System.debug('Welcome');

//Create an instance of ContactManager and invoke method

ContactManager Manager = new ContactManager();

String msg = Manager.createContact();

System.debug(msg);

//List, Set and Map

List<Integer> intList = new List<Integer>();

intList.add(30);

intList.add(20);

intList.add(40);

intList.add(5);

intList.add(50);

//Display List Elements

System.debug(intList);

//Set

Set<Integer> intList = new Set<Integer>();

intList.add(30);

intList.add(20);

intList.add(40);

intList.add(5);

intList.add(50);

intList.add(50);

intList.add(40);

intList.add(30);

//Display List Elements

System.debug(intList);

result is sorted set and duplicates are trimmed

since here it uses sorted set

// Iterate enhanced for loop

for(Integer x :intList){

System.debug(x);

}

in an loop system.debug uses linked hashset

earlier it was hash set now its linked hash set

//Map

Map<String, String> addressMap = new Map<String, String>();

addressMap.put('Street','508 Neelkanth');

addressMap.put('Area','Kodigehalli');

addressMap.put('City','Bangalore');

addressMap.put('Country','India');

if(addressMap.ContainsKey('Street')){

System.debug(addressMap.get('Street'));

}

// list of 1200 accounts -- create 1200 accounts

List<Account> accList = new List<Account>();

for(Integer i=1; i<=1200; i++){

Account acc = new Account(Name='Acc'+i,Type='Customer');

//insert.acc; this is wrong, most dangerous programming, 1200 DML operations cant be triggered. 150 is the limit

// it doesnot mean that it insert first 150 also.. since its all or none, system limit exception is raised. whole transaction is rolled back

// solution to this problem is creating a list

accList.add(acc);

}

insert accList;

// so we user 1 DML instead of 1200 DML

// to note: we can deal with only 10000 records, if we go beyond it gives errors

// SOQL- sales force object query language

//hits the heap so there is no select \* from.. but we need to specify the fields

//there are 3 different types of return types

//1. List of records

// List<Account> accList = [SELECT Name, Type from Account];

// Square bracketed expression or dynamic query are 2 types of retrival

//Display Name and Type of All accounts

for(Account acc: accList){

System.debug(acc.Name+'::'+acc.Type);

}

//Delete all records we created

//Appling LIKE

List<Account> accList = [SELECT Name, Type

from Account

Where Name LIKE 'Acc%'];

System.debug(accList.size());

//Deletion

delete accList;

Here the select returns 3 fields: ID, Name, Type

ID is by default available

Its LIKE doing this

List<Account> accList = [SELECT Name, Type ,Id

from Account

Where Name LIKE 'Acc%'];

//If we query a field which we did not query we get a error

//System.SObjectException: SObject now was retrived via SOQL without querying the requested field: Field name

//Using all rows -- Shows recored from recycle bin too

List<Account> accList = [SELECT Name, Type

from Account ALL ROWS];

System.debug(accList.size());

//Show records only from recycle bin == also undelete

List<Account> accList = [SELECT Name, Type

from Account ALL ROWS];

System.debug(accList.size());

List<Account> accList = [SELECT Name, Type

from Account

Where IsDeleted=True

ALL ROWS];

undelete accList;

System.debug(accList.size());

=========================

control + Shift + O (leter o)

object console-- select which ever object ex: account.obj double click

now it shows all fields

shift select couple of fields and below query button is there

we can click and then see query button to run

select a record and see below open detail page, edit page or create New Record pages

===========================

//Return one single record - Sobject is a default return type

Account acc = [Select Name

FROM Account

WHERE ID='0017F00000MYVpy'];

System.debug(acc);

if this query yields nothing it leads to exception

But we can solve this by doing the return type as List.

So that it yeilds to Empty List

List<Account> acc = [Select Name

FROM Account

WHERE ID='0017F00000MYVpy'];

System.debug(acc);

//Returns Integer

Integer cnt = [Select count()

From Account];

System.debug(cnt);

\*\*\*\*\*Returns list, records or integers, aggregate result(based on group by )

//List of Lists

List<List<sobject>> records = [FIND 'Acme\*' IN ALL FIELDS

RETURNING Account(Name, Type),

Opportunity(Name, Amount where Amount>5000)];

List<Account> accList = records[0];

List<Opportunity> oppList = records[1];

System.debug('Account....');

for(Account acc : accList){

System.debug(acc.Name+'->'+acc.Type);

}

System.debug('Opportunities....');

for(Opportunity opp : oppList){

System.debug(opp.Name+'->'+opp.Amount);

}

// DML

2 types of DML

1.Stand alone DML -- all or none basis execution

if we try insert 10 records, even if 1 is not fitting criteria, none of them execute

insert, update, delete all are examples of same

2. Data base DML

Contact c1 = new Contact(FirstName='John', LastName='Ben');

Contact c2 = new Contact(FirstName='Johnny'); // last name is missed-- so leads to error

//Line: 131, Column: 1

//System.DmlException: Insert failed.

//First exception on row 1; first error: REQUIRED\_FIELD\_MISSING, Required fields are missing: [LastName]: [LastName]

List<Contact> conList = new List<Contact>{c1,c2};

// exam question can be on syntax, instead of calling add 2 times, initialise it with 2 values

//Exception handling for custom error msg

Contact c1 = new Contact(FirstName='John', LastName='Ben');

Contact c2 = new Contact(FirstName='Johnny');

List<Contact> conList = new List<Contact>{c1,c2};

try{

insert conList;

}catch(DmlException ex){

System.debug('Probelm in insert.....'+ ex.getMessage());

}

//Database DML

Contact c1 = new Contact(FirstName='John', LastName='Ben');

Contact c2 = new Contact(FirstName='Johnny');

List<Contact> conList = new List<Contact>{c1,c2};

List<Database.SaveResult> srlist= Database.insert(conList,false);

for(Database.SaveResult sr :srList){

if(!sr.isSuccess()){

System.debug(sr.getErrors());

}

}

//false means partial insert can be done, default its true - wont allow partial insert

//how to understand something failed so we use database.SaveResult-- we dont use try catch in this case

//using for loop we are iterating on result set

//sr.getErrors() lists lot of errors.. so we need to display them again

//to get the list of errors

//Database DML

Contact c1 = new Contact(FirstName='John', LastName='Ben');

Contact c2 = new Contact(FirstName='Johnny');

List<Contact> conList = new List<Contact>{c1,c2};

List<Database.SaveResult> srlist= Database.insert(conList,false);

for(Database.SaveResult sr :srList){

if(!sr.isSuccess()){

List<Database.Error> errorList = sr.getErrors();

for(Database.Error err : errorList){

System.debug(err.getMessage());

}

}

}

to get which error is for which line we can do

Contact c1 = new Contact(FirstName='John', LastName='Ben');

Contact c2 = new Contact(FirstName='Johnny');

List<Contact> conList = new List<Contact>{c1,c2};

List<Database.SaveResult> srlist= Database.insert(conList,false);

for(Integer i=0; i<srList.size();i++){

Database.SaveResult sr = srList[i];

if(!sr.isSuccess()){

List<Database.Error> errorList = sr.getErrors();

System.debug((i+1)+'record has error');

for(Database.Error err : errorList){

System.debug(err.getMessage());

}

}

}

==============

Inheritance

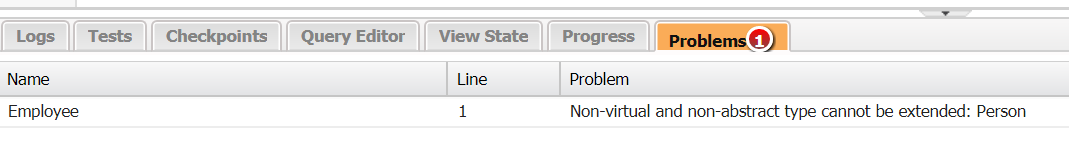
Joins(SOQL)

Triggers-- imp thing in salesforce

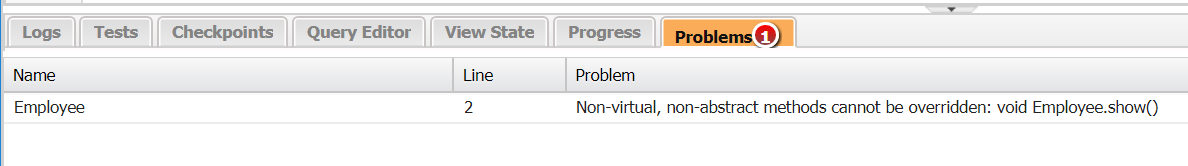
Inheritance is taken from C#.

In Salesforce, all classes are bydefault final. So the parent should be virtual or abstract.

When I do employee extends person.



While overriding



virtual public class Person {

virtual public void show(){

}

abstract public class Person {

abstract public void show(){

}

}

}

public class Employee extends Person{

override public void show(){

}

}

We need to make parent method virtual and use override in child.

Java by default all classes and methods are virtual but in C# we need to mention that.

Instead of virtual u can make it abstract.

public interface Fyler {

void takeoff();

}

In interface we cant have access specifier in methods

public class Employee extends Person implements Fyler{

override public void show(){

}

public void takeoff(){

}

}

**Joins**

We just went to job application and created a few records with and without position.

So that we can do joins

Position is parent and job application is child.

Job Application

Position

Left outer join: Fetch all positions along with job application details

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | JobApplications\_\_r | Position Object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Position | Position\_\_r | Job application Object |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Position | Position\_\_r | Job application Object |

// Left Outer Join

List<Position\_\_c> posList = [SELECT Name,(SELECT Name FROM Job\_Applications\_\_r)

From Position\_\_c];

for(Position\_\_c pos: posList){

System.debug(pos.Name);

for(Job\_Application\_\_c jobApp: pos.Job\_Applications\_\_r){

System.debug('---->'+jobApp.Name);

}

}

//Right outer Join

//list of job applciations and postion assocaited

List<Job\_Application\_\_c> JobAppList = [Select Name, Position\_\_r.Name

From Job\_Application\_\_c];

for(Job\_Application\_\_c jobApp : JobAppList){

System.debug(jobApp.Name+'::'+ jobApp.Position\_\_r.Name);

}

//EQUI Join From Child

//Fetch job applications that have positions

List<Job\_Application\_\_c> JobAppList =[Select Name

From Job\_Application\_\_c

Where Position\_\_c != null];

for(Job\_Application\_\_c jobApp : JobAppList){

System.debug(jobApp.Name);

}

// Equi join from Parent

//we need 2 queries one on parent and one on child- take each postion and check against job //application

// fetch all postions who have job applications

List<Position\_\_c> posList = [Select Name

from Position\_\_c

Where Id in(select Position\_\_c

From Job\_Application\_\_c)];

for(Position\_\_c pos : posList){

System.debug(pos.Name);

}

//to display position and job application name

List<Position\_\_c> posList = [SELECT Name,(SELECT Name FROM Job\_Applications\_\_r)

From Position\_\_c

Where Id in(select Position\_\_c

From Job\_Application\_\_c)];

for(Position\_\_c pos: posList){

System.debug(pos.Name);

for(Job\_Application\_\_c jobApp: pos.Job\_Applications\_\_r){

System.debug('---->'+jobApp.Name);

}

}

// Anti join from Parent

// fetch all postions who donot have job applications

List<Position\_\_c> posList = [Select Name

from Position\_\_c

Where Id NOt IN(select Position\_\_c

From Job\_Application\_\_c)];

for(Position\_\_c pos : posList){

System.debug(pos.Name);

}

//Fetch job applications doesnot that have positions

List<Job\_Application\_\_c> JobAppList =[Select Name

From Job\_Application\_\_c

Where Position\_\_c = null];

for(Job\_Application\_\_c jobApp : JobAppList){

System.debug(jobApp.Name);

}

=============================================================

**Triggers**:

1. Types (Before and after)
2. Events associated – (before/after insert, before/after update, before/after delete, after undelete)
3. Context variables – (New, Old, NewMap, OldMap, isInsert, isUpdate, isdelete, isUndelete, isBefore, isAfter, isExecuting)
4. Syntax and structure
5. Bulkification
6. Execution order
7. Recursive triggers and solution

All trigger context variables are static variables of trigger class.

Trigger.new—returns list of new version of records

Trigger.old – gives list of old version of records

Means before update(old) or after update (new version) incase we are editing any records.

List because—we can use tools to push/delete/update records like dataloader

If I push 1000 records, salesforce will divide then into each csv of 200 records and perform action in batches on salesforce servers.

Triggers always take batch to perform operation and that batch size is 200.

Bulkification means—think abt many records, don’t think abt one record we are using. Think about tools that can work on hundreds of records.

