Public Health &
Intelligence

Unicode and Locale Encoding in SPSS

Document Control	
Version	Version 0.1
Date Issued	June 2017
Author	GPD Team – Roisin Farrell and Tina Fu
Comments to	NSS.isdGPD@nhs.net

Version	Date	Comment	Author
Version 0.1	June 2017	1st version of paper (ISD version)	GPD Team

Contents

Introduction 3
Unicode encoding 4
Locale encoding 4
Issues experienced by analysts in Public Health and Intelligence 5
a. *File originally saved in Locale and subsequently used in Unicode encoding* 5
b. *File saved in Unicode and subsequently used in Locale encoding* 7
Recommendation 7
Appendix 1a – Checking current encoding settings 8
Appendix 1b – File saved in Locale and subsequently used in Unicode encoding 8
Appendix 1c – File saved in Unicode and subsequently used in Locale encoding 9
References 10

Introduction

To begin, it will be helpful to understand the concept of Character Encoding. Whilst humans are familiar viewing documents as lines of text, computers regard them as a binary data series of ones and zeros, called "bits", the smallest unit of data that a computer uses. All computer programmes store information and execute commands in multiple 'bits', called 'bytes' (8 bits/1 byte). Therefore, the characters within a document (or in our case, a string of text), must be represented by numeric codes within a computer programme. In order to accomplish this, there are several types of encoding that modern operating systems (e.g. Windows 7) and programmes (e.g. SPSS) use to interpret these ones and zeros as readable characters (definition adapted from Tech Terms).
For more information, please see How-To Geek or Kunststube.

IBM SPSS Statistics currently uses two types of character encoding: Unicode and Locale. The current default in SPSS is Unicode. Appendix 1a provides syntax which enables a user to check which type of character encoding setting is adopted by their current SPSS session.

This paper is addressed to all analysts who use SPSS and explores the differences between Unicode and Locale encoding. SPSS users who experience issues when matching by string because of unexpected changes in variable length, or those who use non-English characters (e.g. è, ç, ë) will find this paper useful.

Note: Unicode and Locale encoding only affects string variables. Numeric variables will not be affected.

Unicode encoding

Unicode encoding uses a universal set of characters, based on a large number of locale character sets.

Pros:

- This encoding standard aims at universality. Alphabets that use accents (e.g. è, ë, ç) or non-Western alphabets can be used without loss of fidelity. In datasets that contain free text, information in languages other than English (e.g. forenames and surnames) will be kept intact.
- Sending data to external and international organisations is more reliable as it does not depend on the end user's system language or settings.
- When Unicode is used to both save and open files, there are no issues when files are matched.
- Default setting from SPSS version 16 onwards.

Cons:

- String variables are usually trebled in length if a file originally created and saved in Locale is used while Unicode encoding is activated. However, this would not happen if files are created in Unicode and subsequently used in either Unicode or Locale.
- If a file originally saved in Locale and opened in Unicode is then saved, the trebling of the string variable lengths means that the file size is slightly larger than the original.

Locale encoding

Locale encoding supports only those characters that the computer system (e.g. Windows 7) running SPSS is using. For example, the default system language in Public Health and Intelligence (PHI) is *English (UK)*. As a result, SPSS will not recognise any characters not in the English alphabet when using Locale.

Pros:

- Standard look up files (e.g. Scottish Postcode Directory) available on the ISD website are saved in Locale encoding.
- Most analysts within PHI have become used to using Locale and have created syntax based on that file setting.

Cons:

- Unicode is the default of SPSS 16 and above which means that each time an analyst uses a new PC the settings must be changed to accommodate any syntax written in Locale. If this is not done, error or warning messages may be displayed by SPSS, for instance when trying to match files.
- In SPSS, any characters that are not included in that language will not display properly when using Locale encoding, i.e. if a dataset uses free text and includes non-English names, they may not appear accurately.
- External (including international) organisations may have different system language settings than the ones PHI use, which presents issues when sending or receiving data from them.

Issues experienced by analysts in Public Health and Intelligence

Most analysts within PHI use Locale encoding. This is likely a legacy from previous versions of SPSS, as Unicode encoding was first introduced with version 16 of SPSS (released in 2007). However, the two types of encoding are not fully compatible and issues have arisen for analysts when switching between the two.

