Cataloguing the Greek manuscripts of the Lambeth Palace Library: An exercise in transforming Excel into PDF via XML using (Plain) XeTeX

Philip Taylor

Honorary Research Associate
The Hellenic Institute
Royal Holloway, University of London
United Kingdom
Email: `p.taylor@rhul.ac.uk`

Work is in progress to prepare an analytical catalogue of the Greek Manuscripts held in the Lambeth Palace Library. The catalogue will be published online in downloadable PDF format and (at a later stage) in print, using a single set of source documents marked up in the extensible markup language XML. This paper discusses the various stages through which the documents pass, starting as Excel spreadsheets and ending up both in Adobe Portable Document Format (PDF) and as Text Encoding Initiative (TEI)-compliant XML.

1 Introduction

Lambeth Palace, the official London residence of the Archbishop of Canterbury, the spiritual head of the Church of England and of the worldwide Anglican Communion, is also home to the Lambeth Palace Library (LPL), which contains a vast collection of material relating to ecclesiastical history dating as far back as the 9th century AD. The library contains over 120,000 books, as well as the archives of the Archbishops of Canterbury and other church bodies dating back to the 12th century. One small but nonetheless important part of this collection consists of 53 Greek manuscripts acquired by LPL following its founding as a public library in 1610, including those donated in 2007 by Sion College. Dated between the 10th and 19th centuries AD, the manuscripts contain various texts ranging from Gospel books, lectionaries, commentaries on the Bible and theological treatises to historical and literary works including those by classical and post-Byzantine authors. The analytical catalogue of these manuscripts is intended to be an invaluable resource for students and scholars internationally.

The largest single component of the collection is the set of manuscripts acquired by J.D. Carlyle in the eastern Mediterranean in 1800–01 for the establishment of a new critical edition of the New Testament and bought for the Library after his death. It transpired that some of the manuscripts had in fact only been borrowed by Carlyle from religious institutions in the East, and when the Orthodox Patriarch of Jerusalem requested their return, 11 were handed over in 1817 and ordered to be placed in the Patriarchal Library. Three of these are identifiable in the catalogue of that library, but the other eight still remain to be traced.

Following standard methodology, the catalogue will comprise an introduction on the collection, followed by an analytical description of the contents of each manuscript with reference to editions of texts (if published), accompanied by detailed palæographical and codicological examination including the following items: material; date; dimensions and number of folia; number of lines per folio; contents; physical state of the manuscript; collation of gatherings; ruling type(s) on parchment or watermarks on paper; miniatures and other ornamentation; type(s) and style(s) of script; identified scribe(s); colophon(s); notes of ownership, sale and purchase, other memoranda, etc; binding; bibliography. The catalogue will be supported by indices and by images of selected folia and bindings of the manuscripts. A research trip to the Library of the Orthodox Patriarchate in Jerusalem in 2014 will clarify the links between the two collections as detailed above.

The finished catalogue will be published in downloadable PDF format on the web sites of both LPL and of the Hellenic Institute, thus further enhancing the accessibility of, and interest in, this collection among scholars and the public alike. Furthermore, the raw data, encoded as TEI-compliant[1] XML, will also be made available to enable future generations of scholars to carry out their own data-mining exercises. Finally, the catalogue will form the basis of an on-line exhibition (independent from the project), designed to make Byzantine studies more accessible to the general public.

2 Methodology

The scholarly aspects of the project had already started before detailed consideration could be given to the question of how the data collected might best be converted into formats suitable for online browsing, paper publishing, and data mining, although the advantages of converting it to Adobe PDF for the two former purposes were already clear and accepted at this stage. Once the team had been expanded to include some IT (information technology) expertise, the additional benefits of using XML both as the archival format and as a common source format from which all other formats could be derived was readily accepted by the members of the Project Board, but by then a substantial volume of data had already been accumulated in Microsoft Excel format and the first task was to ascertain how easy (or how difficult) it might be to convert the Excel data into XML.

[1] `http://tei-c.org/`

Examination of the Excel files revealed that each column was effectively a single, coarse-grained, XML element (the first few columns are headed MS number, Former MS numbers, Summary content, Date, and Material, respectively) and it was but the work of a few minutes to write a simple Excel formula that would wrap (e.g.,) `<MS_number></MS_number>` around the contents of column 1, `<Former_MS_numbers> </Former_MS_numbers>` around the contents of column 2 and so on. The fact that `<MS_number>` is not necessarily a TEI-compliant element name was not considered important—one of the team had attended a course given by the Centre for Humanities Computing at the University of Oxford, and was reasonably confident that the extensible stylesheet transformation language XSLT could be used to express a mapping from the *de facto* names chosen by the research scholar into the corresponding TEI-compliant names.

Further liaison with Sebastian Rahtz at the University of Oxford resulted in a recommendation that two members of the team (one scholar, one IT consultant) should acquire a copy of the <oXygen/> XML editor from SyncRO Soft SRL; this program would not only provide a convenient means of creating raw XML if needed but would also act as an intelligent framework within which the necessary XSLT code could be developed, tested and deployed. Two copies were obtained, the idea being that the scholarly side of the team could investigate the possibilities of direct data entry in XML while the IT side investigated the XSLT transformation and other technical considerations.

As it turned out, neither copy was initially put to the use intended; the scholarly side of the team, with important deadlines to meet, could not afford to get distracted from the key tasks of manuscript analysis and data entry (even if the latter was still in Excel format) while the IT side decided instead to focus on the possibility of importing the raw data in Excel format directly into the <oXygen/> editor.

Initial experiments with Excel import were not promising, but as soon as it was discovered that simply *opening* an Excel file in <oXygen/> is not the correct way to proceed, and that the File/Import/MS Excel file route was a necessary prerequisite to successful Excel import, success immediately followed, and within a short period of time, a simple and straightforward means of transforming the Excel data into XML had been established.

