Binary, Encodings, and Formats

Binary Representation

- Our normal number system is “decimal” or base 10

- Binary is base 2
    - Octal, base 8
    - Hexadecimal, base 16

- Can convert our number system to/from any other base
    - Can convert any base to any other base

- 1 bit is a single 0 or 1; 1 byte is 8 bits

Convert Decimal to Binary

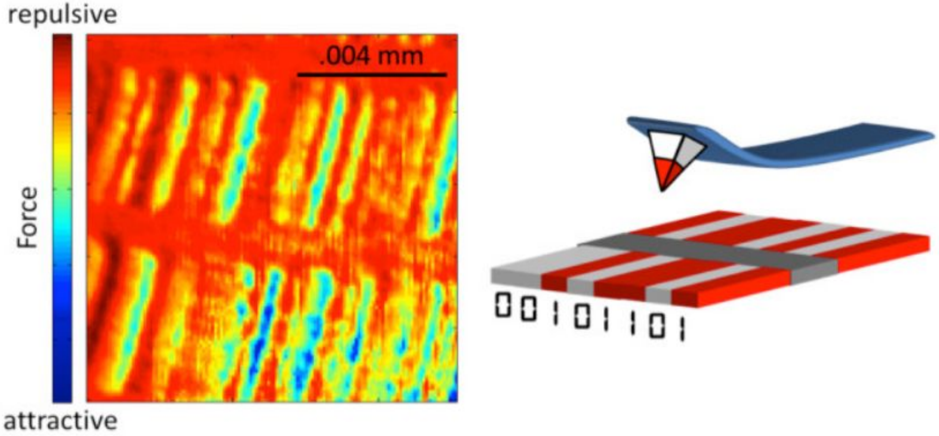
- Divide by new base and take the remainder:
    - 56_{10} => 111000_2
    - 56 / 2 = 28 (0/2) => 0
      28 / 2 = 14 (0/2) => 00
      14 / 2 = 7 (0/2) => 000
      7 / 2 = 3 (1/2) => 1000
      3 / 2 = 1 (1/2) => 11000
      1 / 2 = 0 (1/2) => 111000

Convert Binary to Decimal

- Raise current base to power of current digit multiplied by digit value and sum result:
  - 111000_2 => 56_{10}
  - $$\begin{array}{cccccc} 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ | & | & | & | & | & | \\ 32 & 16 & 8 & 4 & 2 & 1 \\ | & | & | & | & | & | \\ 1 & 1 & 1 & 0 & 0 & 0 \\ \hline \end{array}$$

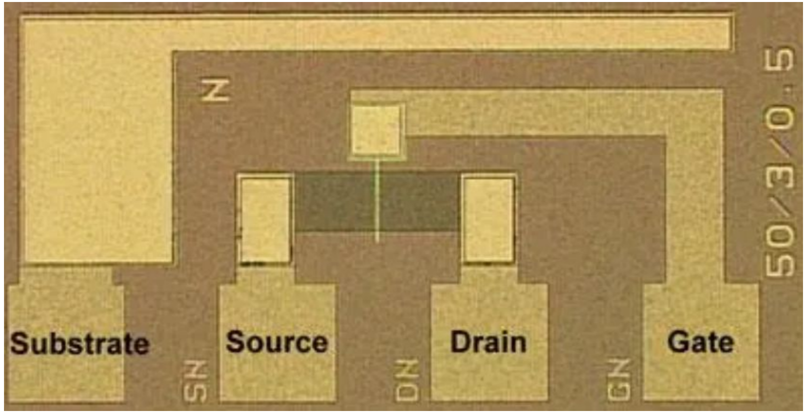
    $32 + 16 + 8 + 0 + 0 + 0 = 56$

Physical Data Storage: Hard Disk Drive (HDD)



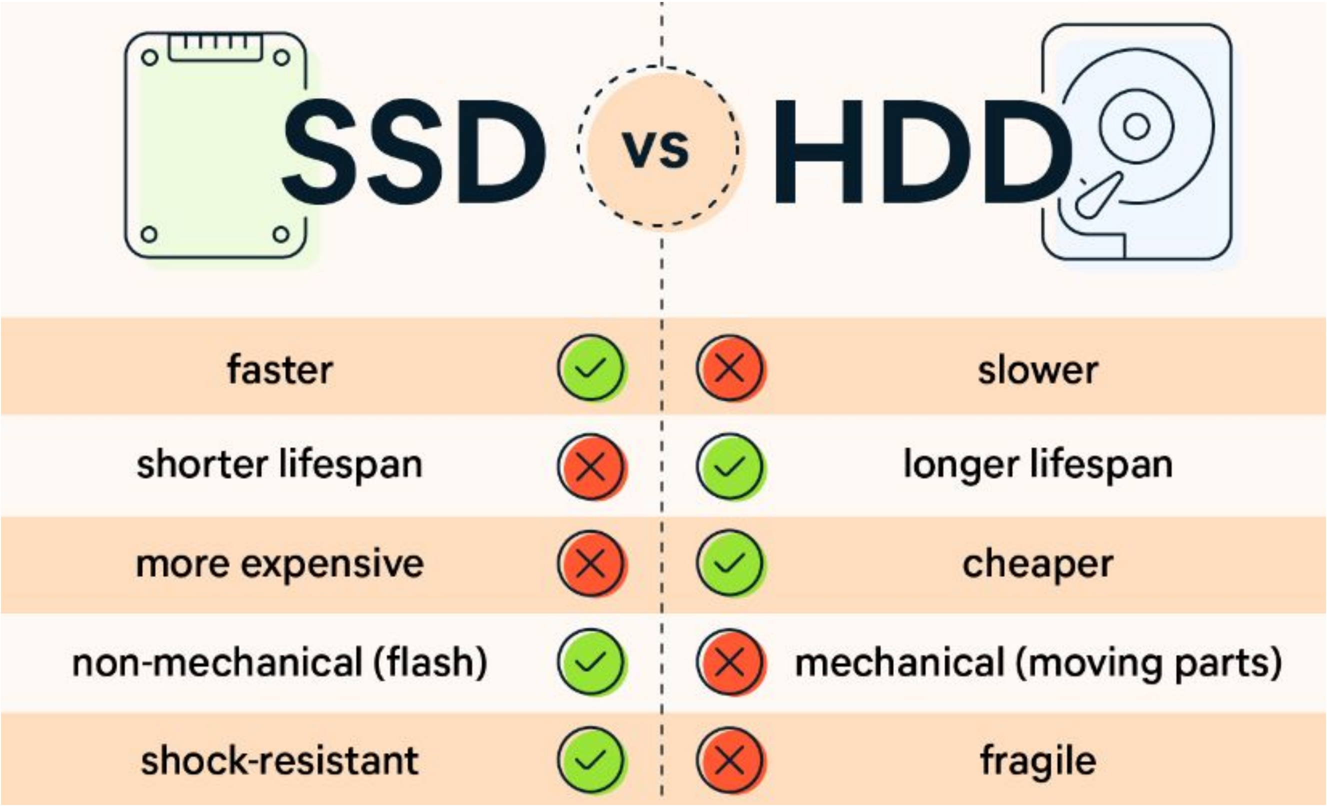
- A thin layer of magnetic alloy is on top of the HDD
- Write: the armature has a magnet that can produce a positive or negative charge and polarize a specific “spot” on the HDD
- Read: each “spot” on the HDD either “attracts” or “repels” the armature, which is converted to an electrical pulse

