

## CronJobs

It is like a timer.

It is a scheduled program.

If you have specific task which needs to be executed at periodic (like Every Sunday @ 8AM IST) → then put that task inside the **Cron job**.

**Example:** - I want to send promotion for new customer (who are within last 1 month) @ every **Sunday 8 AM IST**.

**Example:** - I want to email **daily @ 7 AM IST** about **Top 10 order details** (In terms of the price) to product manager.

**Example:** - Load new products into “SAP Comm” every day @ 7 AM IST.

**Q:** What kind of logic we can write in Cronjob program?

If we have something which needs to be executed at periodic intervals (or) at scheduled time – Then put that something inside the CronJob.

**Q:** What Cronjob contains?

**1) Trigger** = This is used to schedule when to run the job.

For this – We will use “**Cron Expression**”.

# CRON Expressions

A CRON expression is a string representing the schedule for a particular command to execute. The parts of a CRON schedule are as follows:

```
* * * * *  
- - - - -  
| | | | |  
| | | | + year [optional]  
| | | +---- day of week (0 - 7) (Sunday=0 or 7)  
| | +----- month (1 - 12)  
| +----- day of month (1 - 31)  
+----- hour (0 - 23)  
----- min (0 - 59)
```

## Format for cron expression:

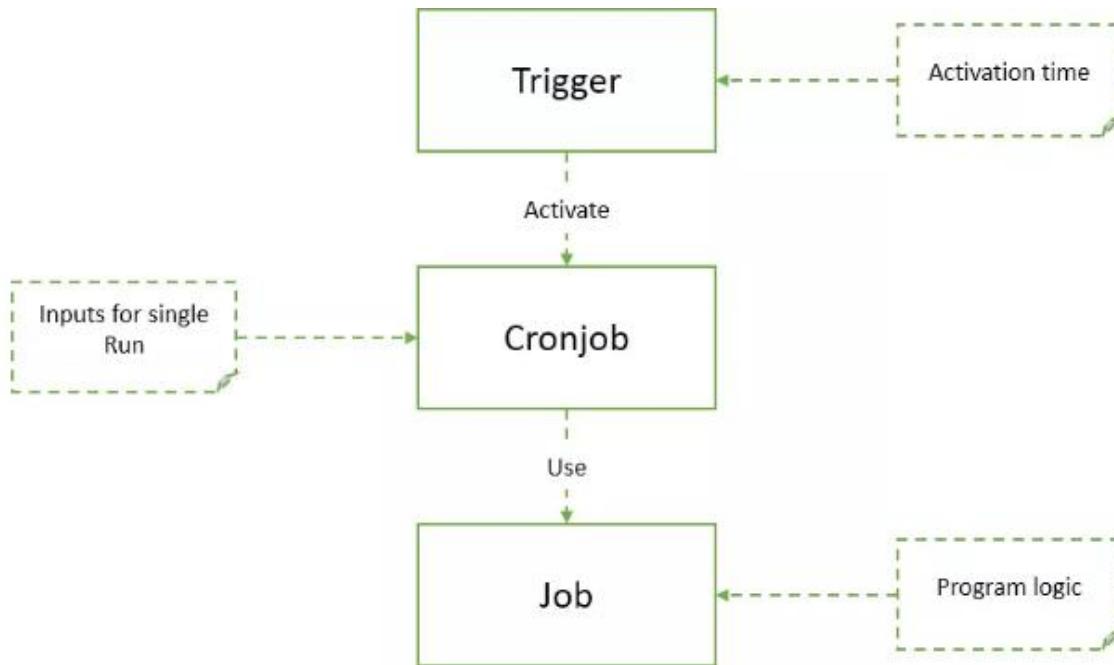
Field Name	Mandatory	Allowed Values	Allowed Special Characters
Seconds	YES	0-59	, - * /
Minutes	YES	0-59	, - * /
Hours	YES	0-23	, - * /
Day of month	YES	1-31	, - * ? / L W
Month	YES	1-12 or JAN-DEC	, - * /
Day of week	YES	1-7 or SUN-SAT	, - * ? / L #
Year	NO	empty, 1970-2099	, - * /

**2) Cronjob** = This hold the business logic which needs to be executed at specified time.

**3) Job** = This consists of logic which defines by “Job Performable”.

## 2 Ways: -

- Create a class which extends **AbstractJobPerformable**
- Create a class which implements **JobPerformable**



**Example:** - We already created “**Courses**” itemtype (Table name = **TrainingCourses**). It has the records.

The screenshot shows the hybris administration console interface. The top navigation bar includes links for Platform, Monitoring, Maintenance, Console, Scripting Languages, FlexibleSearch, ImpEx Import (which is highlighted with a green box), ImpEx Export, and LDAP. The main content area is divided into three sections: 1) An import form titled "Import script" with fields for "Choose file" (set to "securemedias (3).zip"), "Script encoding" (UTF-8), "Max. threads" (1), "Import validation mode" (import\_strict), "Legacy mode" (unchecked), "Enable code execution" (checked with a green checkmark), "Distributed mode" (unchecked), and "Direct persistence" (unchecked). A large green arrow points from the "Import file" button at the bottom to the "Import script" tab. 2) A sidebar titled "Inbox" showing categories: System, Catalog, RRRSTraining (which is expanded), Courses (highlighted with a green box), and Multimedia. A green arrow points from the "Courses" link in the sidebar to the table on the right. 3) A table titled "Courses" listing course details:

Course Code	Course Name	Course Duration	Course Amount
RRRS103	C Lang New	60	500
RRRS102	C Lang	60	500
RRRS101	RRRS Hybris123	80 Hrs	500
101	Latest Hybris123	80 Hrs	500

This table is having itemtype = Courses

	P_CODE	P_NAME	P_DURATION	P_AMOUNT
	102	Data Hub	55 Hrs	500
	103	C Lan	55 Hrs	500

Course Code	Course Name	Course Duration	Course Amount
DH100	DataHub	55	500
HY410	Hybris	80	500

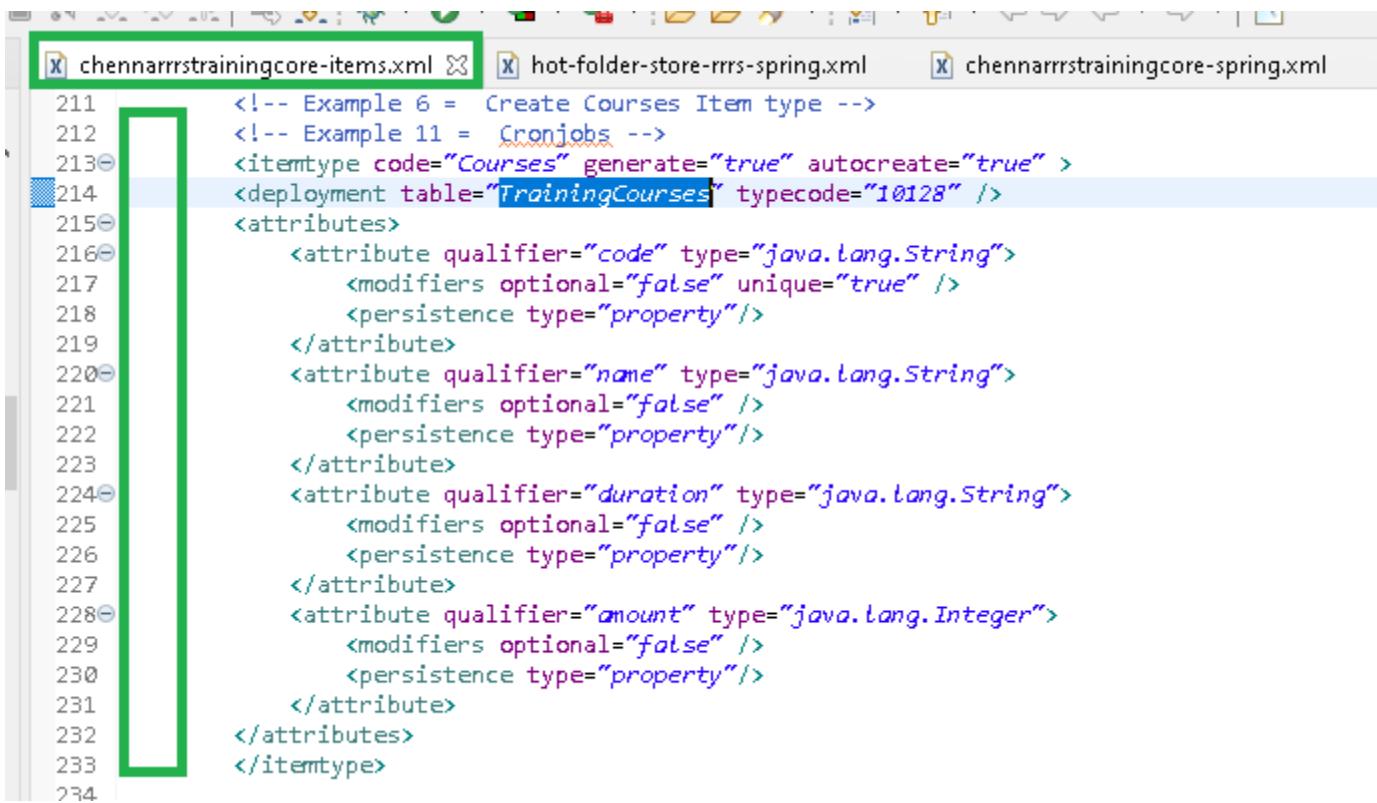
**Business Scenario = Read “TrainingCourses” table records (Or) Courses Itemtype records: -**

- 1) Display records in console every 1 min (Display in Logs every 1 min)
- 2) Email those Records every 1 min

**Step 1 = Create Courses Itemtype (or) TrainingCourses table.**

Also – Insert the records.

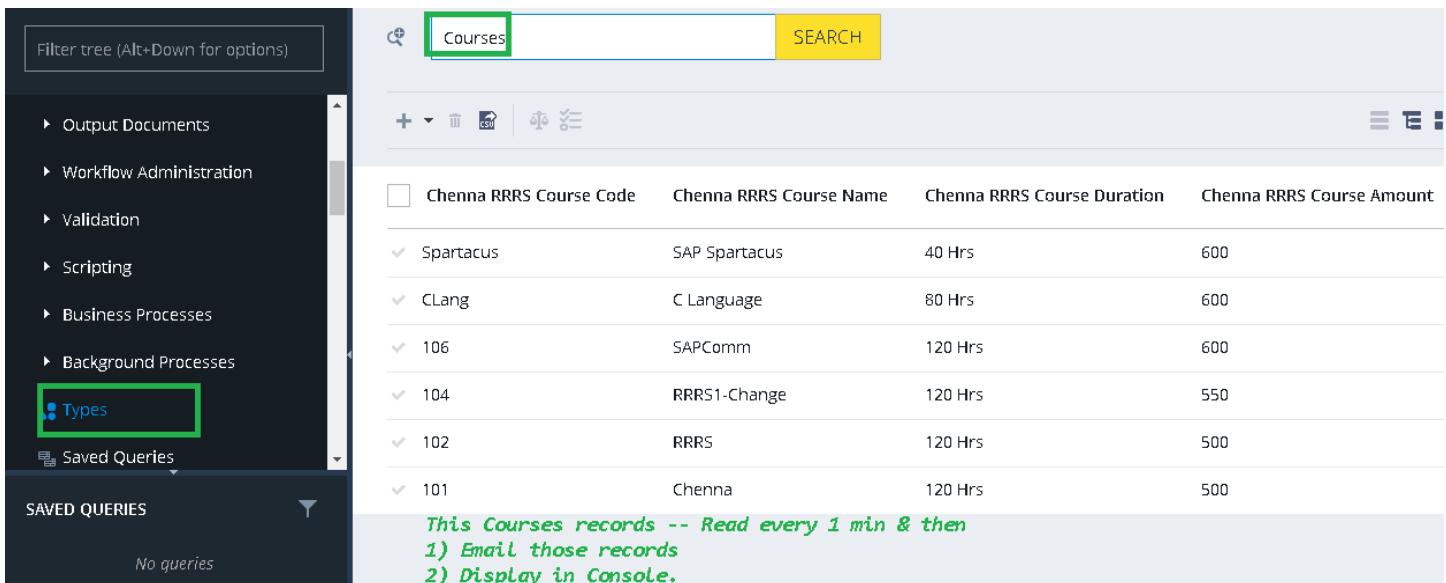
**Example =**



```

211      <!-- Example 6 = Create Courses Item type -->
212      <!-- Example 11 = Cronjobs -->
213      <itemtype code="Courses" generate="true" autocreate="true" >
214          <deployment table="TrainingCourses" typecode="10128" />
215          <attributes>
216              <attribute qualifier="code" type="java.lang.String">
217                  <modifiers optional="false" unique="true" />
218                  <persistence type="property"/>
219              </attribute>
220              <attribute qualifier="name" type="java.lang.String">
221                  <modifiers optional="false" />
222                  <persistence type="property"/>
223              </attribute>
224              <attribute qualifier="duration" type="java.lang.String">
225                  <modifiers optional="false" />
226                  <persistence type="property"/>
227              </attribute>
228              <attribute qualifier="amount" type="java.lang.Integer">
229                  <modifiers optional="false" />
230                  <persistence type="property"/>
231              </attribute>
232          </attributes>
233      </itemtype>
234

```



Filter tree (Alt+Down for options)

- Output Documents
- Workflow Administration
- Validation
- Scripting
- Business Processes
- Background Processes
- Types**
- Saved Queries

SAVED QUERIES

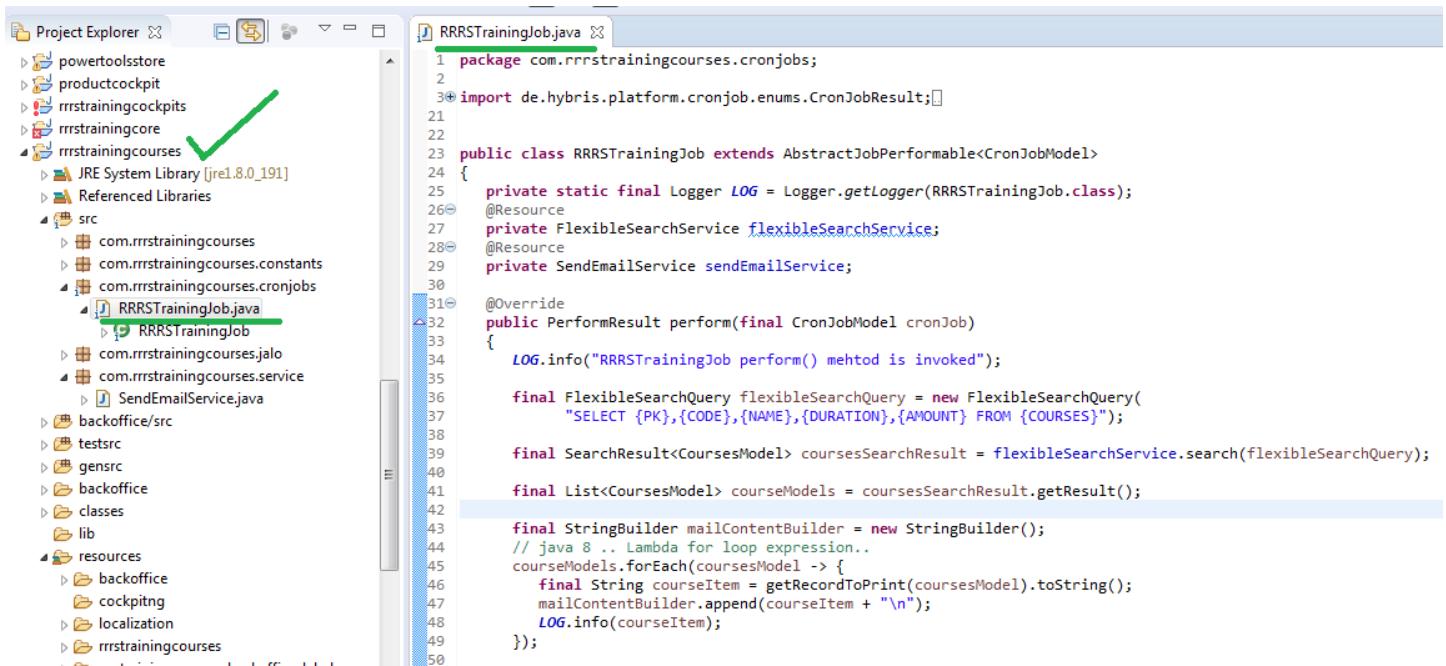
No queries

	Chenna RRRS Course Code	Chenna RRRS Course Name	Chenna RRRS Course Duration	Chenna RRRS Course Amount
✓	Spartacus	SAP Spartacus	40 Hrs	600
✓	CLang	C Language	80 Hrs	600
✓	106	SAPComm	120 Hrs	600
✓	104	RRRS1-Change	120 Hrs	550
✓	102	RRRS	120 Hrs	500
✓	101	Chenna	120 Hrs	500

This Courses records -- Read every 1 min & then  
 1) Email those records  
 2) Display in Console.

**Step 2 = Create new class called “RRRSTrainingJob.java” by extending “AbstractJobPerformable”.**

AbstractJobPerformable is having a method called “**perform()**” & we need to override this method our own business logic.

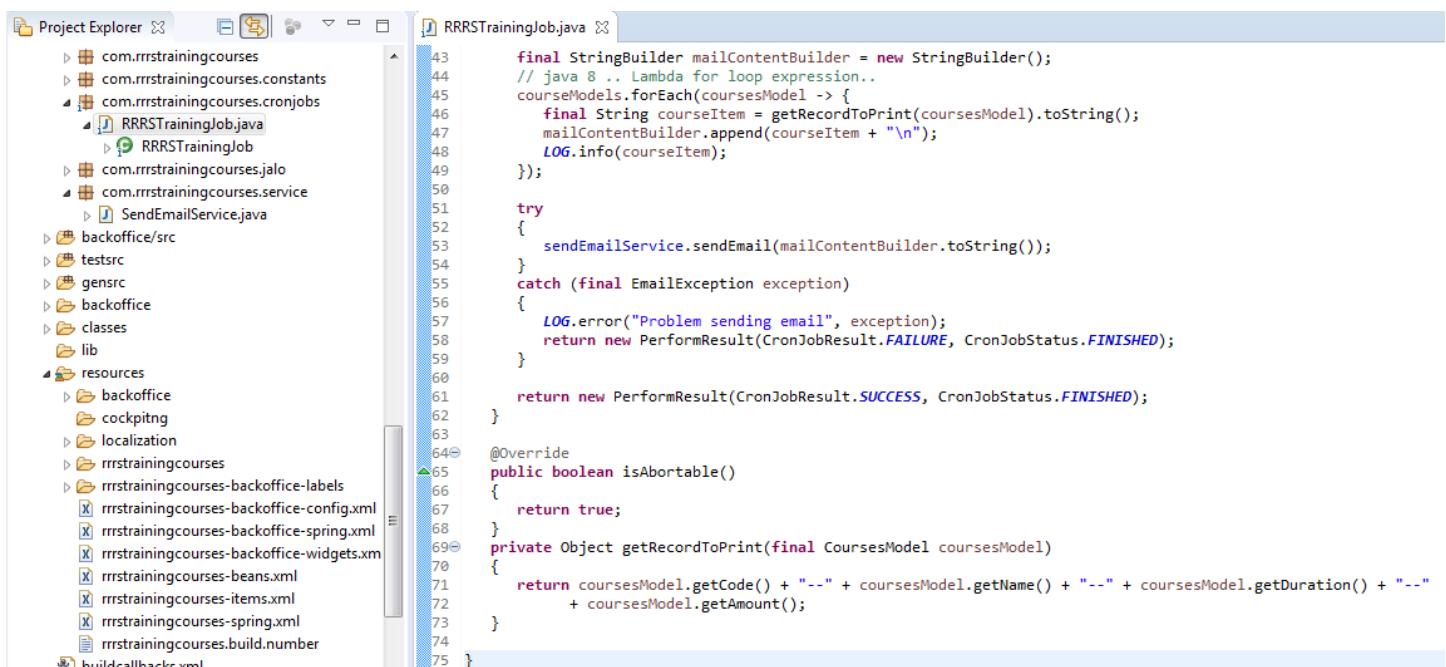


Project Explorer

```

1 package com.rrrstrainingcourses.cronjobs;
2
3 import de.hybris.platform.cronjob.enums.CronJobResult;
4
5 public class RRRTrainingJob extends AbstractJobPerformable<CronJobModel>
6 {
7     private static final Logger LOG = Logger.getLogger(RRRTrainingJob.class);
8
9     @Resource
10    private FlexibleSearchService flexibleSearchService;
11
12    private SendEmailService sendEmailService;
13
14    @Override
15    public PerformResult perform(final CronJobModel cronJob)
16    {
17        LOG.info("RRRTrainingJob perform() method is invoked");
18
19        final FlexibleSearchQuery flexibleSearchQuery = new FlexibleSearchQuery(
20            "SELECT {PK},{CODE},{NAME},{DURATION},{AMOUNT} FROM {COURSES}");
21
22        final SearchResult<CoursesModel> coursesSearchResult = flexibleSearchService.search(flexibleSearchQuery);
23
24        final List<CoursesModel> courseModels = coursesSearchResult.getResult();
25
26        final StringBuilder mailContentBuilder = new StringBuilder();
27        // java 8 .. Lambda for loop expression..
28        courseModels.forEach(coursesModel -> {
29            final String courseItem = getRecordToPrint(coursesModel).toString();
30            mailContentBuilder.append(courseItem + "\n");
31            LOG.info(courseItem);
32        });
33
34    }
35
36    @Override
37    public boolean isAbortable()
38    {
39        return true;
40    }
41
42    private Object getRecordToPrint(final CoursesModel coursesModel)
43    {
44        return coursesModel.getCode() + "--" + coursesModel.getName() + "--" + coursesModel.getDuration() + "--"
45            + coursesModel.getAmount();
46    }
47
48 }
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```



Project Explorer

