GEOS F436/636 Beyond the Mouse

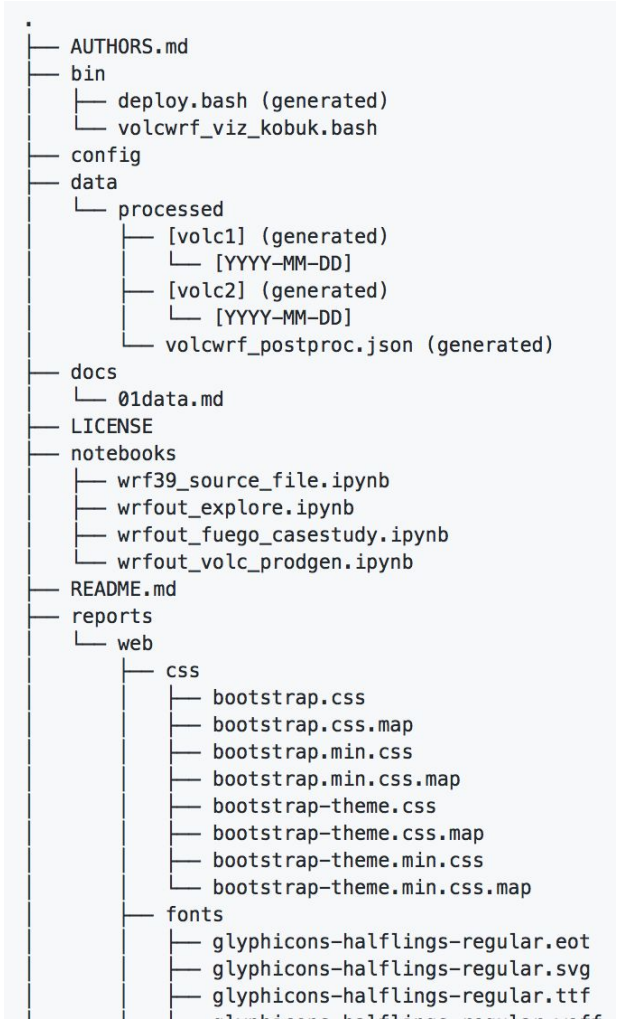
Christine (Chris) Waigl
University of Alaska Fairbanks – Fall 2018
Week 6: MATLAB I/O (=input/output)

Topics for week 6

- Getting data into and out of MATLAB
- Reading and writing text files (including delimited, csv...)
- Looping through files line-by-line
- Binary scientific data formats: HDF5, GeoTIFF

Thinking questions: What is a file? What other ways of storing and accessing data do you know? (2 min)

- Files are self-contained storage containers that hold data (including configuration information, text, source code, binary code...) that is accessible to an operating system.
- "File" is also a *metaphor*, a way of thinking about how information on a computer is organized, in hierarchical (nested) sets of files that live in folders that live in folders....
- Not every kind of data that you want to access and/or write uses the file metaphor. For example, data in a (relational) database, on a web site (OR in general over the network) often uses the "client/server" metaphor.



Files and paths

To get to a file, we can specify (on Windows):

- The absolute (full) path, for example: **C:\Users\chris\MATLAB\2018Ice\makepix.m**
- The relative path. For example, from a file in C:\Users\chris\MATLAB you could just write: **2018Ice\makepix.m**

On Unix/OSX, this could be:

/home/chris/MATLAB/2018Ice/makepix.m

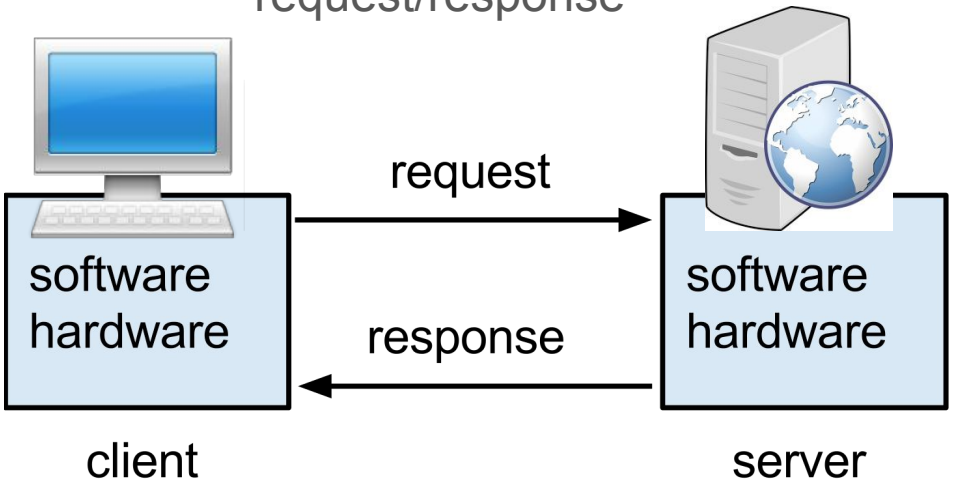
The top level is called the file system *root.*

