COGS 119/219
MATLAB for Experimental Research

Fall 2014 – Week 6

Introduction to Psychtoolbox

Psychtoolbox

Psychtoolbox is a free set of MATLAB functions, which are written to make presenting visual stimuli easier.

You need to download it from here:

http://psychtoolbox.org

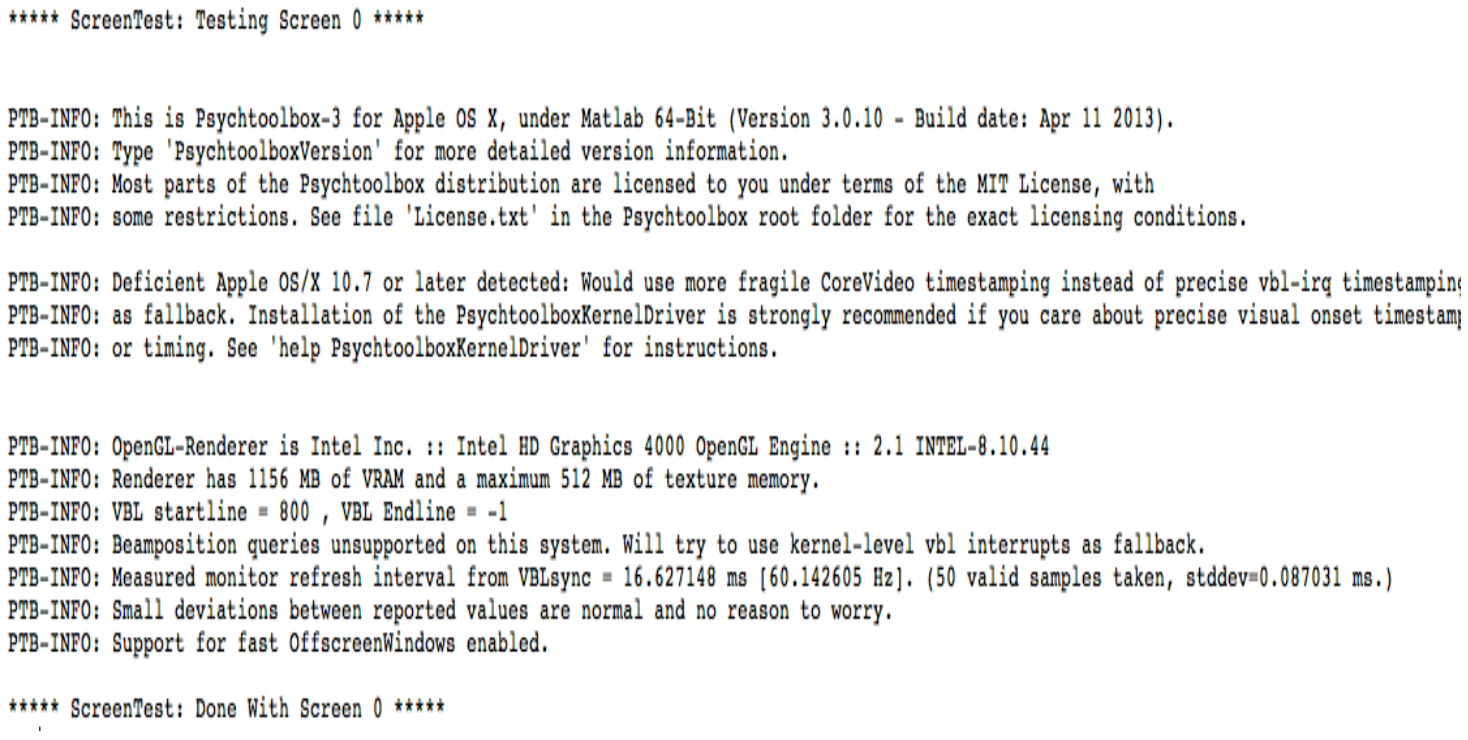
Citing Psychtoolbox:

- Brainard, D.H.(1997). The Psychophysics Toolbox, Spatial Vision10: 443-446.
- Pelli, D.G. (1997). The VideoToolbox software for visual psychophysics: Transforming numbers into movies, Spatial Vision 10: 437-442.

Getting started with Psychtoolbox

```
>> ScreenTest
```

If you get the screen going blank, and then something like below, then the screen test has worked.



```
***** ScreenTest: Testing Screen 0 *****


PTB-INFO: This is Psychtoolbox-3 for Apple OS X, under Matlab 64-Bit (Version 3.0.10 - Build date: Apr 11 2013).
PTB-INFO: Type 'PsychtoolboxVersion' for more detailed version information.
PTB-INFO: Most parts of the Psychtoolbox distribution are licensed to you under terms of the MIT License, with
PTB-INFO: some restrictions. See file 'License.txt' in the Psychtoolbox root folder for the exact licensing conditions.

PTB-INFO: Deficient Apple OS/X 10.7 or later detected: Would use more fragile CoreVideo timestamping instead of precise vbl-irq timestampin
PTB-INFO: as fallback. Installation of the PsychtoolboxKernelDriver is strongly recommended if you care about precise visual onset timestam
PTB-INFO: or timing. See 'help PsychtoolboxKernelDriver' for instructions.


PTB-INFO: OpenGL-Renderer is Intel Inc. :: Intel HD Graphics 4000 OpenGL Engine :: 2.1 INTEL-8.10.44
PTB-INFO: Renderer has 1156 MB of VRAM and a maximum 512 MB of texture memory.
PTB-INFO: VBL startline = 800 , VBL Endline = -1
PTB-INFO: Beamposition queries unsupported on this system. Will try to use kernel-level vbl interrupts as fallback.
PTB-INFO: Measured monitor refresh interval from VBLsync = 16.627148 ms [60.142605 Hz]. (50 valid samples taken, stddev=0.087031 ms.)
PTB-INFO: Small deviations between reported values are normal and no reason to worry.
PTB-INFO: Support for fast OffscreenWindows enabled.

***** ScreenTest: Done With Screen 0 *****
```

