|  |  |
| --- | --- |
| PSP0.1 Development Script | |
| Purpose | To guide the development of small programs |
| Entry Criteria | * Requirements statement * Project Plan Summary form with estimated program ***size and*** development time * Time and Defect Recording logs * Defect Type standard ***and Coding standard*** |

|  |  |  |
| --- | --- | --- |
| Step | Activities | Description |
| 1 | Design | * Review the requirements and produce a design to meet them. * Record in the Defect Recording log any requirements defects found. * Record time in the Time Recording log. |
| 2 | Code | * Implement the design ***following the Coding standard.*** * Record in the Defect Recording log any requirements or design defects found. * Record time in the Time Recording log. |
| 3 | Compile | * Compile the program until there are no compile errors. * Fix all defects found. * Record defects in the Defect Recording log. * Record time in the Time Recording log. |
| 4 | Test | * Test until all tests run without error. * Fix all defects found. * Record defects in the Defect Recording log. * Record time in the Time Recording log. |

|  |  |
| --- | --- |
| Exit Criteria | * A thoroughly tested program ***that conforms to the Coding standard*** * Completed Time and Defect Recording logs |

|  |  |
| --- | --- |
| PSP0.1 Postmortem Script | |
| Purpose | To guide the PSP postmortem process |
| Entry Criteria | * Problem description and requirements statement * Project Plan Summary form with program size and development timedata * Completed Time and Defect Recording logs * A tested and running program ***that conforms to the coding and size measurement standards*** |

|  |  |  |
| --- | --- | --- |
| Step | Activities | Description |
| 1 | Defect Recording | * Review the Project Plan Summary to verify that all of the defects found in each phase were recorded. * Using your best recollection, record any omitted defects. |
| 2 | Defect Data Consistency | * Check that the data on every defect in the Defect Recording log are accurate and complete. * Verify that the numbers of defects injected and removed per phase are reasonable and correct. * Using your best recollection, correct any missing or incorrect defect data. |
| ***3*** | ***Size*** | * ***Count the size of the completed program.*** * ***Determine the size of the base, reused, deleted, modified, added, total, added and modified, and new reusable code.*** * ***Enter these data in the Project Plan Summary form.*** |
| 4 | Time | * Review the completed Time Recording log for errors or omissions. * Using your best recollection, correct any missing or incomplete time data. |

|  |  |
| --- | --- |
| Exit Criteria | * A thoroughly tested program ***that conforms to the coding and size measurement standards*** * Completed Project Plan Summary form * ***Completed PIP forms describing process problems, improvement suggestions, and lessons learned*** * Completed Time and Defect Recording logs |

PSP0.1 Project Plan Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Andres Suarez Alfonso | Date | 4 Feb 2015 |
| Program | Tarea 2 | Program # | CSOF5101\_01\_2 |
| Instructor | Luis Daniel Benavides Navarro | Language |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Program Size*** | ***Plan*** | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Base (B)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Deleted (D)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Modified (M)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added (A)*** |  | | |  | 414 | | |  |  | | |
|  |  | | |  | ***(T − B + D − R)*** | | |  |  | | |
| ***Reused (R)*** |  | | |  | 173 | | |  | 173 | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added and Modified (A+M)*** | 360 | | |  | 414 | | |  | 414 | | |
|  |  | | |  | ***(A + M)*** | | |  |  | | |
| ***Total Size (T)*** |  | | |  | 414 | | |  | 414 | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Total New Reusable*** |  | | |  | 173 | | |  | 173 | | |
|  |  | | |  |  | | |  |  | | |
| **Time in Phase (min.)** | ***Plan*** |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning | 10 |  | 10 | | |  | 14 | | |  | 3.19% |
| Design | 15 |  | 20 | | |  | 23 | | |  | 5.25% |
| Code | 295 |  | 293 | | |  | 390 | | |  | 89.04% |
| Compile | 5 |  | 22 | | |  | 32 | | |  | 7.3% |
| Test | 15 |  | 10 | | |  | 20 | | |  | 4.56% |
| Postmortem | 20 |  | 35 | | |  | 59 | | |  | 13.47% |
| Total | 360 |  | 390 | | |  | 438 | | |  | 100% |
|  |  |  |  | | |  |  | | |  |  |
| **Defects Injected** |  |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  |  | 0 | | |  | 0 | | |  | 0% |
| Design |  |  | 0 | | |  | 0 | | |  | 0% |
| Code |  |  | 6 | | |  | 7 | | |  | 100% |
| Compile |  |  | 0 | | |  | 0 | | |  | 0% |
| Test |  |  | 0 | | |  | 0 | | |  | 0% |
| Total Development |  |  | 6 | | |  | 7 | | |  | 100% |
|  |  |  |  | | |  |  | | |  |  |
| **Defects Removed** |  |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  |  | 0 | | |  | 0 | | |  | 0% |
| Design |  |  | 0 | | |  | 0 | | |  | 0% |
| Code |  |  | 6 | | |  | 7 | | |  | 100% |
| Compile |  |  | 0 | | |  | 0 | | |  | 0% |
| Test |  |  | 0 | | |  | 0 | | |  | 0% |
| Total Development |  |  | 6 | | |  | 7 | | |  | 100% |
| After Development |  |  | 0 | | |  | 0 | | |  |  |