With the raw data now available as XML, work started on the next phase of the project, which was to convert the XML into Adobe PDF, both so as to facilitate the production of the primary deliverable of the project, an online/downloadable PDF, but also to facilitate the production of a printed catalogue at a later stage; clearly the two would almost certainly have a great deal in common, but there would equally clearly be differences. For example, the online/downloadable version of the PDF could easily be enhanced to contain hyperlinks, so that (for example) a relatively small image of a manuscript could act as a hyperlink to a larger, or perhaps even panable/zoomable, version of the same, whilst the printed form of the catalogue might present the same large-scale images as end-plates. How, then, to convert XML into PDF? The chosen solution was, of course, TeX, or to be more precise, XeTeX, the latter offering two key advantages over Knuthian TeX:

- XƎTEX allows direct Unicode input, in which format the Excel data had been entered
- XƎTEX allows direct access to all the fonts on one’s system.

So, given raw data in XML, how easy (or difficult) would it prove to convert this to a XƎTEX-compatible format? The answer is (perhaps surprisingly), not difficult at all—in fact, no conversion was necessary! How could that be? Surely TEX requires backslashes and braces, whilst XML is predicated on the use of angle-brackets; how can the former handle the latter, without some preliminary conversion to re-express the XML in a TEX-compatible syntax? In fact, as stated above, no conversion was necessary; all that was necessary was to write a short TEX preamble that (a) makes a left angle-bracket active, and (b) implements a simple parser such that `<element attribute="value" attribute="value"` … `>` … `</element>` is effectively mapped to `\element {attribute="value"` … `}`. Augment this with a second parser that, given a sequence of `attribute="value"` pairs and a target attribute, returns the value associated with that attribute, and the problem is 90% solved. In practice, mapping `<element>` to `\element` is not a good idea, because `\element` could only too easily turn out to be a TEX primitive or a pre-defined TEX macro, so what we really do is to map `<element>` to `\XML:element`, a control sequence that, within the domain of Plain XƎTEX at least, can be guaranteed not to pre-exist (this last is just one of many many reasons why the present author uses only Plain XƎTEX and never XƎLATEX—the latter creates a large number of internal control sequences that contain “funny” characters, and it is by no means impossible that `\XML:element` (for abitrary `<element>`) might conflict with a pre-defined internal LATEX name. It should perhaps be noted at this point that within the standard Plain TEX `\catcode` régime, a control word such as `\XML:element` cannot be directly entered in the source code; it is, however, trivial to create such control words using the TEX primitives `\csname` and `\endcsname`. Furthermore, ε-TEX adds the extremely useful primitive `\ifcsname`, which allows a TEX program to test whether a particular control sequence already exists. All three of these primitives are heavily exploited in the XML processor discussed here.

So, without further ado, let us have a look at some fragments of the code involved. We start by looking at the basic means by which `<element>` (and its matching `</element>`—we will overlook the potential complications of `<element/>`) can be 1st-class TEX citizens in their own right:

```
\catcode `\< = \active
\def <#1>{\partition #1 \sentinel}
\def \partition #1 #2\sentinel
    {%
        \ifcsname XML:#1\endcsname
        \else
            \expandafter \def \csname XML:#1\endcsname ##1%
            {\Message {Warning : \unexpanded {<#1>} is not defined}%
            }%
        \fi
        \csname XML:#1\endcsname {#2}%
    }
```

Let us look at this in a little more detail. We start by making the catcode of '<' active, which means that it can thereafter function either as a TEX primitive or as a TEX macro. At the next line, we define it as a macro with delimited parameter structure—everything up to (but not including) the first '>' after the initial '<' will form the parameter. The expansion of the macro involves a call to the `\partition` macro (defined on the following line), passing as parameters to `\partition` (a) the same parameter as was passed to '<' (i.e., everything after the '|<|' up to but not including the following '>'), (b) a space (this space is *very* important!), and (c) `\sentinel`; the last is *not* a parameter, but is rather a parameter delimiter, just as '>' is a delimiter in the parameter structure of '<'.

Now we need to consider the definition of `\partition`. It will be seen that it takes two parameters, the first being everything up to (but not including) the first space, whilst the second parameter is everything *after* that first space up to but not including the first occurrence of `\sentinel`. What is key here is that parameter 2 *may* be empty.

Now, before we even consider what `\partition` does with the parameters it is passed, we need to see how those parameters are derived from whatever occurs as an XML element (i.e., whatever features between '<' and '>'). Let us look at some examples of increasing complexity:

1. `<root>`

   Parameter 1 (hereinafter `#1`) of '<' is '`root`'. `\partition` is therefore called with a parameter string consisting of '`root \sentinel`'. `\partition` treats as parameter 1 everything up to the first space, so `#1` of `\partition` is '`root`'. What is `#2` of `\partition`? Well, it is everything that follows the first space, up to and not including `\sentinel`. So what is that? It is the empty string. If we were to examine `#2` within `\partition`, using a test such as `\ifx \relax #2\relax`, the test would be true. What happens when `#1` of '<' is more complex than just '`root`'? We will see that next.

2. `<foreign language="Latin">`

   Much more complex! Let's analyse the behaviour in the same way. `#1` of '<' is '`foreign language="Latin"`', so the parameter string passed to `\partition` is '`foreign language="Latin" \sentinel`'. `\partition` treats as parameter 1 everything up to the first space, so `#1` of `\partition` is '`foreign`'. But this time, parameter 2 is *not* empty: it consists of everything following the first space up to (but not including) `\sentinel`, so `#2` consists of '`language="Latin" `'. One more example, and then we can see what `\partition` does with these parameters.

3. `<foreign language="Hebrew" direction="RTL">`

   We don't need to get any more complex than this, not because more complex parameter structures do not occur (they do: `<image>` has a very complex parameter structure indeed) but simply because any structure with more than two parameters is treated identically to one with exactly two. We analyse the behaviour as before. `#1` of '<' is '`foreign language="Hebrew" direction="RTL"`', so the parameter string passed to `\partition` is '`foreign language="Hebrew" direction="RTL" \sentinel`'. `\partition` treats as parameter 1 everything up to the first space, so `#1` of `\partition` is '`foreign`'. Again, parameter 2 is *not* empty: it consists of everything following the first space up to (but not including) `\sentinel`, so `#2` consists of '`language="Hebrew" direction="RTL"`'. And if '<' had three, four, five or 99 keyword:value pairs, all would be passed to `\partition` as parameter 2.

