

# NeuroElf Resources

## Online resources

<http://neuroelf.net/>

<http://neuroelf.net/wiki/doku.php>

<https://www.youtube.com/channel/UC1sM-zqnmdcZOrRf-i0FLGg> | NeuroElf youtube tutorials!

## Help

For getting detailed help on each method of xff object (e.g. fmr), use:

```
>> fmr.Help % to see all methods
>> fmr.Help('Method') % e.g. fmr.Help('Realign') to see details of each method
```

See <http://neuroelf.net/wiki/doku.php?id=obj.help>

## NeuroElf-based ERA

[http://neuroelf.net/wiki/doku.php?id=neuroelf\\_gui\\_-\\_mdm\\_voi\\_condition\\_average\\_ui](http://neuroelf.net/wiki/doku.php?id=neuroelf_gui_-_mdm_voi_condition_average_ui)

<http://neuroelf.net/wiki/doku.php?id=mdm.voicondaverage>

## DAG installation notes

1) During installation, disable (remove from path) Chronux (if present): conflict with linec.m

2) NeuroElf functions use std.m and var.m as in newer versions of MATLAB, with 3 inputs: e.g. Y = STD(X,FLAG,DIM). Therefore, old NaN toolbox functions std.m and var.m were incompatible and sterr.m was updated. The old version of std.m is now available as ig\_std.m in Igtools (corrcoef\_eval.m has not been revised yet but is not needed for running the pipeline).

3) Starting from NeuroElf\_v09d, many useful auxiliary scripts/functions are “hidden” behind @neuroelf method, and reside in `..\@neuroelf\private`. They can be placed in the path (i.e. copied to the main NeuroElf folder, it is not possible to add to path this private folder directly), or accessed as described in [main installation folder] / README.html > Function library:

```
>> netools = neuroelf;
>> list_of_files = netools.findfiles(startfolder, pattern);
```



The functions below should be in the NeuroElf base (root) folder (e.g. D:\Sources\NeuroElf\_v10\_4503). **Only** this folder should be on the path!



4) There is a bug in NeuroElf v1.0 and v0.9d, to fix please replace the function dcm\_Value.m in /xff/private with the function below. This function is used by createfmr.m

#### dcm\_Value.m

```
function dcmval = dcm_Value(hfile, vkey, varargin)
% DCM::Value - return a tag's value
%
% FORMAT:      dcmval = dcm.Value(key [, ...]);
%
% Input fields:
%
%      key      DICOM key (0008.0010) or tag (PatientsName)
%      ...      default value if not found
%
% Output fields:
%
%      dcmval    value for given key/tag
%
% Using: makelabel, splittocell.

% Version:  v0.9d
% Build:    14082218
% Date:     Aug-22 2014, 6:14 PM EST
% Author:   Jochen Weber, SCAN Unit, Columbia University, NYC, NY, USA
% URL/Info: http://neuroelf.net/
%
% Copyright (c) 2010, 2011, 2014, Jochen Weber
% All rights reserved.
%
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions
% are met:
%      * Redistributions of source code must retain the above copyright
%      notice, this list of conditions and the following disclaimer.
%      * Redistributions in binary form must reproduce the above
%      copyright
%      notice, this list of conditions and the following disclaimer in
%      the
%      documentation and/or other materials provided with the
%      distribution.
%      * Neither the name of Columbia University nor the
%      names of its contributors may be used to endorse or promote
%      products
%      derived from this software without specific prior written
```

```

permission.
%
% THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND
% ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED
% WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE
% DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS BE LIABLE FOR ANY
% DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES
% (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
% LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND
% ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
TORT
% (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS
% SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

% neuroelf library
global ne_methods;

% persistent dictionaries
persistent dcmv_dicts
if isempty(dcmv_dicts)
    dcmv_dicts = struct;

    % load standard dict
    dcmv_dicts.OFFIS = ne_methods.dicom_dic(); % IK added ne_methods
and ()

end

% only valid for single file
if nargin < 2 || ...
    numel(hfile) ~= 1 || ...
    ~xfisobject(hfile, true, 'dcm')
    error( ...
        'xff:BadArgument', ...
        'Invalid call to %s.', ...
        mfilename ...
    );
end

% dict
bc = xffgetcont(hfile.L);
dict = bc.DataDictionary;

% check key