|  |  |  |
| --- | --- | --- |
| PSP0.1 Plan Summary Instructions | |  |
| Purpose | To hold the plan and actual data for programs or program parts | |
| General | * ***Use the most appropriate size measure, either LOC or element count.*** * “To Date” is the total actual to-date values for all products developed. | |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. | |
| *Program Size* | * ***Enter the plan added and modified size value (A+M).*** * ***Enter actual base, deleted, modified, reused, total, and new reusable size.*** * ***Calculate actual added size as T-B+D-R and actual added and modified size as A+M.*** * ***Enter to-date reused, added and modified, total, and new reusable size.*** | |
| Time in Phase | * Enter the estimated total development time. * ***Distribute the estimated total time across the development phases according to the To Date % for the most recently developed program.*** * Enter the actual time by phase and the total time. * To Date: Enter the sum of the actual times for this program plus the to-date times from the most recently developed program. * To Date %: Enter the percentage of to-date time in each phase. | |
| Defects Injected | * Enter the actual defects by phase and the total actual defects. * To Date: Enter the sum of the actual defects injected by phase and the to-date values for the most recent previously developed program. * To Date %: Enter the percentage of the to-date defects injected by phase. | |
| Defects Removed | * To Date: enter the actual defects removed by phase plus the to-date values for the most recent previously developed program. * To Date %: Enter the percentage of the to-date defects removed by phase. * After development, record any defects subsequently found during program testing, use, reuse, or modification. | |

PSP Time Recording Log

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Suarez | Date | 24 enero 2015 |
| Program | Tarea 2 | Program # | CSOF5101\_01\_2 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project** | **Phase** | **Start Date and Time** | **Int. Time** | **Stop Date and Time** | **Delta**  **Time** | **Comments** |
| T2-31\_ene | Plan | 15:03 | 15 | 15:28 | 10 |  |
|  | Desi | 15:28 |  | 15:48 | 20 |  |
|  | Code | 16:00 | 15 | 17:42 | 87 |  |
|  | Code | 18:30 |  | 19:10 | 40 |  |
|  | Code | 17:40 |  | 17:50 | 10 |  |
|  | Code | 22:15 |  | 22:25 | 10 |  |
|  | Code | 23:00 |  | 00:00 | 60 |  |
| 1\_feb | Code | 8.05 |  | 8.48 | 43 |  |
|  | Code | 21:00 | 10 | 21:53 | 43 |  |
|  | Com | 21:53 |  | 22:15 | 22 |  |
|  | Test | 22:15 |  | 22:25 | 10 |  |
|  | Post | 22:25 |  | 23:00 | 35 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Time Recording Log Instructions

|  |  |
| --- | --- |
| Purpose | * Use this form to record the time you spend on each project activity. * For the PSP, phases often have only one activity; larger projects usually have multiple activities in a single process phase. * These data are used to complete the Project Plan Summary. * Keep separate logs for each program. |
| General | * Record all of the time you spend on the project. * Record the time in minutes. * Be as accurate as possible. * If you need additional space, use another copy of the form. * If you forget to record the starting, stopping, or interruption time for an activity, promptly enter your best estimate. |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. |
| Project | Enter the program name or number. |
| Phase | Enter the name of the phase for the activity you worked on, e.g. Planning, Design, Test. |
| Start Date and Time | Enter the date and time when you start working on a process activity. |
| Interruption Time | * Record any interruption time that was not spent on the process activity. * If you have several interruptions, enter their total time. * You may enter the reason for the interrupt in comments. |
| Stop Date and Time | Enter the date and time when you stop working on that process activity. |
| Delta Time | Enter the clock time you actually spent working on the process activity, less the interruption time. |
| Comments | Enter any other pertinent comments that might later remind you of any unusual circumstances regarding this activity. |

