COGS 119/219
MATLAB for Experimental Research

Fall 2014 – Week 8

Keyboard input in Psychtoolbox

Recall

- How have we been getting key presses from the user?

```
>> input
>> pause
```

- Note that the command window needs to be in front to get key presses with these functions, so they don't work well with Psychtoolbox.

- In that case, we need to use some different functions including `GetChar()` or `KbCheck()`.

GetChar

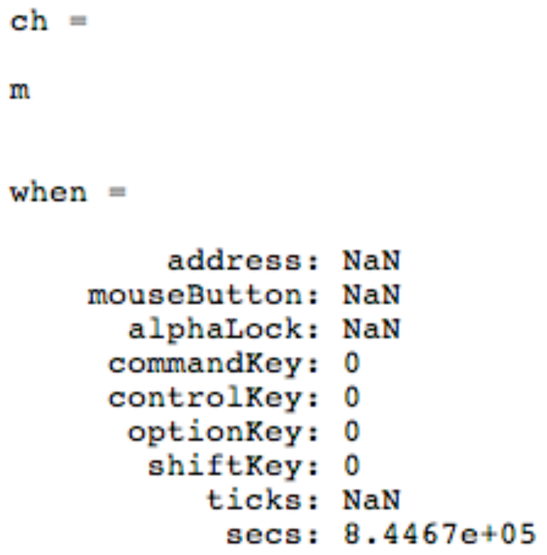
```
>> help GetChar
[ch, when] = GetChar
```

ch: this is the character that was typed.

when: this is a struct that includes the time of the key press, address of the input device, the state of all the modifier keys and the mouse button.

```
>> [ch, when] = GetChar
```

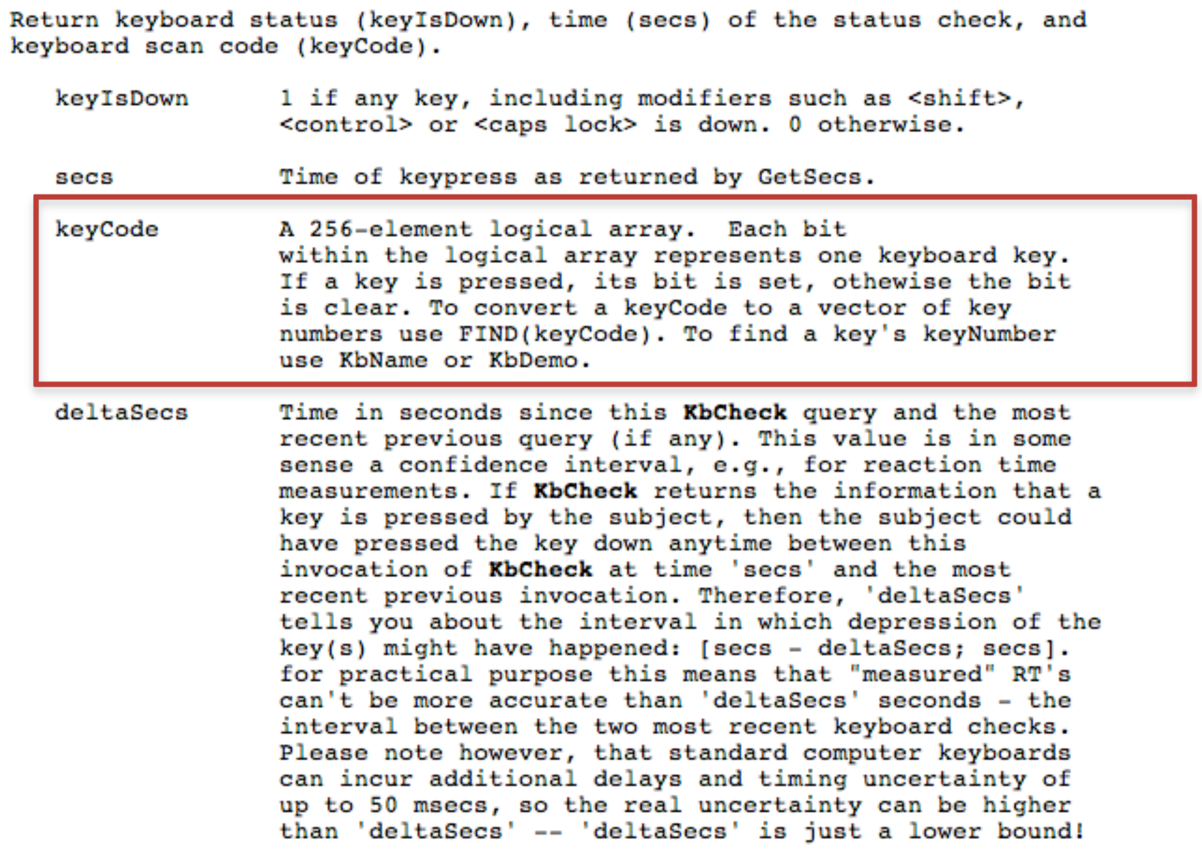
Press m key



```
ch =

m


when =

        address: NaN
    mouseButton: NaN
      alphaLock: NaN
     commandKey: 0
     controlKey: 0
      optionKey: 0
       shiftKey: 0
          ticks: NaN
           secs: 8.4467e+05
```

