mfirstuc.sty v2.08: sentence and title casing

Nicola L.C. Talbot

Dickimaw Books

dickimaw-books.com

2022-10-14

This document is also available as HTML (`mfirstuc-manual.html`).

The mfirstuc package was originally part of the glossaries bundle (added to glossaries v1.12 in 2008) for use with commands like `\Gls`, which need to have the first letter converted to uppercase (sentence-case), but as the commands provided by mfirstuc may be used without glossaries, the two were split into separately maintained packages in 2015 (mfirstuc v2.0 and glossaries v4.18).

Version 2.08 has reimplemented `\makefirstuc` using LaTeX3 commands. If there are any compatibility issues, you can rollback to v2.07:

```
\usepackage{mfirstuc}[=2.07]
```

Contents

1 Sentence-Case

There are two sentence-case commands provided: `\MFUsentencecase` and `\makefirstuc`. A summary of the principle features of the two commands is given in Table 1.1.

Table 1.1: Principle features of `\MFUsentencecase` and `\makefirstuc`

Feature	`\MFUsentencecase`	`\makefirstuc`
Can expand?	✔	�’
Supports exclusions?	✔	✔
Supports blockers?	✗	✔
Supports mappings?	✗	✔
Semantic commands must be robust?	✔	✗

1.1 Examples

Text only (leading UTF-8 now supported):

`\makefirstuc{élite} /` `\MFUsentencecase{élite}`	Élite / Élite

Leading punctuation is ignored:

`` \makefirstuc{`word'} / `` `` \MFUsentencecase{`word'} ``	'Word' / 'Word'

However, if the punctuation character is followed by a blocker or mapping command, the punctuation will need to be excluded otherwise the command will be treated as an exclusion. If possible, use semantic markup instead of directly including the punctuation.

Fragile commands need to be protected with `\MFUsentencecase`:

```
\newcommand{\alert}[1]{\textcolor{red}{\textbf{#1}}}
\makefirstuc{\alert{élite}} /
\MFUsentencecase{\protect\alert{élite}}
```

Élite / **Élite**

Exclusions are supported by both \makefirstuc and \MFUsentencecase:

```
\MFUexcl{\index}
\makefirstuc{\index{word}example} /
\MFUsentencecase{\index{word}example}
```

Example / Example

Blockers are only supported by \makefirstuc. They are treated as exclusions with \MFU-sentencecase, which produces a different result:

```
\MFUblocker{\nameref}
\makefirstuc{\nameref{sec:sentencecase} section} /
\MFUsentencecase{\nameref{sec:sentencecase} section}
```

Sentence-Case section / Sentence-Case Section

Mappings are only supported by \makefirstuc. They are treated as exclusions with \MFU-sentencecase, which produces a different result:

```
\newrobustcmd{\secref}[1]{section~\ref{#1}}
\newrobustcmd{\Secref}[1]{Section~\ref{#1}}
\MFUaddmap{\secref}{\Secref}
\makefirstuc{\secref{sec:sentencecase} example} /
\MFUsentencecase{\secref{sec:sentencecase} example}
```

Section 1 example / section 1 Example

Argument expansion: \MFUsentencecase expands its argument and then skips exclusions whereas \makefirstuc parses its argument without expansion and then passes the relevant content to \glsmakefirstuc. Note that different results will occur with the expanded and unexpanded package options. For example:

```
\usepackage[expanded]{mfirstuc}
\newcommand{\testa}{sample}
\newcommand{\testb}{\testa\ test}
\begin{document}
\makefirstuc{\testb} / \xmakefirstuc{\testb} /
\emakefirstuc{\testb} / \MFUsentencecase{\testb}
\end{document}
```

Sample test / Sample test / Sample test / Sample test

With the default unexpanded option the result is:

sample test / sample Test / Sample test / Sample test

Note that this is different from the result in earlier versions of mfirstuc.

```
\usepackage{mfirstuc}[=v2.07]
\newcommand{\testa}{sample}
\newcommand{\testb}{\testa\ test}
\begin{document}
\makefirstuc{\testb} / \xmakefirstuc{\testb} /
\emakefirstuc{\testb}
\end{document}
```

This produces:

SAMPLE TEST / SAMPLE test / Sample test

If this old result is required, either use rollback or try the grabfirst option.

Argument expansion with mappings (and the default unexpanded option):

```
\newrobustcmd{\secref}[1]{section~\ref{#1}}
\newrobustcmd{\Secref}[1]{Section~\ref{#1}}
\MFUaddmap{\secref}{\Secref}
\newcommand{\testa}{\secref{sec:sentencecase}}
\newcommand{\testb}{\testa\ example}
\makefirstuc{\testb} / \xmakefirstuc{\testb} /
```

```
\emakefirstuc{\testb} / \MFUsentencecase{\testb}
```

section 1 example / section 1 Example / Section 1 example / section 1 Example

1.2 Sentence Case Commands

```
\MFUsentencecase{⟨text⟩}
```

This simply uses `\text_titlecase_first:n`, which is provided by the LaTeX3 kernel. Note that this fully expands the argument. If you use this command, ensure that your semantic commands are robust. For example:

```
\newrobustcmd{\alert}[1]{\textcolor{red}{\textbf{#1}}}
\MFUsentencecase{\alert{sample} text}
```

Sample text

Note that the following will fail:

```
\newcommand{\alert}[1]{\textcolor{red}{\textbf{#1}}}
\MFUsentencecase{\alert{sample} text}
```

This is because the expansion that's performed before the case-change will result in:

```
\textcolor{Red}{\textbf{sample}} text
```

This triggers an error since `Red` isn't a recognised colour name.

```
\makefirstuc{⟨text⟩}
```

This is the original command provided by mfirstuc, which was written to assist sentence-casing in the glossaries package and had to take markup into account. It parses its argument to try to determine which part needs the case-change. The actual case-change is performed by `\glsmakefirstuc`.

