In byte mode, the data string consists of the mode indicator, the character count indicator, and then the raw bytes from the input text.

**Convert to ISO 8859-1 or UTF-8**

The default character set for byte mode is ISO 8859-1, and when possible, you should convert your input text to this character set. The QR code specification discusses ECI mode, which allows you to specify a different character set from ISO 8859-1, but some QR code readers do not understand ECI escape sequences.

If there are characters in the input string that can't be encoded in ISO 8859-1, you may be able to encode it in UTF-8 instead, because some QR code readers are able to detect and correctly display UTF-8 encoding in byte mode without requiring any ECI escape sequences.

To combat this issue, you may wish to test different QR code readers to find out how they deal with non-ISO 8859-1 characters in byte mode, or ask your users to provide feedback about the QR code readers that they use.

**Split the String into 8-bit Bytes**

After converting your input string into ISO 8859-1, or UTF-8 if your users have QR code readers that can recognize it in byte mode, you must split the string into 8-bit bytes.

For example, we will use the input string "Hello, world!" to create a version 1 QR code. Since it contains lowercase letters, a comma, and an exclamation mark, it can't be encoded with alphanumeric mode, which does not include lowercase letters, commas, or exclamation marks.

Example string converted into hexadecimal bytes:

H → 0x48

e → 0x65

l → 0x6c

l → 0x6c

o → 0x6f

, → 0x2c

→ 0x20

w → 0x77

o → 0x6f

r → 0x72

l → 0x6c

d → 0x64

! → 0x21

**Convert Each Byte into Binary**

Convert the byte into an 8-bit binary string. Pad on the left with 0s if necessary to make each one 8-bits long.

Example string with each byte converted into 8-bit binary string:

H → 0x48 → 01001000

e → 0x65 → 01100101

l → 0x6c → 01101100

l → 0x6c → 01101100

o → 0x6f → 01101111

, → 0x2c → 00101100

→ 0x20 → 00100000

w → 0x77 → 01110111

o → 0x6f → 01101111

r → 0x72 → 01110010

l → 0x6c → 01101100

d → 0x64 → 01100100

! → 0x21 → 00100001