KbCheck function

```
>> help KbCheck
[keyIsDown, secs, keyCode, deltaSecs] = KbCheck()
```



```
Return keyboard status (keyIsDown), time (secs) of the status check, and
keyboard scan code (keyCode).

   keyIsDown      1 if any key, including modifiers such as <shift>,
                  <control> or <caps lock> is down. 0 otherwise.

   secs           Time of keypress as returned by GetSecs.

   keyCode        A 256-element logical array.  Each bit
                  within the logical array represents one keyboard key.
                  If a key is pressed, its bit is set, othewise the bit
                  is clear. To convert a keyCode to a vector of key
                  numbers use FIND(keyCode). To find a key's keyNumber
                  use KbName or KbDemo.

   deltaSecs      Time in seconds since this KbCheck query and the most
                  recent previous query (if any). This value is in some
                  sense a confidence interval, e.g., for reaction time
                  measurements. If KbCheck returns the information that a
                  key is pressed by the subject, then the subject could
                  have pressed the key down anytime between this
                  invocation of KbCheck at time 'secs' and the most
                  recent previous invocation. Therefore, 'deltaSecs'
                  tells you about the interval in which depression of the
                  key(s) might have happened: [secs - deltaSecs; secs].
                  for practical purpose this means that "measured" RT's
                  can't be more accurate than 'deltaSecs' seconds - the
                  interval between the two most recent keyboard checks.
                  Please note however, that standard computer keyboards
                  can incur additional delays and timing uncertainty of
                  up to 50 msecs, so the real uncertainty can be higher
                  than 'deltaSecs' -- 'deltaSecs' is just a lower bound!
```

