Final Project

Unicode: Linguistic Diversity in Digital Writing Spaces

Marwa Kotb

ETEC540 (65B)

Text Technologies: The Changing Spaces of Reading and Writing

Introduction

The representation of the world's writing systems has been a significant challenge in the technology world. Until the early 1990s, the English alphabet was the only standard, and only a few non-English alphabets and diacritics could be used or viewed within one's computing system (Crystal, 2011). If there was a file of foreign words with an odd sequence of codes that weren't recognized in the computing system (for example, if you opened a Japanese document from an American computer), the text would be completely disregarded; the user would see weird blanks, question marks, or boxes instead. This is still possible, but things have progressed significantly since the birth of Unicode (Crystal, 2011).

What is Unicode

Unicode is a world-wide character encoding standard that provides "a unique number for every character, no matter what platform, device, application, or language" (Unicode, 2017, July 24, para.7). It is "developed [and maintained] by the Unicode Consortium, a group of companies and institutions with interests in international text-encoding and computing applications" (Bigelow & Holmes, 1993, p.289).

In computer systems, characters (i.e., any significant writing unit) are stored in binary numbers. For characters representation and display, the system uses a code chart which tells, for example, the code number 68 in decimal (i.e., 0X44 in base 16 hexadecimal; U+0044 in the Unicode) represents 'D' (see figure 1). Having a unique pattern for each character have "eliminate[d] the problem of having to keep track of which of many different characters this specific instance of a particular bit pattern is supposed to represent." (Gillam, 2003, "Chapter 1. Language, Computers, and Unicode", para.23).

Latin Alphabet: Uppercase	U+0041	A	65	0101	Latin Capital letter A	0034
	U+0042	B	66	0102	Latin Capital letter B	0035
	U+0043	C	67	0103	Latin Capital letter C	0036
	U+0044	D	68	0104	Latin Capital letter D	0037
	U+0045	E	69	0105	Latin Capital letter E	0038
	U+0046	F	70	0106	Latin Capital letter F	0039
	U+0047	G	71	0107	Latin Capital letter G	0040
	U+0048	H	72	0110	Latin Capital letter H	0041
	U+0049	I	73	0111	Latin Capital letter I	0042
	U+004A	J	74	0112	Latin Capital letter J	0043
	U+004B	K	75	0113	Latin Capital letter K	0044
	U+004C	L	76	0114	Latin Capital letter L	0045
	U+004D	M	77	0115	Latin Capital letter M	0046
	U+004E	N	78	0116	Latin Capital letter N	0047
	U+004F	O	79	0117	Latin Capital letter O	0048
	U+0050	P	80	0120	Latin Capital letter P	0049
	U+0051	Q	81	0121	Latin Capital letter Q	0050
	U+0052	R	82	0122	Latin Capital letter R	0051
	U+0053	S	83	0123	Latin Capital letter S	0052
	U+0054	T	84	0124	Latin Capital letter T	0053
	U+0055	U	85	0125	Latin Capital letter U	0054
	U+0056	V	86	0126	Latin Capital letter V	0055
	U+0057	W	87	0127	Latin Capital letter W	0056
	U+0058	X	88	0130	Latin Capital letter X	0057
	U+0059	Y	89	0131	Latin Capital letter Y	0058
	U+005A	Z	90	0132	Latin Capital letter Z	0059

Figure 1: Web capture of Unicode Latin alphabet-upper-case (Wikipedia, 2021, March).

From telegraph to Unicode

Encoding systems had started long before the advent of the modern digital computer (Gillam, 2003). The beginning was with Samuel Morse code used to transmit messages via telegraph lines in 1837 (Gillam, 2003). The code set used two types of signals: A long “on” (i.e., dash) and a short “on” (i.e., dot) (see figure 2).

