## Downloading and Installing: http://grails.org/download

The first step to getting up and running with Grails is to install the distribution. To do so follow these steps:

* [Download](http://grails.org/Download) a binary distribution of Grails and extract the resulting zip file to a location of your choice
* Set the GRAILS\_HOME environment variable to the location where you extracted the zip
  + On Windows this is typically a matter of setting an environment variable under My Computer/Advanced/Environment Variables
* Then add the bin directory to your PATH variable:
  + On Windows this is done by modifying the Path environment variable under My Computer/Advanced/Environment Variables

If Grails is working correctly you should now be able to type grails -version in the terminal window and see output similar to this:

Grails version: 2.3.8

### Downloading Eclipse: http://grails.org/products/ggts

We recommend that users of [Eclipse](http://www.eclipse.org/) looking to develop Grails application take a look at [Groovy/Grails Tool Suite](http://grails.org/products/ggts), which offers built in support for Grails including automatic classpath management, a GSP editor and quick access to Grails commands. See the [STS Integration](http://grails.org/STS+Integration) page for an overview.

## How to Run: http://grails.org/doc/latest/guide/gettingStarted.html#requirements

Open terminal window, Type: grails RunApp

To Stop grails, Type: grails StopApp

# URL: http://localhost:8080/FHC/

## Rule Learn Document: <http://docs.jboss.org/drools/release/6.0.1.Final/drools-docs/html/index.html>

## Groovy Learn Document: http://groovy.codehaus.org/Beginners+Tutorial

Create new survey steps:

1. Define survey and branch
   1. Find “../FHC/grails-app/conf/BootStrap.groovy” , this file will load the data when the application started
   2. Define the branch and survey, here is the example:

**def** branch = **new** Branch(branchId: 1, name: "Tengda's Branch", logo: "/logos/cfmlogo.jpg", url:"http://www.cfms4.com",email:”sdfdf@dd.com”)

**def** survey = **new** Survey(surveyId: 1, name: "defaultBankSurvey")

survey.save()

branch.survey = survey

branch.save()

Notice : logo file “/logos/cfmlogo.jpg” is under “../FHC/web-app/images”

Survey name “defaultBankSurvey” must match the flow name, will explain later

1. Create new survey flow
   1. Find “../FHC/grails-app/controllers/SurveyController.groovy” , this file is control the survey flow
   2. Create the survey flow, beware the flow name should be “surveyname+Flow” like “defaultBankSurveyFlow”, here is the example:

def defaultBankSurveyFlow = {

}

* 1. Define the first page flow in the survey flow, the is the example:

def defaultBankSurveyFlow = {

defaultBankSurveyFlowStart {

action{

**return** next()

}

on("next").to "demographicInfo"

