mastpoint.com

CSV-1203

CSV File Format Specification

Best practice for **business-to-business** I.T. operations.

CSV-1203, CSV File Format Specification,
First Edition, Revision D, 20 February 2013
by Clive Hudson

Published in the United Kingdom.

Published by mastpoint.com

Author: Clive Hudson
Copyeditor: Michael Chambers
Proofreaders: Michael Chambers, Andrew Keast
Illustrator: Clive Hudson

Other contributors:
Olivier FREIMAN, *Ficorp Technologies*, Paris, France

TABLE OF CONTENTS

Introduction

CSV (***comma-separated values***) is a simple file format widely used by business and scientific applications. However, there is no universally accepted CSV file format specification. CSV-1203 is an open standard which attempts to provide a complete, accurate and reliable description of the CSV file format as used by companies for information exchange.

A standard with benefits

By adopting this standard, you place a prudent limit on the otherwise countless variations that could be implemented by systems developers. Its benefits are clearest when a CSV file forms an information bridge between two companies. Typically you should expect this standard to help reduce the time it takes to establish a data processing connection between your company and your clients and service providers.

It is possible that your IT department has already adopted the measures of this standard, but simply has no formal documentation to support that position. You are welcome to refer to this document when discussing file exchanges with any other organization. The only condition for using this standard is that you respect the author's copyright by not publicly re-publishing all or part of its contents.

Open standard

Should you wish to contribute to the contents of this standard, please contact the author at the web address given below.

Please be aware that this standard may be revised and refined over time. The latest release can be freely downloaded as a PDF from the website:

http://mastpoint.com/csv-1203

Other guidance

Other efforts have been made towards a CSV standard.

RFC4180

The ***Network Working Group*** (NWG) is a working group within the Internet Engineering Task Force (IETF) which is responsible for developing and promoting Internet standards.

NWG's informational memo ***RFC4180*** documents a set of seven statements relating to CSV files. The memo's title is *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. However, the memo falls short of a rigorous and useful specification suitable for commercial *business-to-business* operations.

It appears that RFC4180's rule 2 breaks rule 1. Elsewhere, it contains confusing and misleading information including, but not limited to, the behaviour of Microsoft's Excel spreadsheet application.

The RFC4180 nominated mime type for CSV files is **text/csv**.

The RFC4180 nominated file type extension for CSV files is **.CSV** (this excludes Macintosh).

Other file formats used for B2B data exchange

Today, many companies still regularly use and depend on CSV files for exchanging data with others. However, there are other plain text file formats the systems developer may consider.

CSV is ideal for exchanging *saturated homogeneous* tabular data.

TXT is sometimes used to transfer saturated homogeneous tabular data, but using a fixed-length record format.

XML is ideal for exchanging *sparse, hierarchical* or *heterogeneous* data.

More recent developments include **YAML** and **JSON**. Both claim to be human-readable data serialization formats. They are useful for exchanging hierarchical, heterogeneous and complex data structures, including associative arrays.

CSV's association with Excel

It is perhaps unfortunate that in the minds of many users there is a strong association between **.CSV** files and Microsoft's **Excel** spreadsheet application. Excel is definitely not CSV-file friendly.

To be clear, this standard does not encourage the use of Excel when handling CSV files. Instead, it attempts to mitigate some of the inevitable problems caused by non-technical users who produce and consume CSV files using this tool, unaware of Excel's limitations.

Files that have a single data column

A special case exists where a file contains just a single column of data and is assigned the file type of .CSV.

It is not strictly correct to say it is a CSV file since there are no multiple values being separated with commas. Only the strictest interpretation of this standard would require the presence of at least two data columns. It is recommended that CSV consuming applications are capable of handling this special case.

Terminology

Throughout this document, several words are used to signify the requirements of this specification. These words are capitalized when they should be interpreted as having a strict meaning. The following definitions are taken from **RFC2119** and modified so that they are more appropriately worded for use within this standard.

MUST This word means that the definition is an absolute requirement of this specification.

MUST NOT This phrase, or the phrase "**MUST NEVER**," means that the definition is an absolute prohibition of this specification.

SHOULD This word means that valid reasons may exist in particular circumstances to ignore a particular item, but the full implications must be understood and be carefully considered before choosing a different course.

SHOULD NOT This phrase means that valid reasons may exist in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully considered before implementing any behaviour described with this phrase.

MAY This word means that an item is truly optional.

One business unit may choose to include the item because a particular marketplace requires it or because the business unit feels that it enhances the product while another business may omit the same item.

An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. Conversely, an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides).

ABNF

This document uses ABNF (Augmented BNF for Syntax Specifications, RFC5234) to express certain rules where appropriate. For example:

```
<csvfile>  = csvheader *datarecord [EOF]
```

A complete rule set can be found in Appendix C.

The **CSV**-1203 standard

This is a specification detailing what constitutes a correctly formatted CSV file according to this standard CSV-1203.

CSV-1203 addresses the needs of business-to-business operations and is intended for use by IT professionals. It is arranged into 10 sections. Each section contains supporting notes and two or more rules.

Care has been taken in the choice of the rule vocabulary. Each technical term is followed by its definition when it is first used. Technical terms may also appear in the glossary (starting on page 30).

Some rules are thought to benefit from being paraphrased to help reduce the risk of a misunderstanding. These additions will be given in italics immediately after the rule in question.

1 CSV Files

A CSV file can be thought of as being in transit between a ***producer application*** and a ***consumer application***. Their primary function is to transport text-based data. They are not intended for the transportation of binary data.

A *producer application* is one which creates or appends to a CSV file. A *consumer application* is one which reads a CSV file.

A CSV file consists of two types of data: ***payload data*** and ***payload delivery markers***. *Payload data* is that which is intended to be transported from one application to another. *Delivery markers* are used to organize the payload data while it is carried within the CSV file.