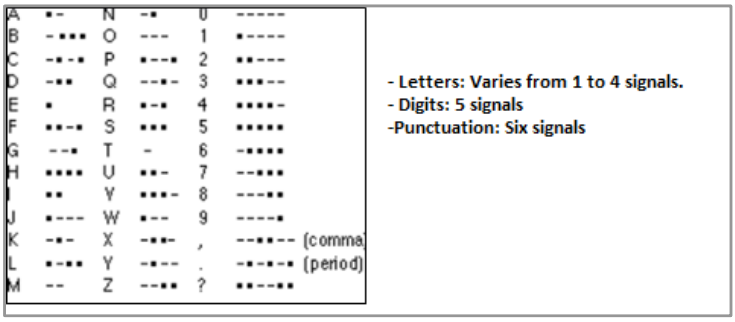

Figure 2: Subset of Morse code (Searle,2004, August 6).

The main problem of Morse code was the variant length of encoding that didn’t fit with the systematic mechanized process needed to place the characters on paper at the time (Gillam, 2003). A major step in this direction was Emile Baudot’s printing telegraph, invented in 1874; his code set was a combination of five keys (i.e., equal length), you may think of as a 5-bit code

(Gillam, 2003). This means that it can afford up 32 combinations (2^5) which were insufficient even to combine letters and digits. Thus, he created two separate sets of character encodings and assigned them different switches (i.e., control characters) to revert among them (see figure 3): The “LTRS” bank, included the upper-case letters, and a “FIGS” bank, included the ten digits and few punctuation marks and symbols (Gillam, 2003).

Binary value	Letters	Figures
00011	A	-
11001	B	?
01110	C	:
01001	D	$
00001	E	3
01101	F	!
11010	G	&
10100	H	STOP
00110	I	8
01011	J	'
01111	K	(
10010	L	)
11100	M	.
01100	N	,
11000	O	9
10110	P	0
10111	Q	1
01010	R	4
00101	S	BELL
10000	T	5
00111	U	7
11110	V	;
10011	W	2
11101	X	/
10101	Y	6
10001	Z	"
00000	n/a	n/a
01000	CR	CR
00010	LF	LF
00100	SP	SP
11111	LTRS	LTRS
11011	FIGS	FIGS

Figure 3: Baudet code set (Searle, 2004, August 6).

Until the first half of the twentieth century, many computerized and communication systems that follow relied prominently on Baudet sets (Searle, 2004, August 6). However, with the rapid development of communications and data processing technologies in the United States, there was a need for a standard character code for exchanging data and for handling the entire character set of the English language (Searle, 2004, August 6). In 1968, the American National Standards Association (ANSI) launched the well-known 7-bit American Standard Code for Information Interchange (ASCII) encoding (Searle, 2004, August 6). With its 32 control characters and 96 printing characters (see figure 4), the ASCII system was sufficient enough to fit all kinds of needs and characters in the English writing system (Searle, 2004, August 6).