Screen

- `Screen` is one of the core functions of Psychtoolbox.

- It is actually not an m file, it is a mex file. It is originally written in C using OpenGL, and is then compiled to run in MATLAB.

```
>> help Screen
```

- The typical format is

  Screen ('SomeCommand', Parameters, …)

Screen functions

```
>> Screen
```

- You will see a long list of commands that you can use with Screen.
- We will see examples of these commands soon.
- You need to give these commands as arguments to Screen, together with the necessary parameters associated with each command.

OpenWindow

- Before you can do anything on display, you need to open a window.
- To get help type

```
>> Screen OpenWindow?
```

or

```
>> Screen(‘OpenWindow?’)
```

```
[windowPtr,rect] =
Screen('OpenWindow',windowPtrOrScreenNumber
[,color] [,rect][,pixelSize][,numberOfBuffers]
[,stereomode][,multisample][,imagingmode]
[,specialFlags][,clientRect]);
```

OpenWindow

```
[windowPtr,rect] =
Screen('OpenWindow',windowPtrOrScreenNumber
[,color] [,rect][,pixelSize][,numberOfBuffers]
[,stereomode][,multisample][,imagingmode]
[,specialFlags][,clientRect]);
```

The function takes 2 <u>required</u> arguments:

‘OpenWindow’ and windowPtrOrScreenNumber

And 7 <u>optional</u> arguments (specified in brackets):

color, rect, pixelSize, numberOfBuffers, stereoMode, multisample, imagingmode, specialFlags, clientRect

For example:

[w, rect] = Screen(‘OpenWindow’, 0, 0, []);

OpenWindow

```
[w, rect] = Screen(‘OpenWindow’, 0, 0, []);
```

<u>INPUTS:</u>

- ‘OpenWindow’ is a string telling Screen what to do.

- Second argument is which computer monitor you want to use. Typically this will be 0 (If you have multiple monitors attached, things may get a bit complicated; for now just use one monitor).

- Third argument indicates that you want the screen to be black (0).

OpenWindow

```
[w, rect] = Screen('OpenWindow', 0, 0, []);
```

<u>INPUTS: Optional arguments</u>

- Fourth argument [] means , use the default. In this case, the experiment screen will cover the whole monitor (screen). You may want to use a smaller screen that doesn't take up the whole monitor when you're developing code as it is easier to recover from crashes that way.

- Optional arguments are indicated with with brackets.

- Optional arguments must be specified in order, without omitting earlier ones, but you can use the empty matrix [] as a placeholder with the same effect of omitting it.

