Recall: Info Representation

- Digitization: representing information by any fixed set of symbols
  - decide how many different items of information you want to represent
    - Tic Tac Toe: 3 items - empty cell or player 1 or player 2
  - decide how many "digits" or positions you want to use
    - Tic Tac Toe: 9 positions - one per board square
  - decide on a set of symbols
    - player 1: \( \times \)
    - empty cell: \( \otimes \)
    - player 2: \( \bigcirc \)

Empty position: \( \otimes \)

- We can represent this game as one 9-digit string:
  \[ \bigcirc \bigotimes \bigtimes \bigtimes \bigcirc \bigotimes \bigtimes \]

- How many possible game states are there?
  \[ 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 \times 3 = 3^9 = 19683 \]
Another encoding

use a different set of symbols
- empty cell: 0
- player 1: 1
- player 2: 2

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

We can represent this game as one 9-digit number:
2 0 0 1 1 2 0 0 0

How many possible game states are there?
» 3 x 3 x 3 x 3 x 3 x 3 = 3^9 = 19683

Info Representation

- Adult humans have 32 teeth
  » sometimes a tooth or two is missing!
- How can we represent a set of teeth?
  » How many different items of information?
    » 2 items - tooth or no tooth
  » How many "digits" or positions to use?
    » 32 positions - one per tooth socket
  » Choose a set of symbols
    no tooth: 0    tooth: 1

Info Representation

- Color monitors combine light from Red, Green, and Blue phosphors to show us colors
- How can we represent a particular color?
  » How many different items of information?
    » 256 items - distinguish 256 levels of brightness
  » How many "digits" or positions to use?
    » 3 positions - one Red, one Green, one Blue
  » Choose a set of symbols
    brightness level represented by the numbers 0 to 255

What's your tooth number?

incisors  canines  pre-molars  molars
0000 0000 0000 0000 0000 0000 0000 0000

no teeth ↔ 0000 0000 0000 0000 0000 0000 0000 0000

1111 1111 1111 1111 1111 1111 0000 0000 0000 0000

no molars ↔ 1111 1111 1111 1111 1111 1111 0000 0000 0000

How many possible combinations? 2 x 2 x 2 x ... x 2 = 2^32 = 4 Billion
What is the pixel's color?

<table>
<thead>
<tr>
<th>red</th>
<th>green</th>
<th>blue</th>
</tr>
</thead>
<tbody>
<tr>
<td>255</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>255</td>
<td>102</td>
<td>0</td>
</tr>
<tr>
<td>128</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

How many possible combinations?

\[256 \times 256 \times 256 = 256^3 = 16 \text{ Million}\]

16 M colors is often called "True Color"

How can we store numbers?

- We want to store numbers
  - 0 to 255 for color brightness
  - 0 to 4B for tooth configuration
  - 0 to 255 for ASCII character codes
- What do we have available in memory?
  - **Binary digits**
    - 0 or 1
    - on or off
    - clockwise or counter-clockwise

\[\ldots00100111000010010001001000100001001\ldots\]

The hardware is binary

- 0 and 1 are the only symbols the computer can actually store directly in memory
  - a single bit is either off or on
- How many numbers can we represent with 0 and 1?
  - How many different items of information?
    - 2 items - off or on
  - How many "digits" or positions to use?
    - let's think about that on the next slide
  - Choose a set of symbols
    - already chosen: 0 and 1

How many positions should we use?

It depends: how many numbers do we need?

\[
\begin{array}{ccc}
\text{one position} & \text{two positions} & \text{three positions} \\
\{0\} & \{00\} & \{000\} \\
\{1\} & \{01\} & \{010\} \\
\end{array}
\]

\[
\begin{array}{ccc}
\text{four numbers} & \text{eight numbers} \\
\{00\} & \{000\} \\
\{01\} & \{010\} \\
\{10\} & \{101\} \\
\{11\} & \{110\} \\
\end{array}
\]
The sky's the limit

- We can get as many numbers as we need by allocating enough positions
  » each additional position means that we get \textit{twice} as many values because we can represent \textit{two} numbers in each position
  » these are base 2 or \textit{binary} numbers
  - each position can represent two different values
- How many different numbers can we represent in base 2 using 4 positions?