```

1 package com.rrrstrainingcourses;
2
3 import com.rrrstrainingcourses.constants;
4
5 import com.rrrstrainingcourses.cronjobs;
6 import com.rrrstrainingcourses.jalo;
7 import com.rrrstrainingcourses.service;
8
9 import de.hybris.platform.cronjob.enums.CronJobResult;
10
11 public class RRRTrainingJob extends AbstractJobPerformable<CronJobModel>
12 {
13     private static final Logger LOG = Logger.getLogger(RRRTrainingJob.class);
14
15     @Resource
16    private FlexibleSearchService flexibleSearchService;
17
18    private SendEmailService sendEmailService;
19
20    @Override
21    public PerformResult perform(final CronJobModel cronJob)
22    {
23        final StringBuilder mailContentBuilder = new StringBuilder();
24        // java 8 .. Lambda for loop expression..
25        courseModels.forEach(coursesModel -> {
26            final String courseItem = getRecordToPrint(coursesModel).toString();
27            mailContentBuilder.append(courseItem + "\n");
28            LOG.info(courseItem);
29        });
30
31        try
32        {
33            sendEmailService.sendEmail(mailContentBuilder.toString());
34        }
35        catch (final EmailException exception)
36        {
37            LOG.error("Problem sending email", exception);
38            return new PerformResult(CronJobResult.FAILURE, CronJobStatus.FINISHED);
39        }
40
41        return new PerformResult(CronJobResult.SUCCESS, CronJobStatus.FINISHED);
42    }
43
44    @Override
45    public boolean isAbortable()
46    {
47        return true;
48    }
49
50    private Object getRecordToPrint(final CoursesModel coursesModel)
51    {
52        return coursesModel.getCode() + "--" + coursesModel.getName() + "--" + coursesModel.getDuration() + "--"
53            + coursesModel.getAmount();
54    }
55
56 }
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

**Step 3 = Create an Email Service called “SendEmailService.java”**

The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer view displays the project structure under 'rrrstrainingcourses'. It includes packages like 'rrrstrainingcockpits', 'rrrstrainingcore', and 'rrrstrainingcourses', along with 'Referenced Libraries' and 'src' containing Java files such as 'RRRSTrainingJob.java' and 'SendEmailService.java'. A green checkmark is placed next to 'SendEmailService.java'. On the right, the code editor shows the 'SendEmailService.java' file:

```

1 package com.rrrstrainingcourses.service;
2
3 import de.hybris.platform.util.mail.MailUtils;
4
5 import org.apache.commons.mail.Email;
6 import org.apache.commons.mail.EmailException;
7 import org.springframework.stereotype.Service;
8
9
10 @Service
11 public class SendEmailService
12 {
13     public void sendEmail(final String mailContent) throws EmailException
14     {
15         sendMail(mailContent);
16     }
17
18     private void sendMail(final String mailContent) throws EmailException
19     {
20         final Email email = MailUtils.getPreConfiguredEmail();
21         email.setSubject("Testing RRRS Training Cronjob");
22         email.addTo(email.getReplyToAddresses().get(0).toString());
23         email.setMsg(mailContent);
24         email.setSSL(true);
25         email.send();
26     }
27
28 }

```

**Step 4 = we have written “RRRSTrainingJob.java” file & “SendEmailService.java”.**

== So, it's time to register those classes (or) Beans.

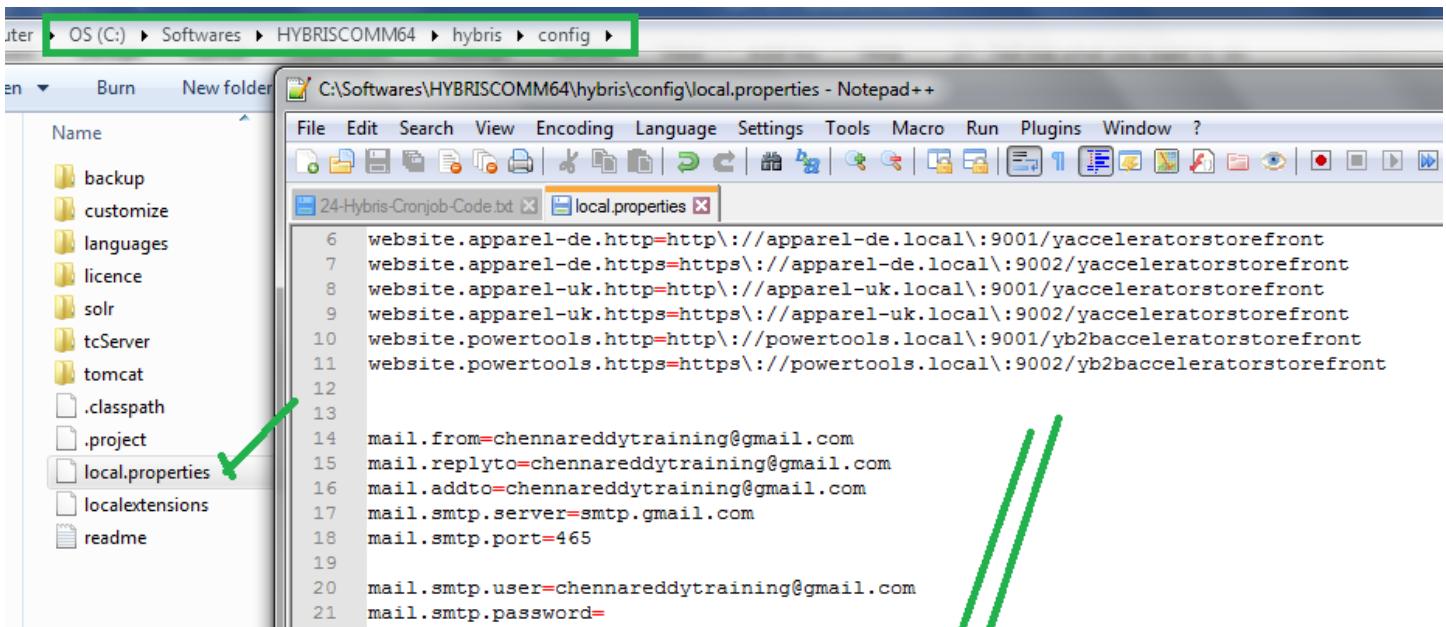
The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer view displays the project structure under 'rrrstrainingcourses'. It includes 'JRE System Library [jre1.8.0\_191]', 'Referenced Libraries', 'src', 'resources', and various configuration files like 'rrrstrainingcourses-backoffice-labels' and 'rrrstrainingcourses-spring.xml'. A green checkmark is placed next to 'rrrstrainingcourses-spring.xml'. On the right, the code editor shows the 'rrrstrainingcourses-spring.xml' file:

```

59 <!--
60
61
62     <!--
63         <bean id="rrrstrainingcoursesProfBean" class="com.hybris.backoffice.aop.YbackofficeProfilingAspect"/>
64             <aop:config proxy-target-class="true">
65                 <aop:aspect id="rrrstrainingcoursesProfAspect" ref="rrrstrainingcoursesProfBean">
66                     <aop:pointcut id="profiledMethods"
67                         expression="execution(* getModificationTime(..))" />
68                     <aop:around pointcut-ref="profiledMethods" method="profile" />
69                 </aop:aspect>
70             </aop:config>
71     <!-- some other examples of a pointcut that matches everything:
72
73         <aop:pointcut id="profiledMethods"
74             expression="bean(de.hybris.platform.jalo.user.Customer) &&
75                 !execution(* getPK(..))" />
76         <aop:pointcut id="profiledMethods" expression="execution(* *(..))" />
77
78     -->
79
80         <bean id="sendEmailService" class="com.rrrstrainingcourses.service.SendEmailService" />
81
82         <bean id="rrrsTrainingJob" class="com.rrrstrainingcourses.cronjobs.RRRSTrainingJob" >
83             <property name="modelService" ref="modelService"/>
84             <property name="flexibleSearchService" ref="flexibleSearchService"/>
85             <property name="sessionService" ref="sessionService"/>
86         </bean>
87
88     </beans>
89
90
91
92
93
94
95
96
97
98
99

```

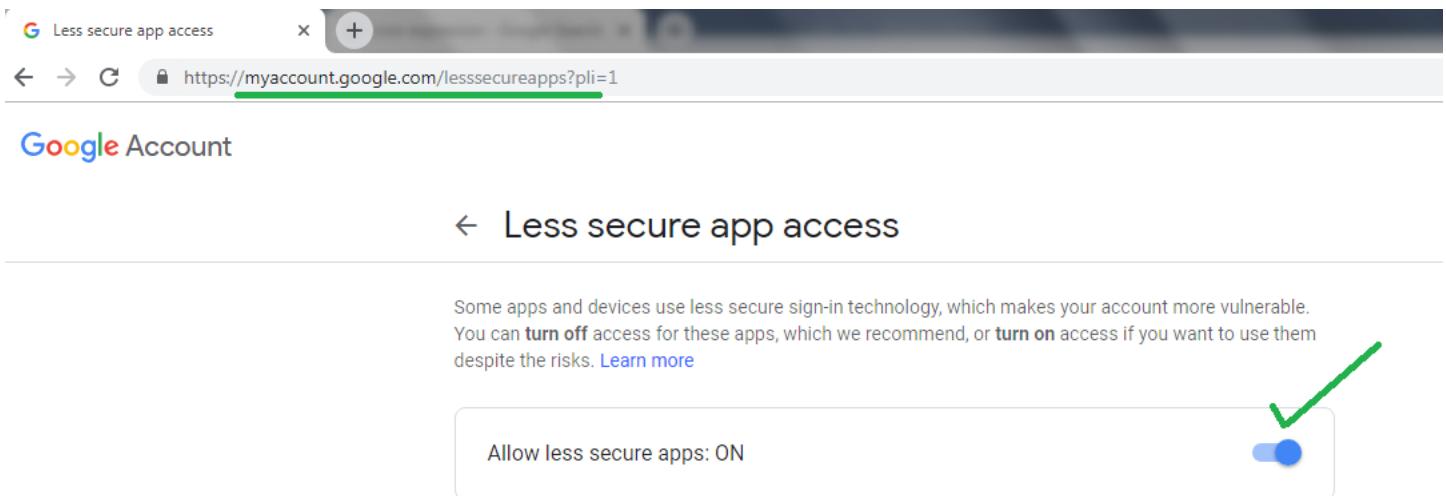
## Step 5 = Define email properties in “local.properties” file



```
6 website.apparel-de.http=http://apparel-de.local:9001/yacceleratorstorefront
7 website.apparel-de.https=https://apparel-de.local:9002/yacceleratorstorefront
8 website.apparel-uk.http=http://apparel-uk.local:9001/yacceleratorstorefront
9 website.apparel-uk.https=https://apparel-uk.local:9002/yacceleratorstorefront
10 website.power-tools.http=http://power-tools.local:9001/yb2bacceleratorstorefront
11 website.power-tools.https=https://power-tools.local:9002/yb2bacceleratorstorefront
12
13
14 mail.from=chennareddytraining@gmail.com
15 mail.replyto=chennareddytraining@gmail.com
16 mail.addto=chennareddytraining@gmail.com
17 mail.smtp.server=smtp.gmail.com
18 mail.smtp.port=465
19
20 mail.smtp.user=chennareddytraining@gmail.com
21 mail.smtp.password=
```

## Step 6 = Port 465 works only if we make allow less secure apps “ON” in your Gmail account.

URL = <https://myaccount.google.com/lesssecureapps?pli=1>



## Step 7 = Do the build (ant clean all)

## Step 9 = Start the Server (hybrisserver.bat)

## Step 9 = Perform the Platform Update (hAC – Update)

## Step 10 = Create Cronjob & Register Cronjob using Trigger.

The screenshot displays two main sections of the hAC interface:

**Left Side (Cronjob Creation):**

- A tree view on the left shows nodes like CRONJOBHISTORIES522SN, CRONJOBHISTORIES, CRONJOBS, CRONJOBS501SN, CRONJOBSLP, and CSAGENTGROUP2BA9007SN.
- An "Info" tab for the "CRONJOBS" node is open, showing properties: catalogName, childTables, exportedKeys, importedKeys, and qualifiedName.
- A code editor window titled "processing-items.xml" contains XML code for creating a CronJob:

```
544
545
546    <itemtype code="CronJob" jaloclass="de.hybris.platform.cronjob.jalo.
547        autocreate="true"
548        generate="true">
549            <deployment table="CronJobs" typecode="501"/>
550            <custom-properties>
551                <property name="systemType">
552                    <value>java.lang.Boolean.TRUE</value>
553                </property>
554                <property name="LegacyPersistence">
555                    <value>java.lang.Boolean.TRUE</value>
```

**Right Side (Trigger Registration):**

- A tree view on the right shows nodes like TESTITEMLP, TITLES, TITLES24SN, TITLESLP, TRIGGERSCJ, and TRIGGERSCJ502SN.
- An "Info" tab for the "TRIGGERSCJ" node is open, showing properties: catalogName, childTables, exportedKeys, importedKeys, and qualifiedName.
- A code editor window titled "processing-items.xml" contains XML code for creating a Trigger:

```
86 /           </itemtype>
868
869
870    <itemtype code="Trigger" jaloclass="de.hybris.platform.c
871        autocreate="true"
872        generate="true">
873            <deployment table="TriggersCJ" typecode="502"/>
874            <custom-properties>
875                <property name="systemType">
876                    <value>java.lang.Boolean.TRUE</value>
877                </property>
878                <property name="LegacyPersistence">
879                    <value>java.lang.Boolean.TRUE</value>
```

**Bottom Section (Job Configuration):**

- A tree view shows nodes like COUPONREDEMPTION, CRONJOBHISTORIES522SN, CRONJOBS, CRONJOBS501SN, CRONJOBSLP, CSAGENTGROUP2BA9007SN, CSAGENTGROUP2BASESTOF, CSCHANGEEVENTEN9004SN, CSCHANGEEVENTENTRY, CSTICKETEMAIL, and CSTICKETEMAIL5005SN.
- An "Info" tab for the "CRONJOBS" node shows properties: catalogName (set to PUBLIC), childTables, exportedKeys, importedKeys, qualifiedName, remarks, schemaName, simpleName, and type.
- A note in green text says: "Here you need to add your chennaTrainingJob". Below it is a code snippet:

```
INSERT_UPDATE CronJob; code[unique=true];job(code);singleExecutable;sessionLanguage(isocode)
;chennaTrainingJob;chennaTrainingJob;false;en
```

- An "Info" tab for the "TRIGGERSCJ" node shows properties: catalogName (set to PUBLIC), childTables, exportedKeys, importedKeys, qualifiedName, remarks, schemaName, simpleName, and type.
- A note in green text says: "Here you need to specify the Cron Expression". Below it is a code snippet:

```
Date & Time
INSERT_UPDATE Trigger;cronjob(code)[unique=true];cronExpression
;chennaTrainingJob; 0 * * ? *
```

You're Administrator [logout](#)

Platform Monitoring Maintenance Console

Scripting Languages FlexibleSearch **ImpEx Import** ImpEx Export LDAP

Import content Import script

**Import content**

```
1 INSERT_UPDATE CronJob; code[unique=true];job(code);singleExecutable;sessionLanguage
2 ;rrrsTrainingJob;rrrsTrainingJob;false;en
3
4 INSERT_UPDATE Trigger;cronjob(code)[unique=true];cronExpression
5 ;rrrsTrainingJob; 0 * * ? * *
```



INSERT\_UPDATE CronJob; code[unique=true];job(code);singleExecutable;sessionLanguage(isocode)  
;rrrsTrainingJob;rrrsTrainingJob;false;en

INSERT\_UPDATE Trigger;cronjob(code) [unique=true];cronExpression  
;rrrsTrainingJob; 0 \* \* ? \* \*

## Step 11 = Test the Results: -

SAP Administration Cockpit

cron

System

Background Processes

CronJobs

SEARCH

chennaTrainingJob : chennaTrainingJob - UNKNOWN - UNKNOWN

LOG TASK RUN AS TIME SCHEDULE SYSTEM RECOVERY ADMINISTRATION

ESSENTIAL

Code	Current status	Job definition	Last result
chennaTrainingJob	FINISHED	chennaTrainingJob	SYSTEM ERROR

Timetable

seconds: 0 minutes: \* hours: \* day

Last start time: Jun 8, 2021 6:46:00 AM

Enabled: True

Last end time: Jun 8, 2021 6:46:01 AM

## Testing RRRS Training Cronjob

Inbox x

chennareddytraining@gmail.com <chennareddytraining@gmail.com>  
to me ▾

10:02 PM

101--Latest Hybris123--80 Hrs--500  
RRRS101--RRRS Hybris123--80 Hrs--500  
RRRS102--C Lang--60--500  
RRRS103--C Lang New--60--500

OS (C:) > rrrssoftware > HYBRISCOMM64 > hybris > log > tomcat

Print Burn

\*C:\Softwares\HYBRISCOMM64\hybris\log\tomcat\console-20190317.log - Notepad++

Name

- access.2019-03-06
- access.2019-03-07
- access.2019-03-08
- access.2019-03-09
- access.2019-03-10
- access.2019-03-11
- access.2019-03-12
- access.2019-03-13
- access.2019-03-14
- access.2019-03-15
- access.2019-03-16
- access.2019-03-17

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

console-20190317.log

```
4567 INFO | jvm 1 | main | 2019/03/17 22:03:08.442 | INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] 101--Latest Hybris123--80 Hrs--500
4568 INFO | jvm 1 | main | 2019/03/17 22:03:08.442 | INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
4569 INFO | jvm 1 | main | 2019/03/17 22:03:08.442 | INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
4570 INFO | jvm 1 | main | 2019/03/17 22:03:08.442 | INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
4571
```

C:\Windows\System32\cmd.exe - hybrisserver.bat

```
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS101--RRRS Hybris123--80 Hrs--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS102--C Lang--60--500
INFO [rrrsTrainingJob::de.hybris.platform.servicelayer.internal.jalo.ServiceLayerJob] RRRS103--C Lang New--60--500
```

**Note:** - We can also run the Cronjobs manually.

The screenshots illustrate the configuration of a CronJob in the hybris Management Console (HMC). The left sidebar shows various system modules like Inbox, System, Advanced Configuration, Tools, Output Documents, Types, Saved Queries, Report Definitions, Personalization, hMC configuration, Workflow Administration, LDAP, and CronJobs. The middle screenshot shows the 'Editor - CronJob' page with the 'Time Schedule' tab selected. It displays the job code 'rrsTrainingJob', job definition 'rrsTrainingJob', timetable 'seconds: 0 minutes: \* hours: \* daysOfMonth: ?', and an enabled checkbox. The right screenshot shows the 'Trigger' configuration page with the 'Time Values' tab selected. It shows the cronjob 'rrsTrainingJob', job 'rrsTrainingJob', and an active checkbox. The execution interval is set to 'Cron expression defined' with the cron expression '0 \* ? \* \*'. A note states: 'The trigger can only be set for a job OR a cronjob. Not for both.'