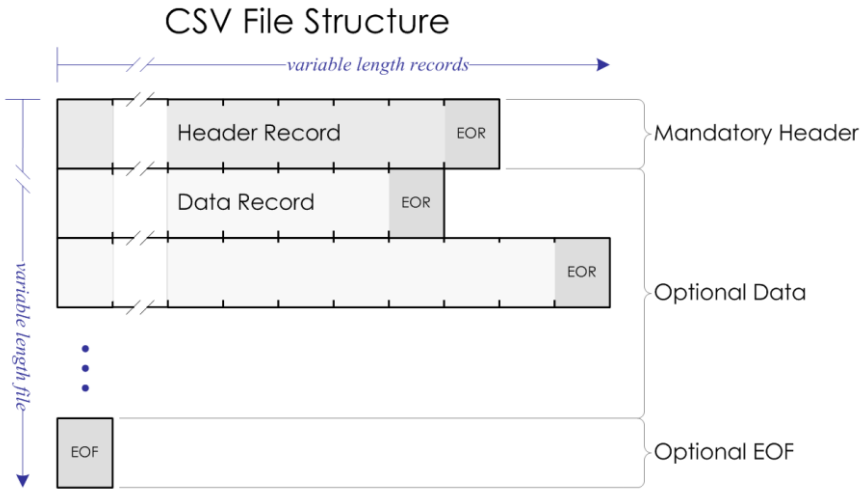


Diagram 1 : CSV File Structure

The following rules apply to the file as a whole.

Rules

1.1 The file type extension MUST be set to **.CSV**

The CSV file extension must be used even when the selected field separator character is not a comma.

1.2 The character set used by data contained in the file MUST be an 8-bit superset of the 7-bit US-ASCII.

This guarantees the consumer application that the first 128 characters are always present and that the payload delivery markers can be reliably interpreted.

Three common 8-bit supersets include Windows 1252, ISO/IEC 8859-1 and UTF-8.

Please refer to Appendix B for the US-ASCII character chart.

1.3 ASCII control characters, other than payload delivery markers, MUST NOT be present in a CSV file.

A CSV file is not intended to transport binary data. This rule prohibits the use of most of the ASCII control characters.

1.4 A CSV file MUST contain at least one record.

CSV file must not be empty (have zero length) or consist only of a logical end of file marker. The minimum number of records a CSV file can contain is one: The header record followed by zero, one, or more data records. There is no specified limit to the number of data records.

Syntax

```
    <csvfile>  = csvheader *datarecord [EOF]
    csvheader  = csvrecord
    datarecord = csvrecord

; Literal terminals
    EOF = %x1A
```

2 CSV Records

A CSV record consists of two elements: a *record payload* followed by an *end-of-record marker* (EOR). The EOR is a payload delivery marker and does not form part of the data delivered by the record.

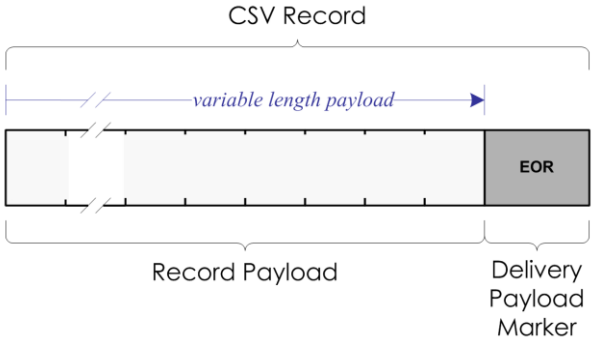


Diagram 2 : CSV Record

There is no specified upper limit to the length of a record payload.

These rules apply to the header record and all data records.

Rules

2.1 A CSV record consists of a payload followed by an end-of-record marker.

This rule applies to every CSV record, including the last record in the file. The lack of an EOR marker can suggest a file has suffered data loss by truncation.

An EOR marker is one to two characters in length and usually depends on the operating system used to run the producer application.

2.2 The length of the record payload MUST be greater than zero.

It may be a common requirement for consumer applications to discard blank records. However, a blank record is non-compliant.

Syntax

The following rules assume that the comma is selected as the field separation character.

```
    csvrecord     = recordpayload EOR
    recordpayload = fieldcolumn COMMA fieldcolumn *(COMMA fieldcolumn)

; Literal terminals
    EOR   = CRLF / CR / LF
    CRLF  = CR LF
    COMMA = ","
    CR    = %x0D
    LF    = %x0A
```

3 Field Columns

CSV is typically used to transport homogeneous tabular data. When viewed in a spreadsheet, payload data inside a CSV is visually arranged into a series of rows (*records*) and columns (*fields*). Hence the term ***field column***.

In fixed-length-field records, the relative location of each field within the record is fixed. However, CSV is a file format which allows variable length records. This enables them to save substantial space over the fixed-length format. To do this, CSV records employ a payload delivery marker called a ***field separator*** which indicates the transition from one field to the next. The field separator is represented by a single character.

The meaning associated with a data item within a *field column* is implied by its relative position within the record. For example, column A might contain a transaction date, while column B contains a transaction amount, etc. It is because the meaning is *implied relative,* the same character can be used to separate every field.

The presence of a header record is therefore vital because it explicitly details the predetermined order of field columns.

The following diagram shows where the field separation characters (SEP) are used. It is possible for a record to consist of nothing but field separators.

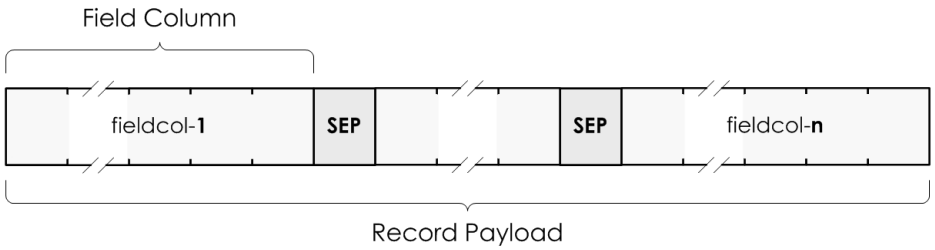


Diagram 3 : Fields within a record payload

A field column can contain zero, one or more characters.

