```
classdef mcMemeb_mfile < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                     matlab.ui.Figure
        GridLayout                   matlab.ui.container.GridLayout
        LeftPanel                    matlab.ui.container.Panel
        Label                        matlab.ui.control.Label
        UIAxes                       matlab.ui.control.UIAxes
        UIAxes2                      matlab.ui.control.UIAxes
        RecordStatusLampLabel        matlab.ui.control.Label
        RecordStatusLamp             matlab.ui.control.Lamp
        CenterPanel                  matlab.ui.container.Panel
        StartGraphingButton          matlab.ui.control.Button
        StopButton                   matlab.ui.control.Button
        RecordLengthEditFieldLabel   matlab.ui.control.Label
        RecordLengthEditField        matlab.ui.control.NumericEditField
        RecordButton                 matlab.ui.control.Button
        ControlsLabel                matlab.ui.control.Label
        smoothButton                 matlab.ui.control.Button
        RightPanel                   matlab.ui.container.Panel
        Label_2                      matlab.ui.control.Label
        fry                          matlab.ui.control.Button
        DataProcessingLabel          matlab.ui.control.Label
        MeanLabel                    matlab.ui.control.Label
        MaxLabel                     matlab.ui.control.Label
        MinLabel                     matlab.ui.control.Label
        StdevLabel                   matlab.ui.control.Label
        MeanLabeldata                matlab.ui.control.Label
        MaxLabeldata                 matlab.ui.control.Label
        MinLabeldata                 matlab.ui.control.Label
        Stdevdata                    matlab.ui.control.Label
        PresentedbyLabel             matlab.ui.control.Label
        TeamLightningMcMemeLabel     matlab.ui.control.Label
    end

    % Properties that correspond to apps with auto-reflow
    properties (Access = private)
        onePanelWidth = 576;
        twoPanelWidth = 768;
    end

    % Team Lightning McMeme
    % Abdallah Hashem, Upamanyu Kashyap, Clay Crisafulli


    properties (Access = private)
        a; % arduino
        stop = true; % boolean that determines if data
        % collection starts or stops
        v; %voltage reading from arduino sensor
        vdata; % array of voltage  data used for recording
        fri; % easter egg :)
    end



    % Callbacks that handle component events
    methods (Access = private)

        % Code that executes after component creation
        function startupFcn(app)
            app.a = arduino(); %loads arduino APIs
            app.RecordStatusLamp.Color = [0,0,0]

        end

        % Button pushed function: StartGraphingButton
        function StartGraphingButtonPushed(app, event)

            cla(app.UIAxes); %clears current plot on UIAxes (reset)
            h = animatedline(app.UIAxes); % defines object
            app.stop = false; %starts data collection
            startTime = datetime('now'); % defines start time
            while ~app.stop % until stop button is pressed
                app.v = readVoltage(app.a,'A0'); % voltage read
                app.v = 500/(10*((5-app.v)/app.v)); % converting
voltage to lumens
                t =  datetime('now') - startTime; % get current time
                addpoints(h,datenum(t),app.v) % add points to
animation
                app.UIAxes.XLim = datenum([t-seconds(15) t]); % moves
axis
                datetick('x','keeplimits') % date markers
                drawnow % updates plot
            end
        end

        % Button pushed function: StopButton
        function StopButtonPushed(app, event)
            app.stop = true; % stops data collection

        end

        % Value changed function: RecordLengthEditField
        function RecordLengthEditFieldValueChanged(app, event)
            value = app.RecordLengthEditField.Value;
            %{
            edit field allows user to define the amount of time
            that the program records data for