## Composite Cron Jobs: -

<https://www.stackextend.com/hybris/use-composite-cronjob-in-hybris/>

**Q:** How to Start a CronJob? = There are different ways to start a cronjob which are given below :

- Manually start using HMC = → hmc → system → cronjobs → select CronJob → "StartCronJobNow".
- Automatically running the CronJob Through Impex file
- Using the ant command → ant runcronjob -d cronjob="CronJobName "
- Using the javacode using the CronJob services we can run the cronjob.

**Q** = What are the major different types of the preconfigured cronjobs in “SAP Comm”?

- SOLR and Lucene related: indexing, updating, removing data
- Clean up unnecessary data from the database or file system
- Product Catalog synchronization
- Regular data export (Product, Price, Inventory, Order Status, and .... Import / Export).
- Workflow
- Impex import.

**Q** = How to stop a cronjob? = We can stop the cronjob by following ways:

- Using the abort method in the java code. It is done automatically after performing the CronJob.
- Manually from hmc we can stop the CronJob.

**Q** = Where to see the created CronJob? = hmc → System → CronJobs

**Q** = How to see the Job Details? =

```
select * from {servicelayerjob} where {code} = 'RRRSTrainingJob'
```

**Q** = How to Run Cron Job through Ant? =

```
ant runcronjob -Dcronjob=rrrsCronJob -Dtenant=master
```

**Q =** What are setting session related attributes to the cron job?

Some time we write a Cron job whose logic requires some session attributes like user, sessionLanguage and sessionCurrency etc.

So how do we set these attributes to the cron job so that we can access them while writing the logic of the cron job ?

It can be done in any one of the 2 ways listed below

- 1) Set the session attributes through impex
- 2) Set the session attributes through code

**Q =** The “SAP Comm” commercefacades extension facades mostly return?

- a) Data objects                    b) Model objects                    c) Data model d) none

**Q =** “SAP Comm” ServiceLayer Models should not be used as part of a facade interface, maintaining a clean abstraction of the?

- a) Business layer and the persistence layer
- b) Business layer and the presentation layer**
- c) Persistence layer and the presentation layer
- d) All

**Q =** All converters should be Spring configured only and should use the ---  
----- base class

- a) **AbstractConverter**                    b) DefaultConverter  
c) a & b                                    d) none of above

**Q** = -----Is an implementation of a Populator pipeline where each population step is evaluated against a Set of Enum values passed by the caller.

**a) DefaultConfigurablePopulator**

b) AbstractPopulatingConverter

c) AbstractConverter

**Q** = Product Populator are a little unique in that they typically extend a --- ----- class that is variant aware and supports the ability of falling back to a variants parent product for attribute values in the event of the source product value being null

a) DefaultProductPopulator

b) AbstractProductPopulatingConverter

**c) AbstractProductPopulator**      d) None

**Q** = .....extension is the template shipped with the “SAP Comm” Accelerator that you can use as a starting point for your own extension.

**a) yacceleratorfacades** b) commercefacades

c) both                          d) none

**Q** = Data Transfer Objects (DTOs) are objects created to contain....

**a) Values**                          b) Business logic

c) Both                                  d) none of them

**Q** = .....template method that allows a concrete sub-class to pick the appropriate target data object implementation.

- a) createListTarget
- b) createTarget**
- c) createSourceToTarget
- d) None of the above

**Q** = Facades are which scoped Spring managed beans

- a) yrequest
- b) singleton**
- c) tenant
- d) request

**Q** = ..... Provides access to various internationalization switches that the user can make when they visit a specific storefront

- a) Store Session façade**
- b) Store Locator façade
- c) User façade
- d) User Locale façade

**Q = Explain different types of interceptors?**

**Model Interceptors** = Intercept the behavior of the lifecycle of Models. Model lifecycle consists of loading from Database, saving to Database and deleting or removing from the Database.

In lifecycle of a model, interceptors can intercept & modify model data. It includes auditing, validating & even restricting the data from being removed/deleted from database if certain conditions are not met.

**There are 5 types of Interceptors: -**

- a) LoadInterceptor = Invoked whenever a model is loaded from the database

- b)** `InitDefaultsInterceptor` = Invoked during `modelService.create()` & `modelService.initDefaults()`
- c)** `PrepareInterceptor` = Invoked before model is saved to database & before `ValidateInterceptor`
- d)** `ValidateInterceptor` = Invoked before a model is saved to database & after `PrepareInterceptor`
- e)** `RemoveInterceptor` = Invoked before a model is removed from database.

**Note** = If any Error like → **javax.servlet.ServletException: File &quot;/WEB-INF/views/responsive/cms/assistedservicecomponent.jsp&quot; not found**

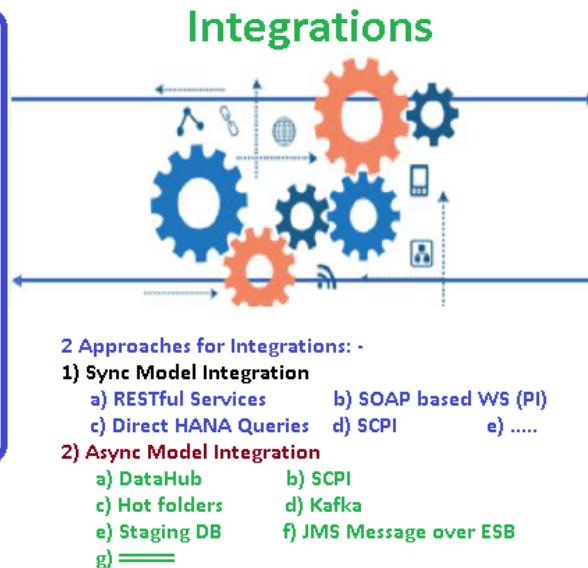
**Solution** = `ant addoninstall -Daddonnames="assistedservicestorefront" -DaddonStorefront.yacceleratorstorefront="rrrstrainingstorefront"`

## Integrations: -

Q = What is the purpose of "SAP Comm"?  
Shopping (or) Placing the Order.

Q = What "SAP Comm" provides?  
Cart ... Checkout ... User ...  
Promotions ... Products ...  
Categories ... Order Status ...  
Order Submit ....

### SAP Comm System



Note = After / Before placing the order (or) shopping -- There are N number of activities to take care.

Example = Shipping, Delivery, Inventory Manage, Order Manage, Warehouse Manage, Pricing, Fulfilment, Availability and ...

These happens generally in 3rd party system like - SAP / PS / and ....

### SAP [3rd Party System]

**Note = Sync** Integration – We can do in N number of ways.

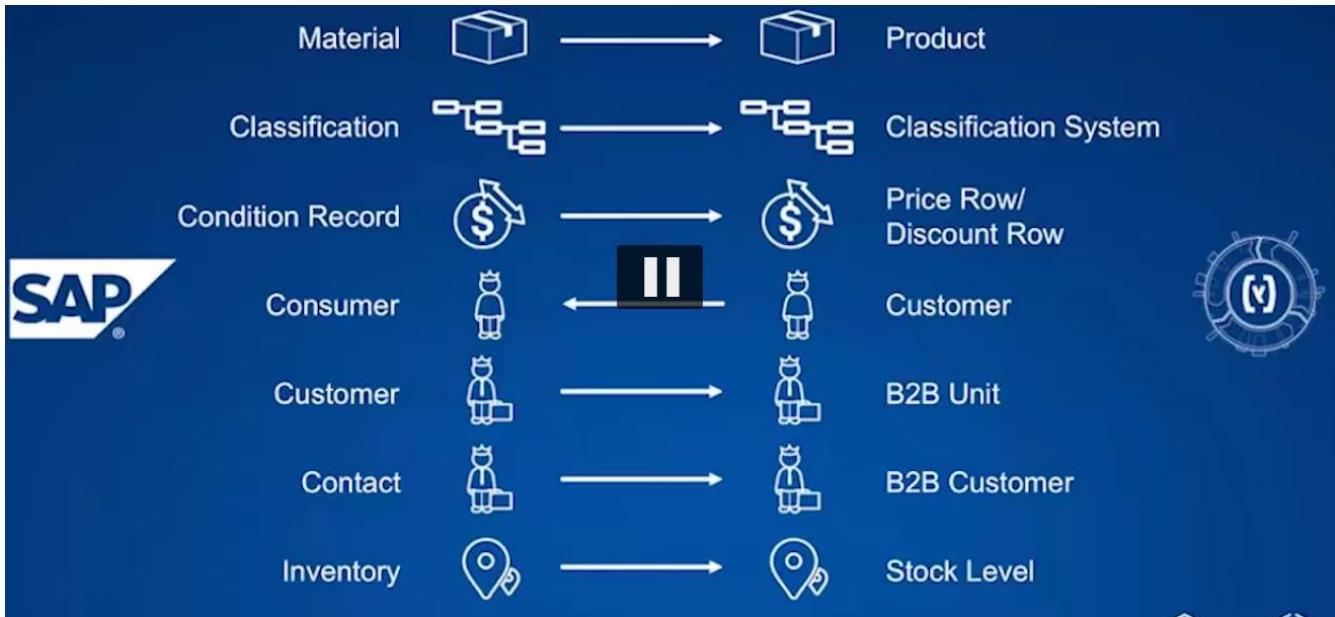
Also, **Async** integration – We can do in N number of ways.

Generally companies will do the **Hybrid** approach.

Hybrid approach =

**Sync** Integration for some scenario.

**Async** Integration for some scenario.



**Q = Explain typical commerce application architecture Layers?**

1. Front-end (i.e. the customer view or presentation layer)
2. Commerce API layer
3. ERP
4. External single-purpose applications

**Note** = Integration between the commerce layer and external, single-purpose applications allows commerce layer to easily display product, inventory, pricing, customer, and order data.

There are **2 approaches** for this (asynchronous and synchronous integration).

**Q = How to Determine which method to use for specific data points is an important decision.**

Data Point	Synchronous	Asynchronous
Product Data		Yes
Customer Data		Yes
Pricing Data	Yes	Yes
Inventory (Stock)	Yes	Yes
Order Status Data	Yes	Yes
Order Submission	Yes	Yes

**Note:** - So decision must of strategic & agreed to across all affected parts of the organization.

Name	Source	Target	Module	Type	Name	Source	Target	Module	Type
Create Order	HYBRIS	SAP	SD	ASync SOAP	Customer Data Feed	SAP	HYBRIS	FI	File
Modify Order	HYBRIS	SAP	SD	ASync SOAP	Product Data Feed	SAP	HYBRIS	PP	File
Cancel Order	HYBRIS	SAP	SD	ASync SOAP	Price Data Feed	SAP	HYBRIS	SD	File
Return Order	HYBRIS	SAP	SD	ASync SOAP	Promotions Data Feed	SAP	HYBRIS	FI	File
Order Enquiry	HYBRIS	SAP	SD	Sync SOAP	Inventory Data Feed	SAP	HYBRIS	MM	File
Order List Enquiry	HYBRIS	SAP	SD	Sync SOAP	Price/ Promotion lookup	SAP	HYBRIS	SD	Sync SOAP
Delivery Options Enquiry	HYBRIS	SAP	LE	Sync SOAP	Fraud Check	HYBRIS	3rd Party	NA	Sync SOAP
Reserve Inventory	HYBRIS	SAP	MM	ASync SOAP	Payment Capture/ Refund	HYBRIS	3rd Party	NA	Sync SOAP
Release Inventory	HYBRIS	SAP	MM	ASync SOAP	Payment Authorization	HYBRIS	3rd Party	NA	URL Redirection
Member Registration	HYBRIS	SAP	FI	ASync SOAP	Web Analytics Integration	HYBRIS	3rd Party	NA	File
Member Update	HYBRIS	SAP	FI	ASync SOAP	Social Media Integration	HYBRIS	3rd Party	NA	File
Member Validation	HYBRIS	SAP	FI	ASync SOAP	Recommendations	HYBRIS	3rd Party	NA	File
Loyalty Registration	HYBRIS	SAP	SD	Sync SOAP	Reviews & Ratings	HYBRIS	3rd Party	NA	File
Points Enquiry	HYBRIS	SAP	SD	Sync SOAP	Address Validation	HYBRIS	3rd Party	NA	Sync SOAP
Redemption	HYBRIS	SAP	FI	Sync SOAP	Tax Calculation	HYBRIS	3rd Party	NA	Sync SOAP
Order Status Update	SAP	HYBRIS	SD	ASync SOAP					

**Note** = To simplify integration between SAP & “SAP Comm”, “SAP Comm” has built connectors, allows data to flow from SAP to “SAP Comm” through Data Hub. Data Hub leverages SAP standards IDOC format to transfer data from SAP to “SAP Comm”.

SAP IDOC	SAP COMM Item Type	Description
MATMAS	Product	Product Data
LOISTD	Stock Level	Stock / Inventory Information
CLSMAS	Category & Product	Classification Hierarchy
CLFMAS	Feature, Feature Value, Feature Assignment, Category	Classification Data
DEBMAS	Customer & B2BUnit	Customer Data
ADRMAS	B2BUnit & Address	Customer Address Data
ADR3MAS	B2BUnit & Address	Customer Address Data
COND_A04	Price Row & Discount Row	Customer-Specific Pricing (Price Condition)

## Q: Explain Pricing Data? =

1 major difference between B2B & B2C is Price Data.

- ✓ In **B2C world**, base price is same for all customers. But in B2B world, customers have negotiated price data based on different business factors. Price conditions are typically written & developed in ERP & need to present to customer through online / offline. Hence,
- ✓ In B2C, it's efficient to load prices into “SAP Comm” commerce platform (Bcoz Promotion management is stronger & flexible than ERP). In B2B, lot of time, money already spent for customizing ERP to make prices handled (custom pricing conditions).
- ✓ Now determining whether to use **Sync (or) Async** is based on: -
  - ✓ **How often price change?** → If Org prices does not change often then go with Async Integration. It reduces load on ERP. Increase page load speed. Impacts customer experience.

- ✓ **How difficult to export price from SAP to SAP Comm?** → If Org has complex pricing condition & can't easily export from SAP then go with Sync Integration.

**Q = Explain Order Status Updates?**

- ✓ After order is placed, getting details of order is most common customer need.
- ✓ In B2C, customers are often obtaining tracking (Delivery Date).
- ✓ In B2B, customers also want additional information like Product allocated for shipment / Hold / ...
- ✓ Product move through various Plants & Warehouses, they all have internal statuses saved in ERP / WMS.
- ✓ It's technology decision to determine best way to deliver status information from BackEnd to FrontEnd.

**There are 2 options: -**

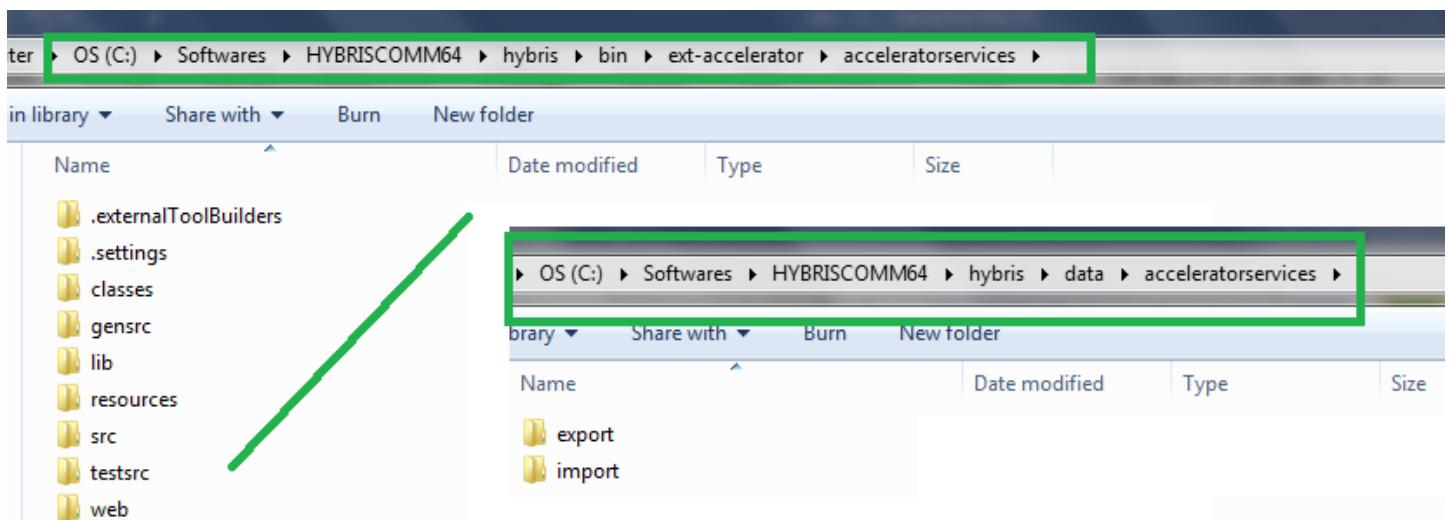
- o Real – Time Web Service Call from “SAP Comm” to SAP
  - o Batch data – load with replication of data from SAP - “SAP Comm”.
- ✓ The decision point (order status change) need to be notified (Push either email / mobile) to customer. If push notification is required then it is best to replicate data.

If push notification is not required then real – time web service should be used.

## Explain Hot Folder

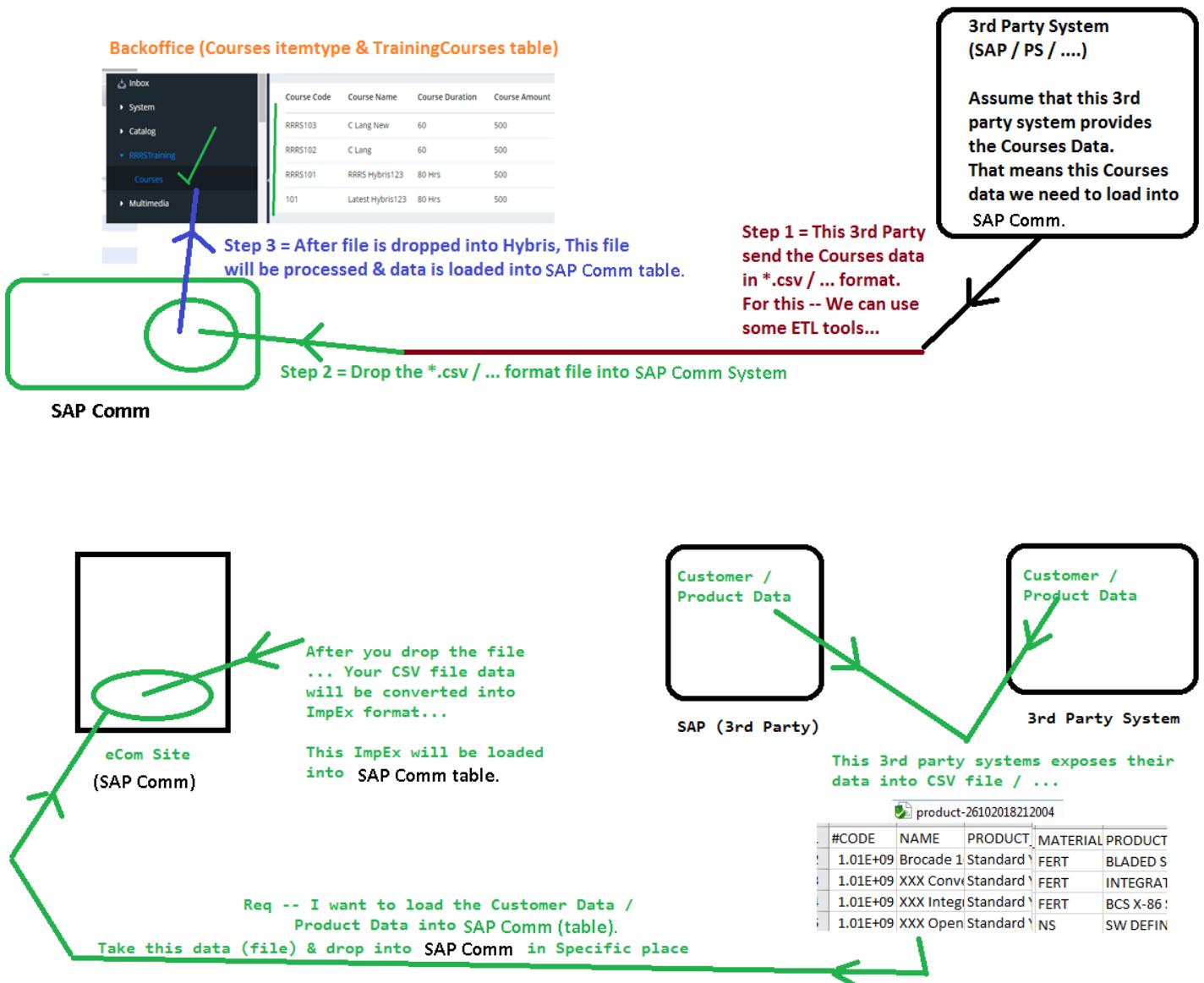
You have already seen how you can use ImpEx files to **import data** into the system. “SAP Comm” supports Hot Folders, which are folders from which data can be automatically imported into the platform by simply placing the data inside of the folder.

