CLPS Matlab Programming Workshop

Session 2 of 3

Written by Jae-Young Son

December 5, 2018

Roadmap

- **Session 1: A programmer's essential toolkit**
  - What is information processing?
  - Variables
  - Functions
  - Arrays and matrices
  - Control logic
- **Session 2: Computer graphics**
  - Shapes
  - Animation
  - Placement of objects in space
  - Layering
  - Images
- **Session 3: Putting it all together**
  - Monitoring and recording user input (from keyboards and mice, and getting RTs)
  - Nesting functions
  - Simple economic experiment

PsychToolbox

- Library of functions that are useful for accomplishing common tasks in psychology
  - Displaying shapes, textures, images, and videos
  - Precise stimulus presentation times
  - Control over keyboard input / precise reaction times
  - Making a psych experiment *look* like an experiment, and not like the control console of the Apollo spaceship

Margaret Hamilton standing next to the Apollo 11 source code, which she and her team at MIT played a fundamental role in developing

1.1 – Calling the PsychToolbox screen

```
% Skip screen calibration tests! Comment out if this is important to you!
Screen('Preference', 'SkipSyncTests', 1);

% VisualDebugLevel 0 turns off most error warnings! Change to 3 for default setting
Screen('Preference', 'VisualDebugLevel', 0);

% Turns off all warnings! Comment out if warnings are important to you!
Screen('Preference', 'SuppressAllWarnings', 1);

% Returns an array of screen numbers
screen = Screen('Screens');

% Actually opens screen
[wPtr, screenres] = Screen('OpenWindow', screen(1));

% Close screen
KbStrokeWait;
Screen('CloseAll');
```

1.1 – Calling the PsychToolbox screen

```
% Skip screen calibration
tests! Comment out if this is
important to you!

Screen('Preference',
'SkipSyncTests', 1);


% VisualDebugLevel 0 turns off
most error warnings! Change to
3 for default setting

Screen('Preference',
'VisualDebugLevel', 0);


% Turns off all warnings!
Comment out if warnings are
important to you!

Screen('Preference',
'SuppressAllWarnings', 1);
```

When you first call the PTB screen function, it normally runs a bunch of diagnostics on your computer, such as making sure that your monitor is syncing correctly with your computer's internal clock

PTB was developed by people who do psychophysics experiments, where stimulus presentation timing *really* matters – if you don't care as much about accurate stimulus presentation times, you can turn off these diagnostics and warnings

1.1 – Calling the PsychToolbox screen

```
% Returns an array of
screen numbers

screen =
Screen('Screens');
```

```
% Actually opens screen

[wPtr, screenres] =
Screen('OpenWindow',
screen(1));
```

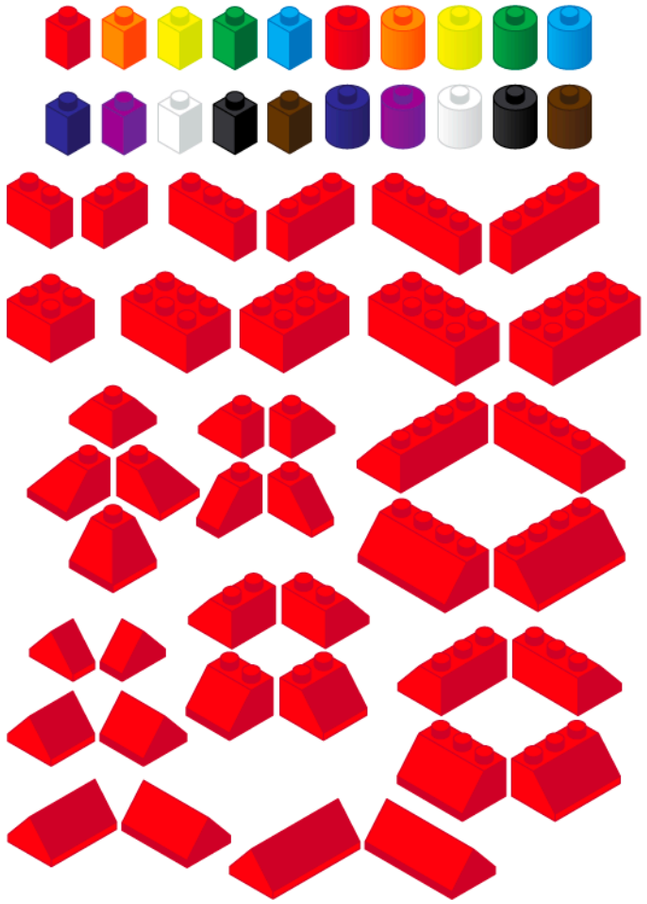
The command **Screen('Screens')** checks to see what displays are connected to the computer. If there are multiple monitors connected to your computer, you can tell Matlab which monitor to use by specifying **screen(x)**.

Once you open the screen, you should create the variables **wPtr** and **screenres**.

wPtr stands for "window pointer" – this is important because you have to tell Matlab which window to draw things in.

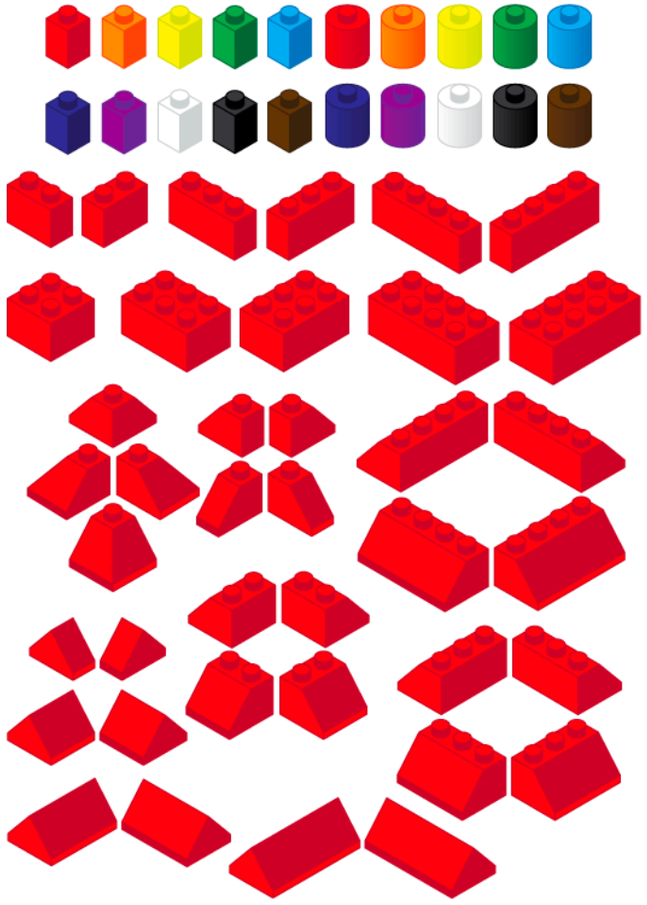
screenres records the dimensions of your screen.

1.2 – Shapes

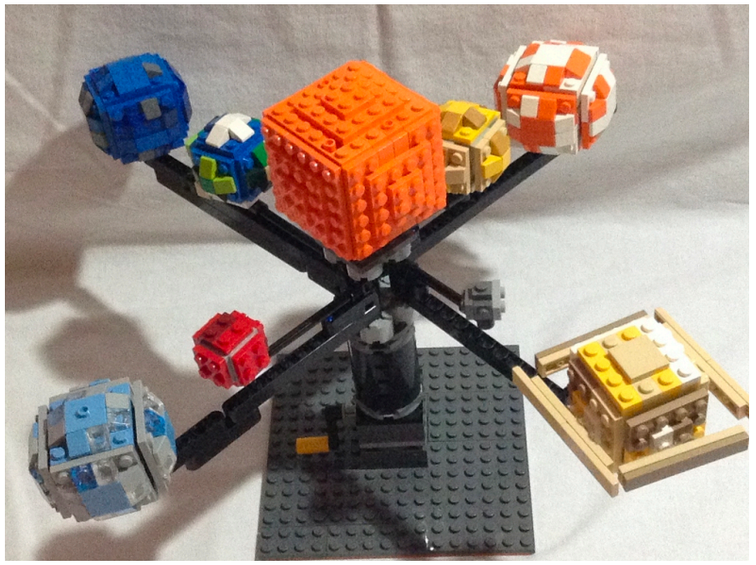


From a small set of pre-defined shapes, you can create almost anything you want!