Dec	Hex	Oct	Binary	Char	Dec	Hex	Oct	Binary	Char	Dec	Hex	Oct	Binary	Char	Dec	Hex	Oct	Binary	Char
0	00	000	0000000	NUL (null character)	32	20	040	0100000	space	64	40	100	1000000	@	96	60	140	1100000	`
1	01	001	0000001	SOH (start of header)	33	21	041	0100001	!	65	41	101	1000001	A	97	61	141	1100001	a
2	02	002	0000010	STX (start of text)	34	22	042	0100010	"	66	42	102	1000010	B	98	62	142	1100010	b
3	03	003	0000011	ETX (end of text)	35	23	043	0100011	#	67	43	103	1000011	C	99	63	143	1100011	c
4	04	004	0000100	EOT (end of transmission)	36	24	044	0100100	$	68	44	104	1000100	D	100	64	144	1100100	d
5	05	005	0000101	ENQ (enquiry)	37	25	045	0100101	%	69	45	105	1000101	E	101	65	145	1100101	e
6	06	006	0000110	ACK (acknowledge)	38	26	046	0100110	&	70	46	106	1000110	F	102	66	146	1100110	f
7	07	007	0000111	BEL (bell (ring))	39	27	047	0100111	'	71	47	107	1000111	G	103	67	147	1100111	g
8	08	010	0001000	BS (backspace)	40	28	050	0101000	(	72	48	110	1001000	H	104	68	150	1101000	h
9	09	011	0001001	HT (horizontal tab)	41	29	051	0101001	)	73	49	111	1001001	I	105	69	151	1101001	i
10	0A	012	0001010	LF (line feed)	42	2A	052	0101010	*	74	4A	112	1001010	J	106	6A	152	1101010	j
11	0B	013	0001011	VT (vertical tab)	43	2B	053	0101011	+	75	4B	113	1001011	K	107	6B	153	1101011	k
12	0C	014	0001100	FF (form feed)	44	2C	054	0101100	,	76	4C	114	1001100	L	108	6C	154	1101100	l
13	0D	015	0001101	CR (carriage return)	45	2D	055	0101101	-	77	4D	115	1001101	M	109	6D	155	1101101	m
14	0E	016	0001110	SO (shift out)	46	2E	056	0101110	.	78	4E	116	1001110	N	110	6E	156	1101110	n
15	0F	017	0001111	SI (shift in)	47	2F	057	0101111	/	79	4F	117	1001111	O	111	6F	157	1101111	o
16	10	020	0010000	DLE (data link escape)	48	30	060	0110000	0	80	50	120	1010000	P	112	70	160	1110000	p
17	11	021	0010001	DC1 (device control 1)	49	31	061	0110001	1	81	51	121	1010001	Q	113	71	161	1110001	q
18	12	022	0010010	DC2 (device control 2)	50	32	062	0110010	2	82	52	122	1010010	R	114	72	162	1110010	r
19	13	023	0010011	DC3 (device control 3)	51	33	063	0110011	3	83	53	123	1010011	S	115	73	163	1110011	s
20	14	024	0010100	DC4 (device control 4)	52	34	064	0110100	4	84	54	124	1010100	T	116	74	164	1110100	t
21	15	025	0010101	NAK (negative acknowledge)	53	35	065	0110101	5	85	55	125	1010101	U	117	75	165	1110101	u
22	16	026	0010110	SYN (synchronize)	54	36	066	0110110	6	86	56	126	1010110	V	118	76	166	1110110	v
23	17	027	0010111	ETB (end transmission block)	55	37	067	0110111	7	87	57	127	1010111	W	119	77	167	1110111	w
24	18	030	0011000	CAN (cancel)	56	38	070	0111000	8	88	58	130	1011000	X	120	78	170	1111000	x
25	19	031	0011001	EM (end of medium)	57	39	071	0111001	9	89	59	131	1011001	Y	121	79	171	1111001	y
26	1A	032	0011010	SUB (substitute)	58	3A	072	0111010	:	90	5A	132	1011010	Z	122	7A	172	1111010	z
27	1B	033	0011011	ESC (escape)	59	3B	073	0111011	;	91	5B	133	1011011	[	123	7B	173	1111011	{
28	1C	034	0011100	FS (file separator)	60	3C	074	0111100	<	92	5C	134	1011100	\	124	7C	174	1111100	|
29	1D	035	0011101	GS (group separator)	61	3D	075	0111101	=	93	5D	135	1011101	]	125	7D	175	1111101	}
30	1E	036	0011110	RS (record separator)	62	3E	076	0111110	>	94	5E	136	1011110	^	126	7E	176	1111110	~
31	1F	037	0011111	US (unit separator)	63	3F	077	0111111	?	95	5F	137	1011111	_	127	7F	177	1111111	DEL

Figure 4: 7-bit ASCII code (Asciitable.xyz, 2018).

By the early 1980s, the 8-bit extended ISO encodings were implemented for several countries across the world (Gillam, 2003). In such encoding systems, the positions 0-127 were reserved for the 7-bit ASCII and the positions 128-255 (i.e., extra bit) were used for the extended character set holding specific characters of each country. For computers to retrieve the correct character on the screen, they must make sure that they were looking into the right code page (John, 2013). For instance, the code number 134 (0X86 in base 16 hexadecimal) could mean the

Greek upper-case alpha with tonos (Ά) if retrieved from the code page 869 (ASCII codes, n.d.-a), or the Hebrew letter zayin (ז) if retrieved from the code page 862 (ASCII codes, n.d.-b).

