Data Representation

Interpreting bits to give them meaning

Part 3: Media - Text and Pictures

Data is more than just numbers!

Data is stored using bits but represents many things:

- Documents
- Pictures
- Sound/music
- Video
- ...

How does this work?

- **File formats**: Structure bits in such a way that mapping between bits and what they represent is unambiguous
  - Standardized or open file formats
    - Specified so that anyone can write programs for them (JPEG, MPEG, and MP3), OpenDocument, HTML, ...
  - "Open" and "standardized" doesn’t mean "free" (MP3, GIF, …)
- A **data capture** or creation program builds the file in the appropriate format
- A **rendering** program converts the file format to a recognizable form (image viewer, web browser, video player, …)

Reminders

**Blown to Bits** reading

- Chapter 4 - Reflection due Wednesday at 10:00am

Homework 3 - due Wednesday, Oct. 29

- Should have picked a topic (a computing innovation)
- Start doing research and making notes!

Upcoming:

- Lab 10 will be Friday
- Project: Have an informal idea and perhaps a team by Friday
Representations of Text

**ASCII**

When everything is 0's and 1's, how do you store or transmit something like “Hello World”?

Answer: Encode characters as binary strings

In early days there were several "encodings"

Most common for basic US/English use is ASCII

- **American Standard Code for Information Interchange**
- Uses 7 bits per character
- Typically embedded in 8-bit bytes
- Hexadecimal bytes → ASCII examples to the right

Less U.S.-centric encoding: Unicode

---

### Representations of Text - What does the highlighted part say?

In early days there were several "encodings"

- **Encoded characters as binary strings**
- **American Standard Code for Information Interchange**
- Uses 7 bits per character
- Typically embedded in 8-bit bytes
- Hexadecimal bytes → ASCII

---

### Representations of Text - The full hex dump!

ASCII - The full hex dump!

- ASCII - What does the highlighted part say?
- ASCII - The full hex dump!

---

### Representations of Text

**ASCII**

When everything is 0's and 1's, how do you store or transmit something like “Hello World”?

Answer: Encode characters as binary strings

In early days there were several "encodings"

Most common for basic US/English use is ASCII

- **American Standard Code for Information Interchange**
- Uses 7 bits per character
- Typically embedded in 8-bit bytes
- Hexadecimal bytes → ASCII examples to the right

Less U.S.-centric encoding: Unicode
ASCII provides letters - what about fonts, sizes, etc?

One option: HTML - HyperText Markup Language
- The "language of web pages"
- "Markup" indicates formatting/style
- All characters are just regular character set (like ASCII) - including markup
- Must be rendered to convert character based markup to formatted text
- A lot of formatting is now in CSS - Cascading Style Sheets
- Much more involved than these examples!

**HTML Source**

This is formatted text, which can be `<b>bold</b>` or `<i>italic</i>` or `<u>underlined</u>` or `<span style="font-size: 150%"/>` or `<span style="font-size: 50%"/>` or `<small/>` or ...

**Rendered Text**

This is formatted text, which can be bold or italic or underlined or big or ... or ...

Pictures

Grayscale

Grayscale images have levels of intensity, but no color
- More information than bi-tonal black and white (like fax machines or most printers)
- Less information than color

Pictures

Grayscale - Pixels

Pixels are "picture elements"

Resolution is pixel density
- Can be in dots/peels per inch (dpi/ppi)
  - Typical monitor: 100ppi
  - Typical printer: 600dpi (bi-tonal)
  - Quality depends on viewing distance (32" high def TV is only 43 ppi - but you don't sit right next to it)
  - Apple "retina display" - 326 ppi on iPhone
Pictures

Grayscale - Pixels as numbers

<table>
<thead>
<tr>
<th>Number of levels</th>
<th>Typically one byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>79 6F 75 B4 E6</td>
<td></td>
</tr>
<tr>
<td>1F 86 98 94 B2</td>
<td></td>
</tr>
<tr>
<td>58 60 42 82 8B</td>
<td></td>
</tr>
<tr>
<td>4D 51 51 4B 37</td>
<td></td>
</tr>
<tr>
<td>35 45 2C 1B 1A</td>
<td></td>
</tr>
</tbody>
</table>

Pictures

Color - Three "color planes" (red, green, blue)

Each color plane is just like a grayscale image, with same issues:
- Resolution (ppi)
- Depth (bpp)

"24 bit color" means 8 bits per pixel in each of the 3 colors.

Pictures

Why does this work?

"Rods" and "cones" signal our brain about light we receive in our eye
- Rods: monochrome only
- Cones: Color - in three kinds, red, green, and blue
Pictures

*Why does this work?*

Bottom line: If humans can only perceive three colors (red, green, and blue) then reconstructing just those three colors allow us to *perceive* everything just as in an original.

Interesting question: What if someone were born with a mutation that gave them purple and yellow receptors?

Summary of Part 3

Files just store bits
- Bits are bits: no different for text or images or ...
- Rendering program makes all the difference
- Text - encodings defined in standards
  - ASCII, Unicode, HTML
- Image formats take advantage of biology
  - Images aren’t “accurate” but we perceive them that way