file/folder (directory) vs. client/server

open/read/write/close

Source: HowtoGeek

request/response



Source: Wikipedia

It is important to remember that these are *metaphors* about how information is organized on a computer.

Now to MATLAB!
First, a reminder about what we've learned already

- We can access *numerical* data in delimited text files with `dlmread`. You can skip rows and columns (eg. header information, comments). dlmread returns a matrix
- We can access delimited files using `textscan`, which returns a cell array
- To access a file we must first get a file identifier using `fileID = open('[filepath]')`, and once we're done close the file using `close(fileID)`. However, some MATLAB functions do this in the background automatically and require you to only provide a file path.

fread is to read a file in one go "as a binary file"

```
>> fileid = fopen('fname.txt')
>> txt = fread(fileid  ← array of numbers
>> fclose(fileid)
>> txt(1:10)           ← a sequence of numbers!
```

Similarly, use fwrite to write binary data in one go. See the documentation for options.

fgetl is for reading text files line by line

```
fileid = fopen('fname.txt')
myline = fgetl(fid);
while ischar(myline)
  disp(myline)
  myline = fgetl(fileid);
end
fclose(fileid);
```

```
fileid = fopen('fname.txt')
myline = fgetl(fid);
while ischar(myline)
  disp(myline)
  myline = fgetl(fileid);
  if strfind(lower(myline), 'pebble')
        disp(myline)
  end
end
fclose(fileid);
```

Writing data (or line-by-line text) to a file: use fprintf

```
>> x = 0:0.1:10;      ← some data (x-values)
>> y = [x; sin(x)];   ← some more data after function application)
>> fileID = open('mydata.dat', 'w')  ← open for reading
>> fprintf(fileID, 'Some data\n\n');
>> fprintf(fileID,'%f %f\n',y);
>> fclose(fileID);
```

MATLAB has specialized reading/writing routines (=functions) that are sometime useful shorthands, sometimes the only way to deal with proprietary data formats (such as Microsoft Excel)

- For CSV (comma-separated values) files: csvread and csvwrite (don't require opening the file and generating a fileID)
- For Microsoft Excel files: xlsread and xlswrite

TIP: If your data comes in Excel files, just save the raw data to text files and work with those. Don't touch the Excel files again.

And now for something different: data files!

There are many formats in which scientific data is communicated. The most common ones are:

- text-based formats like CSV (comma-separated value), tab-delimited, other delimited (use instead of Microsoft Excel files, for portability), JSON, XML
- binary structured raster files such as HDF5, NetCDF, GeoTIFF ....
- Geospatial vector and raster and vector formats (these include database formats, proprietary formats like ESRI Shapefile or KML/KMZ, and specially designed text-based formats (see https://gisgeography.com/gis-formats/ for a list)

Accessing files over the web

Use the webread function. Here is it reading a CSV file:

```
>>webread('https://docs.google.com/spreadsheets/d/e/2PACX-1vSECmDFa87Amdu5APsOrNvzIv4aS0_db5KNhRSRIBWJgkouVpgM249h7x_WY5FsTd6RjcAmzXpIZh6i/pub?gid=87741650&single=true&output=csv')
```

Webread returns a lot of different data formats depending on what the web server handed over to it. It is your job to transform the data after retrieving it. The web server's response embeds information (metadata) about the type of data! With CSV type data, it reads the data into a MATLAB table.

Data and metadata (=data about data)

Scientific data files usually contain ***metadata***: data about data.

Examples are:

- time stamps of creation/acquisition/processing
- units of the data values
- software versions of processing
- author name and affiliation
- usage license
- location information
- plain-text description

An example of a scientific format: HDF5

Why HDF5? It's a good case study for a scientific data format. The netCDF format common for climate data, remote sensing imagerym and model output is based on it. You're likely to encounter it at one point.