OpenWindow

```
[w, rect] = Screen(‘OpenWindow’, 0, 0, []);
```

<u>OUTPUTS:</u>

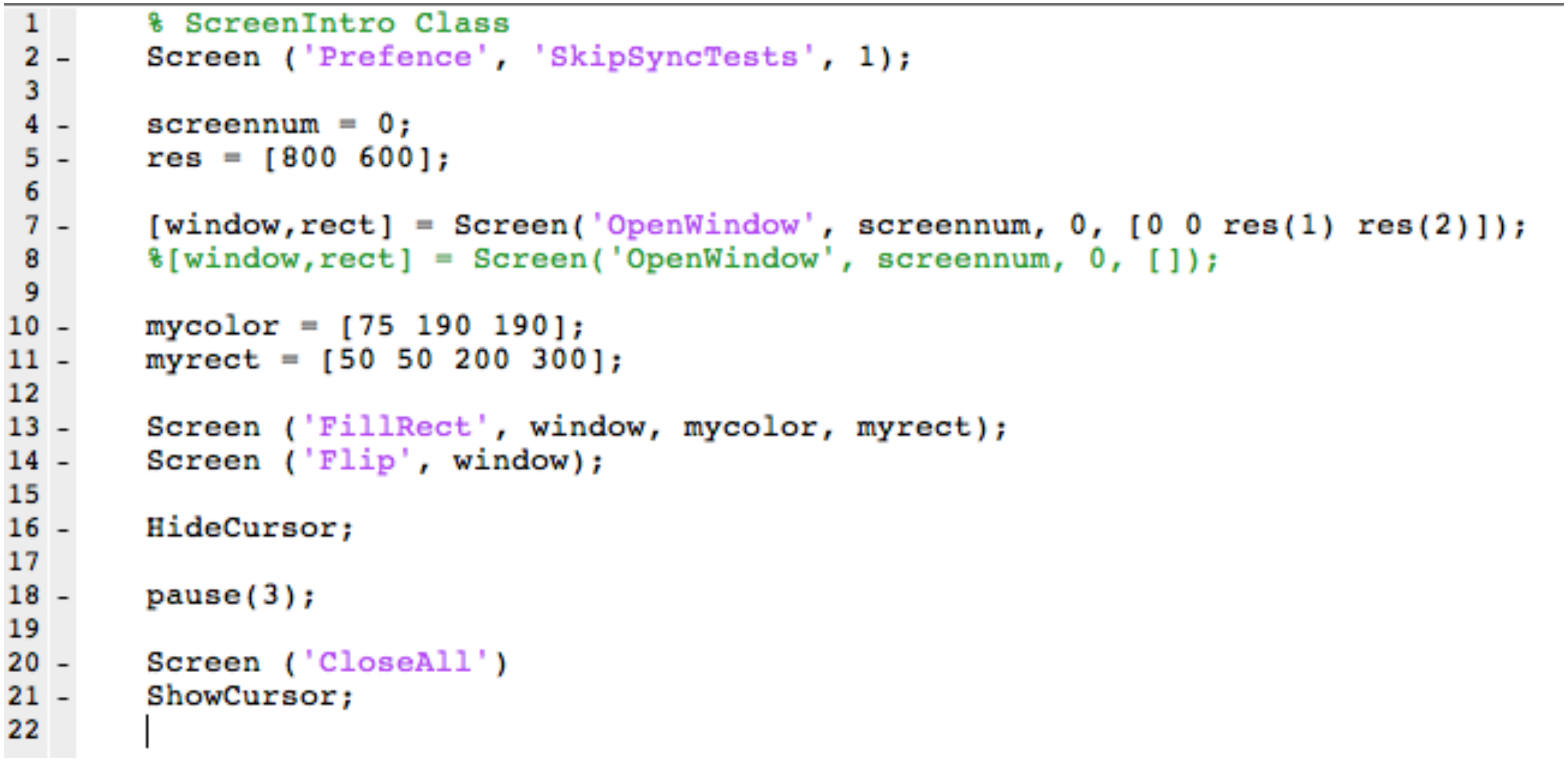
- w is a handle or pointer to the window. You will need it when you want to put stimuli on the screen or in any manner perform other operations on this window. (Similar to how you manipulate files using file handle/pointer)

- rect is a variable (1 x 4 matrix) that contains the size of the window. It will have 4 numbers in it, which are the positions of the start and end coordinates of your window (rectangle).

Coordinate system

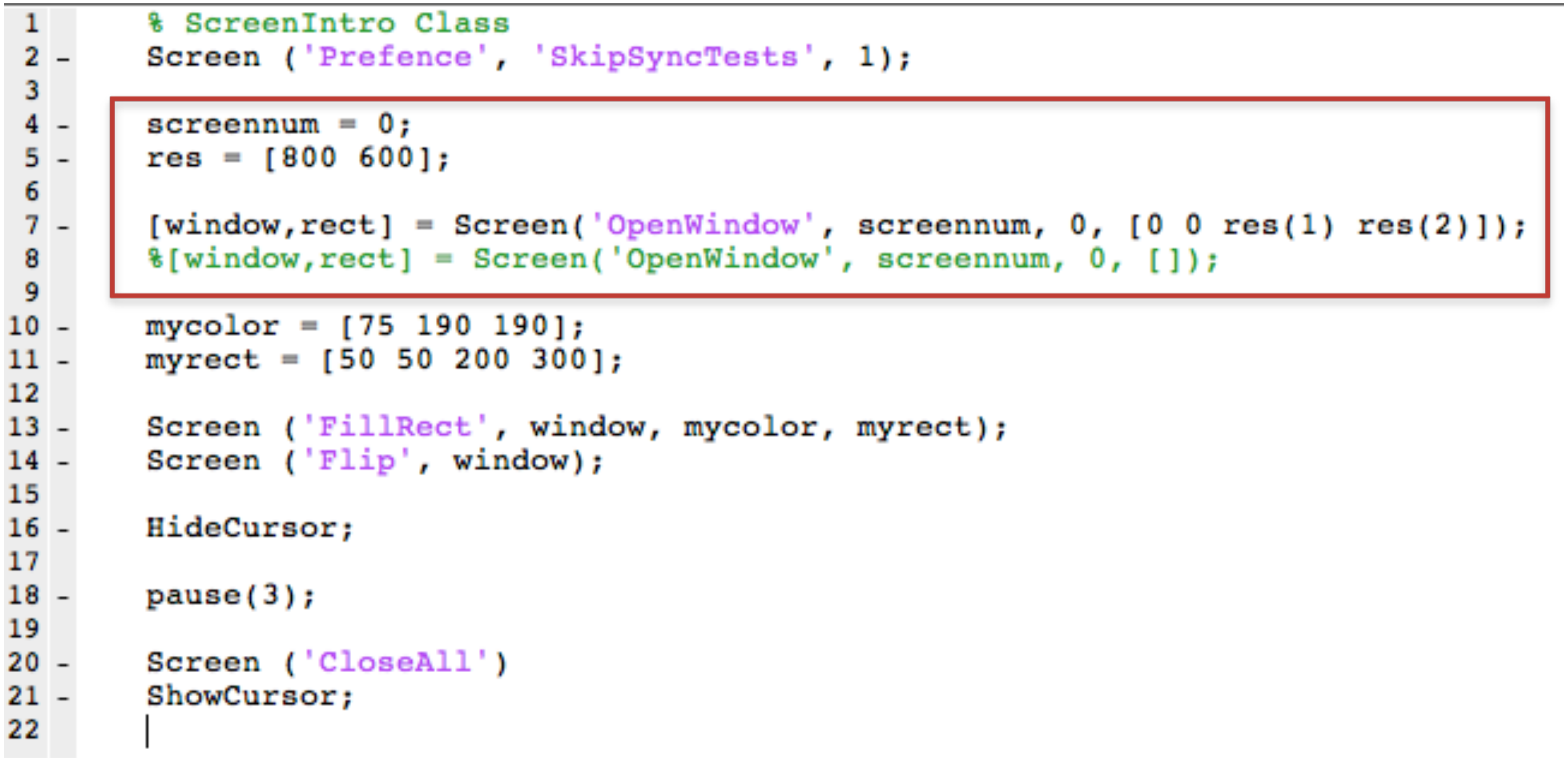
- The coordinate system has 0,0 in the upper left-hand corner.
- The order of these coordinates can be coded as LeTteRBox (Left, Top, Right, Bottom).
- Note that the coordinates start from 0, not 1.
- If you just type `rect`, you can see the size of your current monitor's window.