}

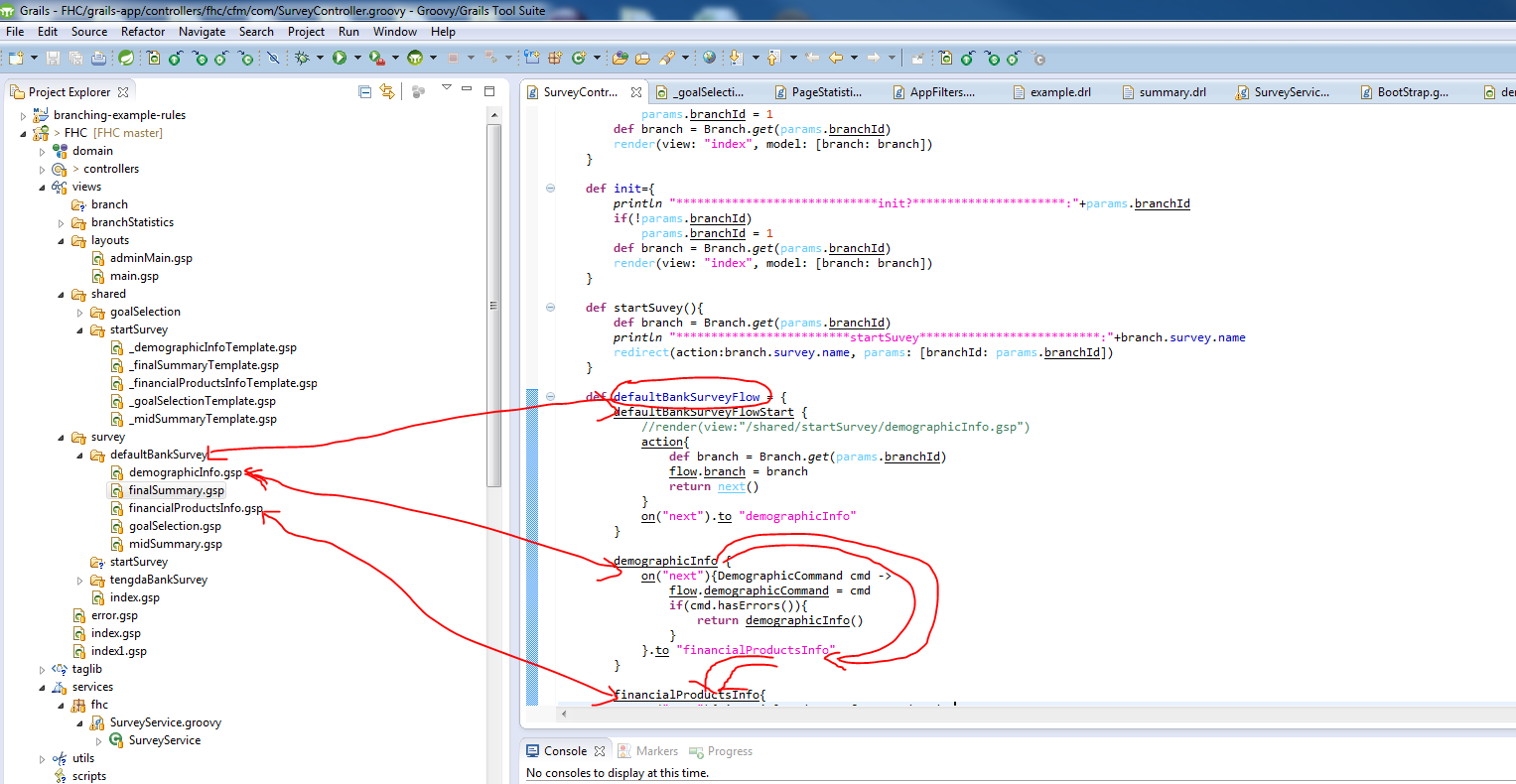
demographicInfo {

on("next").to "financialProductsInfo"

}

}

“demographicInfo” is the first page of the flow, and the next page is “financialProductsInfo”, all these pages are under “/FHC/grails-app/views/survey/defalultBankSurvey”, beware the flow name must be the same name as folder’s name(ex: defalultBankSurvey), and the page name must be the same name we define in the survey flow (ex: demographicinfo.gsp).



