Université Pierre-Mendès France

Digital multimedia

Text and Image Representation

Outline

1. **Representation of media**
   - **Text**
     - ASCII
     - Unicode
   - **Image**
   - Audio
   - Video

ASCII

- Most **text files** are represented using the **ASCII** coding scheme
  - Each character is translated into **7 bits**
  - Can contain at most **128 symbols** (i.e., 2^7):
    - Latin alphabet and Arabic numerals
    - Standard punctuation characters
    - Small set of accents and other European special characters (Latin-I ASCII)
  - **Not enough** for many languages

ASCII Table

	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EDT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

ASCII Table

	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EDT	DC4	$	4	D	T	d	t
0101			%	5			e	u
0110			&	6			f	v
0111			'	7			g	w
1000			(	8			h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Control Characters

Printable Characters

ASCII Table

Most significant bit

Least significant bit

'a' = **110** 0001

	000	001	010	011	100	101	110	111
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C			
0100	EDT	DC4	$	4	D			
0101	ENQ	NAK	%	5	E			
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

ASCII Table (Hexadecimal)

MSD / LSD	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P		p
1	SOH	DC1	!	1	A	Q	a	W
2	STX	DC2	“	2	B	R	b	r
3	ETX	DC3	#	3	C	S		
4	EOT	DC4	$	4	D	T		
5	ENQ	NAK	%	5	E	U		
6	ACJ	SYN	&	6	F	V	f	v
7	BEL	ETB	‘	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

‘a’ = 61_{HEX}

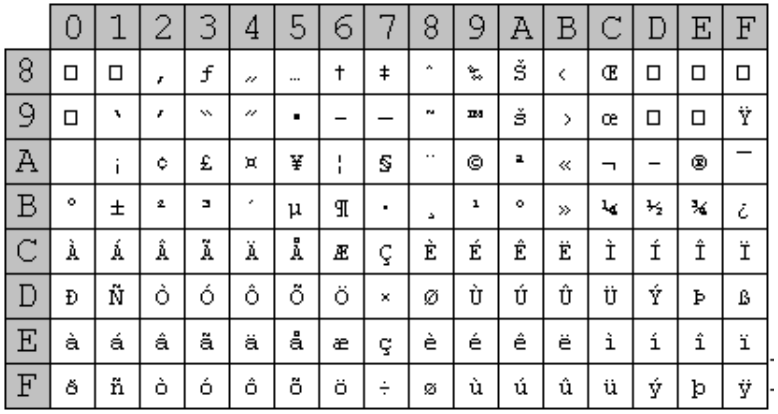
ASCII Example

- Representing "**Hello world**" in ASCII

		ASCII	ASCII HEX	URL Encoded ?
H	=	100 1000	48	
e	=	110 0101	65	
l	=	110 1100	6C	
l	=	110 1100	6C	
o	=	110 1111	6F	
	=	010 0000	20	
w	=	111 0111	77	
o	=	110 1111	6F	
r	=	111 0010	72	
l	=	110 1100	6C	
d	=	110 0100	64	

ASCII Extension

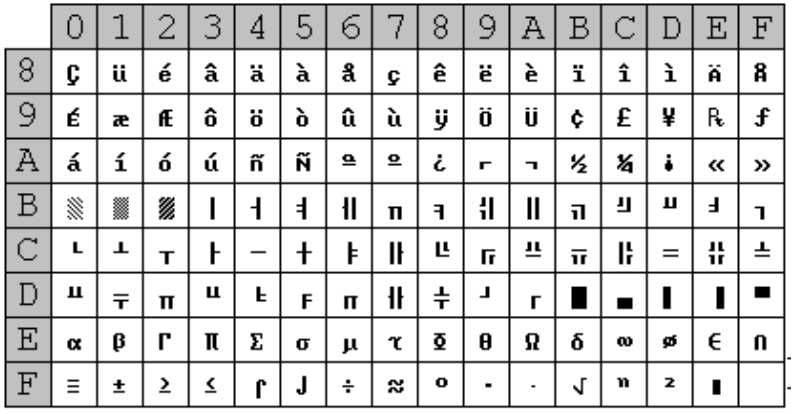
- Uses **8 bits** to represent a character
  - can code **256 letters** or symbols (2^8)
  - better for many European and Asian languages



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	□	□	‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ	□	□	□
9	□	‘	’	“	”	•	–	—	˜	™	š	›	œ	□	□	Ÿ
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
B	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

OEM extended ASCII
IBM-PC

Popular extensions



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
9	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	₧	ƒ
A	á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
B	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E	α	ß	Γ	π	Σ	σ	µ	τ	Φ	Θ	Ω	δ	∞	ø	∈	∩
F	≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	ⁿ	²	■	

