Subject: JEDI User's Guide
Author: Zhang Fan Xing
Date: September 8, 2010

1. Background

JEDI stands for Java EDR (Experimental Data Record) Display Interface. It is an essential EDR quick-view tool that has made important contributions to many past and current NASA/JPL space missions, such as Cassini, MER and Phoenix. However, being a monolithic java-based web application with tightly-coupled client and server components, it showed a sign of technical aging with increasing difficulty for mission adaptation. To better support future space missions, a task calling for a new design and re-implementation was defined and further funded by the MGSS office.

2. Architecture

Listed below are the main features of JEDI:

1. There is a complete separation of presentation from data and logic, which results in independent client and server sides, communicating only through clearly-defined APIs. As such, either side can evolve in its own path, thus is better positioned for easier and much more flexible updates with future web.

2. Some data models in JEDI contain rich science domain knowledge. They have been preserved and improved. Web functionalities are created from scratch leveraging on contemporary web technology.

3. The server side provides individual services covering login/logout, create/browse/display/delete show, etc., all through well-defined APIs. These services can be consumed by the JEDI default client, and any new future in-browser or desktop client.

4. The default JEDI client is an AJAX web client application that mimics JEDI client. It works in all major web browsers out of the box without any library dependency.

3. Client

There is a concept called "show" that a user must be familiar with. It was never clearly defined in JEDI, but is core to the implementation of JEDI and, consequently, the use of default JEDI client.

To a user, the default client provides customized EDR viewing in a slide show fashion for authenticated users.

3.1. Authentication and Authorization

Only an authenticated and authorized user is permitted. A user must provide

valid username and password to the login form, which is typically available at a URL like http://hostname:port/jujube/login. The default port is 8080. Authentication is done via Kerberos verification. Authorization is done via LDAP verifcation.

Project	Environment	URL
Cassini	Development	http://tpsweb-dev/jujube/login
Cassini	Test	http://tpsweb-test/jujube/login
Cassini	Operations	https://tpsweb/jujube/login

Please contact your project representative to set up authentication and authorization credentials.

3.2. Create Show

After logging in, the user is presented with a form to create a show. A show is created using either of two types of data sources:

- a file directory accessible to the server machine that the JEDI application is hosted.
- a in-memory stream data source that can be populated and updated by allowed tlmproc process via URL push.

A user can create an arbitrary number of shows, private to the user. Created shows are listed in a table at the top of the page. Each row corresponds to a show that can be browsed, played and deleted. A show is keyed (and, currently, thus-named) by a combination of creation time and user name.

In current implementation, shows are persist only in server instance memory and will disappear after a server reboot. It is planned to make them survivable over server reboots.

3.3. Browse Show

To browse a show, click on the "browse" link. A "folder-like" page view of the show will be presented.

On this page, each member entry corresponds to a EDR file, which, in turn, can be clicked on and "opened" like a "folder". This process is traversing till to the end, when there is no more links. On each "folder-like" page, there are also links to meta-info in JSON format, which are helpful to developers who are creating new clients.

Technically, the "folder-like" page view of a show is the simplest html view of a webified "data store", which, in this case, is a show.

3.4. Play Show

To play a show, click on the "play" link. A new browser window will be spawned.

In this new window, EDR files contained in the show are displayed one by one in default frequency set up when the show was created. To change the display speed, or tune the show in general, simply click anywhere in the window to hide the "slide show". Now in the same window, the form for modifying show behavior will appear, along with show parameters. Try to change some, just hit return, the change will take effect right way.

3.5. Delete Show

To delete a show, just click on the "delete" link.

3.6. Logout

A user logs out by clicking on the "logout" link. In current implementation, a logged-in user will be auto-logged out after a period of inactivity. The default idle time is 10 min (600 sec). When logged out, whether auto or not, created shows will still be alive.

4. Server

The JEDI server provides individual RESTful services that a client interacts with to create a complete application for end user.

4.1 Services

Listed below are service endpoints that are supported in current implementation:

- `http://hostname:port/jujube/login`
- `http://hostname:port/jujube/list`
- `http://hostname:port/jujube/create`
- `http://hostname:port/jujube/modify`
- `http://hostname:port/jujube/delete`
- `http://hostname:port/jujube/logout`

Most of them are access-controlled by auth credentials.

5. Personality

The personality files are simply ASCII text files that describe the Look-and-Feel that the client and server will adhere to in presenting the EDR data to the user.