            %}

        end

        % Button pushed function: RecordButton
        function RecordButtonPushed(app, event)
```

```
            cla(app.UIAxes2); %clears current plot on UIAxes (reset)
            app.RecordStatusLamp.Color = [0,1,0]

            PushAndRadioButtons = 
[findall(gcf,'Style','Pushbutton');findall(gcf,'Style','radiobutton')];
            % Change to red all these buttons
            set(PushAndRadioButtons,'Backgroundcolor','r');
            % Set to red the current button
            set(gcbo,'Backgroundcolor','r');


            h = animatedline(app.UIAxes2); % creates animated line on
            % second axes object
            app.stop = false; % starts data collection
            startTime = datetime('now'); % sets start time
            tic %internal stopwatch start
            while toc+0.3 < app.RecordLengthEditField.Value %time 
interval picked by user
                app.v = readVoltage(app.a,'A0');
                app.v = 500/(10*((5-app.v)/app.v));
                app.vdata = [app.vdata app.v]; % adds current reading 
to array
                t =  datetime('now') - startTime;
                addpoints(h,datenum(t),app.v)
                app.UIAxes.XLim = datenum([t-seconds(15) t]);
                datetick('x','keeplimits')
                drawnow
            end

            app.RecordStatusLamp.Color = [0,0,0]

            PushAndRadioButtons = 
[findall(gcf,'Style','Pushbutton');findall(gcf,'Style','radiobutton')];
            % Change to red all these buttons
            set(PushAndRadioButtons,'Backgroundcolor','r');
            % Set to red the current button
            set(gcbo,'Backgroundcolor','blue');


            % calculating the mean of data then writing it to the Mean 
label
            datamean = mean(app.vdata);
            app.MeanLabeldata.Text = num2str(datamean);

            % calculating the max of data then writing it to the Max 
label
            datamax = max(app.vdata);
            app.MaxLabeldata.Text = num2str(datamax);

            % calculating the min of data then writing it to the Min 
label
            datamin = min(app.vdata);
            app.MinLabeldata.Text = num2str(datamin);
            % calculating the Std of data then writing it to the Std
label
            datastd = std(app.vdata)*100;
            app.Stdevdata.Text = num2str(datastd);




        end

        % Callback function
        function TextAreaValueChanged(app, event)
            value = mean(app.vdata);


        end

        % Callback function
        function MeanButtonPushed(app, event)


            datam = mean(app.vdata);


            app.Label.Text = num2str(datam);


        end

        % Button pushed function: smoothButton
        function smoothButtonPushed(app, event)


            cla(app.UIAxes); %clears current plot on UIAxes (reset)
            h = animatedline(app.UIAxes);
            app.stop = false;
            startTime = datetime('now');
            while ~app.stop
                % Get current time
                app.v = readVoltage(app.a,'A0');
                app.v = smoothdata(500/(10*((5-app.v)/app.v)));
                t =  datetime('now') - startTime;

                % Add points to animation

                addpoints(h,datenum(t),app.v)
                % Update axes
                app.UIAxes.XLim = datenum([t-seconds(15) t]);

                datetick('x','keeplimits')
                drawnow
            end
```

```
end

% Callback function
function SliderValueChanging(app, event)
    changingValue = event.Value;




end

% Callback function
function medianButtonPushed(app, event)

    h = animatedline(app.UIAxes2);
    app.stop = false;
    startTime = datetime('now');
    while ~app.stop
        % Get current time
        app.v = readVoltage(app.a,'A0');
        app.v = smoothdata(500/(10*((5-app.v)/app.v)));
        t =  datetime('now') - startTime;



        % Add points to animation

        addpoints(h,datenum(t),app.v)
        % Update axes
        app.UIAxes.XLim = datenum([t-seconds(15) t]);