Rules

3.1 Each record within the same CSV file MUST contain the same number of field columns.

This rule perhaps highlights the difference between the casual use and the professional use of CSV files. A communication between two companies should be clear and explicit. The header record declares how many fields the application should expect to process.

3.2 A record payload MUST contain at least two field columns.

If the minimum number was set to one column, it would mean that blank records could exist, breaking rule 2.2. The function of a CSV file is to transport values separated by commas. A system design which prescribes a CSV file with only one field column (a

simple list) would perhaps do better to consider using a normal text file.

3.3 Field columns MUST be separated from each other by a single separation character.

CSV records frequently contain empty fields resulting in a record which contains nothing but separation characters. This is acceptable.

3.4 A field column MUST NOT have leading or trailing whitespace.

This rule includes the header record.

*If the data payload must include leading or trailing whitespace, the field column MUST use double-quotes to delimit the payload (see **field protection** below).*

Although Excel tolerates leading and trailing spaces, it does not follow that it should form part of this standard. Leading and trailing spaces can often be the result of a text cut & paste session, where the user did not intend the spaces to remain.

*Other whitespace characters that appear in the selected character set beyond the 7-bit ASCII sequence, such as the Unicode 'hard-space' character ** ** (`%xA0`), should also be included in this exclusion rule.*

Syntax

The following rules assume that the comma is the selected separator character. This assumption greatly simplifies the ABNF rules.

```
    recordpayload    = fieldcolumn COMMA fieldcolumn *(COMMA fieldcolumn)
    fieldcolumn      = protectedfield / unprotectedfield
    protectedfield   = DQUOTE [EXP] fieldpayload DQUOTE
    fieldpayload     = 1*anychar
    unprotectedfield = [EXP] rawfieldpayload
    rawfieldpayload  = safechar / (safechar *char safechar)
    anychar          = char / COMMA / DDQUOTE / CR / LF / TAB
    char             = safechar / SPACE
    safechar         = %x21 / %x23-2B / %x2D-FF

; Literal terminals
    COMMA   = ","
    EXP     = "~"  ; Excel protection marker
    CR      = %x0D
    DQUOTE  = %x22
    DDQUOTE = %x22 %x22
    LF      = %x0A
    SPACE   = %x20
    TAB     = %x09
```

4 The Field Separator

Although the file format name 'comma separated values' suggests the comma is the field separator, some producer applications use different characters. The following rules limit the choice of separator characters and how they are applied within a CSV file.

Rules

4.1 The field separator MUST be a single character.

4.2 Once selected, the same field separator character MUST be used throughout the entire file.

4.3 The producer application SHOULD use the comma (ASCII 0x2C) as the field separator.

This rule raises one of the most difficult issues faced by developers when implementing code to process a CSV file: Separators embedded in payload data. This is covered by the field protection rules in this specification.

The use of characters other than the comma can indicate that the developer is most likely attempting to avoid the issue of data-embedded commas.

4.4 The consumer application MUST accept either the comma, the tab character (ASCII 0x09), the vertical bar (pipe) character (ASCII 0x7C), or semicolon (ASCII 0x3B) as the field separator.

4.5 The field separator MUST NOT be taken as part of a field.

This ensures data compatibility between systems that use different separator characters.

Syntax

The ABNF rules provided in this document have assumed that the comma is the selected separator character. This assumption greatly simplifies the rules set. The following expresses the isolated rule 4.4.

```
    separator = "," / ";" / PIPE / TAB

; Literal terminals
    PIPE      = %x7C
    TAB       = %x09
```

Standards design note

The use of the comma separator in rule 4.3 is strongly recommended. When used in conjunction with rule 4.4, it can be used to establish a convergence, helping producer applications popularize the use of the recommended separator.

5 The End-of-Record Marker

The end-of-record (EOR) marker used in text files such as CSV often varies depending on the operating system in which the file was created. Two ASCII control characters that are commonly used to form an EOR are the Carriage Return (**CR** - ASCII 0x0D) and the Line Feed (**LF** - ASCII 0x0A). This specification permits three variations of EOR marker.

Rules

5.1 The end-of-record marker MUST be one of the following: CR+LF, LF or CR.

CR is used by a range of operating systems including Mac OS up to version 9 and OS-9. CR+LF is used on Microsoft Windows and MS-DOS variants. LF is used on Unix and Unix-like systems including Linux variants.

5.2 Once selected, the same end-of-record marker MUST be used throughout the file.

This rule enforces consistency. This can become an issue where two files produced on different operating systems are concatenated without first normalizing them.

5.3 The EOR marker MUST NOT be taken as being part of the CSV record.

This rule ensures data compatibility between different operating systems.

If the character(s) which form an EOR are purposefully used within the record's data payload, the field will be double-quote protected and they will be part of the record.

Syntax

```
    EOR = CRLF / CR / LF

; Literal terminals
    CRLF = CR LF
    CR   = %x0D
    LF   = %x0A
```

EOR and EOL distinction

The EOR marker is not an *end-of-line* (EOL) marker or line break. This distinction is required so that CSV files which contain CR/LF characters in their data payload can occupy multiple lines on a page, but still be referred to as a single record.

This standard would also make it clear that the EOR marker is a record terminator and not a record separator.

6 The Logical End-of-File Marker

Today, few producer applications write a logical end-of-file (EOF) marker (**SUB** - ASCII 0x1A) to a file to indicate its end. To permit backward compatibility, a CSV consumer application must handle this marker correctly.

The presence of an EOF must be tested for if the producer application intends to append data records to an existing CSV file. In these cases, the EOF marker must be overwritten so that it does not cause a consumer application to prematurely stop while reading the file. Preserving the presence of an EOF is at the discretion of the producer application's developers.