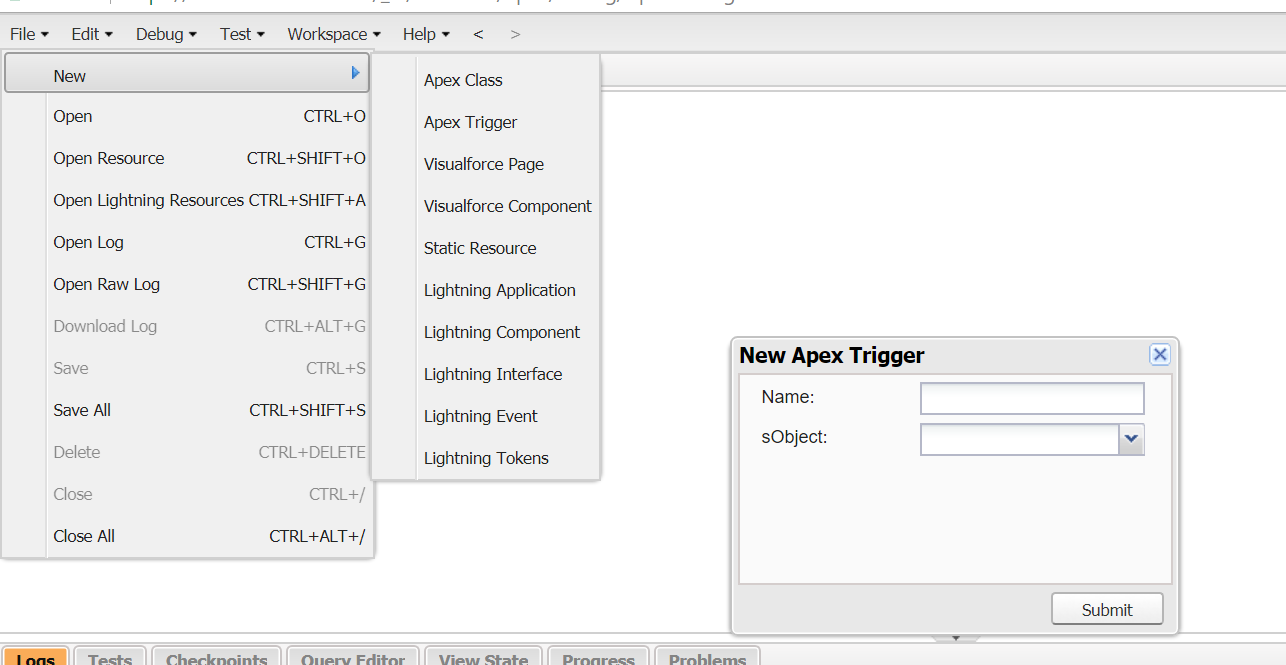
//smallest trigger example on earth :P

How to achieve hello world in trigger

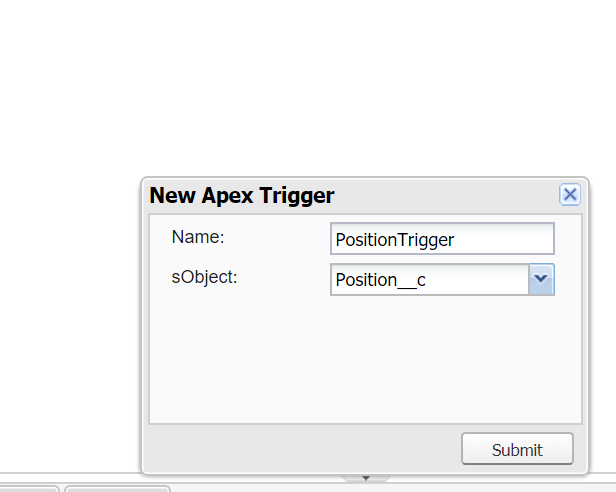
We will create a field called hello on position. Either I create or edit that field it should get world in it.

Create fileds on postion with text data type: Hello.

Go to developer console – new trigger



When we write a trigger, each object will have one trigger and one handler class. One to one relation is done.



Naming convention to follow objectname+trigger and select the object.

Object and trigger are tightly coupled.

Either u create or edit a record we want to populate world.

trigger PositionTrigger on Position\_\_c (before insert,) { …… }

trigger PositionTrigger on Position\_\_c (before insert, before update) {………………}change this way

suppose if I have csv file having 200 records, the 200records will be referred by this reference variable.

trigger PositionTrigger on Position\_\_c (before insert,before update) {

List<Position\_\_c> posList = Trigger.New;

// Check whether Hello\_\_c contains 'WORLD', if not populate the same

// for all the position records

//Iterate through Trigger.New -- nothing but posList

for(Position\_\_c pos : posList){

if(pos.Hello\_\_c != 'World'){

pos.Hello\_\_c = 'World';

}

}

}

Test by creating a record in position tab

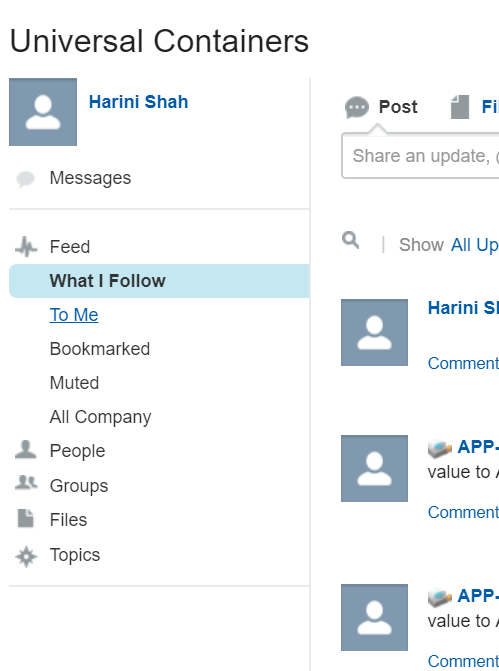
Now a days clients want to be enabled with process—rather than programming. Since programming has governor limits but from salesforce process will not have same.

**Next requirement:**

The moment position status becomes open, this position should be posted on chatter group called All Universal Containers

Functionally its: announce all open positions on chatter of All Universal container

Go to Chatter tab:

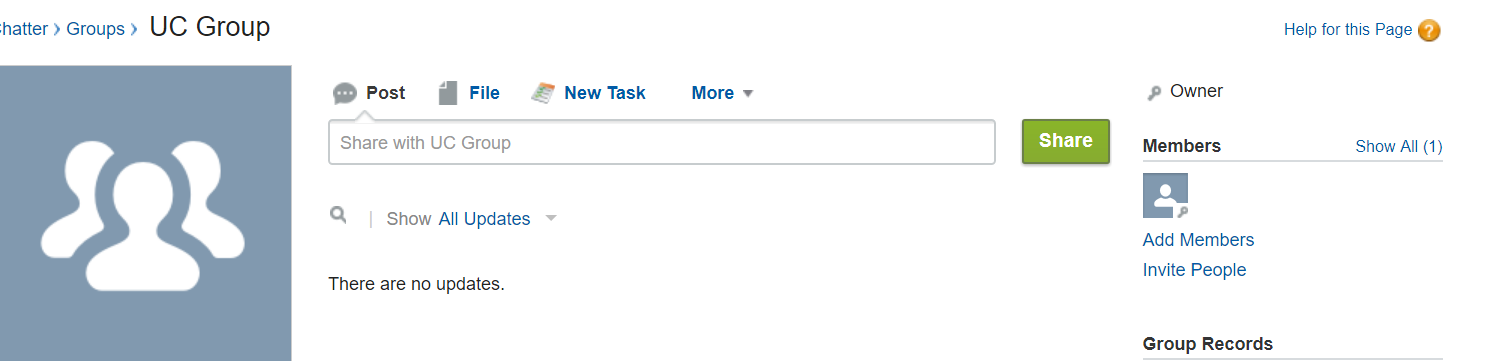


Click on groups,

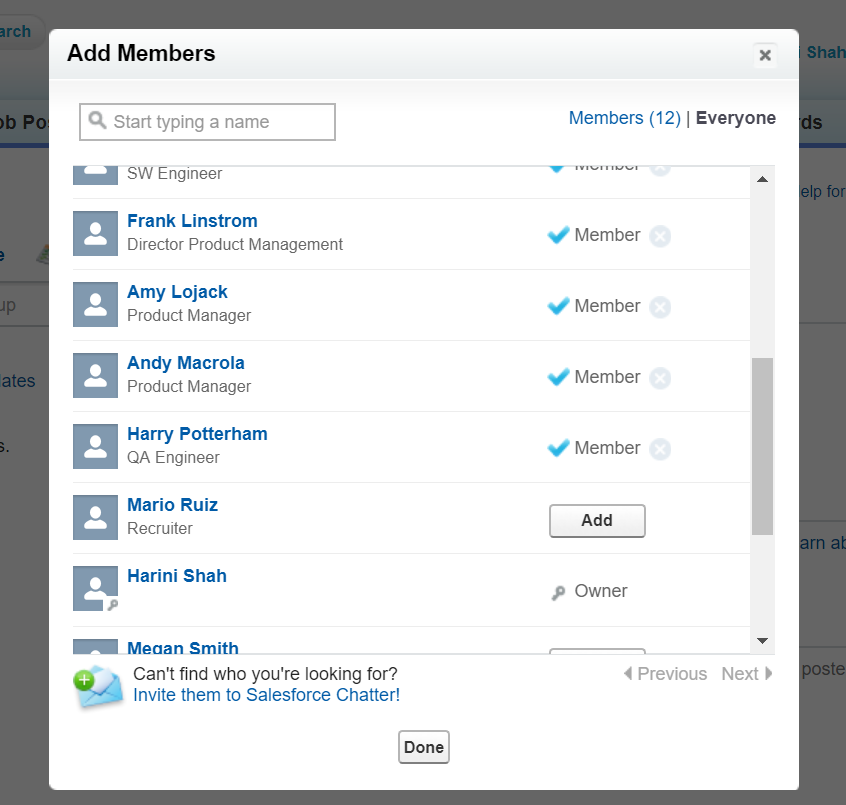
click on all universal containers

I am the only person in the group

So we created a new Group as UC Group, make it public. If we make it private we will have to approve all new members.

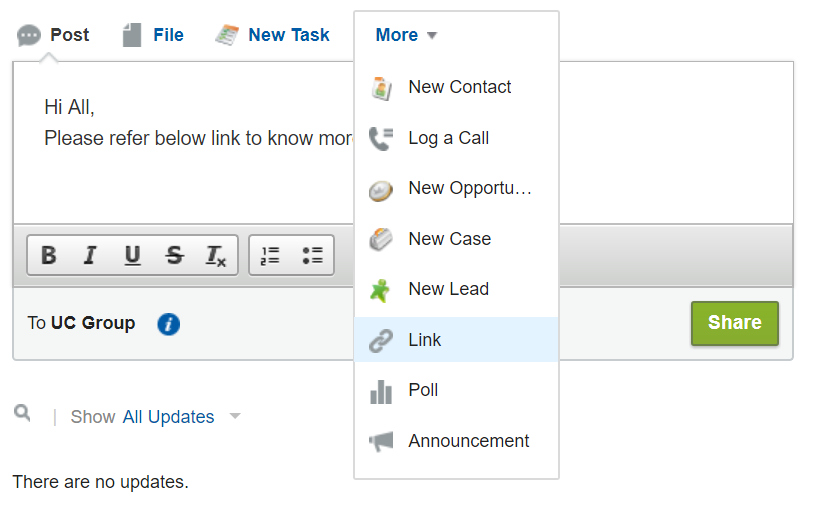


Use add members link and add all users.

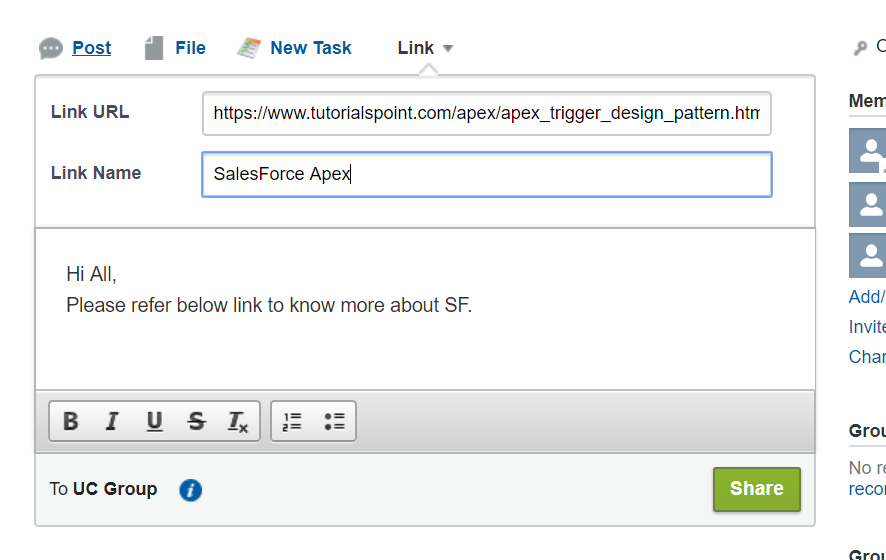


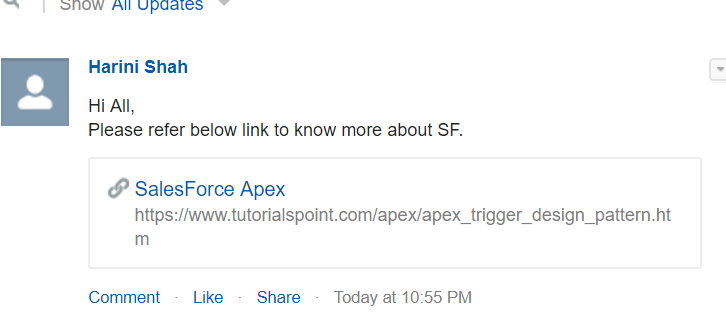
Click on change roles. We see manager check box.

Suppose if participants asks.. if I make Cynthia as manager. She will get privillages to add or remove people from group apart from system admin.. like whats app group admin can be added.



Take some link from google and add here.





Like this we need any new posting of job I need such post on chatter when the status of position is open.

We cant write many triggers on a object. So we move business logic to apexclass and call trigger from there.

1. Create a apex class first

PostionTriggerHandler – name of handler class

PositionTrigger-- name of trigger

We can replace the older code like this:

**PositionTrigger.apxt**

trigger PositionTrigger on Position\_\_c (before insert,before update) {

if(Trigger.isBefore){

if(Trigger.isInsert || Trigger.isUpdate){

PostionTriggerHandler.populateHello(Trigger.New);

}

}

}

PositionTriggerHandler.apxc

public class PostionTriggerHandler {

public static void populateHello(List<Position\_\_c> posList){

// Check whether Hello\_\_c contains 'WORLD', if not populate the same

// for all the position records

//Iterate through Trigger.New -- nothing but posList

for(Position\_\_c pos : posList){

if(pos.Hello\_\_c != 'World'){

pos.Hello\_\_c = 'World';

}

}

}

}

If we want to post something on chatter programmatically, we need the object associated.

In file go to open resource search for feeditem.obj.. this class allows you to post feed on chatter.

Explore the fields of this object.