ANSI extended ASCII
Windows

Unicode

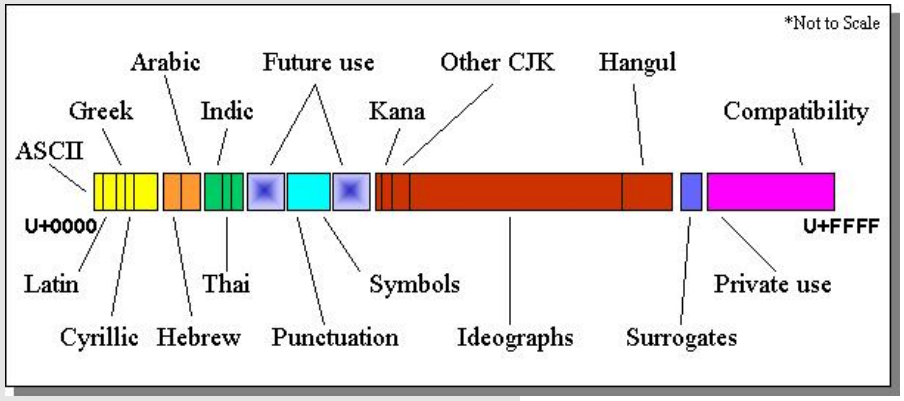
- Unicode is a **16-bit** code
  - Can represent **65, 536** characters (2^{16})
  - Enough for any language character set
  - Useful in international business documents
  - **Subsumes** ASCII
- Unicode helps with the **location** of software

  → Software modification for local-languages

Unicode Code Ranges

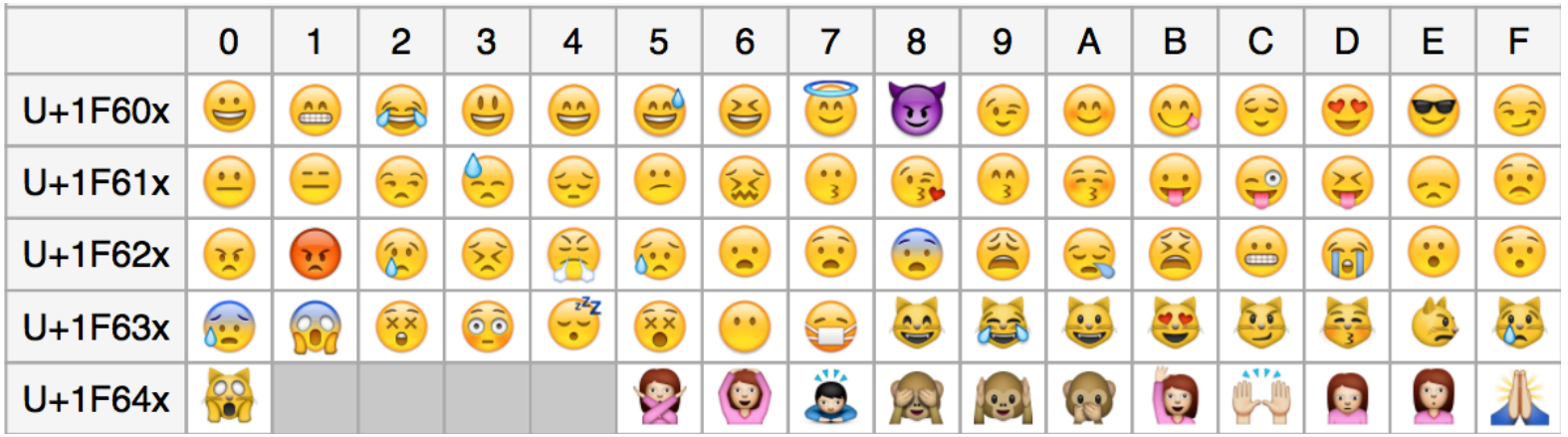
Code range (in hexadecimal)

0000	0000–00FF Latin-1 (ASCII)
1000	General character alphabets: Latin, Cyrillic, Greek, Hebrew, Arabic, Thai, etc.
2000	Symbols and dingbats: punctuation, math, technical, geometric shapes, etc.
3000	3000–33FF Miscellaneous punctuations, symbols, and phonetics for Chinese, Japanese, and Korean
4000	Unassigned
5000	4E00–9FFF Chinese, Japanese, Korean ideographs
A000	Unassigned
B000	
C000	AC00–D7AF Korean Hangul syllables
D000	
E000	Space for surrogates
F000	E000–F8FF Private use
FFFF	Various special characters



Unicode Range Exemple

- **Emoticons** (range 1F600-1F64F)



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+1F60x	😀	😁	😂	😃	😄	😅	😆	😇	😈	😉	😊	😋	😌	😍	😎	😏
U+1F61x	😐	😑	😒	😓	😔	😕	😖	😗	😘	😙	😚	😛	😜	😝	😞	😟
U+1F62x	😠	😡	😢	😣	😤	😥	😦	😧	😨	😩	😪	😫	😬	😭	😮	😯
U+1F63x	😰	😱	😲	😳	😴	😵	😶	😷	😸	😹	😺	😻	😼	😽	😾	😿
U+1F64x	🙀					🙅	🙆	🙇	🙈	🙉	🙊	🙋	🙌	🙍	🙎	🙏

Emoticons in HTML

UTF Encoding

- Unicode is a **machine independent** representation
- UTF-X specifies
  - **how to** serialize a Unicode character
  - the **number of bits** to use (e.g., UTF-8, UTF-16, UTF-32, …)
- UTF-8 is the ***de facto* standard**
  - **Backward compatible** with ASCII (Unix systems)