Rules

6.1 The CSV file MAY be logically terminated with an end-of-file (SUB - ASCII 0x1A) marker.

It is important to note here that the EOF character indicates a logical EOF and not the physical end. It is the responsibility of the producer application to ensure the EOF is correctly positioned at the physical end of the file.

6.2 A consumer application MUST treat the logical EOF as if it were the physical end of the file.

Once a consumer application encounters an EOF marker, it must stop processing the file at that point. All other data that appears after the first EOF will be ignored.

6.3 A logical EOF marker cannot be double-quote protected.

A logical EOF marker is unique among all other CSV file markers in that it cannot be overruled by applying double-quote protection.

6.4 The EOF marker MUST NOT be taken as being part of the last CSV record.

The EOF marker is a payload delivery marker.

Syntax

The `<csvfile>` item is the principal root element to the ABNF rule set.

```
    <csvfile> = csvheader *datarecord [EOF]

; Literal terminals
    EOF = %x1A
```

7 The Header Record

It is common for manually prepared CSV files to have blank or missing columns or to contain additional columns that are not expected by the consumer application. A header record allows the consumer application to guard against these issues by comparing it to the expected presence and order of field columns.

A ***header label*** is the text which appears in a header record's field column.

Rules

7.1 The header record MUST be the first record in the file.

Apart from its physical position at the start of the file, there is no marker to indicate that a record is the header record or a data record.

7.2 A CSV file MUST contain exactly one header record.

7.3 Header labels MUST NOT be blank.

7.4 Header labels MUST NOT contain variable data.

Variable data in the header record prevents it from being used to validate the file's format.

7.5 Each header label SHOULD provide a semantic clue to the nature of the data to be found in its field column.

This is a guide for developers of producer applications. The quality of a column name can only be judged by the developer of the consumer application in the absence of supporting technical documentation.

Syntax

```
csvheader = csvrecord
```

Header labels

The selection of a header label is difficult to prescribe. However, there are some casual guidelines which should be useful to developers.

1. Keep header labels short.
2. Try to use a single word.
3. Do not use spaces.

   If words must be separated, use hyphens or underscore marks.

4. The first field column's header label should avoid using a word beginning with the two capital letters **ID**, without applying *Excel protection*.

   Otherwise an unwanted software feature within Excel will misinterpret the entire file as a mal-formed SYLK file.

8 Data Records

The usual purpose of a CSV data record is to contain payload data relating to a single entity. Ideally, each entity should be of the same type. For example, a list of logged server events, a list of customers or prospects, but not a mix of financial transactions and logged telephone calls.

Rules

8.1 A CSV file MAY contain zero, one, or more data records.

There is no specified upper limit to how many records can be placed inside one file. If a file contains zero data records, it must still contain one header record.

8.2 If a record has a record type identifier, it SHOULD be stored in the first field column in the record.

The purpose of this rule is to help standardize the design of CSV file layouts.

The assumed function of a CSV file is to transport data relating to a single type of entity. The presence of a record type code overrides that basic assumption and should therefore be placed in a prominent position.

8.3 Trailer records MUST NOT be used.

Syntax

```
csvfile    = csvheader *datarecord [EOF]
datarecord = csvrecord
```

Guarding data integrity

It is common for some commercial B2B files to include a trailer record which contains various file metrics that usually include the number of data records in the file.

If a trailer record is present, it must have the same number of field columns as the header record, and each must contain data of the same type as is found in each data record. In addition, there must be some way for the consumer application to recognize it as the trailer and not a mal-formed data record. These issues are difficult to address for a good reason. CSV files are singularly suited to transporting homogeneous tabular data. Mixing data with meta-data abuses that simple design.

Instead, if data integrity is of true importance, it is suggested that an MD5 signature be generated for the entire file, and that the signature be transported via a separate text file using the same file name as the CSV, but using the **.MD5** file extension.

It is possible that the trailer contains a total figure that you expect the consumer application to use as part of a checking mechanism. This thinking is possibly flawed because it assumes the consumer's checking-logic to be perfect.

9 Field Payload Protection

This is the most complex aspect of the CSV file format, and as a result, it is one of the most likely reasons why there is no general CSV standard. Simply changing the field separation character does not guarantee that the resulting file will be immune from format corruption.

Rules

9.1 If a field's payload contains one or more commas, double-quotes, CR or LF characters, then it MUST be delimited by a pair of double-quotes (ASCII 0x22).

When the field does not contain these specific characters, then double-quote protection is not required (see rule 9.3).

Because of the field protection mechanism, a CSV record will always contain either an even number of double-quotes or none at all.

9.2 If a field's payload contains a double-quote character, it MUST be represented by two consecutive double-quotes.

9.3 The producer application SHOULD NOT apply double-quote protection where it is not required.

This may occur if the producer application is blindly protecting every field, even the empty ones. This behaviour wastes space and can have a significant impact on the size of the file. In the worst case scenario it can increase file size by almost 300%.

Excel might behave in an unreliable way if it encounters a single record file with leading double-quote protected fields.

Syntax

```
    protectedfield   = DQUOTE [EXP] fieldpayload DQUOTE
    fieldpayload     = 1*anychar
    anychar          = char / COMMA / DDQUOTE / CR / LF / TAB
    char             = safechar / SPACE
    safechar         = %x21 / %x23-2B / %x2D-FF

; Literal terminals
    COMMA   = ","
    EXP     = "~"  ; Excel protection marker
    CR      = %x0D
    DQUOTE  = %x22
    DDQUOTE = %x22 %x22
    LF      = %x0A
    SPACE   = %x20
    TAB     = %x09
```

Double-quote protection

When a single field contains the character used to separate fields, there has to be a mechanism which prevents the CSV consumer application from becoming confused. The standard mechanism is based on delimiting the field using double-quotes (ASCII 0x22).

