

SDSF data through Java: Overview

- **Problem Statement / Need Addressed:**
 - Access SDSF through a Java application
- **Solution:**
 - Use SDSF/Java classes in your Java application
- **Benefit:**
 - Use Java for access to SDSF
 - Access panels and panel data
 - Retrieve syslog data and issue system commands
 - Retrieve job output
 - Take action to perform functions similar to action characters and overtyping
 - Filter data to reduce output returned
 - View results of interactions
 - Control access through standard SDSF security mechanisms
 - Display and modify system data

Setup

- Update CLASSPATH environment variable to reference SDSF jar file:
 - `export CLASSPATH=/usr/include/java_classes/isfjcall.jar:$CLASSPATH`
- Update LIBPATH to reference SDSF DLL:
 - `export LIBPATH=/usr/lib/java_runtime:$LIBPATH` (31-bit)
 - `export LIBPATH=/usr/lib/java_runtime64:$LIBPATH` (64-bit)
- SDSF requires Java SDK V6
 - Either 31-bit or 64-bit mode

Writing a Java Application

- Create a runner that corresponds to the panel you want to work with
 - A runner is a Java class that provides access to SDSF
 - Contains a results object describing completion of request
- Create request settings and associate it with runner
- Invoke SDSF to create a list of objects
- Process the returned objects and obtain column values for each row
- Invoke methods on a row object to retrieve information or modify the object

Example Java Application

```
// Create optional settings object
ISFRequestSettings settings = new ISFRequestSettings();
settings.addISFPrefix("***"); // Set job name prefix
settings.addISFOwner("ibmuser"); // Set job owner

// Get a runner used to access SDSF ST panel
ISFStatusRunner runner = new ISFStatusRunner(settings);

List<ISFStatus> statObjList = null;

try {
    statObjList = runner.exec();
} catch (ISFException e) {
    // Process exception here
} finally {
    // Print SDSF messages related to request
    results.printMessageList(System.err);
}

// List job properties
if (statObjList != null) {
    for (ISFStatus statObj : statObjList) {
        System.out.println(statObjList.toVerboseString());
    }
}
```

Runners and Settings

- A runner provides access to SDSF similar to SDSF commands
 - Choose the runner corresponding to the panel you want to access
 - ISFStatusRunner – ST (status panel)
 - ISFOutputRunner – O (output panel)
 - ISFHealthCheckRunner – CK (health checks)
 - etc.
 - ISFRunner – slash command, WHO, QUERY
 - Complete cross reference of runners to panels contained in the Javadoc

Example Java Application

```
// Create optional settings object
ISFRequestSettings settings = new ISFRequestSettings();
settings.addISFPrefix("***"); // Set job name prefix
settings.addISFOwner("ibmuser"); // Set job owner
```

```
// Get a runner used to access SDSF ST panel
ISFStatusRunner runner = new ISFStatusRunner(settings);
```

```
List<ISFStatus> statObjList = null;
```

```
try {
    statObjList = runner.exec();
} catch (ISFException e) {
    // Process exception here
} finally {
    // Print SDSF messages related to request
    results.printMessageList(System.err);
}
```

```
// List job properties
if (statObjList != null) {
    for (ISFStatus statObj : statObjList) {
        System.out.println(statObjList.toVerboseString());
    }
}
```

Runners and Settings ...

- Settings are used to qualify the request
 - Job name prefix, owner, destination
 - Most settings correspond to SDSF commands
 - Limit the column values retrieved
- Represented by ISFRequestSettings class
 - Create an instance of settings and associate it with runner
 - Various addISFxxxx methods to add a setting to the object

Runners and Settings ...

```
// Create optional settings object  
ISFRequestSettings settings = new ISFRequestSettings();
```

```
settings.addISFPrefix("***");
```

Corresponds to PREFIX **
command

```
settings.addISFOwner("ibmuser");
```

Corresponds to OWNER IBMUSER
command

```
settings.addISFCols("jname jobid");
```

Requests just the JOBNAME and
JobID columns

```
// Get a runner used to access SDSF ST panel using settings  
ISFStatusRunner runner = new ISFStatusRunner(settings);
```