ScreenIntro.m



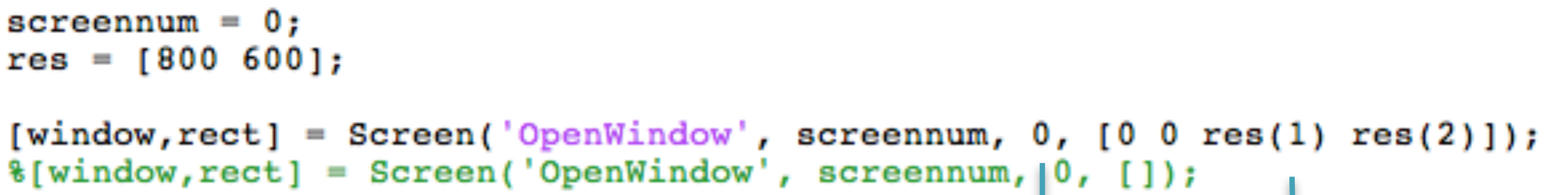
```matlab
% ScreenIntro Class
Screen ('Prefence', 'SkipSyncTests', 1);

screennum = 0;
res = [800 600];

[window,rect] = Screen('OpenWindow', screennum, 0, [0 0 res(1) res(2)]);
%[window,rect] = Screen('OpenWindow', screennum, 0, []);

mycolor = [75 190 190];
myrect = [50 50 200 300];

Screen ('FillRect', window, mycolor, myrect);
Screen ('Flip', window);

HideCursor;

pause(3);

Screen ('CloseAll')
ShowCursor;

```

ScreenIntro.m



```
% ScreenIntro Class
Screen ('Prefence', 'SkipSyncTests', 1);

screennum = 0;
res = [800 600];

[window,rect] = Screen('OpenWindow', screennum, 0, [0 0 res(1) res(2)]);
%[window,rect] = Screen('OpenWindow', screennum, 0, []);

mycolor = [75 190 190];
myrect = [50 50 200 300];

Screen ('FillRect', window, mycolor, myrect);
Screen ('Flip', window);

HideCursor;

pause(3);

Screen ('CloseAll')
ShowCursor;

