



unidata

Updates to the TDS Web User Interface

Summer 2018 Unidata Student Internship

Hailey Johnson

Sean Arms, Christian Ward-Garrison

Introduction

What I did this summer:

1. HTMLwriter → Thymeleaf templating
2. UI design updates
 - a) Customizable CSS
 - b) Extensible HTML
3. Jupyter Notebook dataset “viewer”

Thymeleaf Templating

```
String logoUrl = htmlConfig.prepareUrlStringForHtml(htmlConfig.getInstallLogoUrl());
if (logoUrl != null) {
    sb.append("<img src='\"").append(logoUrl);
    String logoAlt = htmlConfig.getInstallLogoAlt();
    if (logoAlt != null) sb.append("\" alt='\"").append(logoAlt);
    sb.append("\" align='left' valign='top'").append(">\n");
}

sb.append(" Catalog ").append(catname);
sb.append("</h1>");
sb.append("<HR size='1' noshade='noshade'>");

sb.append("<table width='100%' cellpadding='5' cellspacing='0' align='center'>\r\n");

// Render the column headings
sb.append("<tr>\r\n");
sb.append("<th align='left'><font size='+1'>");
sb.append("Dataset");
sb.append("</font></th>\r\n");
sb.append("<th align='center'><font size='+1'>");
sb.append("Size");
sb.append("</font></th>\r\n");
sb.append("<th align='right'><font size='+1'>");
sb.append("Last Modified");
sb.append("</font></th>\r\n");
sb.append("</tr>");

// Recursively render the datasets
doDatasets(cat, cat.getDatasetsLocal(), sb, shade: false, level: 0, isLocalCatalog);

// Render the page footer
sb.append("</table>\r\n");
sb.append("<HR size='1' noshade='noshade'>");
appendSimpleFooter(sb);
sb.append("</body>\r\n");
sb.append("</html>\r\n");
```



Thymeleaf

Natural templates

- Server-side templating engine
- Templates can be rendered as HTML by browsers
- Integrates with Spring

- Variable/Selection expressions
- Basic logical operators, conditionals, loops, mathematical operations, etc.
- Fragment expressions



Thymeleaf

Sample

```
<html lang="en" xmlns:th="http://www.thymeleaf.org">
  <head></head>
  <body>
    <div th:fragment="access" class="tab-content access" id="access">
      <h3>Access:</h3>
      <table class="property-table">
        <tr><th>Service</th><th>Type</th><th>Description</th></tr>
        <tr th:each="access : ${dataset.getAccess()}">
          <td><a th:href="${access.get('href')}">
            <b th:text="${access.get('serviceTypeName')}"></b></a></td>
          <td th:if="${access.get('accessType') != null}" th:text="${access.get('accessType')}"></td>
          <td th:text="${access.get('serviceDesc') == null ? '' : access.get('serviceDesc')}"></td>
        </tr>
      </table>
    </div>
  </body>
</html>
```



Thymeleaf

Which pages use Thymeleaf?

1. Catalog pages
2. Dataset pages
3. NCSS (Grid & Point) pages

Why?

1. Simplicity
2. Efficiency
3. Consistency

Thymeleaf templating

Thymeleaf'd views:

```
public CatalogItemContext(Dataset ds, int level)
{
    // Get display name
    this.displayName = ds.getName();

    // Get data size
    double size = ds.getDataSize();
    if ((size > 0) && !Double.isNaN(size))
        this.dataSize = (Format.formatByteSize(size));

    // Get last modified time.
    DateType lastModDateType = ds.getLastModifiedDate();
    if (lastModDateType != null)
        this.lastModified = lastModDateType.toDateTimeString();

    // Store nesting level
    this.level = level;
}
```

```
protected void populateItemContext(Dataset ds, List<CatalogItemContext> catalogItems,
                                   boolean isLocalCatalog, int level) {
    CatalogItemContext context = new CatalogItemContext(ds, level);

    // add item href
    context.setHref(getCatalogItemHref(ds, isLocalCatalog));

    // add item icon
    context.setIconSrc(getFolderIconSrc(ds));

    // add item to Catalog
    catalogItems.add(context);

    // recursively add subdirectories
    if (!(ds instanceof CatalogRef)) {
        addCatalogItems(ds, catalogItems, isLocalCatalog, level: level + 1);
    }
}
```

