LIFERAY7

Liferay DXP Custom Entity Kaleo Workflow Tutorial

By Javasavvy

MAR 10, 2017 Implement Kaleo Workflow for Custom Assets/Custom Portlet, Kaleo Workflow forCustom Portlet, Liferay 7 workflow example, Liferay DXP Kaleo workflow Tutorial, Liferay kaleo workflow example, Liferay Workflow Custom Entity integration, Liferay Workflow Framework

Liferay DXP Custom Entity Kaleo Workflow Tutorial

In this tutorial, We will see custom modal entity integration with Liferay Kaleo Workflow in Liferay DXP. This tutorial is continuation to Liferay DXP CRUD tutorial on Leave Application and workflow will be enabled for Leave Model Entity.

Liferay 7 CRUD Portlet Tutorial

This tutorial is an extension to Liferay 7 Portlet Tutorial. In this tutorial, we will implement basic leave system application with service builder and liferay components. Installations: Liferay Eclipse Neon IDE 3.1.0 M1 :…READ MORE

Installations:

- Liferay DXP
- Liferay Portal 7

High Level Steps:

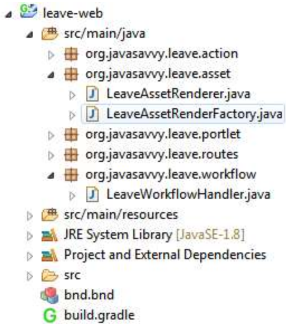
- Make the Leave entity as Asset Type
- Add workflow related columns in service.xml for Leave entity
- Register workflow Handler with Leave entity
- Register AssetRenderer with Leave entity to make Leave entity visible in Workflow Configuration Screen
- Update LeaveLocalServiceImpl with below:
  - Add updateStatus() method, which will be called by Workflow handler at runtime
  - invoke assetLocalService.updateAsset() method to add entry to asset_entry table
  - Invoke WorkflowHandlerRegistryUtil.startWorkflowInstance() method to start workflow
- Map the Leave entity with Workflow definition in the control panel

Liferay Workflow Integration with Custom Entity:

- Before going through this tutorial, Leave Entity must be enabled with AssetRenderer Configuration, so access below tutorial on this:

  Liferay 7 Asset Render Factory Example

  In this tutorial, we will make the custom entity as asset so that it can be accessible Liferay frameworks such as comments,asset publishers and workflow. Custom Entity need to be make as an Asset…READ MORE
- Register Workflow Handler with Leave Entity. In Liferay 6, liferay-portlet.xml file holds workflow-handler tag to register, but in Liferay DXP, you just need to register it as OSGI Service
  - Create package “org.javasavvy.leave.workflow” and create LeaveWorkflowHandler.java class like shown below:



```
leave-web
  src/main/java
    org.javasavvy.leave.action
    org.javasavvy.leave.asset
      LeaveAssetRenderer.java
      LeaveAssetRenderFactory.java
    org.javasavvy.leave.portlet
    org.javasavvy.leave.routes
    org.javasavvy.leave.workflow
      LeaveWorkflowHandler.java
  src/main/resources
  JRE System Library [JavaSE-1.8]
  Project and External Dependencies
  src
  bnd.bnd
  build.gradle
```

- Add @Component annotation and add “modal.class.name” property with Leave entity like shown below:

```
@Component(
 property = {"model.class.name=org.javasavvy.leave.model.Leave"},
 service = WorkflowHandler.class
)
```