Body- content u want to see as body

Link—to add hyperlink for position

parentId – where exactly will we post this post on (either on group or individual profile—example WHO)

Title—any title you give

Create a subject of feeditem and populate Parentid and body

To find parent id of the chatter

- open google- seach for chatter cheat sheet. Slideblast.com

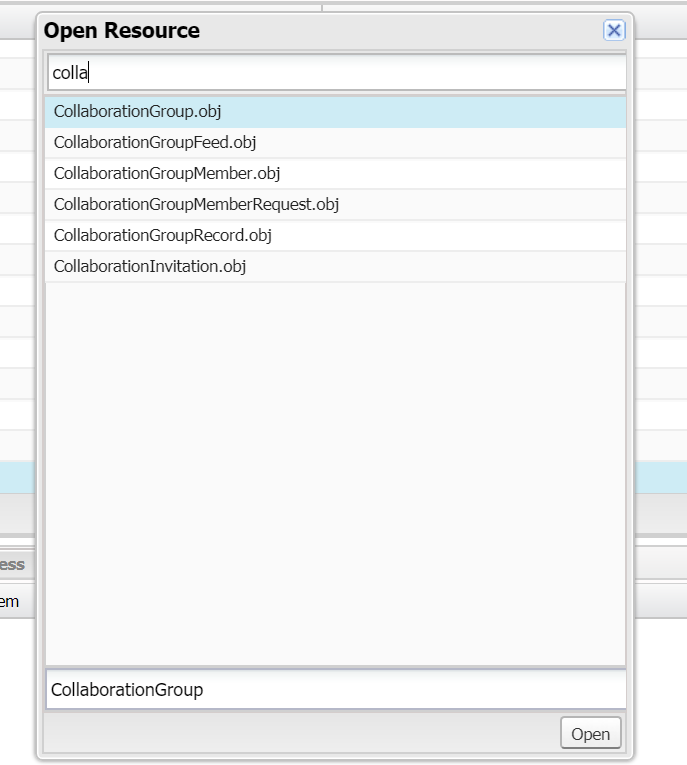
In this we have collaboration groups—gives us collaborationGroup.

We need to fire query on this to know what are all the chattergroups we have

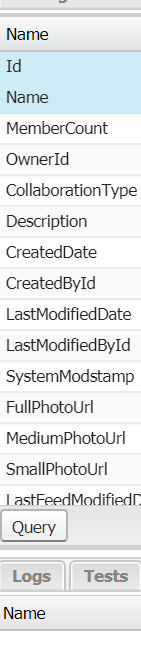
Feeditem object we need Body and ParentId(nothing but collaboration group id)

We get into collaboration group and look for chattergroup based on name and get the Id.

Find collaborationGroup object



Double click



Query

And check for ID

Copy this query and use on our code.

Also we take a feeditem list so that we don’t post in for loop instead we post only once.

For every position we don’t post we post together as a list.

Apex class:

public class PostionTriggerHandler {

public static void populateHello(List<Position\_\_c> posList){

// Check whether Hello\_\_c contains 'WORLD', if not populate the same

// for all the position records

//Iterate through Trigger.New -- nothing but posList

for(Position\_\_c pos : posList){

if(pos.Hello\_\_c != 'World'){

pos.Hello\_\_c = 'World';

}

}

}

//announce open positions on chatter groups

public static void postonChatterGroup(List<Position\_\_c> posList){

List<CollaborationGroup> ucGroupList =[SELECT Id, Name

FROM CollaborationGroup

Where Name Like 'UC%'];

//assigned to list so that if its empty it wont throw exception. but it will throw exception as null if its a singe record

List<FeedItem> fList = new List<FeedItem>();

for(Position\_\_c pos : posList){

//if(pos.Status\_\_c == 'Open' && !ucGroupList.isEmpty())

if(pos.Status\_\_c == 'Open' && ucGroupList.size()>0){

FeedItem fi = new FeedItem();

fi.Body=pos.Name+' has been Opened, Please refer';

fi.ParentId=ucGroupList[0].Id;

fList.add(fi);

}

}

//insert feed item list onky if atelast one record is there in list

if(fList.size()>0){

try{

insert fList;

}catch(DmlException ex){

System.debug('Probelm in insertion'+ex.getMessage());

}

}

}

}

And trigger is as

trigger PositionTrigger on Position\_\_c (before insert,before update, after insert, after update) {

if(Trigger.isBefore){

if(Trigger.isInsert || Trigger.isUpdate){

PostionTriggerHandler.populateHello(Trigger.New);

}

} else{

if(Trigger.isInsert || Trigger.isUpdate){

PostionTriggerHandler.postonChatterGroup(Trigger.New);

}

}

}

**Next Requirement:**

Create a job posting site record. With active state.

Populate reason for deactivation in the description field automatically, whenever a job posting in job posting site is deactivated(active to inactive state).

Workflow cant be done since it doesnot hold old value. Trigger.Old and Trigger.New can be used.

Even in process builder, we can check only current value but not previous values for a record.

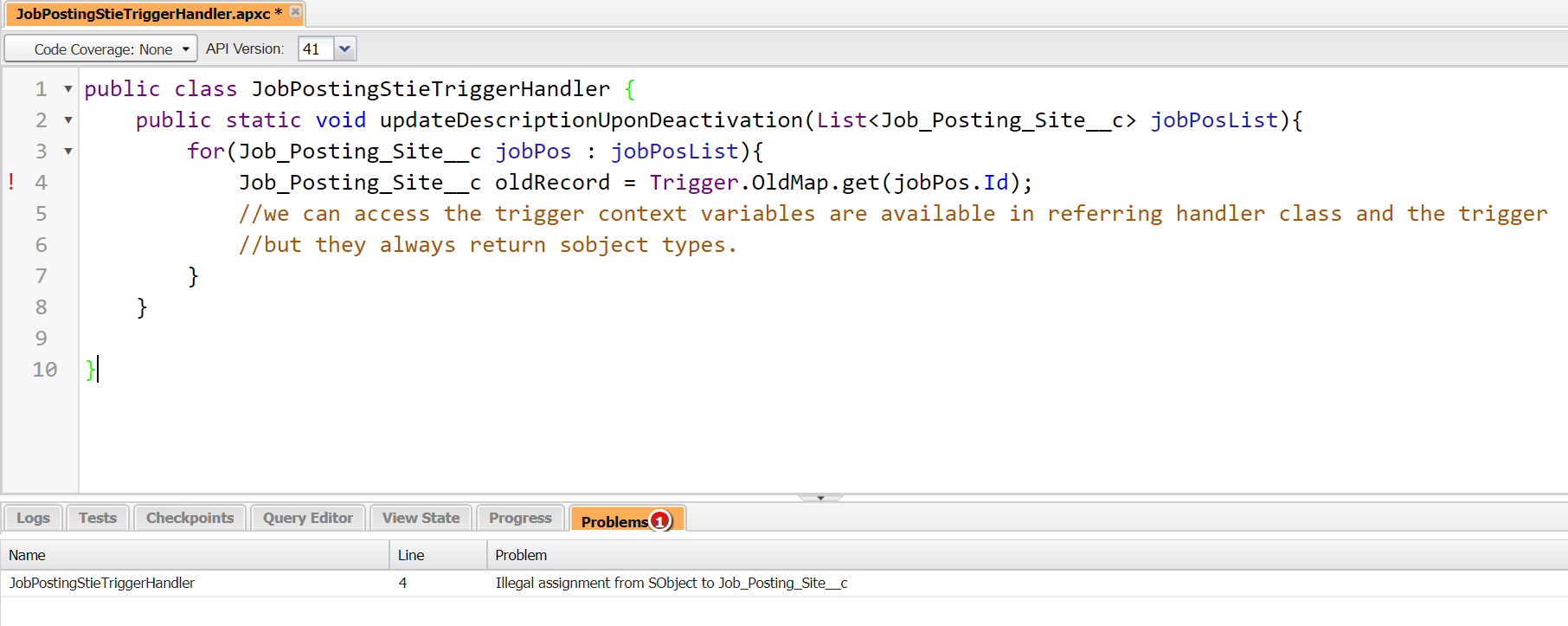
Trigger.oldMap and Trigger.newMap are the context variables used.

Insert will have only Trigger.New

Delete will have only Trigger.Old

Only in update will have both Trigger.Old and Trigger.New

Create a new apex class- JobPostingStieTriggerHandler



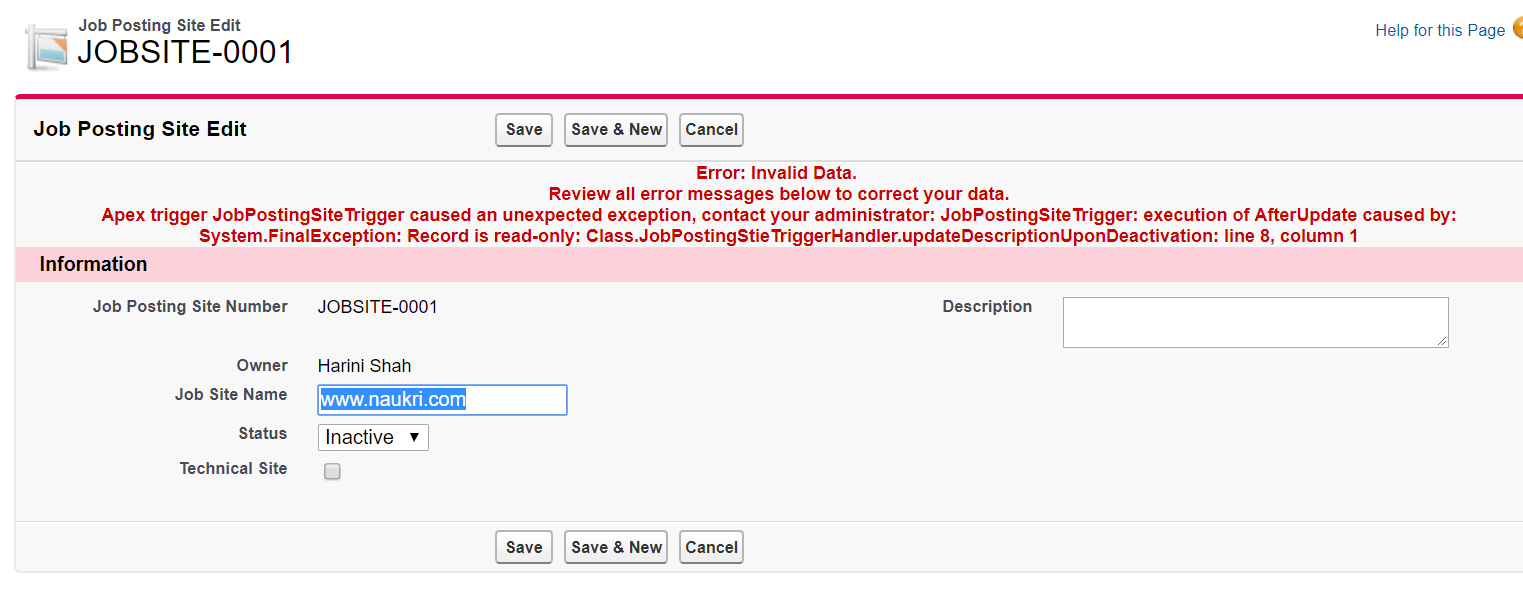
I get this error bcoz we need to typecast from subject to which ever ovject we want as the type casting is not done by framework.

Trigger always returns sobjects.

In the trigger we create the trigger of a object class. So it automatically casts with respect to the class. But here we need to do the same.

Job\_Posting\_Site\_\_c oldRecord = (Job\_Posting\_Site\_\_c)Trigger.OldMap.get(jobPos.Id);

Super type it will do automatically, but subtype we need to change.



Error we got since we had entered after Update while defining trigger

We are trying to update a record stored in database. And now we are trying to edit a readonly record.

trigger JobPostingSiteTrigger on Job\_Posting\_Site\_\_c (after update)

trigger JobPostingSiteTrigger on Job\_Posting\_Site\_\_c (before update) {

JobPostingStieTriggerHandler.updateDescriptionUponDeactivation(Trigger.New);

}

public class JobPostingStieTriggerHandler {

public static void updateDescriptionUponDeactivation(List<Job\_Posting\_Site\_\_c> jobPosList){

for(Job\_Posting\_Site\_\_c jobPos : jobPosList){

Job\_Posting\_Site\_\_c oldRecord = (Job\_Posting\_Site\_\_c)Trigger.OldMap.get(jobPos.Id);

//we can access the trigger context variables are available in referring handler class and the trigger

//but they always return sobject types.

if((oldRecord.Status\_\_c =='Active') && jobPos.Status\_\_c == 'Inactive'){

jobPos.Description\_\_c = 'As on'+System.now()+ 'this site has been deactivated';

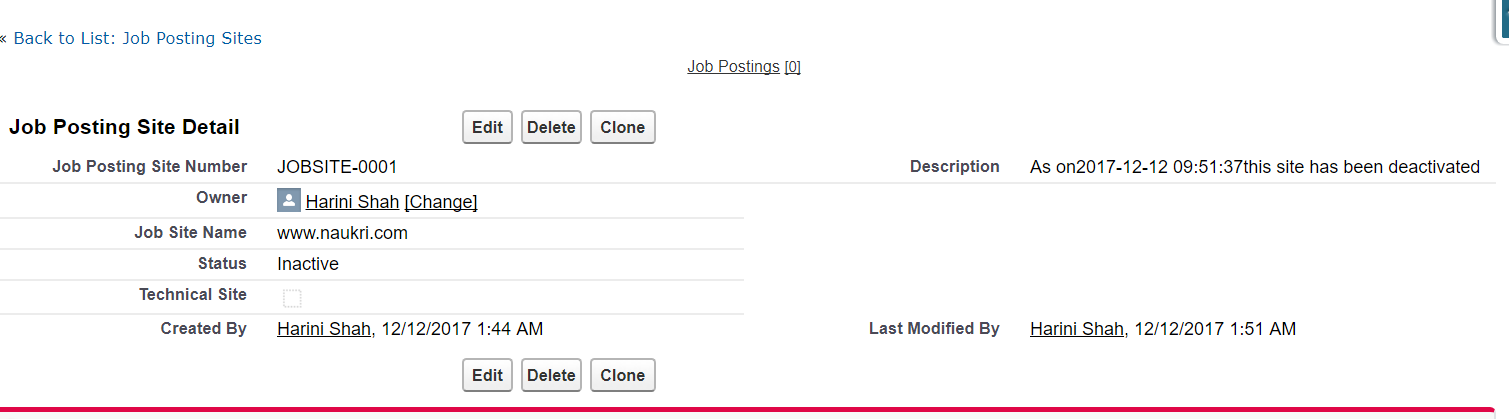
}

}

}

}

Now it works.