Example Java Application

```
// Create optional settings object
ISFRequestSettings settings = new ISFRequestSettings();
settings.addISFPrefix("***"); // Set job name prefix
settings.addISFOwner("ibmuser"); // Set job owner
```

```
// Get a runner used to access SDSF ST panel
ISFStatusRunner runner = new ISFStatusRunner(settings);
```

```
List<ISFStatus> statObjList = null;
```

```
try {
    statObjList = runner.exec();
} catch (ISFException e) {
    // Process exception here
} finally {
    // Print SDSF messages related to request
    results.printMessageList(System.err);
}
```

```
// List job properties
if (statObjList != null) {
    for (ISFStatus statObj : statObjList) {
        System.out.println(statObjList.toVerboseString());
    }
}
```

Retrieving Objects

- Panel objects are returned as lists
 - Each row represented by an object
 - Each object implements an interface for that panel
- Object interfaces
 - ISFStatus for ST row objects
 - ISFOutput for O row objects
 - ISFHealthCheck for CK objects

Retrieving Objects ...

- Use the `exec()` method to retrieve the list of objects
 - `List<ISFStatus> statObjList = runner.exec();`
- The objects returned are limited by the settings in effect
 - If `prefix=userid*`, only jobs starting with that `userid` would be returned
 - Very similar to SDSF interactive and SDSF/REXX

Example Java Application

```
// Create optional settings object
ISFRequestSettings settings = new ISFRequestSettings();
settings.addISFPrefix("***"); // Set job name prefix
settings.addISFOwner("ibmuser"); // Set job owner
```

```
// Get a runner used to access SDSF ST panel
ISFStatusRunner runner = new ISFStatusRunner(settings);
```

```
List<ISFStatus> statObjList = null;

try {
    statObjList = runner.exec();
} catch (ISFException e) {
    // Process exception here
} finally {
    // Print SDSF messages related to request
    results.printMessageList(System.err);
}
```

```
// List job properties
if (statObjList != null) {
    for (ISFStatus statObj : statObjList) {
        System.out.println(statObjList.toVerboseString());
    }
}
```

Request Results

- The runner references an ISFRequestResults object that is updated after each request
 - Contains messages describing completion of request
 - Return and reason codes
 - List of columns returned
 - Convenience methods to print messages
- Always check the results after each request
 - `ISFRequestResults results = runner.getRequestResults();`
 - `results.printMessageList();`

Working with Objects

- SDSF creates one object per row
 - Column values are contained within the object
 - Use `getValue()` method to retrieve a column value
 - Use the SDSF column name (FLD name), not the column title
 - `String jobname=statObj.getValue("jname")`
 - `String owner=statObj.getValue("ownerid")`
 - Use `getFixedField` method
 - `String fixedField=statObj.getFixedField();`

Working with Objects ...

```
statObjList = runner.exec();

for (ISFStatus statObj : statObjList) {

    // Get job name
    String jobname = statObj.getJName();

    // Print short form of row properties
    System.out.println(statObj);

    // Print all properties for row
    System.out.println(statObj.toVerboseString());

}
```

Actions

- You can modify an object similar to an action character
- Actions are represented by methods
 - Available actions defined in the interface for the object
 - Cancel a job:
 - `statObj.cancel();`
 - `results.printResponseList(System.out);`
 - The response list contains any system messages generated in response to the command
 - Similar to SDSF ULOG format

Overtypes

- You can modify an object similar to an overtyping
 - Use the requestPropertyChange method
 - Method takes two input arrays:
 - Column name array
 - Column value array
 - Each column in the name array is changed to the corresponding value in the value array

Overtypes ...

```
// Change job class to class A

// Build column name array
String propName = { "jclass" };

// Build column value array
String propValue = { "a" };

// Change the job class
statObj.requestPropertyChange(propName, propValue);

// Print response list
results.printResponseList(System.out);
```

MVS Commands

- Can issue one or more MVS commands
- Use ISFRRunner with system method
 - Takes an array of string commands
- Get responses using `getRequestResults()`