- LeaveWorkflowHandler class need to extends BaseWorkflowHandler<Leave>
- LeaveWorkflowHandler class is:

```
package org.javasavvy.leave.workflow;


import java.io.Serializable;
import java.util.Locale;
import java.util.Map;
import org.javasavvy.leave.model.Leave;
import org.javasavvy.leave.service.LeaveLocalService;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;
import com.liferay.portal.kernel.exception.PortalException;
import com.liferay.portal.kernel.service.ServiceContext;
import com.liferay.portal.kernel.util.GetterUtil;
import com.liferay.portal.kernel.workflow.BaseWorkflowHandler;
import com.liferay.portal.kernel.workflow.WorkflowConstants;
import com.liferay.portal.kernel.workflow.WorkflowHandler;


@Component(
 property = {"model.class.name=org.javasavvy.leave.model.Leave"},
 service = WorkflowHandler.class
)
public class LeaveWorkflowHandler extends BaseWorkflowHandler<Leave>{


 private LeaveLocalService leaveService;
   @Reference(unbind = "-")
   protected void setLeaveService(LeaveLocalService leaveService) {
         this.leaveService = leaveService;
    }


  @Override
  public String getClassName() {
     return Leave.class.getName();
  }
  @Override
  public String getType(Locale locale) {
     return "leave";
  }


 @Override
 public Leave updateStatus(int status, Map<String, Serializable> workflowContext) throws PortalException {


    long userId = GetterUtil.getLong((String)workflowContext.get(WorkflowConstants.CONTEXT_USER_ID));
    long leaveId = GetterUtil.getLong((String)workflowContext.get(WorkflowConstants.CONTEXT_ENTRY_CLASS_PK));
    ServiceContext serviceContext = (ServiceContext)workflowContext.get("serviceContext");
    Leave leave = leaveService.updateStatus(userId, leaveId, status, serviceContext);
    return leave;
 }


}
```

- edit service.xml file and below workflow related columns.

```
<column name="status" type="int" />
 <column name="statusByUserId" type="long" />
 <column name="statusByUserName" type="String" />
 <column name="statusDate" type="Date" />
```

- add Finder method based on status:

  -
    ```
     <finder name="status" return-type="Collection">
     <finder-column name="groupId"></finder-column>
     <finder-column name="status"></finder-column>
     </finder>
    ```
- add AssetEntry reference to access assetLocalService

  -
    ```
    <reference entity="AssetEntry" package-path="com.liferay.portlet.asset" />
     <reference entity="AssetTag" package-path="com.liferay.portlet.asset" /
    ```

  The Complete Service.xml will be:

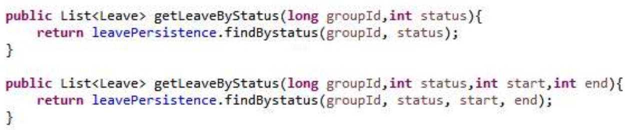
```
<?xml version="1.0"?>
<!DOCTYPE service-builder PUBLIC "-//Liferay//DTD Service Builder 7.0.0//EN"
"http://www.liferay.com/dtd/liferay-service-builder_7_0_0.dtd">

<service-builder package-path="org.javasavvy.leave">
 <namespace>js</namespace>
 <entity local-service="true" name="Leave" remote-service="true" uuid="true">
 <!-- PK fields -->
 <column name="leaveId" primary="true" type="long" />
 <!-- Group instance -->
 <column name="groupId" type="long" />
 <column name="companyId" type="long" />
 <column name="userId" type="long" />
 <column name="userName" type="String" />
 <column name="createDate" type="Date" />
 <column name="modifiedDate" type="Date" />
 <column name="leaveName" type="String" />
 <column name="startDate" type="Date" />
 <column name="endDate" type="Date" />

 <!-- Workflow status Columns -->

 <column name="status" type="int" />
 <column name="statusByUserId" type="long" />
 <column name="statusByUserName" type="String" />
 <column name="statusDate" type="Date" />
 <order by="asc">
      <order-column name="createDate" />
 </order>
 <finder name="userId" return-type="Collection">
      <finder-column name="userId" />
 </finder>
 <finder name="status" return-type="Collection">
      <finder-column name="groupId"></finder-column>
      <finder-column name="status"></finder-column>
 </finder>
 <!-- References -->
 <reference entity="AssetEntry" package-path="com.liferay.portlet.asset" />
 <reference entity="AssetTag" package-path="com.liferay.portlet.asset" />
 </entity>
</service-builder>
```

- Execute the buildService command
- Edit LeaveLocalServiceImpl with below changes:
  - add the below methods to pull the leaves by groupId and status:



```
public List<Leave> getLeaveByStatus(long groupId,int status){
    return leavePersistence.findBystatus(groupId, status);
}

public List<Leave> getLeaveByStatus(long groupId,int status,int start,int end){
    return leavePersistence.findBystatus(groupId, status, start, end);
}
```

  - update addLeave() method with:
    - assetEntryLocalService.updatEentry()