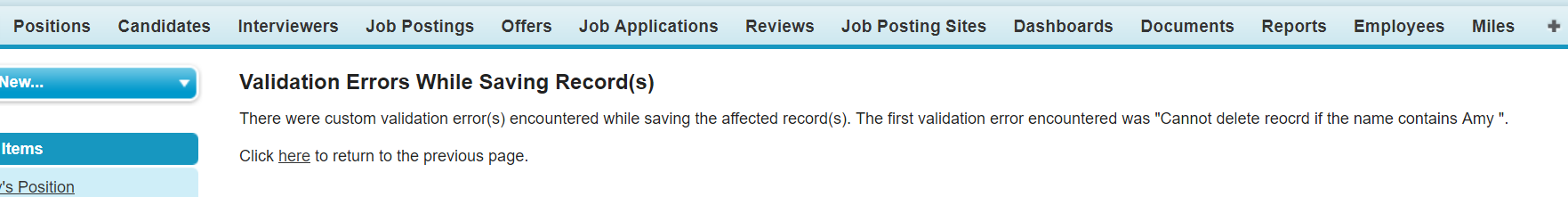


**Next Requirement**:

Prevent deletion of a position record if the name contains Amy

Handler class can have access to trigger context variables. But they retun sboject type and hence we downcast to required class.

addError stops deletion only this record. addError displays the error msg and skips this record deletion. On this record the deletion is terminated.



**Trigger file:**

trigger PositionTrigger on Position\_\_c (before insert,before update, after insert, after update, before delete) {

if(Trigger.isBefore){

if(Trigger.isInsert || Trigger.isUpdate){

PostionTriggerHandler.populateHello(Trigger.New);

}

else if(Trigger.isDelete){

PostionTriggerHandler.preventDeleteion();

}

} else{

if(Trigger.isInsert || Trigger.isUpdate){

PostionTriggerHandler.postonChatterGroup(Trigger.New);

}

}

}

**Code for prevention in the apex class:**

**//prevent deletion of psotion records where name contains amy**

**public static void preventDeleteion(){**

**List<Position\_\_c> posList = (List<Position\_\_c>)Trigger.Old;**

**//we never have access to Trigger.New while deleting**

**// they have access only to Trigger.Old, leads to exception if we make it New**

**for(Position\_\_c pos: posList){**

**if(pos.Name.Contains('Amy')){**

**pos.addError('Cannot delete reocrd if the name contains Amy ');**

**}**

**}**

**}**

**----------------------------------full file-------------------------**

**public class PostionTriggerHandler {**

**public static void populateHello(List<Position\_\_c> posList){**

**// Check whether Hello\_\_c contains 'WORLD', if not populate the same**

**// for all the position records**

**//Iterate through Trigger.New -- nothing but posList**

**for(Position\_\_c pos : posList){**

**if(pos.Hello\_\_c != 'World'){**

**pos.Hello\_\_c = 'World';**

**}**

**}**

**}**

**//announce open positions on chatter groups**

**public static void postonChatterGroup(List<Position\_\_c> posList){**

**List<CollaborationGroup> ucGroupList =[SELECT Id, Name**

**FROM CollaborationGroup**

**Where Name Like 'UC%'];**

**//assigned to list so that if its empty it wont throw exception. but it will throw exception as null if its a singe record**

**List<FeedItem> fList = new List<FeedItem>();**

**for(Position\_\_c pos : posList){**

**//if(pos.Status\_\_c == 'Open' && !ucGroupList.isEmpty())**

**if(pos.Status\_\_c == 'Open' && ucGroupList.size()>0){**

**FeedItem fi = new FeedItem();**

**fi.Body=pos.Name+' has been Opened, Please refer';**

**fi.ParentId=ucGroupList[0].Id;**

**fList.add(fi);**

**}**

**}**

**//insert feed item list onky if atelast one record is there in list**

**if(fList.size()>0){**

**try{**

**insert fList;**

**}catch(DmlException ex){**

**System.debug('Probelm in insertion'+ex.getMessage());**

**}**

**}**

**}**

**//prevent deletion of psotion records where name contains amy**

**public static void preventDeleteion(){**

**List<Position\_\_c> posList = (List<Position\_\_c>)Trigger.Old;**

**//we never have access to Trigger.New while deleting**

**// they have access only to Trigger.Old, leads to exception if we make it New**

**for(Position\_\_c pos: posList){**

**if(pos.Name.Contains('Amy')){**

**pos.addError('Cannot delete reocrd if the name contains Amy ');**

**}**

**}**

**}**

**}**

**------------------End of file----------------------------------**

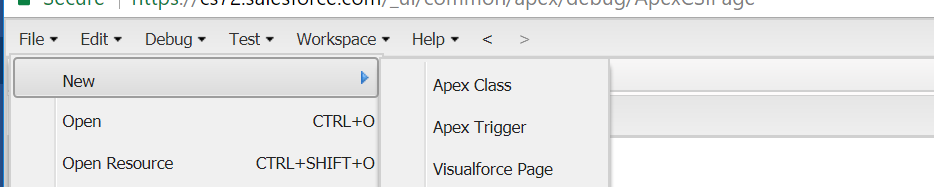
**Visual Force Pages:**

**In offer detail page we want to create a section with 4 columns to display “Quick Edit section” with “offer date”, “status”, “expiry” and a button “update”.**

**After this the offer detail is visible.**

Use VF since standard page layout can have only 2 sections here we need 4 sections.

Create a VFpage



Name it as offerdetailpage

Note: VF follows XML syntax, but will be internally converted to HTML

In this case we can use standard controller as update is nothing but save that is a standard action. So mention with <apex:Page standardController=” Offer\_\_c”> and mention api name of the object, in our case “Offer\_\_c”

Have one page block section and one input field.

On input filed we ned value that is using for binding with data.(data Binding)

The value we enter in text box should be associate with field.

Put this with {!} curly bang this is a sales force expression

<apex:inputField value="{!Offer\_\_c.Offer\_Date\_\_c}" />

Once we save it shows a text box with type date in the page.

Add other fields. By default we see 2 columns.

Now we add columns

<apex:pageBlockSection columns="4">

<apex:inputField value="{!Offer\_\_c.Offer\_Date\_\_c}" />

<apex:inputField value="{!Offer\_\_c.Offer\_Expiration\_Date\_\_c}" />

<apex:inputField value="{!Offer\_\_c.Status\_\_c}" />

</apex:pageBlockSection>

Now add button

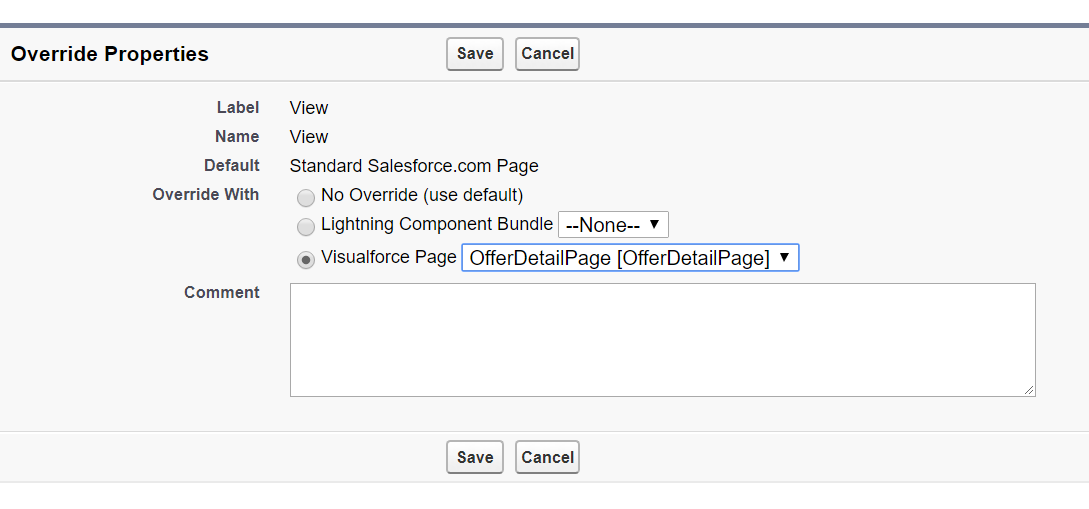
<apex:commandButton value="Update" action="{!save}"/>

Value is label and action is for action binding.

Now we should overriding the standard display.

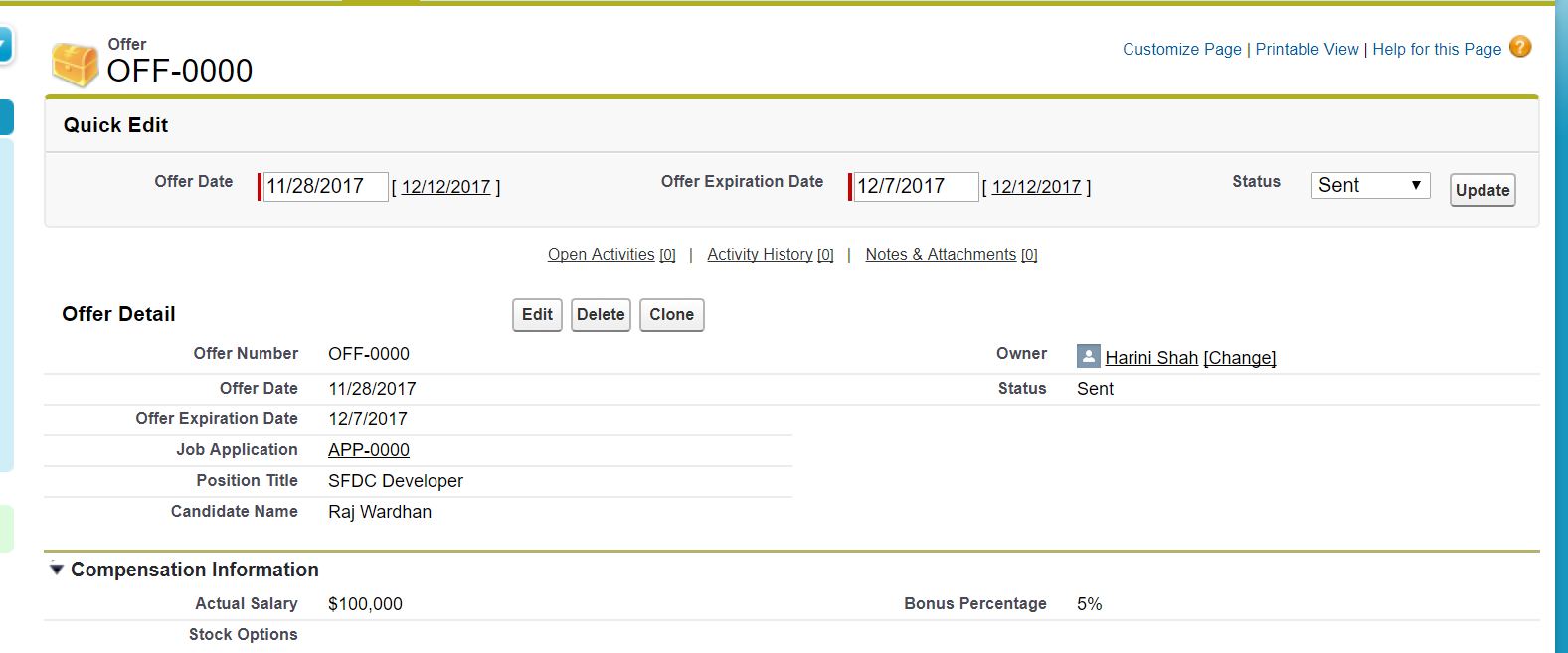
Go to object detail page for offer. Go to button links and actions.

Edit the View.



This will override the view and show only visual page and not original page.

So add <apex:detail> tags



This is what we wanted to achieve.

This is the code in the VF page

<apex:page standardController="Offer\_\_c">

<apex:form>

<apex:pageBlock title="Quick Edit">

<apex:pageBlockSection columns="4">

<apex:inputField value="{!Offer\_\_c.Offer\_Date\_\_c}" />

<apex:inputField value="{!Offer\_\_c.Offer\_Expiration\_Date\_\_c}" />

<apex:inputField value="{!Offer\_\_c.Status\_\_c}" />

<apex:commandButton value="Update" action="{!save}"/>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:form>

<apex:detail/>

</apex:page>

Now in case user says I don’t want to showcase all related lists. Only detail page but not related lists.

<apex:detail relatedList="false"/>

But now he is not happy since Job Application related list is removed.

But offer is a parent to job application, so job application can have offer as a related list but not vice versa.

So its not possible.

Instead we can have open activities

<apex:relatedList list="OpenActivities"/>

**======== full code===============**

**<apex:page standardController="Offer\_\_c">**

**<apex:form>**

**<apex:pageBlock title="Quick Edit">**

**<apex:pageBlockSection columns="4">**

**<apex:inputField value="{!Offer\_\_c.Offer\_Date\_\_c}" />**

**<apex:inputField value="{!Offer\_\_c.Offer\_Expiration\_Date\_\_c}" />**

**<apex:inputField value="{!Offer\_\_c.Status\_\_c}" />**

**<apex:commandButton value="Update" action="{!save}"/>**

**</apex:pageBlockSection>**

**</apex:pageBlock>**

**</apex:form>**

**<apex:detail relatedList="false"/>**

**<apex:relatedList list="OpenActivities"/>**

**</apex:page>**

**============end of code=============**

New example to see all the standard, custom and enhanced controlllers here.

Go to candidate page and try to read first name and last name.

<apex:pageBlockButtons location="top"> by default location is both for pageblock buttons.