1.2 – Shapes

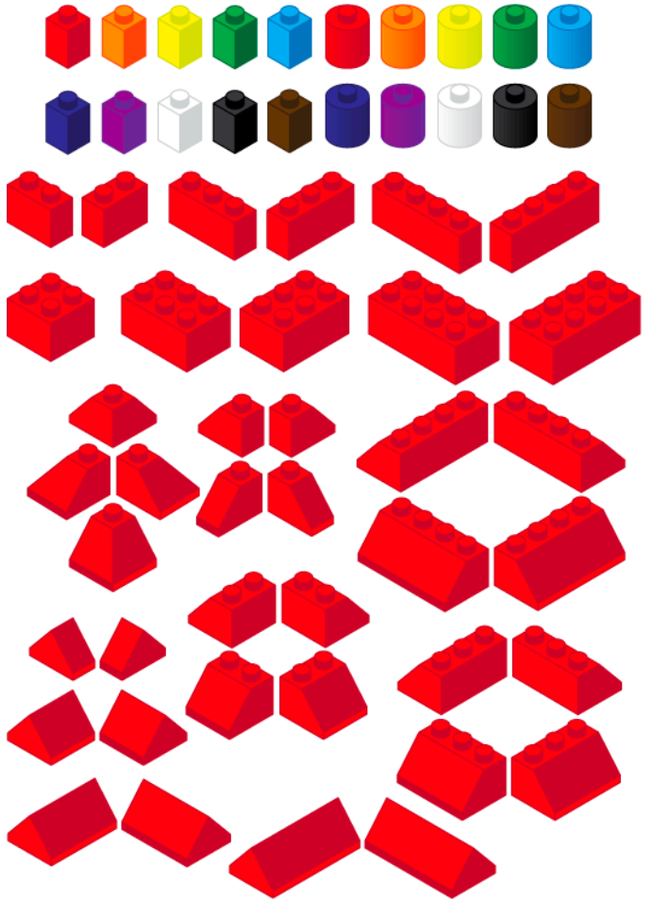


From a small set of pre-defined shapes, you can create almost anything you want!

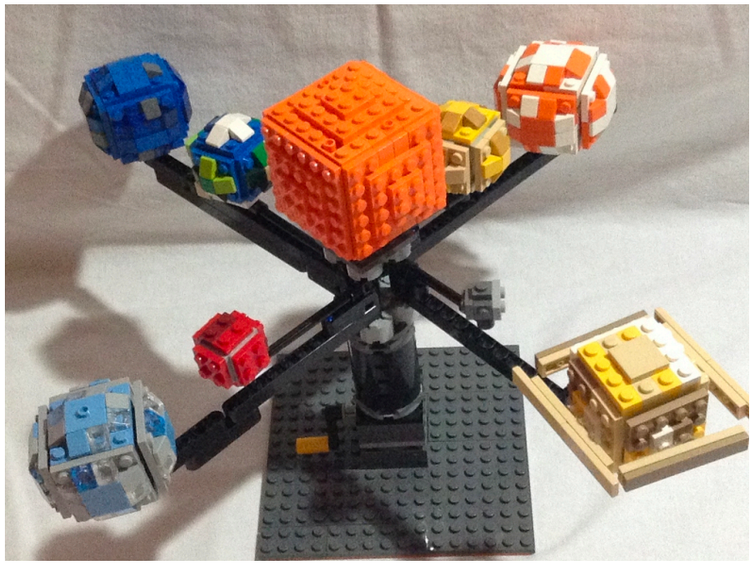


You can use the same basic shapes from a small model of the solar system…

1.2 – Shapes



From a small set of pre-defined shapes, you can create almost anything you want!



You can use the same basic shapes from a small model of the solar system…

…to build a fully-functional (but probably not street legal) Bugatti!

1.2 – Shapes

1.2 – Shapes

```
% Define colors

black = BlackIndex(wPtr);

white = WhiteIndex(wPtr);


% Fill entire screen with black

Screen('FillRect', wPtr, black);

Screen(wPtr, 'Flip');
```

1.2 – Shapes

```
% Define colors

black = BlackIndex(wPtr);

white = WhiteIndex(wPtr);


% Fill entire screen with black

Screen('FillRect', wPtr, black);

Screen(wPtr, 'Flip');
```

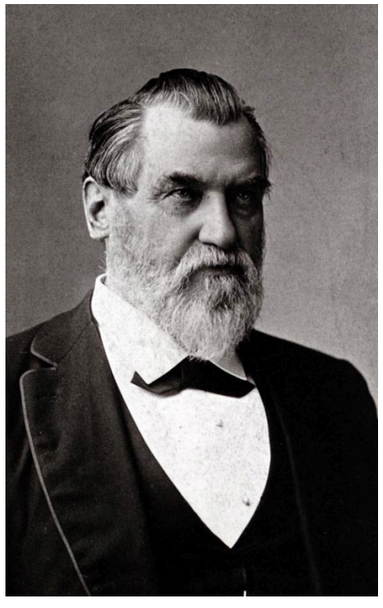
1.2 – Shapes

```
% Define colors

black = BlackIndex(wPtr);

white = WhiteIndex(wPtr);


% Fill entire screen with black

Screen('FillRect', wPtr, black);

Screen(wPtr, 'Flip');
```

In order to ***display*** a graphic, you have to ***draw*** it first

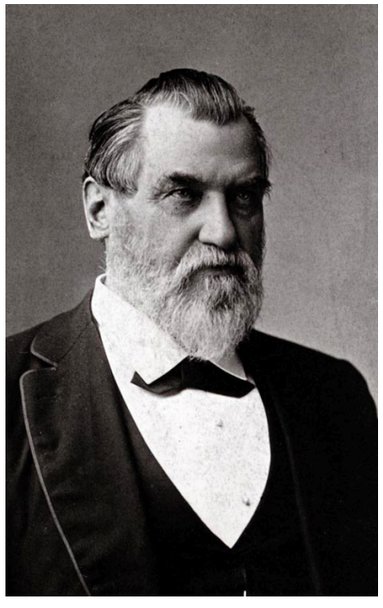
1.3 – Animation

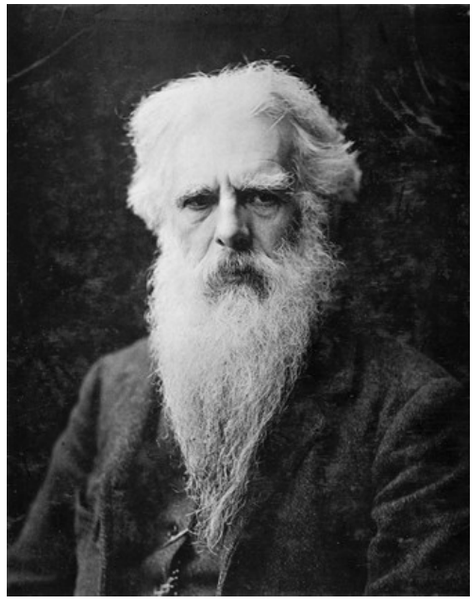
1.3 – Animation



Leland Stanford:
“Yo my robber baron friends got into an argument about whether all four of a horse’s hooves leave the ground while galloping”