    128 UTF-8 first characters correspond to ASCII characters
  - Preferred encoding mechanism for multi-language web pages

UTF-8 Encoding Principle

UTF-8	Serialized Bytes					
Unicode Range	1st	2nd	3rd	4th	5th	6th
U-00000000 – U-0000007F	0nnnnnnn					
U-00000080 – U-000007FF	110nnnnn	10nnnnnn				
U-00000800 – U-0000FFFF	1110nnnn	10nnnnnn	10nnnnnn			
U-00010000 – U-001FFFFF	11110nnn	10nnnnnn	10nnnnnn	10nnnnnn		
U-00200000 – U-03FFFFFF	111110nn	10nnnnnn	10nnnnnn	10nnnnnn	10nnnnnn	
U-04000000 – U-7FFFFFFF	1111110n	10nnnnnn	10nnnnnn	10nnnnnn	10nnnnnn	10nnnnnn

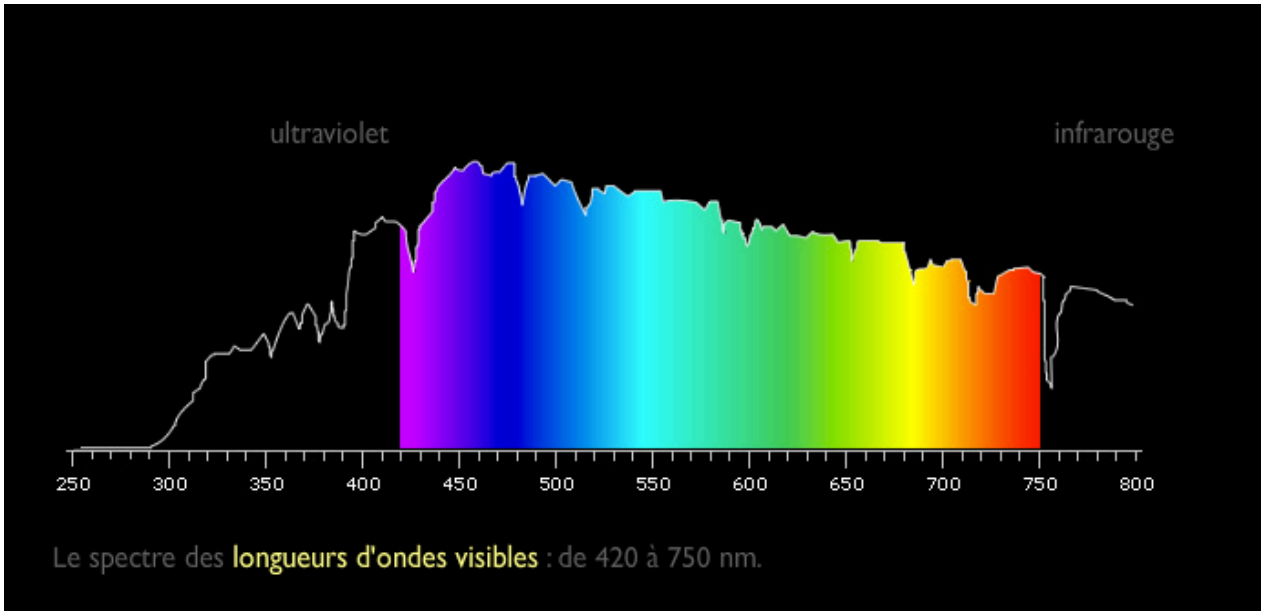
- UTF-8 **serializes** a Unicode characters into **multiple bytes**
- The high bits of the first serialized byte indicate **how many bytes** are used for the serialization of that character
- The bits represented by "**n**"s hold the Unicode character code value

Outline

1. **Representation of media**
   - Text
   - **Image**
     - **Image data types**
     - **Popular formats**
   - Audio
   - Video