PSP Defect Recording Log

|  |  |
| --- | --- |
| Defect Types |  |
| 10 Documentation | 60 Checking |
| 20 Syntax | 70 Data |
| 30 Build, Package | 80 Function |
| 40 Assignment | 90 System |
| 50 Interface | 100 Environment |

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Suarez | Date | 4 Feb 2015 |
| Program | Tarea 2 | Program # | CSOF5101\_01\_2 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 31 Ene |  | 1 |  | Calculo |  | code |  | code |  | 3 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 31 Ene |  | 2 |  | Calculo |  | code |  | code |  | 1 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 31 Ene |  | 3 |  | Calculo |  | code |  | code |  | 2 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 31 Ene |  | 4 |  | Calculo |  | Code |  | Code |  | 6 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 31 Ene |  | 5 |  | NullPointer |  | code |  | code |  | 46 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 2 |  | | 1 Feb |  | 6 |  | NullPointer |  | code |  | code |  | 6 |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

PSP Defect Recording Log Instructions

|  |  |
| --- | --- |
| Purpose | * Use this form to hold data on the defects that you find and correct. * These data are used to complete the Project Plan Summary form. |
| General | * Record each defect separately and completely. * If you need additional space, use another copy of the form. |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. |
| Project | * Give each program a different name or number. * For example, record test program defects against the test program. |
| Date | Enter the date on which you found the defect. |
| Number | * Enter the defect number. * For each program or module, use a sequential number starting with 1 (or 001, etc.). |
| Type | * Enter the defect type from the defect type list summarized in the top left corner of the form. * Use your best judgment in selecting which type applies. |
| Inject | * Enter the phase when this defect was injected. * Use your best judgment. |
| Remove | Enter the phase during which you fixed the defect. (This will generally be the phase when you found the defect.) |
| Fix Time | * Enter the time that you took to find and fix the defect. * This time can be determined by stopwatch or by judgment. |
| Fix Ref. | * If you or someone else injected this defect while fixing another defect, record the number of the improperly fixed defect. * If you cannot identify the defect number, enter an X. |
| Description | Write a succinct description of the defect that is clear enough to later remind you about the error and help you to remember why you made it. |

PSP Defect Type Standard

|  |  |  |
| --- | --- | --- |
| **Type Number** | **Type Name** | **Description** |
| 10 | Documentation | Comments, messages |
| 20 | Syntax | Spelling, punctuation, typos, instruction formats |
| 30 | Build, Package | Change management, library, version control |
| 40 | Assignment | Declaration, duplicate names, scope, limits |
| 50 | Interface | Procedure calls and references, I/O, user formats |
| 60 | Checking | Error messages, inadequate checks |
| 70 | Data | Structure, content |
| 80 | Function | Logic, pointers, loops, recursion, computation, function defects |
| 90 | System | Configuration, timing, memory |
| 100 | Environment | Design, compile, test, or other support system problems |

Source programs listing

*Programa: java-getting-started*

*\* Clase: Main.java*

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.\*;

import org.eclipse.jetty.server.Server;

import org.eclipse.jetty.servlet.\*;

import java.net.URI;

import java.net.URISyntaxException;

import java.sql.\*;

public class Main extends HttpServlet {

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

if (req.getRequestURI().endsWith("/db")) {

showDatabase(req,resp);

} else {

showHome(req,resp);

}

}

private void showHome(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

resp.getWriter().print("Hello from Java! ESTO ES UNA PRUEBA");

resp.getWriter().print("----lo repeti");

}

private void showDatabase(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

Connection connection = null;

try {

connection = getConnection();

-------------

*Programa: lab1*

*\* Clase: lab1.java*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package lab1;

import edu.uniandes.ecos.ASE.app.Cal;

import java.io.BufferedReader;

import java.io.Console;

import java.io.IOException;

import java.io.InputStreamReader;

/\*\*

\*

\* @author asan123456

\*/

public class Lab1 {

/\*\*

\* @param args the command line arguments

\*/

/\*

public static void main(String[] args) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

//System.out.print("Enter String: ");

//String s = br.readLine();

boolean b= true;

System.out.print("Enter Integer: ");

//try{

do{

String s = br.readLine();

if(isNumeric(s)){

int i = Integer.parseInt(br.readLine());

}

else{

b = false;

}

}while(b);

System.out.print("Aca salio");

//}catch(NumberFormatException nfe){

// System.err.println("Invalid Format!");

//}

}\*/

public static void main(String[] args) throws IOException{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Caso 1 o 2: ");

Cal c = new Cal(br.readLine());

System.out.print("Mean: " + c.meanCalc());

System.out.print("\nStd Dev: " + c.stdDev() + "\n");

}

public static boolean isNumeric(String str)

{

return str.matches("-?\\d+(\\.\\d+)?"); //match a number with optional '-' and decimal.