Physical Data Storage: Solid State Drive (SSD)



- Made up of transistors; has a source, gate, and drain
- Write: open the drain (send positive charge) to allow electrons to flow through, some of which get trapped in the gate; send negative charge through gate to push electrons out
- Read: 0s and 1s are determined by charge (presence of electrons) in gate

Physical Data Storage



SSD vs HDD

SSD			HDD
faster	✓	✗	slower
shorter lifespan	✗	✓	longer lifespan
more expensive	✗	✓	cheaper
non-mechanical (flash)	✓	✗	mechanical (moving parts)
shock-resistant	✓	✗	fragile

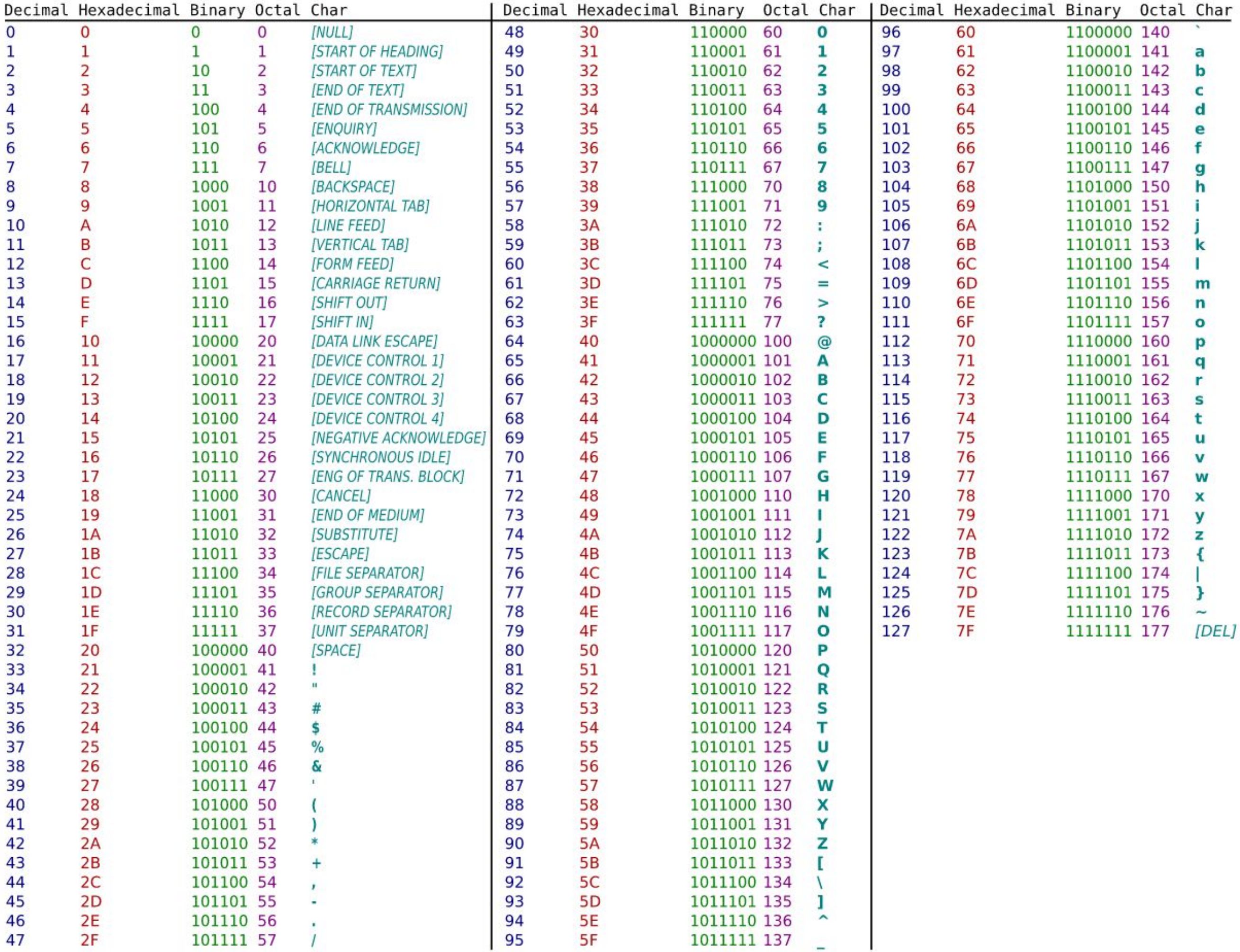
Encoding: Text

- Binary can represent any number, but what about letters and symbols?

