを補完します。キーのリストは現在のバッファで使われている全てのキーから動的に構築されます。
- ‘[’の後では、リンクの省略記法を補完します (see `<undefined>` [リンクの省略記法], page `<undefined>`)).
- After ‘#+’, complete the special keywords like ‘TYP\_TODO’ or file-specific ‘OPTIONS’. After option keyword is complete, pressing *M-TAB* again will insert example settings for that option.
- After `#+STARTUP:`, complete startup keywords.
- When the point is anywhere else, complete dictionary words using Ispell.

If your desktop intercepts the combo *M-TAB* to switch windows, use *C-M-i* or *ESC TAB* as an alternative or customize your environment.

### 15.2 Easy templates

With just a few keystrokes, Org’s easy templates inserts empty pairs of structural elements, such as `#+BEGIN_SRC` and `#+END_SRC`. Easy templates use an expansion mechanism, which is native to Org, in a process similar to `yasnippet` and other Emacs template expansion packages.

< s TAB expands to a ‘src’ code block.

< l TAB expands to:

`#+BEGIN_EXPORT latex`

`#+END_EXPORT`

Org comes with these pre-defined easy templates:



```

s      +#+BEGIN_SRC ... +#+END_SRC
e      +#+BEGIN_EXAMPLE ... +#+END_EXAMPLE
q      +#+BEGIN_QUOTE ... +#+END_QUOTE
v      +#+BEGIN_VERSE ... +#+END_VERSE
c      +#+BEGIN_CENTER ... +#+END_CENTER
C      +#+BEGIN_COMMENT ... +#+END_COMMENT
l      +#+BEGIN_EXPORT latex ... +#+END_EXPORT
L      +#+LATEX:
h      +#+BEGIN_EXPORT html ... +#+END_EXPORT
H      +#+HTML:
a      +#+BEGIN_EXPORT ascii ... +#+END_EXPORT
A      +#+ASCII:
i      +#+INDEX: line
I      +#+INCLUDE: line

```

More templates can be added by customizing the variable `org-structure-template-alist`, whose docstring has additional details.

### 15.3 Speed keys

Single keystrokes can execute custom commands in an Org file when the cursor is on a headline. Without the extra burden of a meta or modifier key, Speed Keys can speed navigation or execute custom commands. Besides faster navigation, Speed Keys may come in handy on small mobile devices that do not have full keyboards. Speed Keys may also work on TTY devices known for their problems when entering Emacs keychords.

By default, Org has Speed Keys disabled. To activate Speed Keys, set the variable `org-use-speed-commands` to a non-nil value. To trigger a Speed Key, the cursor must be at the beginning of an Org headline, before any of the stars.

Org comes with a pre-defined list of Speed Keys. To add or modify Speed Keys, customize the variable, `org-speed-commands-user`. For more details, see the variable's docstring. With Speed Keys activated, *M-x org-speed-command-help*, or *?* when cursor is at the beginning of an Org headline, shows currently active Speed Keys, including the user-defined ones.

### 15.4 コードの評価とセキュリティの問題

Unlike plain text, running code comes with risk. Each `'src'` code block, in terms of risk, is equivalent to an executable file. Org therefore puts a few confirmation prompts by default. This is to alert the casual user from accidentally running untrusted code.

For users who do not run code blocks or write code regularly, Org's default settings should suffice. However, some users may want to tweak the prompts for fewer interruptions. To weigh the risks of automatic execution of code blocks, here are some details about code evaluation.

Org evaluates code in the following circumstances:

ソースコードブロック

Org evaluates `'src'` code blocks in an Org file during export. Org also evaluates a `'src'` code block with the *C-c C-c* key chord. Users exporting or running

code blocks must load files only from trusted sources. Be wary of customizing variables that remove or alter default security measures.

**org-confirm-babel-evaluate** [User Option]

When **t**, Org prompts the user for confirmation before executing each code block. When **nil**, Org executes code blocks without prompting the user for confirmation. When this option is set to a custom function, Org invokes the function with these two arguments: the source code language and the body of the code block. The custom function must return either a **t** or **nil**, which determines if the user is prompted. Each source code language can be handled separately through this function argument.

For example, this function enables execution of ‘ditaa’ code +blocks without prompting:

```
(defun my-org-confirm-babel-evaluate (lang body)
  (not (string= lang "ditaa"))) ; don't ask for ditaa
(setq org-confirm-babel-evaluate 'my-org-confirm-babel-evaluate)
```

以下の **shell** と **elisp** はリンクしています。

Org has two link types that can also directly evaluate code (see Section 4.3 [External links], page 39). Because such code is not visible, these links have a potential risk. Org therefore prompts the user when it encounters such links. The customization variables are:

**org-confirm-shell-link-function** [User Option]

Function that prompts the user before executing a shell link.

**org-confirm-elisp-link-function** [User Option]

Function that prompts the user before executing an Emacs Lisp link.

表中の式      Org executes formulas in tables (see Section 3.5 [The spreadsheet], page 24) either through the *calc* or the *Emacs Lisp* interpreters.

## 15.5 Customization

Org has more than 500 variables for customization. They can be accessed through the usual *M-x org-customize RET* command. Or through the Org menu, **Org->Customization->Browse Org Group**. Org also has per-file settings for some variables (see Section 15.6 [In-buffer settings], page 231).

## 15.6 バッファ中での設定の要約

In-buffer settings start with ‘#+’, followed by a keyword, a colon, and then a word for each setting. Org accepts multiple settings on the same line. Org also accepts multiple lines for a keyword. This manual describes these settings throughout. A summary follows here.

**C-c C-c** activates any changes to the in-buffer settings. Closing and reopening the Org file in Emacs also activates the changes.

**#+ARCHIVE: %s\_done::**

Sets the archive location of the agenda file. This location applies to the lines until the next **#+ARCHIVE** line, if any, in the Org file. The first archive location

in the Org file also applies to any entries before it. The corresponding variable is `org-archive-location`.

**`#+CATEGORY:`**

Sets the category of the agenda file, which applies to the entire document.

**`#+COLUMNS: %25ITEM ...`**

Sets the default format for columns view. Org uses this format for column views where there is no `COLUMNS` property.

**`#+CONSTANTS: name1=value1 ...`**

Set file-local values for constants that table formulas can use. This line sets the local variable `org-table-formula-constants-local`. The global version of this variable is `org-table-formula-constants`.

**`#+FILETAGS: :tag1:tag2:tag3:`**

Set tags that all entries in the file will inherit from here, including the top-level entries.

**`#+LINK: linkword replace`**

Each line specifies one abbreviation for one link. Use multiple `#+LINK:` lines for more, see Section 4.6 [Link abbreviations], page 44. The corresponding variable is `org-link-abbrev-alist`.

**`#+PRIORITIES: highest lowest default`**

This line sets the limits and the default for the priorities. All three must be either letters A–Z or numbers 0–9. The highest priority must have a lower ASCII number than the lowest priority.

**`#+PROPERTY: Property_Name Value`**

この行は現在のバッファ中のエントリがデフォルトで引き継ぐ値を設定します。与えられたプロパティの値を指定するのに、最も便利です。

**`#+SETUPFILE: file or URL`**

The setup file or a URL pointing to such file is for additional in-buffer settings. Org loads this file and parses it for any settings in it only when Org opens the main file. If URL is specified, the contents are downloaded and stored in a temporary file cache. `C-c C-c` on the settings line will parse and load the file, and also reset the temporary file cache. Org also parses and loads the document during normal exporting process. Org parses the contents of this document as if it was included in the buffer. It can be another Org file. To visit the file (not a URL), `C-c '` while the cursor is on the line with the file name.

**`#+STARTUP:`**

Startup options Org uses when first visiting a file.

The first set of options deals with the initial visibility of the outline tree. The corresponding variable for global default settings is `org-startup-folded` with a default value of `t`, which is the same as `overview`.

|                       |                |
|-----------------------|----------------|
| <code>overview</code> | トップレベルの見出しのみ   |
| <code>content</code>  | 全ての見出し         |
| <code>showall</code>  | 全てのエントリを畳み込まない |

`showeverything`    コンテンツの引き出しも表示する

Dynamic virtual indentation is controlled by the variable `org-startup-indented`

`indent`            start with `org-indent-mode` turned on

`noindent`        start with `org-indent-mode` turned off

Aligns tables consistently upon visiting a file; useful for restoring narrowed table columns. The corresponding variable is `org-startup-align-all-tables` with `nil` as default value.

`align`            align all tables

`noalign`        donft align tables on startup

Whether Org should automatically display inline images. The corresponding variable is `org-startup-with-inline-images`, with a default value `nil` to avoid delays when visiting a file.

`inlineimages`    show inline images

`noinlineimages` donft show inline images on startup

Whether Org should automatically convert  $\text{\LaTeX}$  fragments to images. The variable `org-startup-with-latex-preview`, which controls this setting, is set to `nil` by default to avoid startup delays.

`latexpreview`    preview  $\text{\LaTeX}$  fragments

`nolatexpreview` donft preview  $\text{\LaTeX}$  fragments

TODO アイテムの完了と再開のログを取ることとその感覚はこれらのオプションによって設定することが可能です (変数 `org-log-done` と `org-log-note-clock-out`、`org-log-repeat` を参照してください)。

|                                  |   |
|----------------------------------|---|
| <code>logdone</code>             | record a timestamp when an item is marked DONE  |
| <code>lognotedone</code>         | record timestamp and a note when DONE           |
| <code>nologdone</code>           | donft record when items are marked DONE         |
| <code>logrepeat</code>           | record a time when reinstating a repeating item |
| <code>lognoterepeat</code>       | record a note when reinstating a repeating item |
| <code>nologrepeat</code>         | do not record when reinstating repeating item   |
| <code>lognoteclock-out</code>    | record a note when clocking out                 |
| <code>nolognoteclock-out</code>  | donft record a note when clocking out           |
| <code>logreschedule</code>       | record a timestamp when scheduling time changes |
| <code>lognotereschedule</code>   | record a note when scheduling time changes      |
| <code>nologreschedule</code>     | do not record when a scheduling date changes    |
| <code>logredeadline</code>       | record a timestamp when deadline changes        |
| <code>lognoteredecline</code>    | record a note when deadline changes             |
| <code>nologredeadline</code>     | do not record when a deadline date changes      |
| <code>logrefile</code>           | record a timestamp when refiling                |
| <code>lognoterefile</code>       | record a note when refiling                     |
| <code>nologrefile</code>         | do not record when refiling                     |
| <code>logdrawer</code>           | store log into drawer                           |
| <code>nologdrawer</code>         | store log outside of drawer                     |
| <code>logstatesreversed</code>   | reverse the order of states notes               |
| <code>nologstatesreversed</code> | do not reverse the order of states notes        |

These options hide leading stars in outline headings, and indent outlines. The corresponding variables are `org-hide-leading-stars` and `org-odd-levels-only`, both with a default setting of `nil` (meaning `showstars` and `oddeven`).

|                        |  |
|------------------------|--|
| <code>hidestars</code> | hide all stars on the headline except one.     |
| <code>showstars</code> | show all stars on the headline                 |
| <code>indent</code>    | virtual indents according to the outline level |
| <code>noindent</code>  | no virtual indents                             |
| <code>odd</code>       | show odd outline levels only (1,3,...)         |
| <code>oddeven</code>   | show all outline levels                        |

タイムスタンプ (`org-put-time-stamp-overlays` と `org-time-stamp-overlay-formats` 変数) のカスタムフォーマットオーバーレイを切り替えます、

`customtime` カスタマイズされたタイムフォーマットで覆います

以下のオプションは表計算 (`constants-unit-system` 変数) に影響を与えます。

|                       |  |
|-----------------------|--|
| <code>constcgs</code> | <code>constants.el</code> should use the c-g-s unit system |
| <code>constSI</code>  | <code>constants.el</code> should use the SI unit system    |

For footnote settings, use the following keywords. The corresponding variables are `org-footnote-define-inline`, `org-footnote-auto-label`, and `org-footnote-auto-adjust`.

|                         |                                       |
|-------------------------|---------------------------------------|
| <code>fninline</code>   | 脚注のインラインを定義します                        |
| <code>fnnoinline</code> | 分割されたセクションでの脚注を定義します                  |
| <code>fnlocal</code>    | 最初の言及元の近くの脚注を定義しますが、インラインではありません      |
| <code>fnprompt</code>   | 脚注ラベルのプロンプト                           |
| <code>fnauto</code>     | [fn:1] のようなラベルを自動的に作成します (これがデフォルトです) |
| <code>fnconfirm</code>  | 編集と確認用の自動ラベルを用意します                    |
| <code>fnplain</code>    | [1] のようなラベルを自動的に作成します                 |
| <code>fnadjust</code>   | 自動的に脚注の番号を振り直し、ソートします                 |
| <code>nofnadjust</code> | 自動的に番号の振り直しとソートを行いません                 |

To hide blocks on startup, use these keywords. The corresponding variable is `org-hide-block-startup`.

|                           |                          |
|---------------------------|--------------------------|
| <code>hideblocks</code>   | 開始時に全ての開始/終了ブロックを見えなくします |
| <code>nohideblocks</code> | 開始時にブロックを見えなくしません        |

UTF-8 の文字の表示は変数 `org-pretty-entities` と以下のキーワードにより制御されます。

|                             |                      |
|-----------------------------|----------------------|
| <code>entitiespretty</code> | 可能な時、UTF-8 の文字を表示します |
| <code>entitiesplain</code>  | 空白にします               |

**#+TAGS:** *TAG1(c1) TAG2(c2)*

These lines specify valid tags for this file. Org accepts multiple tags lines. Tags could correspond to the *fast tag selection* keys. The corresponding variable is `org-tag-alist`.

**#+TBLFM:** This line is for formulas for the table directly above. A table can have multiple **#+TBLFM:** lines. On table recalculation, Org applies only the first **#+TBLFM:**

line. For details see [Using multiple `#+TBLFM` lines], page 32 in Section 3.5.8 [Editing and debugging formulas], page 31.

`#+TITLE:`, `#+AUTHOR:`, `#+EMAIL:`, `#+LANGUAGE:`, `#+DATE:`,  
`#+OPTIONS:`, `#+BIND:`,  
`#+SELECT_TAGS:`, `#+EXCLUDE_TAGS:`

These lines provide settings for exporting files. For more details see Section 12.2 [Export settings], page 139.

`#+TODO:` `#+SEQ_TODO:` `#+TYP_TODO:`

These lines set the TODO keywords and their significance to the current file. The corresponding variable is `org-todo-keywords`.

## 15.7 The very busy C-c C-c key

The C-c C-c key in Org serves many purposes depending on the context. It is probably the most over-worked, multi-purpose key combination in Org. Its uses are well-documented through out this manual, but here is a consolidated list for easy reference.

- If any highlights shown in the buffer from the creation of a sparse tree, or from clock display, remove such highlights.
- If the cursor is in one of the special `#+KEYWORD` lines, scan the buffer for these lines and update the information. Also reset the Org file cache used to temporary store the contents of URLs used as values for keywords like `#+SETUPFILE`.
- If the cursor is inside a table, realign the table. The table realigns even if automatic table editor is turned off.
- カーソルが`#+TBLFM`行にあるなら、全ての表に式を再適用します。
- If the current buffer is a capture buffer, close the note and file it. With a prefix argument, also jump to the target location after saving the note.
- カーソルが`<<<target>>>`上にあるなら、ラジオターゲットとバッファ中の関係するリンクを更新します。
- If the cursor is on a property line or at the start or end of a property drawer, offer property commands.
- If the cursor is at a footnote reference, go to the corresponding definition, and *vice versa*.
- カーソルが統計データクッキー上にあるなら、それを更新します。
- カーソルがチェックボックス付きのプレーンリスト上にあるなら、チェックボックスのステータスをトグルします。
- カーソルが数字付きリスト上にあるなら、順序を整理しなおします。
- If the cursor is on the `#+BEGIN` line of a dynamic block, the block is updated.
- If the cursor is at a timestamp, fix the day name in the timestamp.

## 15.8 より見やすいアウトラインビュー

Org's default outline with stars and no indents can become too cluttered for short documents. For *book-like* long documents, the effect is not as noticeable. Org provides an alternate stars and indentation scheme, as shown on the right in the following table. It uses only one star and indents text to line with the heading:

|                              |  |                              |
|------------------------------|--|------------------------------|
| * Top level headline         |  | * Top level headline         |
| ** Second level              |  | * Second level               |
| *** 3rd level                |  | * 3rd level                  |
| some text                    |  | some text                    |
| *** 3rd level                |  | * 3rd level                  |
| more text                    |  | more text                    |
| * Another top level headline |  | * Another top level headline |

To turn this mode on, use the minor mode, `org-indent-mode`. Text lines that are not headlines are prefixed with spaces to vertically align with the headline text<sup>1</sup>.

To make more horizontal space, the headlines are shifted by two stars. This can be configured by the `org-indent-indentation-per-level` variable. Only one star on each headline is visible, the rest are masked with the same font color as the background. This font face can be configured with the `org-hide` variable.

Note that turning on `org-indent-mode` sets `org-hide-leading-stars` to `t` and `org-adapt-indentation` to `nil`; ‘2.’ below shows how this works.

To globally turn on `org-indent-mode` for all files, customize the variable `org-startup-indented`.

To turn on indenting for individual files, use `#+STARTUP` option as follows:

```
#+STARTUP: indent
```

Indent on startup makes Org use hard spaces to align text with headings as shown in examples below.

1. *Indentation of text below headlines*

Indent text to align with the headline.

```
*** 3rd level
    more text, now indented
```

Org adapts indentations with paragraph filling, line wrapping, and structure editing<sup>2</sup>.

2. *Hiding leading stars*

Org can make leading stars invisible. For global preference, configure the variable `org-hide-leading-stars`. For per-file preference, use these file `#+STARTUP` options:

```
#+STARTUP: hidestars
#+STARTUP: showstars
```

With stars hidden, the tree is shown as:

```
* Top level headline
* Second level
* 3rd level
...
```

Because Org makes the font color same as the background color to hide to stars, sometimes `org-hide` face may need tweaking to get the effect right. For some black and white combinations, `grey90` on a white background might mask the stars better.

<sup>1</sup> The `org-indent-mode` also sets the `wrap-prefix` correctly for indenting and wrapping long lines of headlines or text. This minor mode handles `visual-line-mode` and directly applied settings through `word-wrap`.

<sup>2</sup> Also see the variable `org-adapt-indentation`.

- Using stars for only odd levels, 1, 3, 5, . . . , can also clean up the clutter. This removes two stars from each level<sup>3</sup>. For Org to properly handle this cleaner structure during edits and exports, configure the variable `org-odd-levels-only`. To set this per-file, use either one of the following lines:

```
#+STARTUP: odd
#+STARTUP: oddeven
```

To switch between single and double stars layouts, use `M-x org-convert-to-odd-levels RET` and `M-x org-convert-to-oddeven-levels RET`.

## 15.9 Using Org on a tty

Org provides alternative key bindings for TTY and modern mobile devices that cannot handle cursor keys and complex modifier key chords. Some of these workarounds may be more cumbersome than necessary. Users should look into customizing these further based on their usage needs. For example, the normal `S-cursor` for editing timestamp might be better with `C-c . chord`.

| デフォルト            | 代替 1                 | スピー<br>ドキー | 代替 2             |
|------------------|----------------------|------------|------------------|
| <i>S-TAB</i>     | <i>C-u TAB</i>       | <i>C</i>   |                  |
| <i>M-LEFT</i>    | <i>C-c C-x l</i>     | <i>l</i>   | <i>Esc LEFT</i>  |
| <i>M-S-LEFT</i>  | <i>C-c C-x L</i>     | <i>L</i>   |                  |
| <i>M-RIGHT</i>   | <i>C-c C-x r</i>     | <i>r</i>   | <i>Esc RIGHT</i> |
| <i>M-S-RIGHT</i> | <i>C-c C-x R</i>     | <i>R</i>   |                  |
| <i>M-UP</i>      | <i>C-c C-x u</i>     |            | <i>Esc UP</i>    |
| <i>M-S-UP</i>    | <i>C-c C-x U</i>     | <i>U</i>   |                  |
| <i>M-DOWN</i>    | <i>C-c C-x d</i>     |            | <i>Esc DOWN</i>  |
| <i>M-S-DOWN</i>  | <i>C-c C-x D</i>     | <i>D</i>   |                  |
| <i>S-RET</i>     | <i>C-c C-x c</i>     |            |                  |
| <i>M-RET</i>     | <i>C-c C-x m</i>     |            | <i>Esc RET</i>   |
| <i>M-S-RET</i>   | <i>C-c C-x M</i>     |            |                  |
| <i>S-LEFT</i>    | <i>C-c LEFT</i>      |            |                  |
| <i>S-RIGHT</i>   | <i>C-c RIGHT</i>     |            |                  |
| <i>S-UP</i>      | <i>C-c UP</i>        |            |                  |
| <i>S-DOWN</i>    | <i>C-c DOWN</i>      |            |                  |
| <i>C-S-LEFT</i>  | <i>C-c C-x LEFT</i>  |            |                  |
| <i>C-S-RIGHT</i> | <i>C-c C-x RIGHT</i> |            |                  |

## 15.10 Interaction with other packages

Org's compatibility and the level of interaction with other Emacs packages are documented here.

---

<sup>3</sup> Because 'LEVEL=2' has 3 stars, 'LEVEL=3' has 4 stars, and so on



### 15.10.1 Packages that Org cooperates with

`calc.el` by Dave Gillespie

Org uses the Calc package for tables to implement spreadsheet functionality (see Section 3.5 [The spreadsheet], page 24). Org also uses Calc for embedded calculations. See Section “Embedded Mode” in *GNU Emacs Calc Manual*.

