4. Using the Command Line

This section describes how to encode data using the command line front end program. The examples given are for the Linux platform, but the same options are available for Windows - just rememer to include the executable file extension. i.e.:

zint.exe -d "This Text"

4.1 Inputting data

The data to encode can be entered at the command line using the -d option, for example

zint -d "This Text"

This will encode the Text "This Text". Zint will use the default symbology, Code 128, and output to the default file out.png in the current directory. Alternatively, if libpng was not present when Zint was built, the default output file will be out.gif. The -d switch and the input data should always be the last entry on the command line input. Any options given after this will be ignored.

The data input to Zint is assumed to be encoded in Unicode (UTF-8) format. If you are encoding characters beyond the 7-bit ASCII set using a scheme other than Unicode then you will need to set the appropriate input options as shown in section 4.11 below.

Non-printing characters can be entered on the command line using the backslash (\) as an escape character. Permissible characters are shown in the table below. Note that this only applies on the command line.

Escape Character	ASCII Equivalent	Interpretation
\0	000	Null
\E	004	End of Transmission
\a	007	Bell
\b	008	Backspace
\t	009	Horizontal Tab
\n	00A	Line Feed
\v	00B	Vertical Tab
\f	00C	Form Feed
\r	00D	Carriage Return
\e	01B	Escape
\G	01D	Group Selector
\R	01E	Record Selector

Input data can be read directly from file using the -i switch as shown below. The input file is assumed to be Unicode (UTF-8) formatted unless an alternative mode is selected. This command replaces the use of the -d switch and should similarly be the last option given.

zint -i ./somefile.txt

4.2 Directing Output

Output can be directed to a file other than the default using the -o switch. For eample:

zint -o here.png -d "This Text"

This draws a Code 128 barcode in the file here.png. If an encapsulated Post Script file is needed simply append the file name with .eps, and so on for the other supported file types:

zint -o there.eps -d "This Text"

4.3 Selecting barcode type

Selecting which type of barcode you wish to produce (i.e. which symbology to use) can be done at the command line using the -b or --barcode= switch followed by the appropriate integer value in the following table. For example to create a Data Matrix symbol you could use:

zint -o datamatrix.png -b 71 -d "Data to encode"

Numeric Value	Barcode Name
1	Code 11
2	Standard Code 2 of 5
3	Interleaved 2 of 5
4	Code 2 of 5 IATA
6	Code 2 of 5 Data Logic
7	Code 2 of 5 Industrial
8	Code 3 of 9 (Code 39)
9	Etended Code 3 of 9 (Code 39+)
13	EAN (Including EAN-8 and EAN-13)
14	EAN + Check Digit
16	GS1-128 (UCC.EAN-128)
18	Codabar
20	Code 128 (automatic subset switching)
21	Deutshe Post Leitcode
22	Deutshe Post Identcode
23	Code 16K
24	Code 49
25	Code 93
28	Flattermarken
29	GS1 DataBar-14
30	GS1 DataBar Limited
31	GS1 DataBar Etended
32	Telepen Alpha

35	UPC A + Check Digit
37	UPC E
38	UPC E + Check Digit
40	PostNet
47	MSI Plessey
49	FIM
50	LOGMARS
51	Pharmacode One-Track
52	PZN
53	Pharmacode Two-Track
55	PDF417
56	PDF417 Truncated
57	Maicode
58	QR Code
60	Code 128 (Subset B)
63	Australia Post Standard Customer
66	Australia Post Reply Paid
67	Australia Post Routing
68	Australia Post Redirection
69	ISBN (EAN-13 with verification stage)
70	Royal Mail 4 State (RM4SCC)
71	Data Matrix (ECC200)
72	EAN-14
73	Vehicle Identification Number (America)
74	Codablock-F
75	NVE-18
76	Japanese Postal Code
77	Korea Post
79	GS1 DataBar-14 Stacked
80	GS1 DataBar-14 Stacked Omnidirectional
81	GS1 DataBar Epanded Stacked
82	PLANET
84	MicroPDF417
85	USPS OneCode
86	Plessey Code
87	Telepen Numeric
89	ITF-14
90	Dutch Post KIX Code