}

}

---------------

*Programa: psp0*

*\* Clase: AppTest.java*

package edu.uniandes.ecos.ASE.app;

import junit.framework.Test;

import junit.framework.TestCase;

import junit.framework.TestSuite;

/\*\*

\* Unit test for simple App.

\*/

public class MainTest

extends TestCase

{

/\*\*

\* Create the test case

\*

\* @param testName name of the test case

\*/

public MainTest( String testName ){

super( testName );

}

/\*\*

\* @return the suite of tests being tested

\*/

public static Test suite(){

return new TestSuite( MainTest.class );

}

/\*\*

\* Rigourous Test :-)

\*/

public void testApp(){

assertTrue( true );

}

}

---------

*Programa: tarea2 v1*

* *Clase: ClassInfo.java*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.util.LinkedList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* @author asan123456

\*/

public class ClassInfo {

private String className;

private String location;

private int totalSize;

private int methodCount;

private List<String> line;

public ClassInfo(String location){

//this.className = className;

this.location = location.trim();

totalSize = 0;

methodCount = 0;

line = new LinkedList<String>();

try {

calcTotalSize();

} catch (IOException ex) {

Logger.getLogger(ClassInfo.class.getName()).log(Level.SEVERE, null, ex);

}

setClassName();

calcMethodCount();

}

/\*\*

\* @return the className

\*/

public String getClassName() {

return className;

}

/\*\*

\* @param className the className to set

\*/

private void setClassName() {

for (int i = location.length()-5; i>0;i-- ) {

if(location.charAt(i) == '/'){

this.className = location.substring(i+1, location.length());

break;

}

}

}

/\*\*

\* @return the totalSize

\*/

public int getTotalSize() {

return totalSize;

}

/\*\*

\* @return the methodCount

\*/

public int getMethodCount() {

return methodCount;

}

/\*\*

\* @param methodCount the methodCount to set

\*/

public void setMethodCount(int methodCount) {

this.methodCount = methodCount;

}

/\*\*

\* @return the location

\*/

public String getLocation() {

return location;

}

/\*\*

\* @param location the location to set

\*/

public void setLocation(String location) {

this.location = location;

}

private void calcTotalSize() throws FileNotFoundException, IOException{

String nombreFich = this.location;

File fichero = new File(nombreFich);

BufferedReader fich = new BufferedReader(new FileReader(nombreFich));

String linea;

while((linea = fich.readLine()) != null){

if(!linea.trim().isEmpty()){

totalSize++;

this.line.add(linea);

}

}

}

}

---------

* *Clase: Main.java*

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

import java.net.URI;

import java.net.URISyntaxException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.LinkedList;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.eclipse.jetty.server.Server;

import org.eclipse.jetty.servlet.ServletContextHandler;

import org.eclipse.jetty.servlet.ServletHolder;

/\*\*

\* Hello world!

\*

\*/

public class Main extends HttpServlet

{

//private PathJavaExplorer pje;

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

if (req.getRequestURI().endsWith("/db")) {

showDatabase(req,resp);

} else {

showHome(req,resp);

}

}

private void showHome(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

PathJavaExplorer pje=new PathJavaExplorer();

PrettyPrintingLOC ppl = pje.getPje();

for (int i = 0; i<pje.getPje().getPrinting().size();i++){

resp.getWriter().print("\n " +pje.getPje().getPrinting().get(i));

}

}

private void showDatabase(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

Connection connection = null;

try {

connection = getConnection();

Statement stmt = connection.createStatement();

stmt.executeUpdate("CREATE TABLE IF NOT EXISTS ticks (tick timestamp)");

stmt.executeUpdate("INSERT INTO ticks VALUES (now())");

ResultSet rs = stmt.executeQuery("SELECT tick FROM ticks");

String out = "Hello!\n";

while (rs.next()) {

out += "Read from DB: " + rs.getTimestamp("tick") + "\n";

}

resp.getWriter().print(out);

} catch (Exception e) {

resp.getWriter().print("There was an error: " + e.getMessage());