So now we need to examine the behaviour of `\partition`, rather than simply analysing how its parameter structure is determined. We'll revert to the simplest example to start with:

1. `<root>`

   Parameter 1 of '<' is '`root`'. `\partition` is called with a parameter string consisting of '`root \sentinel`'. `\partition` treats as parameter 1 everything up to the first space, so `#1` of `\partition` is '`root`', and `#2` is empty. `\partition` starts by checking whether `\XML:#1` (i.e., `\XML:root`) is defined; if it was *not* defined before `\partition` was called, `\partition` defines it to soak up its parameter and issue an informative warning message but otherwise have no effect. Then, regardless of whether `\XML:root` was defined on entry or not, it is called, passing `#2` of `\partition` as a balanced-text parameter. But `#2` of `\partition` was empty, so what is actually passed as parameter to `\XML:root` is an empty balanced text (i.e., the effect, ignoring catcodes, is as if we had written `\XML:root {}`). And thus, if one were (for example) to examine `#1` of `\XML:root` using `\ifx \relax #1\relax`, the test would be true. If you have been following this closely, you should already know what will happen for our next example…

2. `<foreign language="Latin">`

   `#1` of '<' is '`foreign language="Latin"`', so the parameter string passed to `\partition` is '`foreign language="Latin" \sentinel`'. `\partition` treats as parameter 1 everything up to the first space, so `#1` of `\partition` is '`foreign`'. Parameter 2 is not empty: it consists of everything following the first space up to (but not including) `\sentinel`, so `#2` consists of '`language="Latin" `'. As before, `\partition` starts by checking whether `\XML:#1` (i.e., `\XML:foreign`) is defined; if it was not defined before `\partition` was called, `\partition` defines it to soak up its parameter and issue an informative warning message but otherwise have no effect. Then, regardless as to whether `\XML:foreign` was defined on entry or not, it is called, passing `#2` of `\partition` as a balanced text. But in this case, `#2` of `\partition` is *not* empty—it is '`language="Latin" `', so the balanced text `{language="Latin" }` is passed as parameter to `\XML:foreign` (i.e., the effect, ignoring catcodes, is as if we had written `\XML:foreign {language="Latin" }`).

3. `<foreign language="Hebrew" direction="RTL">`

   No further explanation is surely necessary. If `\XML:foreign` was not defined on entry, it is defined to soak up its parameter and issue an informative warning. Then `\XML:foreign` is called, passing the balanced text `{language="Hebrew" direction="RTL" }` as parameter (i.e., it is as if we had written `\XML:foreign {language="Hebrew" direction="RTL" }`).

Now all that remains is to see how `\XML:foreign` and friends retrieve the values for the individual attributes they are expecting.

```
\xmlinlineelement {foreign}
    {\getattribute {language}<\the \attributevalue>}
            {</\the \attributevalue>}
\def \xmlinlineelement #1#2#3%
    {%
        \xmlelement {#1}{\bgroup #2}{#3\egroup}%
    }
\def \xmlelement #1#2#3%
    {%
        \xmldeclaration {#1}{#2}%
        \expandafter \def
            \csname XML:/\splitatfirstspace #1 \sentinel \endcsname
                {#3}%
    }
\def \xmldeclaration #1#2%
    {%
        \expandafter \def \csname XML:#1\endcsname ##1%
            {\attributes = {##1}#2}%
    }
\def \splitatfirstspace #1 #2\sentinel {#1}
\def \getattribute #1%
    {%
        \attributevalue = {}%
        \expandafter \expandafter \expandafter
        \ifx \expandafter \relax \the \attributes \relax
            \message {Attribute list empty}%
        \else
            \expandafter \getattributes \the \attributes \sentinel #1
        \fi
    }
\def \getattributes #1 #2\sentinel #3
    {%
        \splitkeywordvalue #1\sentinel #3
        \ifx \relax #2\relax \else \getattributes#2\sentinel #3 \fi
    }
```

```
\def \splitkeywordvalue #1=#2\sentinel #3
    {%
        \ifnum \strcmp {#1}{#3} = 0
            \attributevalue = \expandafter {\dequote #2}%
        \fi
    }
\def \dequote "#1"{#1}
```

This is fairly complex code, with each macro dependent on the next, so we will work through it at a fairly leisurely pace. To begin, it can be seen that we define `\XML:foreign` by declaring `{foreign}` as an `\xmlinlineelement`. `\xmlinlineelement` takes three parameters: the name of the element, the code to be executed when `<element>` is encountered, and the code to be executed when `</element>` is encountered. `\xmlinlineelement` passes on its three parameters in 1:1 correspondence to `\xmlelement`, prefixing `#2` with `\bgroup` and suffixing `#3` with `\egroup`, thereby ensuring (for example) that any environmental changes such as `\language` required to handle a stretch of text in a particular language are restricted to that stretch of text. `\xmlelement` in turn passes on its first two parameters in 1:1 correspondence to `\xmldeclaration`, and we will return to what `\xmlelement` does with its third parameter later. `\xmldeclaration` requires no helper macros, and defines (using `\csname` ... `\endcsname`) `\XML:#1##1` as `{\attributes = {##1}#2}`. Understanding what `#1`, `##1` and `#2` represent is the key to understanding the whole of this part of the code.

Parameters `#1` and `#2` have been passed through from `\xmlinlinelement`: `#1` is 'foreign', and we will return to `#2` in due course. `##1` is a placeholder, a formal parameter: it denotes the first (and only) formal parameter of `\XML:#1`, which we have just seen is `\XML:foreign` in the present case. The `#` signs are doubled because we are defining a macro within a macro. The effect (ignoring catcodes) is as if we had written `\def \XML:foreign #1`, so what `\xmldeclaration` is doing in this particular case is equivalent (ignoring catcodes) to `\def \XML:foreign #1{\attributes = {#1}...}`, where the mysterious `...` represents what has been passed through to `\xmldeclaration` as `#2`. And we can see what `...` is by looking back at the invocation and declaration of `\xmlineeelement`, from which it is clear that `...` is actually `\bgroup \getattribute {language}<\the \attributevalue>`. So, the ultimate effect of the invocation of `\xmldeclaration` in the present context is (modulo catcodes):

```
\def \XML:foreign #1{\attributes = {#1}\bgroup
        \getattribute {language}<\the \attributevalue>}
```

We will try to put all of this into context. Let us suppose that the XML source contains a stretch of text surrounded by `<foreign language="Greek">` ... `</foreign>`. For example, `<foreign language="Greek">βίβλος ἰησοῦ τοῦ ὑιοῦ ναυῆ</foreign>`. We have already seen that `<foreign ...>` expands (in part) to `\XML:foreign`, and we saw just above that `\XML:foreign #1` expands to `\attributes = {#1}\bgroup \getattribute {language}<\the \attributevalue>`.