As from v2.08, the definition of \makefirstuc has been updated to use LaTeX3 code, but the argument is still parsed, which allows for non-robust semantic commands, and now also takes blockers and mappings into account. For example:

```
\newcommand{\alert}[1]{\textcolor{red}{\textbf{#1}}}
\makefirstuc{\alert{sample} text}
```

Sample text

The rules used when parsing \makefirstuc{⟨*text*⟩} are as follows:

1. if ⟨*text*⟩ is empty do nothing;

2. if ⟨*text*⟩ consists of a single item or starts with a group do \glsmakefirstuc{⟨*text*⟩};

3. if ⟨*text*⟩ starts with a command ⟨*cs*⟩:

    a) if ⟨*cs*⟩ is \protect, skip it and continue to the next step where the command ⟨*cs*⟩ under inspection is now the one that follows \protect:

    b) if ⟨*cs*⟩ has been identified as a blocker, just do ⟨*text*⟩ (that is, no case-change is applied);

    c) if ⟨*cs*⟩ isn't followed by a group then do \glsmakefirstuc{⟨*text*⟩};

    d) otherwise assume the format is ⟨*cs*⟩{⟨*arg*⟩}⟨*following*⟩ (where ⟨*following*⟩ may be empty) and then:

        i. if ⟨*cs*⟩ has been mapped to ⟨*Cs*⟩ then do ⟨*Cs*⟩{⟨*arg*⟩}⟨*following*⟩;

        ii. if ⟨*cs*⟩ has been identified as an exclusion, do ⟨*cs*⟩{⟨*arg*⟩}\makefirstuc {⟨*following*⟩};

        iii. otherwise do ⟨*cs*⟩{\makefirstuc{⟨*arg*⟩}}⟨*following*⟩.

4. otherwise do \glsmakefirstuc{⟨*text*⟩}.

Note that the above algorithm uses recursion. The termination points are: don't implement a change, replace a command with another command, or apply the case-change via \gls-makefirstuc.

For convenience, the following commands are also provided:

\xmakefirstuc{⟨*text*⟩}

This is simply a shortcut for:

```
\expandafter\makefirstuc\expandafter{⟨text⟩}
```

This ensures the first token in ⟨*text*⟩ is expanded before being passed to \makefirstuc.

```
\emakefirstuc{⟨text⟩}
```

This fully expands its argument before passing it to \makefirstuc. For example:

```
\newcommand{\test}[1]{sample #1}
\newcommand{\tmp}{\test{text}}
\makefirstuc{\tmp} /
\xmakefirstuc{\tmp} /
\emakefirstuc{\tmp}
```

```
sample text / sample Text / Sample text
```

If you don't have any mappings or blockers set up, it's simpler to use \MFUsentencecase instead of \emakefirstuc.

> With \emakefirstuc, mappings and blockers will only be detected if they are robust or protected, otherwise they will expand too soon to be detected.

```
\glsmakefirstuc{⟨text⟩}
```

This command is used by \makefirstuc to perform the case-change. The "gls" prefix is for historical reasons as the original code was part of the glossaries package. With the default unexpanded option, this command is defined as:

```
\MFUsentencecase{\unexpanded{⟨text⟩}}
```

The use of \unexpanded inhibits the expansion of ⟨*text*⟩ that would otherwise occur. This allows \makefirstuc to maintain as much backward-compatibility with version 2.07 as possible. This is particularly important when using the glossaries package with a style that automatically converts the description to sentence-case, as there may well be documents with complex descriptions that contain fragile commands.

The expanded package option will redefine \glsmakefirstuc to use \MFUsentencecase without \unexpanded. The grabfirst package option will redefine \glsmakefirstuc to

use \mfugrabfirstuc to emulates the behaviour of \glsmakefirstuc in v2.07 and below without using rollback.

\mfugrabfirstuc{⟨*text*⟩}

avoid where possible

This robust command is provided to emulate v2.07 and its use is discouraged. Note that you may still have different results compared to v2.07, so you may need to use rollback instead. (Note that the grabfirst option additionally redefines \mfirstucMakeUppercase to \MakeUppercase, which you will also need.)

This applies \mfirstucMakeUppercase to the first token in ⟨*text*⟩ and leaves the remainder unchanged. This won't work if ⟨*text*⟩ starts with a UTF-8 character unless you use X⅁LATEX or LuaLATEX, so avoid using this method where possible. For example:

```
\renewcommand{\mfirstucMakeUppercase}[1]{%
  \MakeUppercase{#1}%
}
\mfugrabfirstuc{sample}
```

This is equivalent to:

```
\mfirstucMakeUppercase{s}ample
```

Whereas

```
\renewcommand{\mfirstucMakeUppercase}[1]{%
  \MakeUppercase{#1}%
}
\mfugrabfirstuc{élite}
```

only works with X⅁LATEX or LuaLATEX.

As with earlier implementations of \makefirstuc that tried to only apply the case-change to the first token, this won't produce the desired result if the argument starts with a punctuation character.

For example:

```
\mfugrabfirstuc{`sample'}
```

This is equivalent to:

```
\mfirstucMakeUppercase{`}sample'
```

so no case-change is applied.

1.3 Exclusions

> Exclusions will apply to all LaTeX3 case-changing commands, so will also affect \mfirstucMakeUppercase, but this is generally desirable.

```
\MFUexcl{⟨cs⟩}
```

Identifies the text-block command ⟨*cs*⟩{⟨*arg*⟩} as an exclusion. That is, one that should have its argument excluded from a case-change.

Internally, \MFUexcl adds ⟨*cs*⟩ to LaTeX3's exclusion list \l_text_case_exclude_arg_tl, which means that \MFUsentencecase will skip the command and its argument and apply the case-change to the following content. Some common commands (\begin \cite \end \label and \ref) are automatically added by the LaTeX3 kernel. The exclusions are also recognised by \makefirstuc when it parses its argument.

> The command \NoCaseChange is automatically treated as an exclusion by the LaTeX3 case-changing commands with recent LaTeX kernels. You may want to add it as a blocker for \makefirstuc and keep \MFUskippunc for exclusions.
> Note that \ensuremath is added as an exclusion by mfirstuc otherwise \makefirstuc will pass its argument to \MFUsentencecase, which will cause a problem.

Another way of identifying content that should be excluded with \makefirstuc is to encapsulate it with:

```
\MFUskippunc{⟨content⟩}
```

This is a robust command that simply does its argument and is automatically added to the exclusion list. This may be used for cases where the excluded command isn't a simple text-

block command that only has one mandatory argument or where leading punctuation is followed by a blocker or mapping.