- Different symbols/characters are mapped to different numbers, this mapping is called an “Encoding”

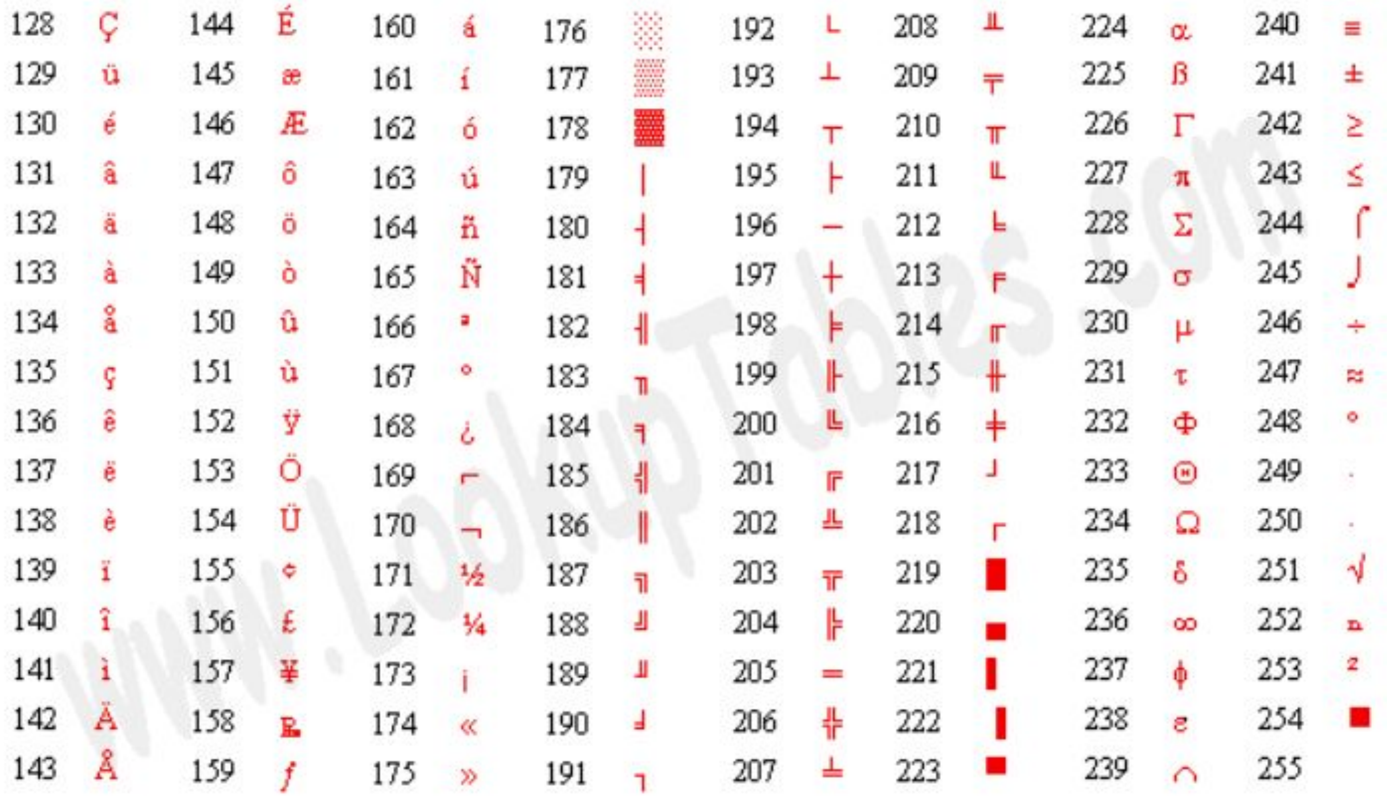
- There exist many different kinds of encoding but the 2 most used are:
  - **ASCII** (American Standard Code for Information Interchange) - Created in 1963, uses numbers 0 - 127 (a total of 7 bits) to represent all english letters, numbers 0-9, all symbols on the keyboard, many kinds of whitespace, etc.
  - **Unicode** - Created in 1991, has encodings for 1,112,064 characters, which represents practically all written languages in use in the world using an integer code (4 bytes); it has a number of different representations called Unicode Transformation Format (UTF): UTF-8, UTF-16, UTF-32
  - Most other text encodings are based on one of these 2

Encoding: ASCII



Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]
1	1	1	1	[START OF HEADING]
2	2	10	2	[START OF TEXT]
3	3	11	3	[END OF TEXT]
4	4	100	4	[END OF TRANSMISSION]
5	5	101	5	[ENQUIRY]
6	6	110	6	[ACKNOWLEDGE]
7	7	111	7	[BELL]
8	8	1000	10	[BACKSPACE]
9	9	1001	11	[HORIZONTAL TAB]
10	A	1010	12	[LINE FEED]
11	B	1011	13	[VERTICAL TAB]
12	C	1100	14	[FORM FEED]
13	D	1101	15	[CARRIAGE RETURN]
14	E	1110	16	[SHIFT OUT]
15	F	1111	17	[SHIFT IN]
16	10	10000	20	[DATA LINK ESCAPE]
17	11	10001	21	[DEVICE CONTROL 1]
18	12	10010	22	[DEVICE CONTROL 2]
19	13	10011	23	[DEVICE CONTROL 3]
20	14	10100	24	[DEVICE CONTROL 4]
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]
22	16	10110	26	[SYNCHRONOUS IDLE]
23	17	10111	27	[ENG OF TRANS. BLOCK]
24	18	11000	30	[CANCEL]
25	19	11001	31	[END OF MEDIUM]
26	1A	11010	32	[SUBSTITUTE]
27	1B	11011	33	[ESCAPE]
28	1C	11100	34	[FILE SEPARATOR]
29	1D	11101	35	[GROUP SEPARATOR]
30	1E	11110	36	[RECORD SEPARATOR]
31	1F	11111	37	[UNIT SEPARATOR]
32	20	100000	40	[SPACE]
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F
71	47	1000111	107	G
72	48	1001000	110	H
73	49	1001001	111	I
74	4A	1001010	112	J
75	4B	1001011	113	K
76	4C	1001100	114	L
77	4D	1001101	115	M
78	4E	1001110	116	N
79	4F	1001111	117	O
80	50	1010000	120	P
81	51	1010001	121	Q
82	52	1010010	122	R
83	53	1010011	123	S
84	54	1010100	124	T
85	55	1010101	125	U
86	56	1010110	126	V
87	57	1010111	127	W
88	58	1011000	130	X
89	59	1011001	131	Y
90	5A	1011010	132	Z
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_
96	60	1100000	140	`
97	61	1100001	141	a
98	62	1100010	142	b
99	63	1100011	143	c
100	64	1100100	144	d
101	65	1100101	145	e
102	66	1100110	146	f
103	67	1100111	147	g
104	68	1101000	150	h
105	69	1101001	151	i
106	6A	1101010	152	j
107	6B	1101011	153	k
108	6C	1101100	154	l
109	6D	1101101	155	m
110	6E	1101110	156	n
111	6F	1101111	157	o
112	70	1110000	160	p
113	71	1110001	161	q
114	72	1110010	162	r
115	73	1110011	163	s
116	74	1110100	164	t
117	75	1110101	165	u
118	76	1110110	166	v
119	77	1110111	167	w
120	78	1111000	170	x
121	79	1111001	171	y
122	7A	1111010	172	z
123	7B	1111011	173	{
124	7C	1111100	174	|
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	[DEL]