- WorkflowHandlerRegistryUtil.startWorkflowInstance()

-
  ```
  public Leave addLeave(ServiceContext serviceContext, String leaveName, Date startDate,Date leaveEndDate ){

  long leaveId = counterLocalService.increment(Leave.class.getName());
  Leave leave = null;
  try {
         User user = userLocalService.getUser(serviceContext.getUserId());
         leave = leaveLocalService.createLeave(leaveId);
         leave.setUserId(serviceContext.getUserId());
         leave.setCreateDate(new Date());
         leave.setLeaveName(leaveName);
         leave.setStartDate(startDate);
         leave.setEndDate(leaveEndDate);
         leave.setUserName(user.getFullName());
         leave.setCompanyId(serviceContext.getCompanyId());
         leave.setGroupId(serviceContext.getScopeGroupId());

         leave.setStatus(WorkflowConstants.STATUS_DRAFT);
         leave.setStatusByUserId(user.getUserId());
         leave.setStatusDate(new Date());
         leave.setStatusByUserName(user.getFullName());
         leave.setStatusByUserUuid(user.getUserUuid());
        leave = leaveLocalService.addLeave(leave);

        AssetEntry assetEntry = assetEntryLocalService.updateEntry( user.getUserId(), serviceContext.getScopeGroupId(), new Date(),
             new Date(), Leave.class.getName(),leave.getLeaveId(), leave.getUuid(), 0, null, null, true, false, new Date(), null,
             new Date(), null, ContentTypes.TEXT_HTML, leave.getLeaveName(), leave.getLeaveName(), null, null, null, 0, 0, null);
        Indexer<Leave> indexer = IndexerRegistryUtil.nullSafeGetIndexer(Leave.class);
        indexer.reindex(leave);

        WorkflowHandlerRegistryUtil.startWorkflowInstance(leave.getCompanyId(), leave.getGroupId(), leave.getUserId(), Leave.class.getName(),
        leave.getPrimaryKey(), leave, serviceContext);
     } catch (PortalException e) {
        e.printStackTrace();
     }
        return leave;
  }
  ```

  - add updateStatus() method which invoked by Workflow handler at runtime to change the status of leave entity

    -
      ```
      public Leave updateStatus(long userId,long leaveId,int status,ServiceContext serviceContext){

       Leave leave = leavePersistence.fetchByPrimaryKey(leaveId);
       leave.setStatus(status);
       leave.setStatusByUserId(userId);
       leave.setStatusDate(new Date());
       User user = null;
       try {
            user = userLocalService.getUser(userId);
            leave.setStatusByUserName(user.getFullName());
             leave.setStatusByUserUuid(user.getUserUuid());
       } catch (PortalException e) {
           e.printStackTrace();
       }
        leave = leavePersistence.update(leave);
       try {
       if (status == WorkflowConstants.STATUS_APPROVED) {
           // update the asset status to visibile
          assetEntryLocalService.updateEntry(Leave.class.getName(), leaveId, new Date(),null, true, true);
       } else {
           // set leave entity status to false
           assetEntryLocalService.updateVisible(Leave.class.getName(), leaveId, false);
       }
       } catch (Exception e) {
           e.printStackTrace();
       }
       return leave;
       }
      ```

- Edit the searc container in view.jsp also:

```
<liferay-ui:search-container emptyResultsMessage="no-leaves-found" iteratorURL="<%=leaveItrUrl %>" >
 <liferay-ui:search-container-results results="<%= LeaveLocalServiceUtil.getLeaveByStatus(scopeGroupId,
WorkflowConstants.STATUS_APPROVED,searchContainer.getStart(),
 searchContainer.getEnd()) %>" >
 </liferay-ui:search-container-results>

 <liferay-ui:search-container-row className="org.javasavvy.leave.model.Leave" modelVar="leave" keyProperty="leaveId" >
 <portlet:renderURL var="rowURL" >
 <portlet:param name="leaveId" value="${leave.leaveId}" />
 <portlet:param name="mvcRenderCommandName" value="viewleave_info"/>
 </portlet:renderURL>
 <liferay-ui:search-container-column-user userId="${leave.userId}" showDetails="false" name="User" />
 <liferay-ui:search-container-column-text property="userName" name="User Name" href="${rowURL}"/>
 <liferay-ui:search-container-column-text property="leaveName" name="Leave Name" href="${rowURL}"/>
 <liferay-ui:search-container-column-date property="startDate" name="Start Date"/>
 <liferay-ui:search-container-column-date property="endDate" name="End Date"/>
 <liferay-ui:search-container-column-status property="status" name="Status" >
 </liferay-ui:search-container-column-status>

 </liferay-ui:search-container-row>
 <liferay-ui:search-iterator />
</liferay-ui:search-container>
```

- Now Liferay workflow integration with custom entity leave is completed.