And we already know that what gets passed as `#1` to `\XML:foreign` will be '`language="Greek" `', in the present context. So the net effect, when TEX encounters `<foreign language="Greek">`, is that `\attributes` (a token-list register) is assigned the balanced text `{language="Greek" }`, `\bgroup \getattribute {language}` is invoked to retrieve the value '`Greek`', and finally `<\the \attribute>` (i.e., `<Greek>`) is invoked. It is therefore at these latter parts of the exercise that we must next look.

It must now be admitted that `\getattribute` lacks some of the elegance of what has gone before. It represents a quick-and-dirty solution, and will almost certainly be refined in the future. However, it does its job, and as the number of attributes is always small, never exceeding 10 (in fact, probably never exceeding five), the resources wasted in continuing the search after the desired attribute has been located are not excessive in the great scheme of things.

`\getattribute` is passed the name of the attribute sought as its sole parameter, so in the example above (for example), `\getattribute` would be called with parameter `{language}`. `\getattribute` commences by clearing any current value in `\attributevalue` (another token-list register) and then looks to see if the token-list register `\attributes` is empty (the code would break if it were not empty but instead started with `\relax`, but that can be guaranteed not to occur within the environment for which the code was developed). The test works by first forcing the expansion of `\the \attributes` and then testing whether the expansion is non-empty by the usual trick of surrounding the expansion by `\ifx \relax ...\relax` as before. If the token list register *is* empty, the code issues a warning (because, by definition, the sought attribute cannot be found), otherwise it invokes the adjunct macro `\getattributes`, passing as parameters (a) the expansion of `\the \attributes`, (b) the sentinel `\sentinel` (again, this must not and will not occur within the attributes), (c) the attribute sought (which must not and will not contain one or more spaces), and (d) a space, as final delimiter.

The adjunct macro `\getattributes`, on being passed this parameter string, parses the string as follows. Everything up to the first space is treated as `#1`, everything thereafter up to `\sentinel` as `#2`, and everything following `\sentinel` up to but not including a space as `#3`. Thus if, for example, the XML contained `<foreign language="Hebrew" direction="RTL">`, `#1` would be `language="Hebrew"`, `#2` would be `direction="RTL"`, and `#3` would be `language` (remember, `\getattribute` was called with sole parameter `language`, and the sole parameter of `\getattribute` ends up as the third parameter of `\getattributes`).

`\getattributes` then calls a further adjunct macro `\splitkeywordvalue` to partition `#1` about its `=` sign, and also passes (as `#3`) the sought attribute (in the present case, '`language`'). `\splitkeywordvalue` then invokes the useful but very un-TEXlike primitive `\strcmp` (I no longer remember when this particular feature was added to ε-TEX) to compare the text to the left of the `=` sign with the sought attribute. If they are a perfect match, `\strcmp` returns 0, otherwise it returns either −1 or +1 depending as the text to the left of the `=` sign is lexicographically less than or greater than the sought parameter. If they are identical, a further adjunct macro `\dequote` is called to remove the quotes around the text to the right of the `=` sign, and the resulting value is assigned as a balanced text to the token list register `\attributevalue`.

It is at this point that things get very ugly. If `\splitkeywordvalue` has already found the sought attribute and returned its value via `\attributevalue`, the code ought to terminate at this point. But it does not. Instead, it iterates through the remainder of the attributes, just in case there is a second instance of the sought parameter, in which case its value will override the value already stored. Ideally, it would not do this. It would either return multiple instances of the sought attribute through a simulated array, or it would stop on encountering the first instance thereof. In a future version, maybe; in the present one not, as the code is already in use!

Now there is just a few loose ends to tidy up before we can draw this saga to a close. What does `\xmlelement` do with its third parameter, for example, and why did `\XML:foreign` end up calling `<Greek>` when the XML contained `<foreign language="Greek">`? Let's deal with the latter first. Bear in mind that the code and the XML were evolving together. Early versions of the XML did not use `<foreign language="whatever">` because there was as yet no attribute handling in the code. So early versions of the XML used `<Greek>` … `</Greek>`, `<Latin>` … `</Latin>`, `<Hebrew>` … `</Hebrew>` and so on. And as the code for `<Greek>` (and `<Latin>`, and `<Hebrew>`) was already written, why not recycle it when attribute handling *was* added the code? So I did. And the third parameter to `\xmlelement`? Well, if you look back, you will see that for (e.g.,) `\xmlinlineelement {foreign}...`, the third parameter to `\xmlelement` will be `</\the \attributevalue>\egroup`, so when `<foreign language="Greek">` is encountered in the input stream, `\xmlelement` will be called with `</Greek>\egroup` as `#3`.

But what does `\xmlelement` *do* with its third parameter? As in much of this code, the answer is not entirely straightforward. Remember that the first line of `\xmlelement` was intended to define the behaviour of (e.g.,) `<foreign>`. Well, the subsequent lines are required to define the behaviour of the corresponding closing tag (i.e., `</foreign>`, in the present case). It does this by defining `\XML:/foreign` as `</\the \attributevalue>`, so when (for example) TeX encounters `<foreign language="Greek">`…|</foreign>|, the initial `<foreign language="Greek">` will set `\attributevalue` to `{Greek}`, and the subsequent `</foreign>` will access the same `\attributevalue` (because any intervening further language changes will be protected by `\bgroup` … `\egroup` pairs) and the effect will therefore be to invoke `\XML:/foreign` which will in turn expand to yield `</Greek>`.