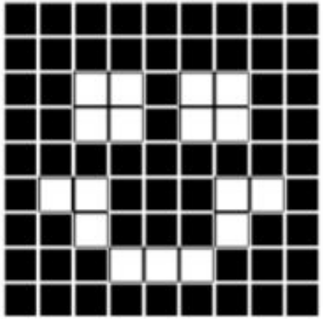
Encoding: Unicode



128	Ç	144	É	160	á	176	░	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	ⁿ
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

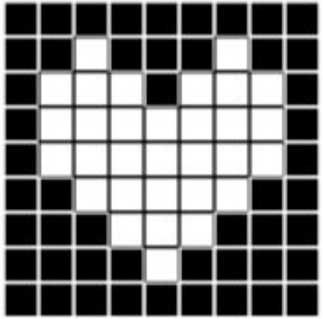
Encoding: Images

- To encode images, bits are used to represent different colors
  - Sound files use bits to represent different frequencies (period, amplitude, shift, etc.)
- Simplest form is black and white images: 0 represent black, 1 represents white (can also be the opposite)
- Raster vs Vector



=

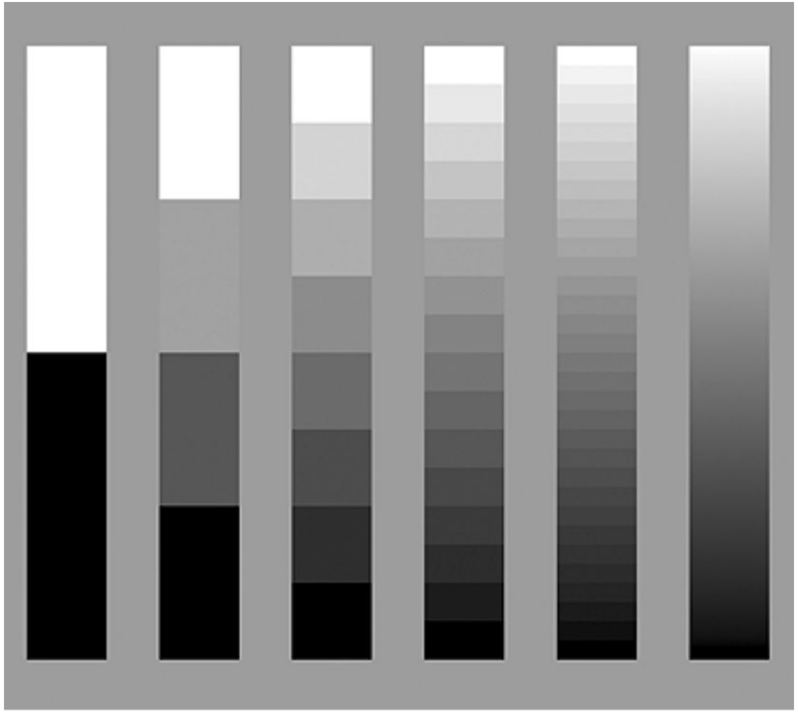
```
000000000
000000000
001101100
001101100
000000000
011000110
001000100
000111000
000000000
```



=

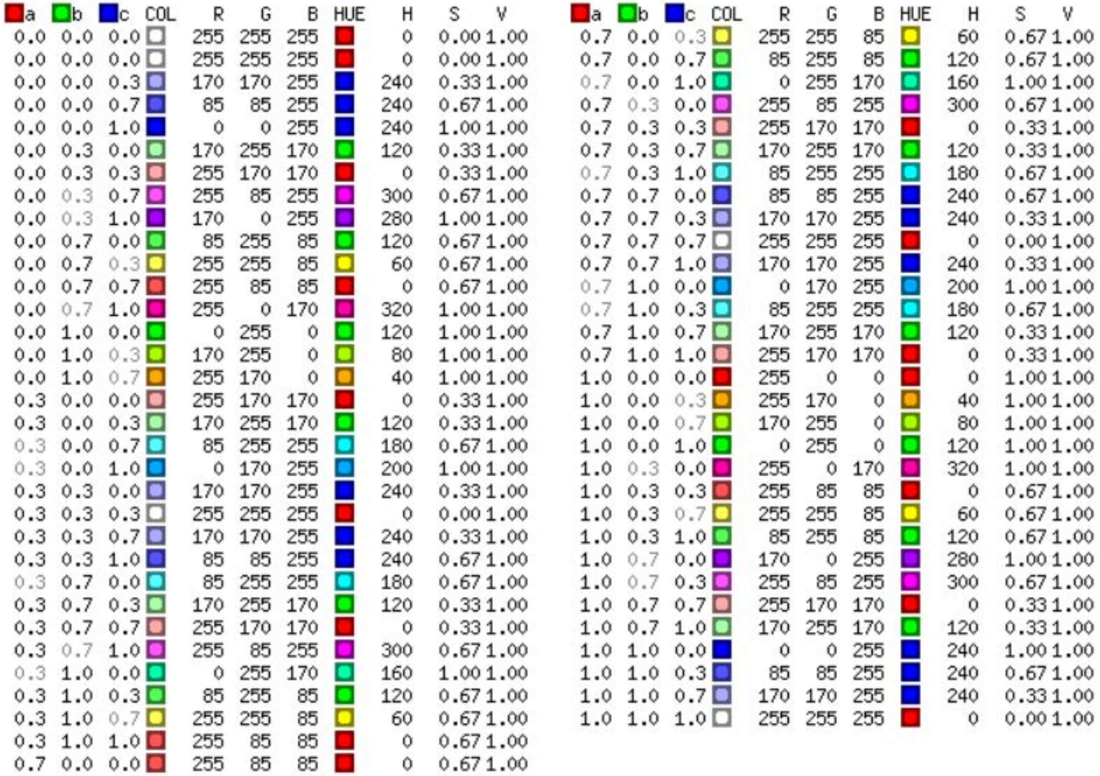
```
000000000
001000100
011101110
011111110
011111110
001111100
000111000
000010000
000000000
```

Encoding: Images



A grayscale representation of encoding images at various bit depths—from the top: 1 bit, 2 bits, 3 bits, 4 bits, 5 bits, and 8 bits.

