By using Activities, we can set some default values to class properties.

* Generally, we can write business logics in Activities.
* In activities contains 108 predefined methods.  
  1. Steps  
  2. Parameters  
  3. security  
  4. Pages and classes.

**1. Steps:**

Steps tab contains series of steps, each and every step we need to call one predefined method.  
Each step contains,  
**a) Label:** It can be used to control the execution of steps. If we don’t want to execute, we need to give “//” symbol in label.  
**b) Description:** It gives the purpose of that method.  
**c) Step page:** In step page we will select the page name , that page name acts as a reference to class properties.

* Py work page is a default work page.
* Step page can be used to create user pages in a clipboard.

**d) Method:** It will define the pre-defined method available in a drop down list based on requirement.

**2. Parameters:**

It can be used to pass the parameter value to properties. They are two types;

**a) Parameter variable:** It is just like a global variables, which means we can acess the parameter variable in other activities also by using “param” keyword.  
**b) Local variable:** We can access the local variable in that activity by using “locale” keyword.

**3. Security:**

They are two types:  
**1) May start:** If we check the may start check box, that activity acts as a calling activity.  
**2) Authenticate:** If we check the Authenticate check box, that activity will be in sharing mode, means other operators can able to access this activity.  
If we uncheck the Authenticate check box, that activity will not be in sharing mode, other operators cannot be able to access this activity.

**3.Pages and classes:**

In a pages and classes tab, if we enter any page name and class name,that page name will be created in clipboard. This page name acts as a referance to class properties or clipboard properties.

**Types:**

**1) Activity:** When activity call from another activity or same activity, then activity should be Activity.  
**2) Utility:** when Activity call from a flow, then Activity should be Utility.  
**3) Connect:** If we are connecting through external data base or web services, then the Activity type should be Connect.  
**4) Trigger:** If we are writing any trigger messages, then activity type should be Trigger.  
**5) Route:** If we want to route the task , then activity type should be route.  
**6) Notify:** If we want to send notification messages through activity, then activity type should be Notify.  
**7) On change:** If we want to call activity in a declare on change rule, then activity type should be On change.

**How to call one activity from another activity:**

They are two types:  
**a) Call:** If we call one activity to another activity by using call, it will execute the called activity and return back to calling activity then execute the remaining steps in a calling activity.  
**b) Branch:** If we call one activity to another activity by using Branch, it will execute the called activity and calling activity then it will not execute the remaining steps after branch.

**Rule Availability:**

**1) Yes**: If Rule Availability is equal to Yes, then Rule is available to a particular set.  
**2) No/Draft Mode:** If Rule Availability is equal to No/Draft Mode, then Rule is not available to a particular set.  
**3) Final:** If Rule Availability is equal to Final, then Rule is not over write or change in a particular rule set.  
**4) Withdrawn:** If rule availability is equal to Withdrawn, then Rule is not available to lower version and it is available in current and higher version .  
**5) Blocked:** If Rule availability is equal to Blocked, then Rule is not available to lower versions of that particular class and inheritance class.

**Inheritance:**

They are two types:  
**1. Pattern inheritance.**  
**2. Direct inheritance.**  
Whenever we are searching for rule, Pega first checks with pattern Inheritance, if pattern inheritance is not able to find that rule then, it will check with Direct inheritance.  
¬ Direct inheritance checks with @base class.  
¬ By default pega follows Pattern inheritance.

**OBJ-METHODS:**

By using OBJ-Methods, we can fetch the data in to internal data base.  
**1. OBJ-OPEN:** By using OBJ-Open method, we can fetch single record from the data base by passing primary key value.  
**2. OBJ-SAVE:** By using OBJ-save method, we can save the data in to data base table.  
**3. OBJ-BROWSE:** By using OBJ-browse method, we can fetch multiple records from the data base table.  
**4. OBJ-DELETE:** By using OBJ-delete method, we can delete in to class insistance in a data base table.