There are a number of ways in which the encoding settings can overlap, and can depend on the encoding used to save or open the file:

a. Files saved in Locale and subsequently opened in Unicode encoding
b. Files saved in Unicode and subsequently opened in Locale encoding

There are other ways they interact but the ones mentioned above are the main questions analysts have encountered.

a. File originally saved in Locale and subsequently used in Unicode encoding

This example considers the case where an analyst is working in an SPSS session set to Unicode encoding. As mentioned above, if an analyst tries to use a file which had previously been saved in a Locale encoding, SPSS automatically trebles all string variable widths to ensure that they are long enough (see example below):

Scottish Postcode Directory 2017_1 originally saved in **Locale** and currently used in **Locale** encoding:

	Name	Type	Width
1	pc7	String	7
2	pc8	String	8
3	SplitChar	String	1
4	PCDistrict	String	4
5	PCSector	String	6

Scottish Postcode Directory 2017_1 originally saved in **Locale** and currently used in **Unicode** encoding:

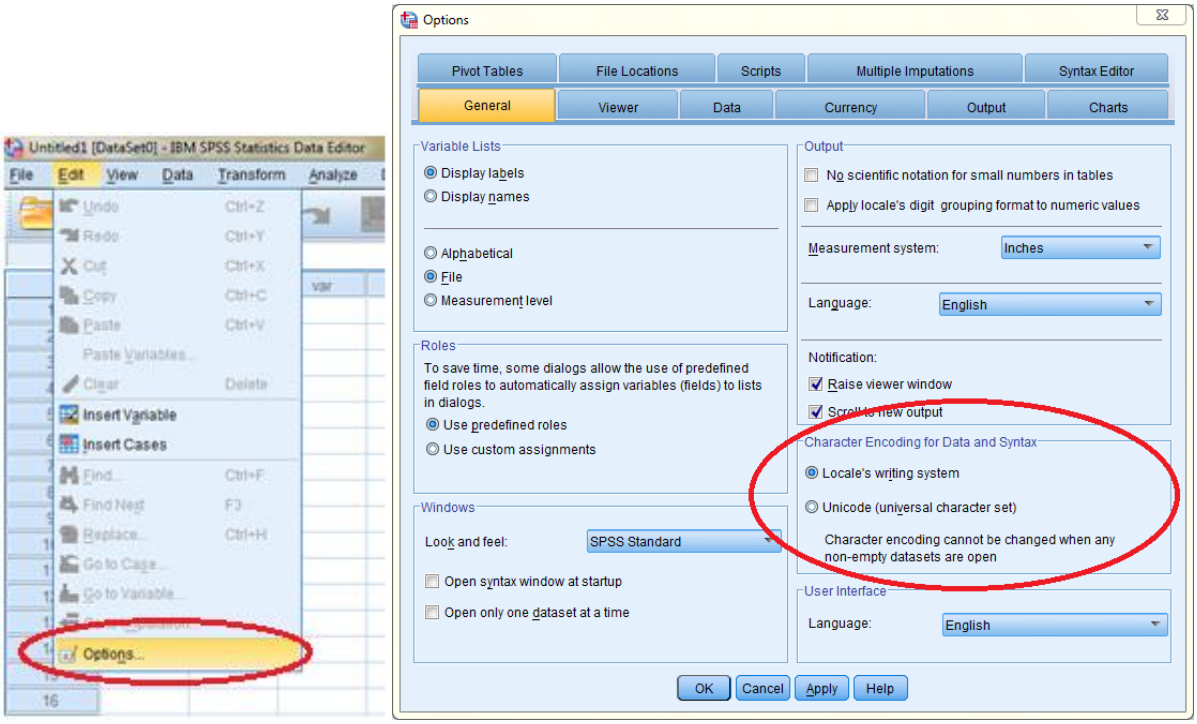
	Name	Type	Width
1	pc7	String	21
2	pc8	String	24
3	SplitChar	String	3
4	PCDistrict	String	12
5	PCSector	String	18

This can lead to problems if analysts use this file to match with or add to other files, unaware of the fact that the string length has been trebled. Appendix 1b provides a working example.

Solutions:

There are a few solutions to overcome this issue:

- Manually change the setting in the menu.
  - Open a blank dataset in SPSS
  - Go to Edit – Options – General: Character Encoding for Data and Syntax – Select “Locale’s writing system” – Click “OK”



- Programmatically

  Without changing the character encoding, there are several ways to write syntax to change the length of string variables.