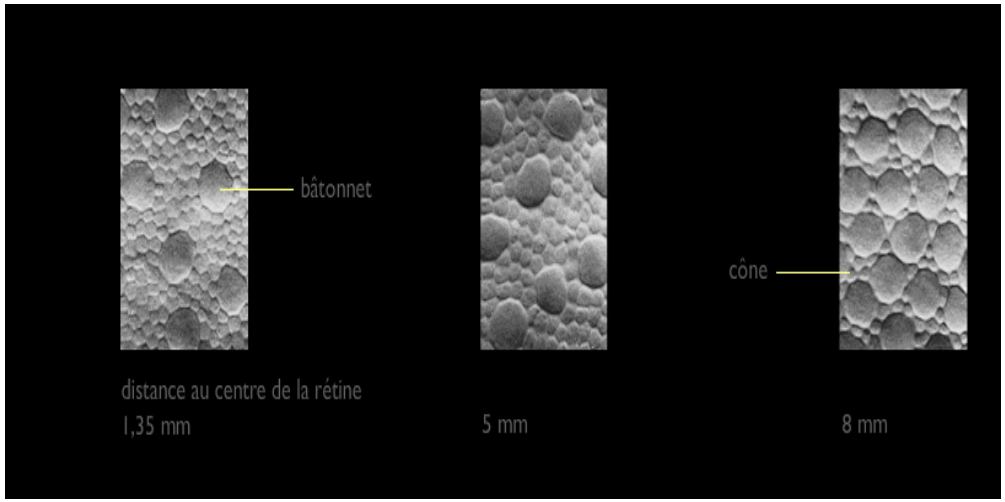
Color Perception

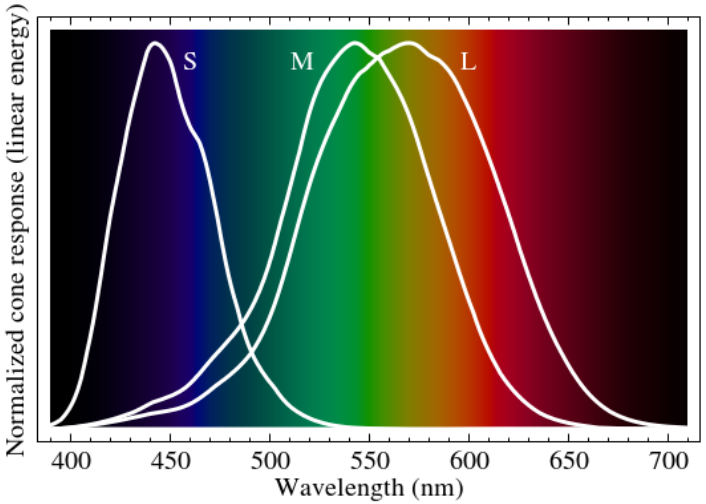
- In the real-world images are a continuous spectrum of colors



Color Perception

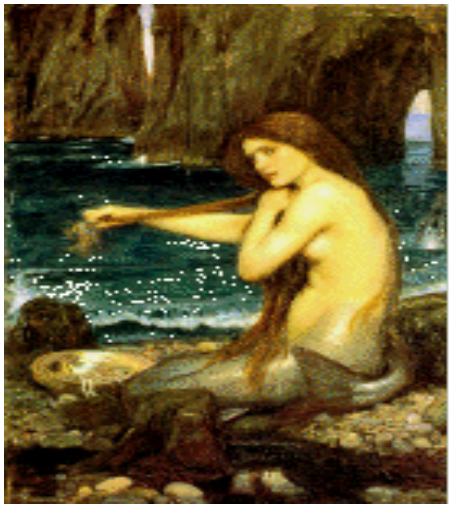
- How humans perceive colors



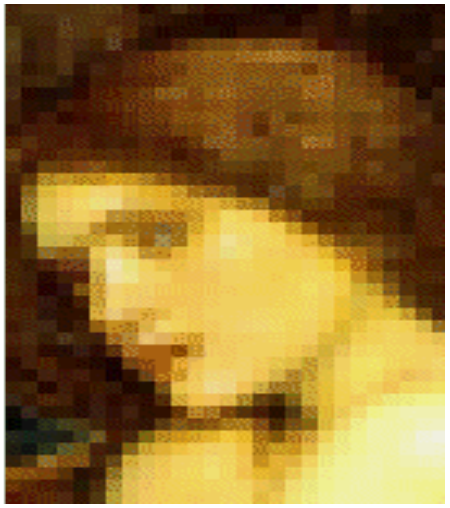


Bitmap and Pixels

- Computer handles real images by "**discretizing**" the color spectrum into small "*bit of lights*" and producing **bitmaps**



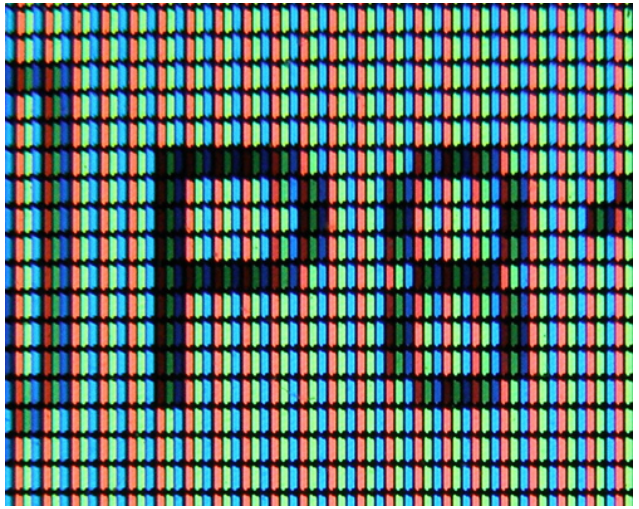
Real world



Bitmap

Bitmap and Pixels

- The colored dots that make up a bitmap are properly called **pixels** (i.e. picture elements)
- In the simplest sort of bitmapped image, each pixel is represented by red, green and blue light



Bitmaps Images

- Images can be **classified** according to the **number of bits** used for representing them:
  - Monochrome (black or white)

    1 bit per pixel
  - Gray scale (black, white and 254 shades of gray)

    1 byte per pixel
  - Color images:
    - 16 colors = 256 colors
    - 24-bit = (16.7 million colors)

1-bit Monochrome Image

- Simplest type of image
  - Contains **no color**
- Each pixel is **stored as a single bit** (0 or 1)
  - → Also referred as *binary image*
- Usage

  Printing tickets, news paper, fax, etc.