92	Aztec Code
93	DAFT Code
97	Micro QR Code
98	HIBC Code 128
99	HIBC Code 39
102	HIBC Data Matrix (ECC200)
104	HIBC QR Code
106	HIBC PDF417
108	HIBC MicroPDF417
112	HIBC Aztec Code
115	DotCode
116	Han Xin (Chinese Sensible) Code
121	Royal Mail 4-State Mailmark
128	Aztec Runes
129	Code 32
130	Composite Symbol with EAN linear component
131	Composite Symbol with GS1-128 linear component
132	Composite Symbol with GS1 DataBar-14 linear component
133	Composite Symbol with GS1 DataBar Limited component
134	Composite Symbol with GS1 DataBar Etended component
135	Composite Symbol with UPC A linear component
136	Composite Symbol with UPC E linear component
137	Composite Symbol with GS1 DataBar-14 Stacked component
138	Composite Symbol with GS1 DataBar-14 Stacked Omnidirectional component
139	Composite Symbol with GS1 DataBar Epanded Stacked component
140	Channel Code
141	Code One
142	Grid Matrix
143	UPNQR - Univerzalni Plačilni Nalog QR

This table is also accessible from the command line by issuing zint –t

4.4 Adjusting height

The height of a linear symbol can be adjusted using the --

zint --height=100 -d "This Text"

This specifies a symbol height of 100 times the x-dimension of the symbol.

4.5 Adjusting whitespace

The amount of whitespace to the left and right of the generated barcode can be altered using the –w switch. For eample:

zint -w 10 -d "This Text"

This specifies a whitespace width of 10 times the x-dimension of the symbol.

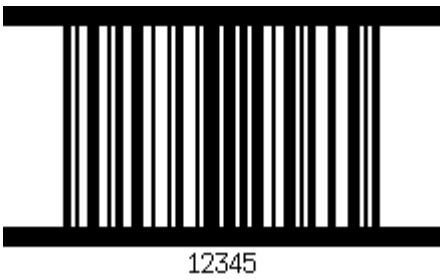
4.6 Adding boundary bars and boxes

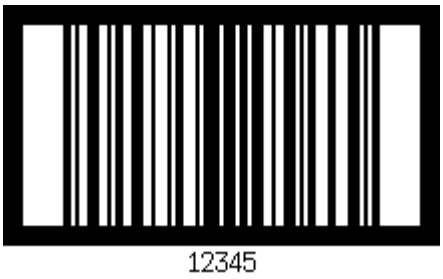
Zint allows the symbol to be bound with 'boundary bars' using the option --bind. These bars help to prevent misreading of the symbol by corrupting a scan if the scanning beam strays off the top or bottom of the symbol. Zint can also put a border right around the symbol and its whitespace with the --box option. This option is automatically selected for ITF-14 symbols.

The width of the boundary or box can be specified using the --border switch. For example:

zint --box --border=10 -d "This"

gives a box with a width 10 times the -resolution of the symbol.





4.7 Using colour

The default colours of a symbol are a black symbol on a white background. Zint allows you to change this. The -r switch allows the default colours to be inverted so that a white symbol is shown on a black background. For eample the command

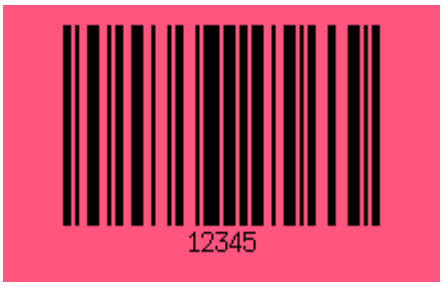
zint -r -d 'This'

gives an inverted Code 128 symbol. This is not practical for most symbologies but white-on-black is allowed by the Data Matrix ECC200 and Aztec Code symbology specifications.