        medreal = movmedian(app.v, [0 15])

        addpoints(h,datenum(t),medreal)

        datetick('x','keeplimits')
        drawnow
    end


end

% Callback function
function PeakButtonPushed(app, event)

peakdata = findpeaks(app.vdata)

app.Label_2.Text = num2str(peakdata);

end

% Button pushed function: fry
function fryPushed(app, event)
```

```
                PushAndRadioButtons =
[findall(gcf,'Style','Pushbutton');findall(gcf,'Style','radiobutton')];
               % Change to red all these buttons
                set(PushAndRadioButtons,'Backgroundcolor','r');
                % Set to green the current button
                set(gcbo,'Backgroundcolor','r');

                app.RecordStatusLamp.Color = [1,1,0]
                l = 0
                while l < 10

                   % for j = 1:20
                    %    figure(j)
                     %   j = j+1;
                   % end

                    for po = 1:20
                        figure(po)
                       imshow('imagee.png')
                    end


                    h = animatedline(app.UIAxes2);
                    app.stop = false;
                    startTime = datetime('now');

                    while ~app.stop
                        % Get current time

                        app.fri = randi(2000,10)
                        t =  datetime('now') - startTime;



                        % Add points to animation

                        %addpoints(h,datenum(t),app.v)

                        plot(app.fri);
                        % Update axes
                        app.UIAxes.XLim = datenum([t-seconds(15) t]);

                        datetick('x','keeplimits')
                        drawnow
                    end
                    l = l+1;
                end
            end

            % Callback function
            function SliderValueChanged(app, event)
                value = app.Slider.Value;
                value = app.vdata ./ 10;
                rvalue = round(value);
                app.TeamLightningMcMemeLabel.Text = num2str(rvalue);




            end

            % Callback function
            function UITableDisplayDataChanged(app, event)
                newDisplayData = app.UITable.DisplayData;


            end

            % Changes arrangement of the app based on UIFigure width
            function updateAppLayout(app, event)
                currentFigureWidth = app.UIFigure.Position(3);
                if(currentFigureWidth <= app.onePanelWidth)
                    % Change to a 3x1 grid
                    app.GridLayout.RowHeight = {663, 663, 663};
                    app.GridLayout.ColumnWidth = {'1x'};
                    app.CenterPanel.Layout.Row = 1;
                    app.CenterPanel.Layout.Column = 1;
                    app.LeftPanel.Layout.Row = 2;
                    app.LeftPanel.Layout.Column = 1;
                    app.RightPanel.Layout.Row = 3;
                    app.RightPanel.Layout.Column = 1;
                elseif (currentFigureWidth > app.onePanelWidth &&
    currentFigureWidth <= app.twoPanelWidth)
                    % Change to a 2x2 grid
                    app.GridLayout.RowHeight = {663, 663};
                    app.GridLayout.ColumnWidth = {'1x', '1x'};
                    app.CenterPanel.Layout.Row = 1;
                    app.CenterPanel.Layout.Column = [1,2];
                    app.LeftPanel.Layout.Row = 2;
                    app.LeftPanel.Layout.Column = 1;
                    app.RightPanel.Layout.Row = 2;
                    app.RightPanel.Layout.Column = 2;
                else
                    % Change to a 1x3 grid
                    app.GridLayout.RowHeight = {'1x'};
                    app.GridLayout.ColumnWidth = {451, '1x', 365};
                    app.LeftPanel.Layout.Row = 1;
                    app.LeftPanel.Layout.Column = 1;
                    app.CenterPanel.Layout.Row = 1;
                    app.CenterPanel.Layout.Column = 2;
                    app.RightPanel.Layout.Row = 1;
                    app.RightPanel.Layout.Column = 3;