* 1. Define the page

<!DOCTYPE html>

<html>

<head>

<meta name=*"layout"* content=*"main"*/>

</head>

<body>

<div id=*"page-body"* role=*"main"*>

<tmpl:/shared/startSurvey/demographicInfoTemplate/>

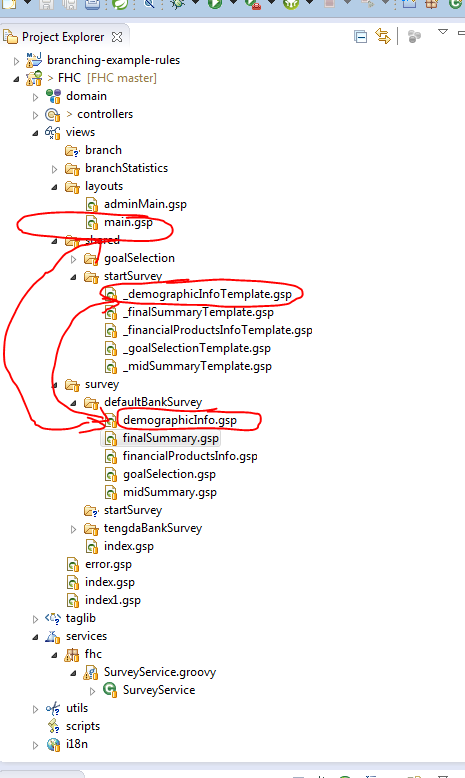
</div>

</body>

</html>

the layout can be found “/FHC/grails-app/views/layouts/main.gsp”, beware names match <meta name=*"layout"* content=*"main"*/> , header and footer define in main.gsp

the body content can be found “/FHC/grails-app/views/shared/startSurvey/\_demographicInfoTemplate.gsp”, beware the name matches



* 1. Carry the data for the page, for example : \_demographicInfoTemplate.gsp
     1. Define the domain command object for carrying the data, and if any data we want to pass, we need to do similar like below
        1. Domain class: DemographicCommand.groovy can be found “/FHC/grails-app/domain/fhc/cfm/com/ DemographicCommand.groovy”