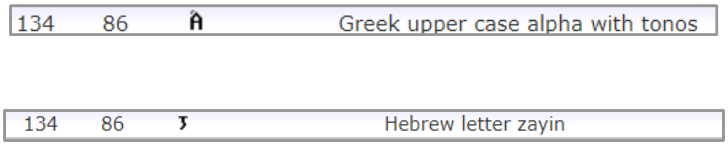

Figure 5: web captures from the Greek and Hebrew extended character sets (Ascii codes, n.d.-a; Ascii codes, n.d.-b).

The 8-bit representation did not allow the extended character sets of many languages to co-exist with each other (i.e., other than English and something else) without having auxiliary data structures to keep track of the encodings used for the different pieces of the text; using a code-switching scheme was cumbersome both for the user and for the processing software (Gillam, 2003). Furthermore, this encoding system was insufficient to hold the thousands of ideographic characters of East Asian scripts (Becker, 1988). Other systems such as the 14-bit JIS (used in Japan) and BIG-5 (used in China) were utilized instead (Searle, 2004, August 6).

In the late 1980's, Joseph Becker of Xerox PARC, along with Lee Collins and Mark Davis of Apple Computers, proposed the 16 bits standard international/ multilingual interchange encoding system unifying the world text encodings across different platforms (Unicode, 2015, March 26). The notion "Unicode" was given by Becker (1988) with the intent to suggest "unique, unified, and universal encoding" (p.3). The first version of the Unicode character set was released in June 1992. Starting with Unicode 2.0 released in July 1996, the standard was extended, adding 16 supplementary planes to the original 16-bit code space, which is now called the basic multilingual plane (BMP), thus, increasing the code space up to 1,114,112 code points (i.e., the number assigned to a character) (Piotrowski, 2012). This increase allowed the encoding of ancient scripts (such as Cuneiform, Gothic and Egyptian Hieroglyphs), besides historical

characters in Latin, Cyrillic, Greek, and other extant scripts into the digital form (Piotrowski, 2012). Ever since, the extension, Unicode scheme known as UTF (Uniform Transformation Format), with its three variants UTF-8, UTF-16, UTF-32, have become the most significant standard to encode text in the global digital world. On March 2020, Unicode released its latest version, 13.0.0, which supports 154 scripts and a total of 143,859 linguistic symbols, along with a range of other representations such as Braille patterns, mathematical symbols, musical notations, emoji characters, and more (Unicode, 2020, March 10).

Design philosophies

According to Liu and Lions (1998), there are three fundamental design philosophies of the Unicode standard. Firstly, the aim toward an ideal of universality. Unicode works to encode characters that occur in a script, regardless of the number of languages that may use it; a single-language script is as important as one used for hundreds of languages (Crystal, 2011). Secondly, uniqueness, Unicode is the first encoding system that utilizes the abstract concept (the semantic unit) of the character rather than encoding the glyphs variations (Liu & Lions, 1998). For instance, in Arabic rendering, a character such as (heh) has four different shapes depending on the text's context (see figure 6). However, the four glyph variants are represented by a single Unicode code value and it is up to text rendering process to determine the glyph per the text context (Gillam, 2003).

Figure 6: Web captures of the four glyphs of (heh) (Arabic Quick, n.d.).

Lastly, uniformity, Unicode works to develop a harmonization of typographic rendering; all the font characters should be displayed in a uniform width manner regardless of the encoding pattern (one-byte-alphabetic characters or four-byte-ideographic characters) (Liu & lions, 1998).

Who decides? Why does it matter?

Unicode Consortium is a non-profit organization; the membership is opened to companies, institutions, non-profit groups, and individuals. However, the membership levels vary, and each level comes with its voting right(s) and a yearly cost (Unicode, 2019a). The Unicode Consortium receive character proposals from members, users, and communities. The decision on these proposals is mainly made by the corporate ($21000) and institutional members (S10,000-$14000); both have the rights of the full vote in the technical committee (i.e., the corporate level has additional voting rights in the board and at full membership meetings) (Unicode, 2019a). The supporting members ($5000-$8750) votes count as half the full vote's weight, and all other members may voice their opinion, but they don't have any voting rights (Unicode, 2019a).