And that just about brings us to the end of the present article. I hope that it has succeeded in convincing you that a TeX document does *not* have to contain backslashes and braces, as is commonly believed. It can (as in the present example) have the form of an XML document, but in reality the syntax that it uses is entirely up to you, the user. All you need to do in order to enable TeX to handle your preferred markup is to write a short TeX preamble that will cause *your* markup to have the desired effect. The final pages of this article illustrate how one manuscript (MS 1214, the Octateuch) may appear in the final catalogue. It started life as an Excel spreadsheet. A few bits of raw XML were added to some of the fields, containing (for example) `<foreign language="Greek">` and `<image source="f1r.pdf" float="right">`; it contains no TeX commands whatsoever. It is then `\input` following the TeX preamble, and immediately followed by `\end`, and the results are as you see. TEX documents do *not* need to contain backslashes and braces in order to be understood by TEX!

3 Epilogue

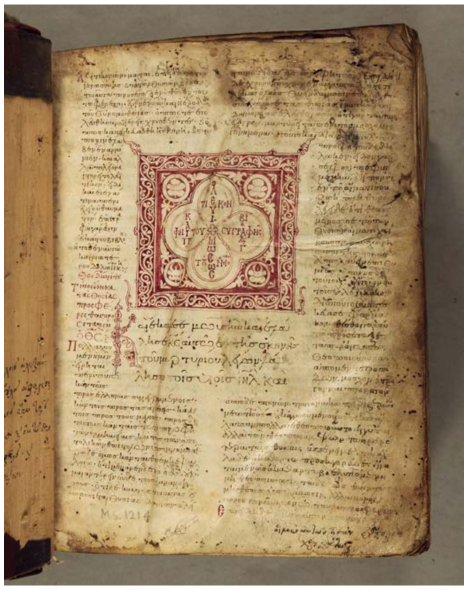
At the beginning of this article, it was stated that the Excel data would be transformed into an online/downloadable PDF, a printed book, and TEI-compliant XML. And those of you familiar with the TEI (the Text Encoding Initiative) will be saying to yourselves "well, it may be XML, but it is most certainly not TEI-compliant XML", and that is perfectly true. And the reason is, the first deliverables are the PDF and the book, for which "any old XML" will suffice, and for which "XML that can conveniently and reliably be parsed by TEX" will not only suffice but is in fact a *sine qua non*. So the dialect of XML that we use is specifically tailored towards (a) ease of parsing by TEX, and (b) being capable of expressing all that needs to be expressed when cataloguing early Greek manuscripts. But the great beauty of XML is that it is capable of transformation, and we confidently expect that converting "our" XML into TEI-compliant XML will be a relatively straightforward task using XSLT.

Postscript

A colour version of the catalogue entry for MS 1214 is placed at `http://hellenic-institute.rhul.ac.uk/Research/LPL/Greek-MSS/Catalogue/MS-1214.pdf` so that readers have the opportunity to appreciate some images of the MS in full colour and at regular paper size. Critical readers of the catalogue entry will note that the spacing is in places sub-optimal, with e.g., "f.", "ff." and so on being followed by end-of-sentence spacing or even by a line-turn. This is a direct result of the style of data entry used, in which "f.", "ff." and so on were not afforded any special treatment; this issue still remains to be addressed, but it is likely that the solution will be to require such abbreviations to be followed by a Unicode non-breaking space (`U+00A0`), which can then be handled appropriately by the XƎTEX code.

Acknowledgements

I would first of all like to most sincerely thank the Editors of *Εὔτυπον* for their kind invitation to compose this paper for publication in their journal. Secondly, I would like to thank Dr Giles Mandelbrote, Librarian and Archivist of Lambeth Palace Library and Chairman of the Project Board, for his gracious permission to describe the project. Thirdly, my sincere thanks to Dr Charalambos Dendrinos, Principal Investigator of the project, for reading a proof copy of the paper and for suggesting a number of improvements. And finally, thanks beyond number to Barbara Beeton who has once again read through one of my papers with the eyes of a typographic hawk. Needless to say, the responsibility for all remaining errors is mine and mine alone.



MS number
MS. 1214

Former MS numbers
None

Summary content
Octateuch (incomplete)

Date
12 November 1103

Material
Parchment

Folios
ff. 412

Dimensions
334 x 246

Gatherings
36 x 8 (288), 6 (294), 13 x 8 (406), 6 (412)

Folios/pages on which gatherings begin
ff. 1, 9, 17, 25, 33, 41, 49, 57, 65, 73, 81, 89, 97, 105, 113, 121, 137, 145, 153, 161, 169, 177, 185, 193, 201, 209, 217, 225, 233, 241, 249, 257, 265, 273, 282, 289, 295, 303, 311, 319, 327, 335, 343, 351, 359, 367, 375, 383, 391, 399, 407

Detailed content
Octateuch (incomplete), with *catena*: ff. 1r-90r: Leviticus (λευιτικὸν μωσέως τοῦ νομοθ<έ>τ<ου> καὶ προφήτου συγ<γ>ραφῆς βιβλί<ον> Γ′) (authors included in *catena*: anonymous; Apollinaris; Cyril of Alexandria; Cyril of Jerusalem; Eusebios; Isidore; Justin Martyr; Origen; Severianos; Symmachos; Theodoret); ff. 90v-196r: Numbers (ἀριθμοὶ τῶν ὑγιῶν ἱ(σρα)ηλ μωσέως· τοῦ θεόπτου συγγραφή βίβλο(ν) τετάρτ<ον>) (authors included in *catena*: anonymous; Apollinaris; Basil of Caesarea; Cyril of Alexandria; Eusebios; Gregory of Nyssa; Irenaeus; Philo Judaeus; Polychronios; Severianos; Severos; Theodoret); ff. 196v-294v: Deuteronomy (δευτερονόμιον μωσέως τοῦ προφήτ<ου> θεόπτου καὶ νομοθέτου, συγγραφῆς, βιβλίον πεμπτο(ν)) (authors included in *catena*: anonymous; Basil of Caesarea; Cyril of Alexandria; Diodoros; Dionysios of Alexandria; Eusebios; Gregory of Nyssa; Irenaeus; Isidore; Ioannis Chrysostom; Neilos of Ankyra; Origen; Philo Judaeus; Severianos; Severos; Theodoret; Victor); ff. 295r-347r: Joshua (βίβλος ἰησοῦ τοῦ ὑιοῦ ναυῆ)