UI Design Updates

Humanity's victories:
















land probe perfectly on a comet 310 million miles away, using science



get stuff on a web page to align properly
using CSS



Dataset	Size	Last Modified
 Realtime data from IDD		--
 Forecast Model Data		--
 Forecast Products and Analyses		--
 Observation Data		--
 Radar Data		--
 Satellite Data		--
 Other Unidata Data		--
 Unidata case studies		--
 Test Datasets		--
 Test Datasets		--
 Test Point Datasets		--
 CF DSG Example Datasets		--
 For GRIB Indexing Purposes		--
 Test Point Datasets		--



Welcome to THREDDS Data Server top-level TDS Catalog.

Hosted by Unidata.

Catalog

Dataset	Size	Last Modified
 Realtime data from IDD		--
 Unidata case Studies		--
 Test Datasets		--
 Test Grid Datasets		--
 Test Point Datasets		--
 Test Restricted Datasets		--
 NARR Test		--
 For GRIB Indexing Purposes		--
 GSD HRRR Datasets		--



Catalog https://thredds-test.unidata.ucar.edu/thredds/catalog/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2/catalog.html

Dataset: [GFS_Global_0p5deg_ana_20180620_0000.grib2](#)

- *Data format:* GRIB-2
- *Data size:* 56.59 Mbytes
- *Feature type:* GRID
- *Naming Authority:* edu.ucar.unidata
- *ID:* grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2

Documentation:

- **summary:** Single reference time Grib Collection
- **Reference Time:** 2018-06-20T00:00:00Z
- [NCEP Model documentation](#)
- **rights:** Freely available
- **processing_level:** Transmitted through Unidata Internet Data Distribution.
- **processing_level:** Read by CDM Grib Collection.
- [NCEP/NWS Model Analyses and Forecasts page](#)
- [Unidata IDD Model Data page](#)
- **summary:** NCEP Global Forecast System Model, previously called AVN/MRF (Medium Range Forecast)
- [COMET MetEd \(Meteorology Education and Training\) documentation](#)
- [NCEP Model Notes](#)
- **summary:** NCEP GFS Model : AWIPS 230 (G) Grid. Global Lat/Lon grid. Model runs at 0, 6, 12, and 18Z. Horizontal= 361 by 720 points, resolution 0.5 degree, Lat/Lon projection. Vertical= 1000 to 100 hPa mandatory pressure levels (10 levels); surface, height above ground, pressure layers.
- **summary:** Analysis grids only.

Access:

1. **OPENDAP:** [/thredds/dodsC/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2.html](#)
2. **CdmRemote:** [/thredds/cdmremote/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
3. **CdmrFeature:** [/thredds/cdmrfeature/grid/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
4. **DAP4:** [/thredds/dap4/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2.dmr.xml](#)
5. **HTTPServer:** [/thredds/fileServer/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
6. **NetcdfSubset:** [/thredds/ncss/grid/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2/dataset.html](#)
7. **WMS:** [/thredds/wms/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
8. **WCS:** [/thredds/wcs/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
9. **ISO:** [/thredds/iso/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
10. **NCML:** [/thredds/ncml/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)
11. **UDDC:** [/thredds/uddc/grib/NCEP/GFS/Global_0p5deg_ana/GFS_Global_0p5deg_ana_20180620_0000.grib2](#)

Dates:

- 2018-06-20T03:24:40Z (modified)

Creators:

- **DOC/NOAA/NWS/NCEP**
 - *email:* http://www.ncep.noaa.gov/mail_liaison.shtml
 - <http://www.ncep.noaa.gov/>



Dataset: DEM_Mtnzs_20180227.nc

Catalog: <http://localhost:8081/thredds/catalog/rtkMtnzs/catalog.html>

<i>dataFormat</i>	NetCDF
<i>featureType</i>	Grid
<i>dataSize</i>	12756088
<i>id</i>	rtkMtnzs/DEM_Mtnzs_20180227.nc

Access Preview**Access:**

Service	Type	Description
OpenDAP	Data Access	Access dataset through OPeNDAP using the DAP2 protocol.
DAP4	Data Access	Access dataset through OPeNDAP using the DAP4 protocol.
HTTP Server	Data Access	HTTP file download.
NCML	Metadata	Provide NCML representation of a dataset.