We want to see a page with 2 save buttons one at top and one at bottom. Frist name and last name. in a section Information ands the page title is candidate edit

<apex:page standardController="Candidate\_\_c">

<apex:form>

<apex:pageBlock title="Candidate Edit">

<apex:pageBlockButtons>

<apex:commandButton value="Save" action="{!save}" />

</apex:pageBlockButtons>

<apex:pageBlockSection title="Information">

<apex:inputField value="{!Candidate\_\_c.First\_Name\_\_c }"/>

<apex:inputField value="{!Candidate\_\_c.Last\_Name\_\_c}"/>

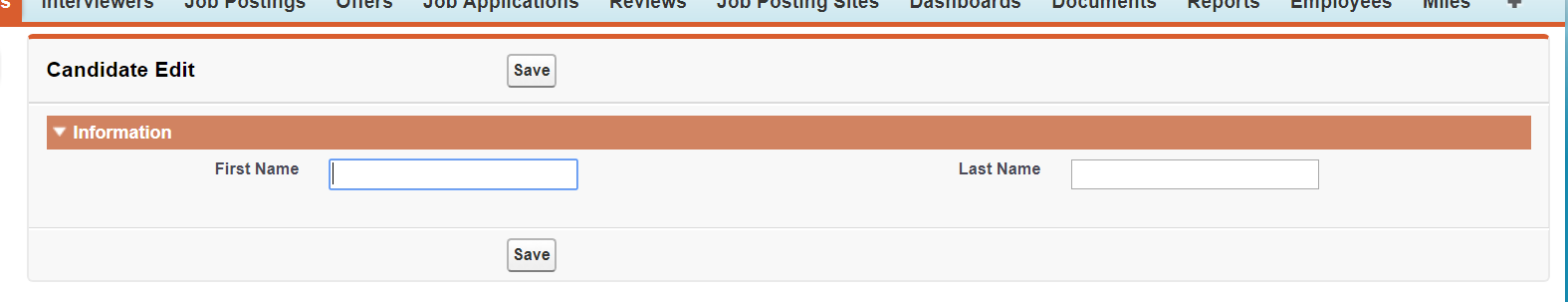
</apex:pageBlockSection>

</apex:pageBlock>

</apex:form>

</apex:page>

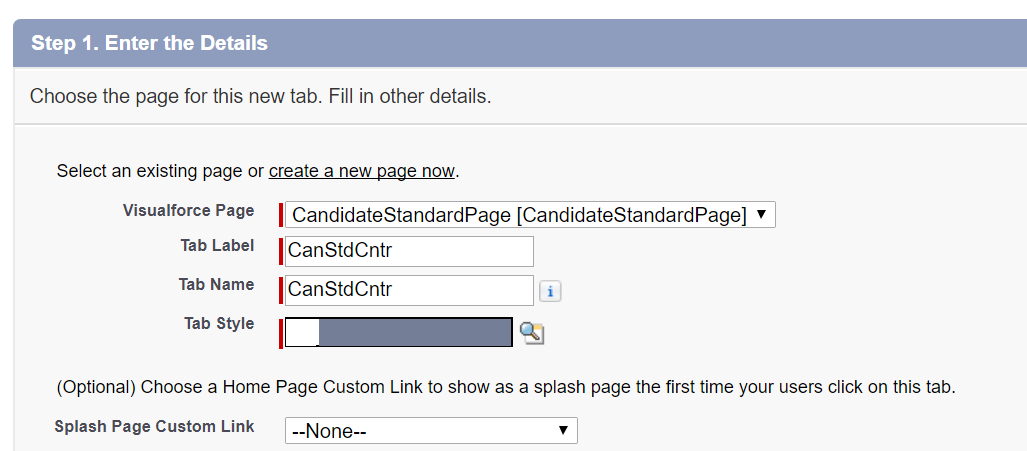
Yields to



Now either we override a button or create a tab.

In this case we will create a tab.

Go to set up-- create a tab—create a tab for visual force page- click new.



Save.

But the tab style is not taken from here. But the style is taken from the default object(candidate) style.

In order to use our own style, we go to tab. Copy tab name =CanStdCntr

(All tabs have extension \_\_tab like \_\_c for pages)

So go to visualforce page in developer console add tabstyle

<apex:page standardController="Candidate\_\_c" tabStyle="CanStdCntr\_\_tab">

And the page takes the color theme selected in tab style.

When we saved the candidate—internally the visual force engine has created the instance, populated the values and inserted the record.

When we create custom we should take care of all the above.

So we will create our own apex class that is our controller.

Lets create a apex class. Name: CandiateCustomController

1. Create a candidate reference.
   1. public Candidate\_\_C candidateRef {get; set;}
   2. get and set are for getting and setter taken from c#. this works only with VFpage associated apex class and not for normal apex class.
   3. In earlier case as soon as page loads constructor is invoked. When we save setter is triggered. So we are understanding and doing the same steps.
   4. define constructor to create a candidate instance (Sobject) since ref we created will be null by default
   5. define an action method to perform insertion
   6. Thumb Rule: all action methods must have return type as PageReference.
   7. PageReference is--- page ref is referring the view where you want to take the user once the action is over.
   8. Like quick save on page layout page—it stays on same as page as page ref is null. But save will take me to record page back so that’s the page reference returned in save.

public class CandiateCustomController {

// Define a reference variable for Candidate\_\_C along with setter and getter

// when we write setter and getter it should always be public and not private

// if we are using any properties that particular attibute must be public

public Candidate\_\_C candidateRef {get; set;}

//define constructor to create a candidate instance (Sobject) since ref we created will be null by default

public CandiateCustomController(){

candidateRef =new Candidate\_\_C();

}

//define an action method to perform insertion

//Thumb Rule: all action methods must have return type as PageReference.

public PageReference mySave(){

try{

insert candidateRef;

}catch(DmlException ex){

System.debug('Problem in insertion'+ ex.getMessage());

}

PageReference ref = new PageReference('/'+candidateRef.Id);

return ref;

}

}

1. Create a visual force page- CandidateCustomControllerPage copy code from previous controller page. Now edit.

When we have our apex class that is acting as controller.

Here we edit the standard controller to controller and give the name of our apex class.

<apex:page Controller="CandiateCustomController">

<apex:form>

<apex:pageBlock title="Candidate Edit">

<apex:pageBlockButtons>

<apex:commandButton value="Save" action="{!mySave}" />

</apex:pageBlockButtons>

<apex:pageBlockSection title="Information">

<apex:inputField value="{!candidateRef.First\_Name\_\_c }"/>

<apex:inputField value="{!candidateRef.Last\_Name\_\_c}"/>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:form>

</apex:page>

Now we will move towards Extension controller.

We will have all standard actions and also add custom actions.

In this instance will be created by the engine and we can use it.

Everything will be created by the engine.

We want to insert in out manner so we do the insert.

Go to vf page of standard controller copy code. candidateextensioncontrollerpage

Create a new VF page. Paste code.

We have standard code and we start customising now.

Lets create a apex class candidateextenstionscontroller

public class CandidateExtensionController {

Candidate\_\_c candidateRef;

//All extension controllers must have a parameterised constructor

//which takes StandardController type as parameter

public CandidateExtensionController(ApexPages.StandardController sc){

candidateRef = (Candidate\_\_c) sc.getRecord();

}

public PageReference mySave(){

try{

insert candidateRef;

}catch(DmlException ex){

System.debug('Problem in insertion'+ ex.getMessage());

}

PageReference ref = new PageReference('/'+candidateRef.Id);

return ref;

}

}

**Controller Visual page:**

<apex:page standardController="Candidate\_\_c" extensions="CandidateExtensionController">

<apex:form>

<apex:pageBlock title="Candidate Edit">

<apex:pageBlockButtons>

<apex:commandButton value="Save" action="{!save}" />

<apex:commandButton value="MySave" action="{!mySave}" />

</apex:pageBlockButtons>

<apex:pageBlockSection title="Information">

<apex:inputField value="{!Candidate\_\_c.First\_Name\_\_c }"/>

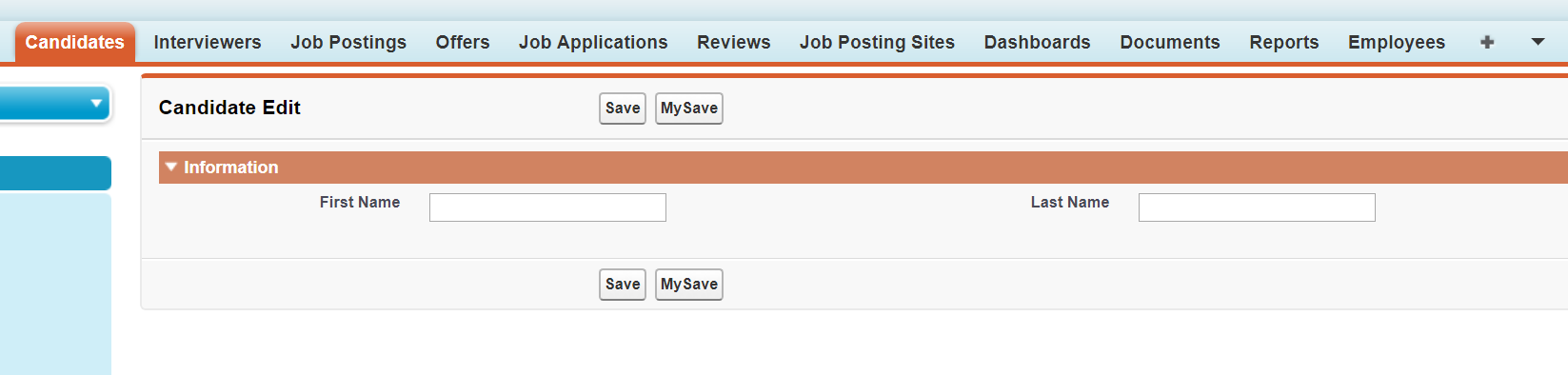
<apex:inputField value="{!Candidate\_\_c.Last\_Name\_\_c}"/>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:form>

</apex:page>



This is how it looks now with 2 buttons.

**Actual Requirement:**

Email

Last Name

First name

Search

Search Text

**Search a name and display where ever it Appears. THEN we add the edit button later.**

**Create new apex class : CandidateSearchController**

**public class CandidateSearchController {**

**public String SearchText {get; set;}**

**public List<Candidate\_\_c> candidateList {get; set;}**

**//action method to fetch candidatesm we are using SOSL and hence use Find and Returning**

**public PageReference doSearch(){**

**List<List<sobject>> records = [FIND :SearchText**

**RETURNING Candidate\_\_c(First\_Name\_\_c,**

**Last\_Name\_\_c,**

**Email\_\_c)];**

**candidateList =records[0]; //List of List of sobject, so records will be of candidate**

**return null;**

**}**

**}**

**Careata a vF page :** **CandidateSearchPage**

Inputfield comes to existance with only object properties like first name, last name etc..

Here we need to use input text since it’s a attribute to searchtext variable and not property of an object so we use inputtext.

Then call the method to search using a command button do search.

Now the records are candidateList so iterate through the page and display.

3 ways of iteration in VF pages:

1. Page block table – built in table structure will be there – predefined structure
2. Data table – cell padding cell spacing border etc we can specify if we use this—custom structure
3. Repeat

Vaule is candidaelist in pageblocktable and can is a variable that holds one record for iteration.

Addition of ajax is also taken care in code.

<apex:page controller="CandidateSearchController" tabStyle="CandidateSearch\_\_tab">

<apex:form>

<apex:pageBlock>

<apex:pageBlockSection title="Search Candidate">

<apex:inputText label="Enter Search Text" value="{!SearchText}"/>

<apex:commandButton value="Do Search" action="{!doSearch}" reRender="cd"/>

<!--reReder to identify what to refresh-->

</apex:pageBlockSection>

<!--id to identify what to refresh --implementation of ajax-->

<apex:pageBlockSection title="Candidates Details" id="cd">

<apex:pageBlockTable value="{!candidateList}" var="can">

<apex:column value="{!can.First\_Name\_\_c}"/>

<apex:column value="{!can.Last\_Name\_\_c}"/>

<apex:column value="{!can.Email\_\_c}"/>

</apex:pageBlockTable>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:form>

</apex:page>

By default only system admins only will have access to VF pages No other users can have the same by default.

So when we logged in as amy we were not able to view anything.

On offer we give access to hiring manager and recruiter.

On candidate search we gave access to hiring manager, recruiter and interviewer.

Also with candidate as amy we are not able to see candidates in candidates in tab, but in search we get 2 candidates.

Here owd of candaidate is private (sharing setting)

Owd is not applied on our class. hence the trouble.

To fix this change the apex calss as

public with sharing class CandidateSearchController {

when we make the class with sharing the class respects OWD model. By default its without sharing.

In this case go to candidate and open a candidate record and select sharing to manually share a record with one user.

So the candidate search works with this user only for that particular candidate.

**List Controllers**

**Create a Vf page: AccountListController.**

**Since we see like 1200 accounts we have we don’t see pagination.**

**So to create the same we are doing this.**

**When we have a recordsetvar in the controller page definition it’s a list controller**

**First last next prev are the additional actions available on list controller. Standard controller have only 6.**

**<apex:page standardController="Account" recordSetVar="accList">**

**<apex:form>**

**<apex:pageBlock title="All Acoounts">**

**<apex:pageBlockButtons>**

**<apex:commandButton value="First" action="{!first}" reRender="rs"/>**

**<apex:commandButton value="Prev" action="{!previous}" reRender="rs"/>**

**<apex:commandButton value="Next" action="{!next}" reRender="rs"/>**

**<apex:commandButton value="Last" action="{!last}" reRender="rs"/>**

**</apex:pageBlockButtons>**

**<apex:pageBlockSection id="rs">**

**<apex:pageBlockTable value="{!accList}" var="acc">**

**<apex:column value="{!acc.Name}"/>**

**<apex:column value="{!acc.Type}"/>**

**</apex:pageBlockTable>**

**</apex:pageBlockSection>**

**</apex:pageBlock>**

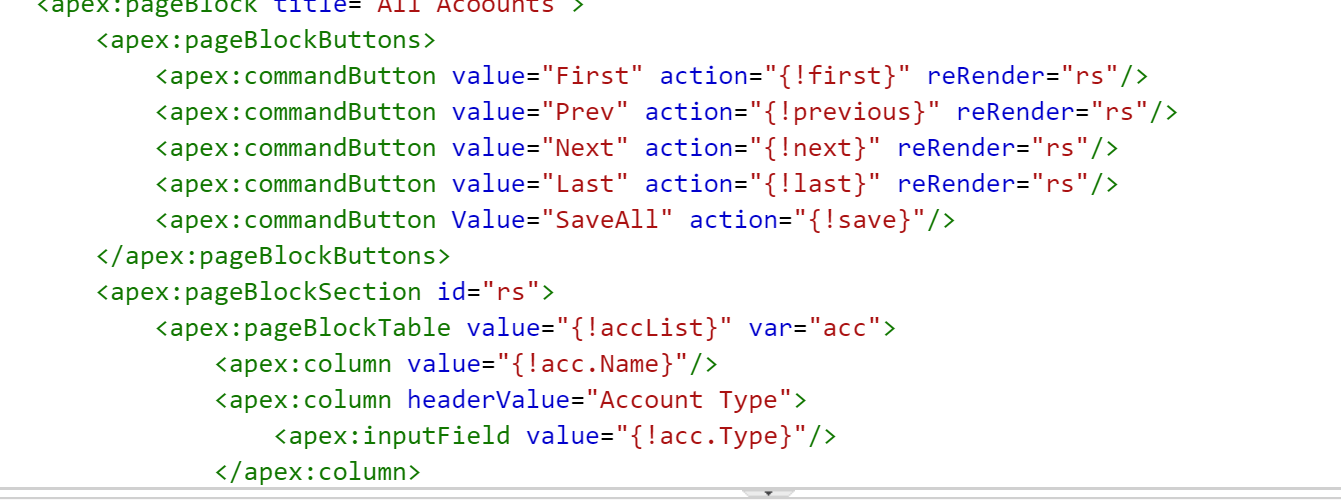
**</apex:form>**

**</apex:page>**

**We did this example on a standard object just to learn. But we cant set the same on the this tab as it’s a standard tab.**