(authors included in *catena*: anonymous; Cyril of Alexandria; Diodoros; Origen; Severos; Theodoret); ff. 347v-399r: Judges (βίβλος κριτῶν τοῦ ι(σρα)ήλ); (authors included in *catena*: anonymous; Cyril of Alexandria; Diodoros; Eusebios; Irenaeus; Josephus; Severos; Theodoret; Victor) ff. 399v-409v, line 12: Ruth (βίβλος τῆς ῥούθ) (authors included in *catena*: anonymous; Basil of Caesarea; Cyril of Alexandria; Ioannis Chrysostom; Theodoret); f. 409v, line 13-f. 411r, line 5: note on translations of the Old Testament from Hebrew into Greek (πόσαι ἐκδόσεις εἰσὶ τῆς θείας γραφῆς εἰτ'οῦν ἀπὸ τοῦ ἑβραϊκοῦ εἰς τὸ ἑλληνικὸν ἑρμηνείαι και τίνες οἱ ταύτην μηνεύσαντες); f. 411r, line 6-f. 411v, line 8: note on occasions when Israel was devastated by invaders (ποσάκις καὶ πότε ἐπορθήσησαν οἱ ἐξ ι(σρα)ήλ); f. 411v, lines 9-18: note on ambiguities in Scripture (πότε καὶ πόθεν γέγονεν ἡ ἐν ταῖς θεῖαις γραφαῖς εὑρισκομένη ἀσάφεια); f. 411v, line 19-f. 412r, line 16: note on the Hebrew names of God (ποίος καὶ πόσοις ὀνόμασι παρ'ἀβραίοις ὀνομάζεται ὁ θ(εὸ)ς); f. 412r, line 17-f. 412v, line 8: note on the unspoken name of God (Τὸ ἐπὶ τοῦ κ(υρίο)υ ταττόμενον ἀνεκφωνητον ὄνομα, διὰ τεσσάρων γράφεται στοιχείων); f. 412v, lines 9-21: colophon of Ioannis Koulix

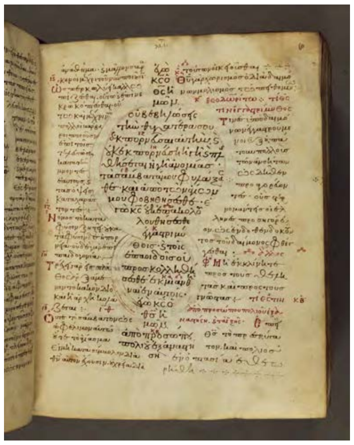
Folio/page numbers

By folios, top right, Arabic numerals, black ink

Quire numbers

At beginning of each quire, recto, and end of each quire, verso, bottom right, Greek numerals, mid-brown ink

Columns & lines



Bible text: single column, 3-30 lines; *catena*: surrounds main text, one or two columns, up to 41 lines; layout varies depending on amount of *catena* text to be accommodated: on most pages Biblical text forms a rectangular block of variable size, but occasionally, usually where *catena* text is greatly predominant, the Biblical text in the centre of the page is arranged in the shape of a cross, a circle, two linked circles one above the other, sometimes with a small rectangular base below and/or a small elliptical protrusion above, a circle above a rectangle above a circle, four circles in a cruciform pattern etc; these arrangements of text are usually surrounded by ornament or preparatory marking for it; occasionally (e.g. ff. 63r, 66r) a third brief *catena* passage appears in the margin beyond the two principal passages

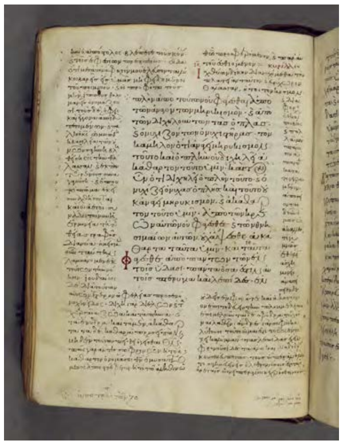
Ruling

System Leroy 9; layout Leroy K24D3 (strictly unclassified under Leroy system due to unusual three-column layout); hardpoint, piercing visible for horizontals and verticals; lines of text ignore ruling. The function of the vertical lines varies according to the relationship between Biblical text and *catena* in the page layout: where the *catena* appear in two columns, the central column divider separates these; the outer divisions delineate the outer edge of the Biblical text where this is of medium width, but are ignored where it is confined to a small area in the centre, while the outer verticals are used when it fills most of the page.

Scribe(s)

Ioannis Koulix (*Repertorium* I 166; II 222): main text, rubric, quire numbers; Hand B: marginal notes and corrections ff. 25v-27r, 34r, 36r, 39v, 42r, 51r, 69v, 70v, 86r, 178v, 210v, 234v, 235r, 265v, 266v; Hand C: notes ff. 89r, 126v, 200v; Hand D: notes ff. 200v, 412r; Hand E: note f. 10v; Hand F: notes ff. 221r, 222r, 233v; Hand G: note f. 407v; Hand H: notes f. 319r; Hand I: notes ff. 3r, 366v; Hand J: note f. 367r; Hand K: note, back board; Hand L: Hebrew note and lections ff. ff. 13r, 21v, 26v, 29v; Hand M: Hebrew lections ff. 36r, 46r, 56v, 64r, 77r, 82v, 91r, 103r, 119r, 132r, 141r, 150r, 159v-160r, 171r, 182r, 183r, 189r, 198r, 205r, 215v, 224v, 236r, 246r, 259r, 270r, 276v, 285v; Hand N: note, overwriting f. 1r; Hand O: note f. 201r; Hand P: note f. 409v; Hand Q: note f. 93v; Hand R: notes f. 412v; Hand S: note f. 412v; Hand T: chapter numbers, running headers; Ioannis: note f. 317r; Hand V: folio counts ff. 90r, 196r, 197r, 295r, 347v; Alexandros: ownership note, front board; Hand X: calculation f. 293v; Hand Y: calculation, back board; Hand Z: calculation, back board; Hand AA: calculation, back board; Hand AB: calculation, back board; Hand AC: calculation, back board

Script (general characteristics)