Lena

1-bit Monochrome Image

- What is the size of this image ? (in bytes)

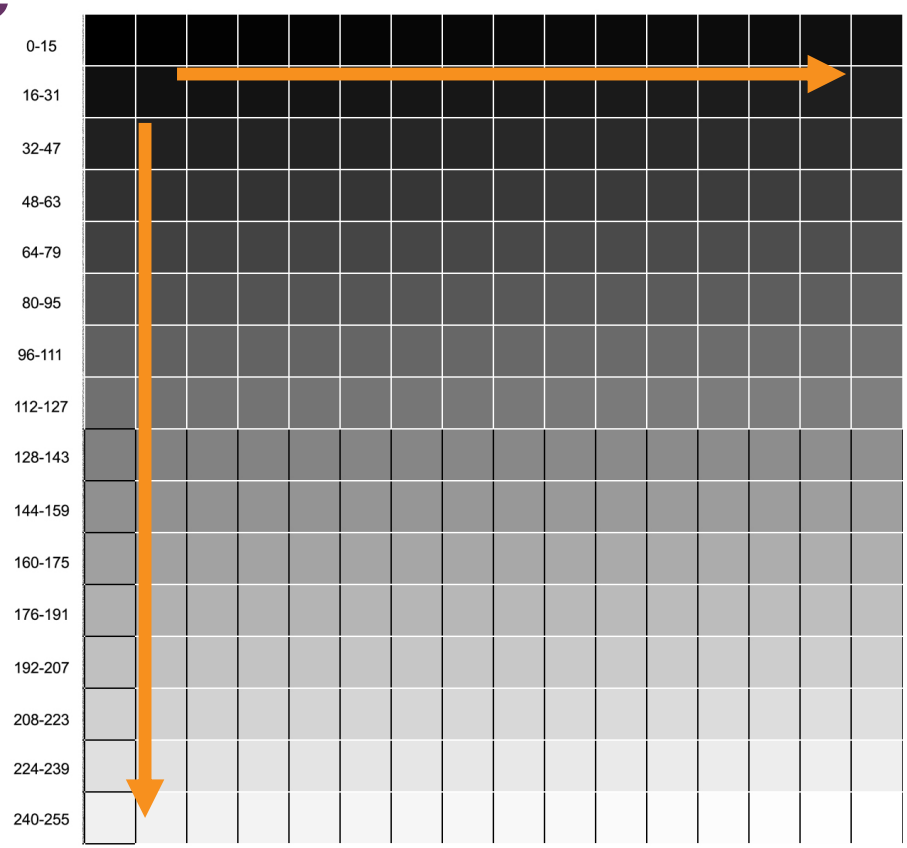
480 px

38,4 Kbytes

640 px

8-bit Gray Scale Image

- Represent **gray** images
- Each pixel is represented by a **single byte**
- Each pixel has a **gray-value** between **0 and 255** (i.e., 2^8)
  - e.g., a dark pixel might have a value of 10, and a bright one might be 230



8-bit Gray Scale Image

- What is the size of an image of this image?

640 px

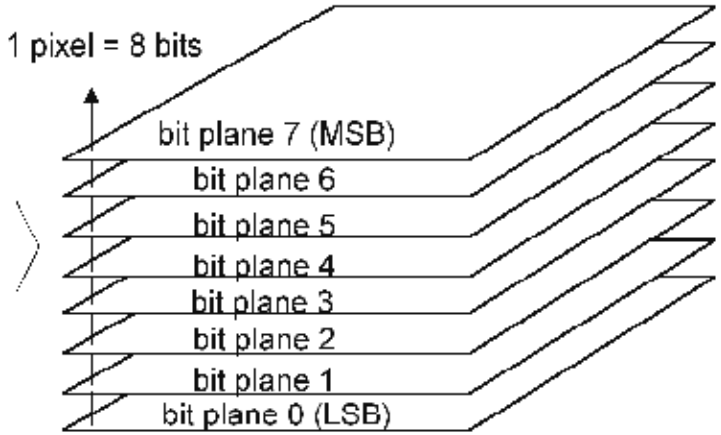
480 px

300 Kbytes

Gray scale image of Lena

Bit planes

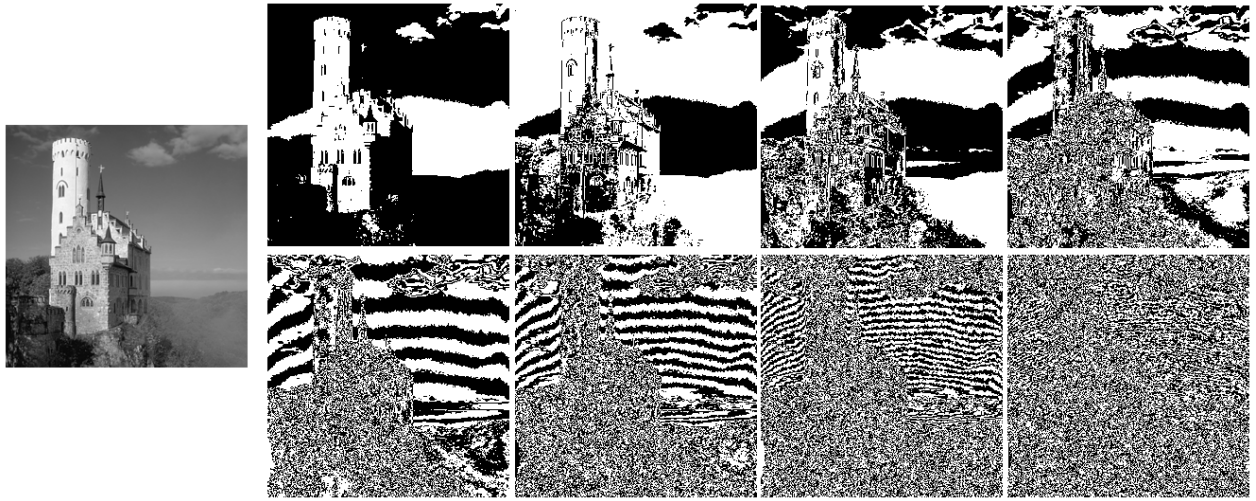
- An 8-bit image can be thought of as a set of **bit-planes**
  - Each plane consists of a **1-bit** representation of the image at **higher levels of elevation**
  - A bit is **turned on** if the image pixel >= bit level