**Dynamic select:**

By using Dynamic select, we can fetch the data from data base table Dynamically by using **OBJ-BROWSE** method.

**Repeating layout:**

By using Repeating lay out, we can display the multiple records data in a user interface.

**RDB-Methods:**

**1) RDB-List:** we can fetch multiple records from external data base.  
**2) RDB-OPEN:** we can fetch the single record from external data base table.  
**3) RDB-SAVE:** we can save the data in to external data base.  
**4) RDB-DELETE:** we can delete the class insistence.

**How to connect with external data base:**

* Whenever we want to connect with external data base, first we need to create the data base.
* Whenever we create the data base, we need to provide the JDBC url name, user name and password.
* Once we created the date base, we need to create the data base class. In a data base class , we need to create the required properties.
* After creation of data base class, we need to create the data base table. At the time of creating data base table, we need to provide the data base name, data base table name and schema name.
* We need to create the connect sql. In a connect sql rule , if we want to fetch multiple records then we need to write the query in browse tab.
* If we want to fetch single record then we need to write query in open tab.
* If we want to save the data in to data base then we need to write the query in save tab.
* Create the activities by using RDB-Methods.

**Data Transforms:**

By using Data Transforms, we can set default values to properties.

* We can copy one page information to another page.
* The default Data Transform in pega is pyDefault Data Transform.
* The default Data Transform belongs to ‘Work-’ class.
* Data Transform can be created under data model category.  
  Correspondence:
* By using Email correspondence we can send emails to other parties.
* By using send email notification method we can send emails to third parties.
* In a send email notification method we need to provide To mail, From mail, subject, message, smtp host name and password.
* For message we need to create the Email correspondence.
* By using Property-set-corr method in a activity we can call the correspondence.

**Decision rules:**

They are four types :  
1) Decision Table  
2) Decision Tree  
3) When Rule  
4) Map value

**1) Decision Table:**

* It can be created under decision category.
* By using property-map-decision table method, we can call the decision table in a activity.
* By using decision shape in a flow we can call decision table.
* Decision table returns only one value at a time.  
  2) Decision Tree:
* Decision Tree can be created under Decision category.
* By using property-map-decision tree method, we can call Decision Tree in a activity.
* By using Decision Shape in a flow, we can call Decision tree.

**Difference between Decision Table and Decision Tree:**

**Decision Table:**

* The logic Implemented in Decision Table is if, else if condition.
* In Decision Table, if first condition is true, it will not check the remaining conditions/if first condition is false , then only it will check next condition.
* For simply logics, we can go for Decision table.  
  Decision Tree:
* The logic implemented in Decision Tree is if, if condition’s.
* In a Decision Tree, if first condition is true or false, it will check the al condition’s and it will return results.
* For simple logics, we can go for Decision Tree.

**VALIDATIONS:**

There are two types of validations:  
**1. Edit validate:**

* By using Edit validate, we can validate single property.
* Edit validate can be created under Data model category.
* Edit validate can be called under property definition advanced tab.
* For Edit validate rule, we need to write java source code.
* By using Edit validate, we can achieve client side validations.

**2. Obj validate:**

* By using Obj validate rule, we can validate the multiple properties.
* Obj validate can be created under process category.
* Obj validate rule can be called in a flow action under Action tab.
* By using ‘obj-‘ validate method in a activity, we can the obj validate.
* By using obj validate, we can achieve server side validations.

**Client side validation:**

If you want to validate properties before submitting the forms, those validation we can call as a client side validations.

**Server side validation:**

If we want to validate properties after submitting the forms, those validations we can call as a server side validation.

**Declarative Rules**

Declarative Rules no need to call separately. Based up on the properties we can create the Declarative Rules.  
Types:  
1. Declare expression  
2. Declare constraints  
3. Declare on change  
4. Declare page  
5. Declare Trigger  
6. Declare Index

**1) Declare expression:**

whenever we are changing the source property value automatically, it will change the target property value.