1.3 – Animation

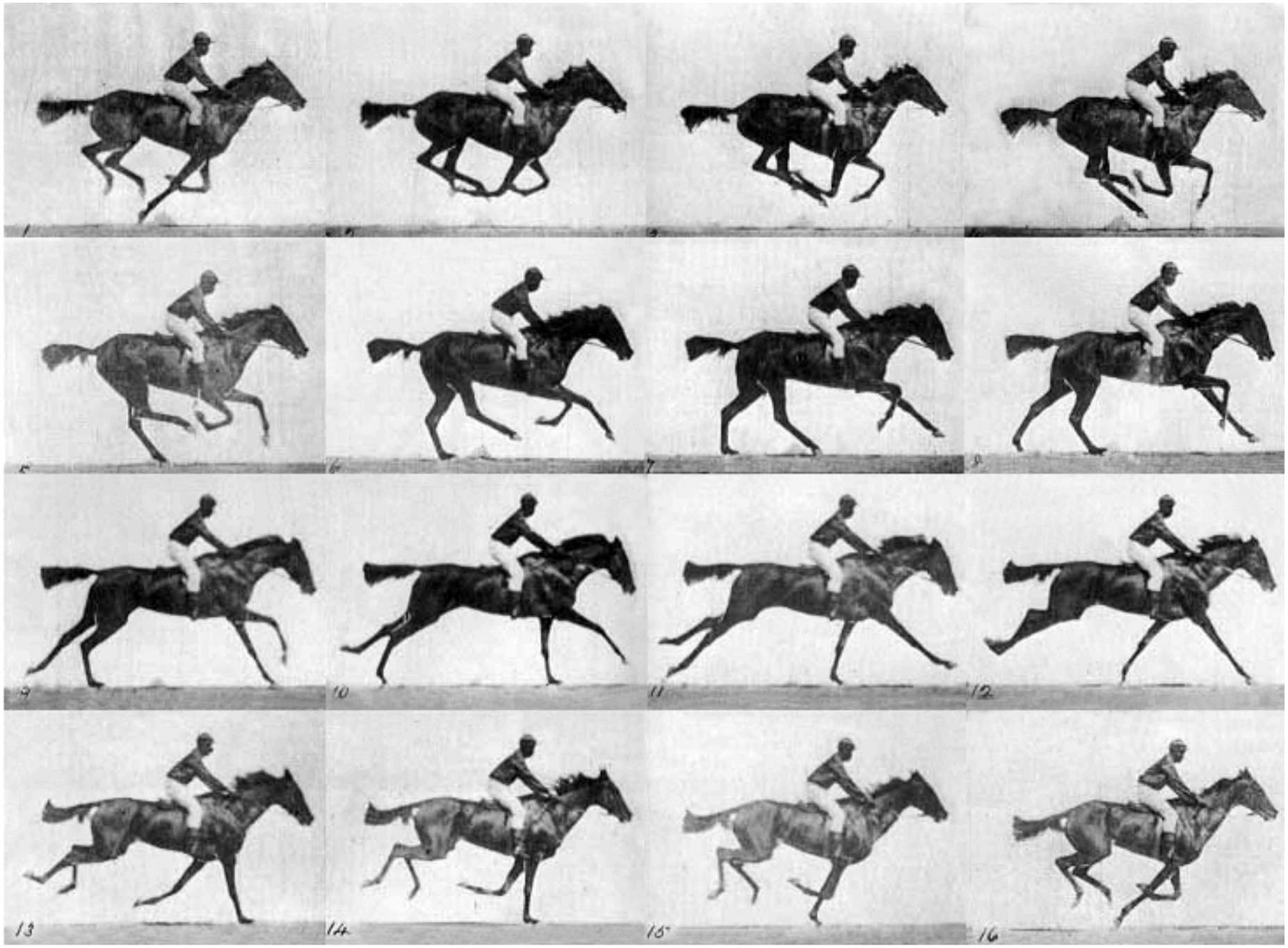




Leland Stanford:
“Yo my robber baron friends got into an argument about whether all four of a horse’s hooves leave the ground while galloping”

Eadweard Muybridge:
“Say no more my man”

1.3 – Animation



1.3 – Animation

1.3 – Animation

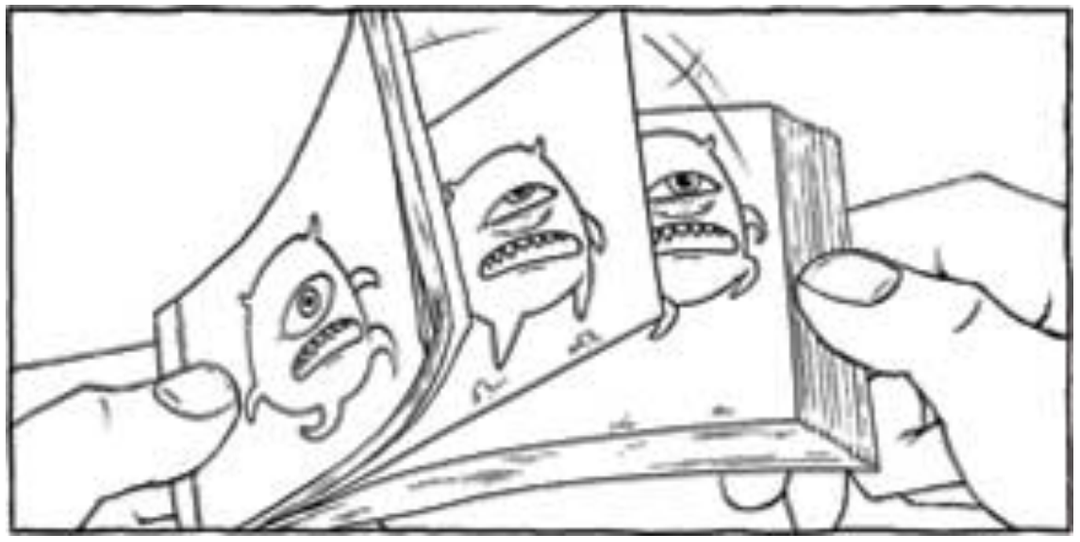
1.3 – Animation: Drawing vs Displaying

1.3 – Animation: Drawing vs Displaying

- In order to ***display*** a graphic, you have to ***draw*** it first

1.3 – Animation: Drawing vs Displaying

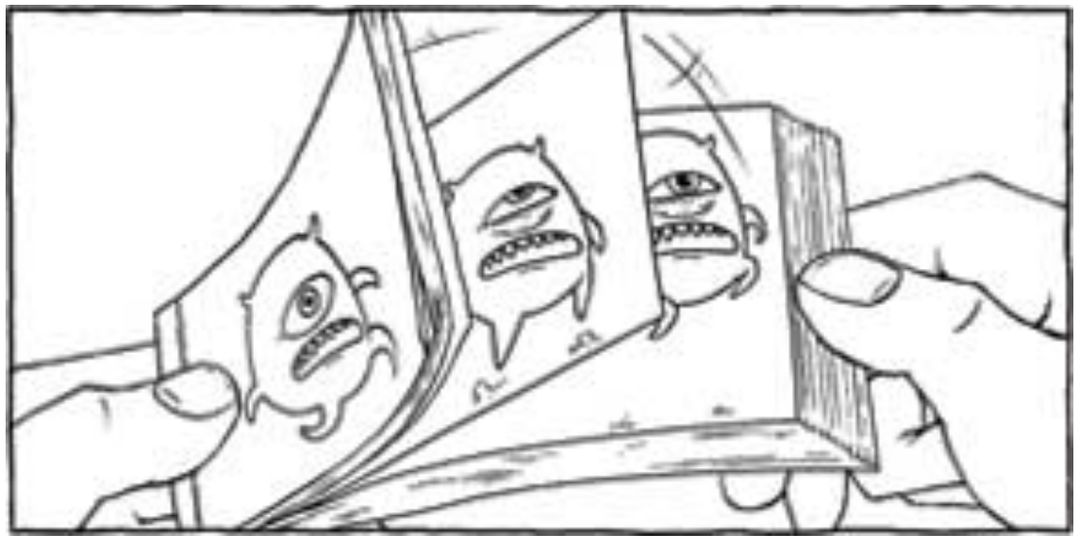
- In order to ***display*** a graphic, you have to ***draw*** it first



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

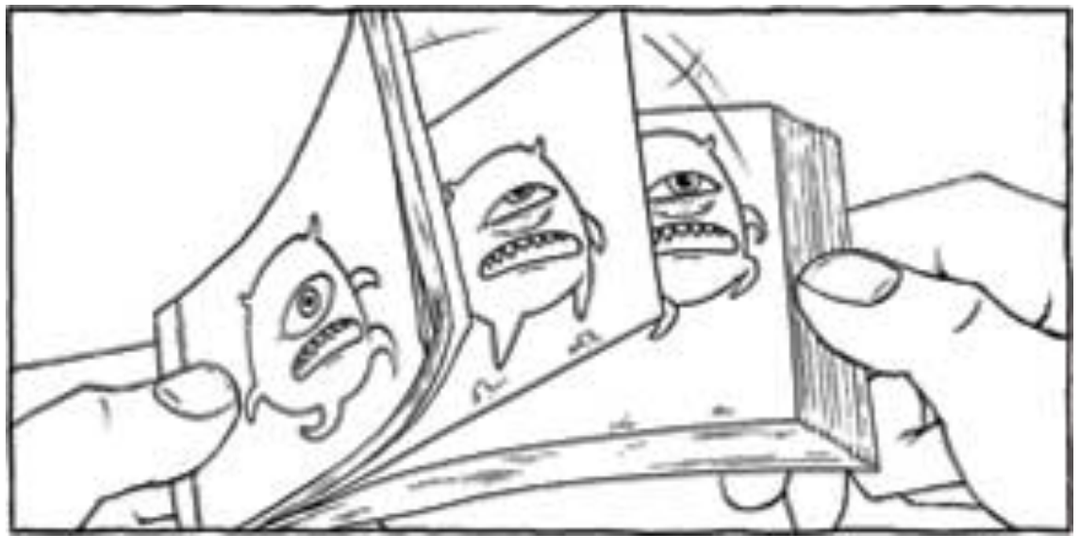
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