Encoding: Images



🟥a	🟩b	🟦c	COL	R	G	B	HUE	H	S	V
0.0	0.0	0.0	⬜	255	255	255	🟥	0	0.00	1.00
0.0	0.0	0.0	⬜	255	255	255	🟥	0	0.00	1.00
0.0	0.0	0.3	🟦	170	170	255	🟦	240	0.33	1.00
0.0	0.0	0.7	🟦	85	85	255	🟦	240	0.67	1.00
0.0	0.0	1.0	🟦	0	0	255	🟦	240	1.00	1.00
0.0	0.3	0.0	🟩	170	255	170	🟩	120	0.33	1.00
0.0	0.3	0.3	🟥	255	170	170	🟥	0	0.33	1.00
0.0	0.3	0.7	🟪	255	85	255	🟪	300	0.67	1.00
0.0	0.3	1.0	🟪	170	0	255	🟪	280	1.00	1.00
0.0	0.7	0.0	🟩	85	255	85	🟩	120	0.67	1.00
0.0	0.7	0.3	🟨	255	255	85	🟨	60	0.67	1.00
0.0	0.7	0.7	🟥	255	85	85	🟥	0	0.67	1.00
0.0	0.7	1.0	🟪	255	0	170	🟪	320	1.00	1.00
0.0	1.0	0.0	🟩	0	255	0	🟩	120	1.00	1.00
0.0	1.0	0.3	🟨	170	255	0	🟨	80	1.00	1.00
0.0	1.0	0.7	🟧	255	170	0	🟧	40	1.00	1.00
0.3	0.0	0.0	🟥	255	170	170	🟥	0	0.33	1.00
0.3	0.0	0.3	🟩	170	255	170	🟩	120	0.33	1.00
0.3	0.0	0.7	🟦	85	255	255	🟦	180	0.67	1.00
0.3	0.0	1.0	🟦	0	170	255	🟦	200	1.00	1.00
0.3	0.3	0.0	🟦	170	170	255	🟦	240	0.33	1.00
0.3	0.3	0.3	⬜	255	255	255	🟥	0	0.00	1.00
0.3	0.3	0.7	🟦	170	170	255	🟦	240	0.33	1.00
0.3	0.3	1.0	🟦	85	85	255	🟦	240	0.67	1.00
0.3	0.7	0.0	🟦	85	255	255	🟦	180	0.67	1.00
0.3	0.7	0.3	🟩	170	255	170	🟩	120	0.33	1.00
0.3	0.7	0.7	🟥	255	170	170	🟥	0	0.33	1.00
0.3	0.7	1.0	🟪	255	85	255	🟪	300	0.67	1.00
0.3	1.0	0.0	🟩	0	255	170	🟩	160	1.00	1.00
0.3	1.0	0.3	🟩	85	255	85	🟩	120	0.67	1.00
0.3	1.0	0.7	🟨	255	255	85	🟨	60	0.67	1.00
0.3	1.0	1.0	🟥	255	85	85	🟥	0	0.67	1.00
0.7	0.0	0.0	🟥	255	85	85	🟥	0	0.67	1.00

🟥a	🟩b	🟦c	COL	R	G	B	HUE	H	S	V
0.7	0.0	0.3	🟨	255	255	85	🟨	60	0.67	1.00
0.7	0.0	0.7	🟩	85	255	85	🟩	120	0.67	1.00
0.7	0.0	1.0	🟩	0	255	170	🟩	160	1.00	1.00
0.7	0.3	0.0	🟪	255	85	255	🟪	300	0.67	1.00
0.7	0.3	0.3	🟥	255	170	170	🟥	0	0.33	1.00
0.7	0.3	0.7	🟩	170	255	170	🟩	120	0.33	1.00
0.7	0.3	1.0	🟦	85	255	255	🟦	180	0.67	1.00
0.7	0.7	0.0	🟦	85	85	255	🟦	240	0.67	1.00
0.7	0.7	0.3	🟦	170	170	255	🟦	240	0.33	1.00
0.7	0.7	0.7	⬜	255	255	255	🟥	0	0.00	1.00
0.7	0.7	1.0	🟦	170	170	255	🟦	240	0.33	1.00
0.7	1.0	0.0	🟦	0	170	255	🟦	200	1.00	1.00
0.7	1.0	0.3	🟦	85	255	255	🟦	180	0.67	1.00
0.7	1.0	0.7	🟩	170	255	170	🟩	120	0.33	1.00
0.7	1.0	1.0	🟥	255	170	170	🟥	0	0.33	1.00
1.0	0.0	0.0	🟥	255	0	0	🟥	0	1.00	1.00
1.0	0.0	0.3	🟧	255	170	0	🟧	40	1.00	1.00
1.0	0.0	0.7	🟨	170	255	0	🟨	80	1.00	1.00
1.0	0.0	1.0	🟩	0	255	0	🟩	120	1.00	1.00
1.0	0.3	0.0	🟪	255	0	170	🟪	320	1.00	1.00
1.0	0.3	0.3	🟥	255	85	85	🟥	0	0.67	1.00
1.0	0.3	0.7	🟨	255	255	85	🟨	60	0.67	1.00
1.0	0.3	1.0	🟩	85	255	85	🟩	120	0.67	1.00
1.0	0.7	0.0	🟪	170	0	255	🟪	280	1.00	1.00
1.0	0.7	0.3	🟪	255	85	255	🟪	300	0.67	1.00
1.0	0.7	0.7	🟥	255	170	170	🟥	0	0.33	1.00
1.0	0.7	1.0	🟩	170	255	170	🟩	120	0.33	1.00
1.0	1.0	0.0	🟦	0	0	255	🟦	240	1.00	1.00
1.0	1.0	0.3	🟦	85	85	255	🟦	240	0.67	1.00
1.0	1.0	0.7	🟦	170	170	255	🟦	240	0.33	1.00
1.0	1.0	1.0	⬜	255	255	255	🟥	0	0.00	1.00