The command name stems from earlier versions where leading punctuation needed to be skipped. This is no longer necessary, but the command remains as a general purpose robust exclusion command.

For example, `\glsadd` (provided by glossaries) should have its argument skipped, since its argument is a label, so the following will ensure that `\glsadd{`⟨*label*⟩`}` will be skipped, and the case-change will be applied to the following text for both `\MFUsentencecase` and `\makefirstuc`.

```
\MFUexcl{\glsadd}
\makefirstuc{\glsadd{ex}some text}
\MFUsentencecase{\glsadd{ex}some text}
```

This will be equivalent to:

```
\glsadd{ex}Some text
```

However, `\glsadd` can take an optional argument which, if present, will cause a problem. For example:

```
\MFUexcl{\glsadd}
\makefirstuc{\glsadd[counter=section]{ex}some text}
\MFUsentencecase{\glsadd[counter=section]{ex}some text}
```

Both commands will cause the following error:

```
! Package glossaries Error: Glossary entry `[' has not been
defined.
```

This is because the open square bracket is assumed to be the argument of `\glsadd`, so this effectively becomes:

```
\glsadd{[}Counter=section] exsome text
\glsadd{[}Counter=section] exsome text
```

which means that now [is considered the label and everything that follows is just text.

For this situation to work, you need to encapsulate the problematic content with an exclusion command, such as `\NoCaseChange` or `\MFUskippunc`:

```
\makefirstuc{\MFUskippunc{\glsadd[counter=section]{ex}}some text}
```

This will also work with \MFUsentencecase because mfirstuc automatically adds \MFUskip-punc to the exclusion list.

> As from glossaries v4.50, \glsadd is automatically added as an exclusion, but be aware of the problem with using the optional argument, as described above.

Punctuation characters are skipped by \MFUsentencecase. For example:

| `\MFUsentencecase{`word'}` | 'Word' |

However, leading punctuation will result in \makefirstuc passing its entire argument to \MFUsentencecase (since the argument doesn't start with $\langle cs\rangle\{\langle arg\rangle\}$), which means that \makefirstuc won't detect any blocker or mapping.

For example:

```
\MFUblocker{\nameref}
\newcommand*{\qt}[1]{``#1''}
\makefirstuc{``\nameref{sec:sentencecase} section''} /
\makefirstuc{\MFUskippunc{``}\nameref{sec:sentencecase} section''}
/
\makefirstuc{\qt{\nameref{sec:sentencecase} section}}
```

"Sentence-Case Section" / "Sentence-Case section" / "Sentence-Case section"

If you are using a package such as csquotes, bear in mind that if the command is followed by an asterisk (a "starred command") or an optional argument then it won't fit the expected $\langle cs\rangle\{\langle arg\rangle\}$ format and unexpected results will occur.

1.4 Blockers

A blocker is a command that prevents any case-change if it occurs at the start of \makefirst-uc or after the argument of an exclusion. Blockers are not supported by \MFUsentencecase

but are instead treated as exclusions.

```
\MFUblocker{⟨cs⟩}
```

Identifies ⟨cs⟩ as a blocker. Since blockers aren't supported by \MFUsentencecase, this automatically implements \MFUexcl{⟨cs⟩} to protect its argument from \MFUsentencecase, but it won't prevent subsequent content from being changed.

For example:

```
\MFUblocker{\nameref}
\makefirstuc{\nameref{sec:sentencecase} section}
```

Sentence-Case section

In the following, the blocker isn't detected by \makefirstuc because the content doesn't start with ⟨cs⟩{⟨arg⟩} or \protect⟨cs⟩{⟨arg⟩}. This means that the entire content is passed to \MFUsentencecase which treats \nameref as an exclusion:

```
\MFUblocker{\nameref}
\makefirstuc{\relax\nameref{sec:sentencecase} section}
```

Sentence-Case Section

If a blocking command is inside the definition of another command, it won't be visible to \makefirstuc unless the other command is expanded before applying \makefirstuc. For example, by using \xmakefirstuc or \emakefirstuc. If it doesn't get expanded until after it has been passed to \MFUsentencecase, then it will be treated as an exclusion instead.

Recent LaTeX kernels provide \NoCaseChange and automatically add it as an exclusion. You may want to add it as a blocker for \makefirstuc. For example:

```
\makefirstuc{\ensuremath{\alpha}-particle} /
\makefirstuc{$\alpha$-particle} /
\MFUsentencecase{\ensuremath{\alpha}-particle} /
```

```
\MFUsentencecase{$\alpha$-particle}

\MFUblocker{\NoCaseChange}%
\makefirstuc{\ensuremath{\alpha}-particle} /
\makefirstuc{\NoCaseChange{$\alpha$}-particle} /
\MFUsentencecase{\ensuremath{\alpha}-particle} /
\MFUsentencecase{\NoCaseChange{$\alpha$}-particle}
```

Note that mfirstuc adds `\ensuremath` as an exclusion so `\makefirstuc` will skip it. Whereas the example above that starts with $ will have the entire argument passed to `\MFUsentence-case`, which skips the maths content. In the first paragraph, `\NoCaseChange` is only an exclusion, but in the second paragraph it's also a blocker.

Exclusion: α-Particle / α-Particle / α-Particle / α-Particle
Blocker: α-Particle / α-particle / α-Particle / α-Particle

1.5 Mappings

A mapping indicates that one command should be substituted for another, instead of applying a case-change. The assumption is that the substituted command should perform the case-change instead. Mappings are not supported by `\MFUsentencecase` but are instead treated as exclusions.

```
\MFUaddmap{⟨cs1⟩}{⟨cs2⟩}
```

Identifies a mapping which indicates that `\makefirstuc` should replace ⟨*cs1*⟩ with ⟨*cs2*⟩ and not apply a case-change. This automatically implements:

```
\MFUexcl{⟨cs1⟩}\MFUblocker{⟨cs2⟩}
```

This means that ⟨*cs2*⟩ is identified as a blocker (since it's assumed to already be a sentence-case command) and ⟨*cs1*⟩ is identified as an exclusion to protect its argument from `\MFUsen-tencecase`, which doesn't support mappings.