```

```
if ischar(vkey)
    vkey = vkey(:)';
    if length(vkey) == 9 && ...
        ~isempty(regexp(vkey, '^([0-9a-f]{4})([0-9a-f]{4})$'))
        vkey = vkey([1:4,6:9]);
    end
    if length(vkey) == 8 && ...
        ~isempty(regexp(vkey, '^([0-9a-f]+)$')) && ...
        ~all(double(vkey) > 64)
        vkey = ['k_' upper(vkey(1:4)) '_' upper(vkey(5:8))];
    elseif ~strcmp(vkey, ne_methods.makelabel(vkey))
        error( ...
            'xff:BadArgument', ...
            'Invalid DICOM tag number/key given.' ...
        );
    else
        if isempty(dict)
            warning( ...
                'xff:InternalWarning', ...
                'Data dictionary not looked up.' ...
            );
            dcm_DetectDictionary(hfile, 'auto');
            dict = bc.DataDictionary;
        end
        if ~isfield(dcmv_dicts, dict)
            error( ...
                'xff:BadSetting', ...
                'Unknown DICOM dictionary set.' ...
            );
        end
        dict = dcmv_dicts.(dict);
        if ~isfield(dict, vkey)
            error( ...
                'xff:BadArgument', ...
                'Unknown DICOM tag key given.' ...
            );
        end
        vkey = dict.(vkey);
        if isfield(bc.DataKeyLookup, vkey)
            dcmval = bc.Data(bc.DataKeyLookup.(vkey)).Value;
        elseif isfield(bc.MetaKeyLookup, vkey)
            dcmval = bc.Meta(bc.MetaKeyLookup.(vkey)).Value;
        elseif nargin > 2
            dcmval = varargin{1};
        else
            error( ...
                'xff:BadArgument', ...
                'Given DICOM tag key not present in file.' ...
            );
        end
    end
end
```

```

        dcmval = interpret_dcmval(dcmval);
        return;
    end
    if ~isfield(bc.DataKeyLookup, vkey)
        if nargin < 3
            error( ...
                'xff:BadArgument', ...
                'Given DICOM tag not present in file.' ...
            );
        else
            dcmval = varargin{1};
            dcmval = interpret_dcmval(dcmval);
            return;
        end
    end
    dcmval = bc.Data(bc.DataKeyLookup.(vkey)).Value;
    dcmval = interpret_dcmval(dcmval);

% numeric key format
elseif isa(vkey, 'double')
    if numel(vkey) == 2 && ...
        ~any(isinf(vkey) | isnan(vkey) | vkey < 0 | vkey > 65535)
        try
            dcmval = dcm_Value(hfile, sprintf('%04x%04x', vkey(1),
vkey(2)), varargin{:});
        catch ne_eo;
            rethrow(ne_eo);
        end
    elseif numel(vkey) == 1 && ...
        nargin > 2 && ...
        isa(varargin{1}, 'double') && ...
        ~isinf(vkey) && ...
        ~isnan(vkey) && ...
        vkey >= 0 && ...
        vkey < 65536 && ...
        numel(varargin{1}) == 1 && ...
        ~isinf(varargin{1}) && ...
        ~isnan(varargin{1}) && ...
        varargin{1} >= 0 && ...
        varargin{1} < 65536
        try
            dcmval = dcm_Value(hfile, sprintf('%04x%04x', vkey,
varargin{1}), varargin{2:end});
        catch ne_eo;
            rethrow(ne_eo);
        end
    else
        error( ...
            'xff:BadArgument', ...
            'Invalid DICOM tag (or key) given.' ...
        );
    end
end

```

```
end
dcmval = interpret_dcmval(dcmval);

else
    error( ...
        'xff:BadArgument', ...
        'Invalid DICOM tag (or key) given.' ...
    );
end

% sub function
function dcmval = interpret_dcmval(dcmval)
    global ne_methods;
    if ischar(dcmval) && ...
        ~isempty(dcmval)
        dcmval = dcmval(:)';
        if ~isempty(regexpi(dcmval, ...
            '^s*[\+\-]?\d+(\.\d+)?([eE][\+\-]?\d+)?(s*\s*s*[\+\-]
            ]?\d+(\.\d+)?([eE][\+\-]?\d+)?)*s*$'))
            dcmvalc = ne_methods.splittocell(dcmval, '\');
            dcmvaln = zeros(1, numel(dcmvalc));
            try
                for vc = 1:numel(dcmvalc)
                    dcmvaln(vc) = str2double(dcmvalc{vc});
                end
            catch ne_eo;
                neuroelf_lasterr(ne_eo);
                return;
            end
            dcmval = dcmvaln;
        end
    elseif isnumeric(dcmval)
        dcmval = double(dcmval);
    end
end
% end of function dcmval = interpret_dcmval(dcmval)
```

From:

<http://dag.dokuwiki.dpz.lokal/> - DAG wiki

Permanent link:

[http://dag.dokuwiki.dpz.lokal/doku.php?id=neuroelf:neuroelf\\_resources&rev=1454496159](http://dag.dokuwiki.dpz.lokal/doku.php?id=neuroelf:neuroelf_resources&rev=1454496159)

Last update: 2022/12/29 07:15