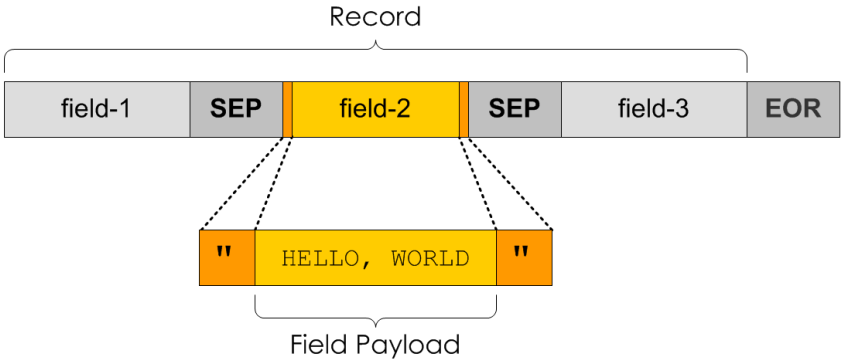


Diagram 4 : A protected field within field separators.

This delimiter mechanism solves the problem of embedded separation characters and EORs, but creates a new problem: What if the field is supposed to contain a double-quote?

Double double-quotes

The second part to the field payload protection mechanism is to allow fields to contain double-quotes. The following diagram is in the context of the consumer application.

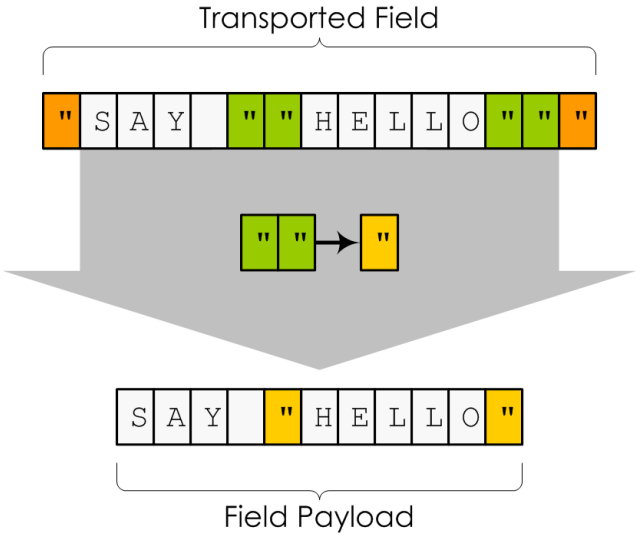


Diagram 5 :Two-for-one translation of double-quote pairs.

This is achieved by using two consecutive double-quotes for every individual double-quote intended to be transported in the field payload. The consumer application need only perform a two-for-one character translation of everything inside the delimited field's payload.

RFC4180 terminology

Contrary to RFC4180's rule 7 description, the first double-quote of the pair is not an *escape character*.

Instead, it is a simple two-for-one translation mechanism. This is because when the first double-quote is encountered by the consumer application, it must check the following character to confirm that a translation is required.

In an escape-character mechanism, the application would know how to process the next character as soon as it encountered the first double-quote. This is not possible because it could just as easily have encountered the closing double-quote at the end of the protected field.

Leading & trailing whitespace

When the field payload contains leading or trailing whitespace, or consists only of whitespace that must be delivered, then double-quote protection must be applied.

10 Excel Protection Marker

The Excel spreadsheet application is known to corrupt payload data when used to edit CSV files. To overcome this, the CSV-1203 specification introduces the tilde '~' (ASCII 0x7E) as the Excel protection marker.

To use this marker, just prefix it to the field's payload. Its presence stops Excel from automatically misinterpreting the contents of the field.

Compliant consumer applications must remove the leading tilde before the field can be processed as normal.

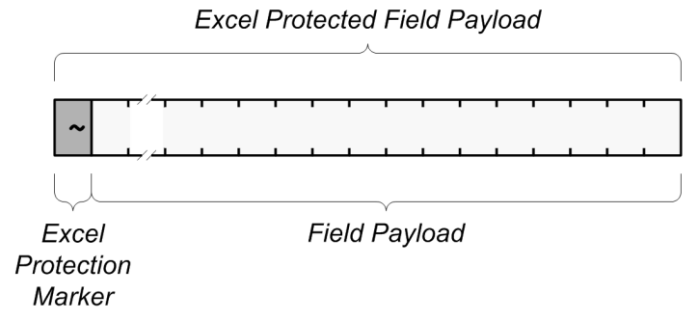


Diagram 6 : The tilde prefix can protect long-number fields from Excel

This data protection mechanism can also be applied to the header record.

Rules

10.1 If a field requires Excel protection, its payload MUST be prefixed with a single tilde character.

When a field has both payload protection and Excel protection, the tilde is within the double-quote enclosure.

The tilde character and field payload remains visible and fully editable in Excel because it is unaware of this standard.

The compliant CSV consumer application must be aware of this marker and discard it before processing the field payload data.

10.2 If a field's payload begins with a tilde character, then Excel protection MUST be applied to that field.

This rule only applies to the tilde character if it is the first character within the field payload. Tilde characters elsewhere in the payload area do not need to be doubled up.

10.3 Any field in any record MAY use Excel protection without requiring any other field to use the same protection.

Dual protection

The presence of a tilde character in a field does not require that field to use double-quote protection.

However, combining both field payload protection and Excel protection is straight forward since the Excel protection marker is treated as if it were the first character of the field payload.

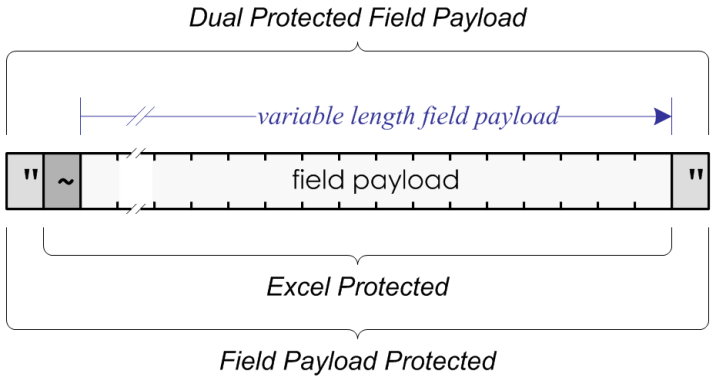


Diagram 7 : Arrangement of payload markers providing dual protection.