Bit Plane	Value	Contribution	Running Total
1st	1	1 * 2^7 = 128	128
2nd	0	0 * 2^6 =0	128
3rd	1	1 * 2^5 = 32	160
4th	1	1 * 2^4 = 16	176
5th	0	0 * 2^3 = 0	176
6th	1	1 * 2^2 = 4	180
7th	0	0 * 2^1 = 0	180
8th	1	1 * 2^0 = 1	181

Bit planes

- Note that
  - the first plane gives the most critical approximation to the image
  - lower the number of the bit plane, less is its contribution to the final image
  - adding a bit plane gives a better approximation



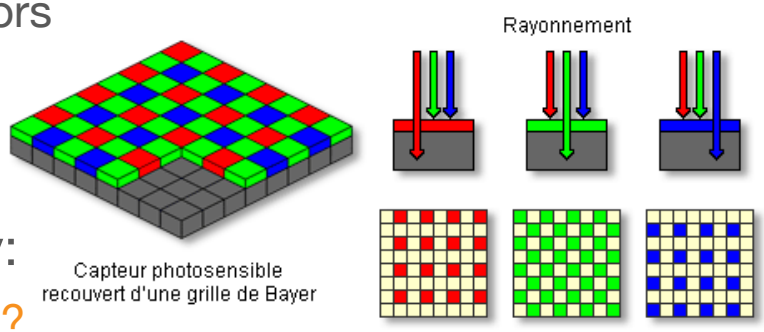
Dithering

- Strategy that trade **intensity resolution** by **spatial resolution** for giving the illusion of **bit-depth** transition

24-bit Color Image

- A pixel is represented by **three bytes** (usually representing **RGB**)
  - Supports 256 x 256 x 256 possible colors (i.e., 16,777,216)
  - Known also as **True Color**

    The human eye discriminates up to 10 million colors



- Such flexibility does result in a storage penalty:
  - Ex. A 640 x 480 24-bit color image would require ?

    921.6 kB of storage (without any compression)

24-bit Color Image

- Sometimes these images are stored as **32-bit images** for representing special effect information
- The **extra byte of data** for each pixel is called **alpha value**
- Can be used for representing **transparency**
  - Ex. for composing several *overlapping objects*

24-bit Color Image

- High-resolution color and separate R, G, B color channel images

Original

Red

Green

Blue

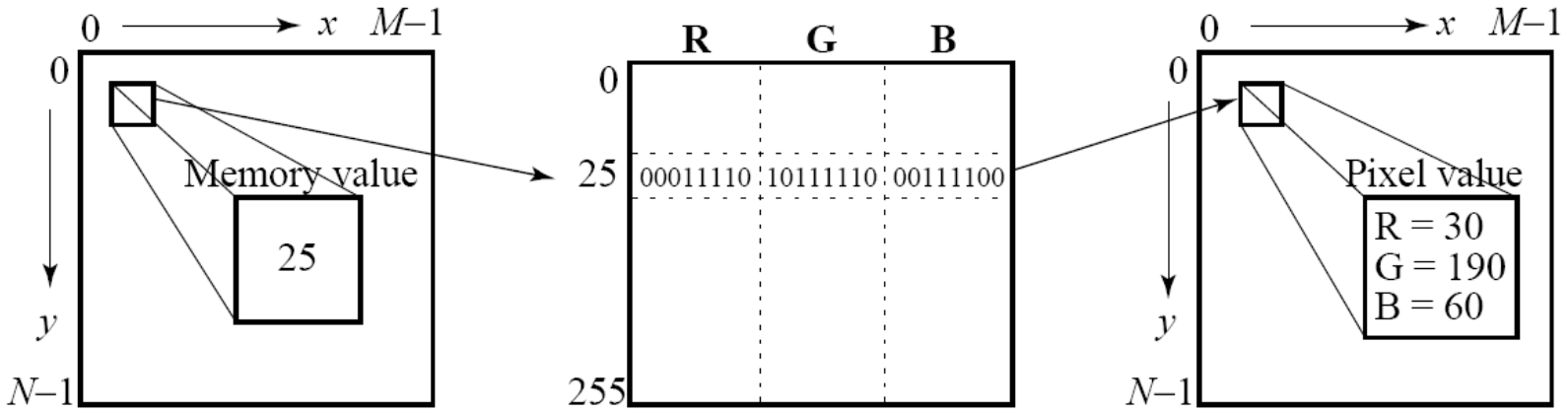
8-bit Color Image

- When space is a concern reasonable accurate color images can be obtained by using 8 bits of color (the so-called “**256 colors**”)
- Many systems can make use of 8 bits color images to be backward compatible
  - Ex. Mobile phones previous to smartphones
- Such image files use the concept of a **lookup table** (LUTs) to store color information
  - The image stores no color but a pointer into a table with 3-byte values that specify the color for a pixel