                end
            end
        end

        % Component initialization
        methods (Access = private)

            % Create UIFigure and components
            function createComponents(app)

                % Create UIFigure and hide until all components are
    created
                app.UIFigure = uifigure('Visible', 'off');
                app.UIFigure.AutoResizeChildren = 'off';
                app.UIFigure.Position = [100 100 999 663];
                app.UIFigure.Name = 'UI Figure';
                app.UIFigure.SizeChangedFcn = createCallbackFcn(app,
    @updateAppLayout, true);

                % Create GridLayout
                app.GridLayout = uigridlayout(app.UIFigure);
                app.GridLayout.ColumnWidth = {451, '1x', 365};
                app.GridLayout.RowHeight = {'1x'};
                app.GridLayout.ColumnSpacing = 0;
                app.GridLayout.RowSpacing = 0;
                app.GridLayout.Padding = [0 0 0 0];
                app.GridLayout.Scrollable = 'on';

                % Create LeftPanel
                app.LeftPanel = uipanel(app.GridLayout);
                app.LeftPanel.BackgroundColor = [1 1 1];
                app.LeftPanel.Layout.Row = 1;
                app.LeftPanel.Layout.Column = 1;

                % Create Label
                app.Label = uilabel(app.LeftPanel);
                app.Label.Position = [34 233 127 22];
                app.Label.Text = '';

                % Create UIAxes
                app.UIAxes = uiaxes(app.LeftPanel);
                title(app.UIAxes, 'Live')
                xlabel(app.UIAxes, 'Time')
                ylabel(app.UIAxes, 'Lumens')
                app.UIAxes.PlotBoxAspectRatio = [1.84182305630027 1 1];
                app.UIAxes.Position = [11 349 430 271];

                % Create UIAxes2
                app.UIAxes2 = uiaxes(app.LeftPanel);
                title(app.UIAxes2, 'Recorded Data')
                xlabel(app.UIAxes2, 'Time')
                ylabel(app.UIAxes2, 'Lumens')
                app.UIAxes2.PlotBoxAspectRatio = [1.94573643410853 1 1];
                app.UIAxes2.Position = [11 55 430 271];

                % Create RecordStatusLampLabel
                app.RecordStatusLampLabel = uilabel(app.LeftPanel);
                app.RecordStatusLampLabel.HorizontalAlignment = 'right';
                app.RecordStatusLampLabel.Position = [313 304 83 22];
                app.RecordStatusLampLabel.Text = 'Record Status';

                % Create RecordStatusLamp
                app.RecordStatusLamp = uilamp(app.LeftPanel);
                app.RecordStatusLamp.Position = [411 304 20 20];

                % Create CenterPanel
                app.CenterPanel = uipanel(app.GridLayout);
                app.CenterPanel.BackgroundColor = [1 1 1];
                app.CenterPanel.Layout.Row = 1;
                app.CenterPanel.Layout.Column = 2;

                % Create StartGraphingButton
                app.StartGraphingButton = 
     uibutton(app.CenterPanel, 'push');
                app.StartGraphingButton.ButtonPushedFcn = 
     createCallbackFcn(app, @StartGraphingButtonPushed, true);
                app.StartGraphingButton.BackgroundColor = [0.3961 0.7412 
     0.2235];
                app.StartGraphingButton.FontSize = 20;
                app.StartGraphingButton.FontColor = [1 1 1];
                app.StartGraphingButton.Position = [8 539 170 47];
                app.StartGraphingButton.Text = 'Start Graphing';

                % Create StopButton
                app.StopButton = uibutton(app.CenterPanel, 'push');
                app.StopButton.ButtonPushedFcn = createCallbackFcn(app, 
     @StopButtonPushed, true);
                app.StopButton.BackgroundColor = [0.851 0.1451 0.1451];
                app.StopButton.FontSize = 20;
                app.StopButton.FontColor = [1 1 1];
                app.StopButton.Position = [7 461 169 47];
                app.StopButton.Text = 'Stop';

                % Create RecordLengthEditFieldLabel
                app.RecordLengthEditFieldLabel = uilabel(app.CenterPanel);