} finally {

if (connection != null) try{connection.close();} catch(SQLException e){}

}

}

private Connection getConnection() throws URISyntaxException, SQLException {

URI dbUri = new URI(System.getenv("DATABASE\_URL"));

String username = dbUri.getUserInfo().split(":")[0];

String password = dbUri.getUserInfo().split(":")[1];

int port = dbUri.getPort();

String dbUrl = "jdbc:postgresql://" + dbUri.getHost() + ":" + port + dbUri.getPath();

return DriverManager.getConnection(dbUrl, username, password);

}

public static void main(String[] args) throws Exception {

Server server = new Server(Integer.valueOf(System.getenv("PORT")));

ServletContextHandler context = new ServletContextHandler(ServletContextHandler.SESSIONS);

context.setContextPath("/");

server.setHandler(context);

context.addServlet(new ServletHolder(new Main()),"/\*");

server.start();

server.join();

}

}

* *Clase: PathJavaExplorer.java*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.io.File;

import java.util.LinkedList;

import java.util.List;

/\*\*

\*

\* @author asan123456

\*/

public class PathJavaExplorer {

private PrettyPrintingLOC pje;

public PathJavaExplorer(){

String sDirectorio = "Java\_Projects";

File f = new File(sDirectorio);

if (f.exists()){ // Directorio existe

//System.out.println("Directorio existe");

File[] ficheros = f.listFiles();

pje = new PrettyPrintingLOC();

int cont =0;

for (int x=1;x<ficheros.length;x++){

crearProyectos(ficheros[x].getName());

lsDirectory(cont,"Java\_Projects/" + ficheros[x].getName(),ficheros[x]);

cont++;

}

pje.getProjectSize();

pje.prettyPrinting();

}

else {

//Directorio no existe

pje.setPrinting("Directorio no existe");

}

}

private void crearProyectos(String name){

ProjectInfo p = new ProjectInfo(name);

getPje().getProjects().add(p);

}

private List<String> obtenerRutasProyecto(){

List<String> l = new LinkedList<String>();

return l;

}

public void lsDirectory ( int index,String padre,File dir ) {

File[] archivos = dir.listFiles();

int j=0;

//if(archivos !=null)

for ( int i=0 ; i<archivos.length;i++)

{

String ruta= padre+"/"+archivos[i].getName();

if(ruta.contains(".java"))

getPje().getProjects().get(index).pushClass(new ClassInfo(ruta));

if ( archivos[i].isDirectory() ) {

lsDirectory( index,padre+"/"+archivos[i].getName(),archivos[i] );

}

}

}

/\*\*

\* @return the pje

\*/

public PrettyPrintingLOC getPje() {

return pje;

}

}

* *Clase: PrettyPrintingLOC.java*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.util.LinkedList;

import java.util.List;

/\*\*

\*

\* @author asan123456

\*/

public class PrettyPrintingLOC {

private List<ProjectInfo> projects;

private List<String> printing;

public PrettyPrintingLOC(){

projects = new LinkedList<ProjectInfo>();

printing = new LinkedList<String>();

}

/\*\*

\* @return the projects

\*/

public List<ProjectInfo> getProjects() {

return projects;

}

/\*\*

\* @param projects the projects to set

\*/

public void setProjects(List<ProjectInfo> projects) {

this.projects = projects;

}

public int getIndexProjectbyName(String name){

int indx= -1;

for(int i = 0 ; i< projects.size();i++){

if(projects.get(i).getProgramName().equals(name)){

indx = i;

}

}

return indx;

}

public void getProjectSize(){

for(int i = 0; i < projects.size() ;i++ ){

projects.get(i).calcTotalSize();

}

}

public void prettyPrinting(){

for(int i = 0 ; i< projects.size();i++){

getPrinting().add("-----------------------------------------------");

getPrinting().add("PROGRAM NAME: " + projects.get(i).getProgramName());

getPrinting().add("Total size: " + projects.get(i).getTotalSize());

for(int j = 0 ; j< projects.get(i).getClasss().size();j++){

getPrinting().add(" Class name: " + projects.get(i).getClasss().get(j).getClassName());

getPrinting().add(" Method count: " + projects.get(i).getClasss().get(j).getMethodCount() );

getPrinting().add(" Location: " + projects.get(i).getClasss().get(j).getLocation());

getPrinting().add(" Size: " + projects.get(i).getClasss().get(j).getTotalSize());

getPrinting().add(" -----------------------------------------------");

}

}

}

/\*\*

\* @return the printing

\*/

public List<String> getPrinting() {

return printing;