**Forward chaining:**

Whenever we are changing the input property value automatically it will change the target property value.

**Backward chaining:**

Whenever we are using target property value, automatically it will recalculate the input values.

**2) Declare constraints:**

* By using declare constraints, we can validate multiple properties.
* By using this, we can achieve client side validation.
* It can be created under Decision category.
* Declare constraints no need to call separately.  
  3) Declare on change:
* If we change any property value, automatically it will run the activity.
* For Declare on change we need to create when rule and activity.
* Declare On change can be created under Decision category.  
  4) Declare Pages:  
  Declare Page is a top-level clipboard created by execution of a declare page rule type. The name of a declarative pages starts with Declare\_. The content of this pages are visible but Read-only to the requestors. Declarative rules can be created and updated through activities.  
  5) Declare Trigger:  
  It can be used to run the Activities automatically when an instances of a specific class are created, updated or deleted in the database. It is defined under Decision category. It implements Forward chaining.  
  6) Declare Index:  
  It is used to define the criteria for automatically maintains index instances for faster access.

**Types of Flow Action**

1. Connector Flow Action:

* It can be called in a outgoing connector.
* In a out going connector we can be able to see which Flow Action we called.
* We can create a normal flow action.
* In a connector flow action, work object will move from one assignment to another assignment.

2. Local flow action:

* It can be called in a Assignment shape.
* In a Assignment shape we cannot able to see which Flow Action we called.
* If we want to display Pop-up’s then, we need Local Flow Action.
* In a Local Flow Action, work object will be in same Assignment shape.

**Types of Flows:**

1. Straight through process:  
The flows contains only starter shape and ending shape without any shapes then we called as a straight through process.  
2. Process flow:

* In a process flow we can create work object and run the flow.
* We can call the flow in a outgoing connector.
* By using process flow, we cannot achieve front and back buttons.
* In a process flow individual Assignment routing is possible.
* We can call SLA, Local flow actions, Tickets in a Assignment shape.
* In a process flow contains many shapes.

3. Screen Flow:

* In this, we cannot create the work object and individually we cannot run the flow.And by using sub process shape we can call the screen flow in some other shape.
* We can call the Flow Action in a Assignment shape.
* In a screen Flow, we can achieve front and back buttons.
* In this, individual assignment is not possible, entire screen flow can route in a starter shape.
* We cannot call SLA, local action, Tickets in a Assignment shape.
* In a screen Flow contains some shapes.  
  Routing  
  By using Routing, we can transfer the work or task to single or multiple operator’s. Routing can be done in two ways:  
  a) Work List:  
  If we want to transfer or route the work to single operator, we can use work list.  
  b) Work Basket:  
  By using work Basket, we can transfer the work to multiple operators.

**SLA**

SLA defines how much time and urgency is required to complete work.  
• SLA can be created under process category.  
• SLA defines 3 types of time intervals.

a. Goal Time: Goal time can be calculated from whenever your work is created or started.

b. Dead Line Time: It can be calculated from whenever task is created or started.  
c. Passed Dead Line Time: It can be calculated from whenever dead line time is completed

• **SLA defines 5 types of urgency values**  
a. Default urgency  
b. Initial urgency.  
c. Goal urgency.  
d. Dead Line urgency.  
e. Passed Dead Line urgency.  
• The default urgency value is 10.  
• The default value urgency is defined in PyDefault Data Transform, its property name is .PxUrgencyWorkClass.

**They are two types of SLA’s**  
1. Assignment level SLA:  
It can be called in flow under Assignment shape.  
2. Work object level SLA:  
It can be called in a default data transform with property name .pySLAname

**Tickets**

* Tickets is just like a go to statement, whenever you want to jump from one assignment to another assignment we can set Tickets.
* It can be created under process category.
* Tickets contains only history tab.
* By using obj-set-Ticket method in a activity we can call Tickets.  
  Rule Inspector.
* By using Rule Inspector, we can identify the flow action, section’s and property names at the run time.
* If we want to Modify sections, flow actions, properties and declarative rules, we can use Rule Inspector.