**We can do this on any custom tab if we have 50-100 records atleast.**

**New:To edit multiple records in one go check the below change on save all for a button and infput field.**



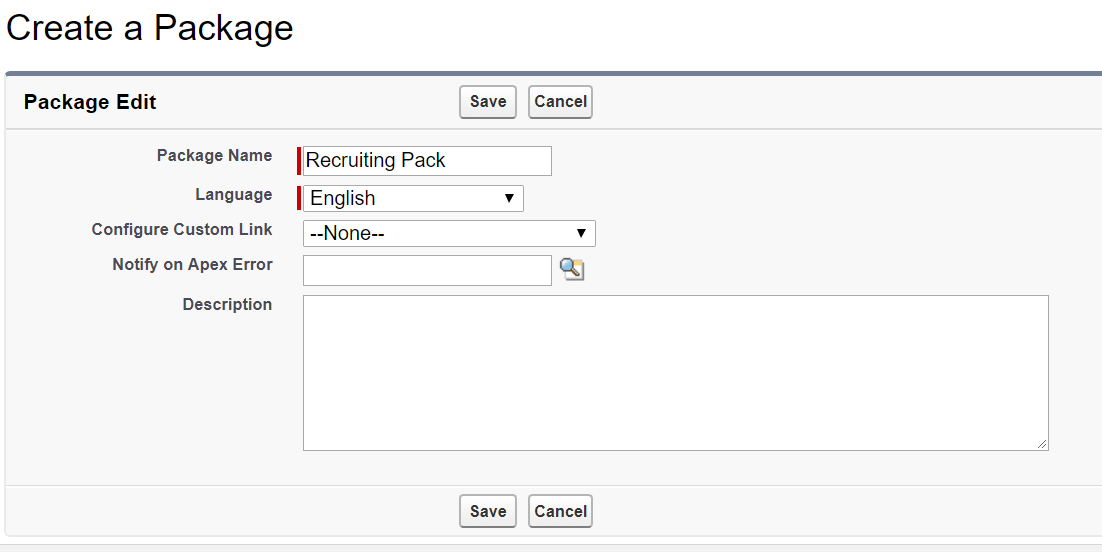
**Lightining:**

1. **Lightning experience**
2. **Salesforce 1 – app meant for mobiles**
3. **Enable my domain**
4. **Start developing lightning components**

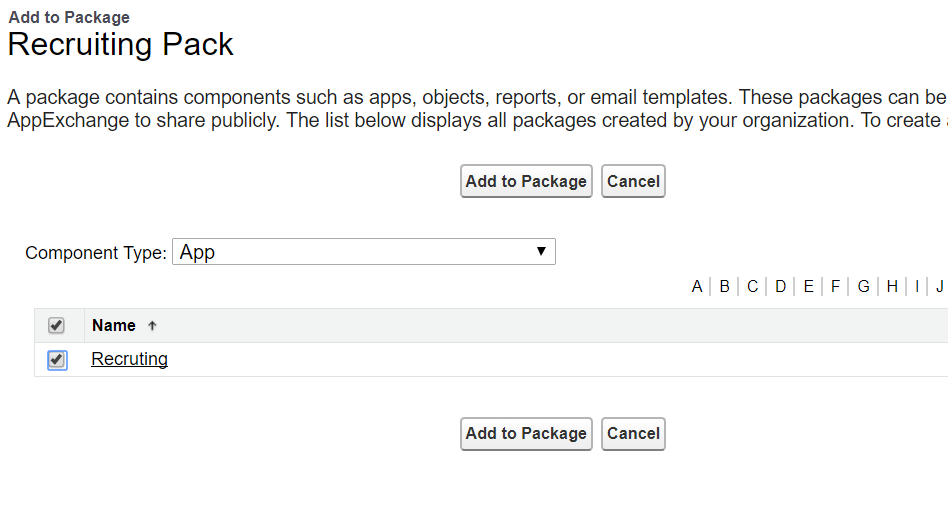
**So far what we used is salesforce classic.**

For packagaing:

Quick search package-> new package

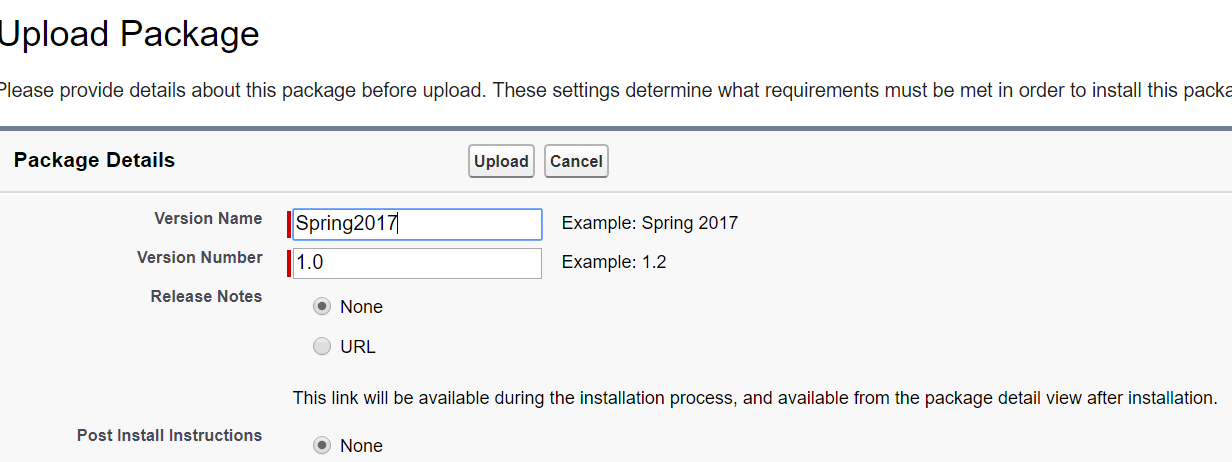


Add components:



All that we have in the recruiting will be there

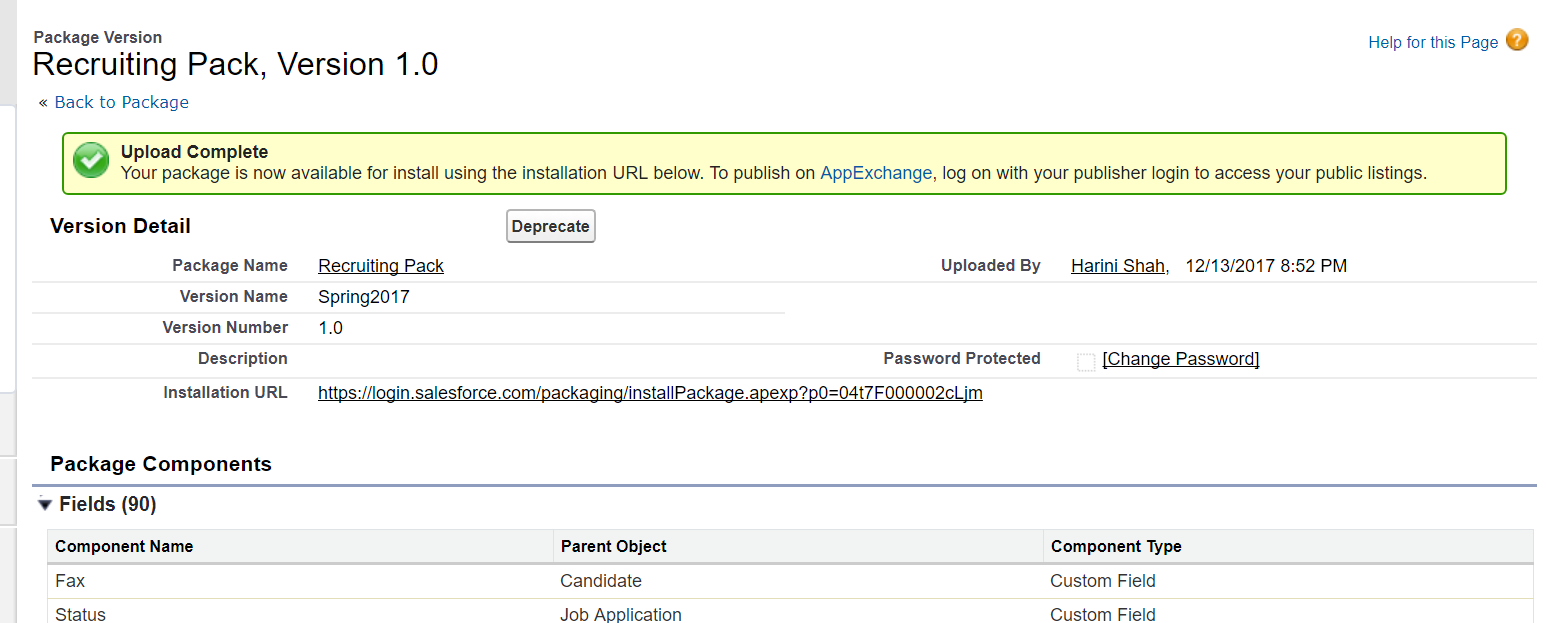
Click on upload



Don’t put any filters. Lets add everything. Click on Upload.

Suppose we have a class, we must have a Testclass with 75% test coverage.(any package that we create anywhere in prod/sandbox anywhere, we need 75% test coverage)

Hence we deleted the class we had here.

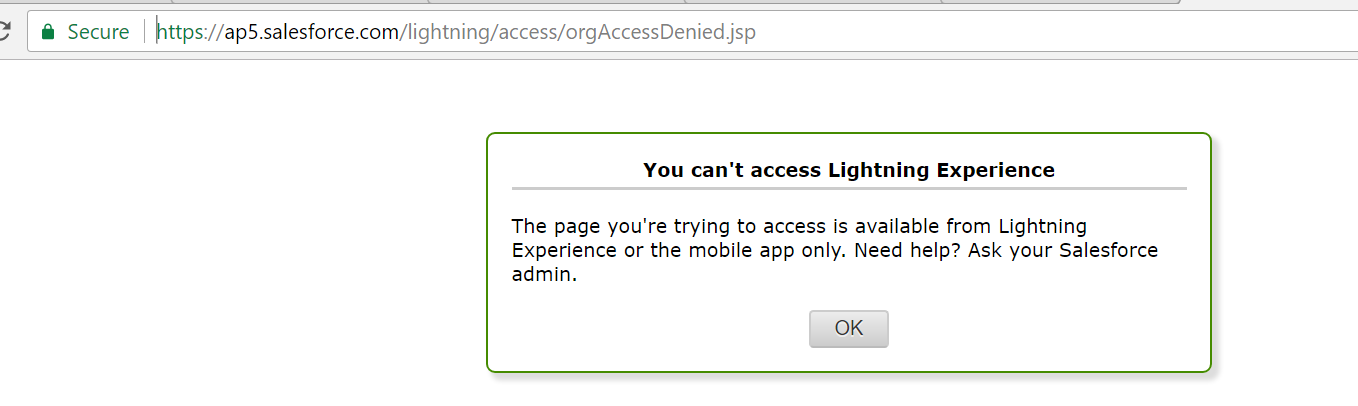


The link here is link to pour package.

[https://login.salesforce.com/packaging/installPackage.apexp?p0=04t7F000002cLjm](https://urldefense.proofpoint.com/v2/url?u=https-3A__login.salesforce.com_packaging_installPackage.apexp-3Fp0-3D04t7F000002cLjm&d=DwMCaQ&c=eIGjsITfXP_y-DLLX0uEHXJvU8nOHrUK8IrwNKOtkVU&r=VCTD03rK1Ug7DEGKmRbYrX-ISwPyf0oxoo3b2AqT0RNX4cOkAczcWHp2Z5329HMJ&m=cNouw7eDZCPEEB8atLbhuOj_B2KI4sQkeFnvuMzVPGA&s=x1SFXXotSPB7eH-U8T12HGobdSIzQ3LoOJKDl_gIVAg&e=) is my package.

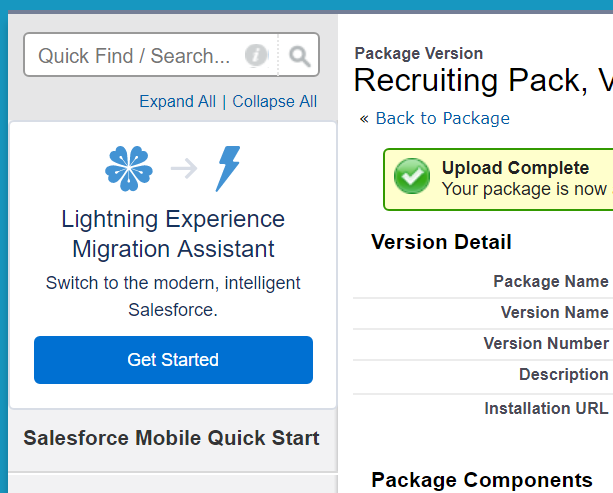
To access lightning

<https://ap5.salesforce.com/one/one.app>



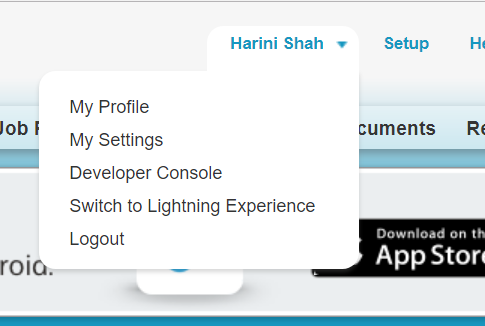
Errors.

So now new way



Go to turn it on tab and use toggle button and turn it on.

Now under my name I can see switch to lightining experience



Now we can select this.

Click on position. List views can be saw there.

Select All. Then u can see all positions. Right side u can see a drop down with Edit, change owner etc.

If you edit, u will get a Edit Modal. And when u save. We see a msg. its called Show Toast.

Even position record detail page looks different.

Shows details and related.

 go to set up icon, and go to set up.

Here the details look very different.

Advantage. When we click on object manager, it shows all object including object and custom.