Color Lookup Table (LUTs)

- Ex. if a pixel stores the value 25, the meaning is to go to row 25 in a LUT

  → a LUT is a kind of **palette of colors** (e.g., web safe colors)



Color Lookup Table (LUTs)

- Choosing the best 256 colors based on an RGB histogram
  - In this example there is almost **no blue**. Thus the 256 colors can be composed mostly of red and green

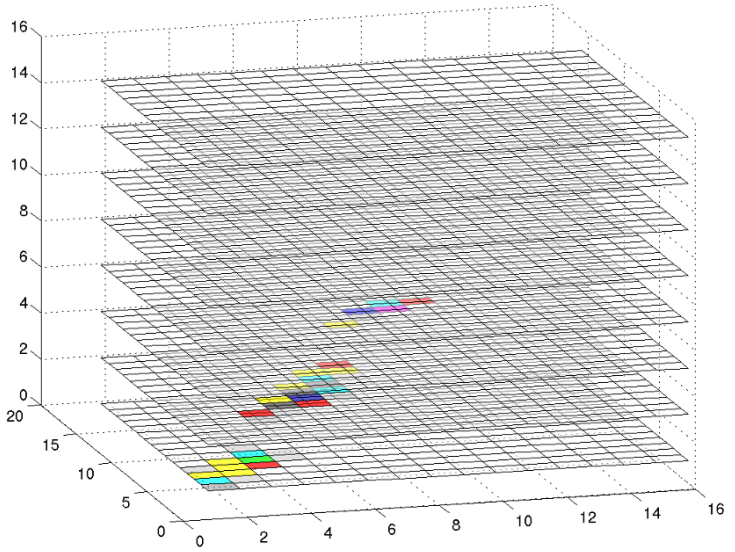


Image File Formats

- The number of **file formats** continues to **proliferate**

File Import					File Export		Native
Image	Palette	Sound	Video	Anim.	Image	Video	
.BMP, .DIB, .GIF, .JPG, .PICT .PNG, .PNT, .PSD, .TGA, .TIFF, .WMF	.PAL .ACT	.AIFF .AU .MP3 .WAV	.AVI .MOV	.DIR .FLA .FLC .FLI .GIF .PPT	.BMP	.AVI .MOV	.DIR .DXR .EXE

Macromedia Director Supported File Formats

Image File Formats

- Popular file formats
  - **GIF**

    One of the most important formats because of its historical connection to the WWW and HTML markup language as the first image type recognized by web browsers
  - **JPEG**

    Currently the most important common file format
  - **PNG**

    Subsumes GIF

GIF (Graphics Interchange Format)

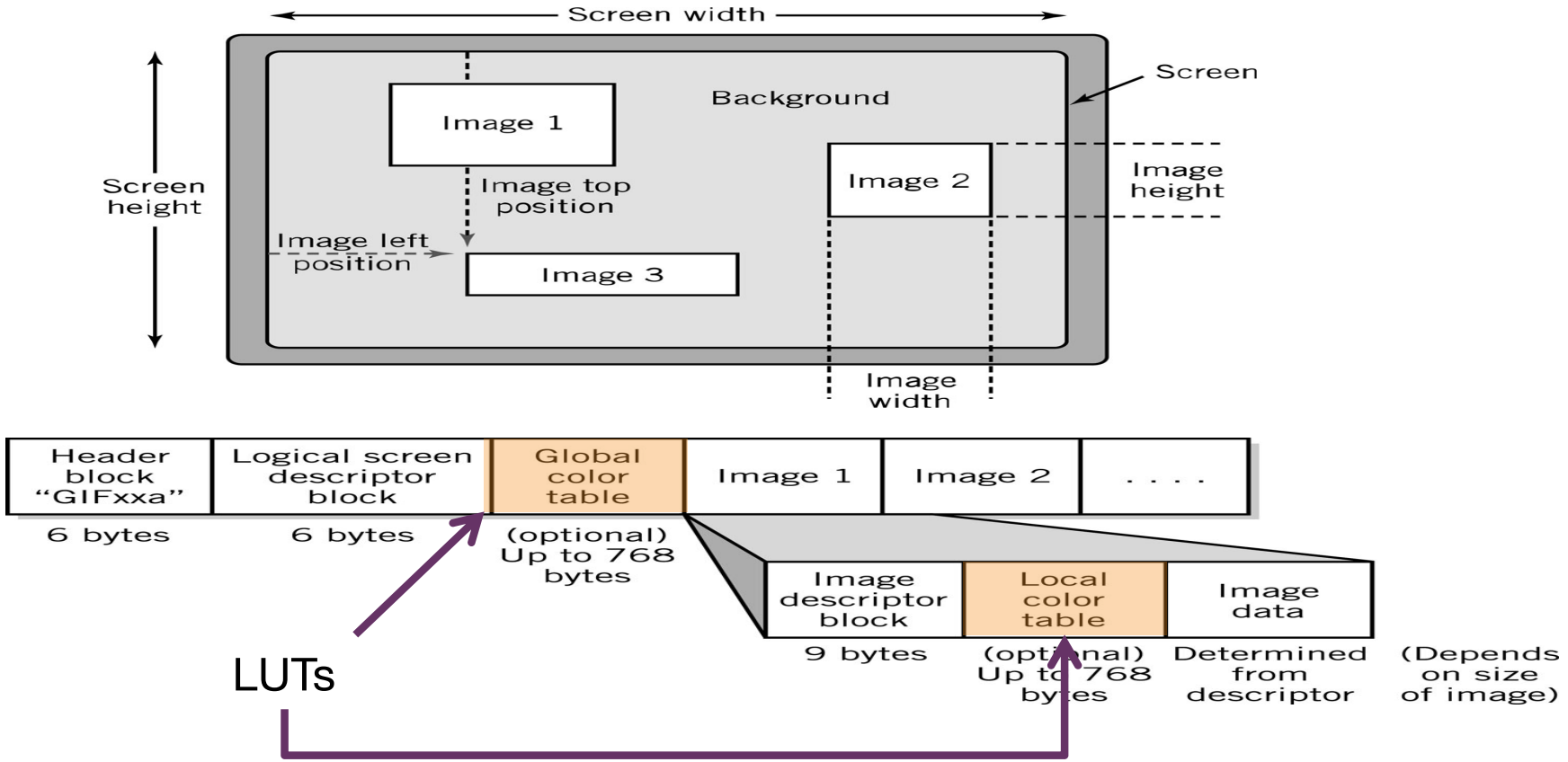
- Developed by CompuServe in 1987
- Most commonly used graphic file formats (especially on the Internet)
  - Limited to 256 colors → 8-bits used
  - Can store **multiple images** and the **controls** to make them appear as **real time animation** (e.g. delay time and transparency index)
  - Allows a special color to be used as "background" so image looks transparent
- **Lossless compression**
  - Reduce the file size without degrading the visual quality
- Preferred for line drawings, clip art and pictures with large blocks of solid color