`constants.el` by Carsten Dominik

Org can use names for constants in formulas in tables. Org can also use calculation suffixes for units, such as ‘M’ for ‘Mega’. For a standard collection of such constants, install the `constants` package. Install version 2.0 of this package, available at <https://staff.fnwi.uva.nl/c.dominik/Tools/>. Org checks if the function `constants-get` has been autoloaded. Installation instructions are in the file, `constants.el`.

`cdlatex.el` by Carsten Dominik

Org mode can use CD<sub>L</sub>A<sub>T</sub>E<sub>X</sub> package to efficiently enter L<sub>A</sub>T<sub>E</sub>X fragments into Org files (see Section 11.8.3 [CD<sub>L</sub>A<sub>T</sub>E<sub>X</sub> mode], page 137).

`imenu.el` by Ake Stenhoff and Lars Lindberg

Imenu creates dynamic menus based on an index of items in a file. Org mode supports Imenu menus. Enable it with a mode hook as follows:

```
(add-hook 'org-mode-hook
  (lambda () (imenu-add-to-menubar "Imenu")))
```

By default the Imenu index is two levels deep. Change the index depth using the variable, `org-imenu-depth`.

`speedbar.el` by Eric M. Ludlam

Speedbar package creates a special Emacs frame for displaying files and index items in files. Org mode supports Speedbar; users can drill into Org files directly from the Speedbar. The < in the Speedbar frame tweaks the agenda commands to that file or to a subtree.

`table.el` by Takaaki Ota

Complex ASCII tables with automatic line wrapping, column- and row-spanning, and alignment can be created using the Emacs table package by Takaaki Ota. Org mode recognizes such tables and export them properly. `C-c` ' to edit these tables in a special buffer, much like Org's 'src' code blocks. Because of interference with other Org mode functionality, Takaaki Ota tables cannot be edited directly in the Org buffer.

```
C-c ' org-edit-special
      Edit a table.el table. Works when the cursor is in a table.el table.
```

```
C-c ~ org-table-create-with-table.el
      Insert a table.el table. If there is already a table at point,
      this command converts it between the table.el format and the
      Org mode format. See the documentation string of the command
      org-convert-table for details.
```

### 15.10.2 Packages that conflict with Org mode

In Emacs, `shift-selection-mode` combines cursor motions with shift key to enlarge regions. Emacs sets this mode by default. This conflicts with Org’s use of `S-cursor` commands to change timestamps, TODO keywords, priorities, and item bullet types, etc. Since `S-cursor` commands outside of specific contexts don’t do anything, Org offers the variable `org-support-shift-select` for customization. Org mode accommodates shift selection by (i) making it available outside of the special contexts where special commands apply, and (ii) extending an existing active region even if the cursor moves across a special context.

`CUA.el` by Kim. F. Storm

Org key bindings conflict with `S-<cursor>` keys used by CUA mode. For Org to relinquish these bindings to CUA mode, configure the variable `org-replace-disputed-keys`. When set, Org moves the following key bindings in Org files, and in the agenda buffer (but not during date selection).

|                       |               |                    |                        |               |                    |
|-----------------------|---------------|--------------------|------------------------|---------------|--------------------|
| <code>S-UP</code>     | $\Rightarrow$ | <code>M-p</code>   | <code>S-DOWN</code>    | $\Rightarrow$ | <code>M-n</code>   |
| <code>S-LEFT</code>   | $\Rightarrow$ | <code>M--</code>   | <code>S-RIGHT</code>   | $\Rightarrow$ | <code>M-+</code>   |
| <code>C-S-LEFT</code> | $\Rightarrow$ | <code>M-S--</code> | <code>C-S-RIGHT</code> | $\Rightarrow$ | <code>M-S-+</code> |

Yes, these are unfortunately more difficult to remember. To define a different replacement keys, look at the variable `org-disputed-keys`.

`ecomplete.el` by Lars Magne Ingebrigtsen [larsi@gnus.org](mailto:larsi@gnus.org)

Ecomplete provides “electric” address completion in address header lines in message buffers. Sadly Orgtbl mode cuts ecompletes power supply: No completion happens when Orgtbl mode is enabled in message buffers while entering text in address header lines. If one wants to use ecomplete one should *not* follow the advice to automatically turn on Orgtbl mode in message buffers (see Section 3.4 [Orgtbl mode], page 23), but instead—after filling in the message headers—turn on Orgtbl mode manually when needed in the messages body.

`filladapt.el` by Kyle Jones

Org mode tries to do the right thing when filling paragraphs, list items and other elements. Many users reported problems using both `filladapt.el` and Org mode, so a safe thing to do is to disable `filladapt` like this:

```
(add-hook 'org-mode-hook 'turn-off-filladapt-mode)
```

`yasnippet.el`

The way Org mode binds the TAB key (binding to `[tab]` instead of `"\t"`) overrules YASnippet’s access to this key. The following code fixed this problem:

```
(add-hook 'org-mode-hook
  (lambda ()
    (setq-local yas/trigger-key [tab])
    (define-key yas/keymap [tab] 'yas/next-field-or-maybe-expand)))
```

The latest version of `yasnippet` doesn’t play well with Org mode. If the above code does not fix the conflict, first define the following function:

```
(defun yas/org-very-safe-expand ())
```

```
(let ((yas/fallback-behavior 'return-nil)) (yas/expand)))
```

Then tell Org mode to use that function:

```
(add-hook 'org-mode-hook
  (lambda ()
    (make-variable-buffer-local 'yas/trigger-key)
    (setq yas/trigger-key [tab])
    (add-to-list 'org-tab-first-hook 'yas/org-very-safe-expand)
    (define-key yas/keymap [tab] 'yas/next-field)))
```

`windmove.el` by Hovav Shacham

This package also uses the *S-cursor* keys, so everything written in the paragraph above about CUA mode also applies here. If you want make the windmove function active in locations where Org mode does not have special functionality on *S-cursor*, add this to your configuration:

```
;; Make windmove work in org-mode:
(add-hook 'org-shiftup-final-hook 'windmove-up)
(add-hook 'org-shiftright-final-hook 'windmove-right)
(add-hook 'org-shiftleft-final-hook 'windmove-left)
(add-hook 'org-shiftdown-final-hook 'windmove-down)
```

`viper.el` by Michael Kifer

Viper uses *C-c /* and therefore makes this key not access the corresponding Org mode command `org-sparse-tree`. You need to find another key for this command, or override the key in `viper-vi-global-user-map` with

```
(define-key viper-vi-global-user-map "C-c /" 'org-sparse-tree)
```

## 15.11 org-crypt.el

Org crypt encrypts the text of an Org entry, but not the headline, or properties. Org crypt uses the Emacs EasyPG library to encrypt and decrypt.

Any text below a headline that has a `:crypt:` tag will be automatically be encrypted when the file is saved. To use a different tag, customize the `org-crypt-tag-matcher` variable.

Suggested Org crypt settings in Emacs init file:

```
(require 'org-crypt)
(org-crypt-use-before-save-magic)
(setq org-tags-exclude-from-inheritance (quote ("crypt")))

(setq org-crypt-key nil)
;; GPG key to use for encryption
;; Either the Key ID or set to nil to use symmetric encryption.

(setq auto-save-default nil)
;; Auto-saving does not cooperate with org-crypt.el: so you need
;; to turn it off if you plan to use org-crypt.el quite often.
;; Otherwise, you'll get an (annoying) message each time you
;; start Org.
```

```
;; To turn it off only locally, you can insert this:  
;;  
;; # -*- buffer-auto-save-file-name: nil; -*-
```

Excluding the crypt tag from inheritance prevents encrypting previously encrypted text.

## Appendix A Hacking

This appendix covers some areas where users can extend the functionality of Org.

### A.1 Hooks

Org has a large number of hook variables for adding functionality. This appendix illustrates using a few. A complete list of hooks with documentation is maintained by the Worg project at <https://orgmode.org/worg/doc.html#hooks>.

### A.2 Add-on packages

Various authors wrote a large number of add-on packages for Org.

These packages are not part of Emacs, but they are distributed as contributed packages with the separate release available at <https://orgmode.org>. See the `contrib/README` file in the source code directory for a list of contributed files. Worg page with more information is at: <https://orgmode.org/worg/org-contrib/>.

### A.3 Adding hyperlink types

Org has many built-in hyperlink types (see Chapter 4 [Hyperlinks], page 38), and an interface for adding new link types. The example file, `org-man.el`, shows the process of adding Org links to Unix man pages, which look like this: ‘`[[man:printf][The printf manpage]]`’:

```
;;; org-man.el - Org-mode で man ページのリンクをサポートする

(require 'org)

(org-add-link-type "man" 'org-man-open)
(add-hook 'org-store-link-functions 'org-man-store-link)

(defcustom org-man-command 'man
  "man のページを表示するための Emacs のコマンド。"
  :group 'org-link
  :type '(choice (const man) (const woman)))

(defun org-man-open (path)
  "パス (PATH) にある manpage を開きます。
  パスはコマンド man に渡せる内容でなければなりません。"
  (funcall org-man-command path))

(defun org-man-store-link ()
  "man ページへのリンクを保存します。"
  (when (memq major-mode '(Man-mode woman-mode))
    ;; これは man ページなので、リンクを作成します。
    (let* ((page (org-man-get-page-name))
           (link (concat "man:" page))
           (description (format "Manpage for %s" page)))
```

```

(org-store-link-props
 :type "man"
 :link link
 :description description))))

(defun org-man-get-page-name ()
  "バッファ名からページ名を抽出します。"
  ;; `Man-mode' と `woman-mode' の両方で動作します。
  (if (string-match "\\(\\S-+\\)\\*" (buffer-name))
      (match-string 1 (buffer-name))
      (error "この man ページへのリンクを作成できません")))

(provide 'org-man)

;;; org-man.el ends here

```

To activate links to man pages in Org, enter this in the init file:

```
(require 'org-man)
```

A review of `org-man.el`:

1. First, `(require 'org)` ensures `org.el` is loaded.
2. The `org-add-link-type` defines a new link type with ‘man’ prefix. The call contains the function to call that follows the link type.
3. The next line adds a function to `org-store-link-functions` that records a useful link with the command `C-c l` in a buffer displaying a man page.

The rest of the file defines necessary variables and functions. First is the customization variable `org-man-command`. It has two options, `man` and `woman`. Next is a function whose argument is the link path, which for man pages is the topic of the man command. To follow the link, the function calls the `org-man-command` to display the man page.

`C-c l` constructs and stores the link.

`C-c l` calls the function `org-man-store-link`, which first checks if the `major-mode` is appropriate. If check fails, the function returns `nil`. Otherwise the function makes a link string by combining the ‘man:’ prefix with the man topic. The function then calls `org-store-link-props` with `:type` and `:link` properties. A `:description` property is an optional string that is displayed when the function inserts the link in the Org buffer.

`C-c C-l` inserts the stored link.

To define new link types, define a function that implements completion support with `C-c C-l`. This function should not accept any arguments but return the appropriate prefix and complete link string.

## A.4 Adding export back-ends

Org’s export engine makes it easy for writing new back-ends. The framework on which the engine was built makes it easy to derive new back-ends from existing ones.

The two main entry points to the export engine are: `org-export-define-backend` and `org-export-define-derived-backend`. To grok these functions, see `ox-latex.el` for an

example of defining a new back-end from scratch, and `ox-beamer.el` for an example of deriving from an existing engine.

For creating a new back-end from scratch, first set its name as a symbol in an alist consisting of elements and export functions. To make the back-end visible to the export dispatcher, set `:menu-entry` keyword. For export options specific to this back-end, set the `:options-alist`.

For creating a new back-end from an existing one, set `:translate-alist` to an alist of export functions. This alist replaces the parent back-end functions.

For complete documentation, see the Org Export Reference on Worg (<https://orgmode.org/worg/dev/org-export-reference.html>).

## A.5 Context-sensitive commands

Org has facilities for building context sensitive commands. Authors of Org add-ons can tap into this functionality.

Some Org commands change depending on the context. The most important example of this behavior is the `C-c C-c` (see Section 15.7 [The very busy C-c C-c key], page 235). Other examples are `M-cursor` and `M-S-cursor`.

These context sensitive commands work by providing a function that detects special context for that add-on and executes functionality appropriate for that context.

## A.6 Tables and lists in arbitrary syntax

Because of Org's success in handling tables with `Orgtbl`, a frequently asked feature is to Org's usability functions to other table formats native to other modems, such as  $\text{\LaTeX}$ . This would be hard to do in a general way without complicated customization nightmares. Moreover, that would take Org away from its simplicity roots that `Orgtbl` has proven. There is, however, an alternate approach to accomplishing the same.

This approach involves implementing a custom *translate* function that operates on a native Org *source table* to produce a table in another format. This strategy would keep the excellently working `Orgtbl` simple and isolate complications, if any, confined to the *translate* function. To add more alien table formats, we just add more *translate* functions. Also the burden of developing custom *translate* functions for new table formats will be in the hands of those who know those formats best.

For an example of how this strategy works, see `Orgstruct` mode. In that mode, Bastien added the ability to use Org's facilities to edit and re-structure lists. He did by turning `orgstruct-mode` on, and then exporting the list locally to another format, such as HTML,  $\text{\LaTeX}$  or Texinfo.

### A.6.1 Radio tables

Radio tables are target locations for translated tables that are not near their source. Org finds the target location and inserts the translated table.

The key to finding the target location are the magic words `BEGIN/END RECEIVE ORGTBL`. They have to appear as comments in the current mode. If the mode is C, then:

```
/* BEGIN RECEIVE ORGTBL table_name */
/* END RECEIVE ORGTBL table_name */
```

At the location of source, Org needs a special line to direct Orgtbl to translate and to find the target for inserting the translated table. For example:

```
#+ORGTBL: SEND table_name translation_function arguments...
```

`table_name` is the table's reference name, which is also used in the receiver lines, and the `translation_function` is the Lisp function that translates. This line, in addition, may also contain alternating key and value arguments at the end. The translation function gets these values as a property list. A few standard parameters are already recognized and acted upon before the translation function is called:

`:skip N` Skip the first `N` lines of the table. Hlines do count; include them if they are to be skipped.

`:skipcols (n1 n2 ...)`

List of columns to be skipped. First Org automatically discards columns with calculation marks and then sends the table to the translator function, which then skips columns as specified in `'skipcols'`.

To keep the source table intact in the buffer without being disturbed when the source file is compiled or otherwise being worked on, use one of these strategies:

- Place the table in a block comment. For example, in C mode you could wrap the table between `/*` and `*/` lines.
- Put the table after an `'END'` statement. For example `'\bye'` in  $\text{\TeX}$  and `'\end{document}'` in  $\text{\LaTeX}$ .
- Comment and uncomment each line of the table during edits. The *M-x orgtbl-toggle-comment RET* command makes toggling easy.

### A.6.2 $\text{\LaTeX}$ でのラジオテーブルの例

To wrap a source table in  $\text{\LaTeX}$ , use the `comment` environment provided by `comment.sty`. To activate it, put `\usepackage{comment}` in the document header. Orgtbl mode inserts a radio table skeleton<sup>1</sup> with the command *M-x orgtbl-insert-radio-table RET*, which prompts for a table name. For example, if `'salesfigures'` is the name, the template inserts:

```
% BEGIN RECEIVE ORGTBL salesfigures
% END RECEIVE ORGTBL salesfigures
\begin{comment}
#+ORGTBL: SEND salesfigures orgtbl-to-latex
| | |
\end{comment}
```

The line `#+ORGTBL: SEND` tells Orgtbl mode to use the function `orgtbl-to-latex` to convert the table to  $\text{\LaTeX}$  format, then insert the table at the target (receive) location named `salesfigures`. Now the table is ready for data entry. It can even use spreadsheet features<sup>2</sup>:

<sup>1</sup> By default this works only for  $\text{\LaTeX}$ , HTML, and Texinfo. Configure the variable `orgtbl-radio-table-templates` to install templates for other export formats.

<sup>2</sup> If the `#+TBLFM` line contains an odd number of dollar characters, this may cause problems with font-lock in  $\text{\LaTeX}$  mode. As shown in the example you can fix this by adding an extra line inside the `comment` environment that is used to balance the dollar expressions. If you are using  $\text{\LaTeX}$  with the font-latex library, a much better solution is to add the `comment` environment to the variable `LaTeX-verbatim-environments`.



```
% BEGIN RECEIVE ORGTBL salesfigures
% END RECEIVE ORGTBL salesfigures
\begin{comment}
#+ORGTBL: SEND salesfigures orgtbl-to-latex
| 月 | 日数 | 販売数 | 一日あたり |
|-----+-----+-----+-----|
| 1月 | 23 | 55 | 2.4 |
| 2月 | 21 | 16 | 0.8 |
| 3月 | 22 | 278 | 12.6 |
#+TBLFM: $4=$3/$2;%.1f
% $ (font-lock 色分けをうまく動作させるための余分なドル記号、脚注参照)
\end{comment}
```

After editing, `C-c C-c` inserts translated table at the target location, between the two marker lines.

For hand-made custom tables, note that the translator needs to skip the first two lines of the source table. Also the command has to *splice* out the target table without the header and footer.

```
\begin{tabular}{lrrrr}
月 & \multicolumn{1}{c}{日数} & 販売数 & 一日あたり & \\
% BEGIN RECEIVE ORGTBL salesfigures
1月 & 23 & 55 & 2.4 & \\
2月 & 21 & 16 & 0.8 & \\
3月 & 22 & 278 & 12.6 & \\
% END RECEIVE ORGTBL salesfigures
\end{tabular}
%
\begin{comment}
#+ORGTBL: SEND salesfigures orgtbl-to-latex :splice t :skip 2
| 月 | 日数 | 販売数 | 1日あたり |
|-----+-----+-----+-----|
| 1月 | 23 | 55 | 2.4 |
| 2月 | 21 | 16 | 0.8 |
| 3月 | 22 | 278 | 12.6 |
#+TBLFM: $4=$3/$2;%.1f
\end{comment}
```

The  $\text{\LaTeX}$  translator function `orgtbl-to-latex` is already part of `Orgtbl` mode and uses `tabular` environment by default to typeset the table and mark the horizontal lines with `\hline`. For additional parameters to control output, see Section A.6.3 [Translator functions], page 247:

`:splice nil/t`

When non-`nil`, returns only table body lines; not wrapped in tabular environment. Default is `nil`.

`:fmt fmt` Format to warp each field. It should contain `%s` for the original field value. For example, to wrap each field value in dollar symbol, you could use `:fmt "$%s$"`. Format can also wrap a property list with column numbers and formats, for

example `:fmt (2 "$s$" 4 "%s\\%")`. In place of a string, a function of one argument can be used; the function must return a formatted string.

`:efmt efmt`

Format numbers as exponentials. The spec should have `%s` twice for inserting mantissa and exponent, for example `"%s\\times10^{%s}"`. This may also be a property list with column numbers and formats, for example `:efmt (2 "$s\\times10^{%s}" 4 "$s\\cdot10^{%s}")`. After `efmt` has been applied to a value, `fmt` will also be applied. Functions with two arguments can be supplied instead of strings. By default, no special formatting is applied.

### A.6.3 Translator functions

Orgtbl mode has built-in translator functions: `orgtbl-to-csv` (comma-separated values), `orgtbl-to-tsv` (TAB-separated values), `orgtbl-to-latex`, `orgtbl-to-html`, `orgtbl-to-texinfo`, `orgtbl-to-unicode` and `orgtbl-to-orgtbl`. They use the generic translator, `orgtbl-to-generic`, which delegates translations to various export back-ends.

Properties passed to the function through the ‘`ORGTBL SEND`’ line take precedence over properties defined inside the function. For example, this overrides the default  $\LaTeX$  line endings, ‘`\\`’, with ‘`\\[2mm]`’:

```
#+ORGTBL: SEND test orgtbl-to-latex :lend " \\[2mm]"
```

For a new language translator, define a converter function. It can be a generic function, such as shown in this example. It marks a beginning and ending of a table with ‘`!BTBL!`’ and ‘`!ETBL!`’; a beginning and ending of lines with ‘`!BL!`’ and ‘`!EL!`’; and uses a TAB for a field separator:

```
(defun orgtbl-to-language (table params)
  "Convert the orgtbl-mode TABLE to language."
  (orgtbl-to-generic
   table
   (org-combine-plists
    '(:tstart "!BTBL!" :tend "!ETBL!" :lstart "!BL!" :lend "!EL!" :sep "\t")
    params)))
```

The documentation for the `orgtbl-to-generic` function shows a complete list of parameters, each of which can be passed through to `orgtbl-to-latex`, `orgtbl-to-texinfo`, and any other function using that generic function.

For complicated translations the generic translator function could be replaced by a custom translator function. Such a custom function must take two arguments and return a single string containing the formatted table. The first argument is the table whose lines are a list of fields or the symbol `hline`. The second argument is the property list consisting of parameters specified in the ‘`#+ORGTBL: SEND`’ line. Please share your translator functions by posting them to the Org users mailing list, [emacs-orgmode@gnu.org](mailto:emacs-orgmode@gnu.org).

### A.6.4 Radio lists

Call the `org-list-insert-radio-list` function to insert a radio list template in HTML,  $\LaTeX$ , and Texinfo mode documents. Sending and receiving radio lists works is the same as for radio tables (see Section A.6.1 [Radio tables], page 244) except for these differences:

- Orgstruct モードが有効でなければなりません。

- Use `ORGLST` keyword instead of `ORGTBL`.
- `C-c C-c` works only on the first list item.