                app.RecordLengthEditFieldLabel.HorizontalAlignment 
     = 'right';
                app.RecordLengthEditFieldLabel.Position = [7 254 85 22];
                app.RecordLengthEditFieldLabel.Text = 'Record Length';

                % Create RecordLengthEditField
                app.RecordLengthEditField = 
     uieditfield(app.CenterPanel, 'numeric');
                app.RecordLengthEditField.ValueChangedFcn = 
     createCallbackFcn(app, @RecordLengthEditFieldValueChanged, true);
                app.RecordLengthEditField.Position = [107 254 71 22];

                % Create RecordButton
                app.RecordButton = uibutton(app.CenterPanel, 'push');
                app.RecordButton.ButtonPushedFcn = createCallbackFcn(app, 
     @RecordButtonPushed, true);
                app.RecordButton.BackgroundColor = [0.1373 0.298 0.6784];
                app.RecordButton.FontColor = [1 1 1];
                app.RecordButton.Position = [14 193 157 31];
                app.RecordButton.Text = {'Record'; ''};

                % Create ControlsLabel
                app.ControlsLabel = uilabel(app.CenterPanel);
                app.ControlsLabel.FontName = 'PT Serif';
                app.ControlsLabel.FontSize = 30;
                app.ControlsLabel.Position = [33 605 120 43];
                app.ControlsLabel.Text = 'Controls';

                % Create smoothButton
                app.smoothButton = uibutton(app.CenterPanel, 'push');
                app.smoothButton.ButtonPushedFcn = createCallbackFcn(app,
     @smoothButtonPushed, true);
                app.smoothButton.BackgroundColor = [0.1373 0.298 0.6784];
                app.smoothButton.FontSize = 20;
                app.smoothButton.FontColor = [1 1 1];
                app.smoothButton.Position = [8.5 387 168 46];
                app.smoothButton.Text = {'smooth'; ''};

                % Create RightPanel
                app.RightPanel = uipanel(app.GridLayout);
                app.RightPanel.BackgroundColor = [1 1 1];
                app.RightPanel.Layout.Row = 1;
                app.RightPanel.Layout.Column = 3;

                % Create Label_2
                app.Label_2 = uilabel(app.RightPanel);
                app.Label_2.Position = [64 440 70 22];
                app.Label_2.Text = '';

                % Create fry
                app.fry = uibutton(app.RightPanel, 'push');
                app.fry.ButtonPushedFcn = createCallbackFcn(app,
     @fryPushed, true);
                app.fry.Icon = 'Screen Shot 2019-06-04 at 8.57.16 PM.png';
                app.fry.IconAlignment = 'center';
                app.fry.BackgroundColor = [1 1 1];
                app.fry.Position = [279 1 73 67];
                app.fry.Text = '';

                % Create DataProcessingLabel
                app.DataProcessingLabel = uilabel(app.RightPanel);
                app.DataProcessingLabel.FontSize = 30;
                app.DataProcessingLabel.Position = [69 605 228 39];
                app.DataProcessingLabel.Text = 'Data Processing';

                % Create MeanLabel
                app.MeanLabel = uilabel(app.RightPanel);
                app.MeanLabel.BackgroundColor = [0 0 0];
                app.MeanLabel.HorizontalAlignment = 'center';
                app.MeanLabel.FontColor = [1 1 1];
                app.MeanLabel.Position = [11 440 86 22];
                app.MeanLabel.Text = {'Mean'; ''};
                % Create MaxLabel
                app.MaxLabel = uilabel(app.RightPanel);
                app.MaxLabel.BackgroundColor = [0 0 0];
                app.MaxLabel.HorizontalAlignment = 'center';
                app.MaxLabel.FontColor = [1 1 1];
                app.MaxLabel.Position = [96 440 86 22];
                app.MaxLabel.Text = 'Max';

                % Create MinLabel
                app.MinLabel = uilabel(app.RightPanel);
                app.MinLabel.BackgroundColor = [0 0 0];
                app.MinLabel.HorizontalAlignment = 'center';
                app.MinLabel.FontColor = [1 1 1];
                app.MinLabel.Position = [181 440 86 22];