If either ⟨*cs1*⟩ or ⟨*cs2*⟩ is empty, no mapping is established, but an exclusion or blocker will be set for the non-empty argument.

For example (with glossaries):

```
\MFUaddmap{\gls}{\Gls}
\makefirstuc{\gls{ex} some text}
```

This will be converted to:

```
\Gls{ex} some text
```

Note that this and similar mappings are automatically added in glossaries v4.50+ and glossaries –extra v1.49+.

> If a mapped command is inside the definition of another command, it won't be visible to \makefirstuc unless the other command is expanded before applying \makefirstuc. For example, by using \xmakefirstuc or \emakefirstuc. If it doesn't get expanded until after it has been passed to \MFUsentencecase, then it will be treated as an exclusion instead.

1.6 Package Options

expanded

Redefines \glsmakefirstuc to simply use \MFUsentencecase.

unexpanded

Redefines \glsmakefirstuc to use \MFUsentencecase without expanding its argument.

grabfirst

avoid where possible

Redefines \glsmakefirstuc to use \mfugrabfirstuc and also redefines \mfirstucMakeUppercase to use \MakeUppercase.

This option is best avoided and is only provided to emulate the v2.07 behaviour of \glsmakefirstuc. However, you may still have different results compared to v2.07, so you may need to use rollback instead.

2 Title-Case

The title-case commands are designed to convert the first letter of each word in a phrase to uppercase. These commands are robust.

> `\capitalisewords{⟨text⟩}`

This command applies a sentence-case command to each word in ⟨text⟩ where the space character is used as the word separator. Note that it has to be an ordinary space character, not another form of space, such as ~ or `\space`. Note that no expansion is performed on ⟨text⟩. For example:

`\capitalisewords` `{a sample phrase}`	A Sample Phrase

See §2.1 for excluding words (such as "of") from the case-changing.

> This isn't the same as the LaTeX3 command `\text_titlecase:n`, which converts the first letter to uppercase and all other letters to lowercase.

For convenience, there are shortcut commands if expansion is required before parsing the argument:

> `\xcapitalisewords{⟨text⟩}`

This is a shortcut for:

> `\expandafter\capitalisewords\expandafter{⟨text⟩}`

> `\ecapitalisewords{⟨text⟩}`

This fully expands ⟨text⟩ before passing it to `\capitalisewords`.

The parser used by `\capitalisewords` first splits up the text on each space character. Each of these space-separated words may actually be a compound, so further parsing is performed on each "word". The divisions within the compound word should be marked up with:

> `\MFUwordbreak{⟨text⟩}`

For example:

`\capitalisewords{a` `big\MFUwordbreak{/}small idea}`	A Big/Small Idea

Each sub-word within the compound word is encapsulated with:

> `\MFUcapword{⟨text⟩}`

Since it's inconvenient to have to markup every hyphen, `\MFUcapword` can be enabled to check for hyphens.

> `\ifMFUhyphen ⟨true⟩\else ⟨false⟩\fi`   *initial:* `\iffalse`

This conditional determines whether or not `\MFUcapword` should consider a hyphen a word break. If this conditional is true, then `\MFUcapword` will encapsulate its argument with:

> `\MFUhyphencapword{⟨text⟩}`

This will parse ⟨text⟩ for hyphen characters and apply the case change to each hyphen-separated word. Otherwise `\MFUcapword` will treat its argument as a single word.

The conditional can be set to true with:

> `\MFUhyphentrue`

and switched back off with:

> `\MFUhyphenfalse`

For example:

```
\capitalisewords{server-side includes} /
\MFUhyphentrue
\capitalisewords{server-side includes}
```

Server-side Includes / Server-Side Includes

The actual case-change of each word is performed with:

`\MFUcapwordfirstuc{`⟨*word*⟩`}`

This defaults to `\makefirstuc{`⟨*word*⟩`}` but may be redefined to use `\MFUsentencecase`, if preferred.

Hyphens and `\MFUwordbreak` must be visible to the parser that searches for word breaks. This means they won't be detected if they are within a group or in the definition of a command.

Formatting for the entire phrase must go outside `\capitalisewords` (unlike `\makefirstuc`). For example:

```
\capitalisewords{\emph{a sample phrase}} /
\emph{\capitalisewords{a sample phrase}}
```

A sample phrase / *A Sample Phrase*

If your phrase is likely to contain formatting commands, you can instead use:

`\capitalisefmtwords{`⟨*text*⟩`}`   *modifier:* `*`

where ⟨*text*⟩ may be just words (as with `\capitalisewords`):

`\capitalisefmtwords{`⟨*words*⟩`}`

or may be entirely enclosed in a formatting command in the form:

`\capitalisefmtwords{`⟨*cs*⟩`{`⟨*words*⟩`}}`

or contain formatted sub-phrases:

> `\capitalisefmtwords{`⟨*words*⟩ ⟨*cs*⟩`{`⟨*sub-phrase*⟩`}` ⟨*words*⟩`}`

The starred form only permits a text-block command at the start of the phrase. See §2.3 for examples.

> ⚠
>
> Avoid declarations, such as `\bfseries` or `\em`.

If expansion is required, you can use:

> 📌
>
> `\xcapitalisefmtwords{`⟨*text*⟩`}`   *modifier:* *

which is a shortcut for:

> `\expandafter\capitalisefmtwords\expandafter{`⟨*text*⟩`}`

(The star modifier will be applied with `\xcapitalisefmtwords*` in an appropriate manner.)

> 📌
>
> `\ecapitalisefmtwords{`⟨*text*⟩`}`   *modifier:* *

This will fully expand its argument before passing it to `\capitalisefmtwords`. Again, the star modifier may be used.

> ℹ
>
> The unstarred `\capitalisefmtwords` is only designed for phrases that contain text-block commands with a single argument, which should be a word or sub-phrase. Anything more complicated is likely to break. Instead, use the starred form or `\capitalisewords`.