GIF (Graphics Interchange Format)

- Note that is **not possible to convert** directly from a 24-bit file (ex. JPEG) to the GIF format
  - You need to convert a 24-bit image to Indexed Color mode first
  - Reduce the number of colors to a palette of 256 or less

    → Create a custom palette generated by the most commonly used color in the image

GIF (Graphics Interchange Format)



Animated GIF

Transparency GIF

- **One position** of the **color palette** is designated as “transparent”
- All pixels of the image that have this particular color index will be painted as transparent **when viewing**

Transparent GIF

Not a transparent GIF

JPEG (Joint Photographers Expert Group)

- Allows more than **16 million colors**
  - Suitable for highly detailed photographs and paintings
- Uses **"lossy" compression** to get more graphics into a smaller file
  - May reduce image resolution
  - Tends to distort sharp lines
- An image written to the JPEG format will be degraded by a **quality factor**
  - Quality factor **close to 100** → almost no degradation but compression is not significant
  - Quality factor **close to 0** → very small file but unrecognizable
  - A quality factor of **75** is usually a good compromise

JPEG Quality Factor

- Example

Q = 100

83,261 Kbytes

Q = 10

4,787 Kbytes

Q = 1

1,523 Kbytes

JPEG Degradation

- The image degradation caused by the JPEG format is cumulative
  - i.e., if you write an image to a JPEG file, and then read it from the JPEG file and write it back to the JPEG format, it will have suffered two passes of image degradation
- It works very, very badly on text, line art or other types of mechanical graphics
  - Degradation will be quite noticeably.
  - These sorts of graphics should be stored in another format (e.g. GIF or PNG)

Progressive JPEG

- There are two types of JPEG files :
  - **Sequential JPEG** stores its image as a simple bitmap
  - **Progressive JPEG** stores its image such that it can appear initially out of focus when it begins to download to a web page and resolve itself as more of the image is received by the web browser
- **Advantage**
  - Provide indication of the whole image to the viewer before the entire image is loaded
- **Disadvantage**
  - Require more computational power to display

GIF vs. JPEG

	GIF	JPEG
Best application	Line Art, Image with few color text	Photographs, Image with many colors
Display speed	Fast	Slower, more computation
Benefits	Transparency, Animation	Greatest compress for photographs, more color
Max. color	256	16.7 million

PNG (Portable Network Graphic)

- Created as an alternative to GIF
  - Lossless compression scheme is used
  - Does not support animation
- Support three image type: *true color*, *gray scale*, *palette-based*
  - JPEG supports the first 2.
  - GIF supports the 3rd one.

?

Quiz

- Why we need to be able to have less that 24-bit color and why this makes for a problem ?
- What do we need to do to adaptively transform a 24-bit image into a 8-bit one?

Quiz

- Suppose we decide to quantize an 8-bit gray scale image down to just 2 bits of accuracy.
- What is the simplest way to do so?
- What ranges of byte values in the original image are mapped to quantized values?

Quiz

- Suppose we have available 24 bits per pixel for a color image. However we notice that humans are more sensitive to R and G than to B – (1.5 times more sensitive).
- How could we best make use of the bits available?

Quiz

- At your job, you have decided to impress the boss by using up more disk space for the company's gray scale images. Instead of using 8 bits per pixel, you'd like to use 48 bits per pixel in RGB.
- How could you store the original gray scale images so that in the new format they would appear the same as they used to, visually?