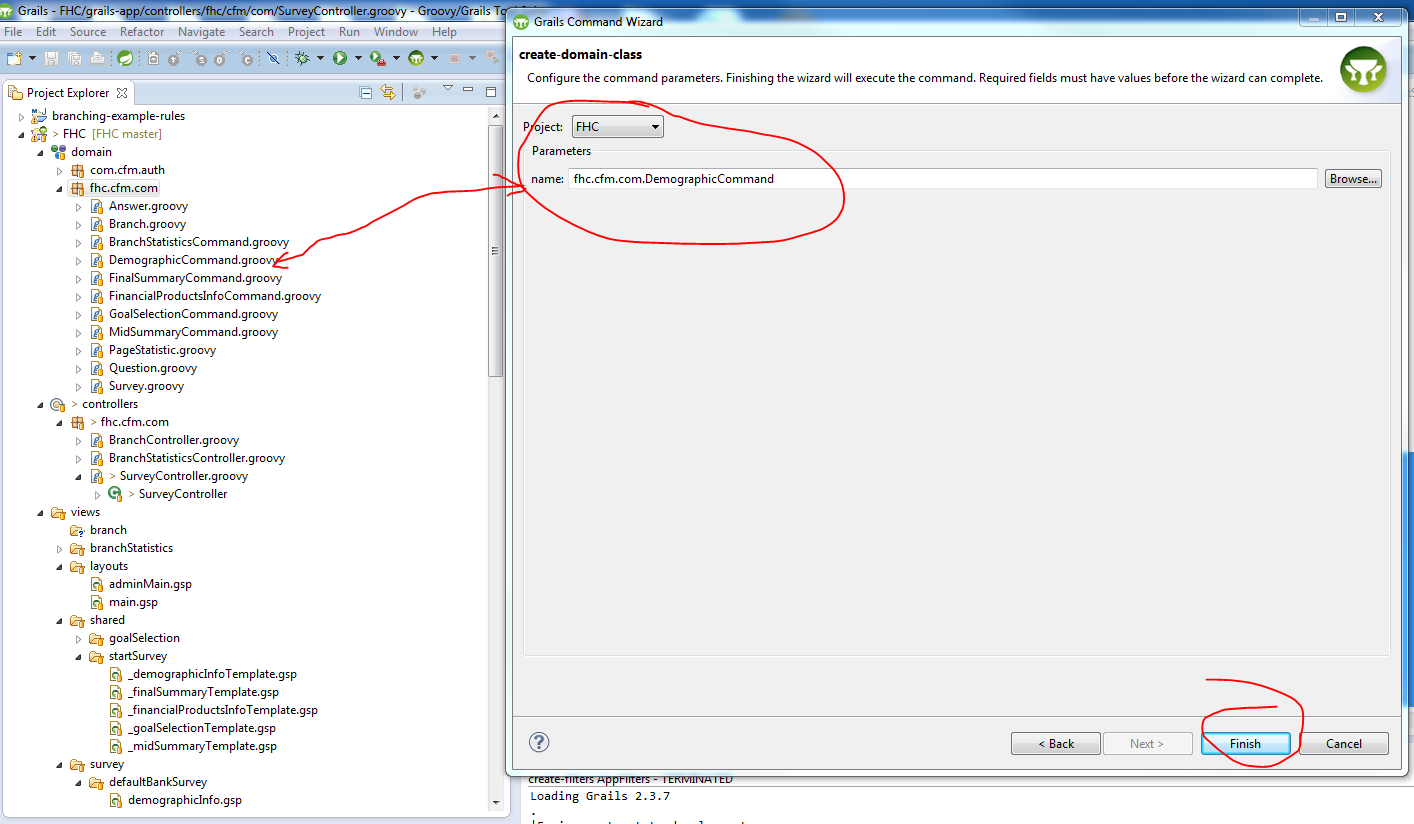
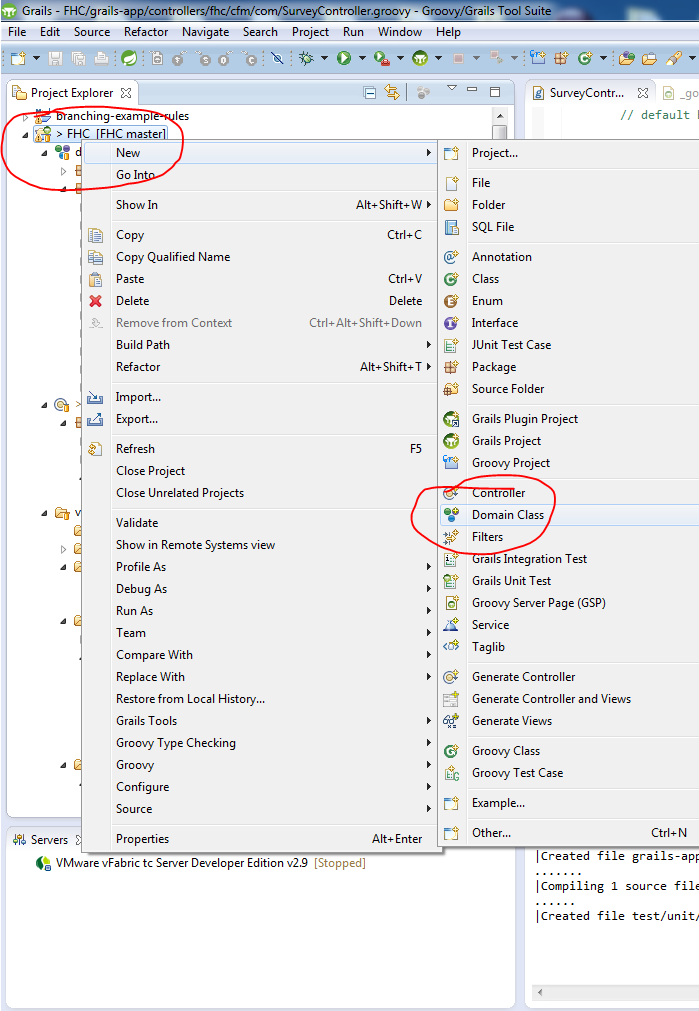
**package** fhc.cfm.com

**class** DemographicCommand **implements** Serializable{

String age;

}

Beware the name conversion; Domain class name must end with Command, and the domain class name match the page name. Here is the step for creating the domain class. In the Eclipse, right click the domain, select new->domain and create