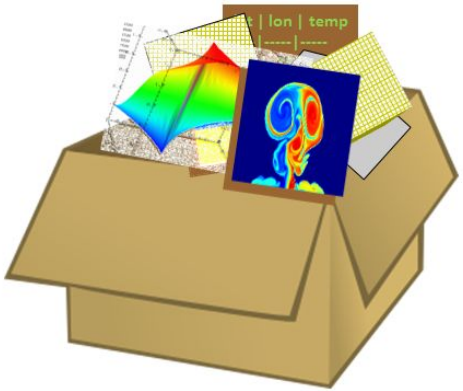
Also, the .mat MATLAB data format is based on the free HDF5 format.

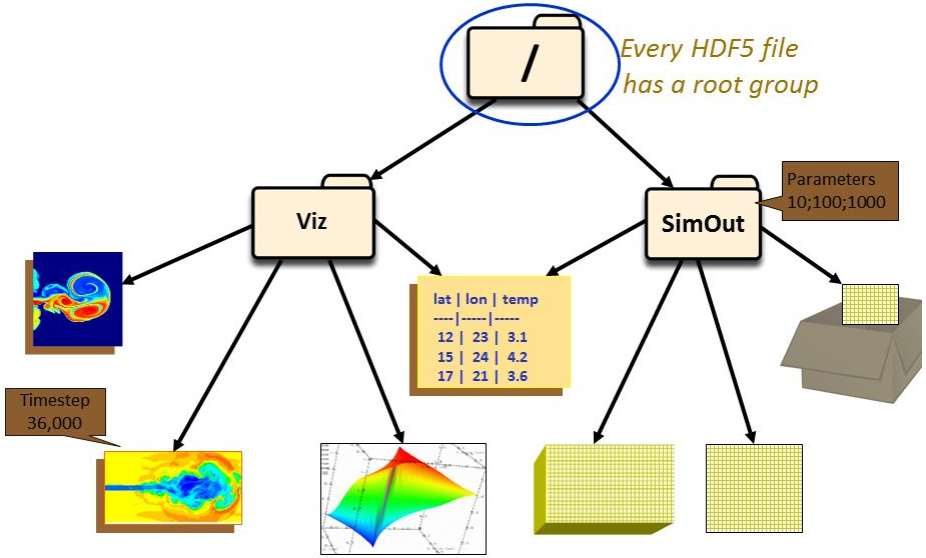
All documentation about HDF5 can be found at the HDFGroup:

https://portal.hdfgroup.org/display/HDF5/Introduction+to+HDF5

HDF5 is a hierarchical data format

HDF5 organizes raster data (=multi-dimensional arrays) into datasets, and datasets into groups. It allows attaching metadata to datasets, groups and the whole file. Examples -- in the practical part.





Metadata

Dataspace

Rank: 3

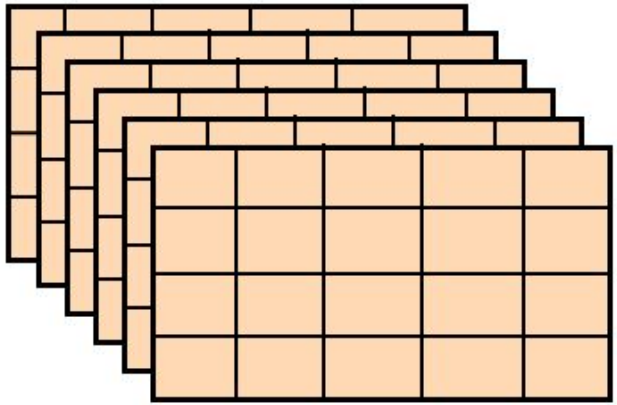
Dimensions:
Dim 1 = 4
Dim 2 = 5
Dim 3 = 6

Datatype
Integer

Attributes
Time = 32.4
Pressure = 987

Properties
Chunked
Compressed

Data



MATLAB functions to access HDF5 files

h5disp and h5info: Display/retrieve general information about a file or dataset:

```
>> h5disp('example.h5');      ← the example.h5 file comes with MATLAB
>> h5disp('example.h5','/g4/world');
>> info = h5info(filename)    ← when you want to use the information in your script
>> info = h5info(filename,location)
```

h5readatt and h5read to read data attributes (metadata) or the data itself:

```
>> data = h5read('example.h5','/g4/world')
>> latunit = h5readatt('example.h5','/g4/lat', 'units')
```

hdf5write to write data to a HDF5 file:

```
>> dataset = {'north', 'south'; 'east', 'west'};
>> hdf5write('myfile.h5', '/group1/dataset1.1', dataset);
```