```

OpenWindow

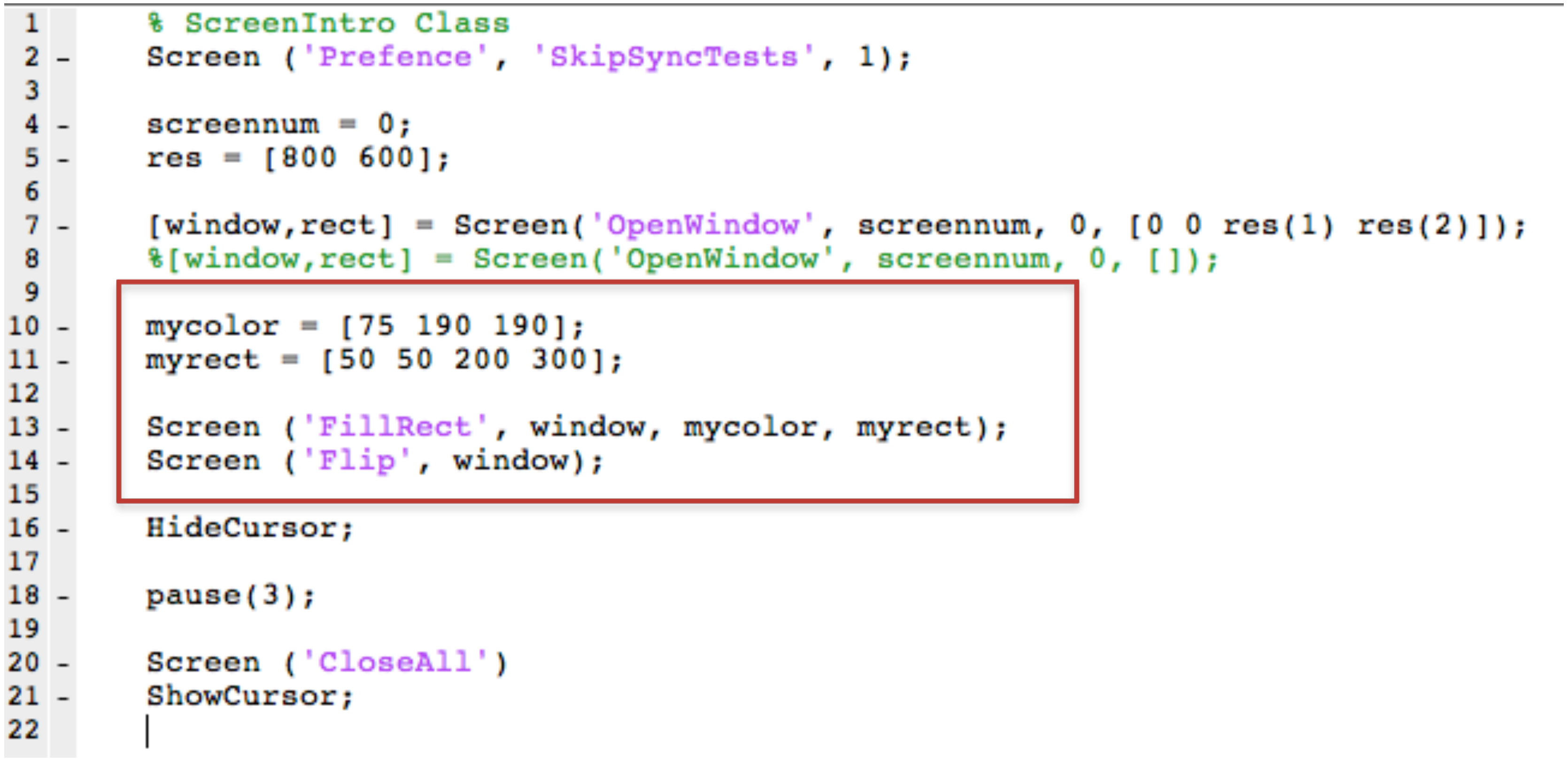


```
screennum = 0;
res = [800 600];

[window,rect] = Screen('OpenWindow', screennum, 0, [0 0 res(1) res(2)]);
%[window,rect] = Screen('OpenWindow', screennum, 0, []);
```

color

window
coordinates

ScreenIntro.m

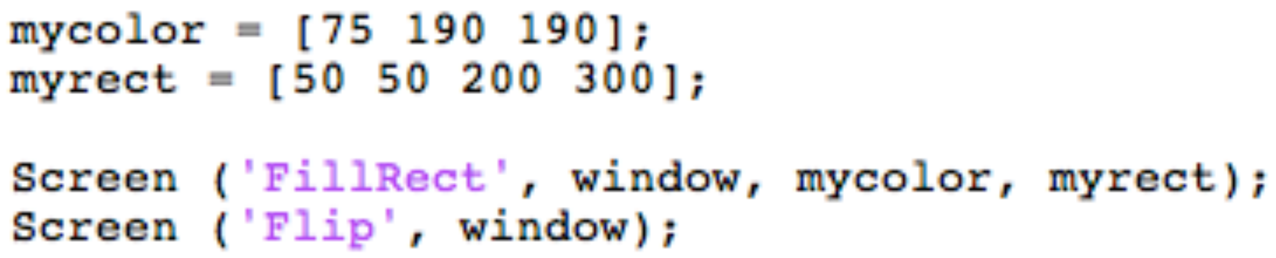


```
% ScreenIntro Class
Screen ('Prefence', 'SkipSyncTests', 1);

screennum = 0;
res = [800 600];

[window,rect] = Screen('OpenWindow', screennum, 0, [0 0 res(1) res(2)]);
%[window,rect] = Screen('OpenWindow', screennum, 0, []);

mycolor = [75 190 190];
myrect = [50 50 200 300];

Screen ('FillRect', window, mycolor, myrect);
Screen ('Flip', window);

HideCursor;

pause(3);

Screen ('CloseAll')
ShowCursor;

```

Flip function

- Once Psychtoolbox (PTB) has opened a main window, you can draw into it.

- There is a "back buffer" onto which everything is drawn (off screen), so that things will only become visible when you flip the buffers:



```
mycolor = [75 190 190];
myrect = [50 50 200 300];

Screen ('FillRect', window, mycolor, myrect);
Screen ('Flip', window);
```

```
>> Screen('FillRect?')
```

On screen vs. off screen window

- On screen window:

  This is where you display your stimulus.

- Off screen window:

  Whenever you draw something with PTB, it will usually be drawn ‘off screen’. The image exists in memory but is not visible at this point. You need to use the Flip function of Screen so that off screen window gets copied to on screen window.