Since its rise, the "full" membership of the Unicode Consortium has been under the control of North American technology companies. The most recent list includes Apple, Google, Facebook, Netflix, Microsoft, SAP, and Salesforce; the only exception is Sultanate of Oman, Ministry of Awqaf, and Religious Affairs (Unicode, 2019b). The supremacy of tech-giants in Unicode has opened several allegations against the encoding decisions that they don't concur with the proclaimed universality philosophy. John (2013) wrote that Unicode's "focus has been on scripts used in business" (p.328); decisions are made for the reasons of economic gain rather than user or community gains. In the same vein, in the controversial topic of emoji standardization, Berard (2018) argued that classifying "emoji" as a distinct subset of text-based

characters for standardization has satisfied the corporate members' needs; it wasn't based on their value. He asked us to consider: Who is left out in this form of a model, if those who have the money to pay for membership get to make the decisions? (Berard, 2018)

The impact of Unicode

Tracing the history of encoding standards enables us to understand that they are fundamental blocks in shaping access and communication in digital spaces; in Berard's (2018) words, what is included and excluded in encoding systems impacts who gets to speak and how? Back in the early iterations of the Internet, English was the dominant language due to ASCII restraints (Crystal, 2011); its superiority on the web has been widely perceived as a threat to all other languages and a "killer language" to the weaker ones (Crystal, 2003; Coulmas, 2018; Taha, 2015). Several scholars contended the exclusive practice, such as Shapard (1993), who wrote about the Japanese elimination in the Internet: "Narrow vision, one-byte seven-bit ASCII biases, the assumptions about character coding that arise from them, inadequate international standards, and local solutions that disregard what international standards there are, and that pay no heed to the ramifications for others—all these are serious related problems that inhibit, rather than enhance, increased connectivity and communication" (p. 256).

At this time, users communicating online in languages other than English needed to adapt to the ASCII environments. The European characters, for instance, were deprived of their cedilla, umlaut, stroke, etc. If there was an online resource for the Turkish scholar "Aşıksoy Gülsüm", the attribution and citation had to change to "Asikoy Gulsum" instead (Coulmas, 2018). As we move to languages written in other alphabets like Arabic, Hebrew, Greek, and non-alphabetic scripts like Chinese, the impact of the missing representation was more visible and immense. For example, in response, trans-scripting into English alphabets (i.e., Romanization) has emerged in

Computer-mediated Communication (CMC) contexts (Coulmas, 2018; Taha, 2015). In Danet and Herring (2003, November) publications about Arabish (see figure 7), Greeklish, and other trans-scripting practices, the authors regarded these adaptations as echoing threats to ideologies and cultural heritages, as well as, negatively influencing the local writing systems and spelling conventions.

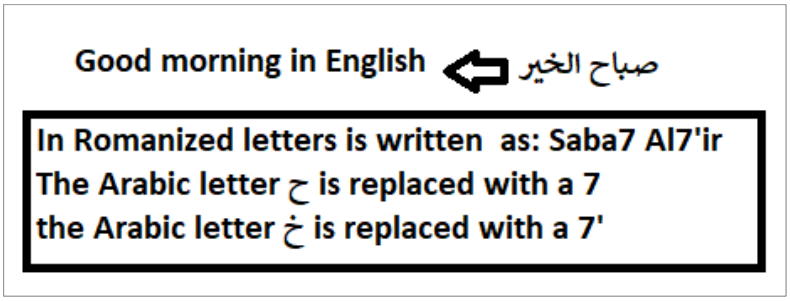

Figure 7: Arabish (Arabizi) phenomenon emerged in response to the lack of support of Arabic-scripting in the 1990's (Taha, 2015).

Fortunately, we moved beyond the limitations of the ASCII code in some respects. Now information can be exchanged in Arabic, Chinese, and many other languages you care to write. This change is inextricably linked to the increasing proliferation and wide-spread adoption of Unicode that have enabled new linguistic varieties (see figure 8). However, it is critical to highlight that not all encoding problems have yet been solved for all software programs, not all of the world's languages were effectively and equally represented, the growing bias in favor of English still exists in technological products, and Romanization practices have continued by those who felt that it was more suitable and facile for the modern technologies (Coulmas, 2018; Taha, 2015).