  - **Alter type**
    - This command specifies a string variable and its length. This is useful when you have only one string variable to match on. Be careful to ensure that by using this syntax, you do not truncate any cases.

```
alter type pc7 (A7).
```

    - This command (A=AMIN) changes the length of all string variables, or a selection of string variables, to the minimum width necessary while avoiding truncating any characters. This is useful when there are multiple variables to match on.

```
Change the length of all string variables: alter type all (A=AMIN).
Change the length of a selection of string variables: alter type pc7 to PCsector (A=AMIN).
```

  - **Character substring**[1]
    - This command extracts part of the string variable 'pc7', starting at position 1 and ending after seven characters (i.e. at position 7). This is another way of changing the variable length.

```
string pc7_extract(A7).
```

[1] The 'char.substr' function applies to characters; the 'substr' function applies to bytes. This means that 'substr' may not work on characters that are more than one byte in length, e.g. non-English characters. In Unicode it is recommended to use 'char.substr'.

```
pc7_extract = char.substr(pc7,1,7).
```

Please note that the above solutions are temporary fixes and do not solve the underlying issue of incompatibility of the two datasets.

- Writing syntax to set Unicode on or off

  Please note that this command will not work if any of your open datasets have data in them.

```
DATASET CLOSE ALL.
NEW FILE.

set unicode on.
set unicode off.

or

set unicode yes.
set unicode no.
```

b. File saved in Unicode and subsequently used in Locale encoding

If a file is originally saved in Unicode and used in Locale encoding, the string length will not change. Hence, there will not be any problem matching files as long as the key variables for matching have the same string length. Appendix 1c provides a working example.

Recommendation

Best practice:
Analysts in PHI should adopt Unicode going forward as the default setting in SPSS.

The majority of syntax files across PHI have been written in a way that accommodates Locale encoding. However, using Unicode uniformly across syntax files and data files would decrease the errors of trebling of variables. Hence it is suggested that analysts in PHI should adopt Unicode going forward as the default setting in SPSS. However, when previous files need to be used in an analysis (e.g., trend analysis), the previous files that were originally saved in Locale should be re-saved in Unicode to ensure consistency. Please be aware of variable lengths when doing this; if variable lengths are trebled, you have to use the syntax below to change all variable lengths to the minimum width necessary:

```
Change the length of all string variables: alter type all (A=AMIN).
```

Appendix 1a – Checking current encoding settings

The following syntax can be used to check the setting in your current SPSS session:

```
Show unicode
```

The following message will appear in the Output Viewer:

System Settings

Keyword	Description	Setting
UNICODE	Whether the Processor is operating in Unicode mode	No

From this it can be seen that the user is currently working in the Locale encoding.

Appendix 1b – File saved in Locale and subsequently used in Unicode encoding

In this example files are matched in Unicode encoding.

```
Show unicode.
*Unicode is on.

*Open file originally saved in Unicode system. This file has the string lengths as required, e.g., pc7 is length 7.
Get file='H:\Stats Governance\Scottish_Postcode_Directory_2017_1_unicode.sav'.

*Match to file "Scottish_Postcode_Directory_2017_1_locale" which is originally saved in Locale system.
*Because the file "Scottish_Postcode_Directory_2017_1_locale" is now used in Unicode encoding, the length
of all string variables in this file gets trebled.

Match files file=*
    /table='U:\Scottish_Postcode_Directory_2017_1_locale.sav'
    /by pc7.
Execute.
```

The following error messages will be displayed in your syntax editor as "Mismatched variable types on the input files":

6	Match files	The BY variable list is invalid. All the variables on the list must exist on all the input files, and each variable must be of the same type (numeric, or strings of the same length) on all the input files.
6	Match files	Mismatched variable types on the input files.
6	Match files	The working file has been restored, and subsequent commands may access the working file.