KbWait

```
>> help KbWait
[secs, keyCode, deltaSecs] = KbWait()
```

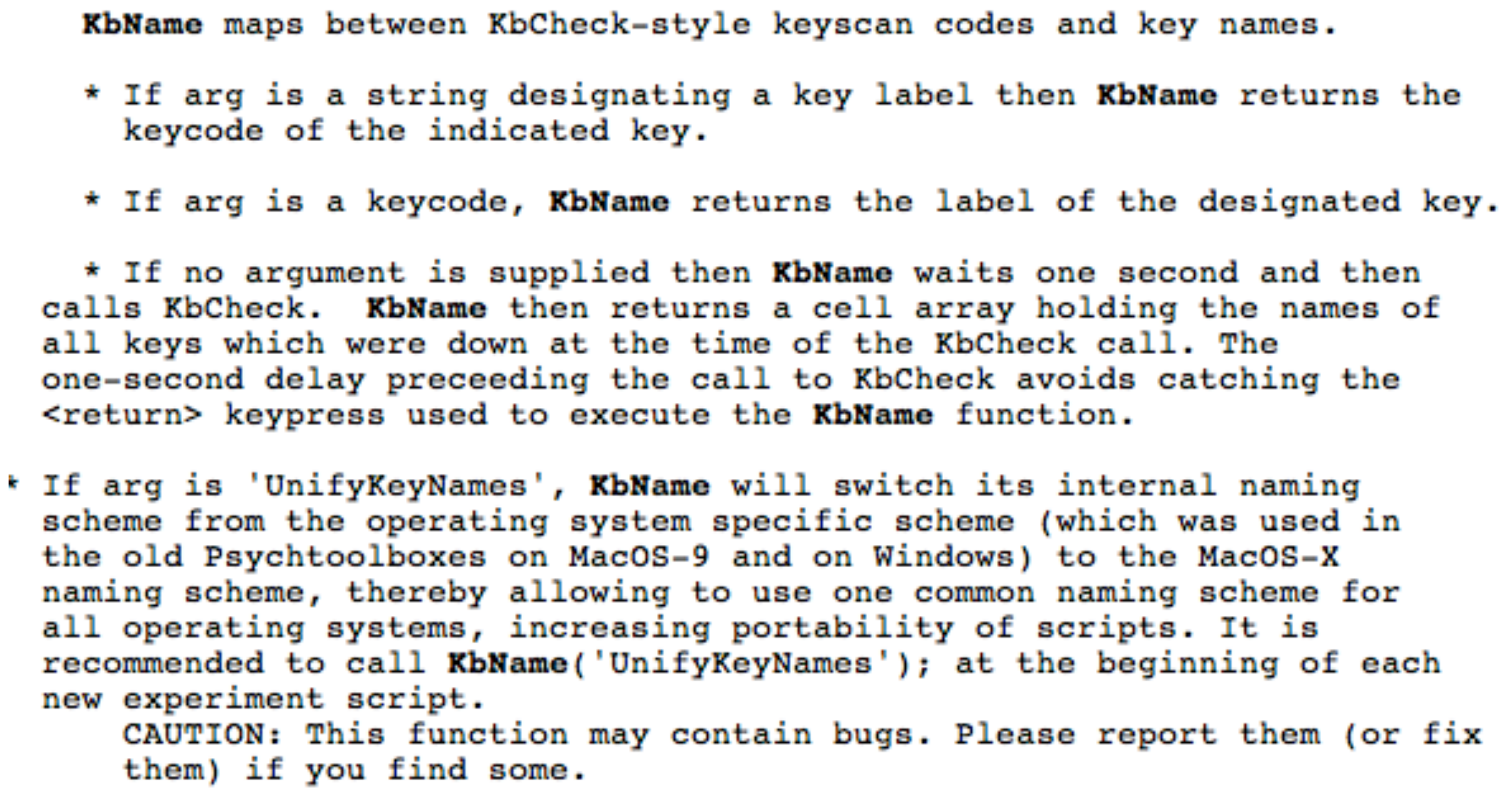
Just like the KbCheck, but it waits until a key on the keyboard is pressed down, and simply return the time the key was pressed.

KbName function

```
>> help KbName
kbNameResult = KbName(arg)
```

Examples

```
k1 = KbName ('q');   % 20
k2 = KbName (20);   % q
k3 = KbName (59); % F2
k4 = KbName (44); % space
k5 = KbName ('tab'); % 43
```



```
  KbName maps between KbCheck-style keyscan codes and key names.

  * If arg is a string designating a key label then KbName returns the
    keycode of the indicated key.

  * If arg is a keycode, KbName returns the label of the designated key.

  * If no argument is supplied then KbName waits one second and then
calls KbCheck.  KbName then returns a cell array holding the names of
all keys which were down at the time of the KbCheck call. The
one-second delay preceeding the call to KbCheck avoids catching the
<return> keypress used to execute the KbName function.

* If arg is 'UnifyKeyNames', KbName will switch its internal naming
scheme from the operating system specific scheme (which was used in
the old Psychtoolboxes on MacOS-9 and on Windows) to the MacOS-X
naming scheme, thereby allowing to use one common naming scheme for
all operating systems, increasing portability of scripts. It is
recommended to call KbName('UnifyKeyNames'); at the beginning of each
new experiment script.
    CAUTION: This function may contain bugs. Please report them (or fix
    them) if you find some.
```

```
>> KbName('UnifyKeyNames');
```

Most keynames are shared between Windows and Macintosh, but not all of them.