Screen close

```
HideCursor;

pause(3);

Screen ('CloseAll')
ShowCursor;
```

Hides the cursor

Shows the cursor

Closes all open onscreen and offscreen windows and textures, movies and video sources.

Screen functions

- Let's play with some Screen functions.
- Download **FunScreen.m** from the class website, open and run it.
- Now, let's look closer to the Screen functions used.

See how these functions are used:

```
>> Screen('GetFlipInterval?')
```

Frame rate: Frame rate is the frequency at which a device (e.g., your monitor) produces unique consecutive images called frames. Most LCD monitors and laptop screens will be operating at 60 Hz, which means 60 frames per second.

```
>> Screen ('FillRect?');

>> Screen('FillOval?');

>> Screen('DrawLine?');

>> Screen('DrawText');
```

Displaying Images in PTB

Download **playwithtextures.m** from the class website.

```
>> Screen(‘MakeTexture?’)
```

```
TextureIndex = Screen('MakeTexture', WindowIndex,
                      imageMatrix, optional arguments…)
```

Textures

```
% Read in image just like usual
myimg = imread('grumpycat','jpg');

% make texture, nothing happens on screen.
% this just prepares the image as texture on offscreen
% window
mytex = Screen('MakeTexture', w, myimg);

Screen('DrawTexture', w, mytex);
Screen('Flip',w);
```

Alternative way to display images in PTB

```
% Read in image just like usual
myimg = imread('grumpycat','jpg');

% Use PutImage function
Screen('PutImage', w, myimg);
Screen('Flip',w);
```

Textures, Screen, Window

What's the difference between window, screen and texture?

http://psychtoolbox.org/wikka.php?wakka=FaqTextureWindow

Images in different locations on screen

- Let’s display an image on different locations on Screen.
- Download **playwithtextures2.m** and ucsdlogo.gif from the class website and run the m file.
- UCSD logo is displayed first on the left and then on the right of the screen.
- See how we can specify where can display an image.

Images in different locations on screen

- The two Screen functions to display images, `PutImage` and `DrawTexture` both require a location for the image as an input argument.

- So, you need to calculate the coordinates of the where you want to put your image (use your geometry knowledge).

- Start by calculating the coordinates of the center of the screen.

```
>> help RectCenter
```

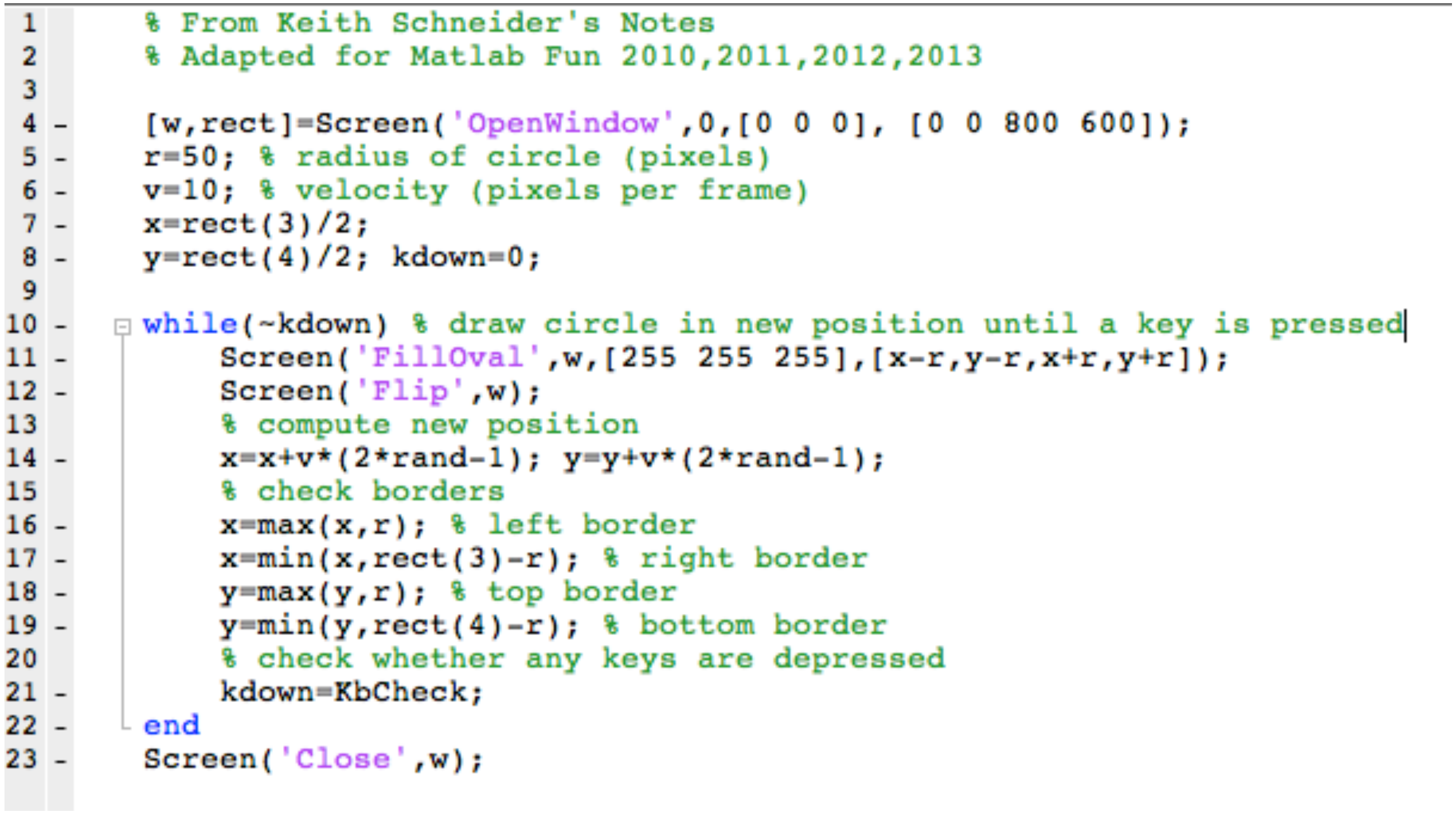
- Then, calculate the coordinates of the image location relative to the center based on the image size (width and height) (see lines 25-32 in **playwithtextures2.m** how this is done).