“SAP Comm” **acceleratorServices** extension template comes with a batch package that enables automated importing of data from hot folders.



The infrastructure enables using simple CSV files (internally translated into “SAP Comm” ImpEx scripts) to import content directly into the product catalogs.

This infrastructure uses **Spring Integration FWK**. Spring Integration provides a pluggable, highly configurable service-based design that can be extended as required.



**Note:** - Assume that, your 3rd party system giving below Courses.csv file.

A	B	C	D
1	1001;Hybris;80 Hrs;500		
2	1002;Data Hub;50 Hrs;500		

TESTITEMPL  
TITLES  
TITLESLP  
**RRRS TRAININGCOURSES**  
TRIGGERSCJ  
TYPESYSTEMPROPS  
UNIT2ACCTMGRGRPREL  
UNIT2APPROVERGRPREL  
UNIT2APPROVERSREL  
UNITS  
UNITSLP  
USERAUDIT  
USERGROUPPROPS  
USERGROUPS  
USERGROUPSLP  
USERPROFILES  
USERROLES

	P_CODE	P_NAME	P_DURATION
	HY410	Hybris	80
	DH100	Data Hub	55 Hrs
	C101	C Lang	60 Hrs
	DS100	Data Structure	60 Hrs

Import this \*.CSV file data which is coming from 3rd party system into Hybris (Courses / TrainingCourses)

courses-001 - Excel

1	1011;Hybris;80 Hrs;500
2	1012;Data Hub;50 Hrs;500
3	1013;C Lang;60 Hrs;500

**Step 1 = Create New Item Type called “ChennaRRRCourses” & Table Name = “RRRSTrainingCourses”**

Photon - rrrstrainingcore/resources/rrrstrainingcore-items.xml - Eclipse IDE

File Edit Navigate Search Project Run SAP Hybris [y] Hybris Design Window Help

Project Explorer ChennaRRRCoursesController.java rrrstrainingcore-items.xml

```

41
42
43    <itemtype code="ChennaRRRCourses" generate="true" autoreate="true" >
44        <deployment table="RRRSTrainingCourses" typecode="10128" />
45        <attributes>
46            <attribute qualifier="code" type="java.Lang.String">
47                <modifiers optional="false" unique="true" />
48                <persistence type="property"/>
49            </attribute>
50            <attribute qualifier="name" type="java.Lang.String">
51                <modifiers optional="false" />
52                <persistence type="property"/>
53            </attribute>
54            <attribute qualifier="duration" type="java.Lang.String">
55                <modifiers optional="false" />
56                <persistence type="property"/>
57            </attribute>
58            <attribute qualifier="amount" type="java.Lang.Integer">
59                <modifiers optional="false" />
60                <persistence type="property"/>
61            </attribute>
62        </attributes>
63    </itemtype>

```

productcockpit  
rrrstrainingcockpits  
rrrstrainingcore  
JRE System Library [jre1.8.0\_191]  
Referenced Libraries  
resources  
cockpitng.widgets.actions.pickupconfig  
localization  
rrrstrainingcore  
rrrstrainingcore.integration  
rrrstrainingcore.messages  
rrrstrainingcore.processes  
rrrstrainingcore.processes.quote  
rrrstrainingcore.test  
test  
rrrstrainingcore-backoffice-config.xml  
rrrstrainingcore-items.xml  
rrrstrainingcore-spring.xml  
rrrstrainingcore.build.number

**Step 2 = Enable Standard hot folder for “rrrstrainingstorefront” & load the data into “RRRSTrainingCourses” table.**

Copy “hot-folder-store-electronics-spring.xml” & Paste with different name “hot-folder-store-rrrs-spring.xml”

Project Explorer

- productcockpit
- rrstrainingcockpits
- rrstrainingcore** ✓
- JRE System Library [jre1.8.0\_191]
- Referenced Libraries
- src
- testsrc
- gensrc
- classes
- lib
- resources
  - cockpitng
  - localization
  - rrstrainingcore
    - import
    - integration
      - hot-folder-common-spring.xml
      - hot-folder-store-apparel-spring.xml
      - hot-folder-store-electronics-spring.xml
      - hot-folder-store-rrrs-spring.xml**
  - messages
  - processes

hot-folder-store-rrrs-spring.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!--
3 [y] hybris Platform
4
5 Copyright (c) 2017 SAP SE or an SAP affiliate company. All rights reserved.
6
7 This software is the confidential and proprietary information of SAP
8 ("Confidential Information"). You shall not disclose such Confidential
9 Information and shall use it only in accordance with the terms of the
10 license agreement you entered into with SAP.
11 -->
12
13 <beans xmlns="http://www.springframework.org/schema/beans"
14   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
15   xmlns:int="http://www.springframework.org/schema/integration"
16   xmlns:file="http://www.springframework.org/schema/integration/file"
17   xmlns:context="http://www.springframework.org/schema/context"
18   xsi:schemaLocation="http://www.springframework.org/schema/beans
19     http://www.springframework.org/schema/beans/spring-beans.xsd
20     http://www.springframework.org/schema/integration
21     http://www.springframework.org/schema/integration/spring-integration.xsd
22     http://www.springframework.org/schema/integration/file
23     http://www.springframework.org/schema/integration/file/spring-integration-file.xsd
24     http://www.springframework.org/schema/context
25     http://www.springframework.org/schema/context/spring-context.xsd">

```

**Step 3 = Open the file “hot-folder-store-rrrs-spring.xml” file & do below: -**

- 1) Replace all occurrences of **electronics** with **rrrs**
- 2) Replace all occurrences of **Electronics** with **Rrrs**

hot-folder-store-rrrs-spring.xml

hot-folder-store-electronics-spring.xml

```

1/   xmlns:context="http://www.springframework.org/schema/context"
2   xsi:schemaLocation="http://www.springframework.org/schema/beans
3     http://www.springframework.org/schema/beans/spring-beans.xsd
4     http://www.springframework.org/schema/integration
5     http://www.springframework.org/schema/integration/spring-integration.xsd
6     http://www.springframework.org/schema/integration/file
7     http://www.springframework.org/schema/integration/file/spring-integration-file.xsd
8     http://www.springframework.org/schema/context
9     http://www.springframework.org/schema/context/spring-context.xsd">
10
11 <context:annotation-config/>
12
13 <bean id="baseDirectoryRrrs" class="java.lang.String">
14   <constructor-arg value="#{baseDirectory}/#${tenantId}/rrrs" />
15 </bean>
16 <!-- 1) Scan for files -->
17 <file:inbound-channel-adapter id="batchFilesRrrs" directory="#{baseDirectoryRrrs}" -->
18   <!-- filename-regex="^(.*)(\d+)\.csv" comparator="fileOrderComparator" -->
19   <int:poller fixed-rate="1000" />
20 </file:inbound-channel-adapter>
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
-->

```

**Q: What is there in “hot-folder-store-rrrs-spring.xml”?**

**1) File Dropping Loc**

```
<bean id="baseDirectoryRrrs" class="java.lang.String">
    <constructor-arg value="#{baseDirectory}/#${tenantId}/rrrs" />
</bean>
```

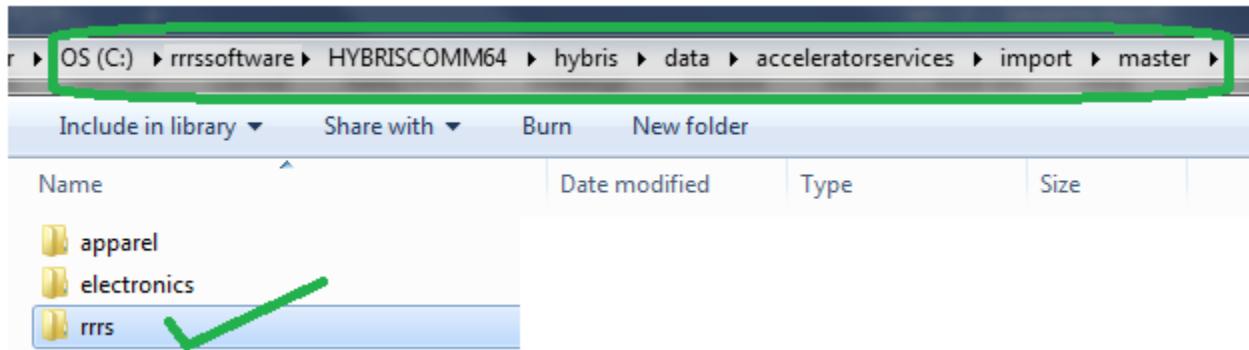
**2) Inbound adapter = This scan for the matching files**

```
<file:inbound-channel-adapter id="batchFilesRrrs" directory="#{baseDirectoryRrrs}"
    filename-regex="^(.*)(\d+)\.csv" comparator="fileOrderComparator"
    <int:poller fixed-rate="1000" />
</file:inbound-channel-adapter>
```

**3) Outbound adapter = This takes the file & move for processing.**

```
<file:outbound-gateway request-channel="batchFilesRrrs" reply-channel="batchFilesRrrsProc"
    directory="#{baseDirectoryRrrs}/processing" delete-source-files="true" />
```

**Step 4 = Create “rrrs” folder.**



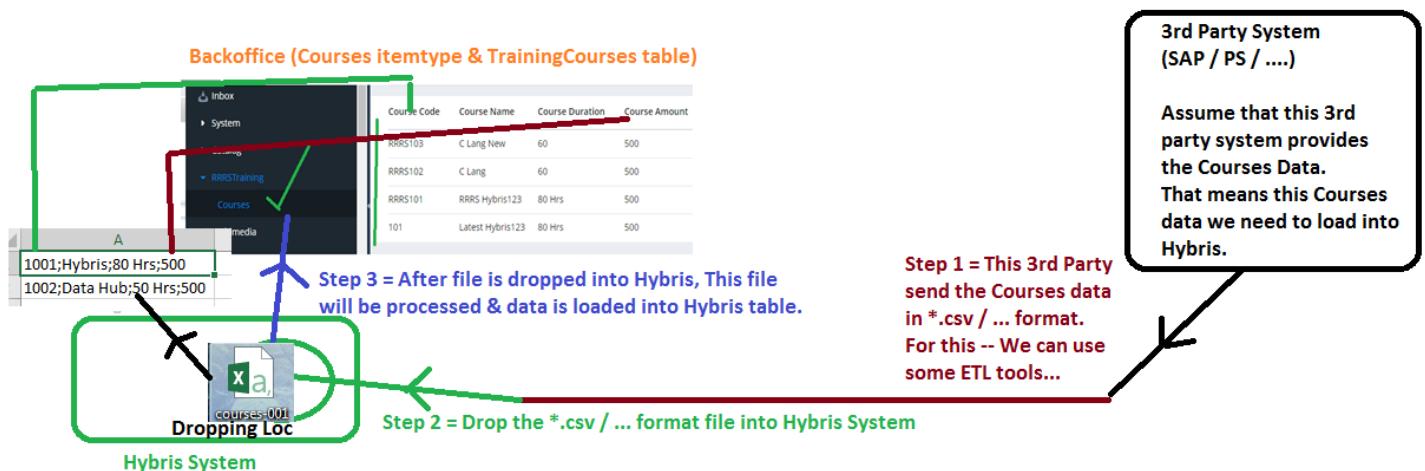
## Step 5 = Enable Spring Integration by adding “hot-folder-store-rrrs-spring.xml” file in “rrrstrainingcore-spring.xml”

```

1/      xsi:schemaLocation=" http://www.springframework.org/schema/beans
18      http://www.springframework.org/schema/beans/spring-beans.xsd
19      http://www.springframework.org/schema/context
20      http://www.springframework.org/schema/context/spring-context.xsd http://www.springframework.o
21
22      <context:annotation-config>
23
24      <!-- Spring Integration -->
25      <import resource="classpath:/rrrstrainingcore/integration/hot-folder-store-electronics-spring.xml"/>
26      <import resource="classpath:/rrrstrainingcore/integration/hot-folder-store-apparel-spring.xml"/>
27      <import resource="classpath:/rrrstrainingcore/integration/hot-folder-common-spring.xml"/>
28      <import resource="classpath:/rrrstrainingcore/integration/hot-folder-store-rrrs-spring.xml"/>
29      <!-- SystemSetup for the RrrstrainingCore -->
30      <bean id="acceleratorCoreSystemSetup" class="com.rrrs.rrrstraining.core.setup.CoreSystemSetup"
31          parent="abstractCoreSystemSetup"/>
32
33      <!-- Solr field value providers -->
34
35      <bean id="volumeAwareProductPriceValueProvider" class="com.rrrs.rrrstraining.core.search.solrfacetse
36          <property name="fieldNameProvider" ref="solrFieldNameProvider"/>
37          <property name="priceService" ref="priceService"/>
38          <property name="commonI18NService" ref="commonI18NService"/>
39          <property name="sessionService" ref="sessionService"/>
40          <property name="userService" ref="userService"/>

```

## Step 6 = Create the mapping definition for ChennaiRRRCourses itemtype.



You dropped \*.csv file (That means, Data is in CSV format).

“SAP Comm” knows only ImpEx.

So – It’s time to convert \*.csv data into equivalent ImpEx.

---

Contact Us = ChennaReddyTraining@RRRS.CO.IN

## Do the Mapping like: -

**Col A** data should go to “Course Code”

**Col B** data should go to “Course Name”

**Col C** data should go to “Course Duration”

**Col D** data should go to “Course Amount”.

```
<bean id="batchCustomerConverterMapping"
      class="de.hybris.platform.acceleratorservices.dataimport.batch.converter.impl.DefaultConverterMapping"
      p:mapping="customer"
      p:converter-ref="batchCustomerConverter"/>

<bean id="batchCoursesConverterMapping"
      class="de.hybris.platform.acceleratorservices.dataimport.batch.converter.impl.DefaultConverterMapping"
      p:mapping="courses"
      p:converter-ref="batchCoursesConverter"/>

<bean id="batchCoursesConverter" class="de.hybris.platform.acceleratorservices.dataimport.batch.converter.impl.DefaultImpexConverter">
    <property name="header">
        <value>#{{defaultImpexProductHeader}}
        # Insert Courses
        INSERT_UPDATE Courses;code[unique=true];name;duration;amount
        </value>
    </property>
    <property name="impexFooter">
        <value>;{+0};{1};{2};{3};</value>
    </property>

```

**Note:** - {+0} = Here + represents required field.

	Info	Content	Row Count	Columns	Primary Key	Exported Keys	Imported Keys	Indexes	Privileges	Column Privileges	Row IDs	Versions
TESTITEMPL												
TITLES												
TITLESLP												
RRRS TRAININGCOURSES												
TRIGGERSJC												
TYPESYSTEMPROPS												
UNIT2ACCTMGRGRPREL												
UNIT2APPROVERGRPREL												
UNIT2APPROVERSREL												
UNITS												
UNITSLP												
USERAUDIT												
USERGROUPPROPS												
USERGROUPS												
USERGROUPSLP												
USERPROFILES												
USERPROPS												
USERRIGHTS .....												

Now -- We need to do the Mapping.  
Means: -  
\*CSV File 1st Col map with Table Code Col  
\*CSV File 2nd Col map with Table Name Col  
\*CSV File 3rd Col map with Table Dura Col  
=====

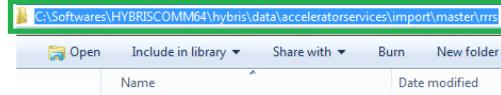
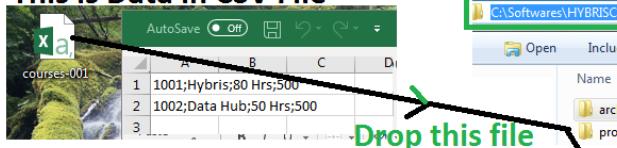
## Step 6 = Do the build (ant clean all)

## Step 7 = Start the Server (hybrisserver.bat)

## Step 8 = Test the Results

The screenshot shows the Hybris backoffice interface. On the left, there's a sidebar with navigation links like System, Types, Search and Navigation, Backoffice, and Indexed Types. The main area has a search bar with 'Courses' and a 'SEARCH' button. Below it, a table titled 'Data Before' lists courses with columns: Course Code, Course Name, Course Duration, and Course Amount. The data includes RRRS103 (C Lang New, 60, 500), RRRS102 (C Lang, 60, 500), and RRRS101 (RRRS Hybris123, 80 Hrs, 500). To the right, a table titled 'Results (Output)' shows the same columns but with different data: 1002 (Data Hub, 50 Hrs, 500) and 1001 (Hybris, 80 Hrs, 500). Both tables have green checkmarks next to their respective course codes.

This is Data in CSV File



Results (Output)

A Windows file explorer window shows the directory structure. The path is Local Disk (E) > rrsssoftware > hybris > data > acceleratorservices > import > master > rrrs > processing. An arrow points up to the 'processing' folder. Another arrow points down to the 'archive' folder, which contains a file named 'ChennaRRRCourses-001.csv\_20210608034...'.

A command prompt window titled 'cmd.exe - hybrisserver.bat' shows the output of an import job. The log includes messages like 'INFO [task-scheduler-8] [DefaultImportService] Starting import synchronous using cronjob with code=000006Y0', 'INFO [task-scheduler-8] [ImpExImportJob] Starting ImpEx cronjob "ImpEx-Import"', 'INFO [task-scheduler-8] [Importer] Finished 1 pass in 0d 00h:00m:00s - processed: 2, no lines dumped <last pass 0>', 'INFO [task-scheduler-8] [DefaultImportService] Import was successful (using cronjob with code=000006Y0)', 'INFO [task-scheduler-8] [TimeMeasurementAspect] Processed AbstractImpexRunnerTask\$EnhancerBySpringCGLIB\$5a33868f8.execute [header=de.hybris.acceleratorservices.dataimport.batch.BatchHeader@73a50d52[file=C:\Softwares\HYBRISCOMM64\hybris\data\acceleratorservices\import\master\g\courses-001.csv,catalog=rssProductCatalog,language=en,net=false]] in 6898ms', and 'INFO [task-scheduler-8] [TenantActivationAspect] Setting tenant <> on the current thread Thread[task-scheduler-8,5,main]'. The log ends with 'INFO [task-scheduler-8] [TimeMeasurementAspect] Processed CleanupTask.execute [header=de.hybris.platform.acceleratorservices.dataimportader@73a50d52[file=C:\Softwares\HYBRISCOMM64\hybris\data\acceleratorservices\import\master\rrrs\processing\courses-001.csv,catalog=rssP,language=en,net=false]] in 22ms'.

## “SAP Comm” / Commerce Security & Roles Configuration

You're Administrator [logout](#)

Assume that, this is Prod hAC.  
Do you think that all people will have all  
the tabs in Prod? = No

Uptime: 1 hour 2 minutes. Enjoy!

Show last: 5 minutes

Memory overview

CPU Load (4 processors)

Top links

1. [Home](#)
2. [Initialization](#)
3. [Cache](#)
4. [Scripting Languages](#)

[Clear history](#)

So – It’s time to control the Tabs & Also Options within the tab.

Platform Monitoring Maintenance **Console**

Scripting Languages FlexibleSearch ImpEx Import **ImpEx Export** LDAP

Uptime: 1 hour 4 minutes. Enjoy!

Venkat will have only Console -- Flexible

Chithu will have only ImpEx

Memory overview

CPU Load (4 processors)

Show last: 5 minutes

“SAP Comm” Security → based on Spring Security.

**Example:** - Create **1 User** & that user should have only “Console tab” & 2 Options (Scripting Language & Flexible Search).

Backoffice / hMC – Users – Employee –

ID = testuser (or) chenna

## Step 1 = Let's create user with admin group.

The screenshot shows the hybris administration console interface. On the left, there is a sidebar with various modules like User, Companies, User Groups, Employees, Customers, Addresses, Titles, Agreements, Order, Price Settings, Internationalization, Marketing, Cockpit, Scripting, WCMS, and Ticket System. The 'Employees' module is currently selected. The main panel shows a form for creating a new user. The 'ID' field is set to 'testuser'. In the 'Groups' section, 'admingroup' and 'employeegroup' are listed under the 'Groups' table. A green box highlights the 'Groups' table and the 'admingroup' entry.

Q: How to give only “Console – FlexibleSearch” for created user (testuser).

The screenshot shows the Eclipse IDE with the 'Project Explorer' view on the left and the 'spring-security-config.xml' file open in the editor on the right. The code defines various intercept URLs for monitoring and flexible search operations. A specific line of code is highlighted with a blue background, which contains the access rule 'ROLE\_HAC\_CONSOLE\_FLEXIBLESEARCH'. This indicates that the 'testuser' user, who has been created with the 'admingroup' and 'employeegroup' roles, will have access to the 'FlexibleSearch' feature through the 'Console'.