                app.MinLabel.Text = 'Min';

                % Create StdevLabel
                app.StdevLabel = uilabel(app.RightPanel);
                app.StdevLabel.BackgroundColor = [0 0 0];
                app.StdevLabel.HorizontalAlignment = 'center';
                app.StdevLabel.FontColor = [1 1 1];
                app.StdevLabel.Position = [266 440 86 22];
                app.StdevLabel.Text = 'Stdev';

                % Create MeanLabeldata
                app.MeanLabeldata = uilabel(app.RightPanel);
                app.MeanLabeldata.BackgroundColor = [0.9412 0.9412
     0.9412];

                app.MeanLabeldata.HorizontalAlignment = 'center';
                app.MeanLabeldata.Position = [11 419 86 22];
                app.MeanLabeldata.Text = {'Mean'; ''};

                % Create MaxLabeldata
                app.MaxLabeldata = uilabel(app.RightPanel);
                app.MaxLabeldata.BackgroundColor = [0.9412 0.9412 0.9412];
                app.MaxLabeldata.HorizontalAlignment = 'center';
                app.MaxLabeldata.Position = [96 419 86 22];
                app.MaxLabeldata.Text = {'Mean'; ''};

                % Create MinLabeldata
                app.MinLabeldata = uilabel(app.RightPanel);
                app.MinLabeldata.BackgroundColor = [0.9412 0.9412 0.9412];
                app.MinLabeldata.HorizontalAlignment = 'center';
                app.MinLabeldata.Position = [181 419 86 22];
                app.MinLabeldata.Text = {'Mean'; ''};

                % Create Stdevdata
                app.Stdevdata = uilabel(app.RightPanel);
                app.Stdevdata.BackgroundColor = [0.9412 0.9412 0.9412];
                app.Stdevdata.HorizontalAlignment = 'center';
                app.Stdevdata.Position = [266 419 86 22];
                app.Stdevdata.Text = {'Mean'; ''};
                % Create PresentedbyLabel
                app.PresentedbyLabel = uilabel(app.RightPanel);
                app.PresentedbyLabel.HorizontalAlignment = 'center';
                app.PresentedbyLabel.FontSize = 30;
                app.PresentedbyLabel.Position = [91 275 183 39];
                app.PresentedbyLabel.Text = 'Presented by';

                % Create TeamLightningMcMemeLabel
                app.TeamLightningMcMemeLabel = uilabel(app.RightPanel);
                app.TeamLightningMcMemeLabel.BackgroundColor = [1 1 0];
                app.TeamLightningMcMemeLabel.HorizontalAlignment
     = 'center';
                app.TeamLightningMcMemeLabel.FontName = 'Comic Sans MS';
                app.TeamLightningMcMemeLabel.FontSize = 28;
                app.TeamLightningMcMemeLabel.FontWeight = 'bold';
                app.TeamLightningMcMemeLabel.FontAngle = 'italic';
                app.TeamLightningMcMemeLabel.FontColor = [1 0 1];
                app.TeamLightningMcMemeLabel.Position = [8 206 344 43];
                app.TeamLightningMcMemeLabel.Text = 'Team Lightning
     McMeme';

                % Show the figure after all components are created
                app.UIFigure.Visible = 'on';
            end
        end

        % App creation and deletion
        methods (Access = public)

            % Construct app
            function app = mcMemeb_mfile

                % Create UIFigure and components
                createComponents(app)

                % Register the app with App Designer
                registerApp(app, app.UIFigure)

                % Execute the startup function
                runStartupFcn(app, @startupFcn)

                if nargout == 0
                    clear app
                end
            end

            % Code that executes before app deletion
            function delete(app)

                % Delete UIFigure when app is deleted
                delete(app.UIFigure)
            end
        end
    end
```

```
Error using mcMemeb_mfile/startupFcn (line 63)
Cannot detect Arduino hardware. Make sure Arduino hardware is properly
 plugged in.

Error in mcMemeb_mfile (line 603)
            runStartupFcn(app, @startupFcn)
```

Published with MATLAB® R2019a