2.1 Excluding Words From Case-Changing

Some words typically shouldn't have their case changed unless they occur at the start. These words can be identified with:

> 📌
>
> `\MFUnocap{`⟨*word*⟩`}`

This only has a local effect. The global version is:

```
\gMFUnocap{⟨word⟩}
```

The list of words that shouldn't be capitalised can be cleared using:

```
\MFUclear
```

For example:

```
\capitalisewords{the wind in the willows}

\MFUnocap{in}%
\MFUnocap{the}%

\capitalisewords{the wind in the willows}
```

The Wind In The Willows
The Wind in the Willows

Since the case-change is ultimately performed by \makefirstuc, you can also use an exclusion to prevent an individual word from being changed. For example:

```
\newcommand{\NoChange}[1]{#1}
\MFUexcl{\NoChange}
\MFUclear
\capitalisewords{the \NoChange{wind} in the willows}
```

The wind In The Willows

This can also work if you redefine \MFUcapwordfirstuc to use \MFUsentencecase provided the exclusion command doesn't expand (so \NoChange would need to be protected or made robust in the above example).

Exceptions only apply to (non-leading) whole words or words separated with \MFUwordbreak but not to parts of a hyphenated word that are split by \MFUhyphencapword.

Examples:

1. Exceptions aren't applied if \MFUwordbreak occurs before the first space.

```
\MFUnocap{a}\MFUnocap{the}%
\capitalisewords{a\MFUwordbreak{/}the something}
```

A/The Something

2. Exceptions are applied for non-leading words:

```
\MFUnocap{and}\MFUnocap{or}%
\capitalisewords{one and\MFUwordbreak{/}or another}
```

One and/or Another

3. Exceptions aren't applied for hyphenated parts:

```
\MFUhyphentrue
\MFUnocap{and}\MFUnocap{or}%
\capitalisewords{one and-or another}
```

One And-Or Another

```
\usepackage{mfirstuc-english}
```

The supplementary package mfirstuc–english loads mfirstuc and uses \MFUnocap to add common English articles and conjunctions, such as "a", "an", "and", "but". You may want to add other words to this list, such as prepositions but, as there's some dispute over whether prepositions should be capitalised, I don't intend to add them to this package. Note that you need to explicitly load mfirstuc–english if you require it. There's no automatic language detection performed by mfirstuc.

If you want to write a similar package for another language, all you need to do is create a file with the extension `.sty` that starts with:

```
\NeedsTeXFormat{LaTeX2e}
```

The next line should identify the package. For example, if you have called the file `mfirstuc-french.sty` then you need:

```
\ProvidesPackage{mfirstuc-french}
```

It's a good idea to also add a version in the final optional argument, for example:

```
\ProvidesPackage{mfirstuc-french}[2014/07/30 v1.0]
```

Next load mfirstuc:

```
\RequirePackage{mfirstuc}
```

Now add all your `\MFUnocap` commands. For example:

```
\MFUnocap{de}
```

At the end of the file add:

```
\endinput
```

Put the file somewhere on TeX's path, and now you can use this package in your document. You might also consider uploading it to CTAN in case other users find it useful.

2.2 PDF Bookmarks

If you are using hyperref and want to use `\capitalisewords` or `\capitalisefmtwords` (or the expanded variants) in a section heading, the PDF bookmarks won't be able to use the command as it's not expandable, so you will get a warning that looks like:

```
Package hyperref Warning: Token not allowed in a PDF string
(PDFDocEncoding):
(hyperref) removing `\capitalisewords'
```

If you want to provide an alternative for the PDF bookmark, you can use hyperref's `\texorpdfstring` command. For example:

```
\chapter{\texorpdfstring
  {\capitalisewords{a book of rhyme}}%  TeX
  {A Book of Rhyme}%  PDF
}
```

Alternatively, you can use hyperref's mechanism for disabling commands within the bookmarks. For example:

```
\pdfstringdefDisableCommands{%
 \let\capitalisewords\@firstofone
}
```

The same applies to `\makefirstuc`. You can, however, use the expandable `\MFUsentencecase`. So you may prefer:

```
\pdfstringdefDisableCommands{%
 \let\capitalisewords\MFUsentencecase
 \let\makefirstuc\MFUsentencecase
}
```

See the hyperref manual for further details.

2.3 Examples

1. Text only:

```
\capitalisewords
{a little book of rhyme}
```
A Little Book Of Rhyme

2. Excluding words (see §2.1):

```
\MFUnocap{of}
\capitalisewords
{a little book of rhyme}
```
A Little Book of Rhyme

3. `\space` isn't recognised as a word boundary:

```
\capitalisewords{a book of rhyme.}

\capitalisewords{a book\space of rhyme.}
```

A Book Of Rhyme.
A Book of Rhyme.

4. Phrase entirely enclosed in a formatting command:

```
\capitalisefmtwords{\emph{a small book of rhyme}}
```

A Small Book Of Rhyme

5. Sub-phrase enclosed in a formatting command:

```
\capitalisefmtwords{a \emph{small book} of rhyme}
```

A *Small Book* Of Rhyme

6. Nested text-block commands:

```
\capitalisefmtwords{\textbf{a \emph{small book}} of rhyme}
```

A *Small Book* Of Rhyme

7. Formatting and case-change exception (see §2.1):

```
\MFUnocap{of}
\capitalisefmtwords{\textbf{a \emph{small book}} of rhyme}
```

A *Small Book* of Rhyme

8. Starred form:

```
\MFUnocap{of}
\capitalisefmtwords*{\emph{a small book of rhyme}}
```

A Small Book of Rhyme

9. The starred form also works with just text (no text-block command):

```
\MFUnocap{of}
\capitalisefmtwords*{a small book of rhyme}
```

A Small Book of Rhyme

10. Expansion:

```
\newcommand{\abc}{\xyz\space four five}
\newcommand{\xyz}{one two three}
No expansion: \capitalisewords{\abc}.

First object one-level expansion: \xcapitalisewords{\abc}.

Fully expanded: \ecapitalisewords{\abc}.
```

> No expansion: one two three four five.
> First object one-level expansion: one two three Four Five.
> Fully expanded: One Two Three Four Five.

Remember that the spaces need to be explicit. In the second case above, using \x-capitalisewords, the space before "four" has been hidden within \space so it's not recognised as a word boundary, but in the third case, \space has been expanded to an actual space character.