Built-in translators functions are: `org-list-to-latex`, `org-list-to-html` and `org-list-to-texinfo`. They use the `org-list-to-generic` translator function. See its documentation for parameters for accurate customizations of lists. Here is a  $\text{\LaTeX}$  example:

```
% BEGIN RECEIVE ORGLST to-buy
% END RECEIVE ORGLST to-buy
\begin{comment}
#+ORGLST: SEND to-buy org-list-to-latex
- 新しい家
- 新しいコンピュータ
  + 新しいキーボード
  + 新しいマウス
- 新しい生活
\end{comment}
```

`C-c C-c` on ‘a new house’ inserts the translated  $\text{\LaTeX}$  list in-between the BEGIN and END marker lines.

## A.7 Dynamic blocks

Org supports *dynamic blocks* in Org documents. They are inserted with begin and end markers like any other ‘src’ code block, but the contents are updated automatically by a user function. For example, `C-c C-x C-r` inserts a dynamic table that updates the work time (see Section 8.4 [Clocking work time], page 81).

Dynamic blocks can have names and function parameters. The syntax is similar to ‘src’ code block specifications:

```
#+BEGIN: myblock :parameter1 value1 :parameter2 value2 ...

#+END:
```

These command update dynamic blocks:

```
C-c C-x C-u                                     org-dblock-update
    その場所での動的なブロックを更新します。
```

```
C-u C-c C-x C-u
    現在のファイルの全ての動的なブロックを更新します。
```

Before updating a dynamic block, Org removes content between the BEGIN and END markers. Org then reads the parameters on the BEGIN line for passing to the writer function. If the function expects to access the removed content, then Org expects an extra parameter, `:content`, on the BEGIN line.

To syntax for calling a writer function with a named block, `myblock` is: `org-dblock-write:myblock`. Parameters come from the BEGIN line.

The following is an example of a dynamic block and a block writer function that updates the time when the function was last run:

```
#+BEGIN: block-update-time :format "on %m/%d/%Y at %H:%M"

#+END:
```

The dynamic block's writer function:

```
(defun org-dblock-write:block-update-time (params)
  (let ((fmt (or (plist-get params :format) "%d. %m. %Y")))
    (insert "Last block update at: "
            (format-time-string fmt))))
```

To keep dynamic blocks up-to-date in an Org file, use the function, `org-update-all-dblocks` in hook, such as `before-save-hook`. The `org-update-all-dblocks` function does not run if the file is not in Org mode.

Dynamic blocks, like any other block, can be narrowed with `org-narrow-to-block`.

## A.8 Special agenda views

Org provides a special hook to further limit items in agenda views: `agenda`, `agenda*`<sup>3</sup>, `todo`, `alltodo`, `tags`, `tags-todo`, `tags-tree`. Specify a custom function that tests inclusion of every matched item in the view. This function can also skip as much as is needed.

For a global condition applicable to agenda views, use the `org-agenda-skip-function-global` variable. Org uses a global condition with `org-agenda-skip-function` for custom searching.

This example defines a function for a custom view showing TODO items with WAITING status. Manually this is a multi step search process, but with a custom view, this can be automated as follows:

The custom function searches the subtree for the WAITING tag and returns `nil` on match. Otherwise it gives the location from where the search continues.

```
(defun my-skip-unless-waiting ()
  "waiting でないツリーをスキップします"
  (let ((subtree-end (save-excursion (org-end-of-subtree t))))
    (if (re-search-forward ":waiting:" subtree-end t)
        nil ; タグが見つかったので、スキップしません
        subtree-end))) ; タグが見つからないので、サブツリーの最後から検索を再開します
```

To use this custom function in a custom agenda command:

```
(org-add-agenda-custom-command
  '("b" todo "PROJECT"
    ((org-agenda-skip-function 'my-skip-unless-waiting)
     (org-agenda-overriding-header "waiting (待ち) 状態のプロジェクト: "))))
```

Note that this also binds `org-agenda-overriding-header` to a more meaningful string suitable for the agenda view.

<sup>3</sup> The `agenda*` view is the same as `agenda` except that it only considers *appointments*, i.e., scheduled and deadline items that have a time specification '[h]h:mm' in their time-stamps.

Search for entries with a limit set on levels for the custom search. This is a general approach to creating custom searches in Org. To include all levels, use ‘LEVEL>0’<sup>4</sup>. Then to selectively pick the matched entries, use `org-agenda-skip-function`, which also accepts Lisp forms, such as `org-agenda-skip-entry-if` and `org-agenda-skip-subtree-if`. For example:

```
(org-agenda-skip-entry-if 'scheduled)
  現在のエントリがスケジューリングされている場合、それをスキップします。

(org-agenda-skip-entry-if 'notscheduled)
  現在のエントリがスケジューリングされていない場合、それをスキップします。

(org-agenda-skip-entry-if 'deadline)
  現在のエントリがデッドラインを設定されている場合、それをスキップします。

(org-agenda-skip-entry-if 'scheduled 'deadline)
  現在のエントリがデッドラインを設定されているかもしくはスケジューリングされてい
  る場合、それをスキップします

(org-agenda-skip-entry-if 'todo '("TODO" "WAITING"))
  現在のエントリの TODO のキーワードが TODO もしくは WAITING の場合、それ
  をスキップします。

(org-agenda-skip-entry-if 'todo 'done)
  現在のエントリの TODO のキーワードが DONE とマークされている場合、それをス
  キップします。

(org-agenda-skip-entry-if 'timestamp)
  現在のエントリが何かしらのタイムスタンプを持っている場合、それをスキップします。
  タイムスタンプはデッドラインかスケジューリングが考えられます。

(org-agenda-skip-entry-if 'regexp "regular expression")
  現在のエントリの中で正規表現「regular expression」にマッチした場合、それをスキ
  ップします。

(org-agenda-skip-entry-if 'notregexp "regular expression")
  現在のエントリの中で正規表現「regular expression」にマッチしない場合、それをス
  キップします。

(org-agenda-skip-subtree-if 'regexp "regular expression")
  2つ上と同じですが、サブツリー全体をチェックしてスキップします。
```

The following is an example of a search for ‘WAITING’ without the special function:

```
(org-add-agenda-custom-command
  '("b" todo "PROJECT"
    ((org-agenda-skip-function '(org-agenda-skip-subtree-if
                                'regexp ":waiting:"))
      (org-agenda-overriding-header "waiting (待ち) 状態のプロジェクト: "))))
```

<sup>4</sup> Note that, for `org-odd-levels-only`, a level number corresponds to order in the hierarchy, not to the number of stars.

## A.9 Speeding up your agendas

Some agenda commands slow down when the Org files grow in size or number. Here are tips to speed up:

1. Reduce the number of Org agenda files to avoid slowdowns due to hard drive accesses.
2. Reduce the number of ‘DONE’ and archived headlines so agenda operations that skip over these can finish faster.
3. Do not dim blocked tasks:

```
(setq org-agenda-dim-blocked-tasks nil)
```

4. Stop preparing agenda buffers on startup:

```
(setq org-agenda-inhibit-startup nil)
```

5. Disable tag inheritance for agendas:

```
(setq org-agenda-use-tag-inheritance nil)
```

These options can be applied to selected agenda views. For more details about generation of agenda views, see the docstrings for the relevant variables, and this dedicated Worg page (<https://orgmode.org/worg/agenda-optimization.html>) for agenda optimization.

## A.10 Extracting agenda information

Org provides commands to access agendas through Emacs batch mode. Through this command-line interface, agendas are automated for further processing or printing.

`org-batch-agenda` creates an agenda view in ASCII and outputs to STDOUT. This command takes one string parameter. When string length=1, Org uses it as a key to `org-agenda-custom-commands`. These are the same ones available through `C-c a`.

This example command line directly prints the TODO list to the printer:

```
emacs -batch -l ~/.emacs -eval '(org-batch-agenda "t")' | lpr
```

When the string parameter length is two or more characters, Org matches it with tags/TODO strings. For example, this example command line prints items tagged with ‘shop’, but excludes items tagged with ‘NewYork’:

```
emacs -batch -l ~/.emacs \
  -eval '(org-batch-agenda "+shop-NewYork")' | lpr
```

An example showing on-the-fly parameter modifications:

```
emacs -batch -l ~/.emacs \
  -eval '(org-batch-agenda "a" \
    org-agenda-span (quote month) \
    org-agenda-include-diary nil \
    org-agenda-files (quote ("~/org/project.org")))' \
  | lpr
```

which will produce an agenda for the next 30 days from just the `~/org/projects.org` file.

For structured processing of agenda output, use `org-batch-agenda-csv` with the following fields:

|                       |   |
|-----------------------|---|
| <code>category</code> | The category of the item                              |
| <code>head</code>     | The headline, without TODO keyword, TAGS and PRIORITY |

|                   |  |                                     |
|-------------------|--|-------------------------------------|
| <b>type</b>       | The type of the agenda entry, can be                   |                                     |
|                   | <b>todo</b>  | selected in TODO match              |
|                   | <b>tagsmatch</b>                                       | selected in tags match              |
|                   | <b>diary</b>   | imported from diary                 |
|                   | <b>deadline</b>  | a deadline                          |
|                   | <b>scheduled</b>                                       | scheduled                           |
|                   | <b>timestamp</b>                                       | appointment, selected by timestamp  |
|                   | <b>closed</b>  | entry was closed on date            |
|                   | <b>upcoming-deadline</b>                               | warning about nearing deadline      |
|                   | <b>past-scheduled</b>                                  | forwarded scheduled item            |
|                   | <b>block</b>   | entry has date block including date |
| <b>todo</b>       | The TODO keyword, if any                               |                                     |
| <b>tags</b>       | All tags including inherited ones, separated by colons |                                     |
| <b>date</b>       | The relevant date, like 2007-2-14                      |                                     |
| <b>time</b>       | The time, like 15:00-16:50                             |                                     |
| <b>extra</b>      | String with extra planning info                        |                                     |
| <b>priority-l</b> | The priority letter if any was given                   |                                     |
| <b>priority-n</b> | The computed numerical priority                        |                                     |

If the selection of the agenda item was based on a timestamp, including those items with DEADLINE and SCHEDULED keywords, then Org includes date and time in the output.

If the selection of the agenda item was based on a timestamp (or deadline/scheduled), then Org includes date and time in the output.

Here is an example of a post-processing script in Perl. It takes the CSV output from Emacs and prints with a checkbox:

```
#!/usr/bin/perl

# 実行する Emacs のコマンドを定義します
$cmd = "emacs -batch -l ~/.emacs -eval '(org-batch-agenda-csv \"t\")'";

# 実行してアウトプットを取得します
$agenda = qx{$cmd 2>/dev/null};

# 全ての行についてループします
foreach $line (split(/\n/, $agenda)) {
  # それぞれの値を取得します
  ($category, $head, $type, $todo, $tags, $date, $time, $extra,
   $priority_l, $priority_n) = split(/,/, $line);
  # 処理して出力します
  print "[ ] $head\n";
}
```

## A.11 Using the property API

Functions for working with properties.

- org-entry-properties** *&optional pom which* [Function]  
 Get all properties of the entry at point-or-marker POM.  
 This includes the TODO keyword, the tags, time strings for deadline, scheduled, and clocking, and any additional properties defined in the entry. The return value is an alist. Keys may occur multiple times if the property key was used several times.  
 POM may also be `nil`, in which case the current entry is used. If `WHICH` is `nil` or `all`, get all properties. If `WHICH` is `special` or `standard`, only get that subclass.
- org-entry-get** *pom property &optional inherit* [Function]  
 Get value of `PROPERTY` for entry at point-or-marker POM. By default, this only looks at properties defined locally in the entry. If `INHERIT` is non-`nil` and the entry does not have the property, then also check higher levels of the hierarchy. If `INHERIT` is the symbol `selective`, use inheritance if and only if the setting of `org-use-property-inheritance` selects `PROPERTY` for inheritance.
- org-entry-delete** *pom property* [Function]  
 Delete the property `PROPERTY` from entry at point-or-marker POM.
- org-entry-put** *pom property value* [Function]  
 Set `PROPERTY` to `VALUE` for entry at point-or-marker POM.
- org-buffer-property-keys** *&optional include-specials* [Function]  
 カレントバッファにある全てのプロパティのキーを取得します。
- org-insert-property-drawer** [Function]  
 Insert a property drawer for the current entry.
- org-entry-put-multivalued-property** *pom property &rest values* [Function]  
 Set `PROPERTY` at point-or-marker POM to `VALUES`. `VALUES` should be a list of strings. They will be concatenated, with spaces as separators.
- org-entry-get-multivalued-property** *pom property* [Function]  
 Treat the value of the property `PROPERTY` as a whitespace-separated list of values and return the values as a list of strings.
- org-entry-add-to-multivalued-property** *pom property value* [Function]  
 Treat the value of the property `PROPERTY` as a whitespace-separated list of values and make sure that `VALUE` is in this list.
- org-entry-remove-from-multivalued-property** *pom property value* [Function]  
 Treat the value of the property `PROPERTY` as a whitespace-separated list of values and make sure that `VALUE` is *not* in this list.
- org-entry-member-in-multivalued-property** *pom property value* [Function]  
 Treat the value of the property `PROPERTY` as a whitespace-separated list of values and check if `VALUE` is in this list.
- org-property-allowed-value-functions** [User Option]  
 特定のプロパティに許可された値を提供する関数へのフックです。その関数はプロパティの名前を単一の引数として受け取り、許可された値の単純なリストを返します。‘:ETC’がその値の1つである場合、その値を補完の候補として使いますが、他の値も入力することができます。もし関数がプロパティと関係無い場合には、関数は `nil` を返します。



## A.12 Using the mapping API

Org has sophisticated mapping capabilities for finding entries. Org uses this functionality internally for generating agenda views. Org also exposes an API for executing arbitrary functions for each selected entry. The API's main entry point is:

**org-map-entries** *func* **&optional** *match scope* **&rest** *skip* [Function]

Call 'FUNC' at each headline selected by MATCH in SCOPE.

'FUNC' is a function or a Lisp form. With the cursor positioned at the beginning of the headline, call the function without arguments. Org returns an alist of return values of calls to the function.

To avoid preserving point, Org wraps the call to FUNC in save-excursion form. After evaluation, Org moves the cursor to the end of the line that was just processed. Search continues from that point forward. This may not always work as expected under some conditions, such as if the current sub-tree was removed by a previous archiving operation. In such rare circumstances, Org skips the next entry entirely when it should not. To stop Org from such skips, make 'FUNC' set the variable **org-map-continue-from** to a specific buffer position.

'MATCH' is a tags/property/TODO match. Org iterates only matched headlines. Org iterates over all headlines when MATCH is nil or t.

'SCOPE' determines the scope of this command. It can be any of:

|                             |  |
|-----------------------------|--|
| <b>nil</b>                  | the current buffer, respecting the restriction if any        |
| <b>tree</b>                 | the subtree started with the entry at point                  |
| <b>region</b>               | The entries within the active region, if any                 |
| <b>file</b>                 | the current buffer, without restriction                      |
| <b>file-with-archives</b>   | the current buffer, and any archives associated with it      |
| <b>agenda</b>               | all agenda files   |
| <b>agenda-with-archives</b> | all agenda files with any archive files associated with them |
| <b>(file1 file2 ...)</b>    | if this is a list, all files in the list will be scanned     |

The remaining args are treated as settings for the scanner's skipping facilities. Valid args are:

|                                     |   |
|-------------------------------------|---|
| <b>archive</b>                      | アーカイブタグ (archive tag) があるツリーをスキップします  |
| <b>comment</b>                      | キーワード COMMENT があるツリーをスキップします  |
| <b>function</b> or <b>Lisp form</b> | org-agenda-skip-functionの値として使われます。<br>関数であればいつも t を返します、<br>FUNC は呼び出されません、<br>関数がポイントを置いたままにした位置から検索は続きます |

The mapping routine can call any arbitrary function, even functions that change meta data or query the property API (see Section A.11 [Using the property API], page 252). Here are some handy functions:

**org-todo** **&optional** *arg* [Function]  
 Change the TODO state of the entry. See the docstring of the functions for the many possible values for the argument **ARG**.

**org-priority** **&optional** *action* [Function]  
 Change the priority of the entry. See the docstring of this function for the possible values for **ACTION**.

**org-toggle-tag** *tag* **&optional** *onoff* [Function]  
 Toggle the tag **TAG** in the current entry. Setting **ONOFF** to either **on** or **off** will not toggle tag, but ensure that it is either on or off.

**org-promote** [Function]  
 現在のエントリを一階層上に引き上げます。

**org-demote** [Function]  
 現在のエントリを一階層下へ引き下げます。

This example turns all entries tagged with **TOMORROW** into **TODO** entries with keyword **UPCOMING**. Org ignores entries in comment trees and archive trees.

```
(org-map-entries
 '(org-todo "UPCOMING")
 "+TOMORROW" 'file 'archive 'comment)
```

以下の例では、全てのアジェンダファイルに渡って、**TODO** のキーワードに **WAITING** があるエントリの数をカウントします。

```
(length (org-map-entries t "/+WAITING" 'agenda))
```

## Appendix B MobileOrg

MobileOrg is a companion mobile app that runs on iOS and Android devices. MobileOrg enables offline-views and capture support for an Org mode system that is rooted on a “real” computer. MobileOrg can record changes to existing entries.

The iOS implementation (<https://github.com/MobileOrg/>) for the *iPhone/iPod Touch/iPad* series of devices, was started by Richard Moreland and is now in the hands Sean Escriva. Android users should check out MobileOrg Android (<http://wiki.github.com/matburt/mobileorg-android/>) by Matt Jones. Though the two implementations are not identical, they offer similar features.

This appendix describes Org’s support for agenda view formats compatible with MobileOrg. It also describes synchronizing changes, such as to notes, between MobileOrg and the computer.

To change tags and TODO states in MobileOrg, first customize the variables `org-todo-keywords` and `org-tag-alist`. These should cover all the important tags and TODO keywords, even if Org files use only some of them. Though MobileOrg has in-buffer settings, it understands TODO states *sets* (see Section 5.2.5 [Per-file keywords], page 49) and *mutually exclusive* tags (see Section 6.2 [Setting tags], page 59) only for those set in these variables.

### B.1 Setting up the staging area

MobileOrg needs access to a file directory on a server to interact with Emacs. With a public server, consider encrypting the files. MobileOrg version 1.5 supports encryption for the iPhone. Org also requires `openssl` installed on the local computer. To turn on encryption, set the same password in MobileOrg and in Emacs. Set the password in the variable `org-mobile-use-encryption`<sup>1</sup>. Note that even after MobileOrg encrypts the file contents, the file names will remain visible on the file systems of the local computer, the server, and the mobile device.

For a server to host files, consider options like Dropbox.com (<http://dropbox.com>) account<sup>2</sup>. On first connection, MobileOrg creates a directory `MobileOrg/` on Dropbox. Pass its location to Emacs through an init file variable as follows:

```
(setq org-mobile-directory "~/Dropbox/MobileOrg")
```

Org copies files to the above directory for MobileOrg. Org also uses the same directory for sharing notes between Org and MobileOrg.

### B.2 Pushing to MobileOrg

Org pushes files listed in `org-mobile-files` to `org-mobile-directory`. Files include agenda files (as listed in `org-agenda-files`). Customize `org-mobile-files` to add other files. File names will be staged with paths relative to `org-directory`, so all files should be inside this directory<sup>3</sup>.

<sup>1</sup> If Emacs is configured for safe storing of passwords, then configure the variable, `org-mobile-encryption-password`; please read the docstring of that variable.

<sup>2</sup> An alternative is to use webdav server. MobileOrg documentation has details of webdav server configuration. Additional help is at FAQ entry (<https://orgmode.org/worg/org-faq.html#mobileorg-webdav>).

<sup>3</sup> Symbolic links in `org-directory` should have the same name as their targets.

Push creates a special Org file `agendas.org` with custom agenda views defined by the user<sup>4</sup>.

Org writes the file `index.org`, containing links to other files. MobileOrg reads this file first from the server to determine what other files to download for agendas. For faster downloads, MobileOrg will read only those files whose checksums<sup>5</sup> have changed.

## B.3 Pulling from MobileOrg

When MobileOrg synchronizes with the server, it pulls the Org files for viewing. It then appends to the file `mobileorg.org` on the server the captured entries, pointers to flagged and changed entries. Org integrates its data in an inbox file format.

1. Org moves all entries found in `mobileorg.org`<sup>6</sup> and appends them to the file pointed to by the variable `org-mobile-inbox-for-pull`. Each captured entry and each editing event is a top-level entry in the inbox file.
2. After moving the entries, Org attempts changes to MobileOrg. Some changes are applied directly and without user interaction. Examples include changes to tags, TODO state, headline and body text. Entries for further action are tagged as `:FLAGGED:`. Org marks entries with problems with an error message in the inbox. They have to be resolved manually.
3. Org generates an agenda view for flagged entries for user intervention to clean up. For notes stored in flagged entries, MobileOrg displays them in the echo area when the cursor is on the corresponding agenda item.

?            Pressing ? displays the entire flagged note in another window. Org also pushes it to the kill ring. To store flagged note as a normal note, use ? z C-y C-c C-c. Pressing ? twice does these things: first it removes the `:FLAGGED:` tag; second, it removes the flagged note from the property drawer; third, it signals that manual editing of the flagged entry is now finished.

C-c a ? returns to the agenda view to finish processing flagged entries. Note that these entries may not be the most recent since MobileOrg searches files that were last pulled. To get an updated agenda view with changes since the last pull, pull again.

---

<sup>4</sup> While creating the agendas, Org mode will force ID properties on all referenced entries, so that these entries can be uniquely identified if MobileOrg flags them for further action. To avoid setting properties configure the variable `org-mobile-force-id-on-agenda-items` to `nil`. Org mode will then rely on outline paths, assuming they are unique.

<sup>5</sup> Checksums are stored automatically in the file `checksums.dat`.