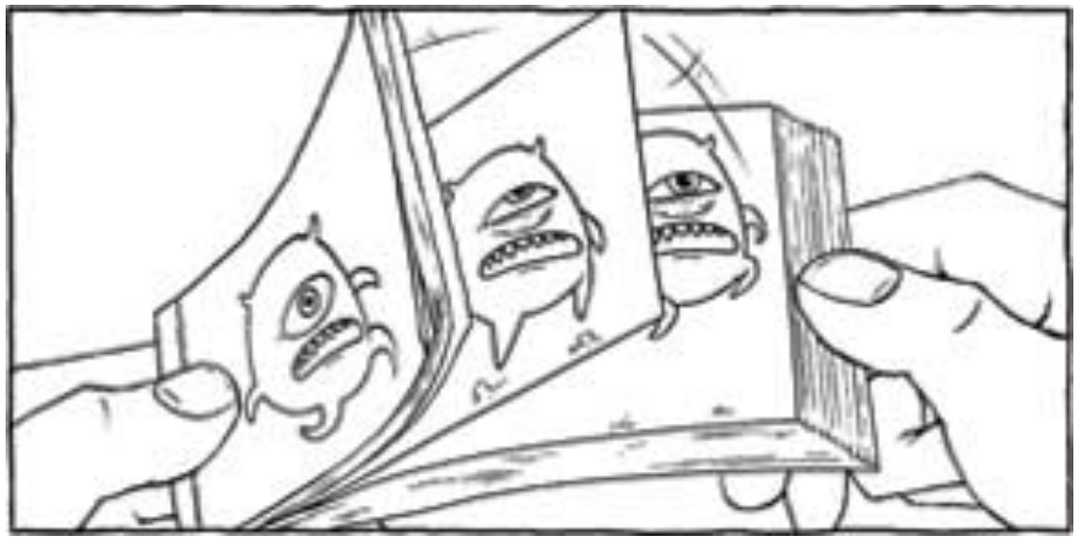
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

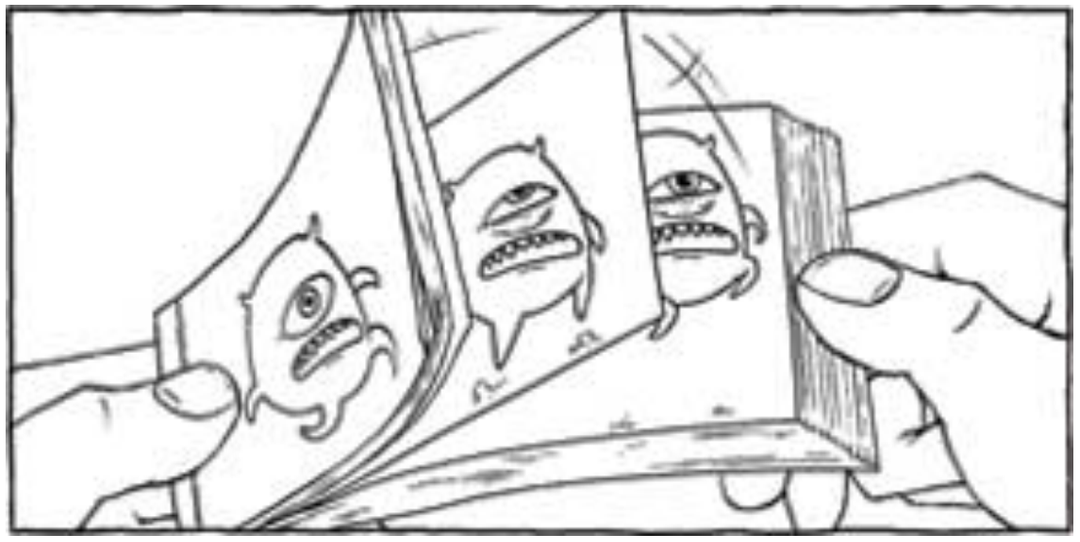
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

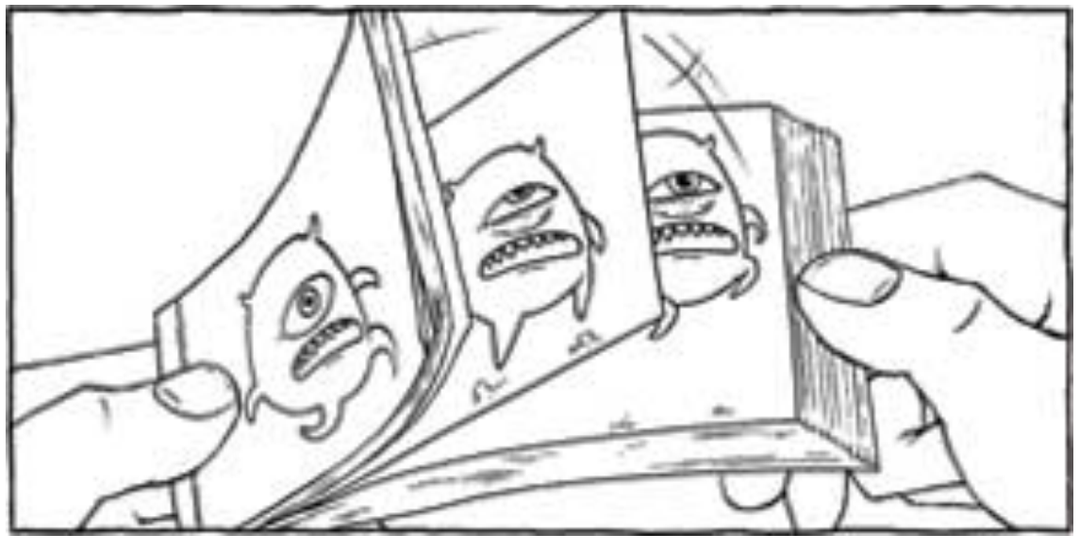
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

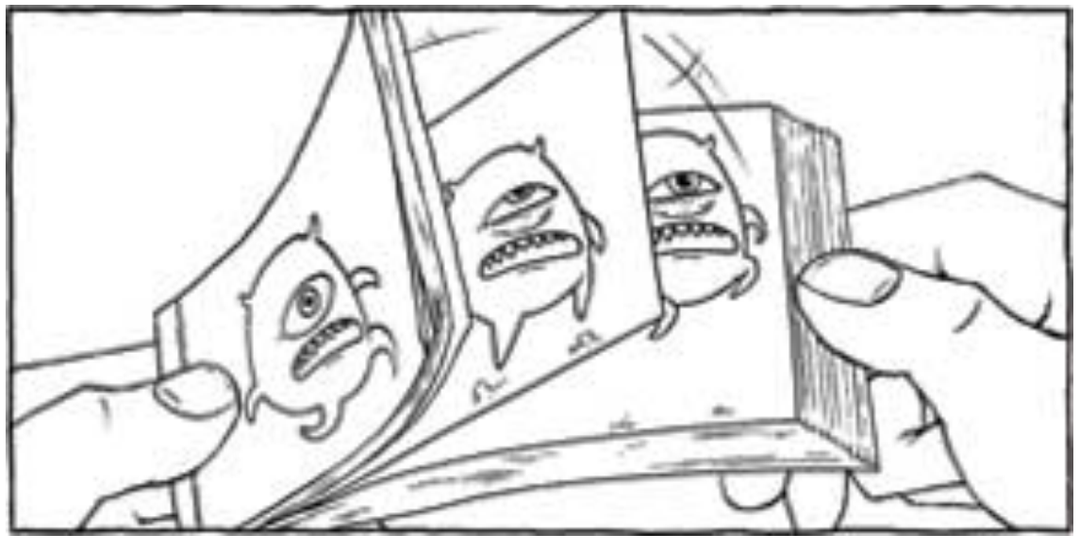
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:
  - "Draw" a single frame