The consumer application must still exclude the Excel protection marker from the field payload.

Syntax

```
    protectedfield   = DQUOTE [EXP] fieldpayload DQUOTE
    unprotectedfield = [EXP] rawfieldpayload
    fieldpayload     = 1*anychar
    anychar          = char / COMMA / DDQUOTE / CR / LF / TAB
    rawfieldpayload  = safechar / (safechar *char safechar)
    char             = safechar / SPACE
    safechar         = %x21 / %x23-2B / %x2D-FF

; Literal terminals
    COMMA   = ","
    EXP     = "~"  ; Excel protection marker
    CR      = %x0D
    DQUOTE  = %x22
    DDQUOTE = %x22 %x22
    LF      = %x0A
    SPACE   = %x20
    TAB     = %x09
```

APPENDIX A

Known issues when using Excel

All of the following issues can be avoided if this standard's *Excel Protection* rules are applied to vulnerable fields within a CSV file.

Excel is a very powerful spreadsheet application, but is often used to create, edit and save database files in CSV format. There are a number of currently known issues that Excel users should be aware of when using it as part of a routine data processing procedure. The principal difficulty is that Excel is not a WYSIWYG (*what-you-see-is-what-you-get*) editor. It will not store all data exactly as it appears on screen.

A.1 Long numbers (unwanted floating-point conversion)

In business, credit card and reference numbers often appear as long integers and are affected by this issue. Excel appears to become problematic at two threshold lengths.

First is the **display threshold**. Numbers that contain 9 to 11 digits are automatically displayed in a shortened scientific notation with a 3-digit mantissa. However, although these values are displayed incorrectly, they are saved correctly. Resizing the column width to fit the contents will display these numbers correctly without using scientific notation. The display threshold appears to be related to the default visual width of the column as displayed in Excel's data grid and may vary from one installation to another.

Second is the **save threshold**. Numbers that contain 12 or more digits are automatically displayed in a shortened scientific notation but saved using a 6-digit mantissa. Be aware that when meeting this threshold, data will be corrupted if the file is saved using Excel. This has the effect of converting the least significant digits to zero because of the internal rounding-up limits of Excel's default single-precision 6-digit mantissa in scientific notation.

The following table shows the dual threshold issue and where the risk of data corruption lies:

Data	Excel Display	Excel Save	Re-loaded Image
12345678	12345678	12345678	12345678
123456789	1.23E+08	123456789	123456789
1234567890	1.23E+09	1234567890	1234567890
12345678901	1.23E+10	12345678901	12345678901
123456789012	1.23E+11	1.23457E+11	123457**000000**
1234567890123	1.23E+12	1.23457E+12	123457**0000000**

A.2 Excel experiment

To see these effects for your installation, try the following experiment. Using a simple text editor, create a CSV file containing the list of numbers given in the *Data* column in the above table. Open the file using Excel. Moving the cursor to each cell shows its expected value

in the field editing bar. Resize column A to reveal how each value will be saved. Save the file as CSV. Close Excel and open the CSV file using a simple text editor. You should then see how the data has been affected.

A.3 Leading and trailing zeroes

Excel will strip trailing zeroes from currency amounts. The effects may differ depending on the regional and language settings of the computer hosting Excel. The last example shows what happens when you *Excel protect* the field.

Data	Excel Display	Excel Edit Bar	Excel Save
0	0	0	0
00	0	0	0
01234123456	1234123456	1234123456	1234123456
01234 123456	01234 123456	01234 123456	01234 123456
" 0"	0	0	0
0	0	0	0
"0 "	0	0	0
0	0	0	0
0001	1	1	1
"0001"	1	1	1
"1.0000"	1	1	1
1.0000	1	1	1
"1,200.00"	1,200.00	1200	"1,200.00"
"£1,200.00"	£1,200.00	1200	"£1,200.00"
"$1,200.00"	$1,200.00	$1,200.00	"$1,200.00"
~001.0000	~001.0000	~001.0000	~001.0000

Telephone numbers and reference numbers 12 digits or longer are likely to be converted into floating-point values.

Leading and trailing zeroes can also be preserved by placing an equal sign ('=') in front of a double-quoted value. For example:

```
="000012.030000"
```

This method breaks the most commonly shared CSV convention relating to the use of double-quotes as payload delimiters, and is therefore not recommended.

A.4 Strings that appear to be calculations

Not surprisingly, if a field begins with an equals sign followed by a simple formula, Excel will attempt to compute the answer. Saving the file will cause the original data to be lost. The last example shows what happens when you apply the Excel protection marker.

Data	Excel Display	Excel Edit Bar	Excel Save	Re-loaded Image
=1/0	#DIV/0!	=1/0	#DIV/0!	#DIV/0!
=3/A	#NAME?	=3/A	#NAME?	#NAME?
=13/12	1.083333	=13/12	1.083333333	1.083333333
~=13/12	~=13/12	~=13/12	~=13/12	~=13/12

A.5 Strings that are interpreted as dates

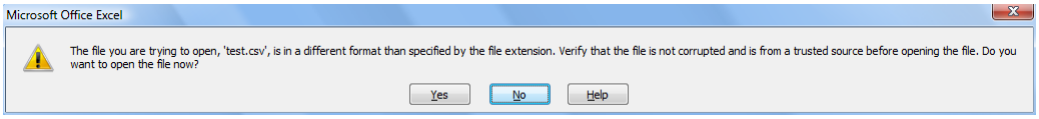
Several problems exist when handling strings that appear to be dates. This problem can affect some address lines which lack the street name. The last example shows what happens when you apply the Excel protection marker to the field.