<sup>6</sup> `mobileorg.org` will be empty after this operation.

## Appendix C History and acknowledgments

### C.1 From Carsten

Org was born in 2003, out of frustration over the user interface of the Emacs Outline mode. I was trying to organize my notes and projects, and using Emacs seemed to be the natural way to go. However, having to remember eleven different commands with two or three keys per command, only to hide and show parts of the outline tree, that seemed entirely unacceptable. Also, when using outlines to take notes, I constantly wanted to restructure the tree, organizing it paralleling my thoughts and plans. *Visibility cycling* and *structure editing* were originally implemented in the package `outline-magic.el`, but quickly moved to the more general `org.el`. As this environment became comfortable for project planning, the next step was adding *TODO entries*, basic *timestamps*, and *table support*. These areas highlighted the two main goals that Org still has today: to be a new, outline-based, plain text mode with innovative and intuitive editing features, and to incorporate project planning functionality directly into a notes file.

Since the first release, literally thousands of emails to me or to `emacs-orgmode@gnu.org` have provided a constant stream of bug reports, feedback, new ideas, and sometimes patches and add-on code. Many thanks to everyone who has helped to improve this package. I am trying to keep here a list of the people who had significant influence in shaping one or more aspects of Org. The list may not be complete, if I have forgotten someone, please accept my apologies and let me know.

Before I get to this list, a few special mentions are in order:

*Bastien Guerry*

Bastien has written a large number of extensions to Org (most of them integrated into the core by now), including the  $\text{\LaTeX}$  exporter and the plain list parser. His support during the early days was central to the success of this project. Bastien also invented Worg, helped establishing the Web presence of Org, and sponsored hosting costs for the `orgmode.org` website. Bastien stepped in as maintainer of Org between 2011 and 2013, at a time when I desperately needed a break.

*Eric Schulte and Dan Davison*

Eric and Dan are jointly responsible for the Org-babel system, which turns Org into a multi-language environment for evaluating code and doing literate programming and reproducible research. This has become one of Org's killer features that define what Org is today.

*John Wiegley*

John has contributed a number of great ideas and patches directly to Org, including the attachment system (`org-attach.el`), integration with Apple Mail (`org-mac-message.el`), hierarchical dependencies of TODO items, habit tracking (`org-habits.el`), and encryption (`org-crypt.el`). Also, the capture system is really an extended copy of his great `remember.el`.

*Sebastian Rose*

Without Sebastian, the HTML/XHTML publishing of Org would be the pitiful work of an ignorant amateur. Sebastian has pushed this part of Org onto a

much higher level. He also wrote `org-info.js`, a Java script for displaying web pages derived from Org using an Info-like or a folding interface with single-key navigation.

See below for the full list of contributions! Again, please let me know what I am missing here!

## C.2 From Bastien

I (Bastien) have been maintaining Org between 2011 and 2013. This appendix would not be complete without adding a few more acknowledgments and thanks.

I am first grateful to Carsten for his trust while handing me over the maintainership of Org. His unremitting support is what really helped me getting more confident over time, with both the community and the code.

When I took over maintainership, I knew I would have to make Org more collaborative than ever, as I would have to rely on people that are more knowledgeable than I am on many parts of the code. Here is a list of the persons I could rely on, they should really be considered co-maintainers, either of the code or the community:

*Eric Schulte*

Eric is maintaining the Babel parts of Org. His reactivity here kept me away from worrying about possible bugs here and let me focus on other parts.

*Nicolas Goaziou*

Nicolas is maintaining the consistency of the deepest parts of Org. His work on `org-element.el` and `ox.el` has been outstanding, and it opened the doors for many new ideas and features. He rewrote many of the old exporters to use the new export engine, and helped with documenting this major change. More importantly (if that's possible), he has been more than reliable during all the work done for Org 8.0, and always very reactive on the mailing list.

*Achim Gratz*

Achim rewrote the building process of Org, turning some *ad hoc* tools into a flexible and conceptually clean process. He patiently coped with the many hiccups that such a change can create for users.

*Nick Dokos*

The Org mode mailing list would not be such a nice place without Nick, who patiently helped users so many times. It is impossible to overestimate such a great help, and the list would not be so active without him.

I received support from so many users that it is clearly impossible to be fair when shortlisting a few of them, but Org's history would not be complete if the ones above were not mentioned in this manual.

## C.3 List of contributions

- *Russel Adams* came up with the idea for drawers.
- *Suvayu Ali* has steadily helped on the mailing list, providing useful feedback on many features and several patches.

- *Luis Anaya* wrote `ox-man.el`.
- *Thomas Baumann* wrote `org-bbdb.el` and `org-mhe.el`.
- *Michael Brand* helped by reporting many bugs and testing many features. He also implemented the distinction between empty fields and 0-value fields in Org's spreadsheets.
- *Christophe Bataillon* created the great unicorn logo that we use on the Org mode website.
- *Alex Bochanek* provided a patch for rounding timestamps.
- *Jan Bock* wrote `org-docview.el`.
- *Brad Bozarth* showed how to pull RSS feed data into Org mode files.
- *Tom Breton* wrote `org-choose.el`.
- *Charles Cave's* suggestion sparked the implementation of templates for Remember, which are now templates for capture.
- *Pavel Chalmoviansky* influenced the agenda treatment of items with specified time.
- *Gregory Chernov* patched support for Lisp forms into table calculations and improved XEmacs compatibility, in particular by porting `nouline.el` to XEmacs.
- *Sacha Chua* suggested copying some linking code from Planner, and helped make Org popular through her blog.
- *Toby S. Cubitt* contributed to the code for clock formats.
- *Baoqiu Cui* contributed the first DocBook exporter. In Org 8.0, we go a different route: you can now export to Texinfo and export the `.texi` file to DocBook using `makeinfo`.
- *Eddward DeVilla* proposed and tested checkbox statistics. He also came up with the idea of properties, and that there should be an API for them.
- *Nick Dokos* tracked down several nasty bugs.
- *Kees Dullemond* used to edit projects lists directly in HTML and so inspired some of the early development, including HTML export. He also asked for a way to narrow wide table columns.
- *Jason Dunsmore* has been maintaining the Org-Mode server at Rackspace for several years now. He also sponsored the hosting costs until Rackspace started to host us for free.
- *Thomas S. Dye* contributed documentation on Worg and helped integrating the Org-Babel documentation into the manual.
- *Christian Egli* converted the documentation into Texinfo format, inspired the agenda, patched CSS formatting into the HTML exporter, and wrote `org-taskjuggler.el`, which has been rewritten by Nicolas Goaziou as `ox-taskjuggler.el` for Org 8.0.
- *David Emery* provided a patch for custom CSS support in exported HTML agendas.
- *Sean Escriva* took over MobileOrg development on the iPhone platform.
- *Nic Ferrier* contributed mailcap and XOXO support.
- *Miguel A. Figueroa-Villanueva* implemented hierarchical checkboxes.
- *John Foerch* figured out how to make incremental search show context around a match in a hidden outline tree.

- *Raimar Finken* wrote `org-git-line.el`.
- *Mikael Fornius* works as a mailing list moderator.
- *Austin Frank* works as a mailing list moderator.
- *Eric Fraga* drove the development of BEAMER export with ideas and testing.
- *Barry Gidden* did proofreading the manual in preparation for the book publication through Network Theory Ltd.
- *Niels Giesen* had the idea to automatically archive DONE trees.
- *Nicolas Goaziou* rewrote much of the plain list code. He also wrote `org-element.el` and `org-export.el`, which was a huge step forward in implementing a clean framework for Org exporters.
- *Kai Grossjohann* pointed out key-binding conflicts with other packages.
- *Brian Gough* of Network Theory Ltd publishes the Org mode manual as a book.
- *Bernt Hansen* has driven much of the support for auto-repeating tasks, task state change logging, and the clocktable. His clear explanations have been critical when we started to adopt the Git version control system.
- *Manuel Hermenegildo* has contributed various ideas, small fixes and patches.
- *Phil Jackson* wrote `org-irc.el`.
- *Scott Jaderholm* proposed footnotes, control over whitespace between folded entries, and column view for properties.
- *Matt Jones* wrote *MobileOrg Android*.
- *Tokuya Kameshima* wrote `org-wl.el` and `org-mew.el`.
- *Jonathan Leech-Pepin* wrote `ox-texinfo.el`.
- *Shidai Liu* ("Leo") asked for embedded L<sup>A</sup>T<sub>E</sub>X and tested it. He also provided frequent feedback and some patches.
- *Matt Lundin* has proposed last-row references for table formulas and named invisible anchors. He has also worked a lot on the FAQ.
- *David Maus* wrote `org-atom.el`, maintains the issues file for Org, and is a prolific contributor on the mailing list with competent replies, small fixes and patches.
- *Jason F. McBrayer* suggested agenda export to CSV format.
- *Max Mikhanosha* came up with the idea of refiling and sticky agendas.
- *Dmitri Minaev* sent a patch to set priority limits on a per-file basis.
- *Stefan Monnier* provided a patch to keep the Emacs-Lisp compiler happy.
- *Richard Moreland* wrote MobileOrg for the iPhone.
- *Rick Moynihan* proposed allowing multiple TODO sequences in a file and being able to quickly restrict the agenda to a subtree.
- *Todd Neal* provided patches for links to Info files and Elisp forms.
- *Greg Newman* refreshed the unicorn logo into its current form.
- *Tim O'Callaghan* suggested in-file links, search options for general file links, and TAGS.
- *Osamu Okano* wrote `orgcard2ref.pl`, a Perl program to create a text version of the reference card.
- *Takeshi Okano* translated the manual and David O'Toole's tutorial into Japanese.



- *Oliver Oppitz* suggested multi-state TODO items.
- *Scott Otterson* sparked the introduction of descriptive text for links, among other things.
- *Pete Phillips* helped during the development of the TAGS feature, and provided frequent feedback.
- *Francesco Pizzolante* provided patches that helped speeding up the agenda generation.
- *Martin Pohlack* provided the code snippet to bundle character insertion into bundles of 20 for undo.
- *Rackspace.com* is hosting our website for free. Thank you Rackspace!
- *T.V. Raman* reported bugs and suggested improvements.
- *Matthias Rempé* (Oelde) provided ideas, Windows support, and quality control.
- *Paul Rivier* provided the basic implementation of named footnotes. He also acted as mailing list moderator for some time.
- *Kevin Rogers* contributed code to access VM files on remote hosts.
- *Frank Ruell* solved the mystery of the `keymapp nil` bug, a conflict with `allout.el`.
- *Jason Riedy* generalized the send-receive mechanism for Orgtbl tables with extensive patches.
- *Philip Rooke* created the Org reference card, provided lots of feedback, developed and applied standards to the Org documentation.
- *Christian Schlauer* proposed angular brackets around links, among other things.
- *Christopher Schmidt* reworked `orgstruct-mode` so that users can enjoy folding in non-org buffers by using Org headlines in comments.
- *Paul Sexton* wrote `org-ctags.el`.
- Linking to VM/BBDB/Gnus was first inspired by *Tom Shannon's* `organizer-mode.el`.
- *Ilya Shlyakhter* proposed the Archive Sibling, line numbering in literal examples, and remote highlighting for referenced code lines.
- *Stathis Sideris* wrote the `ditaa.jar` ASCII to PNG converter that is now packaged into Org's `contrib` directory.
- *Daniel Sinder* came up with the idea of internal archiving by locking subtrees.
- *Dale Smith* proposed link abbreviations.
- *James TD Smith* has contributed a large number of patches for useful tweaks and features.
- *Adam Spiers* asked for global linking commands, inspired the link extension system, added support for mairix, and proposed the mapping API.
- *Ulf Stegemann* created the table to translate special symbols to HTML, L<sup>A</sup>T<sub>E</sub>X, UTF-8, Latin-1 and ASCII.
- *Andy Stewart* contributed code to `org-w3m.el`, to copy HTML content with links transformation to Org syntax.
- *David O'Toole* wrote `org-publish.el` and drafted the manual chapter about publishing.
- *Jambunathan K* contributed the ODT exporter and rewrote the HTML exporter.

- *Sebastien Vauban* reported many issues with L<sup>A</sup>T<sub>E</sub>X and BEAMER export and enabled source code highlighting in Gnus.
- *Stefan Vollmar* organized a video-recorded talk at the Max-Planck-Institute for Neurology. He also inspired the creation of a concept index for HTML export.
- *J<sup>c3</sup>bc rgen Vollmer* contributed code generating the table of contents in HTML output.
- *Samuel Wales* has provided important feedback and bug reports.
- *Chris Wallace* provided a patch implementing the ‘QUOTE’ keyword.
- *David Wainberg* suggested archiving, and improvements to the linking system.
- *Carsten Wimmer* suggested some changes and helped fix a bug in linking to Gnus.
- *Roland Winkler* requested additional key bindings to make Org work on a tty.
- *Piotr Zielinski* wrote `org-mouse.el`, proposed agenda blocks and contributed various ideas and code snippets.
- *Marco Wahl* wrote `org-eww.el`.

## Appendix D GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008, 2013, 2014, 2018 Free Software Foundation, Inc.  
<http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

### 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at

your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.



## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

## 11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

## Concept index

|             |     |                |         |
|-------------|-----|----------------|---------|
| タ           |     | キ              |         |
| タグ .....    | 59  | キャプチャ .....    | 89      |
| 目           |     | タ              |         |
| 目次 .....    | 142 | タグ、設定 .....    | 59      |
| フ           |     | ハ              |         |
| フック .....   | 242 | ハッキング .....    | 242     |
| 数           |     | コ              |         |
| 数学記号 .....  | 134 | コメント行 .....    | 145     |
| 変           |     | ラ              |         |
| 変換関数 .....  | 247 | ラジオリスト .....   | 247     |
| 特           |     | 完              |         |
| 特殊記号 .....  | 134 | 完了、タグの .....   | 59, 229 |
| 下           |     | エ              |         |
| 下付き文字 ..... | 135 | エクスポート .....   | 138     |
| タ           |     | 継              |         |
| タグの継承 ..... | 59  | 継承、タグの .....   | 59      |
| 日           |     | 動              |         |
| 日記の統合 ..... | 105 | 動的なブロック .....  | 248     |
| 検           |     | ラ              |         |
| 検索ビュー ..... | 110 | ラジオテーブル .....  | 244     |
| タ           |     | マ              |         |
| タグの検索 ..... | 63  | マッチング、タグ ..... | 59      |
| タグの設定 ..... | 59  | ア              |         |
| ア           |     | アジェンダビュー ..... | 102     |
| アジェンダ ..... | 104 | タ              |         |
| 上           |     | タグのための検索 ..... | 63      |
| 上付き文字 ..... | 135 | 見              |         |
|             |     | 見出しのタグ付け ..... | 59      |

|                   |     |                       |     |
|-------------------|-----|-----------------------|-----|
| カ                 |     | キ                     |     |
| カレンダーの統合 .....    | 105 | キャプチャテンプレート .....     | 90  |
| ア                 |     | ア                     |     |
| アジェンダファイル .....   | 102 | アジェンダ用のファイル .....     | 102 |
| ソ                 |     | ラ                     |     |
| ソースコード、編集 .....   | 201 | ライブラリ、ソースコード .....    | 205 |
| 抽                 |     | ソ                     |     |
| 抽出、ソースコード .....   | 203 | ソースコード、バッチ処理 .....    | 228 |
| ソ                 |     | ソースコード、評価の結果 .....    | 224 |
| ソースコード、言語 .....   | 205 | 表                     |     |
| ア                 |     | 表、マークアップのルール .....    | 132 |
| アドオンパッケージ .....   | 242 | ラ                     |     |
| 評                 |     | ライブラリ、コードブロック .....   | 205 |
| 評価、ソースコード .....   | 204 | 段                     |     |
| ア                 |     | 段落、マークアップのルール .....   | 131 |
| アジェンダ、パイプ .....   | 251 | コ                     |     |
| 評                 |     | コードブロック、バッチ処理 .....   | 228 |
| 評価、コードブロック .....  | 204 | ソ                     |     |
| リ                 |     | ソースコード、エクスポート .....   | 202 |
| リスト、他のモードで .....  | 244 | ソースコード、ブロック構造 .....   | 200 |
| コ                 |     | コ                     |     |
| コードブロック、編集 .....  | 201 | コードブロック、評価の結果 .....   | 224 |
| コードブロック、言語 .....  | 205 | コードブロック、ヘッダー引数 .....  | 206 |
| コードブロック、構造 .....  | 200 | コードブロック、エクスポート .....  | 202 |
| エ                 |     | ア                     |     |
| エクスポート、されない ..... | 145 | アジェンダのコマンド選択画面 .....  | 103 |
| エラーのバックトレース ..... | 4   | ツ                     |     |
| テ                 |     | ツリーの抽出、タグに基づいた .....  | 59  |
| テーブル、他のモードで ..... | 244 | ハ                     |     |
| ソ                 |     | ハイパーリンク、タイプの追加 .....  | 242 |
| ソースコード、連携する ..... | 199 | ア                     |     |
|                   |     | アジェンダビュー、ユーザー定義 ..... | 249 |

- 文  
文脈に依存するコマンド、フック ..... 244
- ア  
アジェンダのコマンドを選択する ..... 103  
アジェンダビュー、エクスポート ..... 123, 127  
アドオン、文脈に依存するコマンド ..... 244
- リ  
リテラルの例、マークアップのルール ..... 132
- 下  
下位レベル、マッチしたタグへの包含 ..... 59
- マ  
マクロによる置換、エクスポート中に ..... 144
- ソ  
ソースコードの抽出、コードブロック ..... 203  
ソースコード、ブロックのヘッダー引数 .... 206
- イ  
インライン画像、マークアップのルール .... 132
- フ  
ファイル、アジェンダリストに追加する .... 102
- コ  
コードのテキスト、マークアップのルール.. 131
- 太  
太字のテキスト、マークアップのルール.. 131
- 下  
下線のあるテキスト、マークアップのルール  
..... 131
- 文  
文字通りのテキスト、マークアップのルール  
..... 131
- 取  
取り消し線付きテキスト、マークアップのルー  
ル ..... 131
- ソ  
ソースコードのフォーマット、マークアップの  
ルール ..... 132
- エ  
エントリのマッピング、API ..... 254
- コ  
コードブロック、キーバインディング ..... 227
- テ  
テキストエリア、HTML の中の ..... 155
- コ  
コードブロック、noweb リファレンス ..... 226
- ソ  
ソースコード、noweb リファレンス ..... 226
- ラ  
ライブラリ、babel ..... 205
- 言  
言語、babel ..... 205
- 構  
構文、noweb ..... 226
- #  
#+ARCHIVE ..... 100  
#+ASCII ..... 146  
#+ATTR\_ASCII ..... 146  
#+ATTR\_BEAMER ..... 149  
#+ATTR\_HTML ..... 154, 155  
#+ATTR\_LATEX, in example blocks ..... 164  
#+ATTR\_LATEX, in horizontal rules ..... 165  
#+ATTR\_LATEX, in images ..... 162  
#+ATTR\_LATEX, in plain lists ..... 163  
#+ATTR\_LATEX, in source blocks ..... 164  
#+ATTR\_LATEX, in special blocks ..... 165  
#+ATTR\_LATEX, in tables ..... 161  
#+ATTR\_ODT ..... 169, 170, 174  
#+ATTR\_TEXINFO, in images ..... 181  
#+ATTR\_TEXINFO, in plain lists ..... 180  
#+ATTR\_TEXINFO, in special blocks ..... 181  
#+ATTR\_TEXINFO, in tables ..... 181  
#+AUTHOR ..... 139  
#+BEAMER ..... 149  
#+BEAMER\_FONT\_THEME ..... 147