**Exception**

By using Transition check box in a activity we can handle the Exception.

**Tracer**

* Tracer is the one of the debugging tool.
* By using Tracer, we can Trace the activities, decision rules, declarative rules, when rules, flows and data transforms.

**Tracer contains:**

I. Pause: if you want to start or stop  
II. Clear: if you want to clear the tracer, needs to click on clear.  
III. Settings: whatever we want to trace, we need to select the particular rule like activities, when rule.  
IV. Break points: If I want to trace particular activity, particular step then We need to set Break point.  
V. Watch variable: we can able to see parameter variable in a watch.  
VI. Remote Tracer: If I want to trace some other operator work, by using Remote Tracer we can Trace.

**Documentations**

Click on pega logo-Application-tools-Documentation

**War file**

* First we need to check in all the rules.
* Lock the rule set.
* Create a package file.
* Export product file in to PROD OR UAT OR DEV TETSING.

**LET NOW WE GO FOR MOCK INTERVIEW**

**What is built in Pega class?  
How many types of inheritance in Pega?  
What is pattern inheritance and direct inheritance?  
Define data class and work class?  
What is operator and how will you create it?  
What is rule and ruleset in pega prpc?  
How to Create Operators and First Abstract Class?  
How to Create a Class Structure?  
Define page, page list and page group?  
Define single value, value list and value group?  
How to Create Data Table Classes?  
How to Create User interface?  
What is section and flow action?  
Define different types of layout in section?  
How to create a flow action?  
How to create activity.  
Giving Input using property set method.  
Looking at the pyWorkPage.Using Page remove method.  
Checking User pages , created In the clipboard.  
How to insert data  into the data tables through activities.  
How to fetch data from the data tables.  
How to fetch multiple records from the data tables.  
How to conditionally display a section.  
What is decision rule and define the types of decision rule ?  
How to create Decision Table?  
How to create a Decision Tree?  
What is declarative rules and defines the types of decorative rule?  
How to declare expressions?  
How to declare constraints?  
How to Declare on change?**

FQA

**1.     Define Service and Connecter?**

Service— Pega PRPC acts as server and external system acts as client.  
Service SOAP

Connect–  Pega PRPC acts as client and external system acts as server.  
Connect-SOAP, Connect SQl

**2.     Which type of Service or Connecter used in your project?**

Service-SOAP, Connect-SOAP, Connect-SQL…

**3.     How to call one activity from other Activity**

Call/branch

**4.     How to run 2 activities in parallel?**

We will have run in parallel option to run Connect-SOAP  activity..

**5.     Diff b/w Decision tree, Table, Map-Value?**

Tree Complex if else structure..  
Table- if else in tabular format.. less complex..  
Map-Value  based of single property..

**6.     Class group, work pool, workbasket, work group?**

**7.     Different type of connecter in a flow?**

Status,else,always

**8.     Flow, flow action, local action difference?**

Flow—defines process…  
Local action—updates the work item..never advances the flow..  
Connector—advances the flow..

**9.     What are SLA type’s types?**

Workobject level – give sla name at pySLAName property of data transform,…  
Assignment level..Give a Assignement shape

**10.  How you configure SLA in assignment?**

At assignment shape in flow

**11.  What are Work parties?**

Interested to know about the work but do not directly resolve WO…  ex Com,gov,operator,org,Person

**12.  What are different methods used in activities?**

Page-New, Page-Remove,  Obj-open,Obj-List, Obj-Save, RDB-Save,RDB-List,RDB-Delete etc..