For more specific needs the foreground (ink) and background (paper) colours can be specified using the --fg= and --bg= options followed by a number in RRGGBB headecimal notation (the same system used in HTML). For eample the command

zint --fg=004700 -d "This"

alters the symbol to a dark green as shown below.



4.8 Rotating the Symbol

The symbol can be rotated through four orientations using the --rotate= option followed by the angle of rotation as shown below. This option is only available with raster image (PNG, BMP, GIF and PC) output.

--rotate=0 (default)	--rotate=180
--rotate=270	--rotate=90

4.9 Adjusting image size

The scale of the image can be altered using the --scale= option followed by a multiple of the default x-dimension. For eample for PNG images a scale of 5 will increase the x-dimension to 10 pixels.

4.10 Input modes

By default all input data is assumed to be encoded in Unicode (UTF-8) format. Many barcode symbologies encode data using Latin-1 (ISO-8851-1) character encoding, so input is converted from Unicode to Latin-1 before being put in the symbol. In addition QR Code, Micro QR Code, Han in Code

characters which are also converted from Unicode. If Zint encounters characters which can not be encoded using the default character encoding then it will take advantage of the ECI (Etended Channel Interpretations) mechanism to encode the data. Be aware that not all barcode readers support ECI mode, so this can sometimes lead to unreadable barcodes. If you are using characters beyond those supported by Latin-1 then you should check that the resulting barcode can be understood by your target barcode reader. Zint will generate a warning message when ECI codes have been inserted into a symbol.

GS1 data can be encoded in a number of symbologies. Application identifiers should be enclosed in [square brackets] followed by the data to be encoded (see 5.1.12.3). To encode GS1 data use the --gs1 option. GS1 mode is assumed (and doesn't need to be set) for EAN-128, DataBar and Composite symbologies but is also available for Code 16k, Data Matrix, Aztec Code, DotCode and QR Code.

HIBC data may also be encoded in the symbologies Code 39, Code128, Codablock-F, Datamatrix, QR-Code, PDF417 and Aztec-Code. Within this mode, the leading '+' and the check character is automatically added.

The --binary option prevents Zint from performing any convertion of the data before placing in the barcode symbol and should be used if you are encoding raw binary or encrypted data.

If you are using data from file which is not UTF-8 formatted then you can specify the enconding using the --eci= switch followed by the appropriate number from the table below. This procedure adds an ECI flag in the barcode data which

ECI Code	Character Encoding Scheme
3	ISO-8859-1 - Latin alphabet No. 1 (default)
4	ISO-8859-2 - Latin alphabet No. 2
5	ISO-8859-3 - Latin alphabet No. 3
6	ISO-8859-4 - Latin alphabet No. 4
7	ISO-8859-5 - Latin/Cyrillic alphabet
8	ISO-8859-6 - Latin/Arabic alphabet
9	ISO-8859-7 - Latin/Greek alphabet
10	ISO-8859-8 - Latin/Hebrew alphabet
11	ISO-8859-9 - Latin alphabet No. 5
12	ISO-8859-10 - Latin alphabet No. 6
13	ISO-8859-11 - Latin/Thai alphabet
15	ISO-8859-13 - Latin alphabet No. 7
16	ISO-8859-14 - Latin alphabet No. 8 (Celtic)
17	ISO-8859-15 - Latin alphabet No. 9
18	ISO-8859-16 - Latin alphabet No. 10
20	Shift-Jis (JISX 0208 and JISX 0201) ❖
21	Windows-1250 - Latin 2 (Central Europe)
22	Windows-1251 - Cyrillic
23	Windows-1252 – Latin 1
24	Windows-1256 - Arabic
25	UCS-2 Unicode (High Order Byte First) ❖
26	Unicode (UTF-8)
27	ISO-646:1991 7bit Charset
28	Big-5 (Taiwan) Chinese Charset ❖
29	GB(PRC) Chinese Charset ❖
30	Korean Charset (KSX1001:1998) ❖

❖ Note: when using the ECI flag Zint will treat all input data as raw binary, this means that data which is encoded using a multible byte encoding scheme (other than UTF-8) will not use optimal compression. It is therefore recomended that data using these shcemes be converted to UTF-8 using iconv or similar before passing it to Zint.