|  |               |   |              |
|--|---------------|---|--------------|
| <code>#+BEAMER_HEADER</code> .....                             | 147           | <code>#+LATEX_CLASS</code> .....                            | 159, 160     |
| <code>#+BEAMER_INNER_THEME</code> .....                        | 147           | <code>#+LATEX_CLASS_OPTIONS</code> .....                    | 159, 160     |
| <code>#+BEAMER_OUTER_THEME</code> .....                        | 147           | <code>#+LATEX_COMPILER</code> .....                         | 159          |
| <code>#+BEAMER_THEME</code> .....                              | 147           | <code>#+LATEX_HEADER</code> .....                           | 159, 160     |
| <code>#+BEGIN, clocktable</code> .....                         | 83            | <code>#+LATEX_HEADER (HTML)</code> .....                    | 151          |
| <code>#+BEGIN, columnview</code> .....                         | 71            | <code>#+LATEX_HEADER_EXTRA</code> .....                     | 159, 160     |
| <code>#+BEGIN, dynamic block</code> .....                      | 248           | <code>#+LINK</code> .....                                   | 44           |
| <code>#+BEGIN_CENTER</code> .....                              | 131           | <code>#+MACRO</code> .....                                  | 144          |
| <code>#+BEGIN_COMMENT</code> .....                             | 145           | <code>#+NAME</code> .....                                   | 38, 132, 200 |
| <code>#+BEGIN_EXAMPLE</code> .....                             | 132           | <code>#+NAME, for table</code> .....                        | 26           |
| <code>#+BEGIN_EXPORT ascii</code> .....                        | 146           | <code>#+ODT_STYLES_FILE</code> .....                        | 168          |
| <code>#+BEGIN_EXPORT beamer</code> .....                       | 149           | <code>#+OPTIONS</code> .....                                | 139          |
| <code>#+BEGIN_EXPORT html</code> .....                         | 153           | <code>#+ORGLST</code> .....                                 | 247          |
| <code>#+BEGIN_EXPORT latex</code> .....                        | 161           | <code>#+ORGTBL</code> .....                                 | 245          |
| <code>#+BEGIN_EXPORT texinfo</code> .....                      | 180           | <code>#+ORGTBL, ‘SEND’</code> .....                         | 245          |
| <code>#+BEGIN_JUSTIFYLEFT</code> .....                         | 146           | <code>#+PINDEX</code> .....                                 | 180          |
| <code>#+BEGIN_JUSTIFYRIGHT</code> .....                        | 146           | <code>#+PLOT</code> .....                                   | 35           |
| <code>#+BEGIN_QUOTE</code> .....                               | 131           | <code>#+PRIORITIES</code> .....                             | 55           |
| <code>#+BEGIN_SRC</code> .....                                 | 133, 200      | <code>#+PROPERTY</code> .....                               | 65           |
| <code>#+BEGIN_VERSE</code> .....                               | 131           | <code>#+RESULTS</code> .....                                | 204          |
| <code>#+BIND</code> .....                                      | 142           | <code>#+SELECT_TAGS</code> .....                            | 140          |
| <code>#+CALL</code> .....                                      | 204           | <code>#+SEQ_TODO</code> .....                               | 49           |
| <code>#+CAPTION</code> .....                                   | 132, 154, 155 | <code>#+SETUPFILE</code> .....                              | 139, 232     |
| <code>#+CATEGORY</code> .....                                  | 111           | <code>#+STARTUP</code> .....                                | 232          |
| <code>#+CINDEX</code> .....                                    | 180           | <code>#+SUBAUTHOR</code> .....                              | 178          |
| <code>#+COLUMNS</code> .....                                   | 68            | <code>#+SUBTITLE (ASCII)</code> .....                       | 146          |
| <code>#+CONSTANTS</code> .....                                 | 26            | <code>#+SUBTITLE (Beamer)</code> .....                      | 148          |
| <code>#+CREATOR</code> .....                                   | 140           | <code>#+SUBTITLE (HTML)</code> .....                        | 152          |
| <code>#+DATE</code> .....                                      | 140           | <code>#+SUBTITLE (L<sup>A</sup>T<sub>E</sub>X)</code> ..... | 160          |
| <code>#+DESCRIPTION (Beamer)</code> .....                      | 147           | <code>#+SUBTITLE (Texinfo)</code> .....                     | 178          |
| <code>#+DESCRIPTION (HTML)</code> .....                        | 151           | <code>#+TAGS</code> .....                                   | 60           |
| <code>#+DESCRIPTION (L<sup>A</sup>T<sub>E</sub>X)</code> ..... | 159           | <code>#+TBLFM</code> .....                                  | 29, 33, 234  |
| <code>#+DESCRIPTION (ODT)</code> .....                         | 167           | <code>#+TBLFM line, multiple</code> .....                   | 33           |
| <code>#+EMAIL</code> .....                                     | 140           | <code>#+TBLFM, switching</code> .....                       | 33           |
| <code>#+EXCLUDE_TAGS</code> .....                              | 140           | <code>#+TEXINFO</code> .....                                | 180          |
| <code>#+EXPORT_FILE_NAME</code> .....                          | 140           | <code>#+TEXINFO_CLASS</code> .....                          | 178, 179     |
| <code>#+FILETAGS</code> .....                                  | 59            | <code>#+TEXINFO_DIR_CATEGORY</code> .....                   | 178, 179     |
| <code>#+FINDEX</code> .....                                    | 180           | <code>#+TEXINFO_DIR_DESC</code> .....                       | 178, 179     |
| <code>#+HEADER:</code> .....                                   | 208           | <code>#+TEXINFO_DIR_TITLE</code> .....                      | 178, 179     |
| <code>#+HTML</code> .....                                      | 153           | <code>#+TEXINFO_FILENAME</code> .....                       | 178          |
| <code>#+HTML_CONTAINER</code> .....                            | 151           | <code>#+TEXINFO_HEADER</code> .....                         | 178          |
| <code>#+HTML_DOCTYPE</code> .....                              | 151           | <code>#+TEXINFO_POST_HEADER</code> .....                    | 178          |
| <code>#+HTML_HEAD</code> .....                                 | 151, 157      | <code>#+TEXINFO_PRINTED_TITLE</code> .....                  | 178          |
| <code>#+HTML_HEAD_EXTRA</code> .....                           | 151, 157      | <code>#+TINDEX</code> .....                                 | 180          |
| <code>#+HTML_INCLUDE_STYLE</code> .....                        | 157           | <code>#+TITLE</code> .....                                  | 140          |
| <code>#+HTML_LINK_HOME</code> .....                            | 151           | <code>#+TOC</code> .....                                    | 142          |
| <code>#+HTML_LINK_UP</code> .....                              | 151           | <code>#+TODO</code> .....                                   | 49           |
| <code>#+HTML_MATHJAX</code> .....                              | 151           | <code>#+TYP_TODO</code> .....                               | 49           |
| <code>#+INCLUDE</code> .....                                   | 143           | <code>#+VINDEX</code> .....                                 | 180          |
| <code>#+INDEX</code> .....                                     | 196           |   |              |
| <code>#+INFOJS_OPT</code> .....                                | 157           |   |              |
| <code>#+KEYWORDS (Beamer)</code> .....                         | 147           |   |              |
| <code>#+KEYWORDS (HTML)</code> .....                           | 151           |   |              |
| <code>#+KEYWORDS (L<sup>A</sup>T<sub>E</sub>X)</code> .....    | 159           |   |              |
| <code>#+KEYWORDS (ODT)</code> .....                            | 167           |   |              |
| <code>#+KINDEX</code> .....                                    | 180           |   |              |
| <code>#+LANGUAGE</code> .....                                  | 140           |   |              |
| <code>#+LATEX</code> .....                                     | 161           |   |              |

%

%:annotation template expansion in capture ... 98

%:description template expansion in capture ... 98

%:link template expansion in capture ... 98

**:**  
**:base-url** property in open-source protocol.... 98  
**:cache**, src header argument..... 206, 219  
**:colnames**, src header argument..... 221  
**:comments**, src header argument..... 216  
**:dir**, src header argument..... 215  
**:epilogue**, src header argument..... 224  
**:eval**, src header argument..... 223  
**:exports**, src header argument..... 202, 206, 215  
**:file**, src header argument..... 214  
**:file-ext**, src header argument..... 214  
**:hlines**, src header argument..... 220  
**:indic** attribute, in Texinfo export..... 180  
**:mkdirp**, src header argument..... 216  
**:no-expand**, src header argument..... 216  
**:noweb**, src header argument..... 206, 217  
**:noweb-ref**, src header argument..... 218  
**:noweb-sep**, src header argument..... 219  
**:online-suffix** property in open-source protocol  
..... 98  
**:output-dir**, src header argument..... 214  
**:padline**, src header argument..... 216  
**:post**, src header argument..... 223  
**:prologue**, src header argument..... 224  
**:results**, src header argument..... 206, 213, 225  
**:rewrites property** in open-source protocol... 98  
**:rownames**, src header argument..... 222  
**:sep** attribute, in Texinfo export..... 180  
**:sep**, src header argument..... 220  
**:session**, src header argument..... 206, 217  
**:shebang**, src header argument..... 222  
**:table-type** attribute, in Texinfo export..... 180  
**:tangle**, src header argument..... 203, 216  
**:tangle-mode**, src header argument..... 222  
**:var**, src header argument..... 209  
**:working-directory** property in open-source  
protocol..... 98  
**:working-suffix** property in open-source protocol  
..... 98  
**:wrap**, src header argument..... 223

## A

abbreviation, links..... 44  
abstract, in L<sup>A</sup>T<sub>E</sub>X export..... 165  
acknowledgments..... 258  
action, for publishing..... 189  
activation..... 3  
active region..... 11, 21, 166  
agenda files, removing buffers..... 123  
agenda views, custom..... 124  
agenda views, main example..... 124  
agenda views, optimization..... 251  
agenda\*, as an agenda views..... 124  
agenda, as an agenda views..... 124  
agenda, column view..... 129  
agenda, sticky..... 104  
agenda, with block views..... 125

alignment in tables..... 22  
anniversaries, from BBDB..... 106  
API、マッピングの..... 254  
API, for properties..... 72, 252  
appointment..... 73, 106  
appointment reminders..... 106  
**appt.el**..... 106  
archive locations..... 100  
archive tag..... 101  
archived entries, in agenda views..... 102  
Archives mode..... 117  
archiving..... 100  
ASCII export..... 145  
Atom feeds..... 96  
attachments..... 95  
author..... 3  
author, macro..... 144  
autoload..... 3

## B

BBDB links..... 39  
BBDB, anniversaries..... 106  
Beamer export..... 147  
begin block, end block..... 200  
block agenda..... 125  
blocking, of checkboxes..... 57  
blocks, folding..... 16  
Boolean logic, for tag/property searches..... 108  
bug reports..... 3

## C

C-c C-c, overview..... 235  
calc package..... 24  
**calc.el**..... 238  
calculations, in tables..... 21, 24  
calendar commands, from agenda..... 122  
calendar, for selecting date..... 76  
capture protocol..... 97  
capture, %:annotation placeholder..... 98  
capture, %:description placeholder..... 98  
capture, %:link placeholder..... 98  
capturing, from agenda..... 121  
category..... 111  
category filtering, in agenda..... 113  
category, require for tags/property match..... 108  
**cdlatex.el**..... 238  
CDL<sup>A</sup>T<sub>E</sub>X..... 137  
center blocks..... 131  
center image (L<sup>A</sup>T<sub>E</sub>X export)..... 163  
checkbox blocking..... 57  
checkbox statistics..... 57  
checkboxes..... 56  
checkboxes and TODO dependencies..... 51  
children, subtree visibility state..... 6  
clean outline view..... 235  
clocking time..... 81

clocktable, dynamic block ..... 83  
code line references, markup rules ..... 132  
column formula ..... 30  
column view, for properties ..... 68  
column view, in agenda ..... 129  
column, of field coordinates ..... 25  
commands, in agenda buffer ..... 115  
comment trees ..... 145  
commented entries, in agenda views ..... 102  
completion, of dictionary words ..... 229  
completion, of file names ..... 42  
completion, of link abbreviations ..... 229  
completion, of links ..... 42  
completion, of option keywords ..... 49, 229  
completion, of property keys ..... 229  
completion, of  $\text{\TeX}$  symbols ..... 229  
completion, of TODO keywords ..... 47, 229  
concept index, in Texinfo export ..... 180  
constants, in calculations ..... 26  
**constants.el** ..... 238  
**content**, STARTUP keyword ..... 8, 232  
contents, global visibility state ..... 7  
continuous clocking ..... 86  
convert ..... 172  
converter ..... 172  
coordinates, of field ..... 25  
copying notes ..... 99  
copying, of subtrees ..... 9  
countdown timer ..... 87  
counter, macro ..... 144  
creating timestamps ..... 74  
CSS, for HTML export ..... 156  
**CUA.el** ..... 239  
custom agenda views ..... 124  
custom date/time format ..... 77  
custom search strings ..... 45  
customization ..... 231  
cutting, of subtrees ..... 9  
cycling, in plain lists ..... 13  
cycling, of agenda files ..... 103  
cycling, of TODO states ..... 46  
cycling, visibility ..... 6

## D

daily agenda ..... 104  
Dan Davidson ..... 199  
dash, special symbol ..... 135  
data type index, in Texinfo export ..... 180  
date format, custom ..... 77  
date range ..... 74  
date stamp ..... 73  
date stamps ..... 73  
date tree ..... 89  
date, macro ..... 144  
date, reading in minibuffer ..... 75  
dates ..... 73  
**DEADLINE** keyword ..... 77

deadlines ..... 73  
debugging, of table formulas ..... 33  
defining new protocols ..... 97  
demotion, of subtrees ..... 9  
dependencies, of TODO states ..... 50  
diary entries, creating from agenda ..... 123  
dictionary word completion ..... 229  
‘**dir**’ file, in Texinfo export ..... 179  
directories, for publishing ..... 188  
display changing, in agenda ..... 116  
doc, docx, rtf ..... 172  
document structure ..... 6  
document title ..... 140  
Dominik, Carsten ..... 238  
**DONE**, final TODO keyword ..... 49  
dragging, agenda lines ..... 121  
drawer, for properties ..... 64  
drawer, for state change recording ..... 52  
drawers ..... 15  
Duration, computing ..... 29  
dvipng ..... 155, 171  
dvisvgm ..... 155, 171  
dynamic indentation ..... 235

## E

**ecomplete.el** ..... 239  
editing tables ..... 19  
editing, of table formulas ..... 31  
edits, catching invisible ..... 8  
effort estimates ..... 87  
effort filtering, in agenda ..... 113  
Elisp links ..... 39  
ellipsis, special symbol ..... 135  
ELPA ..... 3  
email, macro ..... 144  
embedding images in ODT ..... 169  
**entitiesplain**, STARTUP キーワード ..... 234  
Eric Schulte ..... 199  
escaping characters ..... 134  
evaluate time range ..... 75  
example blocks, in  $\text{\LaTeX}$  export ..... 164  
export back-end ..... 138  
export back-ends, built-in ..... 184  
Export, dispatcher ..... 138  
export, OpenDocument ..... 166  
Export, settings ..... 139  
Export, writing back-ends ..... 243  
exporting agenda views ..... 123, 127  
extended TODO keywords ..... 47  
external archiving ..... 100  
external links ..... 39  
external links, in HTML export ..... 153

## F

faces, for TODO keywords ..... 49  
FAQ ..... 1



|   |     |
|---|-----|
| feedback.....   | 3   |
| field coordinates.....  | 25  |
| field formula.....  | 29  |
| field references.....   | 24  |
| file links.....   | 39  |
| file links, searching.....  | 45  |
| file name completion.....   | 42  |
| files, selecting for publishing.....                                    | 189 |
| <b>filladapt.el</b> .....   | 239 |
| filtering, by tag, category, top headline and effort,<br>in agenda..... | 113 |
| Filters, exporting.....   | 185 |
| folded, subtree visibility state.....                                   | 6   |
| folding, sparse trees.....  | 11  |
| following links.....  | 42  |
| footnotes.....  | 16  |
| format specifier.....   | 26  |
| format, of links.....   | 38  |
| formula debugging.....  | 33  |
| formula editing.....  | 31  |
| formula syntax, Calc.....   | 26  |
| formula, for individual table field.....                                | 29  |
| formula, for range of fields.....                                       | 29  |
| formula, for table column.....  | 30  |
| formula, in tables.....   | 21  |
| function index, in Texinfo export.....                                  | 180 |

## G

|   |     |
|---|-----|
| Gillespie, Dave.....                    | 238 |
| global cycling.....                     | 7   |
| global key bindings.....                | 3   |
| global TODO list.....                   | 106 |
| global visibility states.....           | 7   |
| Gnus links.....                         | 39  |
| graph, in tables.....                   | 35  |
| group tags.....                         | 62  |
| group tags, as regular expressions..... | 108 |
| grouping columns in tables.....         | 23  |

## H

|  |         |
|--|---------|
| habits.....  | 53      |
| handle rewritten URL in open-source protocol..                   | 98      |
| header, for L <sup>A</sup> T <sub>E</sub> X files.....           | 160     |
| headline navigation.....   | 8       |
| headline, promotion and demotion.....                            | 9       |
| headlines.....   | 6       |
| hide text.....   | 6       |
| <b>hideblocks</b> , STARTUP keyword.....                         | 16, 234 |
| hiding leading stars.....  | 235     |
| history.....   | 258     |
| horizontal rules, in ASCII export.....                           | 146     |
| horizontal rules, in L <sup>A</sup> T <sub>E</sub> X export..... | 165     |
| horizontal rules, markup rules.....                              | 132     |
| HTML エクスポート、CSS.....   | 156     |
| HTML、Orgtbl モードと.....  | 247     |
| HTML entities.....   | 134     |

|                                 |     |
|---------------------------------|-----|
| HTML export.....                | 150 |
| HTML5, export new elements..... | 152 |
| hyperlinks.....                 | 38  |

## I

|  |          |
|--|----------|
| iCalendar export.....  | 183      |
| identify, ImageMagick.....                                     | 169      |
| idle, resolve, dangling.....                                   | 85       |
| image, centering (L <sup>A</sup> T <sub>E</sub> X export)..... | 163      |
| imagemagick.....   | 155, 171 |
| images, embedding in ODT.....                                  | 169      |
| images, inline in HTML.....                                    | 154      |
| images, inline in L <sup>A</sup> T <sub>E</sub> X.....         | 162      |
| images, inlining.....  | 43       |
| <b>imenu.el</b> .....  | 238      |
| in-buffer settings.....  | 231      |
| inactive timestamp.....  | 74       |
| include files, during export.....                              | 143      |
| indentation, in source blocks.....                             | 134, 201 |
| index, in a publishing project.....                            | 195      |
| Info directory file, in Texinfo export.....                    | 179      |
| Info links.....  | 39       |
| inheritance, of properties.....                                | 67       |
| inline, in L <sup>A</sup> T <sub>E</sub> X export.....         | 161      |
| <b>inlineimages</b> , STARTUP keyword.....                     | 43, 233  |
| inlining images.....   | 43       |
| inlining images in HTML.....                                   | 154      |
| inlining images in L <sup>A</sup> T <sub>E</sub> X.....        | 162      |
| input file, macro.....   | 144      |
| inserting links.....   | 42       |
| insertion, of templates.....                                   | 229      |
| <b>install-info</b> parameters, in Texinfo export..            | 179      |
| installation.....  | 2        |
| Installing Org protocol.....                                   | 97       |
| internal links.....  | 38       |
| internal links, in HTML export.....                            | 153      |
| introduction.....  | 1        |
| iPhone.....  | 256      |
| IRC links.....   | 39       |
| italic text, markup rules.....                                 | 131      |

## J

|                            |   |
|----------------------------|---|
| jumping, to headlines..... | 8 |
|----------------------------|---|

## K

|   |     |
|---|-----|
| key bindings, global.....               | 3   |
| keystroke index, in Texinfo export..... | 180 |
| keyword options.....                    | 49  |

## L

|   |     |
|---|-----|
| L <sup>A</sup> T <sub>E</sub> X クラス.....      | 160 |
| L <sup>A</sup> T <sub>E</sub> X の解釈.....      | 135 |
| L <sup>A</sup> T <sub>E</sub> X のエクスポート.....  | 158 |
| L <sup>A</sup> T <sub>E</sub> X の断片的なコード..... | 135 |

|  |          |
|--|----------|
| $\text{\LaTeX}$ , Orgtbl モードと .....                    | 245      |
| $\text{\LaTeX}$ entities .....                         | 134      |
| $\text{\LaTeX}$ fragments, preview .....               | 136      |
| $\text{\LaTeX}$ header .....                           | 160      |
| $\text{\LaTeX}$ sectioning structure .....             | 160      |
| <code>latexpreview</code> , STARTUP keyword .....      | 233      |
| Latin-1 でのエクスポート .....                                 | 145      |
| level, require for tags/property match .....           | 108      |
| LibreOffice .....                                      | 166, 167 |
| limits, in agenda .....                                | 115      |
| link abbreviations .....                               | 44       |
| link abbreviations, completion of .....                | 229      |
| link completion .....                                  | 42       |
| link format .....                                      | 38       |
| links, external .....                                  | 39       |
| links, finding next/previous .....                     | 43       |
| links, handling .....                                  | 41       |
| links, in HTML export .....                            | 153      |
| links, in ODT export .....                             | 168      |
| links, internal .....                                  | 38       |
| links, publishing .....                                | 194      |
| links, radio targets .....                             | 39       |
| links, returning to .....                              | 43       |
| linter .....   | 18       |
| Lisp forms, as table formulas .....                    | 28       |
| list of listings .....                                 | 142      |
| list of tables .....                                   | 142      |
| lists, ordered .....                                   | 12       |
| lists, plain .....                                     | 12       |
| <code>logdrawer</code> , STARTUP keyword .....         | 233      |
| logging, of progress .....                             | 52       |
| <code>logstatesreversed</code> , STARTUP keyword ..... | 233      |
| lookup functions in tables .....                       | 30       |
| Ludlam, Eric M. ....                                   | 238      |

## M

|   |     |
|---|-----|
| maintainer .....  | 3   |
| mark ring .....   | 43  |
| Markdown export .....   | 165 |
| marking characters, tables .....                                  | 34  |
| match view .....  | 107 |
| matching, of properties .....                                     | 107 |
| matching, of tags .....   | 107 |
| MathJax .....   | 155 |
| MathML .....  | 170 |
| MH-E links .....  | 39  |
| minor mode for structure editing .....                            | 17  |
| minor mode for tables .....                                       | 23  |
| MobileOrg .....   | 256 |
| mode, for <code>calc</code> .....                                 | 26  |
| modification time, macro .....                                    | 144 |
| motion commands in agenda .....                                   | 115 |
| motion, between headlines .....                                   | 8   |
| Multiple entries in two-column tables, in Texinfo<br>export ..... | 180 |

## N

|  |         |
|--|---------|
| <code>n</code> , macro .....                           | 144     |
| name, of column or field .....                         | 26      |
| named references .....                                 | 26      |
| names as TODO keywords .....                           | 48      |
| narrow columns in tables .....                         | 22      |
| <code>nohideblocks</code> , STARTUP keyword .....      | 16, 234 |
| <code>noinlineimages</code> , STARTUP keyword .....    | 43, 233 |
| <code>nolatexpreview</code> , STARTUP keyword .....    | 233     |
| <code>nologdrawer</code> , STARTUP keyword .....       | 233     |
| <code>nologstatesreversed</code> , STARTUP keyword ... | 233     |