**13.  Activity types**  
  
Activity, utility, LoadDeclarativePage,  Trigger,Validate,Assembly etc..,

**14.  How to create work object through an activity and which method to use to create work object**

Call Work-.new

**15.  Case Management**

**16.  Different type of flow**

ScreenFlow, SubFlow  starter flow..

**17.  Different types of container in section**

**18.  Types of Harness**

New, Confirm, Perform Review Reopen

**19.  Agent types**

Standard,Advanced,Legacy

**20.  How you trace an agent**

Using SMA  
Agent Mgnt ◊ select the required agent-> delay the agent->  
Find the requestor id..  
Requestor Mgt -> trace the agent by selecting requestor id..

**21.  Data Queue agents**

Created by PRPC system.. for Rule-Agent-Queue

**22.  Obj-open ,Obj-Browse**

Obj-open to open single instance..  
Browse is to form the list of embedded pages… for the given criteria..

**23.  Diff b/w Obj type methods and RDB type methods**

Obj-used for both internal and external DBs  
RDB used for only External databases..

RDB methods working with Pega internal DBs causes Loss of data.

**24.  Different types of properties**

11 modes  
Single aggregate..

**25.  Diff b/w single property , aggregate property**

Single-Property simple integer, text, decimal, date, etc..,  
Aggregate — List, group properties

**26.  Define report and types**

Reporting— detailed display of records from DB..

**27.  Diff  b/w list view ,Summary View and Report Definition**

List view—  Detailed record by record display  
Summary View— Chart representation of data  
Report Definition—Can be used as both List View and Summary view

28.  Email Listener ,File listener  
29.  Access Groups, access roles  
30.  Parameter values ,local values  
Parameter values are    across activities..  
Local values  available only in that activity..

31.  Different types of pages in clipboard  
Declare pages, user pages, operator ,Organization, division etc.,

32.  PAL, Pre-flight  
Preflight facilitates to find warnings in activities and decision rules..

33.   Tracer  
34.  How to connect External DB from your application  
create         Data-Admin-DB-Name with java bridge details..  
           Data-Admin-DB-Table with schema name and Table name.  
          Click on test connectivity

35.  Smart BPM Methodologies  
14 steps..

36.  Diff b/w BPM and BRE  
BPM – automatic processing..  
BRE—Only business rules.. flows not required.

37.  Circumstance and types simple circumstance (value based,time based), multiple circumstance (circumstance definition, circumstance template)

38.  Rule set and Rule set Versions  
Container of Rules id Ruleset—multiple versions can be created for a single ruleset

39.  What is grid layout tree layout  
40.  Guardrails in Pega  
Limit custom java, establish robust foundation, build for change, do nthing hard,etc..,

41.  Import ,export rules   Product, ProductPatch

42.  Production rule set  
43.  Agile Methodologies  SCRUM

44.  Work Type:  
fundamental unit of work .can be create, update and resolve  
45.  Work Item:  
Unique Id, Urgency and status.  
6R`s Automation: Receive (accepting), Routing, Report, Respond, Research (decision), Resolve  
46.  What are the fields in the properties panel of an assignment shape?  
Route    Notify Ticket Status Localactions  Harness  
47. Where can we call the activities in a flow action?  
Post activity Preactivity at action tab  
48.  What is Class structure of your Project? Explain about the project Flow?  
Organizatin-> Division-> Framework->Implementation  
49.  What is the Rule availability?  
Rule available for  Rule Resolution algorithm, Yes,No,Final,Blocked,Wirhtdrawn  
50. What is the Final availability and how can change availability of Rule?  
Save into different Ruleset

51.  Can you Override Final rule?  
S.     Only in same Ruleset  
52.  What is the rule resolution Algorithm and can you tell me how it works or search?  
Rule resolution is the search algorithm that the system uses to find the best or most appropriate rule instance to apply in a situation.  
Rule resolution applies to all but a few rule types — classes that inherit from the Rule- base class. Rule resolution does not apply to instances of classes derived from the Work-, Data-, or any other base class.