Ioannis Koulix, main text: Ornate calligraphic mixed minuscule, upright, with significant expansion of letters and many flamboyant ligatures, occasional flourishes on bottom line, with decoration (ff. 51r, 55v, 102v); superscription of letters common, but at line ends only; occasional deletions by strikethrough in red ink; Ioannis Koulix, rubric: Alexandrine majuscule

Script (letters & ligatures)

Ioannis Koulix: all minuscule letter forms and all majuscule letter forms except *mu*, *nu*, *upsilon* present in mixed minuscule; distinctive letter forms: minuscule *eta*, *iota*, *kappa* with doubled ascender; open *theta* with bulbous upper loop; *theta* with x-mark on horizontal; large, kinked minuscule *nu*; flat-bottomed majuscule *omega*; distinctive ligatures:

ligatures with large half-*epsilon*, especially *epsilon-xi*; ligatures with *omicron* surrounding and joining following letter; ligatures with left-hand horizontal of *pi* curving over top to join following letter; ligatures with large open *rho* surrounding following letter; *epsilon-psi* with large half-*epsilon* in main line of text curving up through horizontal of minuscule *psi* to join at top; *theta-epsilon-rho* with split *epsilon* formed of downward-curved stroke from *theta* and separate diagonal stroke joining open *rho*; *sigma-epsilon* with tail of *sigma* joining superscribed long-bottomed half-*epsilon* from below; *omega-rho* with *rho* passing through middle of majuscule *omega*

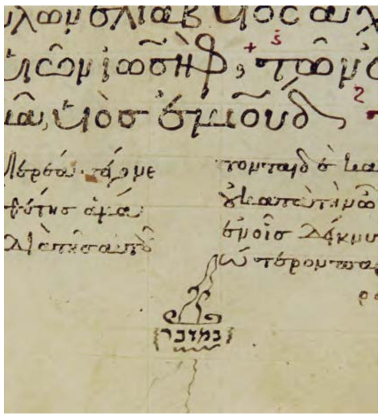
Diacritics & punctuation

Ioannis Koulix: breathings angular, half-angular and round; circumflexes often wide; mute *iota* usually absent, transitional when present; double dot almost always used functionally only, occasionally decorative on *iota* ; double grave accent present on δὲ, ἐπεὶ, κὰν, μὲν, μὴ; use of middle and upper point, lower and middle comma, question mark, full stop; chevron quotation marks; use of hyphen to merge words; horizontal strokes over proper names and numbers

Abbreviations

Ioannis Koulix: *Nomina sacra* (accents separate from strokes); καί; φησί; syllabic (throughout line); horizontal stroke for *nu*; *chi-rho* monogram for Χρυσόστομος

Apparatus

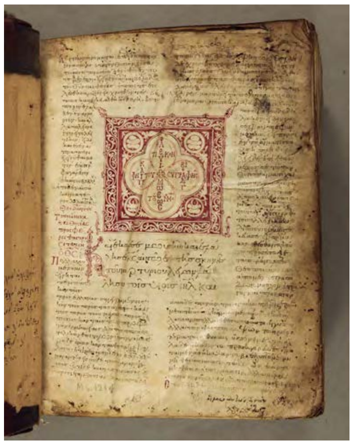


Rubricated author names and occasional headings for catena entries, marginal or in line of text; rubricated reference marks linking *catena* entries to the corresponding point in the Biblical text; marginal corrections, linked to corresponding point in the text by reference marks; ff. 89v, 196r, 197r, 295r, 347v: notes of number of folios occupied by each book; ff. 1v, 174r, 272r, 291v: marginal σημείωσαι marks; f. 91r-v: rubricated numbers of tribal list; ff. 13r, 21v, 29v, 36r, 46r, 56v, 64r, 77r, 82v, 91r, 103r, 119r, 132r, 141r, 150r, 159v-160r, 171r, 182r, 183r, 189r, 198r, 205r, 215v, 224v, 236r, 246r, 259r, 270r, 276v, 285v: Hebrew notes for beginning of Jewish lections, those on ff. 182r, 183r, 189r, 198r, 205r linked by manicules to the corresponding point in the Biblical text (...f. 46r: 'אחר', f. 56v: 'יקד-שנה', f. 64r: 'הכהן' etc.); ff. 1r-294r: running headers for books; marginal modern chapter numbers in Latin and Greek script, Arabic and Greek numerals, linked by reference marks to corresponding point in Biblical text

Ink

Ioannis Koulix: mid-brown and dark red; Hand B: variable brown; Hand C: dark brown; Hand D: dark brown; Hand E: dark brown; Hand F: mid-brown; Hand G: dark brown; Hand H: mid-brown; Hand I: dark brown; Hand J: dark brown; Hand K: black; Hand L: dark brown; Hand M: dark brown; Hand N: dark brown; Hand O: black; Hand P: dark brown; Hand Q: mid-brown; Hand R: black; Hand S: black; Hand T: mid-brown; Ioannis: bright red; Hand V: green; Alexandros: dark brown; Hand X: black; Hand Y: mid-brown; Hand Z: dark brown; Hand AA: black; Hand AB: blue; Hand AC: pencil

Ornament



ff. 1r, 196v, 399v: rubricated borders surround book headings (in epigraphic display majuscule, sometimes with scroll ornament): f. 1r square box containing Sasanian palmettes in medallions, palmettes and rinceaux with demi-palmettes, with vegetative ornament at corners, text in quatrefoil, ff. 196v, 399v rectangular boxes containing rinceaux with demi-palmettes, with palmette ornament at corners; ff. 90v, 295r, 347v: rubricated headpieces precede book headings (in epigraphic display majuscule, sometimes with scroll ornament): ff. 90v, 347v rectangular box containing rinceaux with demi-palmettes, with vegetative ornament at corners, f. 295r rectangular box containing palmettes; ff. 1r, 90v, 198r, 295r, 347v, 400r: rubricated major initials for start of books, usually with vegetative or other ornament; f. 409v: simple rubricated headpiece precedes note on translations: wavy line with space fillers, with palmette terminals; ff. 1v-3v, 26v, 46v-48v, 51v-52r, 58r-60r, 64r, 70v-71r, 89v, 109v-111r, 127v, 283v-284r, 286r-286v, 294v, 304r: borders around Biblical text occupying limited area in centre of page, laid out in decorative shape (ff. 1v-3v, 26v, 46v-47r, 89v, 109v-110r, 127v, 283v-284r, 286r-286v, 294v, 304r: quatrefoil, sometimes with vegetative ornament; ff. 47v, 70v-71r, 110v: circle; ff. 48r-48v, 51v-52r, 58r-60r, 64r, 111r: large circle with into smaller circle or lozenge below, sometimes with oval or rectangular base below that, very small circle above; ff. 1v-3v, 127v, 283v-284r, 286r-286v, 304r rubricated, elsewhere outline faintly marked out in brown crayon prior to copying of text but never rubricated; outline does not always correspond to actual layout of Biblical text); rubricated minor initials, sometimes with scrolling ornament; ff. 36r, 46r, 56v, 64r, 77r, 82v: simple dot ornament to Hebrew lection notes; ff. 91r, 103r, 119r, 132r, 141r, 150r, 159v-160r, 171r, 182r, 183r, 189r, 198r, 205r, 215v, 224v, 236r, 246r, 259r, 270r, 276v, 285v: decorative borders surround Hebrew lecture notes