Documentation Dates Creators Publishers **Variables****Variables:**

- *Vocabulary* []:
 - **PX (eastings)** = Longitude coordinates of area polygon.
 - **PY (northings)** = Latitude coordinates of area polygon.
 - **X (eastings)** = Longitude of collected data points.
 - **Xg (eastings)** = Longitude of gridded data.
 - **Y (northings)** = Latitude of collected data points.
 - **Yg (northings)** = Latitude of gridded data.
 - **Z (meters)** = Elevations of collected data points.
 - **Zg (meters)** = Elevations of gridded data.

UI Updates

Customizable CSS

Extensible HTML

- Existing feature: user contributed stylesheet
- Use templates for consistency:
 - Page templates look for user contributed stylesheets which correspond to that page
 - “standard.css” – loaded on all pages
 - “catalog.css” and “dataset.css” – loaded on catalog/dataset views
 - Page layout is reliably structured:
 - Elements have classes and ids

UI Updates

Customizable CSS

Extensible HTML

- Thymeleaf fragments: portable templates
 - Page templates structured as a rigid skeleton which loads fragments
 - Plugin points: e.g. header, footer, content-section
 - User adds fragments to
`content/thredds/templates/tdsTemplateFragments.html`
 - Overridable fragments: custom or default
 - Optional fragments: custom or none

UI Updates

Customizable CSS

Extensible HTML

Template resolvers:

- Default: SpringResourceTemplateResolver
- Custom: TdsExtensibleTemplateResolver
 - Resolvable patterns: "ext:*"
 - Looks for matches in the content directory

a.

```
<div th:fragment="header-container">
  <div id="header-buffer"></div>
  <div id="header" th:insert="~{ext:tdsTemplateFragments :: header} ?: ~{templates/commonFragments :: header-content}">
  </div>
</div>
```

b.

```
<div class="content" th:if="~{ext:tdsTemplateFragments :: catalogCustomContentBottom}">
  <div th:replace="~{ext:tdsTemplateFragments :: catalogCustomContentBottom}"></div>
</div>
```

UI Updates

Customizable CSS

Extensible HTML

Example: Contributing multiple fragments

templates/tdsTemplateFragments.html

```
<div th:fragment="datasetCustomContentBottom">
  <div th:replace="~{ext:additionalFragments/myFragments :: mySectionHeader}"/>
  <div th:replace="~{ext:additionalFragments/myFragments :: mySectionContent}"/>
</div>
```

templates/additionalFragments/myFragments.html

```
<div th:fragment="mySectionHeader" class="section-header">My Section Name</div>
<div th:fragment="mySectionContent" class="section-content">Your contributed content here.</div>
```


Jupyter Notebook Service

Siphon THREDDS Jupyter Notebook Viewer

Dataset: GOES16_CONUS_20180719_000228_0.47_1km_30.1N_87.1W.nc4



Dependencies:

- Siphon: `pip install siphon`
 - matplotlib: `pip install matplotlib` or `conda install -c conda-forge matplotlib`
 - ipywidgets:
 - `pip install ipywidgets` or `conda install -c conda-forge ipywidgets`
- then
- Using Jupyter Notebook: `jupyter nbextension enable --py widgetsnbextension`
 - Using JupyterLab:
 - Requires nodejs: `conda install nodejs`
 - `jupyter labextension install @jupyter-widgets/jupyterlab-manager`
 - numpy: `pip install numpy` or `conda install numpy`

```
In [ ]: from siphon.catalog import TDSCatalog
import matplotlib.pyplot as plt
import numpy as np
import ipywidgets as widgets
```

```
In [ ]: catUrl = "https://thredds-dev.unidata.ucar.edu/thredds/catalog/satellite/goes16/GOES16/CONUS/Channel01/20180719/catalog.xml";
datasetName = "GOES16_CONUS_20180719_000228_0.47_1km_30.1N_87.1W.nc4";
```

Access a dataset

With the TDS catalog url, we can use Siphon to get the dataset named `datasetName`.

```
In [ ]: catalog = TDSCatalog(catUrl)
```

Jupyter Notebook service

Purpose:

- Return ipynb file pre-populated with catalog URL & Dataset name
- Demo access to datasets via Siphon
- Simple data visualization

```
In [2]: catUrl = "http://localhost:8081/thredds/catalog/rtkMatanzas/catalog.xml";  
datasetName = "DEM_Mtnzs_20171019.nc";
```