The output viewer will display the following error message:

```
    Match files file=*
      /table='U:\Scottish_Postcode_Directory_2017_1_locale.sav'
      /by pc7.

Variable(s) incorrectly used in 'BY'
                                ------Input1------                    ------Input2------
Result                          Type Variable name                   Type Variable name
pc7                             s7   pc7                             s21  pc7

Codes:  num = numeric;  sn = string of length n
Error # 5126
The BY variable list is invalid.  All the variables on the list must exist on all the input
files, and each variable must be of the same type (numeric, or strings of the same length) on
all the input files.
Execution of this command stops.
```

```
Variable(s) with conflicting type:
                                 -------Input1-------                    -------Input2-------
Result                           Type   Variable name                   Type   Variable name
pc7                              s7     pc7                             s21    pc7
pc8                              s8     pc8                             s24    pc8
SplitChar                        s1     SplitChar                       s3     SplitChar
PCDistrict                       s3     PCDistrict                      s9     PCDistrict
PCSector                         s5     PCSector                        s15    PCSector
Date_Of_Introduction             s10    Date_Of_Introduction            s30    Date_Of_Introduction
Date_Of_Deletion                 s10    Date_Of_Deletion                s30    Date_Of_Deletion
PostcodeType                     s1     PostcodeType                    s3     PostcodeType
PCSuLink                         s8     PCSuLink                        s24    PCSuLink
Imputed                          s1     Imputed                         s3     Imputed
Split_Indicator                  s1     Split_Indicator                 s3     Split_Indicator
CA2011                           s9     CA2011                          s27    CA2011
UPC2005                          s9     UPC2005                         s27    UPC2005
SPR2014                          s9     SPR2014                         s27    SPR2014
SPC2014                          s9     SPC2014                         s27    SPC2014
EW2016                           s9     EW2016                          s27    EW2016
HB2014                           s9     HB2014                          s27    HB2014
HB2006                           s9     HB2006                          s27    HB2006
HSCP2016                         s9     HSCP2016                        s27    HSCP2016
CHP_2012                         s9     CHP_2012                        s27    CHP_2012
CHP_2011                         s9     CHP_2011                        s27    CHP_2011
CHP_SUBAREA_2011                 s9     CHP_SUBAREA_2011                s27    CHP_SUBAREA_2011
CHP_2007                         s9     CHP_2007                        s27    CHP_2007
CHP_2004                         s9     CHP_2004                        s27    CHP_2004
OA2011                           s9     OA2011                          s27    OA2011
OA2001                           s9     OA2001                          s27    OA2001
OA1991                           s9     OA1991                          s27    OA1991
DataZone2011                     s9     DataZone2011                    s27    DataZone2011
DataZone2001                     s9     DataZone2001                    s27    DataZone2001
IntZone2011                      s9     IntZone2011                     s27    IntZone2011
IntZone2001                      s9     IntZone2001                     s27    IntZone2001
LAU_Level1_2011                  s9     LAU_Level1_2011                 s27    LAU_Level1_2011
LAU_Level2_2011                  s9     LAU_Level2_2011                 s27    LAU_Level2_2011
NUTS_Level2_2013                 s4     NUTS_Level2_2013                s12    NUTS_Level2_2013
NUTS_Level3_2013                 s5     NUTS_Level3_2013                s15    NUTS_Level3_2013
Locality_2012                    s9     Locality_2012                   s27    Locality_2012
Settlement_2012                  s9     Settlement_2012                 s27    Settlement_2012
Civil_Parish_1930                s9     Civil_Parish_1930               s27    Civil_Parish_1930
Ent_Region_2008                  s9     Ent_Region_2008                 s27    Ent_Region_2008
Reg_Dist_2007                    s9     Reg_Dist_2007                   s27    Reg_Dist_2007
SDPA_2013                        s9     SDPA_2013                       s27    SDPA_2013
TTWA_2011                        s9     TTWA_2011                       s27    TTWA_2011
Gridlink_Ind                     s1     Gridlink_Ind                    s3     Gridlink_Ind

Codes:  num = numeric;  sn = string of length n

Error # 5127
Mismatched variable types on the input files.

Note # 5145
The working file has been restored, and subsequent commands may access the working file.
Exe.
```

The files are not matched successfully. As a short term solution, one or more of the syntax examples mentioned in this guidance may be used.

Appendix 1c – File saved in Unicode and subsequently used in Locale encoding

In this example files will be matched in Locale encoding.

```
Show unicode.
*Locale is on.

*Open file originally saved in Locale system. This file has the string lengths as required, e.g., pc7 is length 7.
Get file='H:\Stats Governance\Scottish_Postcode_Directory_2017_1_locale.sav'.
```

```
*Match to file "Scottish_Postcode_Directory_2017_1_unicode" which is originally saved in Unicode system.
*Because the file "Scottish_Postcode_Directory_2017_1_unicode" is now used in Locale setting, the length of all string variables in this file has not changed.

Match files file=*
      /table='U:\Scottish_Postcode_Directory_2017_1_unicode.sav'
      /by pc7.
Execute.

*Files are matched successfully.
```

Since the string length is not impacted, the files are matched successfully.

References

Tech Terms (https://techterms.com/definition/characterencoding, Accessed 17th July 2017)

How-To Geek (https://www.howtogeek.com/howto/45765/htg-explains-what-are-character-encodings-and-how-do-they-differ/, Accessed 17th July 2017)

Kunststube (http://kunststube.net/encoding/, Accessed 17th July 2017)

SPSS Tutorials (https://www.spss-tutorials.com/spss-unicode-mode/, Accessed 17th July 2017)