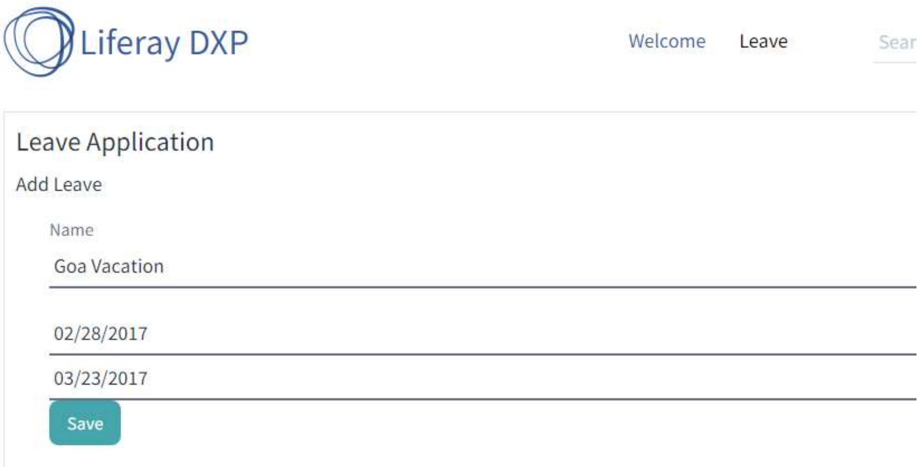
Enable Workflow Configuration in the Control Panel:

Follow the below tutorial to enable to workflow

Liferay DXP Kaleo Workflow configuration

In this tutorial, we will see how to enable workflow for custom entity in the control panel. Liferay allows developers to load the workflow definition file via control panel and enable workflow with custom entity….READ MORE

- Now apply for leave on Liferay portal:



Leave Application

Add Leave

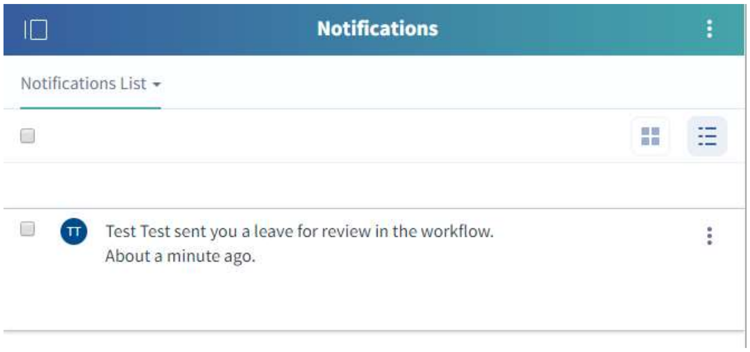
Name

Goa Vacation

02/28/2017

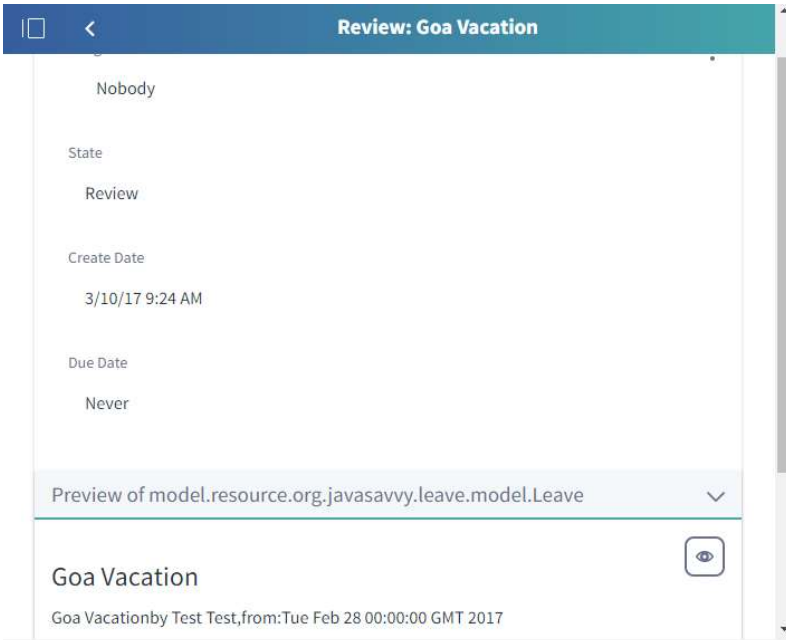
03/23/2017

- A notification will be sent to Portal admin, portlet content reviewer and site admin