## O

|   |         |
|---|---------|
| occur, command .....                          | 11      |
| occur-tree .....                              | 124     |
| odd-levels-only outlines .....                | 235     |
| ODT .....                                     | 166     |
| ODT_STYLES_FILE .....                         | 167     |
| open-source protocol .....                    | 98      |
| OpenDocument .....                            | 166     |
| option keyword completion .....               | 229     |
| options, for custom agenda views .....        | 125     |
| options, for customization .....              | 231     |
| options, for publishing .....                 | 190     |
| ordered lists .....                           | 12      |
| Org entities .....                            | 134     |
| Org export .....                              | 177     |
| Org mode, turning on .....                    | 3       |
| Org protocol, set-up .....                    | 97      |
| Org syntax .....                              | 18      |
| <code>org-agenda</code> , command .....       | 104     |
| <code>org-capture-last-stored</code> .....    | 90      |
| <code>org-crypt.el</code> .....               | 240     |
| <code>org-decrypt-entry</code> .....          | 240     |
| <code>org-hide-block-startup</code> .....     | 234     |
| <code>org-insert-drawer</code> .....          | 15, 65  |
| <code>org-list-insert-radio-list</code> ..... | 247     |
| <code>org-pretty-entities</code> .....        | 234     |
| <code>org-publish-project-alist</code> .....  | 188     |
| Orgstruct mode .....                          | 17      |
| Orgtbl mode .....                             | 23, 244 |
| Ota, Takaaki .....                            | 238     |
| Outline mode .....                            | 6       |
| outline tree .....                            | 6       |
| outlines .....                                | 6       |
| overview, global visibility state .....       | 7       |
| <code>overview</code> , STARTUP keyword ..... | 8, 232  |

## P

|  |     |
|--|-----|
| packages, interaction with other .....       | 237 |
| pasting, of subtrees .....                   | 9   |
| PDF export .....                             | 158 |
| per-file keywords .....                      | 49  |
| plain lists .....                            | 12  |
| plain lists, in $\text{\LaTeX}$ export ..... | 163 |
| plain text external links .....              | 40  |

- plot tables using Gnuplot ..... 35
  - presentation, of agenda items ..... 111
  - print edition ..... 1
  - printing sparse trees ..... 12
  - priorities ..... 55
  - priorities, of agenda items ..... 112
  - program index, in Texinfo export ..... 180
  - progress logging ..... 52
  - projects, for publishing ..... 188
  - promotion, of subtrees ..... 9
  - proof, in L<sup>A</sup>T<sub>E</sub>X export ..... 165
  - properties ..... 64
  - properties, API ..... 72, 252
  - properties, column view ..... 68
  - properties, inheritance ..... 67
  - properties, searching ..... 66
  - properties, special ..... 66
  - property syntax ..... 64
  - property, + ..... 65
  - property, `_ALL` ..... 65
  - property, `ALT_TITLE` ..... 142
  - property, `APPENDIX` ..... 179
  - property, `ARCHIVE` ..... 67, 100
  - property, `ATTACH_DIR` ..... 96
  - property, `ATTACH_DIR_INHERIT` ..... 96
  - property, `BEAMER_ACT` ..... 148
  - property, `BEAMER_COL` ..... 148
  - property, `BEAMER_ENV` ..... 148
  - property, `BEAMER_OPT` ..... 148
  - property, `BEAMER_REF` ..... 148
  - property, `CATEGORY` ..... 67, 111
  - property, `CLOCK_MODELINE_TOTAL` ..... 81
  - property, `COLUMNS` ..... 67, 232
  - property, `COOKIE_DATA` ..... 56, 57
  - property, `COPYING` ..... 179
  - property, `CUSTOM_ID` ..... 38, 41
  - property, `DESCRIPTION` ..... 179, 184
  - property, `EFFORT` ..... 87
  - property, `EXPORT_FILE_NAME` ..... 166
  - property, `EXPORT_LATEX_CLASS` ..... 160
  - property, `EXPORT_LATEX_CLASS_OPTIONS` ..... 160
  - property, `ID` ..... 41, 71, 183
  - property, `INDEX` ..... 180
  - property, `LAST_REPEAT` ..... 81
  - property, `LOCATION` ..... 184
  - property, `LOG_INTO_DRAWER` ..... 52, 81
  - property, `LOGGING` ..... 53, 67
  - property, macro ..... 144
  - property, `ORDERED` ..... 50, 57, 58
  - property, special, `ALLTAGS` ..... 66
  - property, special, `BLOCKED` ..... 66
  - property, special, `CLOCKSUM` ..... 66, 129
  - property, special, `CLOCKSUM_T` ..... 66, 130
  - property, special, `CLOSED` ..... 66
  - property, special, `DEADLINE` ..... 66
  - property, special, `FILE` ..... 66
  - property, special, `ITEM` ..... 66
  - property, special, `PRIORITY` ..... 66
  - property, special, `SCHEDULED` ..... 66
  - property, special, `TAGS` ..... 66
  - property, special, `TIMESTAMP` ..... 66
  - property, special, `TIMESTAMP_IA` ..... 66
  - property, special, `TODO` ..... 66
  - property, `SUMMARY` ..... 184
  - property, `TIMEZONE` ..... 184
  - property, `UNNUMBERED` ..... 141
  - property, `VISIBILITY` ..... 8
  - protocol, capture ..... 97
  - protocol, new protocol ..... 97
  - protocol, open-source ..... 98
  - protocol, open-source rewritten URL ..... 98
  - protocol, open-source, `:base-url` property ..... 98
  - protocol, open-source, `:online-suffix` property ..... 98
  - protocol, open-source, `:rewrites` property ..... 98
  - protocol, open-source, `:working-directory` property ..... 98
  - protocol, open-source, `:working-suffix` property ..... 98
  - protocol, open-source, set-up mapping ..... 99
  - protocol, store-link ..... 97
  - protocols, for external access ..... 97
  - publishing ..... 188
- ## Q
- query editing, in agenda ..... 113
  - quote blocks ..... 131
- ## R
- radio targets ..... 39
  - range formula ..... 29
  - range references ..... 25
  - ranges, time ..... 73
  - recomputing table fields ..... 33
  - references ..... 24
  - references, named ..... 26
  - references, remote ..... 26
  - references, to a different table ..... 26
  - references, to fields ..... 24
  - references, to ranges ..... 25
  - refiling notes ..... 99
  - region, active ..... 11, 21, 166
  - regular expressions, with tags search ..... 108
  - relative timer ..... 87
  - reminders ..... 106
  - remote editing, bulk, from agenda ..... 121
  - remote editing, from agenda ..... 119
  - remote editing, undo ..... 119
  - remote references ..... 26
  - repeated tasks ..... 79
  - report, of clocked time ..... 83
  - resolve idle time ..... 85
  - revealing context ..... 7
  - RMAIL links ..... 39

Rose, Sebastian ..... 157  
 row, of field coordinates ..... 25  
 RSS フィールド ..... 96  
 rsync ..... 196

## S

**SCHEDULED** keyword ..... 78  
 scheduling ..... 73  
 Scripts, for agenda processing ..... 251  
 search option in file links ..... 45  
 search strings, custom ..... 45  
 searching, for text ..... 110  
 searching, of properties ..... 66  
 sectioning structure, for L<sup>A</sup>T<sub>E</sub>X export ..... 160  
 set startup visibility, command ..... 7  
 set-up mappings in open-source protocol ..... 99  
 SHELL links ..... 39  
 shift-selection-mode ..... 14, 239  
 show all, command ..... 7  
 show all, global visibility state ..... 7  
 show branches, command ..... 7  
 show children, command ..... 7  
 show hidden text ..... 6  
**showall**, STARTUP keyword ..... 8, 232  
**showeverything**, STARTUP keyword ..... 8, 232  
 shy hyphen, special symbol ..... 135  
 sitemap, of published pages ..... 194  
 sorting, of agenda items ..... 112  
 sorting, of plain list ..... 15  
 sorting, of subtrees ..... 9  
 source blocks, in L<sup>A</sup>T<sub>E</sub>X export ..... 164  
 source code, inline ..... 200  
 source code, language ..... 200  
 source code, switches ..... 200  
 sparse tree, for deadlines ..... 79  
 sparse tree, for TODO ..... 46  
 sparse trees ..... 11  
 special blocks, in ASCII export ..... 146  
 special blocks, in L<sup>A</sup>T<sub>E</sub>X export ..... 165  
 special keywords ..... 231  
 special symbols, in-buffer display ..... 135  
 speed keys ..... 230  
**speedbar.el** ..... 238  
 spreadsheet capabilities ..... 24  
 square brackets, around links ..... 40  
 STARTUP キーワード、**align** ..... 233  
 STARTUP キーワード、**constcgs** ..... 234  
 STARTUP キーワード、**constSI** ..... 234  
 STARTUP キーワード、**customtime** ..... 234  
 STARTUP キーワード、**entitiespretty** ..... 135, 234  
 STARTUP キーワード、**even** ..... 234  
 STARTUP キーワード、**fnadjust** ..... 234  
 STARTUP キーワード、**fnauto** ..... 234  
 STARTUP キーワード、**fnconfirm** ..... 234  
 STARTUP キーワード、**fninline** ..... 234  
 STARTUP キーワード、**fnlocal** ..... 234

STARTUP キーワード、**fnplain** ..... 234  
 STARTUP キーワード、**fnprompt** ..... 234  
 STARTUP キーワード、**hidestars** ..... 234  
 STARTUP キーワード、**indent** ..... 233  
 STARTUP キーワード、**logdone** ..... 233  
 STARTUP キーワード、**lognoredeadline** ..... 233  
 STARTUP キーワード、**lognoteclock-out** ..... 233  
 STARTUP キーワード、**lognotedone** ..... 233  
 STARTUP キーワード、**lognoterefile** ..... 233  
 STARTUP キーワード、**lognoterepeat** ..... 233  
 STARTUP キーワード、**lognotereschedule** ..... 233  
 STARTUP キーワード、**logredeadline** ..... 233  
 STARTUP キーワード、**logrefile** ..... 233  
 STARTUP キーワード、**logrepeat** ..... 233  
 STARTUP キーワード、**logschedule** ..... 233  
 STARTUP キーワード、**noalign** ..... 233  
 STARTUP キーワード、**nofnadjust** ..... 234  
 STARTUP キーワード、**nofninline** ..... 234  
 STARTUP キーワード、**noindent** ..... 233  
 STARTUP キーワード、**nologdone** ..... 233  
 STARTUP キーワード、**nolognoteclock-out** ..... 233  
 STARTUP キーワード、**nologredeadline** ..... 233  
 STARTUP キーワード、**nologrefile** ..... 233  
 STARTUP キーワード、**nologrepeat** ..... 233  
 STARTUP キーワード、**nologreschedule** ..... 233  
 STARTUP キーワード、**odd** ..... 234  
 STARTUP キーワード、**showstars** ..... 234  
 statistics, for checkboxes ..... 57  
 statistics, for TODO items ..... 55  
 store-link protocol ..... 97  
 storing links ..... 41  
 Storm, Kim. F. .... 239  
 structure editing ..... 9  
 structure of document ..... 6  
 styles, custom ..... 168, 173  
 sublevels, inclusion into TODO list ..... 107  
 SUBTITLE (ODT) ..... 167  
 subtree cycling ..... 6  
 subtree visibility states ..... 6  
 subtree, cut and paste ..... 9  
 subtree, subtree visibility state ..... 6  
 subtrees, cut and paste ..... 9  
 summary ..... 1  
 syntax checker ..... 18  
 syntax, of formulas ..... 26

## T

table editor, built-in ..... 19  
 table editor, **table.el** ..... 238  
 table lookup functions ..... 30  
**table.el** ..... 238  
 tables ..... 19  
 tables, in HTML ..... 154  
 tables, in L<sup>A</sup>T<sub>E</sub>X export ..... 161  
 tables, in ODT export ..... 168, 174  
 tag completion ..... 229

- tag filtering, in agenda..... 113
  - tag hierarchy..... 62
  - tags view..... 107
  - tags, as an agenda view..... 124
  - tags, groups..... 62
  - tags-todo..... 124
  - tags-tree..... 124
  - tangling..... 203
  - targets, for links..... 38
  - targets, radio..... 39
  - tasks, breaking down..... 55
  - tasks, repeated..... 79
  - template insertion..... 229
  - template, custom..... 168, 173
  - TeX の解釈..... 135
  - TeX symbol completion..... 229
  - Texinfo export..... 177
  - Texinfo export, data type index..... 180
  - Texinfo export, ‘dir’ file..... 179
  - Texinfo export, index, concept..... 180
  - Texinfo export, index, function..... 180
  - Texinfo export, Info directory file..... 179
  - Texinfo export, **install-info** parameters..... 179
  - Texinfo export, keystroke index..... 180
  - Texinfo export, program index..... 180
  - Texinfo export, Top node..... 180
  - Texinfo export, variable index..... 180
  - text search..... 110
  - thanks..... 258
  - The Top node, in Texinfo export..... 180
  - time clocking..... 81
  - time format, custom..... 77
  - time grid..... 112
  - Time, computing..... 29
  - time, macro..... 144
  - time, reading in minibuffer..... 75
  - time-of-day specification..... 111
  - timerange..... 74
  - times..... 73
  - timestamp..... 73
  - timestamp, inactive..... 74
  - timestamp, with repeater interval..... 73
  - timestamps..... 73
  - timestamps, creating..... 74
  - title, macro..... 144
  - todo, as an agenda view..... 124
  - todo-tree..... 124
  - TODO dependencies..... 50
  - TODO dependencies, NOBLOCKING..... 50
  - TODO items..... 46
  - TODO keyword matching..... 107
  - TODO keyword matching, with tags search... 108
  - TODO keyword sets..... 48
  - TODO keywords completion..... 229
  - TODO list, global..... 106
  - TODO types..... 48
  - TODO workflow..... 47
  - top headline filtering, in agenda..... 113
  - transient mark mode..... 11, 21
  - transient-mark-mode**..... 166
  - trees, sparse..... 11
  - trees, visibility..... 6
  - tty key bindings..... 237
  - Two-column tables, in Texinfo export..... 180
  - types as TODO keywords..... 48
- ## U
- undoing remote-editing events..... 119
  - unison..... 196
  - unoconv**..... 167
  - updating, table..... 33
  - URL links..... 39
  - USENET links..... 39
  - UTF-8 でのエクスポート..... 145
- ## V
- variable index, in Texinfo export..... 180
  - variables, for customization..... 231
  - vectors, in table calculations..... 26
  - verbatim blocks, in L<sup>A</sup>T<sub>E</sub>X export..... 164
  - verse blocks..... 131
  - viper.el**..... 240
  - visibility cycling..... 6
  - visibility cycling, drawers..... 15
  - visibility, initialize..... 8
  - visible text, printing..... 12
  - VM links..... 40
- ## W
- WANDERLUST links..... 40
  - weekly agenda..... 104
  - windmove.el**..... 240
  - workflow states as TODO keywords..... 47
- ## Y
- yasnippet.el**..... 239
- ## Z
- zip..... 166

## Key index

|             |                       |             |                  |
|-------------|-----------------------|-------------|------------------|
| <b>\$</b>   |                       | <b>&gt;</b> |                  |
| \$.....     | 119                   | >.....      | 71, 76, 103, 120 |
| <b>%</b>    |                       | <b>?</b>    |                  |
| %.....      | 121                   | ?.....      | 257              |
| <b>,</b>    |                       | <b>[</b>    |                  |
| '.....      | 137                   | [.....      | 113, 117         |
| <b>*</b>    |                       | <b>]</b>    |                  |
| *.....      | 121                   | ].....      | 113              |
| <b>+</b>    |                       | <b>^</b>    |                  |
| +.....      | 120                   | ^.....      | 114, 119, 137    |
| <b>,</b>    |                       | <b>-</b>    |                  |
| ,.....      | 120                   | -.....      | 114, 119, 137    |
| <b>-</b>    |                       | <b>‘</b>    |                  |
| -.....      | 120                   | `.....      | 137              |
| <b>.</b>    |                       | <b> </b>    |                  |
| .....       | 117                   | .....       | 114, 119         |
| <b>/</b>    |                       | {.....      | 113              |
| /.....      | 113, 118              | }.....      | 113              |
| <b>:</b>    |                       | <b>~</b>    |                  |
| :.....      | 120                   | ~.....      | 115              |
| <b>;</b>    |                       | <b>A</b>    |                  |
| ;.....      | 87                    | a.....      | 71, 119          |
| <b>&lt;</b> |                       | A.....      | 116              |
| <.....      | 71, 76, 103, 114, 118 | <b>B</b>    |                  |
| <b>=</b>    |                       | b.....      | 117              |
| =.....      | 114, 119              | B.....      | 121              |
|             |                       | <b>C</b>    |                  |
|             |                       | c.....      | 122              |
|             |                       | C.....      | 123              |
|             |                       | C-#.....    | 34               |
|             |                       | C-'.....    | 103              |

|                   |                            |  |                      |
|-------------------|----------------------------|--|----------------------|
| C-, .....         | 103                        | C-c C-a f.....   | 96                   |
| C_ .....          | 119                        | C-c C-a F.....   | 96                   |
| C-O C-c C-w ..... | 100                        | C-c C-a i.....   | 96                   |
| C-c ! .....       | 74                         | C-c C-a l.....   | 95                   |
| C-c # .....       | 58                         | C-c C-a m.....   | 95                   |
| C-c \$ .....      | 100                        | C-c C-a n.....   | 95                   |
| C-c % .....       | 43                         | C-c C-a o.....   | 95                   |
| C-c & .....       | 43                         | C-c C-a O.....   | 96                   |
| C-c ' .....       | 17, 32, 134, 143, 201, 238 | C-c C-a s.....   | 96                   |
| C-c * .....       | 11, 15, 33                 | C-c C-a u.....   | 95                   |
| C-c + .....       | 21                         | C-c C-a z.....   | 95                   |
| C-c , .....       | 55                         | C-c C-b.....   | 8, 150               |
| C-c - .....       | 14, 20                     | C-c C-c... 14, 17, 19, 32, 33, 57, 60, 65, 70, 72, 74, |                      |
| C-c .....         | 74                         | 82, 83, 89, 137, 204, 227, 235, 238                    |                      |
| C-c / .....       | 11, 240                    | C-c C-c c.....   | 66                   |
| C-c / / .....     | 11                         | C-c C-c d.....   | 66                   |
| C-c / a.....      | 79                         | C-c C-c D.....   | 66                   |
| C-c / b.....      | 79                         | C-c C-c s.....   | 65                   |
| C-c / d.....      | 78                         | C-c C-d.....   | 78, 120              |
| C-c / m.....      | 63, 66                     | C-c C-e .....  | 138                  |
| C-c / p.....      | 67                         | C-c C-e c a .....                                      | 183                  |
| C-c / r.....      | 11                         | C-c C-e c c .....                                      | 183                  |
| C-c / t.....      | 46                         | C-c C-e c f .....                                      | 183                  |
| C-c ; .....       | 145                        | C-c C-e C-v .....                                      | 12                   |
| C-c < .....       | 74                         | C-c C-e h h .....                                      | 151                  |
| C-c = .....       | 30, 31                     | C-c C-e h H .....                                      | 151                  |
| C-c > .....       | 74                         | C-c C-e i i .....                                      | 177                  |
| C-c ? .....       | 31                         | C-c C-e i t .....                                      | 177                  |
| C-c @ .....       | 10                         | C-c C-e l b .....                                      | 147                  |
| C-c [ .....       | 102                        | C-c C-e l B .....                                      | 147                  |
| C-c ] .....       | 102                        | C-c C-e l l .....                                      | 158                  |
| C-c ^ .....       | 11, 15, 20                 | C-c C-e l L .....                                      | 158                  |
| C-c ` .....       | 21                         | C-c C-e l p .....                                      | 158                  |
| C-c \ .....       | 63, 66                     | C-c C-e l P .....                                      | 147                  |
| C-c   .....       | 19, 22                     | C-c C-e m m .....                                      | 166                  |
| C-c { .....       | 31, 137                    | C-c C-e m M .....                                      | 166                  |
| C-c } .....       | 31, 32                     | C-c C-e o o .....                                      | 166                  |
| C-c ~ .....       | 238                        | C-c C-e O o .....                                      | 177                  |
| C-c a .....       | 5                          | C-c C-e O O .....                                      | 177                  |
| C-c a ! .....     | 110                        | C-c C-e P a .....                                      | 198                  |
| C-c a # .....     | 110                        | C-c C-e P f .....                                      | 197                  |
| C-c a ? .....     | 257                        | C-c C-e P p .....                                      | 197                  |
| C-c a a .....     | 104                        | C-c C-e P x .....                                      | 197                  |
| C-c a C .....     | 124                        | C-c C-e t a/l/u.....                                   | 145                  |
| C-c a e .....     | 128                        | C-c C-e t A/L/U.....                                   | 145                  |
| C-c a m .....     | 63, 66, 107                | C-c C-f.....   | 8                    |
| C-c a M .....     | 63, 66, 107                | C-c C-j.....   | 8                    |
| C-c a s .....     | 110                        | C-c C-k.....   | 7, 90                |
| C-c a t .....     | 47, 107                    | C-c C-l .....  | 42                   |
| C-c a T .....     | 107                        | C-c C-n.....   | 8                    |
| C-c c .....       | 5, 89                      | C-c C-o.....   | 17, 42, 74, 116, 227 |
| C-c c C .....     | 90                         | C-c C-p.....   | 8                    |
| C-c C-* .....     | 15                         | C-c C-q.....   | 32, 59               |
| C-c C-a .....     | 95, 120                    | C-c C-r.....   | 7, 32                |
| C-c C-a a .....   | 95                         | C-c C-s.....   | 78, 120              |
| C-c C-a c .....   | 95                         | C-c C-t.....   | 46, 82               |
| C-c C-a d .....   | 96                         | C-c C-u.....   | 8                    |
| C-c C-a D .....   | 96                         | C-c C-v a .....  | 228                  |