Exercise

- Draw a white circle that moves randomly on screen without crossing the top, bottom, left and right borders of the screen.

- How can you do that? Any ideas?

randompath.m



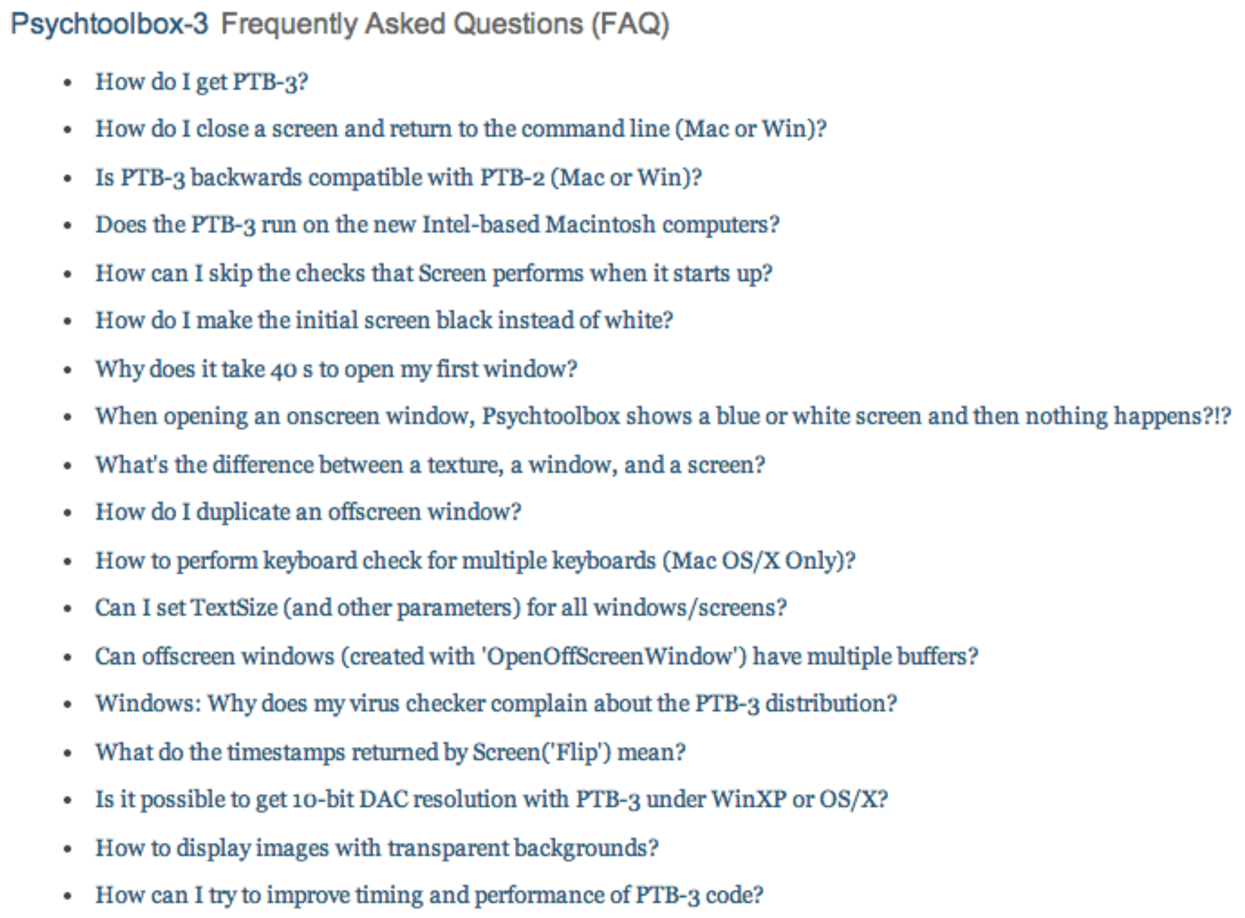
```
% From Keith Schneider's Notes
% Adapted for Matlab Fun 2010,2011,2012,2013

[w,rect]=Screen('OpenWindow',0,[0 0 0], [0 0 800 600]);
r=50; % radius of circle (pixels)
v=10; % velocity (pixels per frame)
x=rect(3)/2;
y=rect(4)/2; kdown=0;

while(~kdown) % draw circle in new position until a key is pressed
    Screen('FillOval',w,[255 255 255],[x-r,y-r,x+r,y+r]);
    Screen('Flip',w);
    % compute new position
    x=x+v*(2*rand-1); y=y+v*(2*rand-1);
    % check borders
    x=max(x,r); % left border
    x=min(x,rect(3)-r); % right border
    y=max(y,r); % top border
    y=min(y,rect(4)-r); % bottom border
    % check whether any keys are depressed
    kdown=KbCheck;
end
Screen('Close',w);
```

