```matlab
classdef class_gigecam < handle
    %CLASS_GIGECAM Communication with a camera via gigecam
    %
    properties (SetAccess = immutable)
        name = 'NAME';      % an human readable identifier
        format = '';        % format for gigecam function
        model = '';         % manufacturer like in gigecamlist().Model[i]
        manufacturer = ''; % manufacturer like in gigecamlist().Manufacturer[i]
        ip = false;         % IP-Address
    end

    properties (Access = private)
        handle = false;     % raw handle of the camera connection
    end

    methods
        % constructor
        function obj = class_gigecam(name, format, model, manufacturer, list)
            obj.name = name;
            obj.model = model;
            obj.manufacturer = manufacturer;
            obj.format = format;

            if nargin < 5
                list = gigecamlist();
            end

            % find camera index
            index = -1;
            [model_i, ~] = find(strcmp(list.Model, model));
            [manuf_i, ~] = find(strcmp(list.Manufacturer, manufacturer));
            i = intersect(manuf_i, model_i);

            if size(i, 1) == 1
                index = i;
            end;

            % return if camera is not found
            if index == -1
                error('Can''t find gigecam %s. (Manufacturer: %s, Model: %s)', name, manufacturer, model);
            end

            % initialize camera
            obj.ip = list.IPAddress{index, 1};
            fprintf('Found gigecam camera %s at IP %s.\n', name, obj.ip);
        end

        % connect to the camera
        function success = connect(obj)
            success = false;
            if obj.handle ~= false
                success = true;
                return;
            end

            if obj.ip == false
                warning('Can''t connect to %s, don''t know IP.', obj.name);
                return;
            end

            try
                obj.handle = gigecam(obj.ip, 'PixelFormat', obj.format);
                fprintf('Connected to %s\n', obj.name);
                success = true;
            catch e
                warning('Connecting to %s failed: %s', obj.name, getReport(e));
            end
        end

        % close camera connection
        function close(obj)
            if obj.handle ~= false
                clear obj.handle;
                obj.handle = false;
            end
            fprintf('Closed connection to %s\n', obj.name);
        end

        % Call config_function with handle if connected
        function [success, varargout] = config(obj, config_function)
            success = false;
            varargout = cell(1, nargout - 1);

            % warn and return if connecting failes
            if obj.connect() == false
                warning('Config error: Can''t connect to camera %s.', obj.name);
                return;
            end

            try
                if isa(config_function, 'function_handle')
                    [obj.handle, varargout{1:nargout - 1}] = config_function(obj.handle);
                elseif iscell(config_function) && isa(config_function{1}, 'function_handle')
                    [obj.handle, varargout{1:nargout - 1}] = config_function{1}(obj.handle, config_function{2:end});
                else
                    error('gigecam config callback is not callable');
                end

                success = true;
            catch e
                warning('Exception while config %s: %s', obj.name, getReport(e));
            end
        end

        % preview live image
        function success = preview(obj, adjust_function, axes)
            success = false;

            % warn and return if connecting failes
            if obj.connect() == false
                warning('Preview error: Can''t connect to camera %s.', obj.name);
                return;
            end

            try
                res = obj.handle;

                if nargin < 2
                    % preview as figure
                    hImage = image(zeros(res.Height, res.Width, 1));
                else
                    % preview on axes in GUI
                    hImage = image(zeros(res.Height, res.Width, 1), 'Parent', axes);
                    axes.DataAspectRatio = [1, 1, 1];
                end

                if isa(adjust_function, 'function_handle')
```

```
                    setappdata(hImage,
                    'UpdatePreviewWindowFcn', adjust_function);
                end

                preview(obj.handle, hImage);

                success = true;
            catch e
                warning('Exception while preview %s: %s',
                obj.name, getReport(e));
            end
        end

        % stop preview live image
        function success = stoppreview(obj)
            success = false;
            if obj.handle == false
                return;
            end

            try
                closePreview(obj.handle);
                success = true;
            catch e
                warning('Exception while preview %s: %s',
                obj.name, getReport(e));
            end
        end

        function [success, img] = snapshot(obj,
        adjust_function, axes)
            success = false;

            try
                if nargin < 2
                    adjust_function = false;
                end

                img = false;

                obj.stoppreview();

                % warn and return if connecting failes
                if obj.connect() == false
                    warning('Snapshot error: Can''t connect to
                    camera %s.', obj.name);
                    return;
                end

                img = snapshot(obj.handle);

                if isa(adjust_function, 'function_handle')
                    img =  adjust_function(img);
                end

                if nargin > 2
                    imshow(img, 'Parent', axes);
                end

                success = true;
            catch e
                warning('Exception while snapshot %s: %s', obj.name, getReport(e));
            end
        end

        function delete(obj)
            obj.close();
        end
    end

end

```