Data	Excel Display	Excel Edit Bar	Excel Save	Re-loaded Image
0/1	0/1	0/1	0/1	0/1
1/1	01-Jan	01-01-2012	01-Jan	01-Jan
12/12	12-Dec	12-12-2012	12-Dec	12-Dec
13/12	13-Dec	13-12-2012	13-Dec	13-Dec
32/12	32/12	32/12	32/12	32/12
12/13	Dec-13	01-12-2013	Dec-13	Dec-13
12/30	Dec-30	01-12-1930	Dec-30	Dec-30
~1/1	**~1/1**	**~1/1**	**~1/1**	**~1/1**

Excel attempts to determine the century based on the current calendar year.

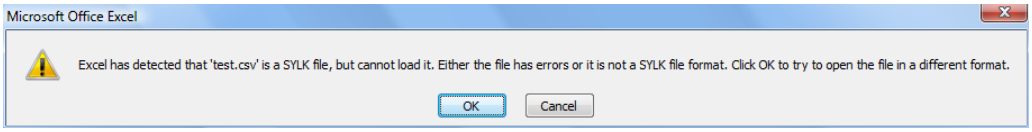
A.6 SYLK file identification issue

In certain versions of Excel, loading any CSV file will raise the following warning message box if the first two characters of the file contain the capital letters **ID**.



> *"The file you are trying to open, <filename>, is in a different format than specified by the file extension. Verify that the file is not corrupted and is from a trusted source before opening the file. Do you want to open the file now?"*

If you answer yes (which you must do in order to open the file), you will see the following message box.



> *"Excel has detected that <filename> is a SYLK file, but cannot load it. Either the file has errors or it is not a SYLK file format. Click OK to try to open the file in a different format."*

This issue is eliminated if the header record is *Excel protected*.

A.7 User generated issues

Some non-technical Excel users may believe that XLS and CSV files are interchangeable, and so they might mistakenly save a table in XLS, XLSX, XLSM or XLSB format and simply rename the file to .CSV.

A.8 Manually editing @Twitter names

Excel will not allow users to enter a twitter name if the @ symbol is used as a prefix. Attempting to do so will raise an error message box stating "*That function is not valid.*".

Loading a file that already contains @Twitter names is acceptable, but you may not be able to edit them without encountering the error message box.

A.9 The surname of "TRUE"

Excel will automatically treat the valid surname "True" as if it were a Boolean value of TRUE. The same assumption also affects fields containing the word "false".

The presence of the surname True has the potential to cause problems when sorting a file by surname. Because Excel sees the cell contents as a simple Boolean value and not a string, it fails to sort True and False into their correct ASCII character order.

APPENDIX B

US-ASCII character chart

The characters defined as standard *payload delivery markers* are highlighted in blue. Alternate field separators are highlighted in green.

Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex
(nul)	0	0x00	(sp)	32	0x20	@	64	0x40	`	96	0x60
(soh)	1	0x01	!	33	0x21	A	65	0x41	a	97	0x61
(stx)	2	0x02	"	34	0x22	B	66	0x42	b	98	0x62
(etx)	3	0x03	#	35	0x23	C	67	0x43	c	99	0x63
(eot)	4	0x04	$	36	0x24	D	68	0x44	d	100	0x64
(enq)	5	0x05	%	37	0x25	E	69	0x45	e	101	0x65
(ack)	6	0x06	&	38	0x26	F	70	0x46	f	102	0x66
(bel)	7	0x07	'	39	0x27	G	71	0x47	g	103	0x67
(bs)	8	0x08	(	40	0x28	H	72	0x48	h	104	0x68
(ht)	9	0x09	)	41	0x29	I	73	0x49	i	105	0x69
(lf)	10	0x0A	*	42	0x2A	J	74	0x4A	j	106	0x6A
(vt)	11	0x0B	+	43	0x2B	K	75	0x4B	k	107	0x6B
(ff)	12	0x0C	,	44	0x2C	L	76	0x4C	l	108	0x6C
(cr)	13	0x0D	-	45	0x2D	M	77	0x4D	m	109	0x6D
(so)	14	0x0E	.	46	0x2E	N	78	0x4E	n	110	0x6E
(si)	15	0x0F	/	47	0x2F	O	79	0x4F	o	111	0x6F
(dle)	16	0x10	0	48	0x30	P	80	0x50	p	112	0x70
(dc1)	17	0x11	1	49	0x31	Q	81	0x51	q	113	0x71
(dc2)	18	0x12	2	50	0x32	R	82	0x52	r	114	0x72
(dc3)	19	0x13	3	51	0x33	S	83	0x53	s	115	0x73
(dc4)	20	0x14	4	52	0x34	T	84	0x54	t	116	0x74
(nak)	21	0x15	5	53	0x35	U	85	0x55	u	117	0x75
(syn)	22	0x16	6	54	0x36	V	86	0x56	v	118	0x76
(etb)	23	0x17	7	55	0x37	W	87	0x57	w	119	0x77
(can)	24	0x18	8	56	0x38	X	88	0x58	x	120	0x78
(em)	25	0x19	9	57	0x39	Y	89	0x59	y	121	0x79
(sub)	26	0x1A	:	58	0x3A	Z	90	0x5A	z	122	0x7A
(esc)	27	0x1B	;	59	0x3B	[	91	0x5B	{	123	0x7B
(fs)	28	0x1C	<	60	0x3C	\	92	0x5C	|	124	0x7C
(gs)	29	0x1D	=	61	0x3D	]	93	0x5D	}	125	0x7D
(rs)	30	0x1E	>	62	0x3E	^	94	0x5E	~	126	0x7E
(us)	31	0x1F	?	63	0x3F	_	95	0x5F	(del)	127	0x7F

APPENDIX C

CSV-1203 expressed as an ABNF rule set

In the following rule set, the comma is assumed to be the selected separator character. The low-level rules describing character sets would need to be adjusted if you intend to use a different separator.