```

<intercept-url pattern="/console/flexiblesearch/**" access="ROLE_ADMINGROUP, ROLE_HAC_CONSOLE_FLEXIBLESEARCH" require="https://localhost:9002"/>

```

The screenshot shows the hybris administration console interface. On the left, there is a sidebar with a tree view of the application structure:

- Multimedia
- User
  - Companies
  - User Groups
  - Employees** (highlighted with a green arrow)
  - Customers
  - Addresses
  - Titles
  - Agreements
- Order
- Price Settings
- Internationalization
- Marketing
- Cockpit
- Scripting
- WCMS
- Ticket System

The main content area has a blue header bar with buttons for Save, Reload, Copy, and Delete. Below the header, there are tabs for General, Addresses, Password, Orders, Employee Prices, Personalization, Access Rights, and Com. The General tab is selected.

The General tab displays the following information for the user 'testuser':

ID:	testuser	Name:	testuser
-----	----------	-------	----------

Below this, there are sections for Properties, Description, Standard Language, Standard Currency, and Groups. The Groups section contains a table:

ID	Name
hac_console_flexiblesearch	n/a
employeegroup	n/a

At the bottom of the main content area, there is a banner with the text '(v) hybris administration console', the message 'You're testuser', and a 'logout' button.

**Q: How to provide log access (or) download log to created user (testuser).**

The screenshot shows an IDE interface with the 'Project Explorer' and 'spring-security-config.xml' file open. The 'spring-security-config.xml' file contains the following configuration:

```

<!-- Session management session authentication strategy -->
<!--> <!-->
80 intercept-url pattern="/Login.jsp" access="PERMIT_ALL" requires-channel="https"/>
81 intercept-url pattern="/tenants/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_TENANTS" requires-channel="https"/>
82 intercept-url pattern="/platform/config/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CONFIGURATION" requi
83 intercept-url pattern="/platform/config/valuechanged/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CONFIGURATION" r
84 intercept-url pattern="/platform/config/*/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CONFIGURATION" requires-ch
85 intercept-url pattern="/platform/config/*/*" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CONFIGURATION, ROLE_HAC_PLAT
86 intercept-url pattern="/platform/system/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_SYSTEM" requires-channel="ht
87 intercept-url pattern="/platform/log4j/changeLevel/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_LOGGING" requires-
88 intercept-url pattern="/platform/log4j/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_LOGGING, ROLE_HAC_PLATFORM_LOG
89 intercept-url pattern="/platform/extensions/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_EXTENSIONS" requires-char
90 intercept-url pattern="/platform/init/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_INITIALIZATION, HYBRIS_NOT_INITI
91 intercept-url pattern="/platform/update/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_UPDATE" requires-channel="htt
92 intercept-url pattern="/platform/dryrun/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_SQLSCRIPTS, HYBRIS_NOT_INITIA
93 intercept-url pattern="/platform/jars/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CLASSPATHANALYZER" requires-ch
94 intercept-url pattern="/platform/jars_howto/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_CLASSPATHANALYZER" requir
95 intercept-url pattern="/platform/license/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_LICENSE" requires-channel="ht
96 intercept-url pattern="/platform/support/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_SUPPORT" requires-channel="ht
97 intercept-url pattern="/platform/pkanalyzer/**" access="ROLE_ADMINGROUP, ROLE_HAC_PLATFORM_PKANALYZER" requires-char

```

**User Groups**

ID	Name
hac_console_flexiblesearch	n/a
employeegroup	n/a
hac_platform_support	n/a

The ZIP file includes the system definition (item), the properties files and installed extensions. A you can add the console access logfiles or other the log folder to this zip.

**Note:** - In case B2B, we will be having different types of customers.

1) Customer – Just can see the prices

2) Customer – They can see prices & Add the items to Cart

3) Customer – They can see prices, Add the items to Cart & Place Order.

**Requirement** = If customer having Place order role then only enable “Submit Order” button?.

```
<a id="call_place_order_button" data-href="#"> ${XXX} data-orderwaitingheader=""  
 class="btn btn-primary btn-block review-btn font-size-normal"  
 <sec:authorize access="!hasAnyRole('ROLE_CREATEORDER')">disabled</sec:authorize>">  
 <xss:theme code="Submit Order" />  
</a>
```

Roles in Backoffice are used to specific instance of a widget / node.

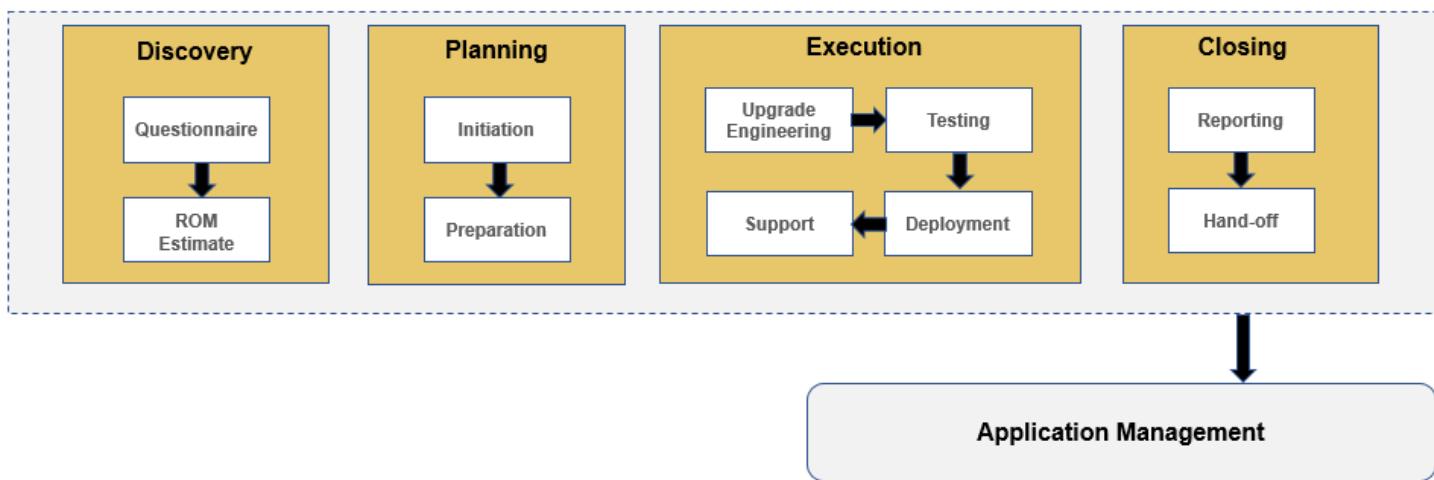
Backoffice node structure is defined in **xxx-backoffice-config.xml** file.

**Example:** - promotionsbackoffice-backoffice-config.xml, voucherbackoffice-backoffice-config.xml, cockpit-backoffice-config.xml, platformbackoffice-backoffice-config.xml etc.

```
79      </context>  
80      <context merge-by="module" type="Product" component="editor-area">  
81          <editorArea:editorArea xmlns:editorArea="http://www.hybris.com/cockpitng/component/editorArea">  
82              <editorArea:tab name="hmc.tab.product.properties">  
83                  <editorArea:section name="hmc.product.descriptions">  
84                      <editorArea:attribute xmlns="http://www.hybris.com/cockpitng/component/editorArea" quali  
85                      </editorArea:section>  
86              </editorArea:tab>  
87              <editorArea:tab name="hmc.tab.product.multimedia">  
88                  <editorArea:section name="hmc.section.product.additionalmedias">  
89                      <editorArea:attribute xmlns="http://www.hybris.com/cockpitng/component/editorArea" quali  
90                      </editorArea:section>  
91      </editorArea:tab>  
92      <editorArea:tab name="hmc.tab.product.stock" position="35">  
93          <editorArea:section name="hmc.tab.product.stockfinder"/>  
94          <editorArea:section name="hmc.section.warehouse.stocklevels">  
95              <editorArea:attribute xmlns="http://www.hybris.com/cockpitng/component/editorArea"  
96                  editor="de.hybris.platform.commerceservices.backoffice.  
97                  qualifier="stockLevels" label="hmc.text.product.usesea  
98                  <editorArea:editor-parameter>  
99                      <editorArea:name>stockLevelSearchField</editorArea:name>  
100                     <editorArea:value>product</editorArea:value>  
101                 </editorArea:editor-parameter>  
102             </editorArea:attribute>
```

## Scenario = Upgrade [Example “1811 ---> 2011”]?

Software Name	1811	2011
Oracle JRE / JDK	Java 8	Java 11
SAP JVM	8.1	SapMachine 11
Apache Tomcat	8.5.57	8.5.56, bundled
Database	HANA 1.00 SPS12, HANA 2.0	HANA 1.00 SPS12, HANA 2.0
Solr	7.7	8.6
Browsers	IE 10, 11, MS Edge & Other latest	Microsoft Edge
Commerce Directory Structure	bin [platform, <b>Exts</b> & custom]	bin [platform, <b>modules</b> & custom]
Recipes	b2c	cx
Spring [ <u>aop</u> , aspects, beans, context]	4.3.21	5.2.9
Spring Security [Config, Core]	4.2.6	5.2.3
Backoffice changes	ABC has good amount of changes	Cross check those with steps
Spring oAuth2	2.2.0	2.3.6
Site Responsive Impact	Yes	We need to verify



## Q = How to find “End of Mainstream Maintenance”?

help.sap.com/viewer/dc198ac31ba24dce96149c8480be955f/2011/en-US/1c6c687ad0ed4964bb43d409818d23a2.html

Release Version	General Availability	End of Mainstream Maintenance
2011	November 11, 2020	February 11, 2023
2005	May 13, 2020	August 13, 2022
1905	May 29, 2019	August 29, 2022
1811	December 12, 2018	August 29, 2021
1808	August 8, 2018	August 29, 2021
6.7	April 11, 2018	August 8, 2020

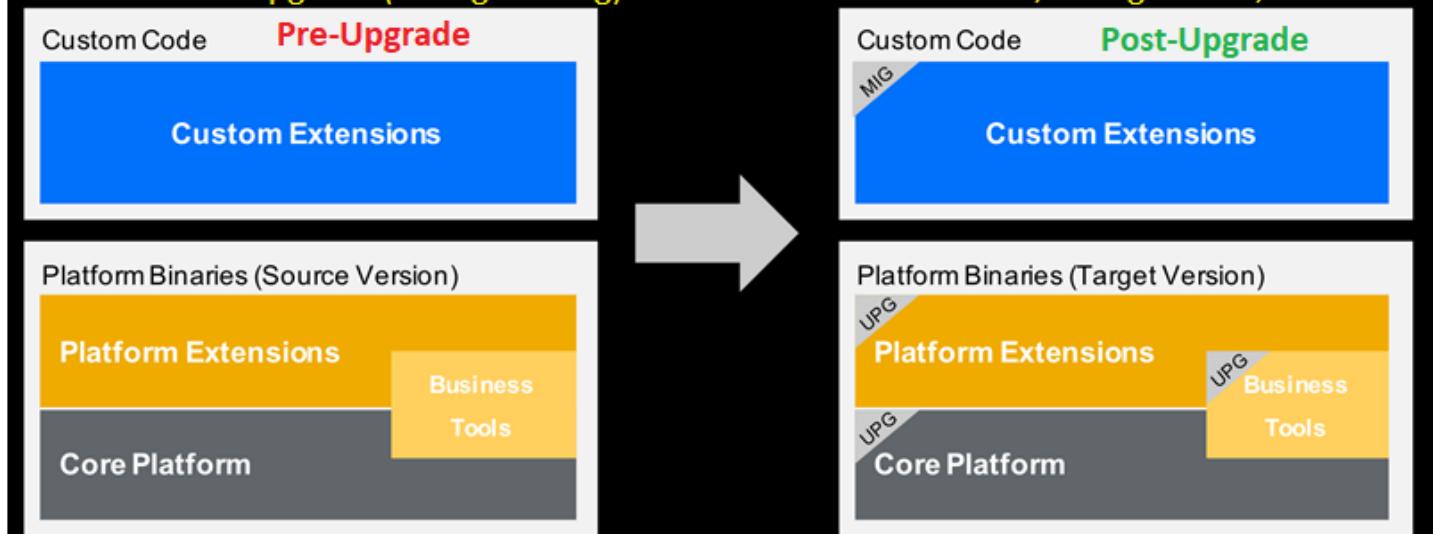
## Upgrade System Doc =

[https://www.sap.com/cxworks/article/441185299/upgrading\\_your\\_sap\\_commerce\\_solution](https://www.sap.com/cxworks/article/441185299/upgrading_your_sap_commerce_solution)



## Non-Accelerator Based Solution =

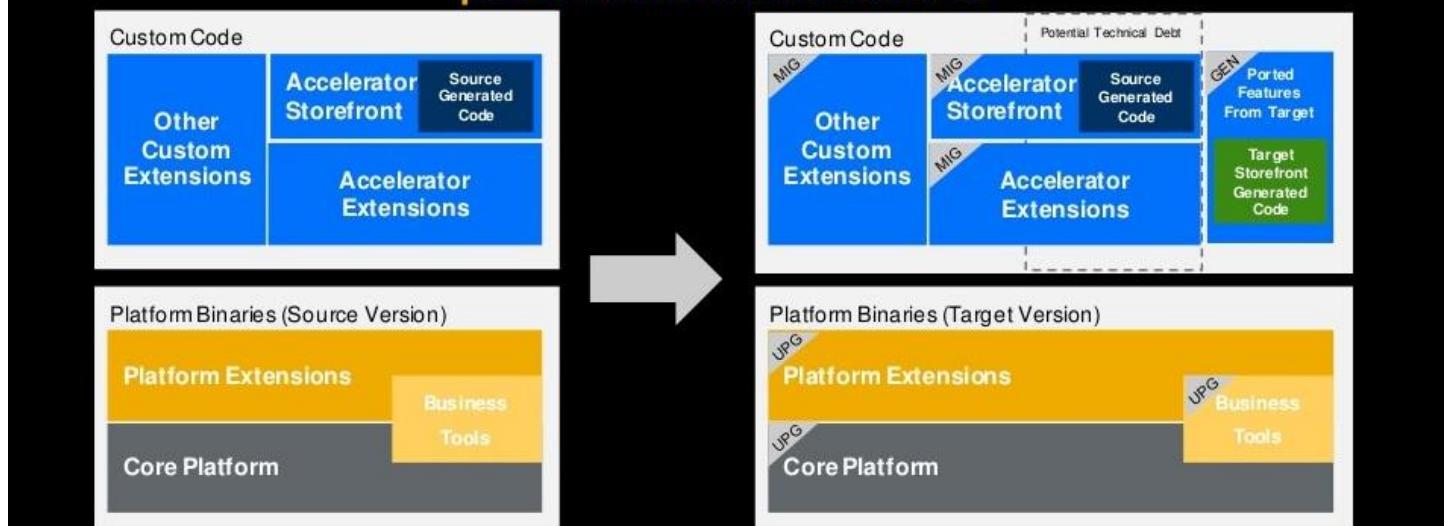
**Non-Accelerator Based Solution** = platform binaries being upgraded to the target version, with technical upgrade (re-engineering) of custom extensions code, configuration, and data.



	<b>Upgraded?</b>	<b>Technically Migrated?</b>	<b>Comments</b>
Core Platform	Yes	No	N/A
Platform extensions	Yes	No	N/A
Business Tools	Yes	No	N/A
Business Tools customizations and custom configuration	No	Yes	When customer build additional configuration (implementation) in hMC or Cockpits to provide custom business logic.
Custom extensions	No	Yes	Will probably require changes to your custom code to ensure it can compile, system can start, and <b>upgrade running system can run without errors.</b>

## Accelerator Based Solution =

**Same as last option but extra work to migrate accelerator configurations and port in some critical features**



## Q = Software architecture impacts?

- 1) Third party compatibility
- 2) Leverage Commerce APIs
- 3) Move to Spartacus

## Q = Upgrade plan estimates?

Resource Model 50/50 Onshore-Offshore	
Role	Month 1
Project Manager	0.25
Architect	0.25
Team / Technical Lead	1.00
Functional / Tester	1.00
Sr Comm Developer	1.00
Comm / UI Developer	1.00
Total	4.5

DRAFT

Role	Month 1	Month 2	Month 3	Month 4
Project Manager	0.25	0.25	0.25	0.25
Architect	0.25	0.25	0.25	0.25
Team / Technical Lead	1.00	1.00	1.00	1.00
Functional / Tester	1.00	1.00	1.00	1.00
Sr Comm Developer	1.00	1.00	1.00	1.00
Comm / UI Developer	1.00	1.00	1.00	1.00
Total	4.5	4.5	4.5	4.5

Assessment	
Role	Month 0
Project Manager	0.50
Architect	1.00
Team / Technical Lead	1.00
Functional / Tester	1.00
UI / Comm Developer	1.00
Total	4.5

DRAFT

Core Platform	Platform core functionality. This is what is found in the hybris/bin/platform directory
Platform extensions	Platform core extensions usually placed in hybris/bin/ext-(...) . These extensions provide extensive functionality and are built upon the core platform
Business Tools	The tools that are used for performing business functionality. These are comprised of SAP Administration Console, hMC, backoffice and cockpits
Custom extensions	Any extension that changes or extends behavior of the platform, usually placed in hybris/bin/custom
Platform Template	Standard templates that can be used to accelerate development. These include B2C/B2B accelerator, Telco etc.
Template Customizations	Any customization that extends/overwrites the behavior of the generated code from the platform template

## **Q = Explain Technical Approach Steps?**

- 1) Regression Testing [Unit, Integration, UI, Security, Performance]
- 2) Update Deployment Procedure [update the type system]
- 3) Business Data Technical Migration
- 4) Configuration Technical Migration
  - a. local.properties
  - b. localextensions.xml
  - c. buildcallback.xml
  - d. customize folder [hybris/config/customize]
  - e. Tomcat and TcServer Configuration
- 5) Data Model Changes
- 6) Custom Code Technical Migration
  - a. Java Code [hybris/bin/custom]. **There are several API changes:**
    - i. Method signature has changed
    - ii. Method name has changed
    - iii. Method was moved to another class
    - iv. Method is deprecated or was removed
    - v. Method visibility has changed
    - vi. Interface has additional method declarations
    - vii. Class was renamed
    - viii. Class location (package) has changed
    - ix. Class is deprecated or was removed
    - x. Functional module (set of interfaces, classes or methods) was re-engineered (refactored) between source & target.
  - b. Spring Context
  - c. Third Party Libraries
  - d. Integrations
  - e. Performance fixes
  - f. Security

## **Upgrade high level procedure (or) steps =**

### **Step 1 = Download, unzip, and build the new distribution**

- Download the SAP Comm Platform “YYMM [Example - 2011]” distribution and all the needed modules.
- There are two installation ZIP file
  - CXCOM**200500**P\_0-70004955.ZIP = Contains all the files to install the SAP Hybris Commerce and run it out-of-the-box
  - HYBRISCRMINT00P\_0-70005781.ZIP = This file contains the “SAP Comm” CRM Integration dependencies

You must download and extract the contents of the HYBRISCRMINT00P\_0-70005781.ZIP file into the same directory to which you extract the SAP Commerce CXCOM**200500**P\_0-70004955.ZIP file

- Extract all the downloaded ZIP files to a new directory, where the Target version system will be installed
- Run **ant clean / ant clean all** on the new system. This will create the config folder in the new instance of Hybris

**Step 2 =** Copy (and adjust if needed) your custom extensions ( `${HYBRIS_BIN_DIR}/custom`) from your **old** instance of SAP Hybris Commerce to the **new** instance of SAP Commerce

**Step 3 =** Copy and adjust the configuration files

- From your **old** Platform location copy your **local.properties** and **localextensions.xml** configuration files to the **new** config folder (**HYBRIS\_CONFIG\_DIR**)
- Copy any 3rd party database drivers if you using from **HYBRIS\_BIN\_DIR/platform/lib/dbdriver** folder to **new** instance location
- If necessary, edit the **localextensions.xml** file in order to add additional extensions (for example the ones which are new in the release)
- If necessary, adjust the local.properties file

**Step 4 = Copy data (HYBRIS\_DATA\_DIR/hsqldb folder)**

**Step 5 = Copy media files (media data files are located in HYBRIS\_DATA\_DIR/media)**

**Step 6 = Reinitialize Solr Facet Search indexes.**

**Step 7 = Build “SAP Comm” platform**

- Go to **HYBRIS\_BIN\_DIR/platform** directory & execute **setantenv.bat** on Windows or. **./setantenv.sh** on Linux.
- Execute **ant all & wait** for the build to complete
- Run ant **update system (or) “hAC – Platform – Update”**

**Step 8 = “ant clean all” – Fix If any challenges.**

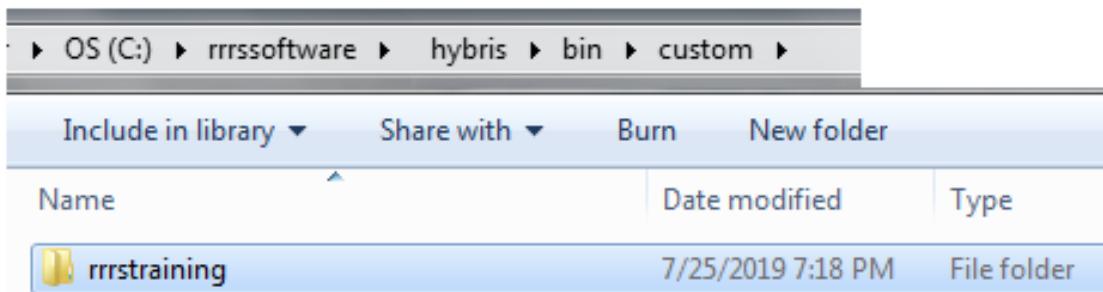
Start the server – Fix if any challenges.

Test the site – Fix if any challenges.

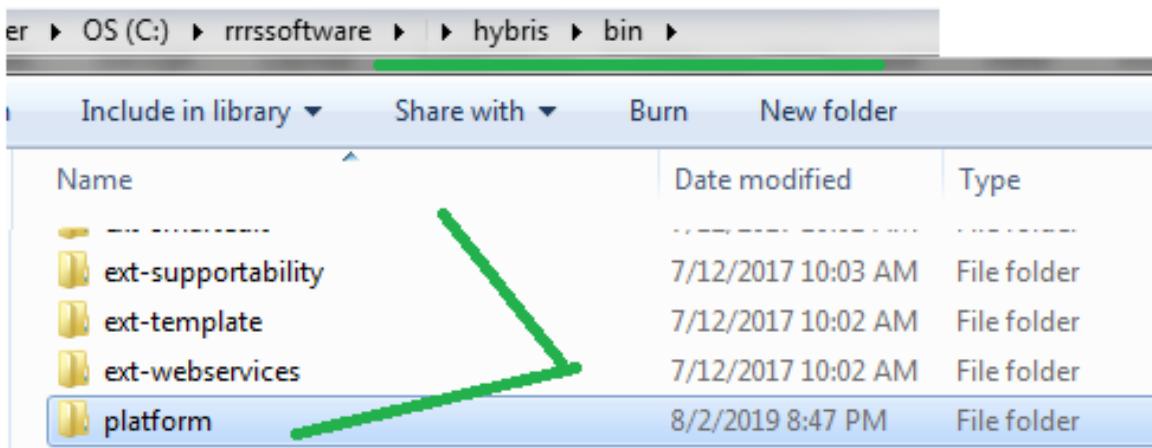
## Scenario = Ant Customize

In SAP Comm, we have 2 types of files & directories: -

1) Custom Files = These we change based on client requirements.



2) Read only files (or) Binary files = These generally we don't change directly.



**Note** = Sometimes, we may need to modify these binary files (or) xml files (or) standard files (or) CSS files (or) ....

==> To do all above... We can use **ant customize**

The screenshot shows a Windows desktop environment. At the top, there's a taskbar with icons for File Explorer, Start, Task View, and others. Below it is a file explorer window showing the path: OS (C:) > rrrsoftware > HYBRISCOMM64 > hybris > bin > platform >. On the left, a tree view lists various files and folders under 'Name'. The 'build' folder is selected. The main pane displays the contents of the build.xml file in Notepad++. A specific section of the XML code is highlighted with a green box and a red arrow pointing to it from the left. The code snippet is:

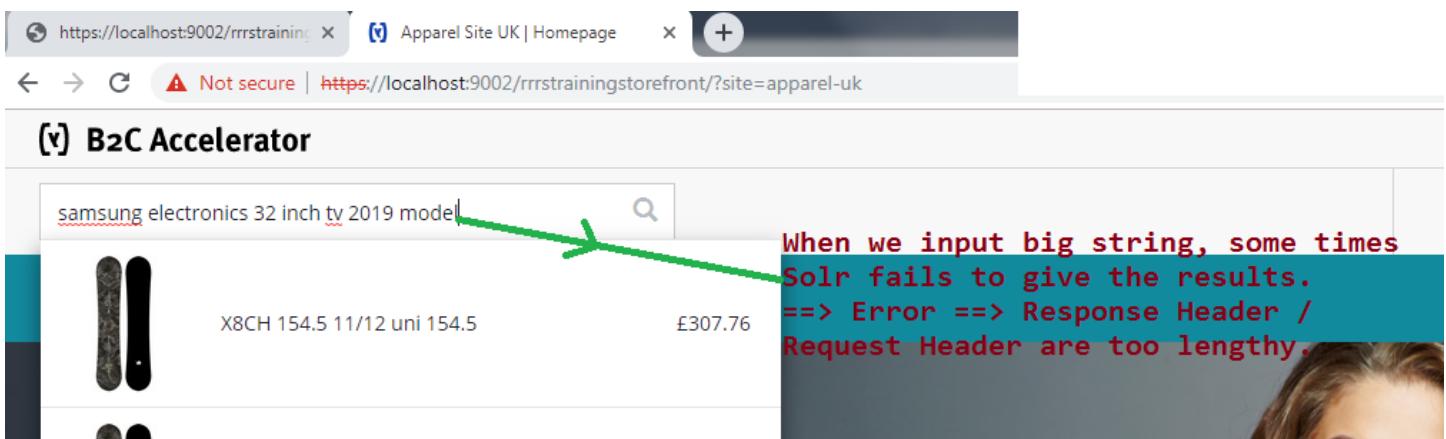
```