Illustration

None

Binding

[Byzantine and western structural and decorative elements. The codex has encountered previous repairs and rebinding. Black tanned skin over wooden boards. Tooled in blind with small tools and fillets. Marks of five metal bosses on each cover, one only survives on the lower cover. Protruding plain primary endbands extending over the board edges. Vestiges of four tanned skin fastening straps through the lower cover.]

State of preservation

Moderate swarm damage to ff. 1-3, 412, minor swarm damage to ff. 4-11, 408-411; small tears from edges of ff. 1-2, 8, 10-11, 83, 95, 103, 211-214, 217; head of f. 1 repaired with paper patch; lower outer corner of f. 207 repaired with parchment patch; small holes in ff. 7, 76, 128, 129, 214, 407-412; close cropping to foot of f. 410; occasional partial loss of marginal notes and quire numbers due to cropping; possible singing to head of ff. 1-4, 410-412, diminishing inward; slight discoloration from water damage to head; some undulation, significant in early part of manuscript; some corrugation

Colophon(s)

f. 412v: 'ἐτελειώθη ή ίερὰ αὕτη βίβλ(ος) συν θ(ε)ῶ τῆς όκτατεύχου ἐπὶ βασιλέως μεγάλου ἐν χ(ριστ)ῶ πιστοῦ καὶ ὀρθοδόξου αὐτοκράτορος ῥωμαίων ἀλεξίου τοῦ κομνηνοῦ (καὶ) ἰω(άννου) μεγάλου βασιλέως τοῦ πορφυρογεννήτου μη(νὸς) νό<βεμβρίου> IB′ νυκτ(ὸς) ὥρ<ας> Θ′ ἔτους ἀπὸ κτίσεως κοσμ<ου> ΣΤΧΙΒ′ ινδ<ικτιώνος> IB′ προστάξει Λέοντ(ος) τοῦ μεγαλ'επιφανεστάτου πτωτονωβελλισιμ<ου> (καὶ) οἰκείου ἀν(θρώπ)ου τοῦ κρατ<αιοῦ> (καὶ) ἁγίου ἡμῶν βασιλ<έως>, τοῦ νικερίτ<ου>· διὰ χειρὸς ἰω(άννου) τοῦ εὐτε<λοῦς> (καὶ) ξένου τοῦ κούλικ(ος)· (καὶ) οί ἀναγινωσκοντες εὔχεσθε υπ<ὲρ> ἡμῶν διὰ τὸν κ(ύριο)ν:- ἀμὴν'

Notes and marks of ownership

Inside front board: ‘κ(αὶ) τὸ δε πρὸς τῖς ἄλλοις ἀλεξάνδρου, ὅστις δ′ ἄν βουλοι ἀφαιρέσει τὴν βίβλον, ὑπόδικος ἔστω ταῖς τῶν τριακοσί(ων) δέκα κ(αὶ) ὀκτῶ θεοφόρων πατέρων ἀραῖς, κ(αὶ) ποιστῶν τῶν δικαίων’; *ex libris* of Archbishop Charles Manners-Sutton 1805

Other notes

Foot: '+ Λευιτικον'; f. 26v: Hebrew note recording omission of two verses; f. 293v: calculation; f. 317r: ‘κ(ύρι)ε βοὴθὴ το σον δοῦλου ιω(άννου) αμαρτ(ώλου)’; f. 298r: partially illegible note identifying book of Joshua: 'יהושוע[....]' f. 319r: ‘γνωστὸν ἔστοσι ὦ ὁ κλησιάρχ ὅτι διὰ τὴν ἀγάπην τῆ ὀ αγι(ος) ὀσιήν’; f. 412v: ‘429 ἀπό τὸ ἀ.ας του χ(ριστο)ῦ τετρακόσια ἔτη κ(αὶ) εικοσι ἐννέα’; '412 leaves'; back board: ‘ἐν ἐτη ἀπὸ κτήσεος ΣΤΧΙΒ′’; calculations; ff. 1r, 3r, 10v, 89r, 93v, 126v, 136r, 200v, 201r, 221r, 245v, 265r, 266v, 268r, 269r, 269v, 271r, 272r, 366v, 367r, 407v, 409v, 412r, 412v: various brief annotations, some partially lost through cropping

Lost marks

f. 197r: partially erased folio count; f. 298r: partially erased Hebrew note; f. 319r: partially erased notes

Provenance

Acquired by J. D. Carlyle in the eastern Mediterranean 1800-1, purchased by Archbishop Charles Manners-Sutton and deposited in LPL 17 March 1806

Dating

Dated by colophon to 12 November 1103

Bibliography

Todd (1812), p. 264; James (1932), p. xxiii, pp. 840-843; Costas N. Constantinides and Robert Browning, *Dated Greek Manuscripts from Cyprus to the Year 1570* (Washington D.C. and Nikosia 1993), p. 68 and n. 2; *Repertorium*, vol. 1, no. 166, vol. 2, no. 222; Vogel and Gardthausen, pp. 174-175; Annemarie W. Carr, *Byzantine Illumination 1150-1250: the study of a provincial tradition* (Chicago 1987), p. 159 n. 16