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

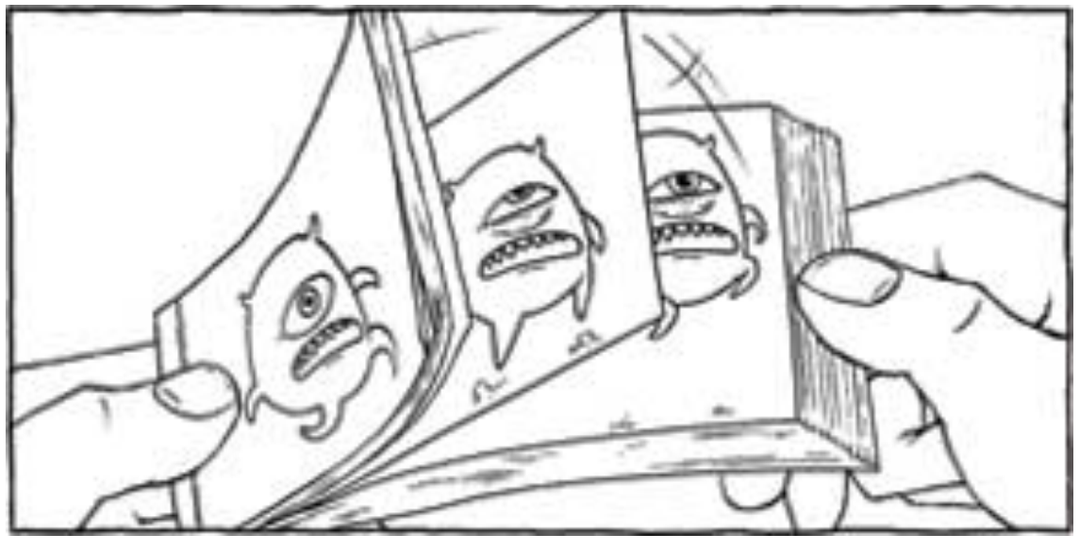
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:
  - "Draw" a single frame
  - Temporarily hold it in memory until it's ready to be deployed



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

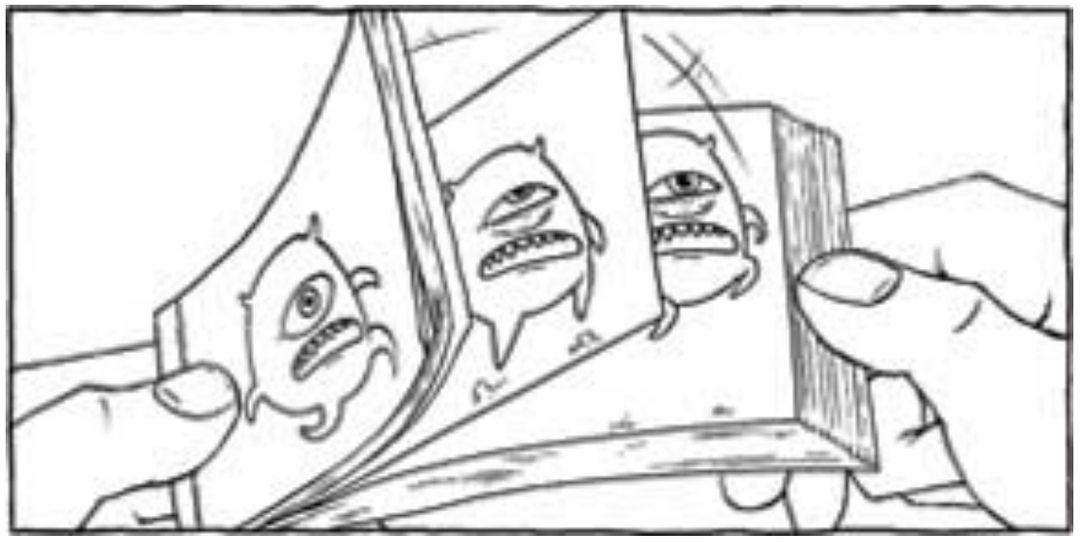
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:
  - "Draw" a single frame
  - Temporarily hold it in memory until it's ready to be deployed
  - Display it once the preceding frame disappears



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

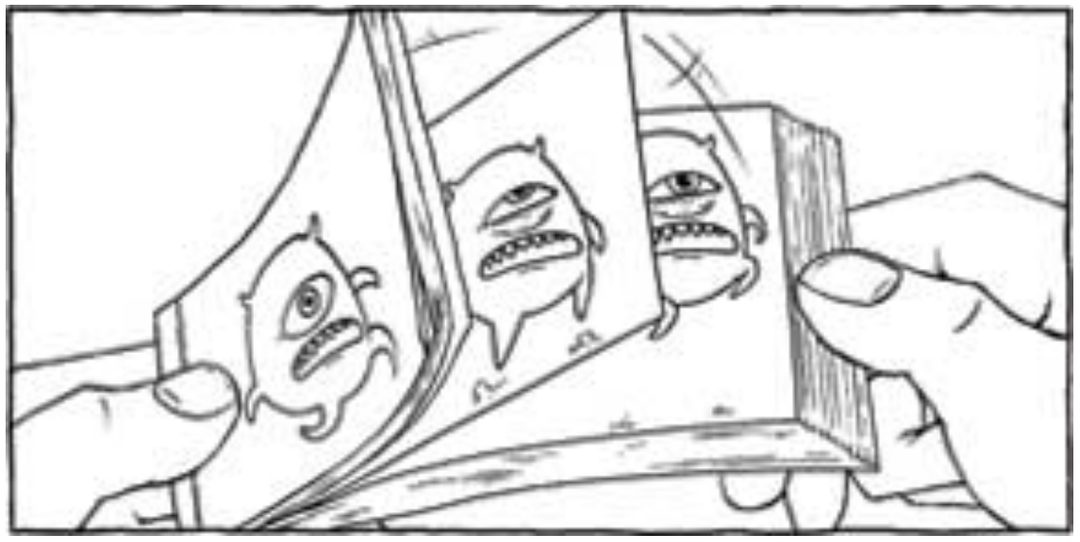
- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:
  - "Draw" a single frame
  - Temporarily hold it in memory until it's ready to be deployed
  - Display it once the preceding frame disappears
  - Delete it from memory once it's been displayed



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

- In order to ***display*** a graphic, you have to ***draw*** it first
- Strategy 1 for animating images:
  - Draw and display each frame of the horse running in real-time
  - Extreme computational costs – impractical
- Strategy 2 for animating images:
  - "Draw" a single frame
  - Temporarily hold it in memory until it's ready to be deployed
  - Display it once the preceding frame disappears
  - Delete it from memory once it's been displayed
  - Always a "buffer" of images held in memory waiting to be deployed



Muybridge's 1872 horse photographs took advantage of the invention of the flipbook in 1868 by John Barnes Linnett – for historical context, this was nearly ten years AFTER Darwin published "On the Origin of Species"

1.3 – Animation: Drawing vs Displaying

1.3 – Animation: Drawing vs Displaying

- PsychToolbox has to be told what to display on the screen

1.3 – Animation: Drawing vs Displaying

- PsychToolbox has to be told what to display on the screen
- But in order to display things on the screen, you have to draw them first

1.3 – Animation: Drawing vs Displaying

- PsychToolbox has to be told what to display on the screen
- But in order to display things on the screen, you have to draw them first
- This is why there are separate commands for drawing visual objects and displaying them

1.3 – Animation: Drawing vs Displaying

- PsychToolbox has to be told what to display on the screen
- But in order to display things on the screen, you have to draw them first
- This is why there are separate commands for drawing visual objects and displaying them
- This is also why the command to display objects is **Screen(wPtr, 'Flip');**