File Formats

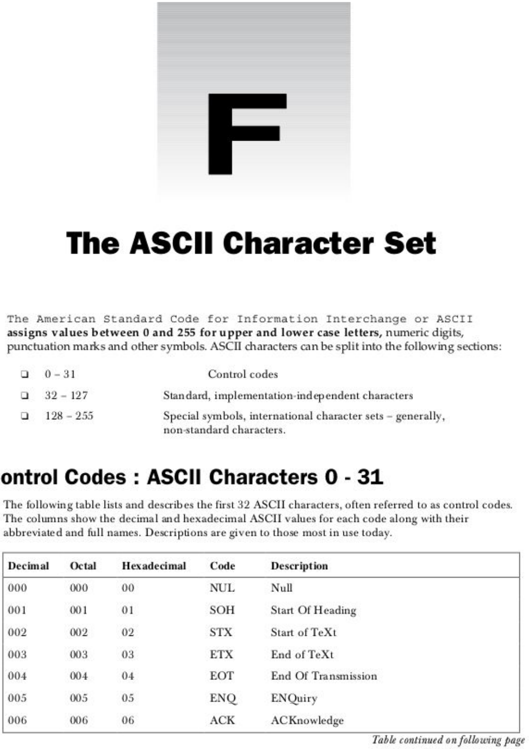
- How data/information is structured for storage

- A specifications document is often available (not always, e.g. proprietary formats) so the format can be decoded and used by many applications
  - Specifications can include: structure, extended encodings, tags, metadata, extensions, etc.
  - Note: file formats often have an extension to help identify them, but these are not necessary

- There are about 3000 "common" file formats, and many thousands more "uncommon": https://en.wikipedia.org/wiki/List_of_file_formats

- Examples:
  - Text: PDF (.pdf), Word (.doc), plain text (.txt), etc.
  - Image: JFIF (used by .jpeg), TIFF (.tif), etc.
  - Sound: WAV (.wav), MP4 (.mp4), etc.

File Formats: Text



The ASCII Character Set

The American Standard Code for Information Interchange or ASCII **assigns values between 0 and 255 for upper and lower case letters,** numeric digits, punctuation marks and other symbols. ASCII characters can be split into the following sections:

❑	0 – 31	Control codes
❑	32 – 127	Standard, implementation-independent characters
❑	128 – 255	Special symbols, international character sets – generally, non-standard characters.

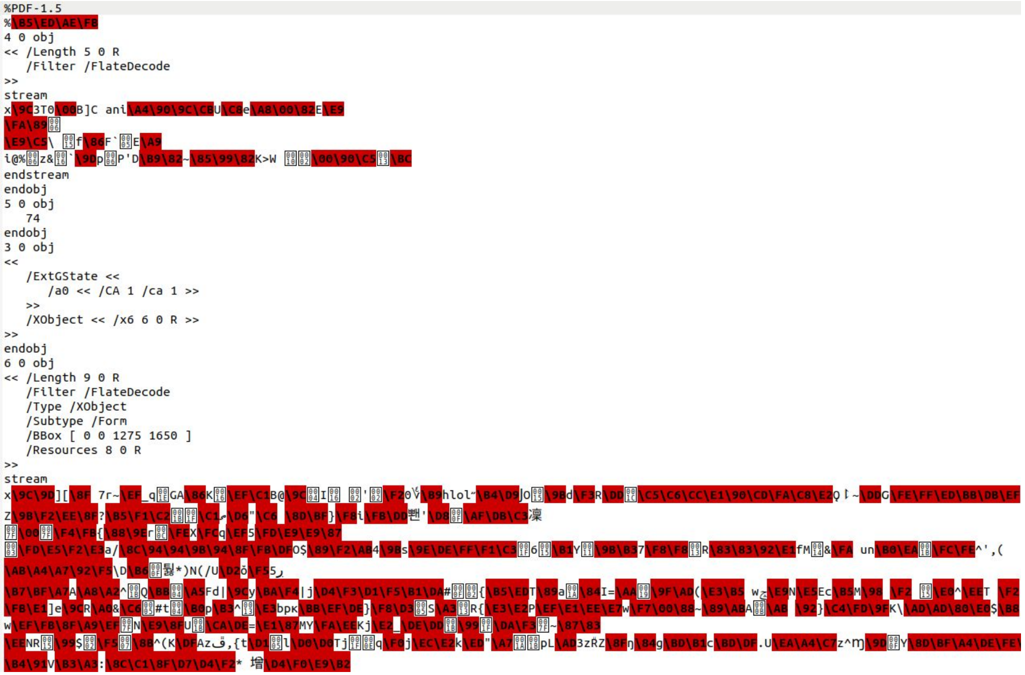
Control Codes : ASCII Characters 0 - 31

The following table lists and describes the first 32 ASCII characters, often referred to as control codes. The columns show the decimal and hexadecimal ASCII values for each code along with their abbreviated and full names. Descriptions are given to those most in use today.

