Documenting GAP *code with* GAPDoc

Frank Lübeck
Lehrstuhl D für Mathematik, RWTH Aachen
Aachen, Germany

Braunschweig, September 2007

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

The basic idea of GAPDoc

- define a markup language for GAP documentation that specifies *meaning* (and not the layout).
- documents should allow high quality rendering in various formats (`pdflatex` with `hyperref`, text for display in a terminal, HTML, and potentially future formats)
- the markup should fit with GAP terminology
- use XML to define the markup language
- provide sample converters to mentioned output formats

[Example: Beginning of `3k+1.xml`]

In the rest of this talk I will highlight some changes from GAPDoc version 0.99999 to 1.0.

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for < and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark=". . .»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for $<$ and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark="...»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for `<` and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark="...»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for `<` and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark="...»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for `<` and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark="...»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Changes of GAPDoc language in 1.0

- Made entities for LaTeXspecial characters (`&tamp;`, `&percent;` and so on) unnecessary (but kept them for backward compatibility). Now just write characters in content and attribute values directly. (But use `&lt;` for `<` and `&amp;` for `&`.)
- A `<ManSection>` does now allow an optional `<Heading>`.
- `<Index>` does now allow an optional `<Subkey>` element to specify subkeys with further markup (not possible in `Subkey` attribute).
- `<URL>`, `<Email>`, `<Homepage>` now allow optional elements `<Link>` and `<LinkText>` to specify text with further markup (not possible in `Text` attribute).
- New element `<Ignore Remark="...»`. Can be used everywhere (e.g., for additional data like source code, or to hide non finished parts of the document).

Some changes of the converter programs

- The "XML parser" can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf's know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The "XML parser" can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf's know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

Some changes of the converter programs

- The “XML parser” can now deal with several encodings: UTF-8, all latin?, ASCII (internally, it works with unicode).
- An improved `ComposedDocument` can be used by the parser to give error messages with the original position of the input.
- Links in pdf- and HTML-documents no longer depend on section numbers, they remain valid as long as a section stays inside the same chapter.
- LaTeX- (pdf-)version: hyphenation of URLs and index entries, more options (no color, ..), pdf’s know their paper size.
- Text version: Color markup can be customized by the user (`SetGAPDocTextTheme`), text is now in UTF-8 and translated on the fly to terminal encoding by the help system.
- HTML-version: More markup for CSS configuration, new sample `gapdoc.css`, links to subsections on top of each page (chapter).
- Utilities `ManualExamples` and `TestManualExamples` to extract and test examples from the manuals (maybe we can discuss this later).

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, "XML", "URL" and other encodings.
- Some non-injective (partial) maps from unicode: a "LaTeX" encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

General text utilities

GAPDoc contains utilities for manipulating texts which may be of independent interest:
text attributes (ANSI escape sequences), `StripBeginEnd`, `FormatParagraph`, ...

Unicode strings

- Introduced unicode strings and characters as GAP objects, `Unicode` can get input in various encodings or integer lists.
- Translations between unicode strings and GAP strings in various character encodings, `Encode` can translate to UTF-8, ISO-8859-X, “XML”, “URL” and other encodings.
- Some non-injective (partial) maps from unicode: a “LaTeX” encoding, simplifications to ASCII or latin1, conversions to lowercase and uppercase characters.

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>`, `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`.
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example ...]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example . . .]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example ...]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LaTEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example . . .]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example . . .]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example . . .]

Utilities for bibliographies in GAPDoc

- Functions for parsing and writing BibTEX files, function for parsing and normalizing names (`author`, `editor` entries). But: BibTEX is designed for use with LATEX, not HTML or Text display (non-ASCII characters, URLs, formulae, macro expansion).
- Introduced a BibXMLext format: extension of an (existing) `bibxml.dtd` which is an XML version of the BibTEX definition, plus elements `<name>`, `<M>`, `<Math>`, `<URL>` `<C>`, `<Alt>`, `<string>`, `<value>`, `<string>`, `<other>`, `<Wrap>`
- functions to parse BibXMLext files, and translate entries to BibTEX, text, HTML (these are user adjustable and extendible).
- BibXMLext data can now be used with GAPDoc instead of BibTEX files (and this is suggested).

[an example . . .]