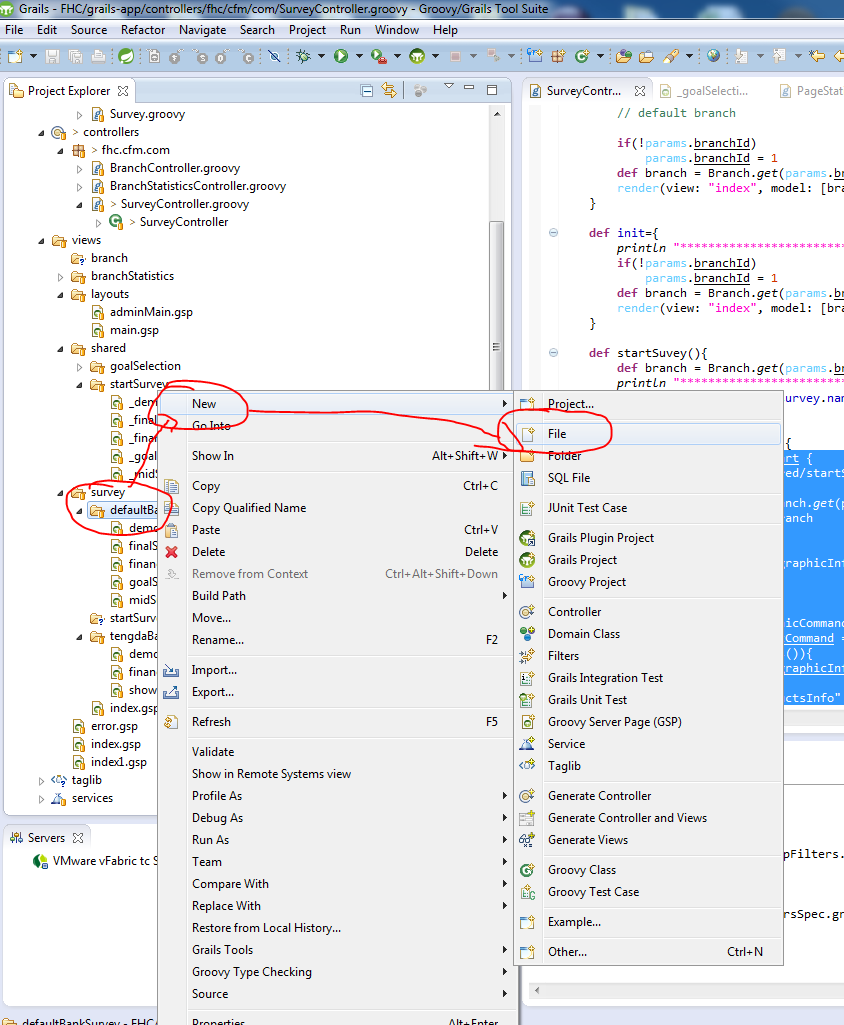


* + - 1. Page:

<g:textField id=*"age"* name=*"age"* value=*"*${demographicCommand==**null**?0:demographicCommand.age?:age}*"*/>

<g:submitButton name="next" value="Next"></g:submitButton>

In the Eclipse, right click the views, select new->file and create



* + - 1. Back to the flow, we need to do something look like this in order to carry the data, beware “next” value in submitButton matchs the on(“next”) we define in the flow, means as soon as we click the next, info will get passed

def defaultBankSurveyFlow = {

defaultBankSurveyFlowStart {

action{

**return** next()

}

on("next").to "demographicInfo"

}

demographicInfo {

on("next"){DemographicCommand cmd ->

flow.demographicCommand = cmd

**if**(cmd.hasErrors()){

**return** demographicInfo()

}

}.to "financialProductsInfo"

}

}

* + - 1. How to deal with rules?
         1. Fire the rules in the flow, we put method like “def command = surveyService.fireMidSummaryRule(questions)”
         2. The rule service is located in “..FHC/grails-app/services/fhc/SruveyService.groovy”
         3. The Rule file is in “..FHC/src/groovy/rules/summary.drl”
         4. 