Decimal	Octal	Hexadecimal	Code	Description
000	000	00	NUL	Null
001	001	01	SOH	Start Of Heading
002	002	02	STX	Start of TeXt
003	003	03	ETX	End of TeXt
004	004	04	EOT	End Of Transmission
005	005	05	ENQ	ENQuiry
006	006	06	ACK	ACKnowledge

Table continued on following page

PDF using a PDF reader

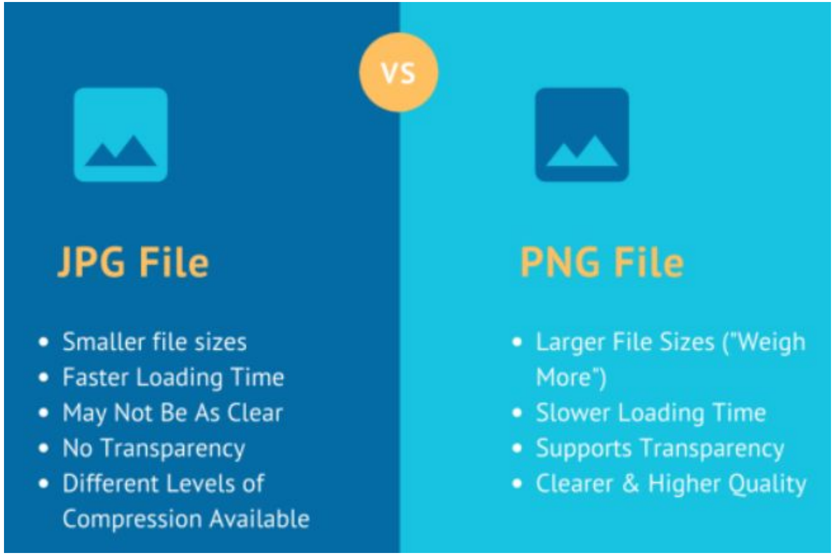


```
%PDF-1.5
%\B5\ED\AE\FB
4 0 obj
<< /Length 5 0 R
   /Filter /FlateDecode
>>
stream
x\9C3T0\00B]C ani\A4\90\9C\CBU\C8e\A8\00\82E\E9
\FA\89□
\E9\C5\ □f\86F`□E\A9
i@%□z&□`\9Dp□P'D\B9\82~\85\99\82K>W □□\00\90\C5□\BC
endstream
endobj
5 0 obj
   74
endobj
3 0 obj
<<
   /ExtGState <<
      /a0 << /CA 1 /ca 1 >>
   >>
   /XObject << /x6 6 0 R >>
>>
endobj
6 0 obj
<< /Length 9 0 R
   /Filter /FlateDecode
   /Type /XObject
   /Subtype /Form
   /BBox [ 0 0 1275 1650 ]
   /Resources 8 0 R
>>
stream
x\9C\9D][\8F 7r~\EF_q□GA\86K□\EF\C1B@\9C□I□ □'□\F20\89hlol˝\B4\D9JO□\9Bd\F3R\DD□\C5\C6\CC\E1\90\CD\FA\C8\E2Q↿~\DDG\FE\FF\ED\BB\DB\EF
Z\9B\F2\EE\8F?\B5\F1\C2□□\C1\D6"\C6 \8D\BF}\F8i\FB\DD뾉'\D8□\AF\DB\C3凜
□\00□\F4\FB{\88\9Er□\FEX\FCq\EF5\FD\E9\E9\87
□\FD\E5\F2\E3a/\8C\94\94\9B\94\8F\FB\DFO$\89\F2\AB4\9Bs\9E\DE\FF\F1\C3□6□\B1Y□\9B\B37\F8\F8□R\83\83\92\E1fM□&\FA un\B0\EA□\FC\FE^',(
\AB\A4\A7\92\F5\D\B6□턟*)N(/U\D2ŏ\F55ر
\B7\BF\A7A\A8\A2^□Q\BB□\A5Fd|\9Cy\BA\F4|j\D4\F3\D1\F5\B1\DA#□□{\B5\EDT\89a□\84I=\AA□\9F\AD(\E3\B5 wڪ\E9N\E5Ec\B5M\98 \F2 □\E0^\EET \F2
\FB\E1]e\9CR\A0&\C6□#t□\B0p\B3^□\E3bpĸ\BB\EF\DE}\F8\D3□S\A3□R{\E3\E2P\EF\E1\EE\E7w\F7\00\88~\89\ABA□\AB \92}\C4\FD\9FK\\AD\AD\80\E0$\B8
w\EF\FB\8F\A9\EF□N\E9\8FU□\CA\DE=\E1\87MY\FA\EEKj\E2_\DE\DD□\99□\DA\F3□~\87\83
\EENR□\99$□\F5□\88^(K\DFAzق,{t\D1□l\D0\D0Tj□□q\F0j\EC\E2k\ED"\A7□□pL\AD3zŘZ\8Fŋ\84g\BD\B1c\BD\DF.U\EA\A4\C7z^ɱ\9D□Y\8D\BF\A4\DE\FE\
\B4\91V\B3\A3:\8C\C1\8F\D7\D4\F2* 増\D4\F0\E9\B2
```

Plain text PDF file

File Formats: Images

- For any file format (not just images), it's all about the features offered



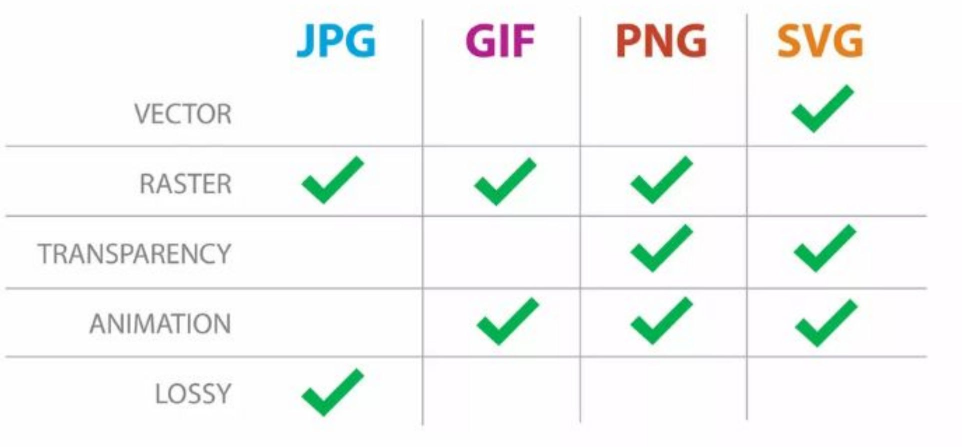
VS

JPG File

- Smaller file sizes
- Faster Loading Time
- May Not Be As Clear
- No Transparency
- Different Levels of Compression Available