How can we read binary numbers?

Let's look at the equivalent \textit{decimal} (ie, base 10) numbers.

<table>
<thead>
<tr>
<th>Binary (base 2)</th>
<th>Decimal (base 10)</th>
<th>Binary (base 2)</th>
<th>Decimal (base 10)</th>
<th>Binary (base 2)</th>
<th>Decimal (base 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>0000</td>
<td>0</td>
<td>0000</td>
<td>0</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>0010</td>
<td>2</td>
<td>0100</td>
<td>4</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
<td>0101</td>
<td>2</td>
<td>0110</td>
<td>6</td>
</tr>
<tr>
<td>0110</td>
<td>6</td>
<td>1100</td>
<td>12</td>
<td>1100</td>
<td>12</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
<td>1110</td>
<td>14</td>
<td>1111</td>
<td>16</td>
</tr>
<tr>
<td>1010</td>
<td>10</td>
<td>1001</td>
<td>9</td>
<td>1001</td>
<td>9</td>
</tr>
<tr>
<td>1011</td>
<td>11</td>
<td>1010</td>
<td>10</td>
<td>1010</td>
<td>10</td>
</tr>
<tr>
<td>1100</td>
<td>12</td>
<td>1101</td>
<td>11</td>
<td>1101</td>
<td>11</td>
</tr>
<tr>
<td>1110</td>
<td>14</td>
<td>1110</td>
<td>16</td>
<td>1111</td>
<td>16</td>
</tr>
<tr>
<td>1111</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1112 represents \textit{exactly the same quantity} as 7_{10}

They are just different ways of representing the same number.

Position matters!

<table>
<thead>
<tr>
<th>Binary (base 2)</th>
<th>Decimal (base 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
</tr>
<tr>
<td>0110</td>
<td>6</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
</tr>
<tr>
<td>1010</td>
<td>10</td>
</tr>
<tr>
<td>1100</td>
<td>12</td>
</tr>
<tr>
<td>1110</td>
<td>14</td>
</tr>
<tr>
<td>1111</td>
<td>16</td>
</tr>
</tbody>
</table>

Each position represents one more multiplication by the base value.
For binary numbers, the base value is 2, so each new column represents a multiplication by 2.

What base 10 decimal value is equivalent to the base 2 binary value 100010102 shown above?

<table>
<thead>
<tr>
<th>Base 10</th>
<th>Base 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000</td>
<td>1001</td>
</tr>
<tr>
<td>1000</td>
<td>1010</td>
</tr>
<tr>
<td>100</td>
<td>1101</td>
</tr>
<tr>
<td>10</td>
<td>1110</td>
</tr>
<tr>
<td>1</td>
<td>1111</td>
</tr>
</tbody>
</table>

What base 10 decimal value is equivalent to the base 2 binary value 100010102 shown above?
Some Examples

<table>
<thead>
<tr>
<th>$2^7$</th>
<th>$2^6$</th>
<th>$2^5$</th>
<th>$2^4$</th>
<th>$2^3$</th>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>64</td>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

This is an old and very important idea

- “You see, more than 5000 years ago, the Babylonians—and probably the Sumerians before them—had the idea of positional notation for numbers. They mostly used base 60—not base 10—which is actually presumably where our hours, minutes, seconds scheme comes from. But they had the idea of using the same digits to represent multiples of different powers of 60."

- “Well, this fine abstract Babylonian scheme for doing things was almost forgotten for nearly 3000 years. And instead, what mostly was used, I suspect, were more natural-language-based schemes, where there were different symbols for tens, hundreds, etc.”

- Quoted from Mathematical Notation: Past and Future Keynote address presented by Stephen Wolfram at MathML and Math on the Web: MathML International Conference 2000