1.4 – Placing objects in space

1.4 – Placing objects in space

- Pixels are the fundamental building blocks of computer graphics

1.4 – Placing objects in space

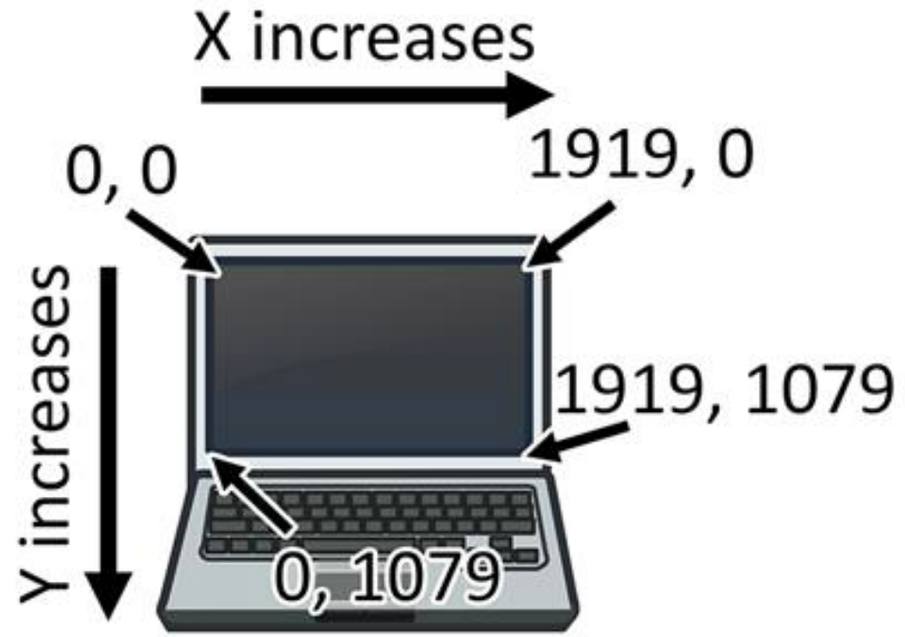
- Pixels are the fundamental building blocks of computer graphics
- Each point on your computer monitor has a single X and Y coordinate

1.4 – Placing objects in space

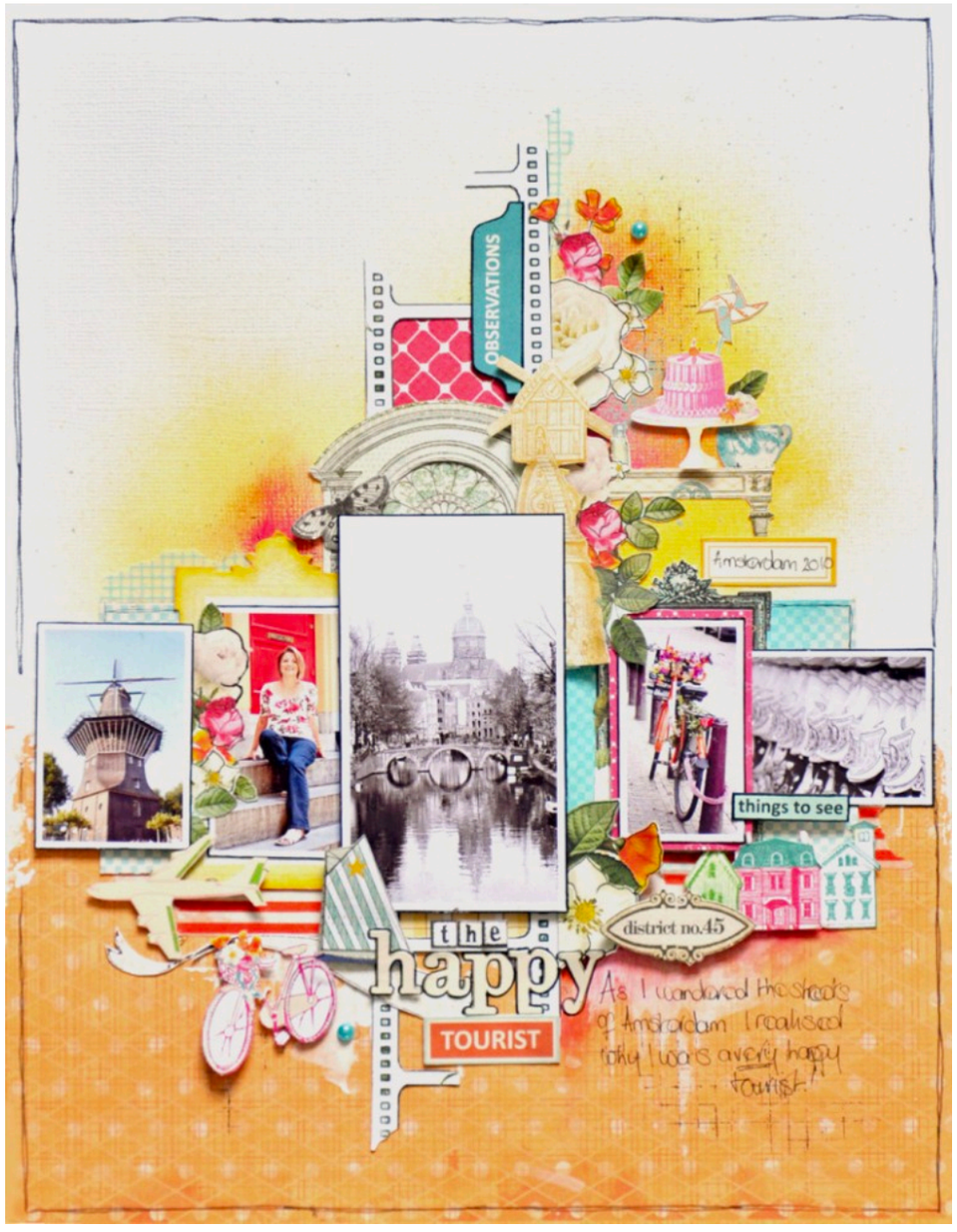
- Pixels are the fundamental building blocks of computer graphics
- Each point on your computer monitor has a single X and Y coordinate
- If you've ever played Battleship, placing pixels on the screen is exactly like placing ships on the grid

1.4 – Placing objects in space

```
% Define screen dimensions
bottom = screenres(4);
right = screenres(3);
xCenter = right/2;
yCenter = bottom/2;

% Black background
Screen('FillRect', wPtr, black);

% White centered square
rectSize = [0 0 200 200];
rectCentered = CenterRectOnPointd(rectSize, xCenter, yCenter);
rectColor = white;
Screen('FillRect', wPtr, rectColor, rectCentered);

% Black centered text
DrawFormattedText(wPtr, 'Hello!', 'center', 'center', black);

% Display
DrawFormattedText(wPtr, 'Flip');
```



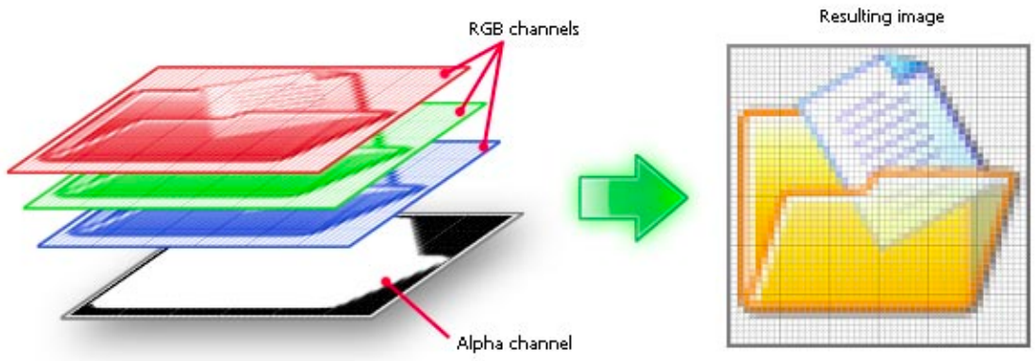
1.5 – Layering



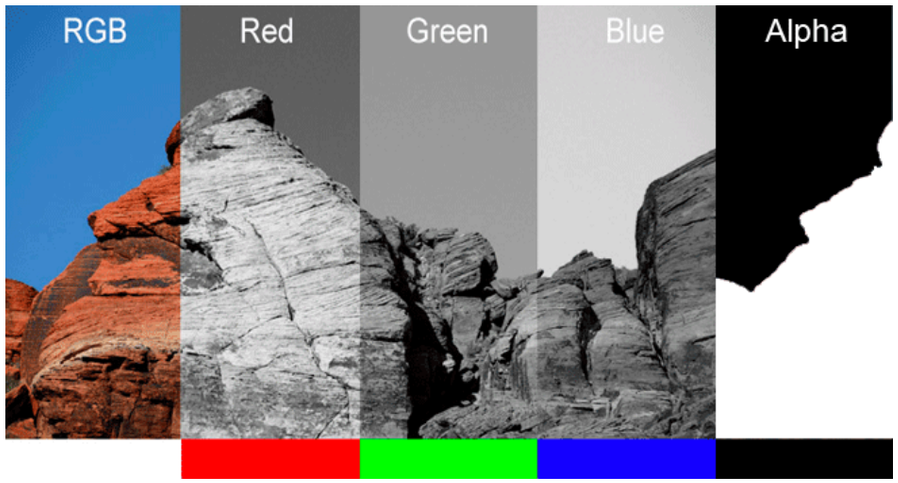
In short: the order in which you draw objects matters because that's the order in which PsychToolbox will layer your graphical objects if they're overlapping!