Some helpful tips about PTB I

See "`help PsychDemos`" for many demos which demonstrate Screen's capabilities.

Some helpful tips about PTB II

http://psychtoolbox.org/PsychtoolboxFaq



Psychtoolbox-3 Frequently Asked Questions (FAQ)

- How do I get PTB-3?
- How do I close a screen and return to the command line (Mac or Win)?
- Is PTB-3 backwards compatible with PTB-2 (Mac or Win)?
- Does the PTB-3 run on the new Intel-based Macintosh computers?
- How can I skip the checks that Screen performs when it starts up?
- How do I make the initial screen black instead of white?
- Why does it take 40 s to open my first window?
- When opening an onscreen window, Psychtoolbox shows a blue or white screen and then nothing happens?!?
- What's the difference between a texture, a window, and a screen?
- How do I duplicate an offscreen window?
- How to perform keyboard check for multiple keyboards (Mac OS/X Only)?
- Can I set TextSize (and other parameters) for all windows/screens?
- Can offscreen windows (created with 'OpenOffScreenWindow') have multiple buffers?
- Windows: Why does my virus checker complain about the PTB-3 distribution?
- What do the timestamps returned by Screen('Flip') mean?
- Is it possible to get 10-bit DAC resolution with PTB-3 under WinXP or OS/X?
- How to display images with transparent backgrounds?
- How can I try to improve timing and performance of PTB-3 code?

Some helpful tips about PTB III

SYNC Troubles:

If you see some timing errors, complaints about sync or VBL, see this: http://docs.psychtoolbox.org/SyncTrouble.

Some helpful tips about PTB IV

- If your computer only has one screen (the typical scenario) and your program produces a Matlab error while your full-screen window is open, you'll hear the beep, but you won't be able to see the Matlab Command Window.

- Follow the instructions below for bringing forward the command window, then type "`clear Screen`" to flush just the Screen MEX file, or "`clear mex`" to flush all the MEX files. When flushed, as part of its exit sequence, Screen closes all its windows, restores the screen's normal color table, and shows the cursor.

- Or you can get just those effects, without flushing, by calling `Screen('CloseAll')` or sca - which is an abbreviation for `Screen('CloseAll')`.

Some helpful tips about PTB IV

- You can use Matlab's EVAL command to do this for you automatically. E.g. if your program is called "foo.m", run your program by calling EVAL:

```
eval('foo','clear screen;error("error in foo")’)
```

- If an error occurs in FOO, Matlab, instead of halting, will execute the second argument to EVAL, which restores your screen and reports the error.

Some helpful tips about PTB V

- Command-zero brings the Matlab Command window forward. (Type a zero "0" while holding the apple-cloverleaf "command" key down.)

- Ctrl-C halts any program. (Type a "c" while holding down the "Ctrl" key). Sometimes, Ctrl-C fails to halt progams executing in a Matlab process run with the "-nojvm" option. To halt a runaway Psychtoolbox script in Psychtoolbox you might resort to the Windows Task Manager to kill the Matlab process. (Use Ctrl-Alt-Delete to open a window from which you can start the Task Manager.)

Some helpful tips about PTB V

- Windows:: Ctrl-Alt-Delete allows you to launch the Windows task manager, which reduces the Psychtoolbox onscreen windows when it opens. (Simultaneosly press the "Ctrl", "Alt", and "Delete" keys.) There are also simpler ways of reducing the Psychtoolbox window which are specific to particular versions of Windows.
- Windows 2000: Alt-Tab will bring another application to the foreground, minimizing the Matlab Psychtoolbox window.
- OS-X: Apple-Command-Escape executes "Force Quit" on Matlab, closing Matlab and all of its windows.
- Linux: Ctrl-Alt-Escape, followed by a mouse click kills the onscreen windows and your Matlab session.

Some helpful tips about PTB VI

OPENGL:

OpenGL is a library for computer graphics. It allows us to create applications which render high-quality color images with 3D objects. You are not required to go into details of OpenGL at the moment.
http://www.opengl.org/about/overview

Some helpful tips about PTB VII

WARNINGS:

You can disable PTB’s warnings about graphics and timing for the time being by including the following in your code:

```
Screen('Preference', 'Verbosity', 0);
Screen('Preference', 'SkipSyncTests',1);
Screen('Preference', 'VisualDebugLevel',0);
```

Type `Screen(‘Preference?’)` in the command line to get more info.