Test Test sent you a leave for review in the workflow.
About a minute ago.

- You can also navigate to leave info from notifications and below screen will be opened:



Nobody

State

Review

Create Date

3/10/17 9:24 AM

Due Date

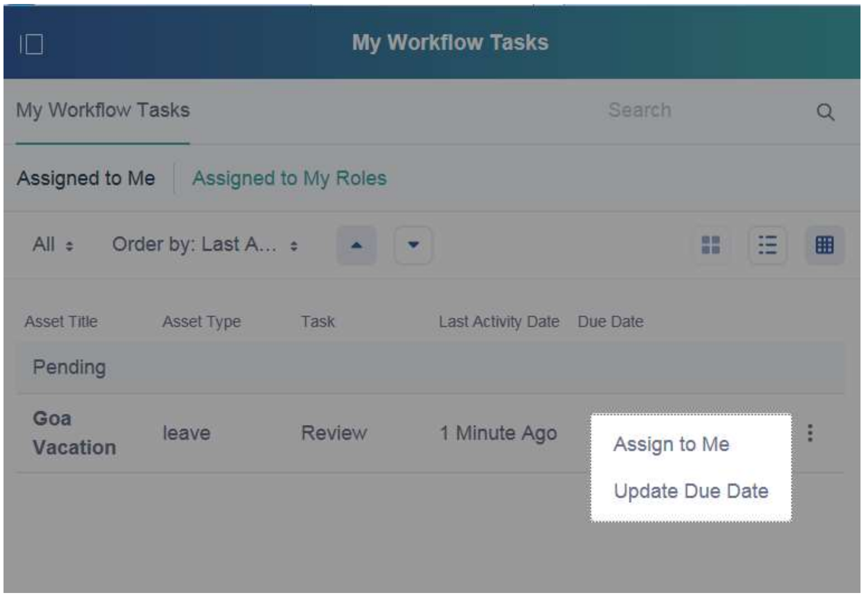
Never

Preview of model.resource.org.javasavvy.leave.model.Leave

Goa Vacation

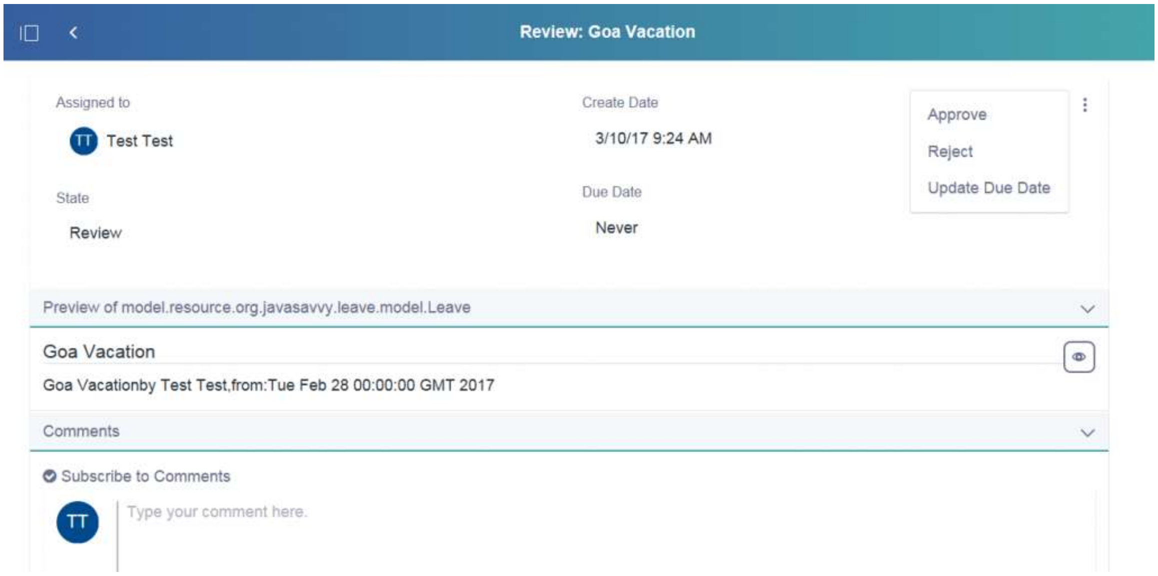
Goa Vacationby Test Test,from:Tue Feb 28 00:00:00 GMT 2017