In this way, layering objects in space is analogous to layering photographs inside a scrapbook.

1.5 – Images

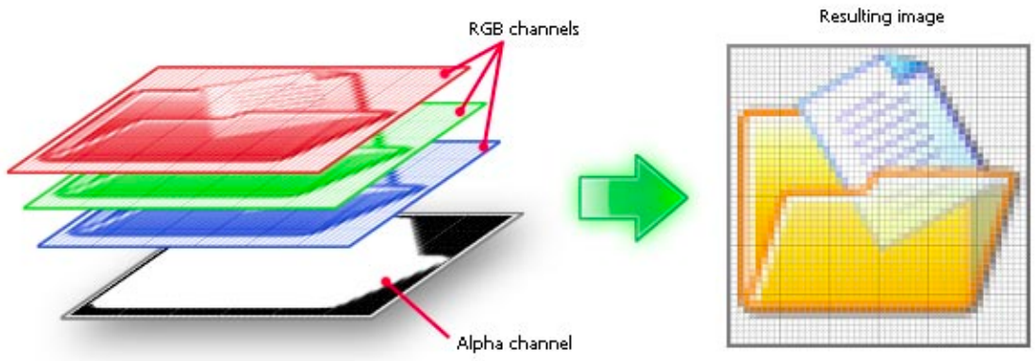


Structure of an image with Alpha Channel (RGB/A)

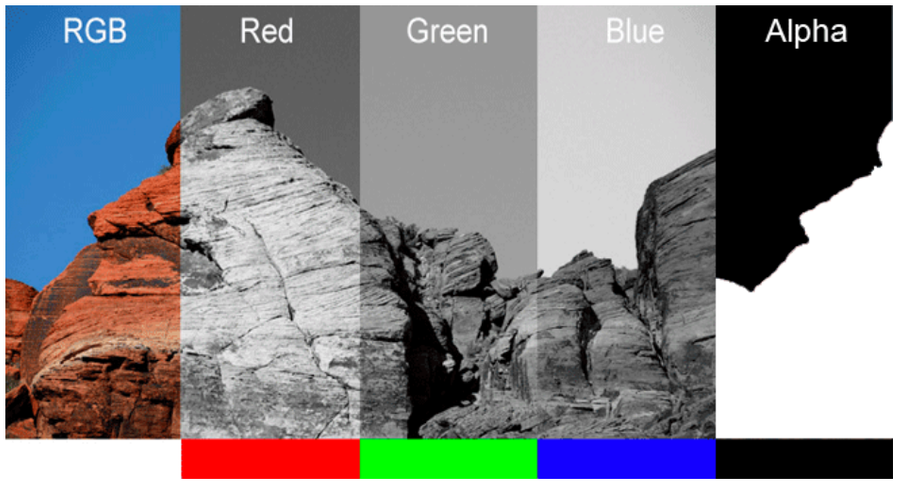


1.5 – Images

- Image files are nothing more than containers for a set of color matrices

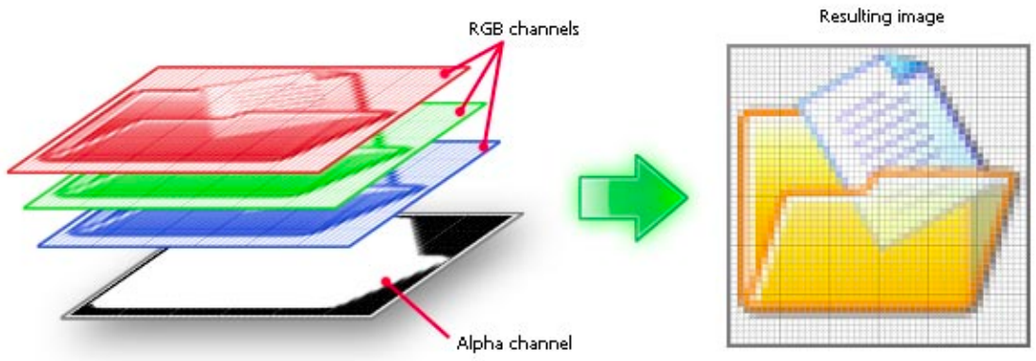


Structure of an image with Alpha Channel (RGB/A)

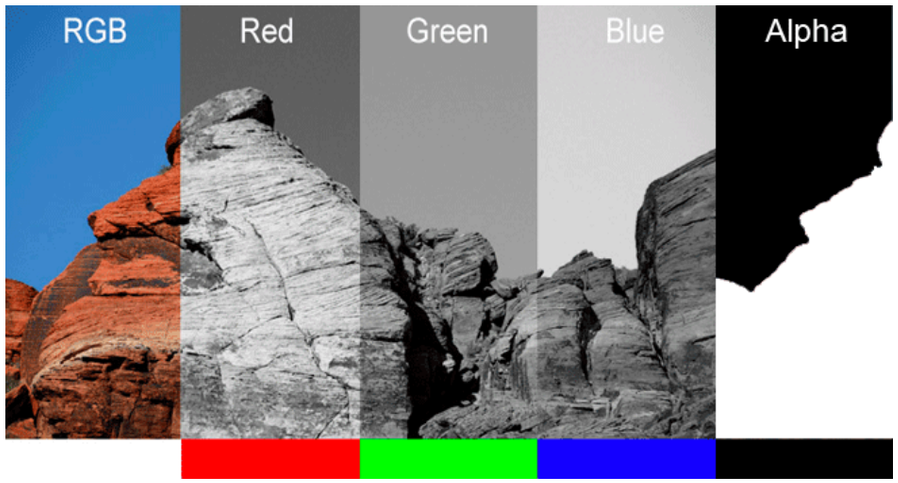


1.5 – Images

- Image files are nothing more than containers for a set of color matrices
- Let’s say we’re working with a 300x300 pixel JPG file

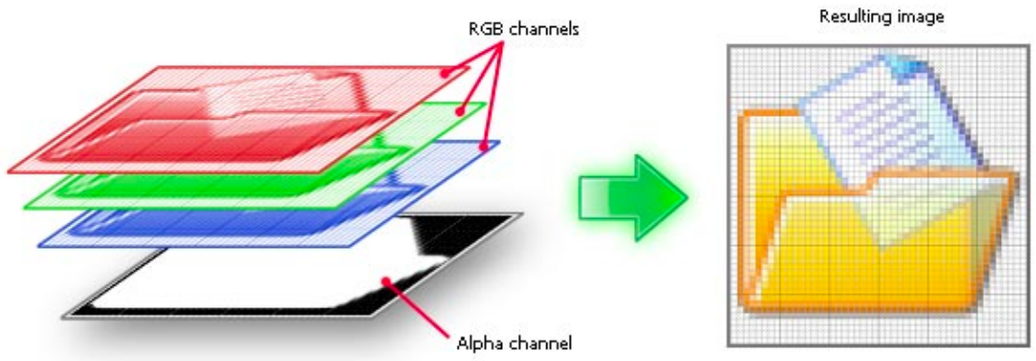


Structure of an image with Alpha Channel (RGB/A)

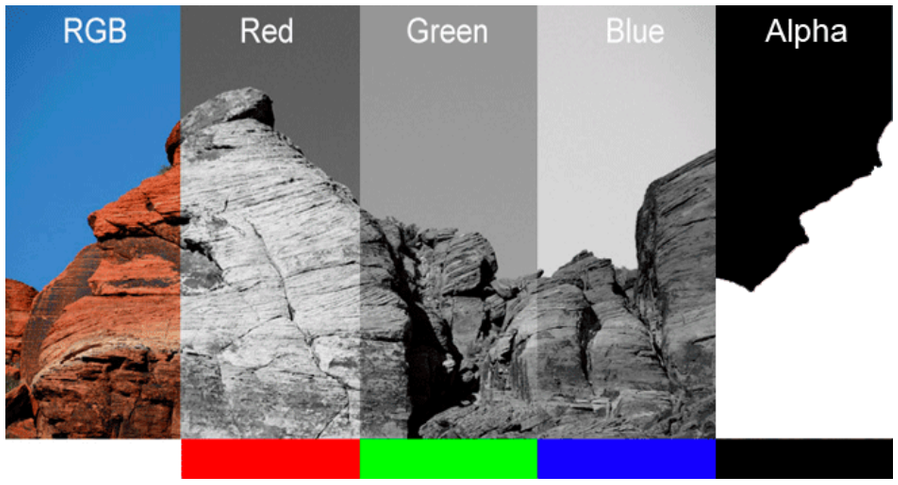


1.5 – Images

- Image files are nothing more than containers for a set of color matrices
- Let's say we're working with a 300x300 pixel JPG file
  - That means that there are 300 pixels in the X-dimension and 300 pixels in the Y-dimension