<target name="customize" description="Copies all files from '/config/customize' folder to '/bin' folder recursively">
    <callback extname="" target="before_customize"/>
    <customize/>
    <callback extname="" target="after_customize"/>

```

Below the code, another file explorer window shows the contents of the 'config' folder, specifically the 'customize' subfolder, which contains 'ext-platform-optional' and 'dummy' files.

## Requirement 1 =

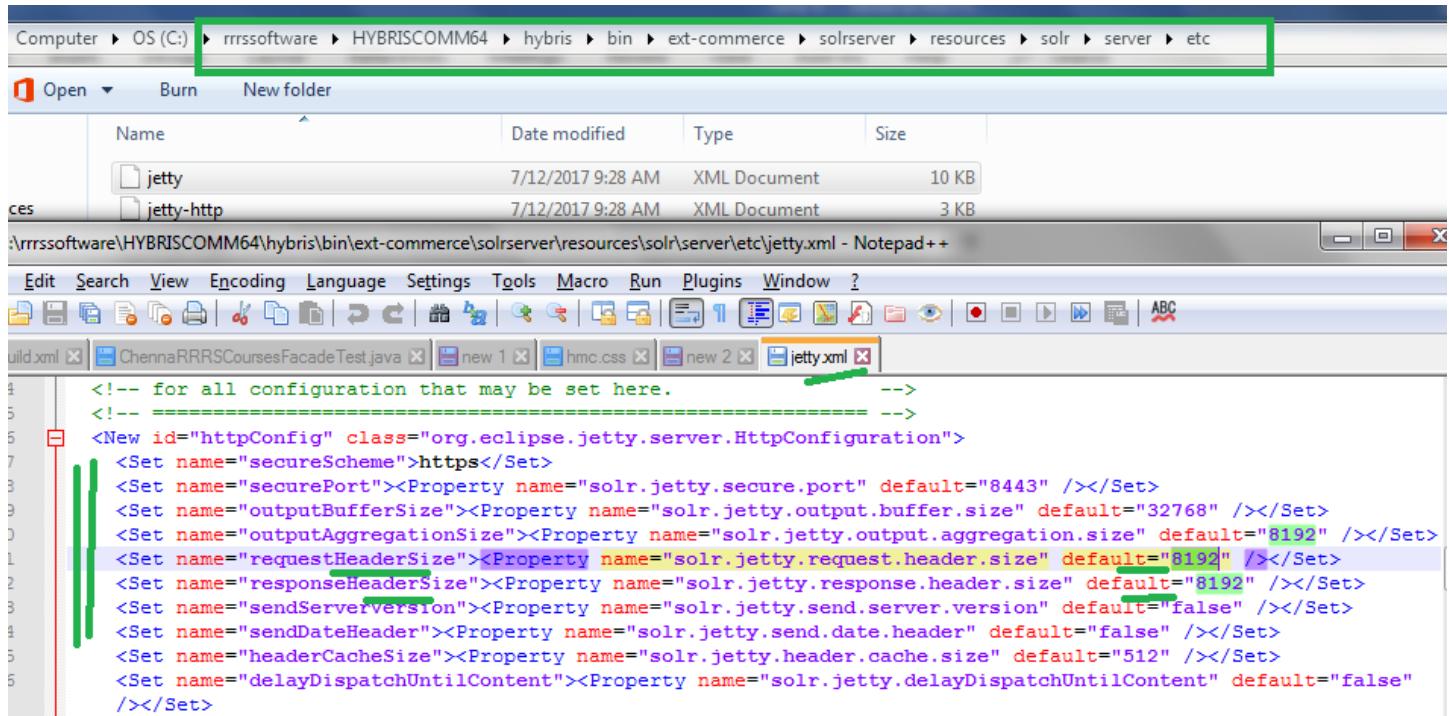


Sometimes fails to get the data when we input more characters in search box.

==== Error === Request header / Response header are too lengthy.

**Q: How to fix above?**

**Solution =**



The screenshot shows a Windows file explorer window with a green border around the address bar. The address bar displays the path: Computer > OS (C:) > rrssoftware > HYBRISCOMM64 > hybris > bin > ext-commerce > solrserver > resources > solr > server > etc. Below the address bar is a toolbar with icons for Open, Burn, and New folder. The main area shows a list of files:

Name	Date modified	Type	Size
jetty	7/12/2017 9:28 AM	XML Document	10 KB
jetty-http	7/12/2017 9:28 AM	XML Document	3 KB

Below the file list is a Notepad++ window titled '\rrssoftware\HYBRISCOMM64\hybris\bin\ext-commerce\solrserver\resources\solr\server\etc\jetty.xml - Notepad++'. The menu bar includes Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and Help. The toolbar below the menu has various icons for file operations. The status bar at the bottom shows tabs for build.xml, ChennaRRRSCoursesFacadeTest.java, new 1, hmc.css, new 2, and jetty.xml.

The XML code in the Notepad++ window is as follows:

