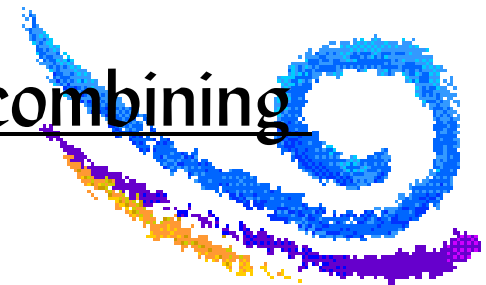


Character Encoding

Introduction to EBCDIC, ASCII, and Unicode

Characters

- ▶ We have learned over the years that to address character encoding, we need to be precise
- ▶ A "character" is an abstract entity with properties
 - A name, such as "LATIN SMALL LETTER A"
 - A glyph, or visual representation, such as "a"
 - Other possible properties such as direction, combining attributes, etc. - too arcane to discuss here



Character Encoding

- ▶ In a computer, we need to assign bit patterns (strings of one's and zero's) to each character we want to use
 - So when the user presses a keyboard, "a", that keypress gets converted into a bit pattern in computer memory
- ▶ Historically, we organized characters in chunks of 8-bit patterns: bytes
 - Each character is to be represented by one byte of memory - 8 bits; 8 one's and zero's



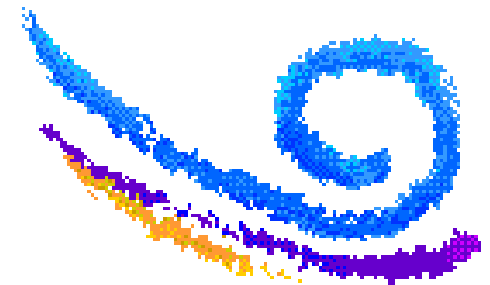
EBCDIC and ASCII

- ▶ EBCDIC is the classic character encoding scheme for mainframe and AS/400 machines
 - Each character is represented in an 8-bit byte
- ▶ ASCII is the classic character encoding scheme for most other hardware in use today
 - Each character is represented in 7 bits of an 8-bit byte
 - Various 8-bit extensions of the 7-bit code have been established as standards
- ▶ "The thing I like about standards is there are so many to choose from"
- source unknown



8-bit Limitations

- ▶ With 8-bits, there are 256 possible patterns of one's and zero's
 - Plenty of room for the Latin character set, even allowing some characters to be used for controlling hardware devices
 - But if you want to allow Arabic, Cyrillic, Greek, Devanagari, Hebrew, and so on, things get crowded



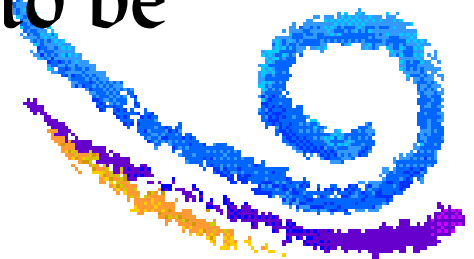
Codepages

- ▶ One of the earliest approaches to dealing with many characters was to establish codepages
 - Each codepage could handle a specific set of characters
 - So pressing a key on a keyboard creates a bit pattern in memory
 - And whether that pattern represents "LATIN SMALL LETTER A" or, say, "MYANMAR LETTER KA" depends on the codepage currently being used



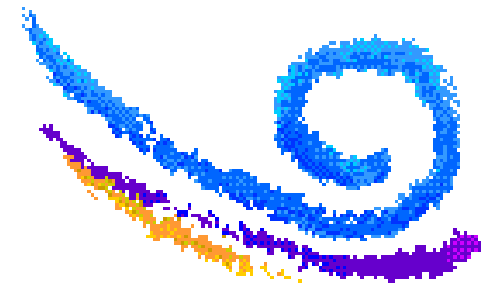
Unicode

- ▶ The problem with codepages is: you can only have one codepage active at a time
- ▶ So if you want to have, say, a web page with characters from the Latin and Cyrillic alphabets, you're out of luck using classic character encoding schemes
- ▶ We need an alternative - and there are several available today, but the most viable seems to be Unicode



Unicode, 2

- ▶ Unicode has become the standard encoding for newer technologies
 - HTML 4.1, XML, Web Services, Java
- ▶ Unicode is supported on IBM mainframes in both hardware and software
 - For techie details, visit
[//http://www.trainersfriend.com/Papers/uncdtalk.pdf](http://www.trainersfriend.com/Papers/uncdtalk.pdf)





6790 East Cedar Avenue, Suite 201
Denver, Colorado 80224
USA

<http://www.trainersfriend.com>
303.393.8716

Sales: kitty@trainersfriend.com
Technical: steve@trainersfriend.com