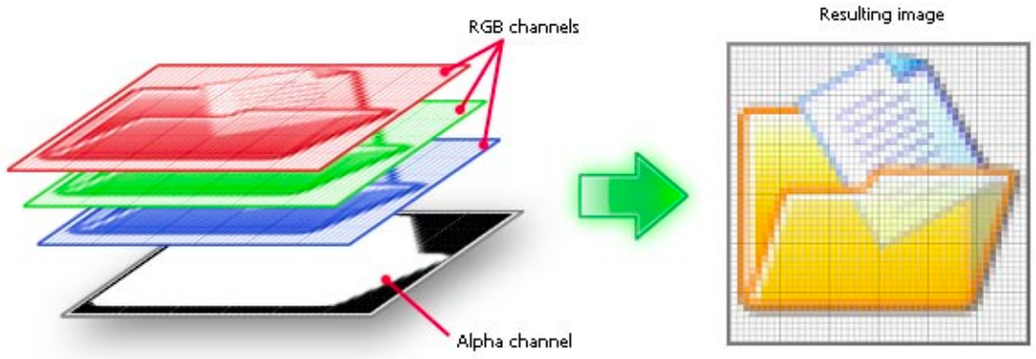


Structure of an image with Alpha Channel (RGB/A)

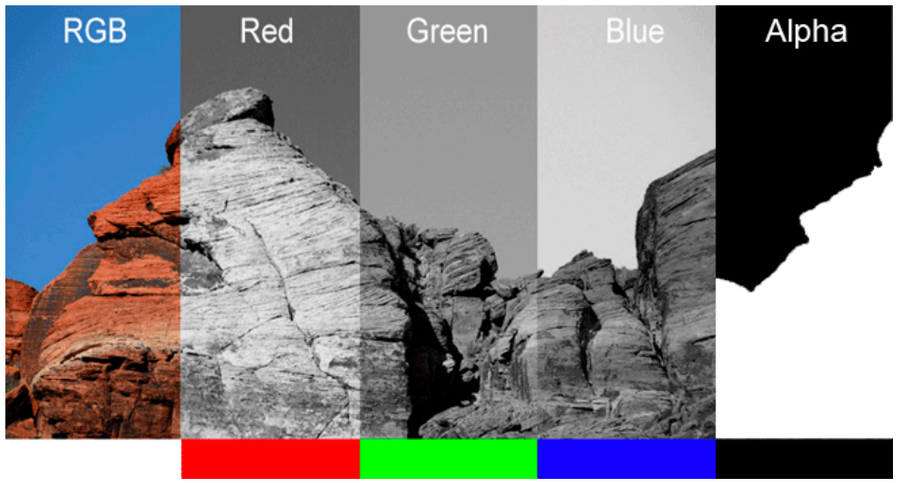


1.5 – Images

- Image files are nothing more than containers for a set of color matrices
- Let's say we're working with a 300x300 pixel JPG file
  - That means that there are 300 pixels in the X-dimension and 300 pixels in the Y-dimension
  - Each pixel is therefore a single element inside a 300x300 matrix

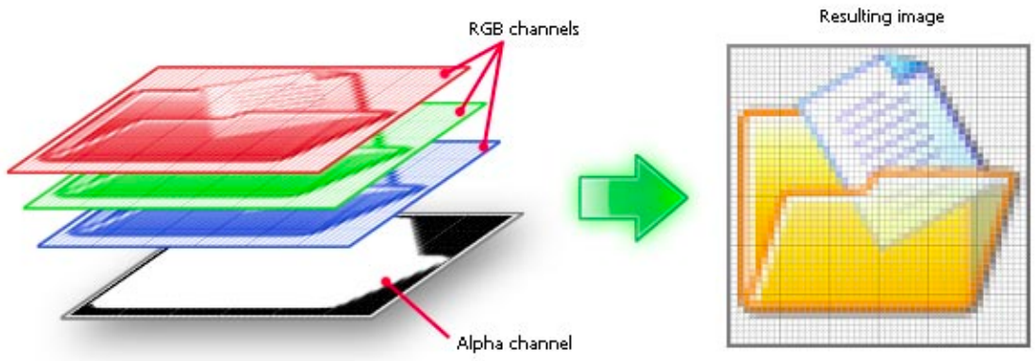


Structure of an image with Alpha Channel (RGB/A)

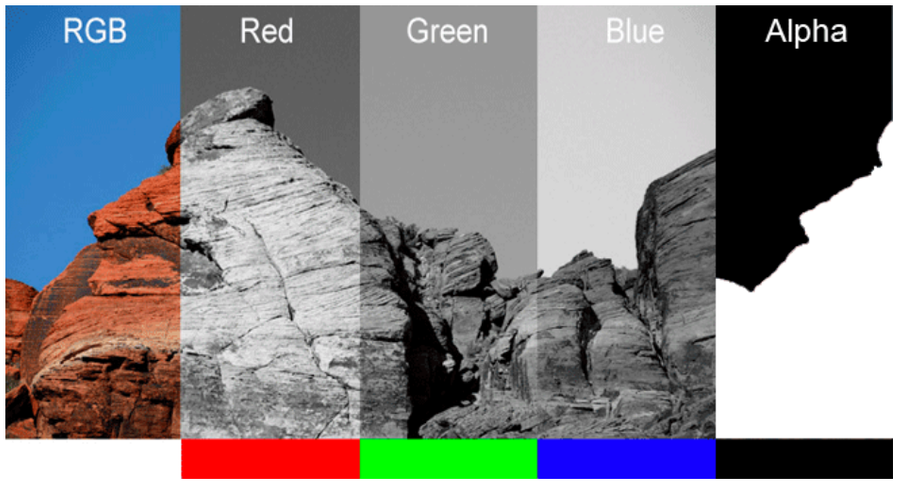


1.5 – Images

- Image files are nothing more than containers for a set of color matrices
- Let's say we're working with a 300x300 pixel JPG file
  - That means that there are 300 pixels in the X-dimension and 300 pixels in the Y-dimension
  - Each pixel is therefore a single element inside a 300x300 matrix
- Computers represent colors and transparency using a third dimension containing four layers: Red, Blue, Green, and "Alpha"



Structure of an image with Alpha Channel (RGB/A)



1.5 – Images

- To display an image in your PTB experiment, you need to do the following:
  - "Read" in an image file as a numerical matrix
  - Convert that matrix into a "texture"
  - Draw the texture
  - Display the texture
    - If you don't want your image to be centered (default), there's an entirely separate setup that is bewilderingly complicated…

```
puppyImage = imread('cutepuppy.jpg');

puppyTexture = Screen('MakeTexture', wPtr, puppyImage);

Screen('DrawTexture', wPtr, puppyTexture);

Screen('Flip', wPtr);
```

1.5 – Images

```
puppyImage = imread('cutepuppy.jpg');
puppyTexture = Screen('MakeTexture', wPtr, puppyImage);

[imHeight, imWidth, colorChannels] = size(puppyImage);
xPos = xCenter-imWidth/2;  % Centered
yPos = (yCenter-imHeight/2)-100;  % Offset
imRect = [xPos, yPos, xPos+imWidth, yPos+ImHeight];

Screen('DrawTexture', wPtr, puppyTexture, [], imRect);
Screen('Flip', wPtr);
```

A few closing thoughts

A few closing thoughts

- Getting the hang of computer graphics is one of the most difficult things for a new programmer to learn

A few closing thoughts

- Getting the hang of computer graphics is one of the most difficult things for a new programmer to learn
- I've been doing this for a while, and even I get tripped up!

A few closing thoughts

- Getting the hang of computer graphics is one of the most difficult things for a new programmer to learn
- I've been doing this for a while, and even I get tripped up!
- More than anything else in programming, playing around with computer graphics is a trial-and-error process – so don't be afraid to try things out and make mistakes!