```
<!-- for all configuration that may be set here. -->
<!-- ===== -->
<New id="httpConfig" class="org.eclipse.jetty.server.HttpConfiguration">
<Set name="secureScheme">https</Set>
<Set name="securePort"><Property name="solr.jetty.secure.port" default="8443" /></Set>
<Set name="outputBufferSize"><Property name="solr.jetty.output.buffer.size" default="32768" /></Set>
<Set name="outputAggregationSize"><Property name="solr.jetty.output.aggregation.size" default="8192" /></Set>
<Set name="requestHeaderSize"><Property name="solr.jetty.request.header.size" default="8192" /></Set>
<Set name="responseHeaderSize"><Property name="solr.jetty.response.header.size" default="8192" /></Set>
<Set name="sendServerVersion"><Property name="solr.jetty.send.server.version" default="false" /></Set>
<Set name="sendDateHeader"><Property name="solr.jetty.send.date.header" default="false" /></Set>
<Set name="headerCacheSize"><Property name="solr.jetty.header.cache.size" default="512" /></Set>
<Set name="delayDispatchUntilContent"><Property name="solr.jetty.delayDispatchUntilContent" default="false" /></Set>
```

**Challenge** → Above is the standard xml file available in bin folder.

Generally, we don't modify anything available within \bin (or) \platform folder.

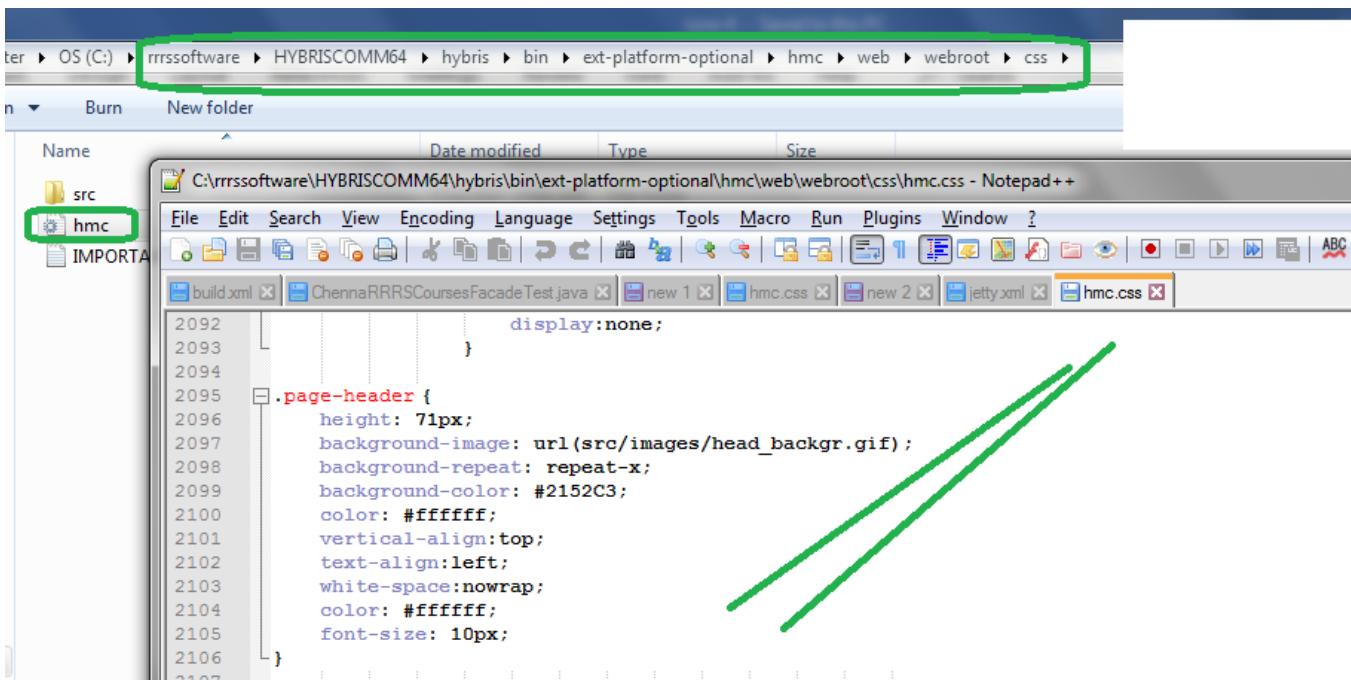
==> Bcoz, your changes may lose when upgrade happens.

**Best Solution = Use “ant customize” for this.**

## Requirement 2 =



Q = Where hmc standard CSS file is available?



**Challenge** → Above is the standard CSS file available in bin folder.

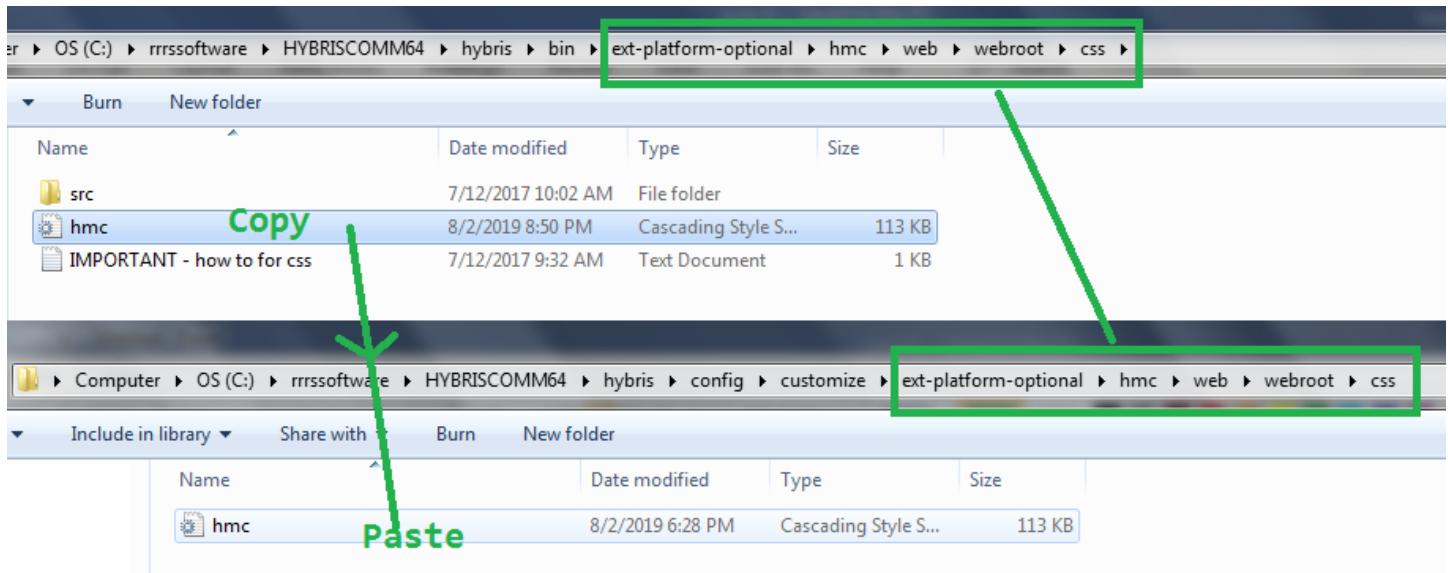
Generally, we don't modify anything available within \bin (or) \platform folder

==> Bcoz, your changes may lose when upgrade happens.

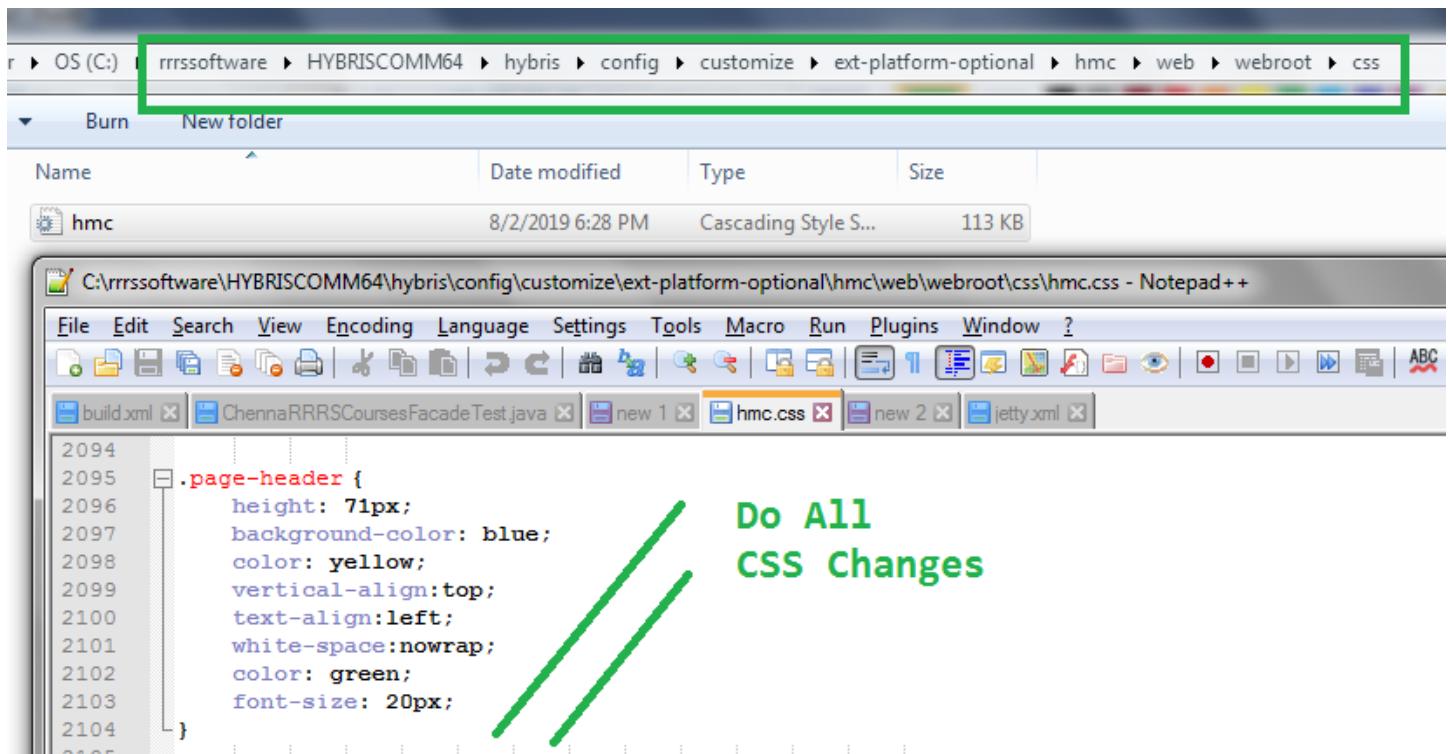
**Best Solution = Use “ant customize” for this.**

## How to use “ant customize”?

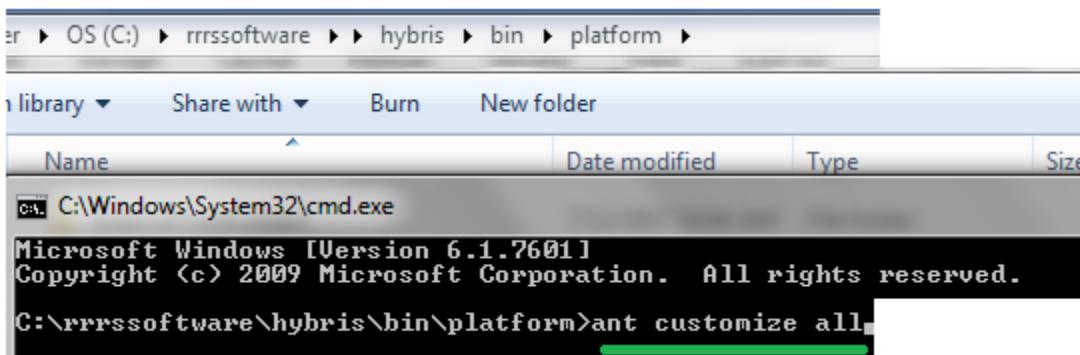
Step 1 = Copy hmc.css file from bin & paste in custom.



Step 2 = Do the CSS changes

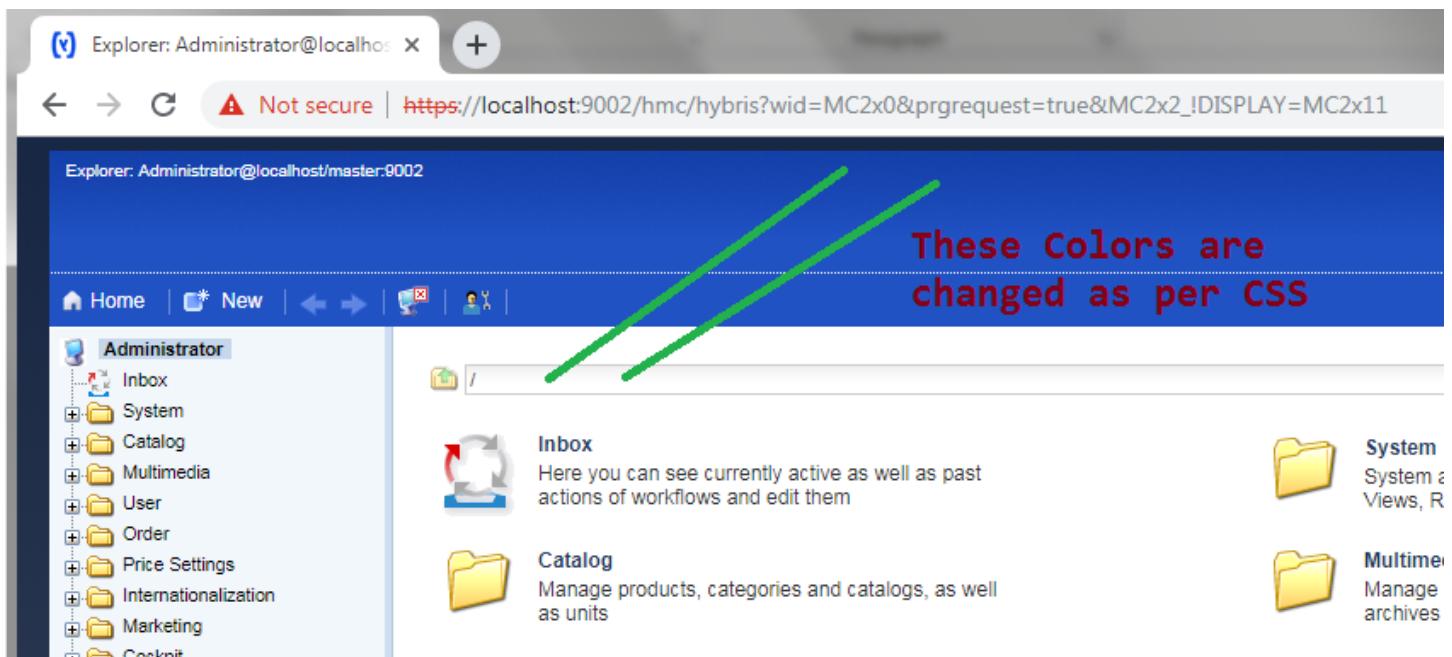


**Step 3 = Do “ant customize” (or) “ant customize all” (or) “ant clean customize all”**

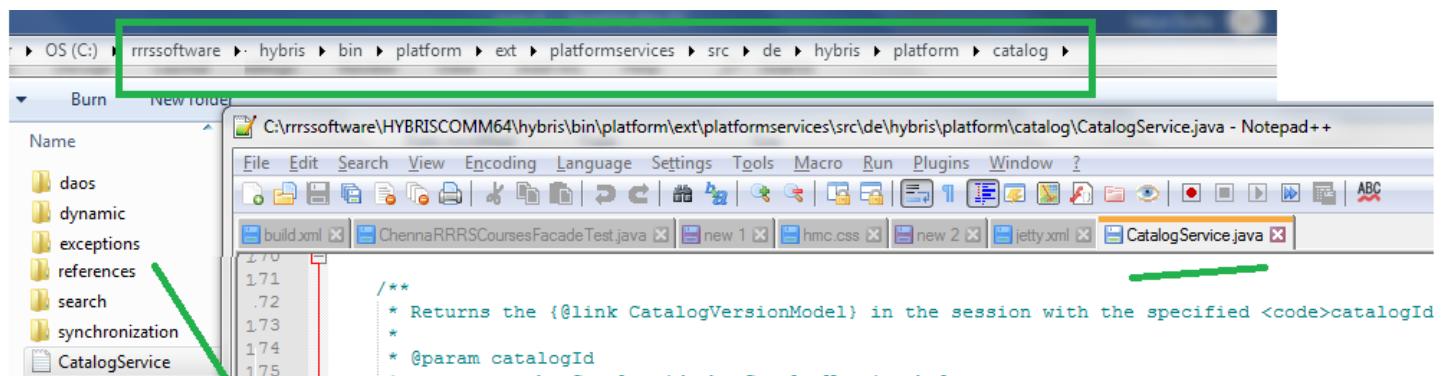


**Note:** - This replaces the customized files with original files.

**Step 4 = Do the build ... Start the Server ... Test the Results.**



**Another requirement [3] =**

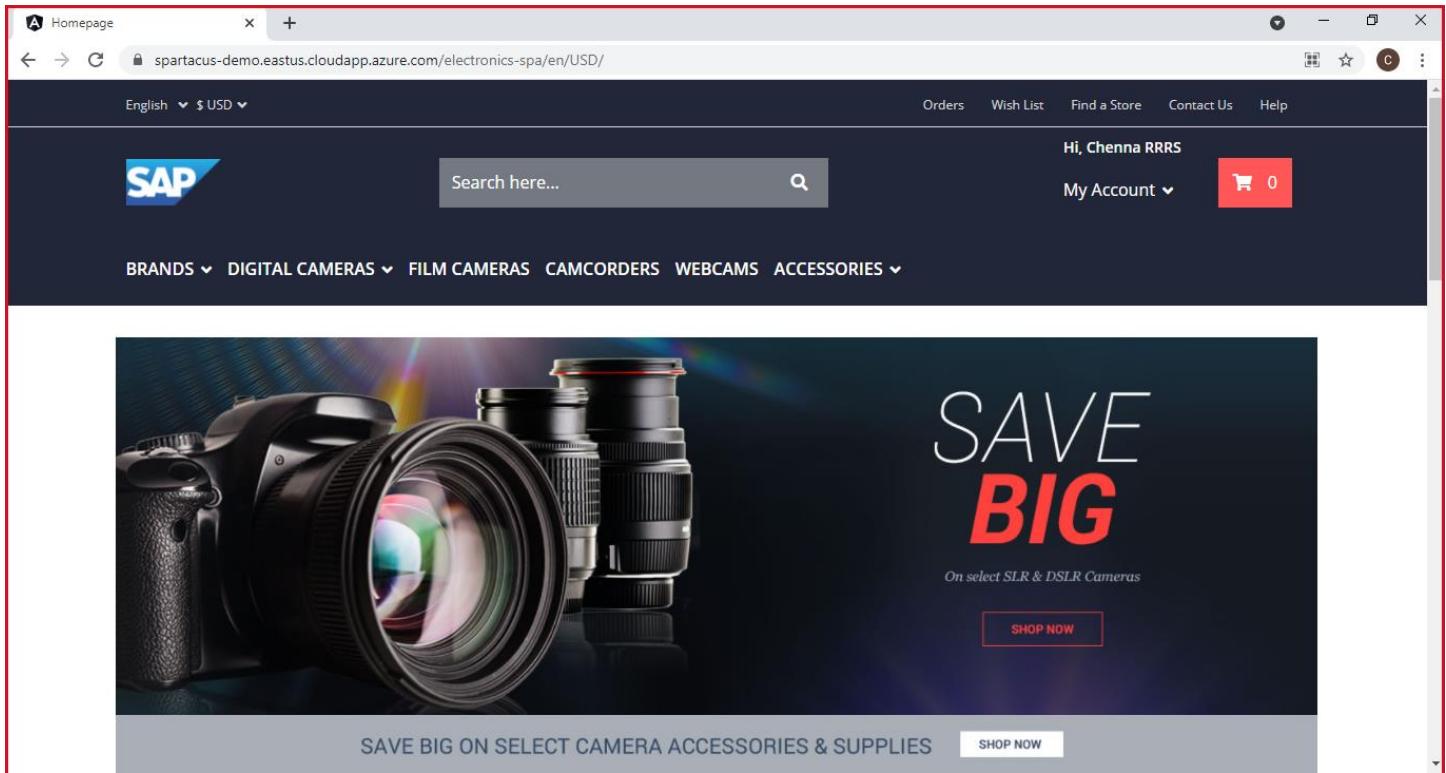


**Spartacus =**

Docs = <https://sap.github.io/spartacus-docs/>

Demo Site URL =

<https://spartacus-demo.eastus.cloudapp.azure.com/electronics-spa/en/USD/>



**Note:** - Spartacus interact with “SAP Comm” through OCC V2. Example: -

A screenshot of a web browser showing a product page for a camera. The URL is spartacus.c39j2-walkersde1-d4-public.model-t.cc.commerce.onDemand.com/electronics-spa/en/USD/product/1981415/PL60%20Silver. The page displays the camera and an "Add To Cart" button. Below the page, the browser's developer tools Network tab is open, showing a timeline and a list of requests under the "v2" tab. One request is highlighted: "entries?code=1981415&amp;qty=1&amp;lang=en&amp;curr=USD" with a status of 200, type xhr, initiator "polyfills.9d1a14d...js:1", size 1.53 kB, and time 1.53 s. The Network tab also shows other requests like "entries?code=1981415&amp;qty=1&amp;lang=en&amp;curr=USD" with a status of 200, type fetch, initiator "ngsw-worker.js:2726", size 1.5 KB, and time 357 ms.

Contact Us = ChennaReddyTraining@RRRS.CO.IN

## OCC Services =

URL = <https://localhost:9002/occ/v2/swagger-ui.html#/>

The screenshot shows the Swagger UI interface for the OCC Commerce Webservices. The title bar says "Swagger UI". The address bar shows "localhost:9002/occ/v2/swagger-ui.html#/". The top navigation bar has a "swagger" logo and a "Select a spec" dropdown set to "default". Below the header, the main content area is titled "Commerce Webservices 2.0". It includes a note about the base URL being "localhost:9002/occ/v2" and provides a link to "https://localhost:9002/occ/v2/api-docs". A message states that these services manage common commerce functionality and include AddOns customizations. It also mentions the "commercewebservices" extension and the SAP product terms of use. On the right side of the content area, there is a green "Authorize" button with a lock icon. The main content lists several service controllers:

- Address** Address Controller
- B2B Carts** B2B Carts Controller
- B2B Categories** B2B Categories Controller
- B2B Cost Centers** B2B Cost Centers Controller

The screenshot shows the Swagger UI interface for the OCC Core Services. The title bar says "Swagger UI". The address bar shows "localhost:9002/occ/v2/swagger-ui.html#/". The top navigation bar has a "swagger" logo and a "Select a spec" dropdown set to "default". Below the header, the main content area lists various service controllers:

- Catalogs** Catalogs Controller
- Components** Component Controller
- Consents** Consents Controller
- Consignment Tracking** Consignment Tracking Controller
- Countries** Countries Controller
- Customer Coupons** Customer Coupons Controller
- Customer Groups** Customer Groups Controller
- Export** Export Controller
- Extended Carts** Extended Carts Controller
- Feeds** Feeds Controller

## **Scenario = B2B**

**Q = Explain Organizational Structure in “SAP SD”?**

- **1st Level** = Organization (IBM)

IBM Producing Plants (SAP MM) = TX, MI / ...

o **2nd Level** = Sales Org (From where you get Products – TX / MI / ...) = SAP SD

▪ **3rd Level** = Distribution Channel (TX – Irving, TX – Dallas / ...)

- **4th Level** = Division

o **5th Level** = Sales Office

▪ **6th Level** = Sales Group

**Note:** - Sales Area (SA) = Sales Org + Distribution Channel + Division

**Q = What is Business Partner?**

Parties involved in a business transaction (Customer, Carrier, Employee and ...). There are 4 default business partner parties (Payer, Sold-To, Ship-To and Bill-To).

**Q = Explain SAP Partner Functions?**

It is roles & responsibility a business partner can take on in a business Tr.

- We have 2 types of **Master Data**: -

o **Vendor** (Who Provides products) = Material Management Side

Verizon get products from Office Depo (Vendor).

In Office Depo System – Whom to send Invoice in Verizon = **BillTo** (AP Dept).

In Office Depo System – Whom to Ship in Verizon = **ShipTo** (Front Office).

o **Customer** (Who take your product) = Sales Side

Verizon sells Fios to CCN.

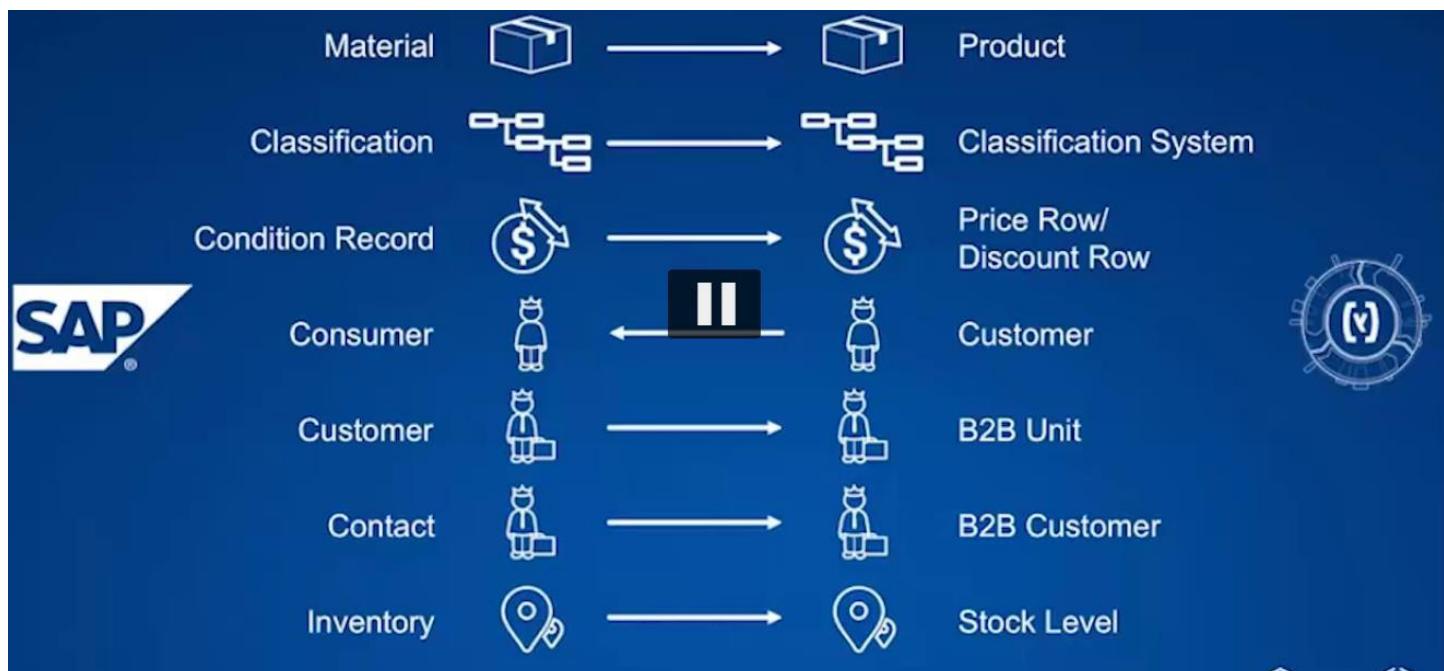
In Verizon System – Whom to Sell = **SoldTo**

In Verizon System – Whom to Invoice = **BillTo**

SRM = Company Buy (Purchase Order) &

CRM = Company Sell (Sales Order)

**Q** = What are the mapping entities between SAP & “SAP Comm”?



SAP IDOC	"SAP Comm" Item Type	Description