PNG File

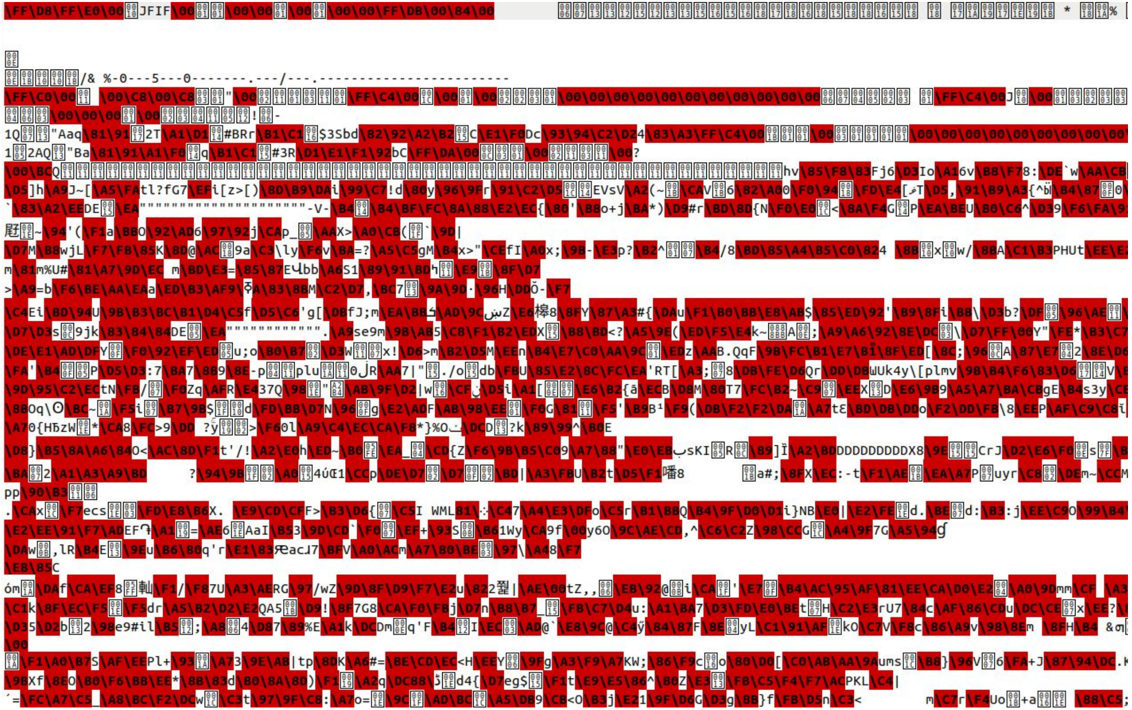
- Larger File Sizes ("Weigh More")
- Slower Loading Time
- Supports Transparency
- Clearer & Higher Quality



	JPG	GIF	PNG	SVG
VECTOR				✓
RASTER	✓	✓	✓	
TRANSPARENCY			✓	✓
ANIMATION		✓	✓	✓
LOSSY	✓			

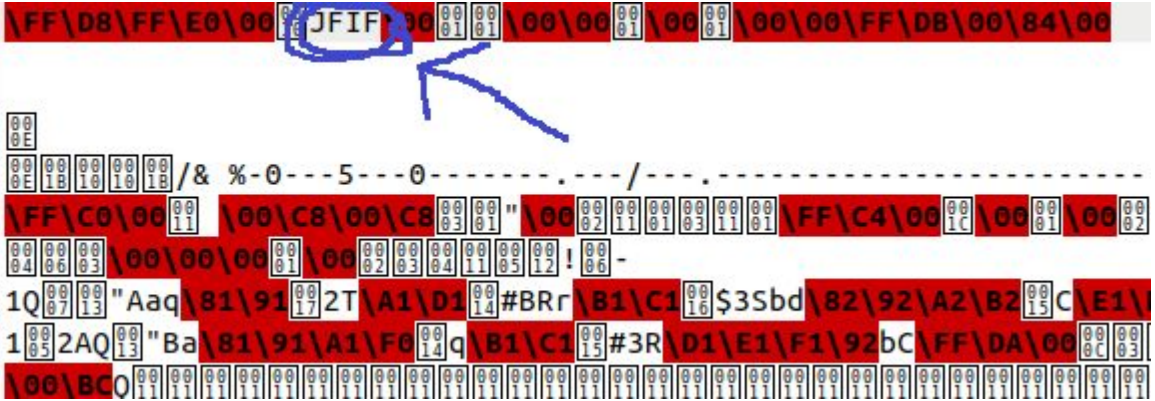
File Formats: Images

JPEG in an image viewer



Plain text JPEG file

File Format vs Encoding



```
\FF\D8\FF\E0\00[0010]JFIF\00[0001][0001]\00\00[0001]\00[0001]\00\00\FF\DB\00\84\00

[000E]
[000E][001B][0010][0010][001B]/& %-0---5---0-------.---/---.----------------------
\FF\C0\00[0011] \00\C8\00\C8[0003][0001]"\00[0002][0011][0001][0003][0011][0001]\FF\C4\00[001C]\00[0001]\00[0002]
[0004][0006][0003]\00\00\00[0001]\00[0002][0003][0004][0011][0005][0012]![0006]-
1Q[0007][0013]"Aaq\81\91[0017]2T\A1\D1[0014]#BRr\B1\C1[0016]$3Sbd\82\92\A2\B2[0015]C\E1\
1[0005]2AQ[0013]"Ba\81\91\A1\F0[0014]q\B1\C1[0015]#3R\D1\E1\F1\92bC\FF\DA\00[000C][0003]
\00\BC0[0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011][0011]
```

- The encoding is the specific mapping of symbols, colors, frequencies, etc., the file format is the organization and utilization of that encoding
- The encoding is ONLY the mapping of non-numerical data to numbers (and in some cases also contains a compression algorithms), it is just a representation
- File formats provide particular features, constraints, actions (e.g., GIF), etc. to the user to best utilize the underlying encoding
- Example: JFIF (JPEG File Interchange Format) is the format and JPEG is the encoding (also includes the compression algorithm)
  - Sometimes can be confusing: PNG is an encoding and often referred to as a file format as well but the technical file format for PNG is MIME