For a list of them, you can check KbName.m

```
>> edit KbName
```

If you type this command at the beginning of your code, `KbName` will try to use a mostly shared (between Mac and Windows) name mapping. (This doesn't work that well).

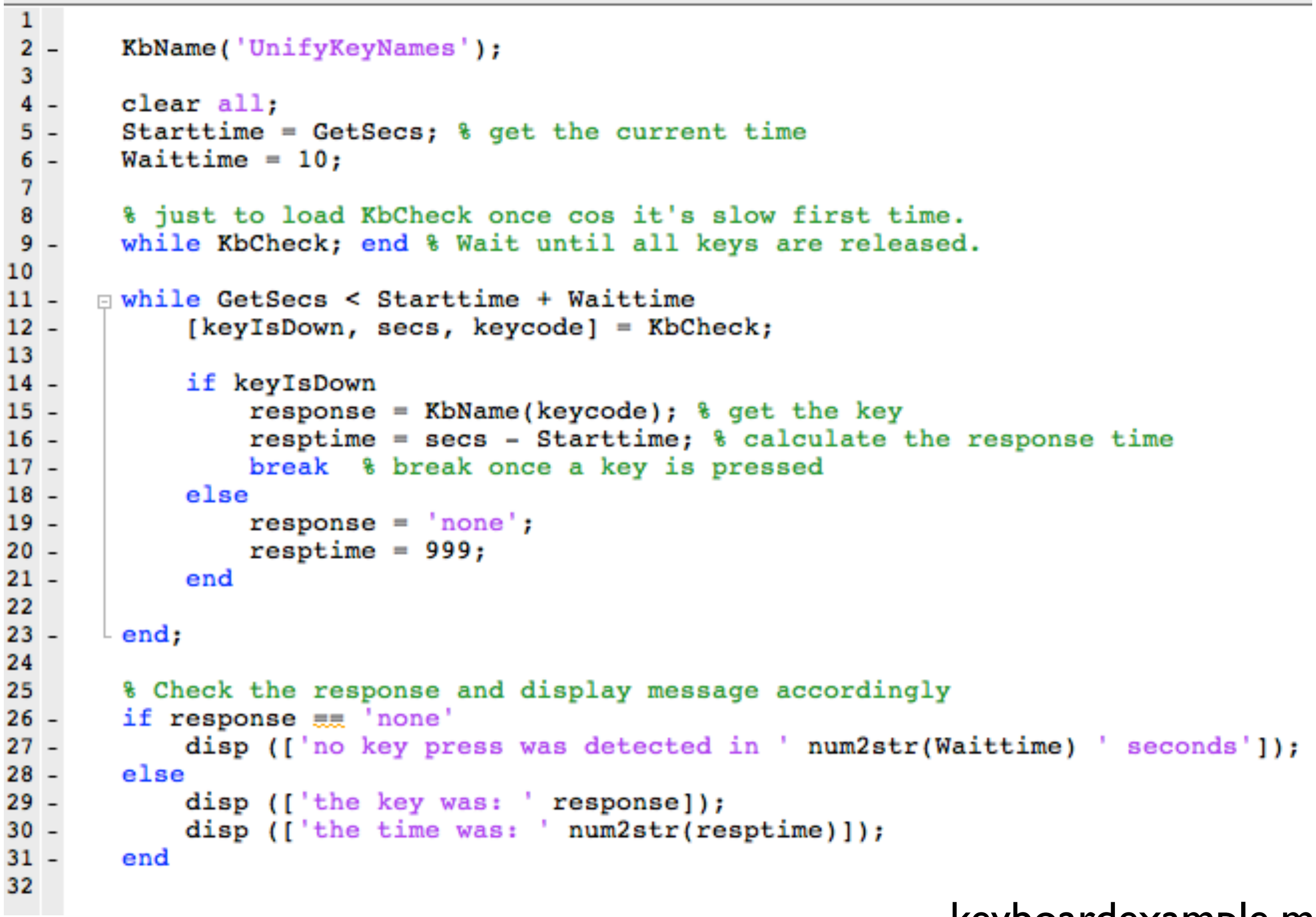
```
>> KbDemo
```

This is a demo of Psychtoolbox that shows the functionality of `KbCheck` and `KbWait`.