4.11 Batch processing

Data can be batch processed by reading from a Text file and producing a separate barcode image for each line of Text in that file. To do this use the --batch switch. To select the input file from which to read data use the –i option. Zint will automatically detect the end of a line of Text (in either Uni or Windows formatted Text files) and produce a symbol each time it finds this. Input files should end with a return character – if this is not present then Zint will not encode the last line of Text, and will warn you that there is a problem.

By default Zint will output numbered filenames starting with 00001.png, 00002.png etc. To change this behaviour use the –o option in combination with batch using special characters in the output file name as shown in the table below:

Input Character	Interpretation
~	Insert a number or ‘0’
#	Insert a number or space
@	Insert a number or “*”
Any other	Insert literally

The following table shows some examples to clarify this method:

Input	Finenames Generated
-o file~~~.svg	file001.svg, file002.svg, file003.svg
-o @@@@bar.png	***1.png, ***2.png, ***3.png
-o my~~~bar.eps	my001.bar.eps, my002.bar.eps, my003bar.eps
-o t@es~t~.png	t*es0t1.png, t*es0t2.png, t*es0t3.png

4.12 Direct output

The finished image files can be output directly to stdout for use as part of a pipe by using the --direct option. By default

--direct will output data as a PNG image, but this can be altered by supplimenting the --direct option with a --filetype= option followed by the suffix of the file type required. For example:

zint -b 84 --direct --filetype=pcx -d "Data to encode"

This command will output the symbol as a PCX file to stdout. The currently supported output file formats are shown in the following table:

Abbreviation	File format
BMP	Windows Bitmap
EMF	Enhanced Metafile
EPS	Encapsulated PostScript
GIF	Graphics Interchange Format
PCX	ZSoft Paintbrush image
PNG	Portable Network Graphic
SVG	Scalable Vector Graphic
TIF	Tagged Image File Format
TXT	Text file (see 4.16)

CAUTION: Outputting binary files to the command shell without catching that data in a pipe can have unpredictable results. Use with care!

4.13 Automatic filenames

The --mirror option instructs Zint to use the data to be encoded as an indicator of the filename to be used. This is particularly useful if you are processing batch data. For example the input data "1234567" will result in a file named 1234567.png.

There are restrictions, however, on what characters can be stored in a file name, so the file name may vary from the

data if the data includes non-printable characters, for example, and may be shortened if the data input is long.

To set the output file format use the --filetype= option as detailed in section 4.12.

4.14 Working with Dots

Matrix codes can be rendered as a series of dots or circles rather than the normal squares by using the --dotty option. This option is only available for matrix symbologies, and is automatically selected for DotCode. The size of the dots can be adjusted using the --dotsize= option followed by the radius of the dot, where that radius is given as a multiple of the x-dimension.

4.15 Help options

There are three help options which give information about how to use the command line. The -h or --help option will display a list of all of the valid options available, and also gives the exact version of the software.

The -t or --types option gives the table of symbologies along with the symbol ID numbers.

The -e or --ecinos option gives a list of the ECI codes supported by Zint.

4.16 Other output options

For linear barcodes the Text present in the output image can be removed by using the --noText option.

The text can be set to bold using the --bold option, or a

The --bold and --small options can be used together if required.

Zint can output a representation of the symbol data as a set of hexadecimal values if asked to output to a text file (*.txt) or if given the option --filetype=txt. This can be used for test and diagnostic purposes.

The --cmyk option is specific to output in encapsulated PostScript, and converts the RGB colours used to the CMYK colour space. Setting custom colours at the command line will still need to be done in RRGGBB format.

Additional options are available which are specific to certain symbologies. These may, for eample, control the amount of error correction data or the size of the symbol. These options are discussed in section 6 of this guide.