# The mysteries of character encoding

So, I had to explain, several times, character encodings to a friend of mine. Yeah, you know who you are! This can mean one of two things:

1. My friend simply can't/won't understand character encoding (there's something wrong with him?)
2. I suck at explaining stuff to others (in person)

While several people would agree that I just can't explain stuff, and should avoid teaching of any kind at all cost, I'll show them what I think about that by writing this explanation here as simple as possible.

## Let's start with the obvious (to me)

The name "ANSI" is a misnomer, since it doesn't correspond to any actual ANSI standard. Actually, ANSI is not just a "slight" misnomer, it is a completely wrong name. This name clearly implies that whatever it refers to is an ANSI standard, which it is not. With that said, it's so widely used (and Microsoft accepted it) that we're stuck with it.

ANSI encoding is a quasi generic term used to refer to the standard code page on a system, usually Windows. It is more properly referred to as Windows-1252 (at least on Western/U.S. systems, it can represent certain other Windows code pages on other systems). This is essentially an extension of the ASCII character set in that it includes all the ASCII characters with an additional 128 character codes. This difference is due to the fact that "ANSI" encoding is 8-bit rather than 7-bit as ASCII is (ASCII is almost always encoded nowadays as 8-bit bytes with the MSB set to 0).

ANSI does not necessarily have to map to CP1252. It does, however, always refer to the legacy code-page set for the system. This may be CP1252 on western European or US systems but don't count on that.

## So what are the differences?

- "Unicode encoding" is more properly known as UTF-16: 2 bytes per "code point". This is the native format of strings in .NET. Values outside the Basic Multilingual Plane (BMP) are encoded as surrogate pairs.
- UTF-8: Variable length encoding, 1-4 bytes per code point. ASCII values are encoded as ASCII using 1 byte.
- UTF-7: Usually used for mail encoding. If you're not doing mail and you think you need it, you're

most likely wrong.
- UTF-32: Fixed width encoding using 4 bytes per code point. This isn't very efficient, but makes life easier outside the BMP.
- ASCII: Single byte encoding only using the bottom 7 bits. (Unicode code points 0-127.) No accents etc.
- ANSI: There's no one fixed ANSI encoding – there are lots of them. Usually when people say "ANSI" they mean "the default locale/code-page for my system" which is obtained via Encoding. Default, and is often Windows-1252 but can be other locales.

## To clarify

- **An encoding** is the set of rules with which to convert something from one representation to another.
- **Character set, charset** is a set of characters that can be encoded. "The ASCII encoding encompasses a character set of 128 characters." Essentially synonymous to "encoding".
- **Code page** is a "page" of codes that map a character to a number or bit sequence. A.k.a. "the table". Essentially synonymous to "encoding".

## Resources

- The Unicode Consortium
- Unicode Character Table
- More detailed explanation ← Read this one, it is very detailed and informative