keyboardintro.m

```
% A simple program that gets key presses and displays the key pressed
function keyboardintro

KbName('UnifyKeyNames');

escapeKey = KbName('ESCAPE');
while KbCheck; end % Wait until all keys are released.

while 1
    % Check the state of the keyboard.
          [ keyIsDown, seconds, keyCode ] = KbCheck;

    % If the user is pressing a key, then display its code number and name.
    if keyIsDown

        % Note that we use find(keyCode) because keyCode is an array.
        % See 'help KbCheck'
        fprintf('You pressed key %i which is %s\n', find(keyCode), KbName(keyCode));

        if keyCode(escapeKey)
            break;
        end

        % If the user holds down a key, KbCheck will report multiple events.
        % To condense multiple 'keyDown' events into a single event, we wait until all
        % keys have been released.
    end
end
```

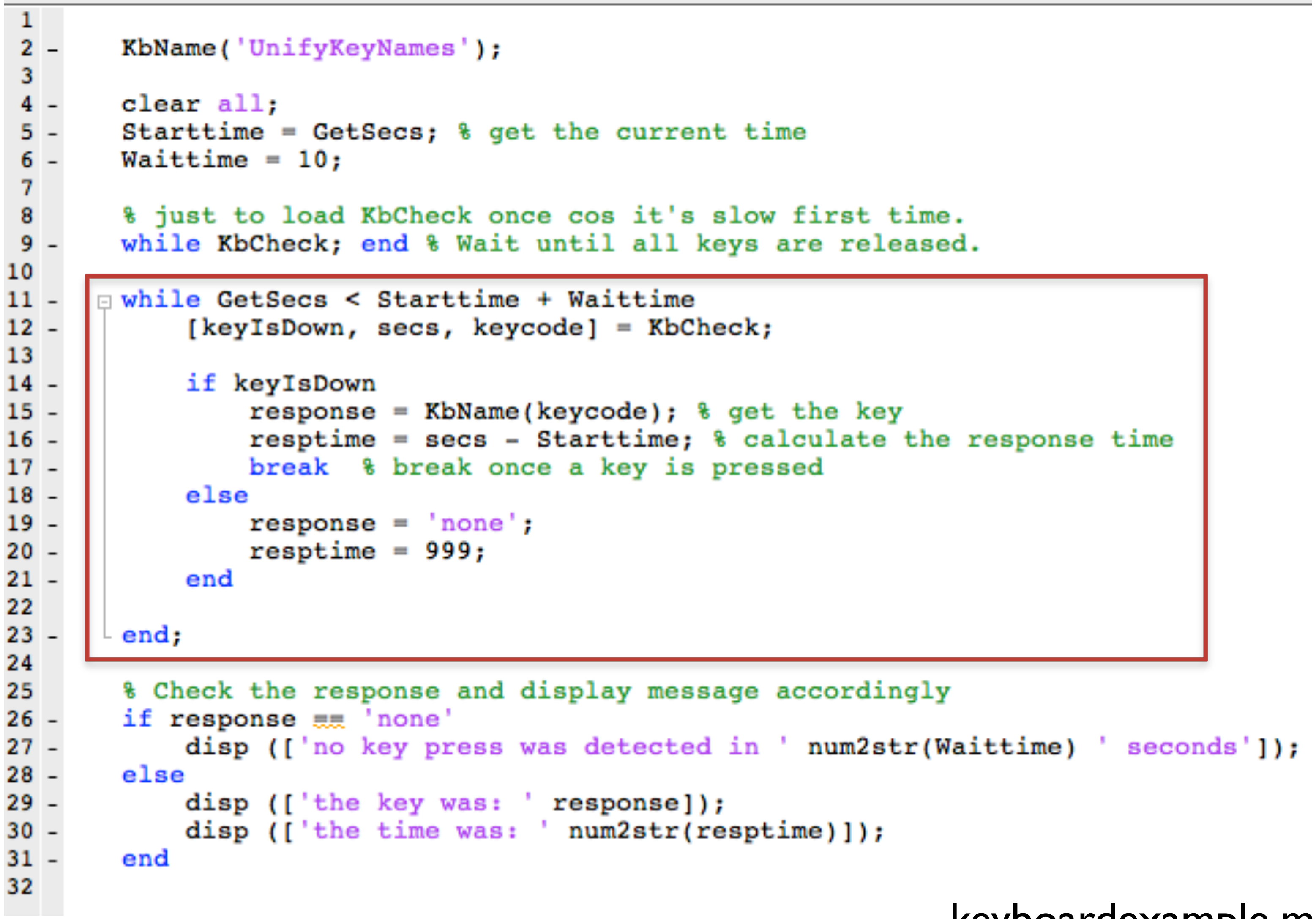
Simple program to get key presses



```matlab

KbName('UnifyKeyNames');

clear all;
Starttime = GetSecs; % get the current time
Waittime = 10;

% just to load KbCheck once cos it's slow first time.
while KbCheck; end % Wait until all keys are released.

while GetSecs < Starttime + Waittime
    [keyIsDown, secs, keycode] = KbCheck;

    if keyIsDown
        response = KbName(keycode); % get the key
        resptime = secs - Starttime; % calculate the response time
        break  % break once a key is pressed
    else
        response = 'none';
        resptime = 999;
    end

end;

% Check the response and display message accordingly
if response == 'none'
    disp (['no key press was detected in ' num2str(Waittime) ' seconds']);
else
    disp (['the key was: ' response]);
    disp (['the time was: ' num2str(resptime)]);
end