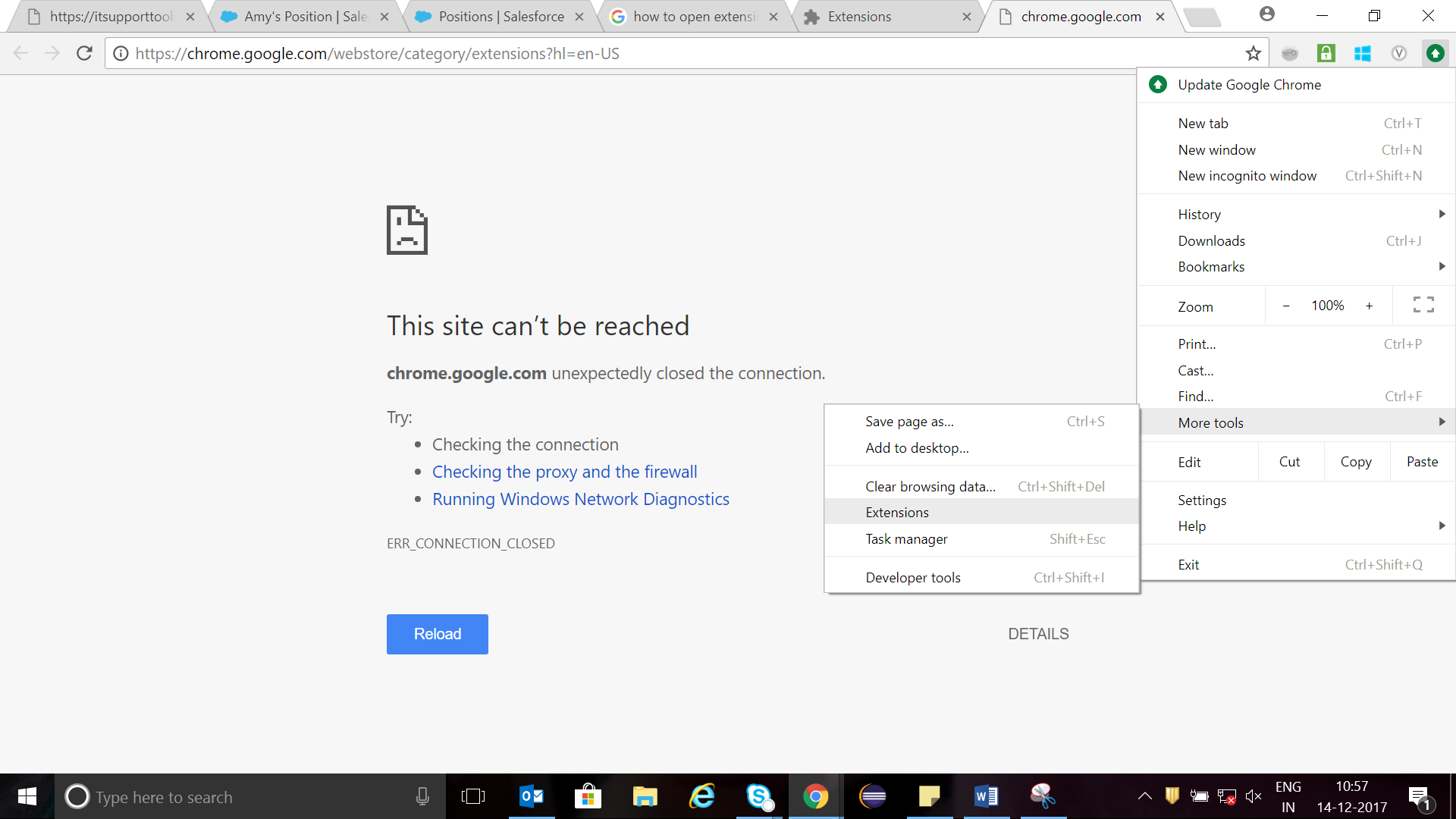
All objects are in one place.

If u click on any object, u can see related list in left panel and details in the big centre pane.

It provides Same UI to custom and standard object.

Salesforce1 app we will do simulation in web.

Go to extensions in google chrome, get more extensions: search for “Salesforce1 simulator”



Get more extensions: search salaesforce simulator

Or

Go to google and search salesforce simulator. And add that in to google extensions.

My Domain

In lightning development we have Lightning application and lightening component.

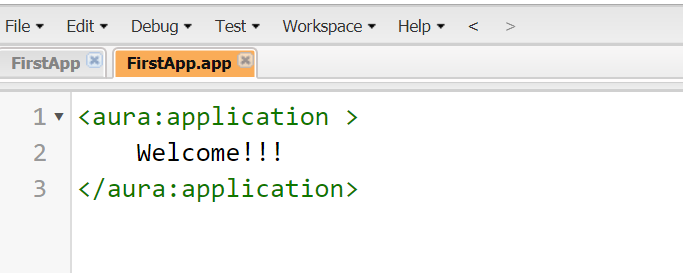
Open the developer console from lightning.

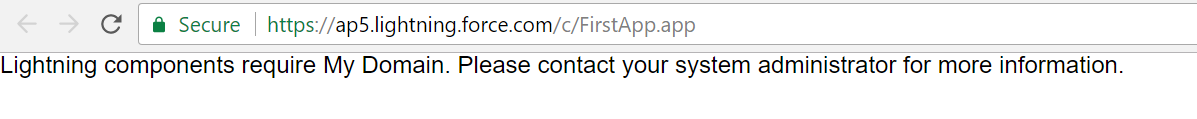
File- new- lightning application

Namr: FristApp

Submit

This is under aura framework.

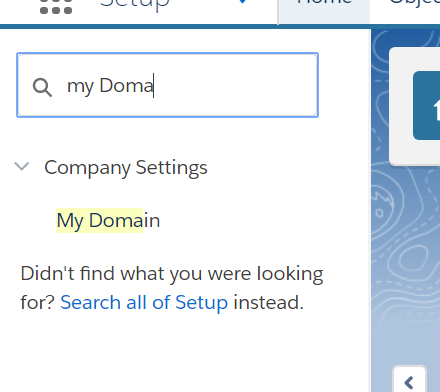




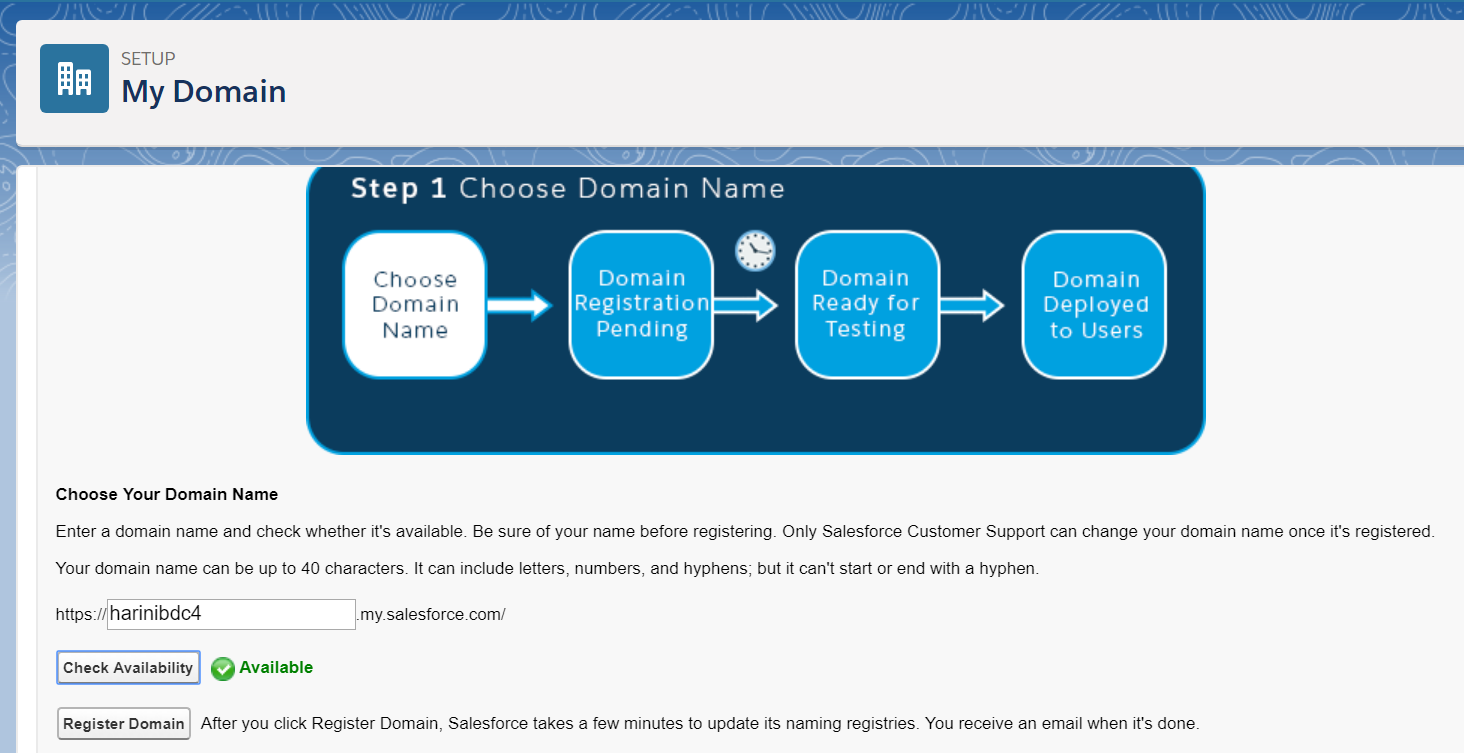
This is expected.

So mydomain is needed before getting into lightning.

Go to set up, setup, search my domain.



1. Set up a custom domain.



1. Register
2. Refresh the page we should see login. – open
3. Now a new page loads with my domain. https://harinibdc4.my.salesforce.com/domainname/DomainName.apexp
4. Click on deploy to users.
5. Now we have a domain on which we will work and we are protecting lightning components.
6. Sometimes we get error msg for preview. We need to delete the application and recreate the same. The problem is null in url instead of .app

Lightning helps us develop applications for both desktop and mobile.

Here we have lightning application(standalone application i.e is not reusable) and a lightning components ( that is re usable)

In a vf page there is no segregation. We can use style for css, script for java script.

But in lightning I can have bundles. The right side we see in a small block with preview is the bundles.

Application—ui related things, buttons etc..

Controller – javascripts, event actions, business logic, (controller.js)

Style- css related things apply it on application, uses aspect oriented programming from spring. autowiting also (style.css)

Documentation – documentation abt the application -java docs kind of, documents for developers regarding the application

Helper—repeated logic are stored here. And from here we use it in controller. (Helper.js)

Renderer- rerendering the page related controls. In case of using mobile applications( we may copy paste or read and type otp) may be we want to stop the page from refereshing (pull to refresh). This is just a example.

SVG - this is scalar vector graphics to use simple icons.

Now Create a new lightning component: in developer console.

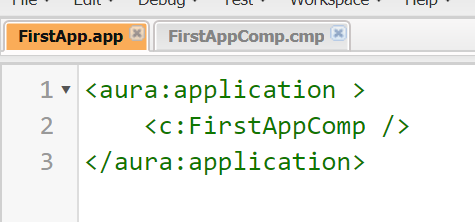
We cannot have component and application with same name.

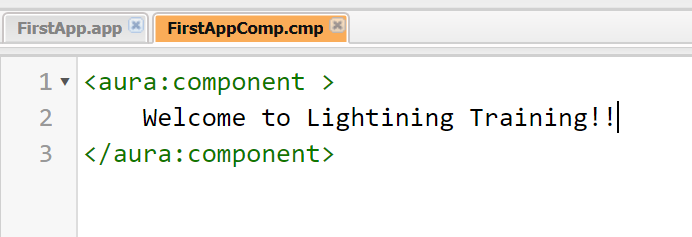
App has extension app, component has comp

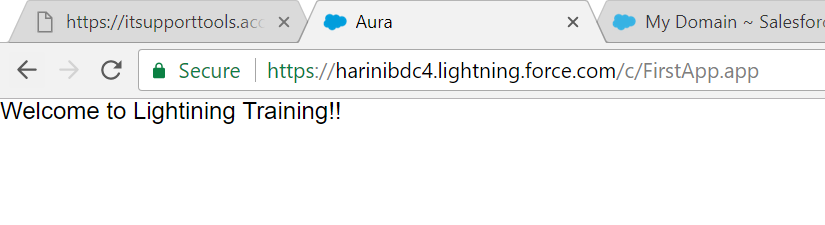
Here we see aura:component firstappcomp is our custom component.

In component we see additional thing in bundle, “design”

In application we need to use c: as all our applciations will be in c name space.







We can resue the component just by calling using the same command.

<aura:application >

<c:FirstAppComp /><br/>

<c:FirstAppComp />

</aura:application>

There are 3 Value providers:

V notation – represents view, value port, used to access attributes.

C notation – used for controller.js functions

Global value providers.

From the app we can access the attributes of the component. ( from anywhere I can refer the component, we can give value to a attribute).

{!v.msg} is just accessing the attribute to display the attribute.

Req:

Create a component called calculator and it should have 2 attributes.

We cant have attribute name with 1 character like x,y it should be min 2 characters num1 num2 type

{!} is used to evaluate expression.

Salesforce lightning design system(SLDS)—is available by default in salesforce1 and lightning experience but not available for standalone application. To get the SLDS to our application we add this extends="force:slds".