If there is a text-block command within the argument of the starred form, it's assumed to be at the start of the argument. Unexpected results can occur if there are other commands. For example:

```
\MFUnocap{of}
\capitalisefmtwords*{\emph{a small} book \textbf{of rhyme}}
```

A Small Book **Of rhyme**

In this case \textbf{of rhyme} is considered a single word. Similarly if the text-block command occurs in the middle of the argument:

```
\MFUnocap{of}
\capitalisefmtwords*{a \emph{very small} book of rhyme}
```

A *Very small* Book of Rhyme

In this case \emph{very small} is considered a single word.

Grouping causes interference. As with all the commands described here, avoid declarations.

```
\capitalisefmtwords{{\bfseries a \emph{small book}} of rhyme}
```

> As a general rule, it's better to define semantic commands rather than directly using font commands and declarations within the document.

Avoid complicated commands in the unstarred version. For example, the following breaks:

```
\newcommand*{\swap}[2]{{#2}{#1}}
\capitalisefmtwords{a \swap{bo}{ok} of rhyme}
```

However it can work with the starred form and the simpler \capitalisewords:

```
\newcommand*{\swap}[2]{{#2}{#1}}
\capitalisefmtwords*{a \swap{bo}{ok} of rhyme}

\capitalisewords{a \swap{bo}{ok} of rhyme}
```

A okBo Of Rhyme
A okBo Of Rhyme

Note that the case change is applied to the first argument.

3 Miscellaneous

3.1 Saving Exclusions, Blockers and Mappings in the `aux` File

If the exclusions, mappings and blockers are required by some external tool, the information can be saved in the aux file.

> 📌
>
> `\MFUsaveatend`

This saves the information at the end of the document using a delayed write. This means that it can register all information identified throughout the document, but there's a chance the document may end before the write takes place (for example, if the last page only contains floats). This command may be counteracted by:

> 📌
>
> `\MFUsave`

This saves the information using a protected write at the point where this command occurs, which may be too soon if additional exclusions, mappings or blockers are identified later. This command will counteract any instance of `\MFUsaveatend`, regardless of whether or not `\MFUsaveatend` comes before or after `\MFUsave`.

The associated `aux` commands are listed below. In each case, a definition is provided in the aux file that does nothing.

> 📌
>
> `\@mfu@excls{`⟨*exclusions*⟩`}`

Lists all exclusions. For example:

```
\@mfu@excls{\begin \cite \end \label
\ref \cite \NoCaseChange \ensuremath
\MFUskippunc \gls \glspl}
```

```
\@mfu@blockers{⟨blockers⟩}
```

Lists all blockers. For example:

```
\@mfu@blockers{\Gls \Glspl }
```

```
\@mfu@mappings{⟨mappings⟩}
```

Lists all mappings as a ⟨*key*⟩=⟨*value*⟩ list. For example:

```
\@mfu@mappings{ {\gls }= {\Gls }, {\glspl }= {\Glspl }}
```

3.2 All-Caps

```
\mfirstucMakeUppercase{⟨text⟩}
```

This command was originally used to perform the actual conversion to uppercase and was defined to use command `\MakeUppercase`. The glossaries package (before v4.50) formerly loaded the textcase package and redefined `\mfirstucMakeUppercase` to use `\MakeText-Uppercase`, which was better than `\MakeUppercase`.

The textcase package has been deprecated as from 2022 and it now simply sets `\Make-TextUppercase` to `\MakeUppercase` because the new kernel now defines `\MakeUppercase` to use the newer LaTeX3 command `\text_uppercase:n`. Although that command is expandable, `\MakeUppercase` is robust.

As from v2.08, mfirstuc now defines `\mfirstucMakeUppercase` so that it uses `\text_titlecase_first:n` directly, rather than indirectly through `\MakeUppercase`, which means that it's now expandable. However, `\mfirstucMakeUppercase` is no longer used by mfirstuc except in `\mfugrabfirstuc`, which is provided to emulate v2.07. Note that the grabfirst option will also redefine `\mfirstucMakeUppercase` to use `\MakeUppercase`.

3.3 UTF-8

> ℹ
>
> See Binary Files, Text Files and File Encodings*ᵃ* if you are confused about how file
> encodings, such as UTF-8, relate to text files.
>
> ---
> *ᵃ*`dickimaw-books.com/blog/binary-files-text-files-and-file-encodings/`

This section only applies to rollback or the use of `\mfugrabfirstuc` (implemented via the
`grabfirst` option).

Prior to version 2.08, the case-change applied by the `\glsmakefirstuc` command worked
by utilizing the fact that, in most cases, TeX doesn't require a regular argument to be enclosed
in braces if it only consists of a single token. (This is why you can do, say, `\frac12` instead
of `\frac{1}{2}` or `x^2` instead of `x^{2}`, although this practice is discouraged by some.)

A simplistic version of the original `\glsmakefirstuc` command is:

```
\newcommand*{\FirstUC}[1]{\MakeUppercase #1}
```

Here

```
\FirstUC{abc}
```

is equivalent to

```
\MakeUppercase abc
```

and since `\MakeUppercase` requires an argument, it grabs the first token (the character "a"
in this case) and uses that as the argument so that the result is "Abc". This behaviour can be
achieved with `\mfugrabfirstuc`.

Unfortunately, this will fail if the content starts with a UTF-8 character and you are using
pdfLaTeX, where each octet of the UTF-8 character is a separate token. This isn't a problem
with XꟾLaTeX and LuaLaTeX which both treat the entire multibyte character as a single token.

LaTeX3 now provides `\text_titlecase_first:n` to convert the first character to upper-
case, so now the case-change can be applied with:

```
\ExplSyntaxOn
\text_titlecase_first:n{élite}
\ExplSyntaxOff
```

This is exactly what `\MFUsentencecase` does without the need to switch on LaTeX3 syntax.

The difference with the default definition of `\glsmakefirstuc` is that it prevents its argument from being expanded.

Symbols

Symbol	Description
📌	The syntax and usage of a command, environment or option etc.
i	An important message.
❶	Prominent information.
📄	LaTeX code to insert into your document.
📄	Problematic code which should be avoided.
📄	How the example code should appear in the PDF.
🔍	Text in a transcript or log file or written to STDOUT or STDERR.
📄 Un	Code that requires a native Unicode engine (XƎLaTeX or LuaLaTeX).
☰	An option that doesn't take a value.
⚠	A warning.

