```matlab
classdef class_videoinput < handle
    %CLASS_WINVIDEO Communication with a camera via videoinput
    %
    properties (SetAccess = immutable)
        name = 'NAME';     % an human readable identifier
        type = '';         % videoinput type
        device_name = '';  % like in imaqhwinfo(type, id).DeviceName
        format = '';       % format for videoinput function
        color_space = '';  % ReturnedColorSpace
        id = false;        % ID-Number
    end

    properties (Access = private)
        handle = false;    % raw handle of the camera connection
        prev_timer = false;% timer for async snapshot preview
    end

    methods
        % constructor
        function obj = class_videoinput(name, type, format, color_space, device_name)
            obj.name = name;
            obj.type = type;
            obj.format = format;
            obj.color_space = color_space;
            obj.device_name = device_name;

            % find camera id
            cam_id = -1;
            info = imaqhwinfo(obj.type);
            for i = info.DeviceIDs
                dev_info = imaqhwinfo(obj.type, i{1});
                if strcmp(dev_info.DeviceName, device_name)
                    cam_id = i{1};
                end
            end

            % return if device_name is not found
            if cam_id == -1
                error('Can''t find camera %s by DeviceName "%s"', name, device_name);
            end

            % initialize camera
            obj.id = cam_id;
            fprintf('Found videoinput camera %s at ID %d.\n', name, cam_id);
        end

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

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

            try
                obj.handle = videoinput(obj.type, obj.id, obj.format);
                obj.handle.ReturnedColorSpace = obj.color_space;
                fprintf('Connected to %s\n', obj.name);
                success = true;
            catch e
                warning('Connecting to %s failed: videoinput(''%s'', %d, ''%s''). %s', obj.type, obj.name, obj.id, obj.format, getReport(e));
            end
        end

        % close camera connection
        function close(obj)
            if obj.handle ~= false
                delete(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('videoinput 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, rotate)
            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
                size = obj.handle.ROIPosition;
                bands = obj.handle.NumberOfBands;

                if nargin > 3 && rotate == true
                    size([3, 4]) = size([4, 3]);
                end

                if nargin < 3
                    % preview as figure
                    hImage = image(zeros(size(4), size(3), bands));
                else
                    % preview on axes in GUI
                    hImage = image(zeros(size(4), size(3),
                    bands), 'Parent', axes);
                    axes.DataAspectRatio = [1, 1, 1];
                end

                if nargin > 1 && 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
                if obj.prev_timer ~= false
                    stop(obj.prev_timer);
                    delete(obj.prev_timer);
                    obj.prev_timer = false;
                end

                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 = getsnapshot(obj.handle);

                if nargin > 2
                    if isa(adjust_function, 'function_handle')
                        img = adjust_function(img);
                    end

                    imshow(img, 'Parent', axes);
                else
                    imshow(img);
                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

```