Character Encoding

Creating a unified code for characters is a significant achievement in technical communication. The *American Standards Institute* initially standardized a 7-bit code, known as **ASCII** (American Standard Code for Information Interchange). This code includes the 26 uppercase and lowercase letters of the English alphabet, digits, punctuation marks, and so-called **control characters**. Control characters were used to manage text formatting and control specific applications.

ASCII: The First Standard

ASCII was designed for English and includes 128 characters, where:

- The first 32 (0-31) are control characters (e.g., carriage return, line feed).
- The characters from 32 to 127 represent printable symbols (letters, digits, punctuation).

However, this 7-bit code couldn't handle characters from languages with accents, mathematical symbols, or special graphical symbols. As a result, a need arose to extend the character set.

ANSI and Extended Encoding

To address the need for more characters, an 8-bit extension was developed and named **ANSI** (American National Standards Institute) encoding. This provided 256 characters and allowed for additional language-specific characters. One widely used extension, particularly for text editing, is the **IBM Code Page 852**, which includes the 18 accented characters used in the Hungarian language.

ISO Standards

To further standardize character sets globally, the **ISO 8859** series was introduced:

- **ISO-8859-1 (Latin-1):** Encodes Western European accented characters.
- **ISO-8859-2 (Latin-2):** Used for Central and Eastern European languages, including Hungarian, and contains special characters like **ű** and **ő**.

Unicode and UCS: Universal Character Sets

As digital communication expanded, a more comprehensive character set was required to accommodate all languages and symbols. The **ISO-10646** standard defines a **Universal Character Set (UCS)**, ensuring that no information is lost when converting any character to UCS and back. UCS includes characters from all known languages, both living and extinct, as well as known mathematical and scientific symbols.

- UCS was first published in 1993 as **ISO-10646-1** and originally used 31-bit encoding.
- The range between **0x0000** and **0xFFFD** represents the 16-bit domain called the **Basic Multilingual Plane (BMP)**.

- In 2001, **ISO-10646-2** was introduced to cover characters outside the BMP, and by 2003, both sets were unified under the ISO-10646 standard.

Unicode

The **Unicode** standard is widely used alongside UCS. Each Unicode character is identified by both a number and a name. For example, **U+0041** stands for “Latin capital letter A.” Unicode's range:

- From **U+0000** to **U+007F** is equivalent to 7-bit ASCII.
- From **U+0080** to **U+00FF** corresponds to Latin-1.

Differences Between ISO 10646 and Unicode

While ISO 10646 defines the code table of characters, Unicode adds further functionality by providing:

- **Typography standards** for different writing systems (like Arabic and Hebrew).
- **Text rendering rules** that handle multiple writing directions in a document.
- **Collation and text comparison algorithms** that allow for sorting and comparison of characters in different languages.

Practical Example

In practice, if you're encoding a text that contains the characters **A**, **ű**, and **Ω**:

- In ASCII, only **A** would be represented (U+0041).
- In ISO-8859-2 (Latin-2), both **A** and **ű** (U+0171) would be encoded.
- In Unicode, all three characters, **A** (U+0041), **ű** (U+0171), and **Ω** (U+03A9), would be included.

Examples of Encoding Systems

- **ASCII**: Limited to 128 characters (0-127), primarily for English.
- **ANSI (Extended ASCII)**: 256 characters, includes language-specific letters.
- **ISO-8859-1 (Latin-1)**: Encodes characters for Western European languages.
- **ISO-8859-2 (Latin-2)**: Supports Eastern European languages, including Hungarian.
- **Unicode (UTF-8)**: Supports virtually all characters from all writing systems worldwide.

With Unicode's rise, character encoding has become highly flexible, supporting diverse languages, symbols, and historical texts. This makes Unicode the most widely adopted standard for modern software systems.

—

This extended version adds more details and examples to make the concept of character encoding clearer for your students, especially in real-world applications like handling multilingual text.