```

keyboardexample.m

Simple program to get key presses



```

KbName('UnifyKeyNames');

clear all;
Starttime = GetSecs; % get the current time
Waittime = 10;

% just to load KbCheck once cos it's slow first time.
while KbCheck; end % Wait until all keys are released.

while GetSecs < Starttime + Waittime
    [keyIsDown, secs, keycode] = KbCheck;

    if keyIsDown
        response = KbName(keycode); % get the key
        resptime = secs - Starttime; % calculate the response time
        break  % break once a key is pressed
    else
        response = 'none';
        resptime = 999;
    end

end;

% Check the response and display message accordingly
if response == 'none'
    disp (['no key press was detected in ' num2str(Waittime) ' seconds']);
else
    disp (['the key was: ' response]);
    disp (['the time was: ' num2str(resptime)]);
end

```

keyboardexample.m

FlushEvents function

```
>> help FlushEvents
```

When you type into keyboard (or any other input device), the responses are saved into a store called event queue. When you type really fast, if the computer is busy with some other tasks, there will be a pause before the characters suddenly appear on screen.

It is therefore important to empty events from the event queue before collecting responses, especially with `GetChar` since it reads from the event queue.

`FlushEvents` removes events from the system event queue.

Calling KbCheck to clear

- If a key is pressed long there may still be elements in the event queue after FlushEvents
- You can use this:

```
    while KbCheck; end
```

Keyboard and File

- Now, let's get key presses and write them into a text file together with the time information.
- Download **keyboardexamplefile.m** from the class website and run it.
- Check the output file: keyboarddata.txt

Keyboard use with Screen

- Download **keyboardexamplescreen.m** file from the class website.

- This is a simple experiment in Psychtoolbox that displays an image either upright or inverted in each trial (for a total of 6 trials), gets subject's response ('u' or 'd' for correct responses), calculates reaction time and accuracy, and writes the results in a text file.

- Run it and check the output file image_output.txt

- Let's now closely look at the **keyboardexamplescreen.m**

Exercise

- Read Fine & Boynton Chapter 12 and go over the exercises in the chapter in Matlab. → Note: This is on TED and not on the class website.
- See also Schneider notes pdf on TED.