|                   |                 |                       |              |
|-------------------|-----------------|-----------------------|--------------|
| C-c C-v b .....   | 227             | C-c C-x C-b .....     | 57           |
| C-c C-v c .....   | 228             | C-c C-x C-c .....     | 70, 118, 129 |
| C-c C-v C-a ..... | 228             | C-c C-x C-d .....     | 82           |
| C-c C-v C-b ..... | 227             | C-c C-x C-e .....     | 82, 87       |
| C-c C-v C-c ..... | 228             | C-c C-x C-i .....     | 81           |
| C-c C-v C-d ..... | 227             | C-c C-x C-j .....     | 82           |
| C-c C-v C-e ..... | 227             | C-c C-x C-l .....     | 136          |
| C-c C-v C-f ..... | 228             | C-c C-x C-n .....     | 43           |
| C-c C-v C-g ..... | 227             | C-c C-x C-o .....     | 81           |
| C-c C-v C-h ..... | 228             | C-c C-x C-p .....     | 43           |
| C-c C-v C-i ..... | 228             | C-c C-x C-q .....     | 82           |
| C-c C-v C-I ..... | 228             | C-c C-x C-r .....     | 83           |
| C-c C-v C-j ..... | 228             | C-c C-x C-s .....     | 100, 119     |
| C-c C-v C-l ..... | 228             | C-c C-x C-t .....     | 77           |
| C-c C-v C-n ..... | 227             | C-c C-x C-u .....     | 72, 83, 248  |
| C-c C-v C-o ..... | 227             | C-c C-x C-v .....     | 43           |
| C-c C-v C-p ..... | 227             | C-c C-x C-w .....     | 10, 21       |
| C-c C-v C-r ..... | 227             | C-c C-x C-x .....     | 82           |
| C-c C-v C-s ..... | 227             | C-c C-x C-y .....     | 10, 21       |
| C-c C-v C-t ..... | 227             | C-c C-x d .....       | 15           |
| C-c C-v C-u ..... | 227             | C-c C-x e .....       | 87           |
| C-c C-v C-v ..... | 227             | C-c C-x f .....       | 16           |
| C-c C-v C-x ..... | 228             | C-c C-x g .....       | 96           |
| C-c C-v C-z ..... | 228             | C-c C-x G .....       | 96           |
| C-c C-v d .....   | 227             | C-c C-x i .....       | 72           |
| C-c C-v e .....   | 227             | C-c C-x M-w .....     | 10, 21       |
| C-c C-v f .....   | 228             | C-c C-x o .....       | 50, 58       |
| C-c C-v g .....   | 227             | C-c C-x p .....       | 65, 207      |
| C-c C-v h .....   | 228             | C-c C-x q .....       | 63           |
| C-c C-v i .....   | 205, 228        | C-c C-x v .....       | 7            |
| C-c C-v I .....   | 228             | C-c C-y .....         | 75, 82       |
| C-c C-v j .....   | 228             | C-c C-z .....         | 16, 120      |
| C-c C-v l .....   | 228             | C-c l .....           | 41, 134      |
| C-c C-v n .....   | 227             | C-c M-w .....         | 99           |
| C-c C-v o .....   | 227             | C-c RET .....         | 20           |
| C-c C-v p .....   | 227             | C-c SPC .....         | 19           |
| C-c C-v r .....   | 227             | C-c TAB .....         | 7            |
| C-c C-v s .....   | 227             | C-k .....             | 119          |
| C-c C-v t .....   | 203, 227        | C-M-i .....           | 229          |
| C-c C-v u .....   | 227             | C-RET .....           | 9            |
| C-c C-v v .....   | 227             | C-S-LEFT .....        | 48, 119      |
| C-c C-v x .....   | 228             | C-S-RET .....         | 9            |
| C-c C-v z .....   | 228             | C-S-RIGHT .....       | 48, 119      |
| C-c C-w .....     | 10, 89, 99, 119 | C-S-up/down .....     | 82           |
| C-c C-x , .....   | 88              | C-TAB .....           | 101          |
| C-c C-x - .....   | 88              | C-u C-c ! .....       | 74           |
| C-c C-x .....     | 88              | C-u C-c * .....       | 33           |
| C-c C-x ; .....   | 88              | C-u C-c .....         | 74           |
| C-c C-x < .....   | 103             | C-u C-c = .....       | 30, 31       |
| C-c C-x > .....   | 103, 118        | C-u C-c c .....       | 90           |
| C-c C-x _ .....   | 88              | C-u C-c C-c .....     | 33           |
| C-c C-x \ .....   | 135             | C-u C-c C-l .....     | 42           |
| C-c C-x 0 .....   | 88              | C-u C-c C-t .....     | 46           |
| C-c C-x a .....   | 101, 119        | C-u C-c C-w .....     | 99           |
| C-c C-x A .....   | 101, 119        | C-u C-c C-x a .....   | 101          |
| C-c C-x b .....   | 7, 116          | C-u C-c C-x C-s ..... | 100          |
| C-c C-x c .....   | 10              | C-u C-c C-x C-u ..... | 72, 83, 248  |
| C-c C-x C-a ..... | 100, 119        | C-u C-u C-c * .....   | 33           |



C-u C-u C-c = ..... 31  
 C-u C-u C-c c ..... 90  
 C-u C-u C-c C-c ..... 33  
 C-u C-u C-c C-t ..... 48  
 C-u C-u C-c C-w ..... 99  
 C-u C-u C-c C-x C-s ..... 100  
 C-u C-u C-u C-c C-t ..... 50  
 C-u C-u C-u TAB ..... 7  
 C-u C-u TAB ..... 7, 8  
 C-v ..... 76  
 C-x C-s ..... 32, 118  
 C-x C-w ..... 123, 127  
 C-x n b ..... 11  
 C-x n s ..... 11  
 C-x n w ..... 11  
 C-y ..... 10

## D

d ..... 116  
 D ..... 117

## E

e ..... 70  
 E ..... 118

## F

f ..... 117  
 F ..... 116

## G

g ..... 70, 118  
 G ..... 118

## H

H ..... 123

## I

i ..... 123  
 I ..... 121

## J

j ..... 117  
 J ..... 117, 121

## K

k ..... 121

## L

l ..... 117

L ..... 116

## M

m ..... 121  
 M-\*. ..... 121  
 M-<down> ..... 121  
 M-<up> ..... 121  
 M-a ..... 20  
 M-DOWN ..... 10, 14, 20, 32, 227  
 M-e ..... 20  
 M-g M-n ..... 12  
 M-g M-p ..... 12  
 M-g n ..... 12  
 M-g p ..... 12  
 M-h ..... 10  
 M-LEFT ..... 9, 14, 20  
 M-m ..... 121  
 M-RET ..... 9, 14, 21, 88  
 M-RIGHT ..... 9, 14, 20  
 M-S-DOWN ..... 20, 32, 76  
 M-S-LEFT ..... 9, 14, 20, 76  
 M-S-RET ..... 9, 14, 58  
 M-S-RIGHT ..... 10, 14, 20, 76  
 M-S-UP ..... 20, 32, 76  
 M-TAB ..... 32, 49, 59, 65, 229  
 M-UP ..... 10, 14, 20, 32, 227  
 M-v ..... 76  
 mouse-1 ..... 17, 43, 76  
 mouse-2 ..... 17, 43, 116  
 mouse-3 ..... 43, 116  
 M ..... 123

## N

n ..... 70, 115  
 N ..... 115

## O

o ..... 116  
 O ..... 121

## P

p ..... 70, 115  
 P ..... 116, 120

## Q

q ..... 70, 123

## R

r ..... 70, 107, 118  
 R ..... 117  
 RET ..... 20, 43, 61, 76, 116

**S**

|                          |   |
|--------------------------|---|
| <b>s</b> .....           | 118                                     |
| <b>S</b> .....           | 123                                     |
| <b>S-DOWN</b> .....      | 14, 32, 55, 74, 76, 120                 |
| <b>S-LEFT</b> .....      | 15, 32, 46, 48, 65, 70, 74, 76, 83, 120 |
| <b>S-M-LEFT</b> .....    | 71                                      |
| <b>S-M-RET</b> .....     | 47                                      |
| <b>S-M-RIGHT</b> .....   | 71                                      |
| <b>S-M-up/down</b> ..... | 82                                      |
| <b>S-RET</b> .....       | 21                                      |
| <b>S-RIGHT</b> .....     | 15, 32, 46, 48, 65, 70, 74, 76, 83, 120 |
| <b>S-TAB</b> .....       | 7, 20                                   |
| <b>S-UP</b> .....        | 32, 55, 74, 76, 120                     |
| <b>SPC</b> .....         | 61, 116                                 |

**T**

|                  |                                |
|------------------|--------------------------------|
| <b>t</b> .....   | 119                            |
| <b>T</b> .....   | 119                            |
| <b>TAB</b> ..... | 7, 9, 13, 20, 32, 61, 116, 137 |

**U**

|                |     |
|----------------|-----|
| <b>u</b> ..... | 121 |
| <b>U</b> ..... | 121 |

**V**

|                    |     |
|--------------------|-----|
| <b>v</b> .....     | 71  |
| <b>v [</b> .....   | 117 |
| <b>v a</b> .....   | 117 |
| <b>v A</b> .....   | 117 |
| <b>v c</b> .....   | 118 |
| <b>v d</b> .....   | 116 |
| <b>v E</b> .....   | 118 |
| <b>v l</b> .....   | 117 |
| <b>v L</b> .....   | 117 |
| <b>v m</b> .....   | 116 |
| <b>v R</b> .....   | 117 |
| <b>v SPC</b> ..... | 116 |
| <b>v t</b> .....   | 116 |
| <b>v w</b> .....   | 116 |
| <b>v y</b> .....   | 116 |

**W**

|                |     |
|----------------|-----|
| <b>w</b> ..... | 116 |
|----------------|-----|

**X**

|                |     |
|----------------|-----|
| <b>x</b> ..... | 123 |
| <b>X</b> ..... | 121 |

**Z**

|                |     |
|----------------|-----|
| <b>z</b> ..... | 120 |
|----------------|-----|

## Command and function index

### L

lisp-complete-symbol ..... 32

### N

next-error ..... 12

### O

org-agenda ..... 3, 5  
 org-agenda-add-note ..... 120  
 org-agenda-archive ..... 119  
 org-agenda-archive-default-with-confirmation ..... 119  
 org-agenda-archive-to-archive-sibling ... 119  
 org-agenda-archives-mode ..... 117  
 org-agenda-archives-mode 'files ..... 117  
 org-agenda-bulk-action ..... 121  
 org-agenda-bulk-mark ..... 121  
 org-agenda-bulk-mark-all ..... 121  
 org-agenda-bulk-mark-regexp ..... 121  
 org-agenda-bulk-remove-all-marks ..... 121  
 org-agenda-bulk-toggle ..... 121  
 org-agenda-bulk-toggle-all ..... 121  
 org-agenda-bulk-unmark ..... 121  
 org-agenda-capture ..... 121  
 org-agenda-clock-cancel ..... 121  
 org-agenda-clock-goto ..... 117, 121  
 org-agenda-clock-in ..... 121  
 org-agenda-clock-out ..... 121  
 org-agenda-clockreport-mode ..... 117  
 org-agenda-columns ..... 118, 129  
 org-agenda-convert-date ..... 123  
 org-agenda-date-prompt ..... 120  
 org-agenda-day-view ..... 116  
 org-agenda-deadline ..... 120  
 org-agenda-diary-entry ..... 123  
 org-agenda-do-date-earlier ..... 120  
 org-agenda-do-date-later ..... 120  
 org-agenda-drag-line-backward ..... 121  
 org-agenda-drag-line-forward ..... 121  
 org-agenda-earlier ..... 117  
 org-agenda-entry-text-mode ..... 118  
 org-agenda-exit ..... 123  
 org-agenda-file-to-front ..... 102  
 org-agenda-filter-by-category ..... 114, 118  
 org-agenda-filter-by-effort ..... 114, 119  
 org-agenda-filter-by-regexp ..... 114, 119  
 org-agenda-filter-by-tag ..... 113, 118  
 org-agenda-filter-by-top-headline ... 114, 119  
 org-agenda-filter-remove-all ..... 114, 119  
 org-agenda-follow-mode ..... 116  
 org-agenda-fortnight-view ..... 116  
 org-agenda-goto ..... 116

org-agenda-goto-calendar ..... 122  
 org-agenda-goto-date ..... 117  
 org-agenda-goto-today ..... 117  
 org-agenda-holidays ..... 123  
 org-agenda-kill ..... 119  
 org-agenda-later ..... 117  
 org-agenda-limit-interactively ..... 115  
 org-agenda-list ..... 104  
 org-agenda-list-stuck-projects ..... 110  
 org-agenda-log-mode ..... 117  
 org-agenda-manipulate-query-add ..... 117  
 org-agenda-month-view ..... 116  
 org-agenda-next-item ..... 115  
 org-agenda-next-line ..... 115  
 org-agenda-open-link ..... 116  
 org-agenda-phases-of-moon ..... 123  
 org-agenda-previous-item ..... 116  
 org-agenda-previous-line ..... 115  
 org-agenda-priority-down ..... 120  
 org-agenda-priority-up ..... 120  
 org-agenda-quit ..... 123  
 org-agenda-recenter ..... 116  
 org-agenda-redo ..... 118  
 org-agenda-refile ..... 119  
 org-agenda-remove-restriction-lock .. 103, 118  
 org-agenda-reset-view ..... 116  
 org-agenda-schedule ..... 120  
 org-agenda-set-restriction-lock ..... 103  
 org-agenda-set-tags ..... 120  
 org-agenda-show-and-scroll-up ..... 116  
 org-agenda-show-priority ..... 120  
 org-agenda-show-tags ..... 119  
 org-agenda-sunrise-sunset ..... 123  
 org-agenda-switch-to ..... 116  
 org-agenda-todo ..... 119  
 org-agenda-todo-nextset ..... 119  
 org-agenda-todo-previousset ..... 119  
 org-agenda-toggle-archive-tag ..... 119  
 org-agenda-toggle-diary ..... 117  
 org-agenda-toggle-time-grid ..... 118  
 org-agenda-tree-to-indirect-buffer ..... 116  
 org-agenda-undo ..... 119  
 org-agenda-week-view ..... 116  
 org-agenda-write ..... 123, 127  
 org-agenda-year-view ..... 116  
 org-archive-subtree ..... 100  
 org-archive-subtree-default ..... 100  
 org-archive-to-archive-sibling ..... 101  
 org-ascii-export-as-ascii ..... 145  
 org-ascii-export-to-ascii ..... 145  
 org-attach ..... 95, 120  
 org-attach-attach ..... 95  
 org-attach-delete-all ..... 96  
 org-attach-delete-one ..... 96  
 org-attach-new ..... 95

|   |             |  |           |
|---|-------------|--|-----------|
| org-attach-open.....                    | 95          | org-demote.....                            | 255       |
| org-attach-open-in-emacs.....           | 96          | org-demote-subtree.....                    | 10        |
| org-attach-reveal.....                  | 96          | org-do-demote.....                         | 9         |
| org-attach-reveal-in-emacs.....         | 96          | org-do-promote.....                        | 9         |
| org-attach-set-directory.....           | 96          | org-edit-special.....                      | 238       |
| org-attach-set-inherit.....             | 96          | org-entities-help.....                     | 134       |
| org-attach-sync.....                    | 95          | org-entry-add-to-multivalued-property...   | 253       |
| org-attach-url.....                     | 95          | org-entry-delete.....                      | 253       |
| org-backward-same-level.....            | 8           | org-entry-get.....                         | 253       |
| org-beamer-export-as-latex.....         | 147         | org-entry-get-multivalued-property.....    | 253       |
| org-beamer-export-to-latex.....         | 147         | org-entry-member-in-multivalued-property   |           |
| org-beamer-export-to-pdf.....           | 147         | .....                                      | 253       |
| org-beamer-select-environment.....      | 150         | org-entry-properties.....                  | 253       |
| org-buffer-property-keys.....           | 253         | org-entry-put.....                         | 253       |
| org-calendar-goto-agenda.....           | 122         | org-entry-put-multivalued-property.....    | 253       |
| org-capture.....                        | 3, 5, 89    | org-entry-remove-from-multivalued-property |           |
| org-capture-finalize.....               | 89          | .....                                      | 253       |
| org-capture-kill.....                   | 90          | org-evaluate-time-range.....               | 75, 82    |
| org-capture-refile.....                 | 89          | org-export-dispatch.....                   | 138       |
| org-check-after-date.....               | 79          | org-feed-goto-inbox.....                   | 96        |
| org-check-before-date.....              | 79          | org-feed-update-all.....                   | 96        |
| org-check-deadlines.....                | 78          | org-force-cycle-archived.....              | 101       |
| org-clock-cancel.....                   | 82          | org-forward-same-level.....                | 8         |
| org-clock-display.....                  | 82          | org-global-cycle.....                      | 7         |
| org-clock-goto.....                     | 82          | org-goto.....                              | 8         |
| org-clock-in.....                       | 81          | org-goto-calendar.....                     | 74        |
| org-clock-in-last.....                  | 82          | org-html-export-as-html.....               | 151       |
| org-clock-modify-effort-estimate.....   | 82, 87      | org-html-export-to-html.....               | 151       |
| org-clock-out.....                      | 81          | org-icalendar-combine-agenda-files.....    | 183       |
| org-clock-report.....                   | 83          | org-icalendar-export-agenda-files.....     | 183       |
| org-clock-timestamps-up/down.....       | 82          | org-icalendar-export-to-ics.....           | 183       |
| org-clocktable-try-shift.....           | 83          | org-insert-columns-dblock.....             | 72        |
| org-clone-subtree-with-time-shift.....  | 10          | org-insert-heading.....                    | 14, 88    |
| org-columns.....                        | 70          | org-insert-heading-respect-content.....    | 9         |
| org-columns-delete.....                 | 71          | org-insert-link.....                       | 42        |
| org-columns-edit-allowed.....           | 71          | org-insert-property-drawer.....            | 253       |
| org-columns-edit-value.....             | 70          | org-insert-todo-heading.....               | 9, 47, 58 |
| org-columns-narrow.....                 | 71          | org-insert-todo-heading-respect-content... | 9         |
| org-columns-new.....                    | 71          | org-latex-export-as-latex.....             | 158       |
| org-columns-next-allowed-value.....     | 70          | org-latex-export-to-latex.....             | 158       |
| org-columns-previous-allowed-value..... | 70          | org-latex-export-to-pdf.....               | 158       |
| org-columns-quit.....                   | 70          | org-lookup-all.....                        | 31        |
| org-columns-redo.....                   | 70          | org-lookup-first.....                      | 30        |
| org-columns-set-tags-or-toggle.....     | 70          | org-lookup-last.....                       | 31        |
| org-columns-show-value.....             | 71          | org-map-entries.....                       | 254       |
| org-columns-widen.....                  | 71          | org-mark-element.....                      | 10        |
| org-compute-property-at-point.....      | 66          | org-mark-ring-goto.....                    | 43        |
| org-copy.....                           | 99          | org-mark-ring-push.....                    | 43        |
| org-copy-subtree.....                   | 10          | org-mark-subtree.....                      | 10        |
| org-copy-visible.....                   | 7           | org-match-sparse-tree.....                 | 63, 66    |
| org-cut-subtree.....                    | 10          | org-md-export-as-markdown.....             | 166       |
| org-cycle.....                          | 7, 9, 13    | org-md-export-to-markdown.....             | 166       |
| org-cycle-agenda-files.....             | 103         | org-meta-return.....                       | 9         |
| org-date-from-calendar.....             | 74          | org-move-subtree-down.....                 | 10        |
| org-dblock-update.....                  | 72, 83, 248 | org-move-subtree-up.....                   | 10        |
| org-deadline.....                       | 78          | org-narrow-to-block.....                   | 11        |
| org-delete-property.....                | 66          | org-narrow-to-subtree.....                 | 11        |
| org-delete-property-globally.....       | 66          | org-next-link.....                         | 43        |