Syntax

```
; CSV-1203 file syntax

    <csvfile>        = csvheader *datarecord [EOF]
    csvheader        = csvrecord
    datarecord       = csvrecord
    csvrecord        = recordpayload EOR
    recordpayload    = fieldcolumn COMMA fieldcolumn *(COMMA fieldcolumn)
    fieldcolumn      = protectedfield / unprotectedfield
    protectedfield   = DQUOTE [EXP] fieldpayload DQUOTE
    unprotectedfield = [EXP] rawfieldpayload
    fieldpayload     = 1*anychar
    rawfieldpayload  = safechar / (safechar *char safechar)

; Character collections

    anychar  = char / COMMA / DDQUOTE / CR / LF / TAB
    char     = safechar / SPACE
    EOR      = CRLF / CR / LF
    safechar = %x21 / %x23-2B / %x2D-FF

; Literal terminals

    CRLF    = CR LF

    COMMA   = ","
    EXP     = "~"          ; Excel protection marker

    CR      = %x0D
    DQUOTE  = %x22
    DDQUOTE = %x22 %x22
    EOF     = %x1A
    LF      = %x0A
    SPACE   = %x20
    TAB     = %x09
```

<csvfile> is the root rule that defines the syntax for the entire CSV-1203 compliant CSV file.

APPENDIX V

Version Summary

Please be aware that this standard may be revised and refined over time. The latest release can be freely downloaded as a PDF from the website:

http://mastpoint.com/csv-1203

Date	Notes
17-03-2012	Initial release.
07-04-2012	Second draft released.
10-04-2012	Third draft released.
01-09-2012	E1. Revision B. Minor improvement to typography used in some diagrams. Documented issue of .CSV files which contain a single data column.
09-11-2012	E1. Revision C. Addition of new rules 6.3 and 6.4 (EOF Marker)
02-02-2013	E1. Revision D. Documented two additional operational issues relating to the use of Excel: A.8 - @Twitter names, and A.9 - The surname of "TRUE".

GLOSSARY

ABNF
Augmented BNF.

associative array
A collection of key and value pairs. The key indicates what type of information the value represents. The value can be any type of text-based data and can sometimes include other nested arrays.

B2B
Business-to-business.

BNF
Backus-Naur Form. A notation often used to describe the syntax of text-based instructions from single commands to entire artificial languages.

consumer application
A software application which reads a CSV file.

CSV
Comma Separated Values.

delimit
The act of placing a sequence of one or more characters at the start and end of a field's payload in order to clearly mark the boundary or limits of that payload.

EOF
End-of-file.

EOR
End-of-record.

field
A collection of contiguous characters that represent a single data item or value.

field column
A conceptual arrangement of data items. When a CSV file is loaded into a spreadsheet application, the data items are visually arranged into rows and columns. A field column is one such column, often referred to by its column letter code (A, B, C, ... Z, AA, AB,,, etc,) or by a simple field column index (0, 1, 2, ... etc.).

fixed-length record
The fixed-length record format implies that every field it contains is also fixed in length. Every record throughout the file has the same length.

This is perhaps one of the oldest forms of data storage with its origins dating back to at least the early part of the 18th Century, when punched cards were used to control looms.

heterogeneous data
In the context of data records, the word *heterogeneous* indicates a fundamental difference in the type of data the record contains, compared to neighbouring records. Compare *homogeneous data*.

hierarchical data
Data that can be viewed as belonging to a tree-like hierarchy, such as a components list of a system and sub-systems. The physical arrangement of the data conveys the structure of the hierarchy.

homogeneous data
In the context of data records, the word *homogeneous* indicates a uniform similarity in the type of data the record contains, compared to neighbouring records. Compare *heterogeneous data*.

JSON
JavaScript Object Notation. A lightweight text-based data format, suited to Internet-only applications.

nbsp
Non-breaking space. A whitespace character in Unicode (U+00A0) frequently used in the formatting of text that appears within HTML documents.

payload data
The collection of data items or values transported from the producer to the consumer application.

payload delivery marker
One or more characters whose function is to deliver data to a consumer application. Often, such markers are used to delimit the extent of a collection of characters that form a field, or a collection of fields which form a record.

pipe separated values
A variation of CSV wherein the comma is replaced with the vertical bar character. Invariably, the consumer application relies on the assumption that pipe characters can never appear in the payload data. This greatly simplifies the consumer's logic.

producer application
A software application that writes or appends information to a CSV file.

RFC2119
Request For Comments: 2119. Key words for use in RFCs to Indicate Requirement Levels.

RFC4180
Request For Comments: 4180. Common Format and MIME Type for Comma-Separated Values (CSV) Files.

RFC5234
Request For Comments: 5234. Augmented BNF for Syntax Specification: ABNF.

saturation
Saturation is a file-level metric which refers to the abundance of data provided in a CSV file. For example, a highly saturated file may contain data items in 90% or more of its columns in any given row.

The saturation level of a CSV file can be expressed as a percentage from 0% to 100%.

sparse data
Sparseness refers to the lack of data in proportion to the number of field columns. For example, if a spreadsheet contained 100 columns, it would be accurate to say it contained sparse data if only 10% of the columns were populated with values in any given row.

Sparseness is inversely proportional to the file's saturation level. See *saturation*.

tab separated values
A variation of CSV where the comma is replaced with the tab character. Invariably, the consumer application relies on the assumption that tab characters can never appear in the payload data. This greatly simplifies the consumer's logic.

whitespace
In typography, whitespace is a contiguous series of one or more characters that when printed, commands a horizontal or vertical movement of the printhead carriage mechanism without resulting in a visual mark on the page.

The ASCII whitespace characters permissible in CSV-1203 files are **HT**, **LF**, **CR** and **SP**. Please refer to the character chart in appendix B.

YAML
Yet Another Markup Language. A text-based data-oriented markup language capable of representing complex hierarchical data structures. Suited to Internet-only applications.