In google refer to salesforce lightning design system

https://www.lightningdesignsystem.com/icons#utility

variant also has few values that we can use:

<https://harinibdc4.lightning.force.com/auradocs>

(mydomain/auradocs)

Go to references

Left side we will see applications—in that expand under c it will show our apps

Expans componets-> lightining->buttons all description about buttons is here

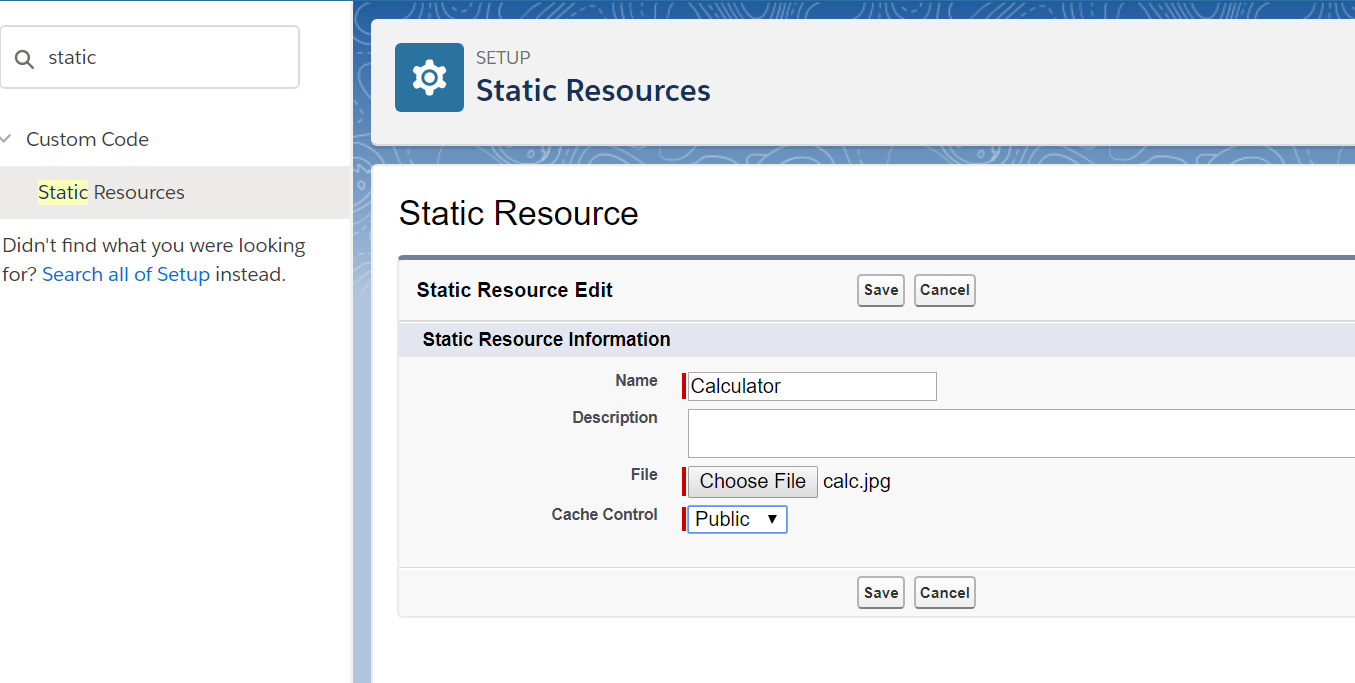
1. Create component: we can create a tab for component or surface it through app, or u can keep the component on page or vf page.
2. Create app test basic working, app is just referring the component(surfacing), so that we can access. App is container where we are keeping the component.
3. Create controller for dynamic behaviour, in json format
4. Create style-- .THIS represents component. .THIS.mystyle will be our css. .THIS is always capital.

Controller should be under component bundle.

Html5 v/s lightning- if my data type is email, lightning:input type=email, automatically the pad(key board) will come with builtin email features. If the type is integer, it will come with numbers.

If we want to use any image or anything else, we should use static resource.

In set up search – static resource.



<img src="{$Resource.Calculator}" /> is used to place the image in component file.

$ is global value provider.

.THIS {

When I want to apply stylesheet on child components we use a space “ “

Myborder is parent and rest 2 are child div’s so we used space here. .THIS .myResult and .THIS .myStyle

}

.THIS .myStyle{

width : 30%;

}

.THIS .myResult{

color :RED;

font-size : 20px;

}

.THIS.myborder{

width :50%;

border : 5px SOLID RED;

}

We cant see this component on a tab until unless we have a interface on it.

**Case study on opportunity:**

Server Side programming:

Create a apex class oppurtunitycontorller.

public class OpportunityController {

@AuraEnabled

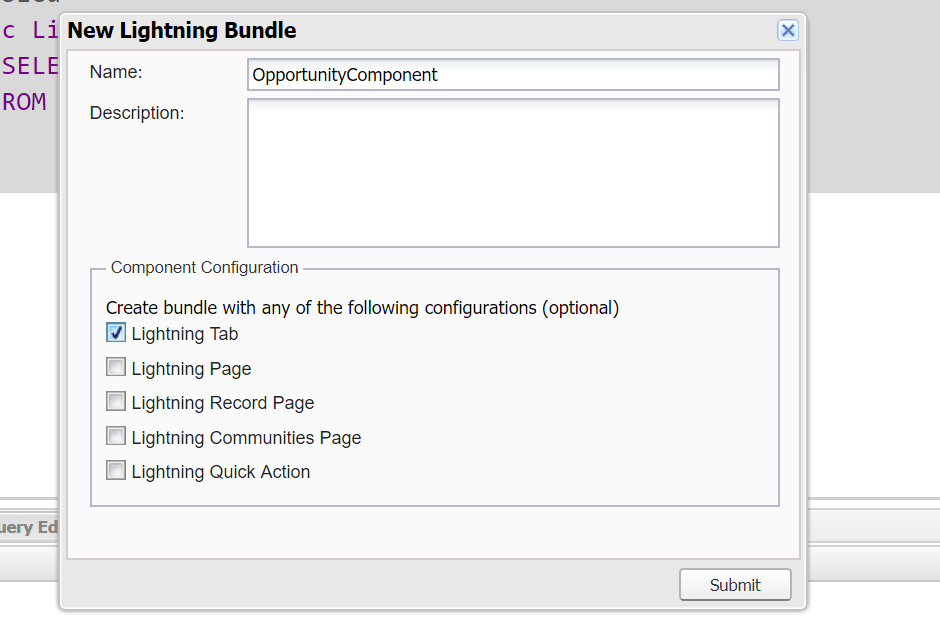
public static List<Opportunity> getAllOpportunities(){

return [SELECT Name, Amount, StageName

FROM Opportunity];

}

}



Component function name and controller function name should not be same. getAllOpps and getAllOppurtunity

<aura:component implements="force:appHostable" >

</aura:component>

When a component implements this interface we can have a tab else not

Associate this component with a controller

Creating a tab on a lightning component is called surfacing lightning components.

Surfacing is nothing but placing and referring a lightning component.

1. Standalone component
2. Using a lightning Tab(force:apphostable is the interface – implement the interface)
3. Paste it on lightning page (flexipage:availableForAllPageTypes)
4. We can place it on visual force page (for the clients who want to yet go with classic view rather than Lightning experience(Lex) we go for this) placing lightning component on VF page. We cant refer component directly, we need to create a instance of a lightning component. We need some intermediary.

Creating a lightning page, on top I want to see recent items(standard components – 3-5 recent records), then calculator and then opportunity. One below the other.

In set up search—go to lightning app builder- new- AppPage

AllCompsPage

**Points to know**

The advantage of lightning page is that we can have custom as well as Standard components.

Normal custom tab is associated with one object (1-1 realtion) in lightning tab, one tab can refer only one lightning component.

For 10 components I don’t need 10 tabs. We can create a component that is a container that acts as a container for all other components.

(ContainerComp that we created)

A custom component can refer only other custom components.

So a page associated with this can have only custom components. So we go for lightning page.

We can place it on visual force page

1. Create a standalone application and make it lightning out app – CompExpoOut App, check the check box to lightning out. Never preview this app. This is to give to external world not for lightning ppl.
2. Define all lightning components which you would like to expose as dependency resources

<aura:dependency resource= "c:Calculator"/>

<aura:dependency resource ="c:OpportunityComponent"/>

1. Create a vf page – LightningVFPage
   1. Earlier we mentioned <script src=” lightning/lightning.out.js”> </script>

We don’t do it anymore. But it still works.

**Test cases:**

Go to sandbox and open the position trigger handler file.

Create a apex calss—PostionTriggerHandlerTest

We can make test class and sub class as private- Test classes can be private.

When we make test class private we get a error which can be supressed by adding @isTest.

Annotation makes the VM to create a separate environment where database operations will not be committed. Also the code will not be adhering to any limits on number of lines of code for governer limits.

All test methods must be static, either we can use keyword testmethod or annotation @isTest.

If class is public, we can make methods public. Since our class is private we need not do anything on methods.

Any database operations we do in a test method are local to the method, once control comes out of the method it will be rolled back.

Test class will not access database, it will not have access to org data bydefault. We can get it by programming.

What ever data we create will only be available only till we are executing.

Locally created objects are not available in other methods.

@isTest

private class testDemo {

@testSetup

static void loadData(){

Account acc = new Account();

acc.Name='Test1';

insert acc;

List<account> accList = [SELECT Name

from Account];

System.debug(accList.size());

}

testmethod static void m1(){

Account acc = new Account();

acc.Name='Test1';

insert acc;

List<account> accList = [SELECT Name

from Account];

System.debug(accList.size());

}

@isTest

static void m2(){

Account acc = new Account();

acc.Name='Test1';

insert acc;

List<account> accList = [SELECT Name

from Account];

System.debug(accList.size());

}

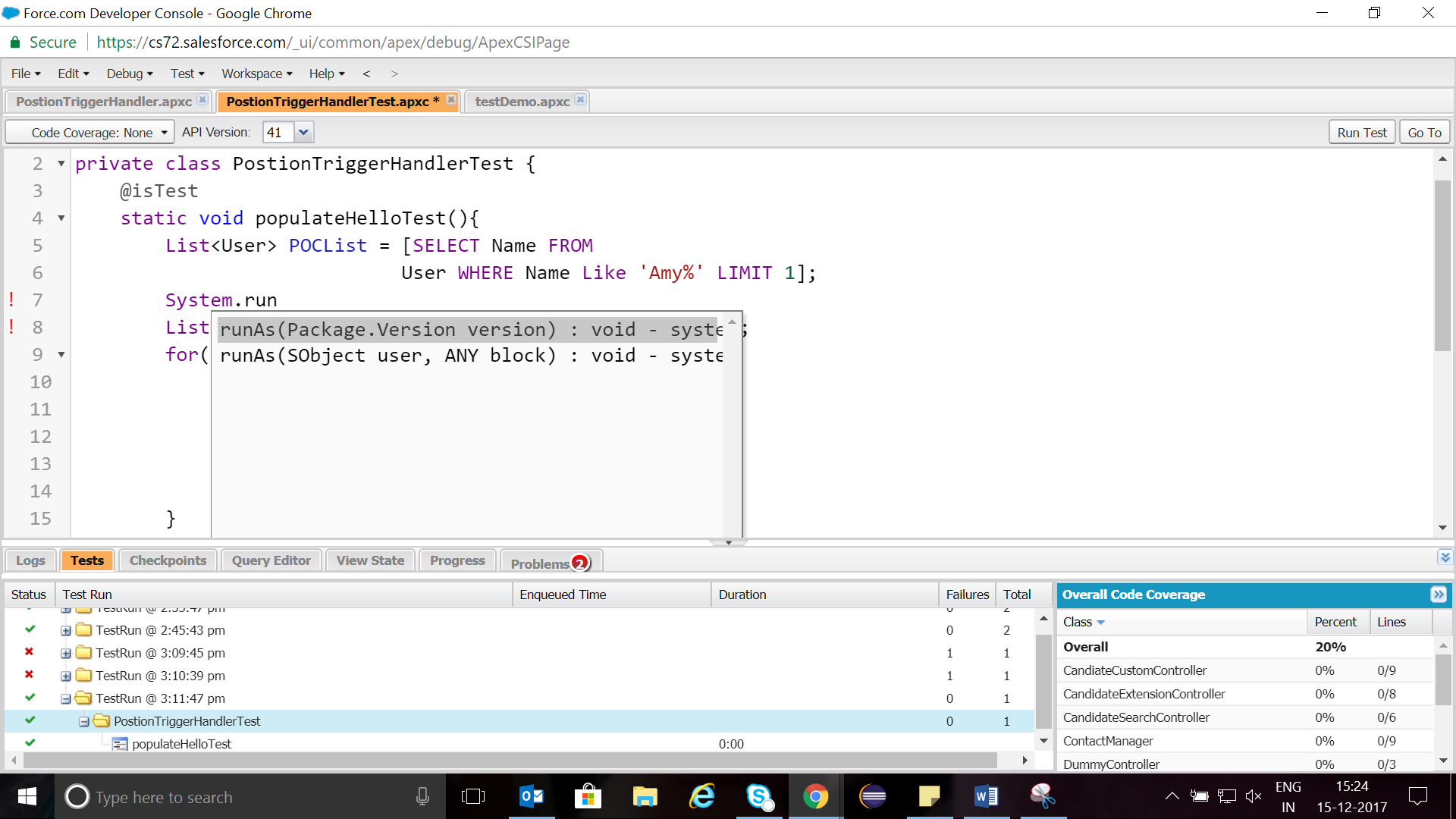
}

**To access data from org we need to do**

@isTest(seeAllData=true) at the start of the test class. Testsetup annotation is commented in this case. They both wont go together.

Accessing org data is dangerous, since many ppl will be working on same records. We cant control the same. For org data test results may vary. And test scenarios may not succeed. Also the data is usually private in test class to a user, we cant access each others data.

By default testing works in system context. To change it we use System.runAs(user, block of code)



Our test case for position

@isTest

private class PostionTriggerHandlerTest {

@isTest

static void populateHelloTest(){

List<User> POCList = [SELECT Name FROM

User WHERE Name Like 'Amy%' LIMIT 1];

List<Position\_\_c> posList = new List<Position\_\_c>();

for(Integer i=0; i<20;i++){

Position\_\_c pos =new Position\_\_c();

pos.Name='Test'+i;

pos.Job\_Description\_\_c='Test'+i;

pos.POC\_\_c=POCList[0].Id;

posList.add(pos);

}

insert posList;

List<Position\_\_c> posListReturn= [SELECT Hello\_\_c

From Position\_\_c];

Integer count=0;

for(Position\_\_c pos: posListReturn){

if(pos.Hello\_\_c =='World'){

count++;

}

}

System.assertEquals(20, count, 'MisMatch');

}

}

**in case we have multiple statements inside a method and couple of them consume 140 DML’s. and rest of them are consuming fewer DML.**

**M(){**

**S1()**

**s2() these 2 take 30 DMLS**

**we put here Test.startTest**

**s3()**

**s4() thse 2 take 140 DMLS**

**We put here Test.stopTest**

**s5()**

**s6() another 40 DMLS**

**}**

**So where start test is encountered a new block is created so that governer limits are reinitialised. Since we can have max 150 DML in a method. When it sees stopTest it will trace back to old block of memory and use another 40 DML’s**