5.1 Cassini Example

Let's take a look at the default Cassini Personality as an example. Note: The lines that begin with a # sign denote comments.

```
# Each personality file must contain a PersonalityName.
# This is usually the name of the instrument.
# It will be used to match the role that the user has been assigned to.
PersonalityName=CASISS

# DisplayGeometry lets the user specifies the x, y location of the
# JEDI window (applicable for stand-alone version only) and the
# width and height of the window.
DisplayGeometry[x=10, y=100, width=620, height=600]

# InstrumentDNScale specifies how to scale the DN's into a 0-256 range.
# The short specification tells JEDI how many bits are used for the DN range
# from the instrument. This helps in scaling as many cameras of late use
# 12-bit ranges. Since such values are stored as 16-bit shorts, the short=12
# specification helps JEDI knows how many bits are actually used to convey
the
# DN from the instrument readings. The byte specification is fo rthe
# same use. Typically, this will always be set to 8. It is included here
# simply for completeness. Also, if ever there were an instrument that used
# less than a byt'es worth of DN range, this might be helpfu.
# You may enter a -1 for auto scaling, i.e., let the software figure out
# the range and scaling.
InstrumentDNScale[byte=8, short=12]

# IgnoreValues specify which values will be ignored when calculating
# the mean, max, min values etc. This enhanced the quality of the
# picture by ignoring the saturated points, for example. You may
# have up to 10 ignored values. Index starts from 1.
IgnoreValues[1=4095, 2=255]

# IgnoreRange is for specifying a bigger range of values to be
# ignored. Any value <= the lower value or >= the upper value will
# be ignored.
IgnoreRange[lower=0, upper=4095]

# EDRCanvas specifies the information related to the EDR:
#   name is the name of the EDR canvas
#   x, y, width, & height specifies the geometry of the canvas.
#   stretchType can be percent or linear, and is the stretch method used.
#   param0 and param1 are the lower and upper limits fo the stretch.
# The following 5 parameters are optional, but must be specified as a group.
i.e.,
# if you just specify 1 to 4 parameters, the special check is not performed.
#   minBelowBg specifies the absolute minimum.
#   minLowerLimit
#   minDifference
#   minAbsolute
#   maxAboslute
# The following 4 parameters are optional, but must be specified as a group.
i.e.,
# if you just specify 1 to 3 parameters, the special check is not performed.
#   missingLineDN specifies the value that indicates a missing line.
#   missingLineR, missingLineG, & missingLineB are the RGB values for the
pixel w/
#    with a matching missingLineDN.
EDRCanvas[name=iss, x=30, y=10, width=256, height=256, stretchTyp = percent,
\
param0=0.001, param1=0.001, minBelowBg=15.0, minLowerLimit=30.0,
minDifference=50.0, \
minAbsolute=0.0, maxAbsolute=4095.0, missingLineDN=0, missingLineR=255,
missingLineG=255, \
missingLineB=0]

# GreyWedge is a grey bar that shows the graduation??
GreyWedge[name=iss, x=11, y=10, width=16, height=256, ascending=false,
vertical=true, steps=256]

# HistogramCanvas specifies the information related to the histogram:
#   name is the name of the histogram
#   x, y, width & height specifies the geometry of the canvas.
#   axis can be log or linear. It specifies whethe the histogram is shown as
log or linear.
```

```
HistogramCanvas[name=iss, x=330, y=480, width=256, height=100, axis=log]

# IconCanvas specifies the information related to the icon:
#   x & y specifies the location of the icon
#   URL specifies the location of the image used for the icon
IconCanvas[x=520, y=10,
URL=https://tpsweb.jpl.nasa.gov/Jedi/images/casslogo_4.gif]

# KeywordCanvas specifies the information related to a keyword:
#   x & y specifies the location of the keyword
#   keyword is the tag used to match the keyword in the EDR label for
retrieving the value
#   value is a place holder to indicate that there's a value associated w/
this keyword
#   show is the text for the keyword that is displayed
#   delimiter specifies the symbol used to spearate the keyword and its
value.
#   color specifies the color of the text
#   alarmValue & alarmColor specifies the special alarmed condition
#   alarmOp is the alarm operation that will be used to determine if the
current value
#     is in alarm condition. 0 means no comparison, 1 is for =, 2 is for !=,
3 is for >,
#     4 is <.
#   evaluatedAsString is set to false if the comparison is not on numbers;
#     otherwise it's true.
#   fontsize specifies the font size
KeywordCanvas[x=330,y=20,keyword=ANTIBLOOMING_STATE_FLAG,value=---
,show=Antiblooming,\

delimiter=:,color=FFFF00,alarmValue=,alarmColor=FF0000,alarmOp=0,evaluatedAs
String=true,\
   fontSize=14]
KeywordCanvas[x=330,y=40,keyword=CALIBRATION_LAMP_STATE_FLAG,value=---
,show=Cal. Lamp,\

delimiter=:,color=FFFF00,alarmValue=,alarmColor=FF0000,alarmOp=0,evaluatedAs
String=true,\
   fontSize=14]
```

5.2 Extension

New syntax has been introduced for personality file in JEDI.

Note that

- new property name is in camel case and case-sensitive.
- new property takes in standard css value if applicable.

For possible values, please check any good online css reference such as http://www.w3schools.com/css/css_reference.asp

List below are the extensions:

- The very first line should be like # version 1.1.0

Key	Additional Values
DisplayGeometry	backgroundColor, backgroundImage, backgroundRepeat
EDRCanvas	maskBorderThickness, maskBorderColor
IconCanvas	Property URL
KeywardCanvas	a keyword group leader must have property group
KeywordCanvas	a keyword group leader can have properties color, fontSize, fontWeight, fontFamily and align. Please

	note that align is not a css property and its value can be one of left, right, delimiter. left is the default