MATMAS	Product	Product Data
LOISTD	Stock Level	Stock / Inventory Information
CLSMAS	Category & Product	Classification Hierarchy
CLFMAS	Feature, Feature Value, Feature Assignment, Category	Classification Data
DEBMAS	Customer & B2BUnit	Customer Data
ADRMAS	B2BUnit & Address	Customer Address Data
ADR3MAS	B2BUnit & Address	Customer Address Data
COND_A04	Price Row & Discount Row	Customer-Specific Pricing (Price Condition)

## B2B Unit =

An Org Unit representing a Company Code, Department Node / Branch Node.

Branches of the B2B Root Unit is **B2B Org**.

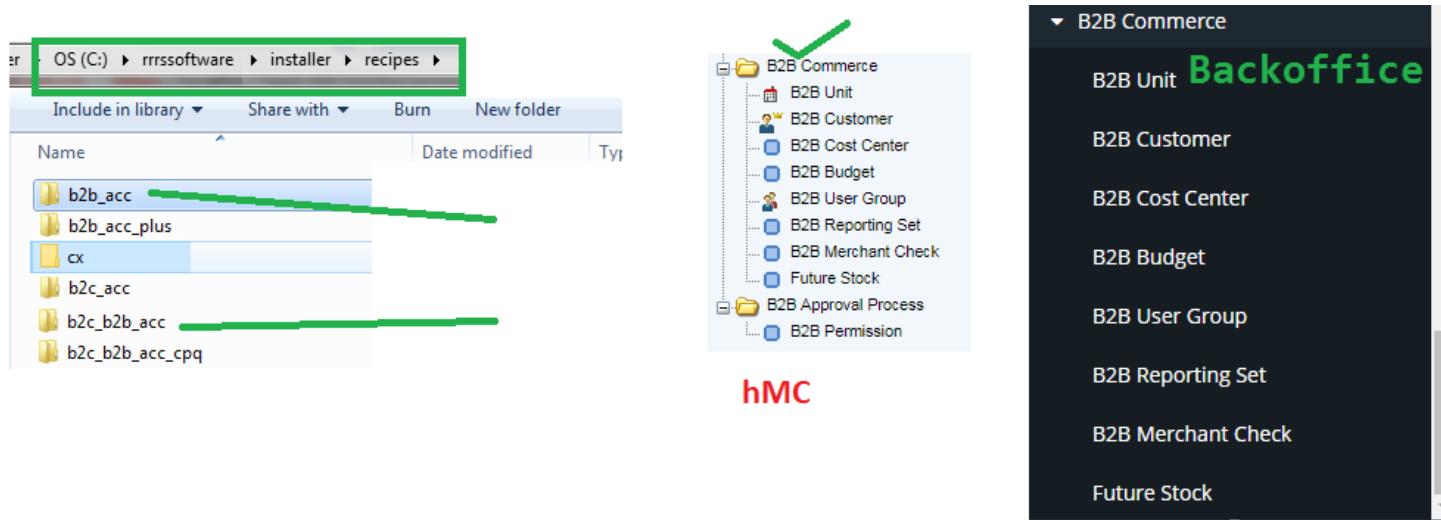
Orders are assigned to **B2B Units**.

B2B Units can have an approvers group assigned.

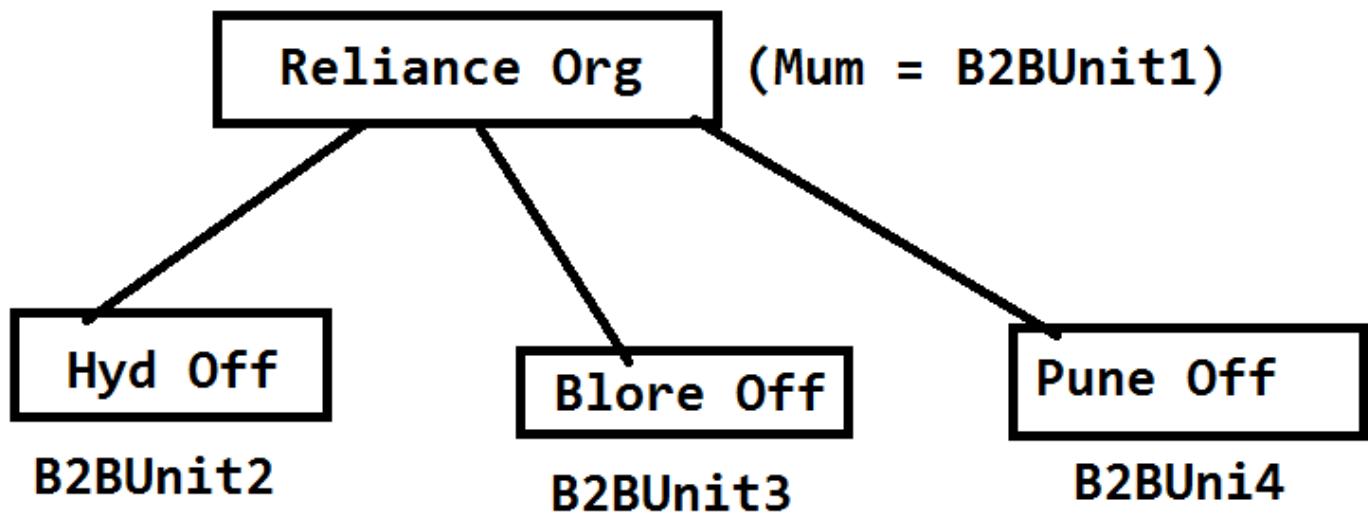
B2B Customers are assigned to B2B Unit. They also members of B2B Group.

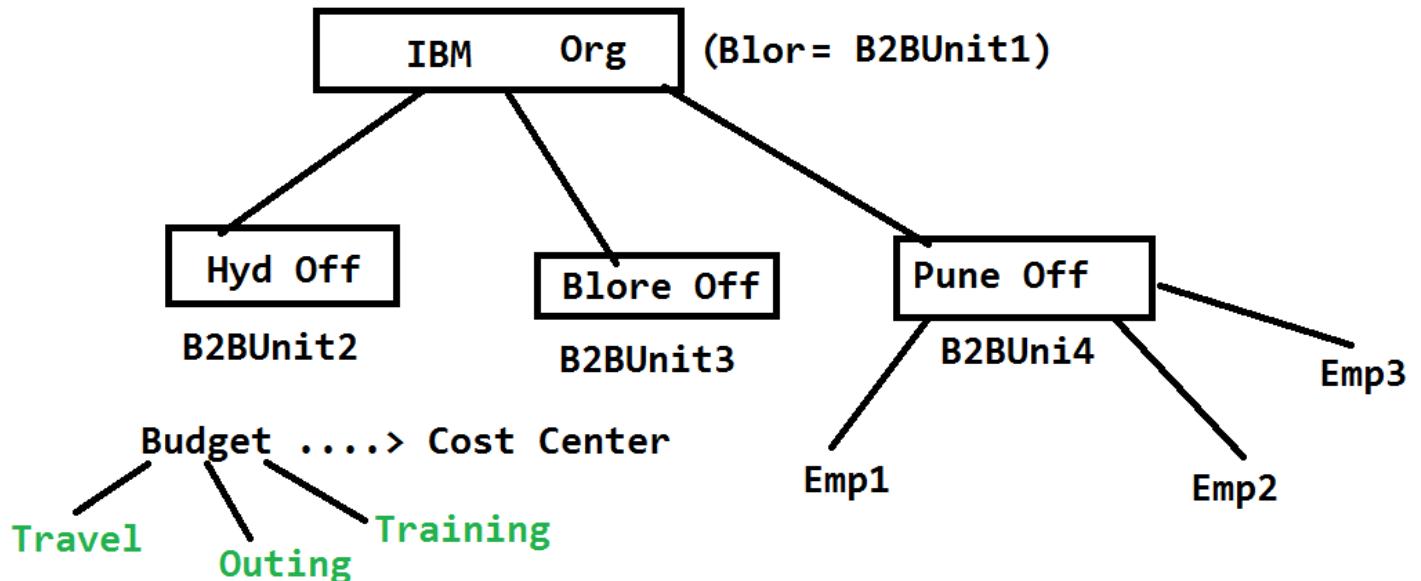
- AG = Sold To Party (Customers Who Place the Orders)
- RE = Bill To Party (Customers Who Receive the Invoice)
- RG = Payer (Customers Who Pay the Invoice)
- WE = Ship To Party (Customers Who Receive the Goods Recipient / Services)
- ZU = Author (Forwarding Agent)

B2B eCommerce is based on “B2B Unit ... B2B Customer ... ShipTo ... SoldTo ... Budget ... Cost Center ... B2B Approval Process ... B2B Permission ... ”.



**B2B Unit** = Organization / Division / Department / Office / .....





**Note** = If any B2BUnit is not having Parent B2BUnit then it's that B2BUnit is called “Root Organization”. You can activate / deactivate B2BUnits.

The screenshot shows the SAP Administration Cockpit interface. On the left, a navigation tree is visible with 'B2B Commerce' expanded, and 'B2B Unit' selected. A green circle highlights the 'B2B Unit' node. In the center, a modal window titled 'Create New B2B Unit' is open. Another green circle highlights the '+' button in the toolbar of the modal. The modal has tabs for 'GENERAL' and 'READ & WRITE'. The 'GENERAL' tab is active, showing fields for 'ID' (set to 'B2BUnit1') and 'Name' (also set to 'B2BUnit1'). The 'READ & WRITE' tab is shown below. At the bottom right of the modal, there are 'CANCEL', 'NEXT', and a yellow 'DONE' button.

SAP Administration Cockpit

Filter tree (Alt+Down for options)

B2B Commerce

- Document
- PunchOut Credential
- B2B PunchOut Mapping
- B2B Unit**
- B2B Customer
- B2B Cost Center
- B2B Budget

SAVED QUERIES

No queries

B2BUnit2

SEARCH

REFRESH SAV

APPROVERS GENERAL ADDRESSES ORGANIZATION LANGUAGES PRICES PERSONALIZATION COST CENTER MERCHANT CHECK

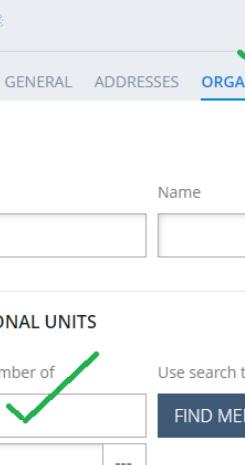
ESSENTIAL

ID	Name	Name
B2BUnit2		B2B Unit 2

ORGANIZATIONAL UNITS

Company is member of  Use search to get Members

FIND MEMBERS OF THIS USERGROUP



SAP Administration Cockpit

Filter tree (Alt+Down for options)

**B2B Commerce**

- Document
- PunchOut Credential
- B2B PunchOut Mapping
- B2B Unit
- B2B Customer
- B2B Cost Center
- B2B Budget**

SAVED QUERIES

No queries

Create New B2B Budget

MANDATORY VALUES

Parent B2B Unit:

Active:  True  False

Date Range:

START: Apr 1, 2021 12:00:00 AM

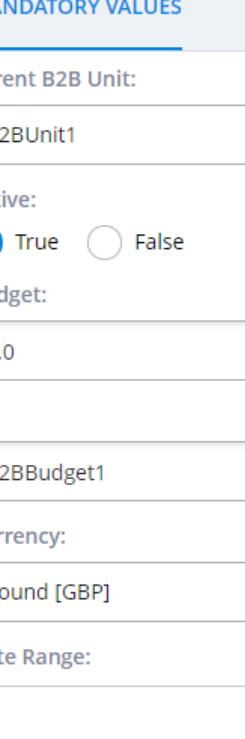
END: Jul 1, 2021 11:59:59 PM

Budget:

Currency: Pound [GBP]

Time created:

Date Range: CANCEL DONE



SAP Administration Cockpit

Filter tree (Alt+Down for options)

- B2B Commerce
- Document
- PunchOut Credential
- B2B PunchOut Mapping
- B2B Unit
- B2B Customer
- B2B Cost Center**
- B2B Budget

SAVED QUERIES

No queries

0 ITEMS SAVED

### Create New B2B Cost Center

**MANDATORY VALUES**

Parent B2B Unit:  
B2BUnit2

Active:  
 True  False

ID:  
CostCenter1

Currency:  
Pound [GBP]

Time created:  
Jul 23, 2021 5:59:18 AM

Filter tree (Alt+Down for options)

- B2B Unit
- B2B Customer
- B2B Cost Center
- B2B Budget
- B2B User Group
- B2B Reporting Set
- B2B Merchant Check**
- Future Stock

SAVED QUERIES

No queries

0 ITEMS SAVED

### Create New B2B Credit Limit

**GENERAL**

ID :  
B2BMerchantCheck

Active:  
 True  False

Amount :  
1000

Currency:  
Pound [GBP]

Date Period:  
START

**B2B Unit**

- B2B Customer
- B2B Cost Center
- B2B Budget
- B2B User Group
- B2B Reporting Set
- B2B Merchant Check**
- Future Stock

**SAVED QUERIES**

No queries

**B2BMerchantCheck**

**GENERAL** ADMINISTRATION

**PROPERTIES**

Active	Date Period	Date Range	Currency
<input checked="" type="radio"/> True <input type="radio"/> False	START Jul 23, 2021 6:01:16 AM	END Jul 24, 2021 6:01:18 AM	Pound [GBP]
Amount	B2B Unit	Alert Threshold	Alert Rate Type
1000.00000000	B2BUnit2	20.00000000	PERCENTAGE
...			

**B2B Unit**

- B2B Customer
- B2B Cost Center
- B2B Budget
- B2B User Group
- B2B Reporting Set
- B2B Merchant Check
- Future Stock

**SAVED QUERIES**

No queries

**B2BUnit2**

**COST CENTER**

**ESSENTIAL**

ID	Name	Name
B2BUnit2		B2B Unit 2

**COST CENTER**

B2B Budgets	B2B Cost Centers	B2B Orders
ID	Name	ID
R2BRRBudget1	CostCenter1	
...	...	...

**B2B Unit**

- B2B Customer
- B2B Cost Center
- B2B Budget
- B2B User Group
- B2B Reporting Set
- B2B Merchant Check
- Future Stock

**SAVED QUERIES**

No queries

**B2BUnit2**

**MERCHANT CHECK**

**ACCOUNT MANAGERS**

Account Manager	Account Manager Groups
...	...

**CREDIT LIMIT**

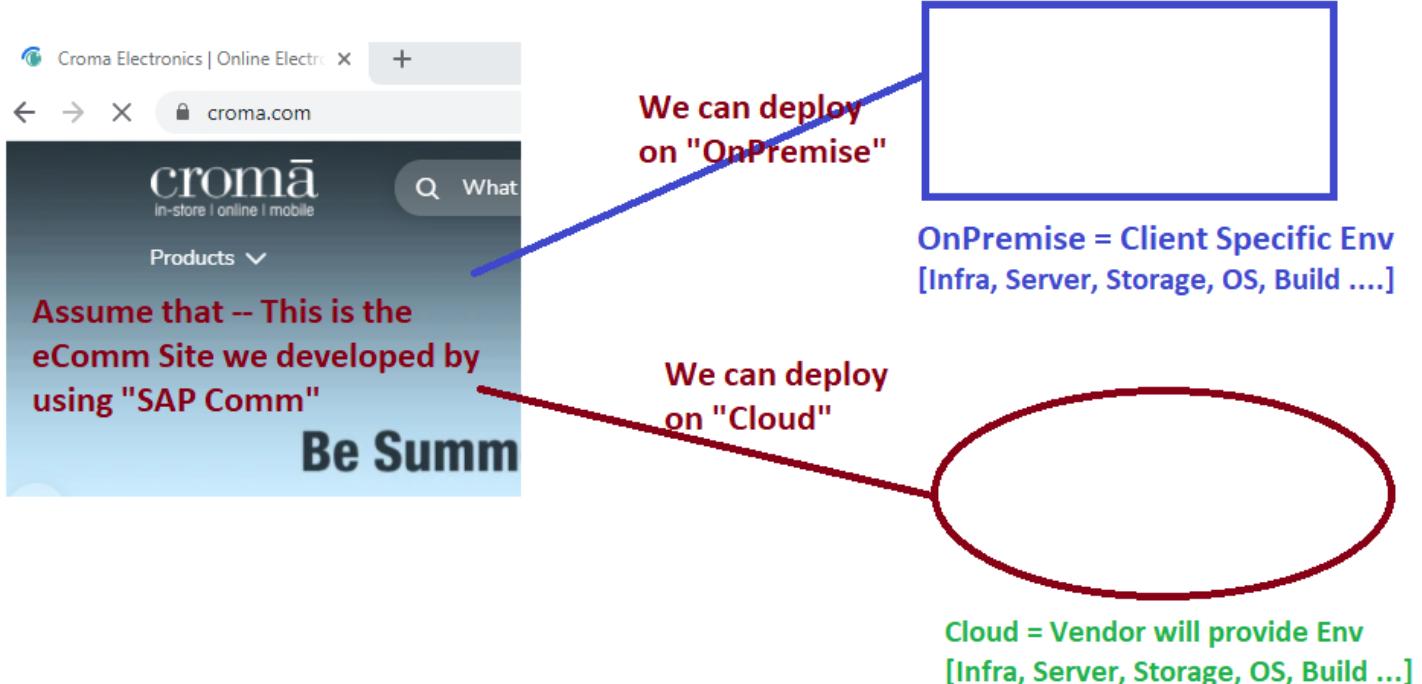
Credit Limit
B2BMerchantCheck

# Scenario = Deployment

After developing the eComm / Shopping Site, we need deploy / host it :-

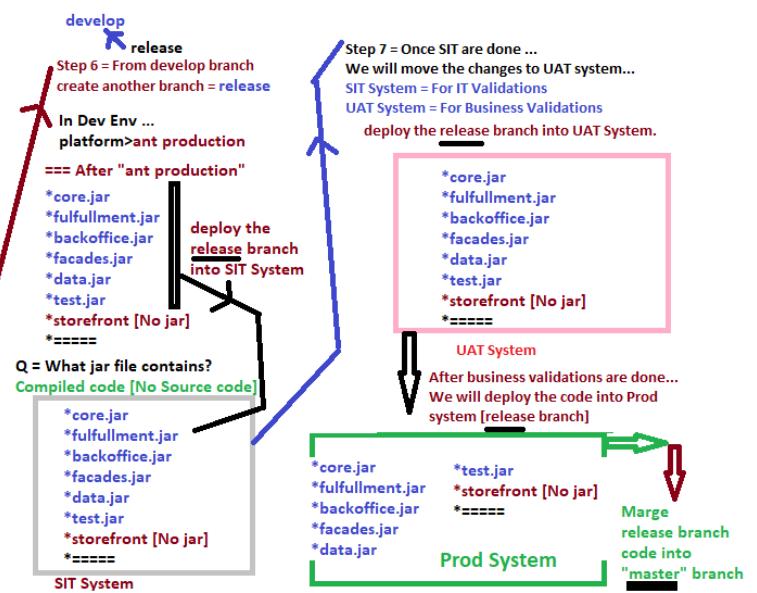
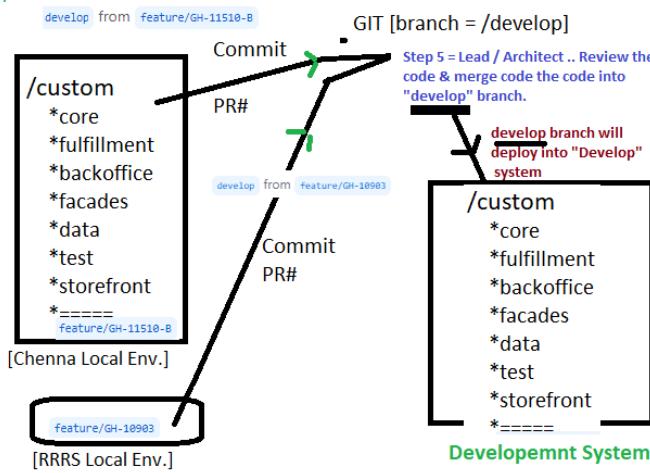
1) We can use client specific Environment / Infra = **OnPremise**

2) We can use Vendor specific Environment / Infra = **Cloud**



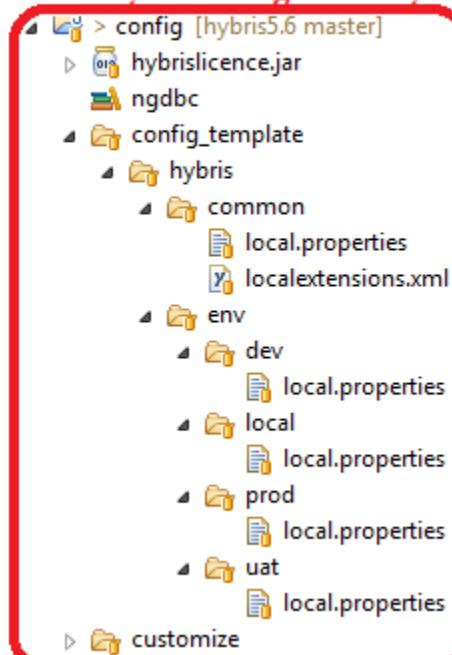
How the deployment happens in "OnPremise"

- Step 1 = Take code from /develop branch
- Step 2 = Give local branch name = feature/GH-11510-8
- Step 3 = Chennai will do the changes in "feature/GH-11510-8" & test it locally
- Step 4 = Commit the code & Raise PR#



**Q = Explain How to Manage the configuration properties for multiple environments [Dev ... QA ... UAT ... Prod]?**

**I. Create below folder structure inside hybris config directory**



**Step 2 =**

Place your properties in each one of local files as per environment related values. The property which have same values across environment will go to common local property file.

**Step 3 =**

Create a build.xml file inside config project and copy below content.

**Step 4 =**

Now we have a ANT build xml, which will create a environment specific property file. You can run like below pass the environment name as required.

```
c:\hybris\config>ant -Denv=uat
```

Now you have a flexible solution to configure your hybris system more efficiently.

**Q = Production Infrastructure for SAP Commerce (or) What is ideal Production server setup? (or) How many servers, cores / nodes do customers need?**

