Ping Manual

Use it

August 2017

Sommaire

[1. Introduction 3](#_Toc489972090)

[2. Usage 3](#_Toc489972091)

[3. Installation 3](#_Toc489972092)

[4. Build a scenario 3](#_Toc489972093)

[4.1. Process 3](#_Toc489972094)

[4.2. Command Scenario 6](#_Toc489972095)

[4.3. Groovy Scenario 7](#_Toc489972096)

[4.4. Use an external server 7](#_Toc489972097)

[5. Export / Import the scenario 7](#_Toc489972098)

[6. Run a scenario 7](#_Toc489972099)

[6.5. Manually 7](#_Toc489972100)

[6.6. By a REST CALL 7](#_Toc489972101)

[6.6.1. Start 7](#_Toc489972102)

[6.6.2. Status 8](#_Toc489972103)

[6.7. Inside a CI environment 8](#_Toc489972104)

Content

# Introduction

This document explains how to use the Ping Page.

Bonita portal has a huge function: it accepts to deploy some custom page. There are two way to build a custom page in Bonita

* Using the UI Designer
* Using the Groovy page architecture

With the UI Designer, you design your page using the standard and custom widget. You can call the Bonita Rest API, and you can call some custom REST API.

You face theses limitation with the UI Designer

* No control on the AngularJS controller. Then, when you create a JavaScript variable, the UI Designer page call this script every time: you add a character in the Input? your script is executed again. Same, if you have two different scripts, you don’t know which script is called first.
* No access to the HTML. If you need a special control, then you are supposed to build a Custom Widget. Then, for simple usage like have a button to call a JavaScript, it’s quite impossible to do that with the UI Designer
* Deployment issue. If you page need 3 or 4 custom RESTAPI, you must build them separately, and deploy it separately. You have no control to verify that the RESTAPI has the correct version for your page.

But the UI Designer is wonderful to build 80% of your custom page, because it’s a WHYISWIG tool, and not required any development.

If your need is most complex, then the Bonita Portal offer you the second level: the groovy page architecture. The Ping page is a tutorial to explain you how to deploy and use it. It contains too some tools to help you to build some powerful page using Bonita Portal. To use this way, you should know:

* HTML to build the page
* AngularJS to be able to add special behavior on the page
* Java or Groovy for the server side

# Principe

# Page RestAPI

# Events

# Properties

# Chart

# Modal (Dialog)

# Generate a Exel file in local

# Download a file

Html :

On the page, place a link:

<a class="btn btn-info" href="?page=custompage\_ping&action=downloadPdf" target="blanck">Download a PDF</a>

Nota: using Bootstrat, the link look like a button



AngularJS :

Nothing

Groovy :

On the action “download”, you must send a document

} **else** **if** ("downloadPdf".equals(action))

{

logger.info("Download the PDF");

// get the document which is saved in the Custom Page directory

InputStream input = pageResourceProvider.getResourceAsStream("doc/Ping Manual.pdf");

// we get the outpuStream in order to send the document as it

OutputStream output = response.getOutputStream();

**byte**[] buffer = **new** **byte**[1024];

**int** bytesRead;

**while** ((bytesRead = input.read(buffer)) != -1)

{

output.write( buffer, 0, bytesRead);

}

output.flush();

output.close();

// then add the name and the correct content type

response.addHeader("content-disposition", "attachment; filename=\"Ping Manual.pdf\"");

response.addHeader("content-type", "application/pdf");

**return**;

}

Nota : don’t call before the method

PrintWriter out = response.getWriter()

Because else Outstream is lock by the PrintWriter and then the method response.getOutputStream() will failed.

# Drag and drop and upload a file

The idea is to have a HTML zone where user can drip a file. Then the page will ommediately send the page to the server,and you can open it

Html :

In the Header, add

<link rel="stylesheet" href="pageResource?page=custompage\_meteor&location=style/dropbox.css">

Place in your HTML

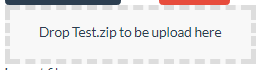
<div ng-file-drop ng-model="importfiles" class="drop-box ng-isolate-scope ng-valid ng-dirty"

drag-over-class="{accept:'dragover', reject:'dragover-err', delay:100}"

multiple="true" allow-dir="false">Drop Test.zip to be upload here</div>

<div ng-no-file-drop class="drop-box" style="display: none;">File Drop is not supported for this browser</div>

With the result



And add in the bottom

<script src="pageResource?page=custompage\_ping&location=js/angular-file-upload-shim.min.js"></script>

<script src="pageResource?page=custompage\_ping&location=js/angular-file-upload.min.js"></script>

The different files (angular-file-upload-shim.js, angular-file-upload.min.js, dropbox.css has to be place.

AngularJS :

Add the angularFIleUpload in the angular.module

**var** appCommand = angular.module('pingmonitor', ['googlechart', 'ui.bootstrap', 'ngSanitize','ngModal','angularFileUpload']);

Add in the function the module $upload

appCommand.controller('PingControler',

**function** ( $http, $scope, $sce, $interval, $timeout, $upload ) {

and watch the drop down area

**var** me = **this**;

$scope.$watch('importfiles', **function**() {

**if** ($scope.importfiles)

{

**for** (**var** i = 0; i < $scope.importfiles.length; i++) {

**var** file = $scope.importfiles[i];

// V6 : url is fileUpload

// V7 : /bonita/portal/fileUpload

$scope.upload = $upload.upload({

url: '/bonita/portal/fileUpload',

method: 'POST',

data: {myObj: $scope.myModelObj},

file: file

}).progress(**function**(evt) {

// console.log('progress: ' + parseInt(100.0 \* evt.loaded / evt.total) + '% file :'+ evt.config.file.name);

}).success(**function**(data, status, headers, config) {

console.log('file ' + config.file.name + 'is uploaded successfully. Response: ' + data);

me.fileIsDropped(data);

});

}

} // end $scope.importfiles

});

Write the method “fileIsDropped”:

**this**.fileIsDropped = **function**( testfileimported ) {

self.configwait=**true**;

$http.get( '?page=custompage\_meteor&action=import&filename='+testfileimported )

.success( **function** ( jsonResult ) {

self.config.list = jsonResult.configList;

self.listeventsconfig = jsonResult.listeventsconfig;

self.configwait=**false**;

});

Groovy :

On the action you have the file name, let get the file itself.

The point is this location change on the different BonitaVersion, so you have to explore different path.

HEADQUARTERS

PARIS, FRANCE

76 boulevard de la République

92100 Boulogne-Billancourt

EMEA, ASIA & LATIN AMERICA

GRENOBLE, FRANCE

32, rue Gustave Eiffel  
38000 Grenoble

NORTH AMERICA

SAN FRANCISCO, USA

44 Tehama Street  
San Francisco, CA 94105

NEW YORK, USA

33 Nassau Avenue

Brooklyn NY 11222