- Navigate to User -> My Workflow Tasks and click on "Assigned to My Roles".
- You can see the Leaves with pending status and click on actions and select "Assign to me"



Asset Title	Asset Type	Task	Last Activity Date	Due Date
Pending				
Goa Vacation	leave	Review	1 Minute Ago	

- Click on the leave and you can see the leave details. Click on actions and approve the leave.



Assigned to

TT Test Test

State

Review

Create Date

3/10/17 9:24 AM

Due Date

Never

Preview of model.resource.org.javasavvy.leave.model.Leave

Goa Vacation

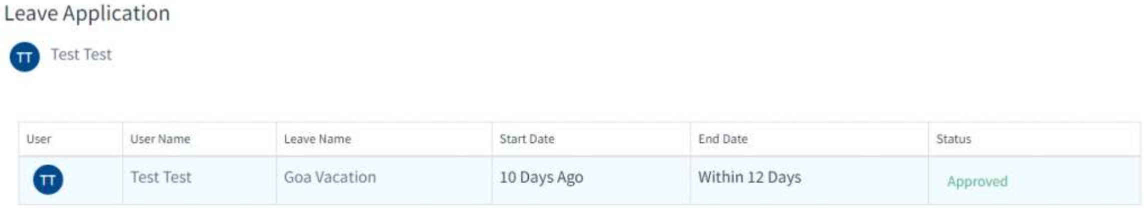
Goa Vacationby Test Test,from:Tue Feb 28 00:00:00 GMT 2017

Comments

Subscribe to Comments

TT Type your comment here.

- Refresh the leave page and you can see the leaves with status “Approved”



Leave Application

TT Test Test

User	User Name	Leave Name	Start Date	End Date	Status
TT	Test Test	Goa Vacation	10 Days Ago	Within 12 Days	Approved

Hope this helps:

Some time you might get below error and you won’t able to see the created entities under my workflow tasks tab:

```
ERROR [liferay/kaleo_graph_walker-2][runtime:60] Error executing FreeMarker template FreeMarker template error:
The following has evaluated to null or missing:
 ==> entryType [in template "Review Notificationcom.liferay.portal.workflow.kaleo.model.KaleoNode28907"
 at line 1, column 26] ---- Tip: If the failing expression is known to be legally refer to
 something that's sometimes null or missing,
 either specify a default value like myOptionalVar!myDefault, or use <#if myOptionalVar??>when-present<#else>when-missing</#if>.
 (These only cover the last step of the expression; to cover the whole expression, use parenthesis:
 (myOptionalVar.foo)!myDefault, (myOptionalVar.foo)?? ---- ---- FTL stack trace ("~" means nesting-related):
 - Failed at: ${entryType} [in template
 "Review Notificationcom.liferay.portal.workflow.kaleo.model.KaleoNode28907" at line 1, column 24]
 ---- Java stack trace (for programmers):
```

If you get the below error then make sure that you have added below method in LiferayWorkflowHandler class,

```
 @Override
  public String getType(Locale locale) {
     return "leave";
  }
```

« Liferay DXP Kaleo Workflow configuration

Liferay 7 Asset Render Factory Example »

YOU MISSED

EXCEL PARSING WITH APACHE POI

How to remove duplicate elements from ArrayList

AUG 30, 2019 JAVASAVVY

JAVA

Tomcat Maven Deployment in Eclipse

SEP 22, 2017 JAVASAVVY

LIFERAY7

Liferay Gradle SSLHandshake or PKIX path building Issue

SEP 11, 2017 JAVASAVVY

JAVA

Import SSL Certificate into Java keystore

SEP 11, 2017 JAVASAVVY