Glossary

Blocker

A command that prohibits case-changing. If encountered by `\makefirstuc`, it won't attempt to apply any case-changing. See §1.4.

Exclusion command

A command whose argument should be skipped by the case-changer. The content that follows the command should have its case-changed instead. See §1.3.

Exclusion word

A word that shouldn't have its case changed by title-case commands, unless the word occurs at the start.

Mapping

A command that should be substituted by another, if encountered by `\makefirstuc`, instead of applying a case-change. See §1.5.

Semantic command

Essentially, this is a command associated with a particular element, idea or topic that hides the font and other stylistic formatting inside its definition. For example, Latin taxonomy is usually displayed in italic. Explicitly using font commands, for example

```
\textit{Clostridium}
```

is syntactic markup. Whereas defining a command called, say, `\bacteria` that displays its argument in italics is a semantic command. The actual styling is hidden and the command relates specifically to a particular concept.

```
Syntactic: \textit{Clostridium}

\newrobustcmd*{\bacteria}[1]\emph{#1}%
Semantic: \bacteria{Clostridium}
```

The end result is the same:

```
Syntactic: Clostridium
Semantic: Clostridium
```

The advantage with semantic commands is that it's much easier to change the style, simply by adjusting the command definition. Note that I've used \newrobustcmd to make the semantic command robust as the style commands can cause a problem if they expand too soon.

Sentence-case

Content that should appear at the start of a sentence that needs to have its first significant character converted to uppercase. See §1.

Title-case

Content that needs to appear in a title that should have each significant word converted to sentence-case. See §2.

Unicode Transformation Format (8-bit) (UTF-8)

A variable-width encoding that uses 8-bit code units. This means that some characters are represented by more that one byte. X͏ǝLͣATͤEX and LuaLͣATͤEX treat the multi-byte sequence as a single token, but the older LͣATͤEX formats have single-byte tokens, which can cause complications, although these have mostly been addressed with the newer kernels introduced over the past few years. Related blog article: Binary Files, Text Files and File Encodings.[1]

---

[1]`dickimaw-books.com/blog/binary-files-text-files-and-file-encodings/`

Command Summary

@

> `\@mfu@blockers{⟨blockers⟩}`   mfirstuc v2.08+

An aux file command that lists all the identified blockers.

> `\@mfu@excls{⟨exclusions⟩}`   mfirstuc v2.08+

An aux file command that lists all the identified exclusions.

> `\@mfu@mappings{⟨mappings⟩}`   mfirstuc v2.08+

An aux file command that lists all the identified mappings as a ⟨key⟩=⟨value⟩ list.

C

> `\capitalisefmtwords{⟨text⟩}`   *modifier:* *   mfirstuc v2.03+

Similar to \capitalisewords but for content that may contain formatting commands.

> `\capitalisewords{⟨text⟩}`   mfirstuc v1.06+

Applies title-case to ⟨text⟩. This will apply \MFUcapword to each word in ⟨text⟩ (unless the word has been identified as an exclusion word and doesn't occur at the start of ⟨text⟩).

E

> **\ecapitalisefmtwords**{⟨*text*⟩}　　　　*modifier:* ∗　mfirstuc v2.03+

Fully expands ⟨*text*⟩ and passes it to \capitalisefmtwords including the ∗ modifier, if provided.

> **\ecapitalisewords**{⟨*text*⟩}　　　　mfirstuc v1.10+

Fully expands ⟨*text*⟩ before passing it to \capitalisewords.

> **\emakefirstuc**{⟨*text*⟩}　　　　mfirstuc v1.10+

Fully expands ⟨*text*⟩ before passing it to \makefirstuc.

G

> **\glsmakefirstuc**{⟨*text*⟩}　　　　mfirstuc v1.05+

Used by \makefirstuc to perform the case-change. This uses \MFUsentencecase{⟨*text*⟩} but by default will prevent its argument from being expanded.

> **\gMFUnocap**{⟨*word*⟩}　　　　mfirstuc v1.09+

Globally adds ⟨*word*⟩ to the exclusion word list for \capitalisewords.

I

> **\ifMFUhyphen** ⟨*true*⟩\else ⟨*false*⟩\fi　　　*initial:* \iffalse　mfirstuc v2.03+

Conditional that determines whether or not hyphens should be considered word boundaries by the title-case commands.

M

`\makefirstuc{`⟨*text*⟩`}`	mfirstuc v1.0+	§1.2; 4

Converts ⟨*text*⟩ to sentence-case internally using `\glsmakefirstuc` to perform the case-change, but first parses ⟨*text*⟩ to deal with special cases.

`\mfirstucMakeUppercase{`⟨*text*⟩`}`	mfirstuc v1.07+	§3.2; 27

Converts ⟨*text*⟩ to uppercase. Now only used in `\mfugrabfirstuc`.

`\MFUaddmap{`⟨*cs1*⟩`}{`⟨*cs2*⟩`}`	mfirstuc v2.08+	§1.5; 12

Identifies a mapping from the command ⟨*cs1*⟩ to command ⟨*cs2*⟩.

`\MFUblocker{`⟨*cs*⟩`}`	mfirstuc v2.08+	§1.4; 11

Locally identifies ⟨*cs*⟩ as a blocker command.

`\MFUcapword{`⟨*text*⟩`}`	mfirstuc v2.03+	§2; 15

Uses either `\MFUhyphencapword` or `\MFUcapwordfirstuc` depending on `\ifMFUhyphen`.

`\MFUcapwordfirstuc{`⟨*word*⟩`}`	mfirstuc v2.07+	§2; 16

Used by `\MFUcapword` and `\MFUhyphencapword` to apply the case-change to the given word. This just does `\makefirstuc{`⟨*text*⟩`}` by default.

`\MFUclear`	mfirstuc v1.09+	§2.1; 18

Locally clears the exclusion word list for `\capitalisewords`.

`\MFUexcl{⟨cs⟩}`   mfirstuc v2.08+   §1.3; 8

Locally identifies ⟨cs⟩ as an exclusion command.