|  |         |  |             |
|--|---------|--|-------------|
| org-next-visible-heading.....                | 8       | org-table-fedit-abort.....               | 32          |
| org-occur.....                               | 11      | org-table-fedit-finish.....              | 32          |
| org-odt-export-to-odt.....                   | 166     | org-table-fedit-line-down.....           | 32          |
| org-open-at-point.....                       | 42, 74  | org-table-fedit-line-up.....             | 32          |
| org-org-export-as-org.....                   | 177     | org-table-fedit-lisp-indent.....         | 32          |
| org-org-export-to-org.....                   | 177     | org-table-fedit-ref-down.....            | 32          |
| org-paste-subtree.....                       | 10      | org-table-fedit-ref-left.....            | 32          |
| org-previous-link.....                       | 43      | org-table-fedit-ref-right.....           | 32          |
| org-previous-visible-heading.....            | 8       | org-table-fedit-ref-up.....              | 32          |
| org-priority.....                            | 55, 255 | org-table-fedit-scroll-down.....         | 32          |
| org-priority-down.....                       | 55      | org-table-fedit-scroll-up.....           | 32          |
| org-priority-up.....                         | 55      | org-table-fedit-toggle-ref-type.....     | 32          |
| org-promote.....                             | 255     | org-table-field-info.....                | 31          |
| org-promote-subtree.....                     | 9       | org-table-hline-and-move.....            | 20          |
| org-property-action.....                     | 65      | org-table-insert-column.....             | 20          |
| org-property-next-allowed-value.....         | 65      | org-table-insert-hline.....              | 20          |
| org-property-previous-allowed-value.....     | 65      | org-table-insert-row.....                | 20          |
| org-protocol-create.....                     | 99      | org-table-iterate.....                   | 33          |
| org-protocol-create-for-org.....             | 99      | org-table-iterate-buffer-tables.....     | 34          |
| org-publish.....                             | 197     | org-table-kill-row.....                  | 20          |
| org-publish-all.....                         | 198     | org-table-move-column-left.....          | 20          |
| org-publish-current-file.....                | 197     | org-table-move-column-right.....         | 20          |
| org-publish-current-project.....             | 197     | org-table-move-row-down.....             | 20          |
| org-refile.....                              | 10, 99  | org-table-move-row-up.....               | 20          |
| org-refile-cache-clear.....                  | 100     | org-table-next-field.....                | 20          |
| org-refile-goto-last-stored.....             | 99      | org-table-next-row.....                  | 20          |
| org-remove-file.....                         | 102     | org-table-paste-rectangle.....           | 21          |
| org-reveal.....                              | 7       | org-table-previous-field.....            | 20          |
| org-save-all-org-buffers.....                | 118     | org-table-recalculate.....               | 33          |
| org-schedule.....                            | 78      | org-table-recalculate-buffer-tables..... | 34          |
| org-search-view.....                         | 110     | org-table-rotate-recalc-marks.....       | 34          |
| org-set-effort.....                          | 87      | org-table-sort-lines.....                | 20          |
| org-set-property.....                        | 65      | org-table-sum.....                       | 21          |
| org-set-startup-visibility.....              | 7, 8    | org-table-toggle-coordinate-overlays..   | 31, 32      |
| org-set-tags-command.....                    | 59, 60  | org-table-toggle-formula-debugger.....   | 31          |
| org-show-todo-tree.....                      | 46      | org-table-wrap-region.....               | 21          |
| org-sort.....                                | 11      | org-tags-view.....                       | 63, 66, 107 |
| org-sparse-tree.....                         | 11      | org-texinfo-export-to-info.....          | 177         |
| org-speed-command-help.....                  | 230     | org-texinfo-export-to-texinfo.....       | 177         |
| org-speedbar-set-agenda-restriction.....     | 103     | org-time-stamp.....                      | 74          |
| org-store-agenda-views.....                  | 128     | org-time-stamp-inactive.....             | 74          |
| org-store-link.....                          | 3, 41   | org-timer.....                           | 88          |
| org-switchb.....                             | 3       | org-timer-item.....                      | 88          |
| org-table-align.....                         | 19      | org-timer-pause-or-continue.....         | 88          |
| org-table-beginning-of-field.....            | 20      | org-timer-set-timer.....                 | 88          |
| org-table-blank-field.....                   | 19      | org-timer-start.....                     | 88          |
| org-table-copy-down.....                     | 21      | org-timer-stop.....                      | 88          |
| org-table-copy-region.....                   | 21      | org-timestamp-down-day.....              | 74          |
| org-table-create-or-convert-from-region..... | 19, 22  | org-timestamp-down-down.....             | 74          |
| org-table-create-with-table.el.....          | 238     | org-timestamp-up.....                    | 74          |
| org-table-cut-region.....                    | 21      | org-timestamp-up-day.....                | 74          |
| org-table-delete-column.....                 | 20      | org-timestamp-up/down.....               | 82          |
| org-table-edit-field.....                    | 21      | org-todo.....                            | 46, 82, 255 |
| org-table-edit-formulas.....                 | 32      | org-todo-list.....                       | 47, 107     |
| org-table-end-of-field.....                  | 20      | org-toggle-archive-tag.....              | 101         |
| org-table-eval-formula.....                  | 30, 31  | org-toggle-checkbox.....                 | 57          |
| org-table-export.....                        | 22      | org-toggle-heading.....                  | 11          |
|  |         | org-toggle-inline-images.....            | 43          |

org-toggle-ordered-property ..... 50, 58  
org-toggle-tag ..... 255  
org-toggle-time-stamp-overlays ..... 77  
org-tree-to-indirect-buffer ..... 7  
org-update-all-dblocks ..... 72  
org-update-statistics-cookies ..... 58  
org-yank ..... 10  
outline-show-all ..... 7  
outline-show-branches ..... 7  
outline-show-children ..... 7

outline-up-heading ..... 8

## P

pcomplete ..... 65  
previous-error ..... 12

## W

widen ..... 11

## Variable index

This is not a complete index of variables and faces, only the ones that are mentioned in the manual. For a complete list, use *M-x org-customize RET*.

### C

cdlatex-simplify-sub-super-scripts..... 137  
constants-unit-system..... 26, 234

### H

htmlize-output-type..... 127

### L

~~La~~TeX-verbatim-environments..... 245

### O

org-adapt-indentation..... 236  
org-agenda-add-entry-text-maxlines..... 127  
org-agenda-bulk-custom-functions..... 121  
org-agenda-category-filter-preset..... 114  
org-agenda-clock-consistency-checks..... 118  
org-agenda-columns-add-appointments-to-  
  effort-sum..... 87  
org-agenda-confirm-kill..... 119  
org-agenda-custom-commands.. 12, 124, 125, 126,  
  251  
org-agenda-custom-commands-contexts..... 126  
org-agenda-diary-file..... 123  
org-agenda-dim-blocked-tasks..... 51, 251  
org-agenda-effort-filter-preset..... 114  
org-agenda-entry-text-maxlines..... 118  
org-agenda-exporter-settings..... 123, 127  
org-agenda-files..... 102, 112, 183  
org-agenda-inhibit-startup..... 8, 251  
org-agenda-log-mode-items..... 117  
org-agenda-max-effort..... 115  
org-agenda-max-entries..... 115  
org-agenda-max-tags..... 115  
org-agenda-max-todos..... 115  
org-agenda-ndays..... 104  
org-agenda-overriding-header..... 249  
org-agenda-prefix-format..... 111  
org-agenda-regexp-filter-preset..... 114  
org-agenda-restore-windows-after-quit... 102  
org-agenda-search-headline-for-time..... 111  
org-agenda-show-inherited-tags..... 120, 251  
org-agenda-skip-archived-trees..... 101, 102  
org-agenda-skip-comment-trees..... 102  
org-agenda-skip-deadline-prewarning-if-  
  scheduled..... 77  
org-agenda-skip-function..... 249, 254  
org-agenda-skip-function-global..... 249

org-agenda-skip-scheduled-delay-if-deadline  
  ..... 78  
org-agenda-skip-scheduled-if-deadline-is-  
  shown..... 80  
org-agenda-skip-scheduled-if-done..... 78  
org-agenda-sorting-strategy..... 112  
org-agenda-span..... 104, 116  
org-agenda-start-day..... 104  
org-agenda-start-on-weekday..... 104  
org-agenda-start-with-clockreport-mode.. 117  
org-agenda-start-with-entry-text-mode... 118  
org-agenda-start-with-follow-mode..... 116  
org-agenda-sticky..... 104  
org-agenda-tag-filter-preset..... 113  
org-agenda-tags-column..... 111  
org-agenda-tags-todo-honor-ignore-options  
  ..... 108  
org-agenda-text-search-extra-files.. 104, 110  
org-agenda-time-grid..... 112, 118  
org-agenda-todo-ignore-deadlines..... 107  
org-agenda-todo-ignore-scheduled..... 107  
org-agenda-todo-ignore-timestamp..... 107  
org-agenda-todo-ignore-with-date..... 107  
org-agenda-todo-list-sublevels..... 55, 107  
org-agenda-use-tag-inheritance..... 59, 251  
org-agenda-use-time-grid..... 112, 118  
org-agenda-window-setup..... 102  
org-archive-default-command..... 100, 119  
org-archive-location..... 100, 231  
org-archive-save-context-info..... 101  
org-ascii-links-to-notes..... 145  
org-ascii-text-width..... 145  
org-attach-directory..... 95  
org-attach-method..... 95  
org-babel-default-header-args..... 206, 207  
org-babel-load-languages..... 205  
org-beamer-environments-default..... 148  
org-beamer-environments-extra..... 148  
org-beamer-frame-level..... 148  
org-beamer-subtitle-format..... 148  
org-beamer-theme..... 147  
org-calc-default-modes..... 26  
org-capture-bookmark..... 90  
org-capture-templates..... 90  
org-capture-templates-contexts..... 94  
org-capture-use-agenda-date..... 121  
org-catch-invisible-edits..... 8  
org-checkbox-hierarchical-statistics..... 57  
org-clock-continuously..... 81, 82, 86  
org-clock-idle-time..... 85  
org-clock-into-drawer..... 81  
org-clock-modeline-total..... 81

- org-clock-report-include-clocking-task .. 117
- org-clock-x11idle-program-name .. 85
- org-clocktable-defaults .. 83
- org-closed-keep-when-no-todo .. 52
- org-coderef-label-format .. 134
- org-columns-default-format .... 70, 87, 118, 129
- org-columns-skip-archived-trees .. 101
- org-columns-summary-types .. 69
- org-confirm-babel-evaluate .. 231
- org-confirm-elisp-link-function .. 231
- org-confirm-shell-link-function .. 231
- org-create-file-search-functions .. 45
- org-ctrl-c-ctrl-c-hook .. 244
- org-ctrl-k-protect-subtree .. 6
- org-cycle-emulate-tab .. 7
- org-cycle-global-at-bob .. 7
- org-cycle-include-plain-lists .. 13
- org-cycle-separator-lines .. 6
- org-deadline-warning-days .. 77, 79
- org-default-notes-file .. 89, 91
- org-default-priority .. 55, 232
- org-display-custom-times .. 77
- org-display-internal-link-with-indirect-  
buffer .. 43
- org-disputed-keys .. 239
- org-done (face) .. 49
- org-edit-footnote-reference .. 17
- org-edit-src-auto-save-idle-delay .. 201
- org-edit-src-turn-on-auto-save .. 201
- org-emphasis-alist .. 131
- org-emphasis-regexp-components .. 131
- org-enforce-todo-dependencies .. 50, 51
- org-entities-user .. 134
- org-execute-file-search-functions .. 45
- org-export-allow-bind-keywords .. 142
- org-export-async-init-file .. 139
- org-export-backends .. 138, 184
- org-export-before-parsing-hook .. 184
- org-export-before-processing-hook .. 184
- org-export-creator-string .. 140, 153
- org-export-date-timestamp-format .. 140
- org-export-default-language .. 140
- org-export-dispatch-use-expert-ui .. 138
- org-export-exclude-tags .. 140
- org-export-global-macros .. 144
- org-export-headline-levels .. 141
- org-export-in-background .. 139
- org-export-initial-scope .. 139
- org-export-preserve-breaks .. 140
- org-export-select-tags .. 140
- org-export-time-stamp-file .. 142, 153
- org-export-use-babel .. 202
- org-export-with-archived-trees .. 101, 141
- org-export-with-author .. 141
- org-export-with-broken-links .. 141
- org-export-with-clocks .. 141
- org-export-with-creator .. 141
- org-export-with-date .. 141
- org-export-with-drawers .. 16, 141
- org-export-with-email .. 141
- org-export-with-entities .. 141
- org-export-with-fixed-width .. 140
- org-export-with-footnotes .. 141
- org-export-with-inlinetasks .. 141
- org-export-with-latex .. 136, 141
- org-export-with-planning .. 141
- org-export-with-priority .. 141
- org-export-with-properties .. 16, 141
- org-export-with-section-numbers .. 141
- org-export-with-smart-quotes .. 140
- org-export-with-special-strings .. 140
- org-export-with-statistics-cookies .. 141
- org-export-with-sub-superscripts .. 140
- org-export-with-tables .. 142
- org-export-with-tags .. 141
- org-export-with-tasks .. 141
- org-export-with-timestamps .. 140
- org-export-with-title .. 142
- org-export-with-toc .. 142
- org-export-with-todo-keywords .. 142
- org-fast-tag-selection-include-todo .. 49
- org-fast-tag-selection-single-key .. 61
- org-file-apps .. 42, 96
- org-fontify-emphasized-text .. 131
- org-footnote-auto-adjust .. 16, 234
- org-footnote-auto-label .. 16, 234
- org-footnote-define-inline .. 16, 234
- org-footnote-section .. 6, 16
- org-format-latex-header .. 135, 136
- org-format-latex-options .. 136
- org-from-is-user-regexp .. 94
- org-global-properties .. 65, 87
- org-goto-auto-isearch .. 8
- org-goto-interface .. 9
- org-group-tags .. 63
- org-hide (face) .. 236
- org-hide-block-startup .. 16
- org-hide-leading-stars .. 233, 236
- org-hierarchical-todo-statistics .. 56
- org-highest-priority .. 55, 232
- org-html-container-element .. 151
- org-html-doctype .. 151, 152
- org-html-doctype-alist .. 152
- org-html-head .. 151, 157
- org-html-head-extra .. 151, 157
- org-html-head-include-default-style .. 157
- org-html-html5-elements .. 153
- org-html-html5-fancy .. 152
- org-html-infojs-options .. 158
- org-html-inline-images .. 154
- org-html-link-home .. 151
- org-html-link-org-files-as-html .. 153
- org-html-link-up .. 151
- org-html-mathjax-options .. 151
- org-html-postamble .. 153
- org-html-postamble-format .. 153



|  |              |   |                    |
|--|--------------|---|--------------------|
| org-html-preamble .....                    | 153          | org-log-repeat .....                        | 79, 233            |
| org-html-preamble-format .....             | 153          | org-log-states-order-reversed .....         | 52                 |
| org-html-style-default .....               | 157          | org-lowest-priority .....                   | 55, 232            |
| org-html-table-align-individual-fields ..  | 154          | org-M-RET-may-split-line .....              | 9, 14              |
| org-html-table-caption-above .....         | 154          | org-md-headline-style .....                 | 166                |
| org-html-table-data-tags .....             | 154          | org-odd-levels-only .....                   | 108, 233, 237, 249 |
| org-html-table-default-attributes .....    | 154          | org-odt-category-map-alist .....            | 172                |
| org-html-table-header-tags .....           | 154          | org-odt-convert .....                       | 167                |
| org-html-table-row-tags .....              | 154          | org-odt-convert-capabilities .....          | 172                |
| org-html-table-use-header-tags-for-first-  |              | org-odt-convert-process .....               | 172                |
| column .....                               | 154          | org-odt-convert-processes .....             | 172                |
| org-html-tag-class-prefix .....            | 156          | org-odt-create-custom-styles-for-srcblocks  |                    |
| org-html-todo-kwd-class-prefix .....       | 156          | .....                                       | 172                |
| org-html-use-infojs .....                  | 158          | org-odt-fontify-srcblocks .....             | 172                |
| org-html-validation-link .....             | 153          | org-odt-pixels-per-inch .....               | 169                |
| org-icalendar-alarm-time .....             | 183          | org-odt-preferred-output-format .....       | 166, 167           |
| org-icalendar-categories .....             | 183          | org-odt-schema-dir .....                    | 177                |
| org-icalendar-combined-agenda-file .....   | 184          | org-odt-styles-file .....                   | 167, 168           |
| org-icalendar-include-body .....           | 184          | org-odt-table-styles .....                  | 175, 176           |
| org-icalendar-include-todo .....           | 183          | org-outline-path-complete-in-steps .....    | 99                 |
| org-icalendar-store-UID .....              | 183          | org-overriding-columns-format .....         | 129                |
| org-icalendar-use-deadline .....           | 183          | org-plain-list-ordered-item-terminator ...  | 12,                |
| org-icalendar-use-scheduled .....          | 183          | 14  |                    |
| org-id-link-to-org-use-id .....            | 41           | org-popup-calendar-for-date-prompt .....    | 76                 |
| org-imenu-depth .....                      | 238          | org-preview-latex-default-process .....     | 136                |
| org-insert-mode-line-in-empty-file .....   | 3            | org-priority-faces .....                    | 55                 |
| org-irc-link-to-logs .....                 | 41           | org-priority-start-cycle-with-default ..... | 55                 |
| org-keep-stored-link-after-insertion ..... | 42           | org-property-allowed-value-functions .....  | 253                |
| org-latex-bibtex-compiler .....            | 158          | org-protocol-default-template-key .....     | 98                 |
| org-latex-classes .....                    | 159, 160     | org-protocol-project-alist .....            | 98                 |
| org-latex-compiler .....                   | 158, 159     | org-publish-project-alist .....             | 188, 190           |
| org-latex-default-class .....              | 159, 160     | org-publish-use-timestamps-flag .....       | 198                |
| org-latex-default-packages-alist ....      | 158, 160     | org-put-time-stamp-overlays .....           | 234                |
| org-latex-default-table-environment .....  | 161          | org-read-date-display-live .....            | 76                 |
| org-latex-default-table-mode .....         | 161          | org-read-date-force-compatible-dates .....  | 76                 |
| org-latex-images-centered .....            | 163          | org-read-date-prefer-future .....           | 75                 |
| org-latex-listings .....                   | 132          | org-refile-allow-creating-parent-nodes ...  | 99                 |
| org-latex-listings-options .....           | 164          | org-refile-keep .....                       | 99                 |
| org-latex-minted-options .....             | 164          | org-refile-targets .....                    | 99                 |
| org-latex-packages-alist .....             | 160          | org-refile-use-cache .....                  | 99                 |
| org-latex-subtitle-format .....            | 160          | org-refile-use-outline-path .....           | 99                 |
| org-latex-subtitle-separate .....          | 160          | org-remove-highlights-with-change .....     | 12, 82             |
| org-latex-tables-booktabs .....            | 162          | org-replace-disputed-keys .....             | 239                |
| org-latex-tables-centered .....            | 162          | org-return-follows-link .....               | 43                 |
| org-latex-to-mathml-convert-command .....  | 170          | org-reverse-note-order .....                | 99                 |
| org-latex-to-mathml-jar-file .....         | 170          | org-scheduled-delay-days .....              | 78                 |
| org-link-abbrev-alist .....                | 44, 232      | org-show-context-detail .....               | 11                 |
| org-link-frame-setup .....                 | 42           | org-sort-agenda-noeffort-is-high .....      | 114                |
| org-link-parameters .....                  | 42           | org-sparse-tree-open-archived-trees .....   | 101                |
| org-list-allow-alphabetical .....          | 12           | org-special-ctrl-a/e .....                  | 6                  |
| org-list-automatic-rules .....             | 13, 14, 56   | org-special-ctrl-k .....                    | 6                  |
| org-list-demote-modify-bullet .....        | 13           | org-speed-commands-user .....               | 230                |
| org-list-indent-offset .....               | 13           | org-startup-align-all-tables .....          | 23, 233            |
| org-list-use-circular-motion .....         | 14           | org-startup-folded .....                    | 8, 232, 251        |
| org-log-done .....                         | 53, 117, 233 | org-startup-indented .....                  | 233                |
| org-log-into-drawer .....                  | 52, 120      | org-startup-with-inline-images .....        | 43, 233            |
| org-log-note-clock-out .....               | 82, 233      | org-startup-with-latex-preview .....        | 137, 233           |
| org-log-refile .....                       | 99           | org-store-link-functions .....              | 243                |

|   |                  |   |                   |
|---|------------------|---|-------------------|
| org-stuck-projects .....                | 110              | org-todo-state-tags-triggers .....          | 47                |
| org-support-shift-select .....          | 14, 15, 239      | org-track-ordered-property-with-tag ...     | 50, 58            |
| org-sytle-open-archived-trees .....     | 101              | org-treat-insert-todo-heading-as-state-     |                   |
| org-table-auto-blank-field .....        | 19               | change .....                                | 9                 |
| org-table-copy-increment .....          | 21               | org-treat-S-cursor-todo-selection-as-state- |                   |
| org-table-duration-custom-format .....  | 29               | change .....                                | 46                |
| org-table-export-default-format .....   | 22               | org-use-fast-todo-selection .....           | 46                |
| org-table-formula .....                 | 232              | org-use-property-inheritance .....          | 67, 184, 207, 253 |
| org-table-formula-constants .....       | 26, 232, 238     | org-use-speed-commands .....                | 230               |
| org-table-use-standard-references ..... | 24, 31           | org-use-sub-superscripts .....              | 135               |
| org-tag-alist .....                     | 60, 234          | org-use-tag-inheritance .....               | 59                |
| org-tag-faces .....                     | 59               | org-yank-adjusted-subtrees .....            | 10                |
| org-tag-persistent-alist .....          | 60               | org-yank-folded-subtrees .....              | 10                |
| org-tags-column .....                   | 59               | orgstruct-heading-prefix-regexp .....       | 18                |
| org-tags-exclude-from-inheritance ..... | 59               |   |                   |
| org-tags-match-list-sublevels ..        | 59, 63, 67, 108  |   |                   |
| org-texinfo-classes .....               | 178, 179         |   |                   |
| org-texinfo-coding-system .....         | 178              |   |                   |
| org-texinfo-default-class .....         | 178, 179         |   |                   |
| org-texinfo-info-process .....          | 177              |   |                   |
| org-texinfo-table-default-markup .....  | 180              |   |                   |
| org-time-stamp-custom-formats .....     | 77               |   |                   |
| org-time-stamp-overlay-formats .....    | 234              |   |                   |
| org-time-stamp-rounding-minutes .....   | 74               |   |                   |
| org-todo (face) .....                   | 49               |   |                   |
| org-todo-keyword-faces .....            | 49               |   |                   |
| org-todo-keywords .....                 | 46, 47, 107, 235 |   |                   |
| org-todo-repeat-to-state .....          | 79               |   |                   |

## P

|                            |     |
|----------------------------|-----|
| parse-time-months .....    | 76  |
| parse-time-weekdays .....  | 76  |
| ps-landscape-mode .....    | 127 |
| ps-number-of-columns ..... | 127 |

## U

|                         |     |
|-------------------------|-----|
| user-full-name .....    | 139 |
| user-mail-address ..... | 140 |