Access a dataset

With the TDS catalog url, we can use Siphon to get the dataset named `datasetName`.

```
In [4]: catalog = TDSCatalog(catUrl)
```

```
In [5]: ds = catalog.datasets[datasetName]  
ds.name
```

```
Out[5]: 'DEM_Mtnzs_20171019.nc'
```

Jupyter Notebook service

How it works:

1. On TDS startup:
 - a) TDS creates a cache to store Notebooks and their mappings to datasets
 - b) Parses all ipynb files in the “Notebooks” directory as NotebookMetadata objects; saved permanently in the cache
 - c) Registers “JupyterNotebookViewer” as a data viewer
2. On Dataset page load
 - a) TDS accesses or creates a mapping to the appropriate Notebook
3. On Notebook service request
 - a) NotebookController reads the mapped ipynb file and inserts the Catalog URL and Dataset Name where appropriate
 - b) Returns the edited Notebook file

Mapping Notebooks to Datasets

Edit Notebook Metadata

Manually edit the JSON below to manipulate the metadata for this notebook. We recommend putting custom metadata attributes in an appropriately named substructure, so they don't conflict with those of others.

```
1 {
2   "kernel_spec": {
3     "name": "python3",
4     "display_name": "Python 3",
5     "language": "python"
6   },
7   "language_info": {
8     "name": "python",
9     "version": "3.6.5",
10    "mimetype": "text/x-python",
11    "codemirror_mode": {
12      "name": "ipython",
13      "version": 3
14    },
15    "pygments_lexer": "ipython3",
16    "nbconvert_exporter": "python",
17    "file_extension": ".py"
18  },
19  "viewer_info": {
20    "accept_catalogs": [
21      "http://localhost:8081/thredds/catalog/rtkMatanzas/catalog.html"
22    ],
23    "order": 2
24  }
25 }
```

Cancel

Edit

Jupyter Notebook service

Mapping notebooks to datasets:

```
private class NotebookMetadata {  
    public String filename;  
    public boolean accept_all;  
    public List<String> accept_datasetIDs;  
    public List<String> accept_catalogs;  
    public List<String> accept_dataset_types;  
    public int order;  
  
    public boolean isValidForDataset(Dataset ds) {  
  
    public int compareNotebookForDataset(Dataset ds, NotebookMetadata md) {  
  
@Component  
public class JupyterNotebookServiceCache {  
    static private final Logger logger = LoggerFactory.getLogger(JupyterNotebookServiceCache.class);  
  
    @Autowired  
    TdsContext tdsContext;  
  
    private List<NotebookMetadata> allNotebooks;  
  
    private Cache<String, NotebookMetadata> notebookMappingCache;
```

Jupyter Notebook service

Contributing your own Notebook:

1. Create your notebook (ipynb)
2. Update the “viewer_info” metadata
3. Place notebook in `content/thredds/notebooks`



TDS “in the wild”

Demo #1

Intro

Progress/Results

Demo

Applications

Going Forward

Demo #2



TDS not “in the wild”

Intro

Progress/Results

Demo

Applications

Going Forward

Ok, that's kinda cool, but so what?

Intro

Progress/Results

Demo

Applications

Going Forward



Goals/Benefits

- Administrative users:
 - Curate data more effectively
 - Target users and use cases
- End users:
 - Lower barrier-to-entry for data use
 - Web navigation
 - Access and usage examples/instructions
 - Wider range of users



Use cases

- Large repositories:
 - Data discoverability, visualization, access
- Education:
 - Interactive instructional notebooks
- Publication:
 - Small data repositories as supplementary material
 - Verify methods and results

Future work

Unexciting

Slightly more exciting

Exciting

- **Baseline work:**
 - Document contributing Notebooks
 - Add tests
 - Notebook service
 - UI tests

Future work

Unexciting

Slightly more exciting

Exciting

- Notes from building a demo:
 - Free-form metadata type for datasets and catalogs
 - Top content-section for dataset page
 - Pattern-matching for notebook registration



Future work

Unexciting

Slightly more exciting

Exciting

- More features:
 - Multiple Notebook viewers per dataset (end user's choice)
 - Distribute common templates for features



Summer 2018: a visual summary



Thank you, everyone!