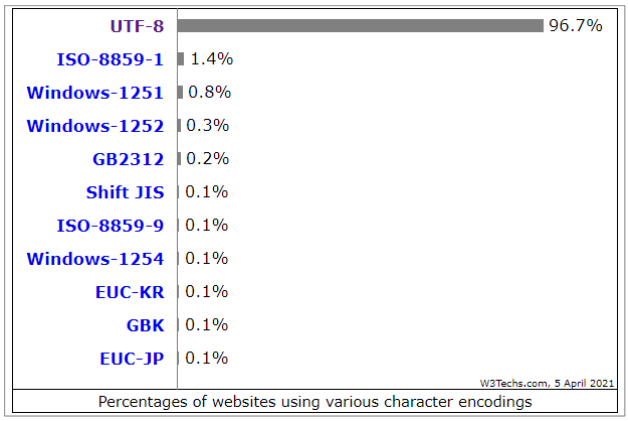

Figure 8: UTF 8 is used by 96.7% of websites known to W3tech (2020, April)

The benefits of Unicode exist in various educational fields of computer science, humanities, typography, and linguistics (Unicode, 2021, April). Several ongoing projects have been taken by a number of scholars and institutions for the revitalization and preservation of historical and minority languages in collaboration with Unicode, such as the Script Encoding Initiative (SEI) and the Missing Script Project (MSP) (Piotrowski, 2012). Even though many scripts are not yet included in Unicode and the task of harmonizing the scripts (i.e., to grant the uniformity property) is immense (SEI, n.d.; ATypI, 2018, September 25). Still, the progress brought by Unicode and linguists has allowed preserving some of the cultural and historical heritage and making it more accessible, both to students and scholars, by molding them into digital text, which can be searched and otherwise automatically processed (Piotrowski, 2012); this has opened up new possibilities for online education, study, and publication. Overall, Unicode encourages native-language education and universal literacy, helps to overcome linguistic barriers to participation, and enables embracing one's cultural and linguistic identity throughout the learning process (SEI, n.d.).

Looking forward

Unicode standard is a technology that “brought deep-reaching changes in the republic of characters representation” (Coulmas, 2018, p.204). A sociolinguistic system, you may say, that circulates in online resources, software solutions, mobile applications, in symbolic systems such as emojis, and in educational practices. However, as noted, the Unicode standard has certain politics and uses embedded in its design—what might be called “biases” (John, 2013). And thus, computational literacy needs to go beyond its traditional view (i.e., coding literacy) and extend to encompass the political, cultural, and social apparatuses of the technology; critical literacy and critical pedagogy are essential. It is crucial to promote student’s critical thinking: How are character codes issued and by whom? Why do companies invest in full membership? How ideologies, cultural, and linguistic practices are impacted due to choices made about what characters or visual material included in Unicode? (Berard, 2018). Also, there is a need to draw students’ attention to the marginalized and the rights of those who have less access to resources due to the encoding constraints, and ensure bringing into computer science education the alternatives to the dominant technology and make room for analyses of competing, though less successful technologies, such as the TRON encoding system, as well as, the emerging encoding systems such as Noto, launched by Google to “sideline” Unicode and further the company’s dominance on global online communications (John, 2013; Coulmas, 2018).