`\mfugrabfirstuc{⟨text⟩}`   mfirstuc v2.08+   §1.2; 7
(avoid where possible)

Provide to emulate `\glsmakefirstuc` in v2.07 and below, but can still produce different
results to v2.07. This will attempt to grab only the first character of ⟨text⟩ and convert it to
uppercase, leaving the rest of ⟨text⟩ unchanged. This won't work if ⟨text⟩ starts with a UTF-8
character, unless you are using X$_{\overline{\text{E}}}$LATEX or LuaLATEX.

`\MFUhyphencapword{⟨text⟩}`   mfirstuc v2.07+   §2; 15

Used by `\MFUcapword` to apply `\MFUcapwordfirstuc` to each hyphen-separated word in
⟨text⟩, if hyphens should indicate a word-break.

`\MFUhyphenfalse`   mfirstuc v2.03+   §2; 15

Sets `\ifMFUhyphen` to false.

`\MFUhyphentrue`   mfirstuc v2.03+   §2; 15

Sets `\ifMFUhyphen` to true.

`\MFUnocap{⟨word⟩}`   mfirstuc v1.09+   §2.1; 17

Locally adds ⟨word⟩ to the exclusion word list for `\capitalisewords`.

`\MFUsave`   mfirstuc v2.08+   §3.1; 26

Saves the list of exclusions, blockers and mappings to the aux file (if required by some external tool). This command sets itself to `\relax` so it doesn't repeat the action if used multiple times, and counteracts any use of `\MFUsaveatend`.

> **`\MFUsaveatend`** mfirstuc v2.08+

§3.1; 26

Saves the list of exclusions, blockers and mappings to the aux file (if required by some external tool) at the end of the document. This command sets itself to `\relax` so it doesn't repeat the action if used multiple times, but it can be overridden by `\MFUsave`.

> **`\MFUsentencecase{`**⟨*text*⟩**`}`** mfirstuc v2.08+

§1.2; 4

Converts ⟨*text*⟩ to sentence-case with expansion.

> **`\MFUskippunc{`**⟨*content*⟩**`}`** mfirstuc v2.07+

§1.3; 8

An exclusion command.

> **`\MFUwordbreak{`**⟨*text*⟩**`}`** mfirstuc v2.07+

§2; 15

Used to markup a character or command that should be treated as a word break by the title-case commands.

X

> **`\xcapitalisefmtwords{`**⟨*text*⟩**`}`** *modifier:* `*` mfirstuc v2.03+

§2; 17

Shortcut for `\expandafter\capitalisefmtwords\expandafter{`⟨*text*⟩`}` including the `*` modifier, if provided.

> **`\xcapitalisewords{`**⟨*text*⟩**`}`** mfirstuc v1.06+

§2; 14

Shortcut for \expandafter\capitalisewords\expandafter{⟨*text*⟩}.

\xmakefirstuc{⟨*text*⟩}   mfirstuc v1.01+   §1.2; 5

Shortcut for \expandafter\makefirstuc\expandafter{⟨*text*⟩}.

Index

Index

M

\makefirstuc . **§1.2**; Table 1.1; a, 1, 2, 4–13, 16, 18, 21, 31, 34, *35*, 38

\MakeTextUppercase 27

\MakeUppercase 7, 13, 27, 28

mapping . . Table 1.1; 1–3, 5, 9, 10, 12, 13, 26, 27, *31*, 33, 35, 37

mfirstuc–english package . . . . **§2.1**; 19

mfirstuc package 4

\mfirstucMakeUppercase . . . **§3.2**; 7, 8, 13, 27, *35*

\MFUaddmap **§1.5**; 12, *35*, *see also* \MFUexcl & \MFUblocker

\MFUblocker . . . . **§1.4**; 11, *35*, *see also* \MFUexcl & \MFUaddmap

\MFUcapword . . . . **§2**; 15, 33, *35*, 36

\MFUcapwordfirstuc . **§2**; 16, 18, *35*, 36, *see also* \MFUcapword & \MFUhyphencapword

\MFUclear . . . . **§2.1**; 18, *35*, *see also* \gMFUnocap & \MFUnocap

\MFUexcl . . . . **§1.3**; 8, 11, *36*, *see also* \MFUblocker & \MFUaddmap

\mfugrabfirstuc . **§1.2**; 7, 13, 27, 28, 35, *36*

\MFUhyphencapword . . **§2**; 15, 18, 35, *36*, *see also* \ifMFUhyphen & \MFUcapword

\MFUhyphenfalse . . . . **§2**; 15, *36*

\MFUhyphentrue . . . . **§2**; 15, *36*

\MFUnocap . . . . **§2.1**; 17, 19, 20, *36*, *see also* \gMFUnocap & \MFUclear

\MFUsave . . . . **§3.1**; 26, *36*, 37

\MFUsaveatend . . . . **§3.1**; 26, *37*

\MFUsentencecase . **§1.2**; Table 1.1; 1, 2, 4, 6, 8–13, 16, 18, 21, 28, 34, *37*

\MFUskippunc . . . . **§1.3**; 8–10, 26, *37*

\MFUwordbreak . . . . . . **§2**; 15, 16, 18, 19, *37*

N

\nameref 2, 10, 11

\NoCaseChange 8, 9, 11, 12, 26

P

\pdfstringdefDisableCommands 21

\protect 5

punctuation 1, 9, 10

R

\ref 8, 26

S

semantic command 1, 4, 5, 24, *31*

sentence-case . . . . . a, 1, 6, 12, 14, *32*, 35, 37

\space 14, 21, 23, 24

T

\texorpdfstring 20

textcase package 27

\text_titlecase:n 14

\text_titlecase_first:n 4, 27, 28

\text_uppercase:n 27

title-case 14, 31, *32*, 33, 34, 37

U

\unexpanded 6

unexpanded . . . . **§1.6**; 2, 3, 6, 13

uppercase 14, 27, *32*, 35, 36

UTF-8 . . . . 1, 7, 28, *32*, 36

X

\xcapitalisefmtwords . . . . **§2**; 17, *37*

\xcapitalisewords . . . . **§2**; 14, 24, *37*

\xmakefirstuc . . . . **§1.2**; 5, 11, 13, *38*