References

Arabic Quick. (n.d.). Learn the Arabic letter Ha. Retrieved from http://arabicquick.com/learn-the-arabic-letter-ha/

ASCII codes. (n.d.-a). Code page 869 (Greek language). Retrieved from https://www.ascii-codes.com/cp869.html

ASCII codes. (n.d.-b). Code page 862 (Hebrew language). Retrieved from https://www.ascii-codes.com/cp862.html

AsciiTable.xyz. (2018). ASCII Table Online. Retrieved from https://www.asciitable.xyz/

ATypI. (2018, September 25). *Johannes Bergerhausen, Morgane Pierson - The missing scripts project* [Video post]. Retrieved from https://www.youtube.com/watch?v=CHh2Ww_bdyQ

Becker, J. (1988). Unicode 88. *Unicode*. Retrieved from https://unicode.org/history/unicode88.pdf

Berard, B. (2018). I second that emoji: The standards, structures, and social production of emoji. *First Monday, 23*(9). https://doi.org/10.5210/fm.v23i9.9381

Bigelow, C., & Holmes, K. (1993). The design of a Unicode font. *Electronic Publishing, 6*(3), 289-305. Retrieved from https://ezproxy.library.ubc.ca/login?url=https://www-proquest-com.ezproxy.library.ubc.ca/scholarly-journals/design-unicode-font/docview/57324723/se-2?accountid=14656

Coulmas, F., & ProQuest (Firm). (2018). *An introduction to multilingualism: Language in a changing world* (First ed.). Oxford University Press.

Crystal, D., CRKN MiL Collection, & Cambridge Core EBA eBooks Complete Collection. (2003). *English as a global language* (2nd ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511486999

Crystal, D., Taylor & Francis eBooks A-Z, & Ebook Central. (2011). *Internet linguistics: A student guide*. Routledge. https://doi.org/10.4324/9780203830901

Danet, B., & Herring, S. (2003, November). Introduction: The multilingual Internet. *Journal of Computer-Mediated Communication, 9*(1). https://doi.org/10.1111/j.1083-6101.2003.tb00354.x

Gillam, R., & O'Reilly for Higher Education. (2003). *Unicode demystified: A practical programmer's guide to the encoding standard*. Addison-Wesley.

John, N. A. (2013). The construction of the multilingual internet: Unicode, Hebrew, and globalization. *Journal of Computer-Mediated Communication, 18*(3), 321-338. https://doi.org/10.1111/jcc4.12015

Piotrowski, M., Synthesis Collection Four, & ebrary, I. (2012). *Natural language processing for historical texts.* Morgan & Claypool. https://doi.org/10.2200/S00436ED1V01Y201207HLT017

Searle, S. (2004, August 6). A brief history of character codes in North America, Europe, and East Asia. *Tron Web*. Retrieved from http://tronweb.super-nova.co.jp/characcodehist.html

SEI. (n.d.). Welcome to the script encoding initiative. *Berkeley University of California*. Retrieved from https://linguistics.berkeley.edu/sei/

Shapard, J. (1993). Islands in the (data)stream: Language, character codes, and electronic isolation in Japan. In L. M. Harasim (Ed.), *Global networks: Computers and international communication* (pp.255–270). Cambridge, Mass.: MIT Press.

Taha. M. (2015). Arabizi: Is code-switching a threat to the Arabic language. *The Asian* Conference on Arts & Humanities. Retrieved from http://25qt511nswfi49iayd31ch80-wpengine.netdna-ssl.com/wp-content/uploads/papers/acah2015/ACAH2015_13058.pdf

Unicode (2015, March, 26). Early years of Unicode: The Unicode® standard"…begin at 0 and add the next character". Retrieved from https://unicode.org/history/earlyyears.html

Unicode (2017, July, 24). What is Unicode? Retrieved from https://unicode.org/standard/WhatIsUnicode.html#:~:text=The%20Unicode%20Standard%20provides%20a,devices%20and%20applications%20without%20corruption.

Unicode. (2019a). Membership levels. Retrieved from https://home.unicode.org/membership/membership-levels/

Unicode. (2019b). Members. Retrieved from https://home.unicode.org/membership/members/

Unicode. (2020, March 10). Unicode® 13.0.0. Retrieved from https://unicode.org/versions/Unicode13.0.0/

Unicode. (2021, April). Students and educators. Retrieved from https://www.unicode.org/education/students.html

Wikipedia. (2021, March). List of Unicode characters. Retrieved from https://en.wikipedia.org/wiki/List_of_Unicode_characters

W3tech (2021, April). Usage statistics of character encodings for websites. Retrieved from https://w3techs.com/technologies/overview/character_encoding