}

/\*\*

\* @param printing the printing to set

\*/

public void setPrinting(String msj) {

this.printing.add(msj);

}

}

* *Clase: ProjectInfo.java*

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.util.LinkedList;

import java.util.List;

/\*\*

\*

\* @author asan123456

\*/

public class ProjectInfo {

private String programName;

private List<ClassInfo> classs;

private int totalSize;

public ProjectInfo(String programName){

this.programName = programName;

this.classs = new LinkedList<ClassInfo>();

this.totalSize = 0;

}

/\*\*

\* @return the classs

\*/

public List<ClassInfo> getClasss() {

return classs;

}

/\*\*

\* @param classs the classs to set

\*/

public void setClasss(List<ClassInfo> classs) {

this.classs = classs;

}

/\*\*

\* @return the totalSize

\*/

public int getTotalSize() {

return totalSize;

}

/\*\*

\* @return the programName

\*/

public String getProgramName() {

return programName;

}

public void pushClass(ClassInfo c){

classs.add(c);

}

public void calcTotalSize(){

for (int i =0;i<classs.size();i++){

this.totalSize = this.totalSize+classs.get(i).getTotalSize();

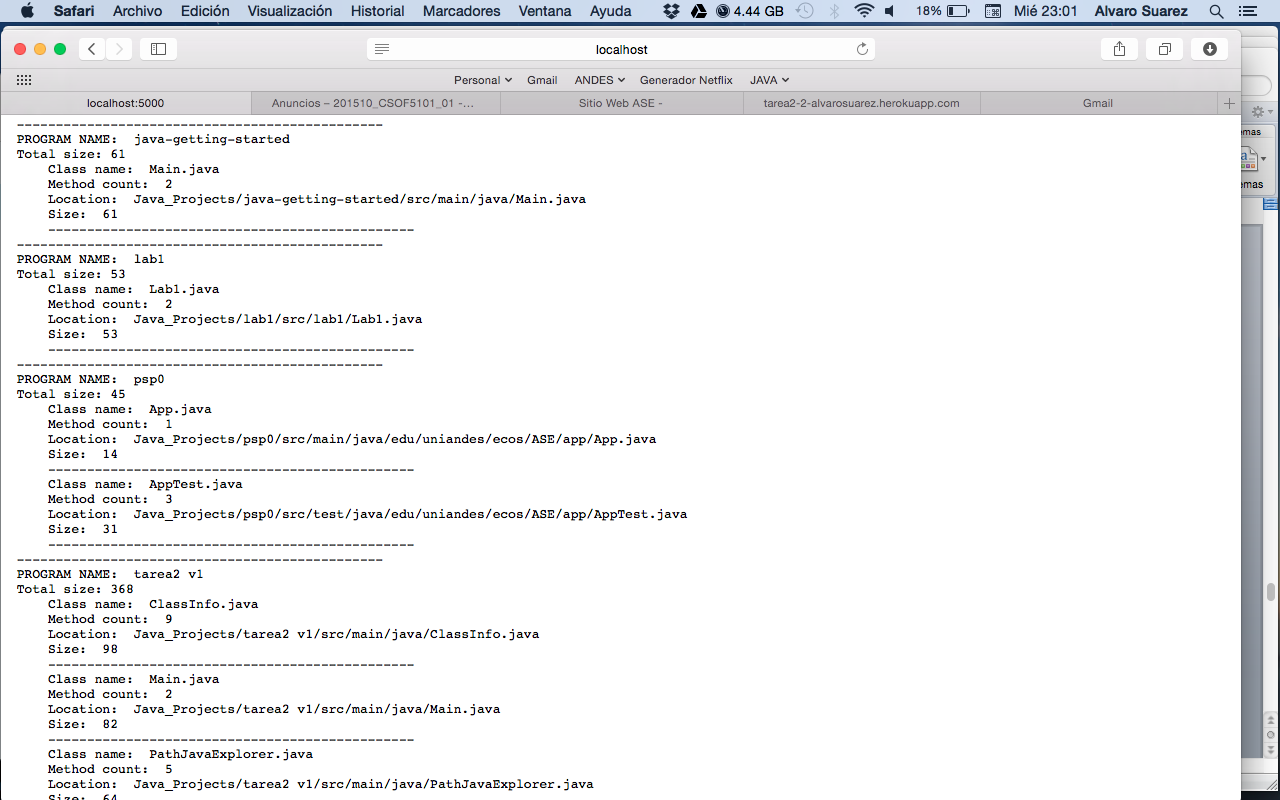
}

}

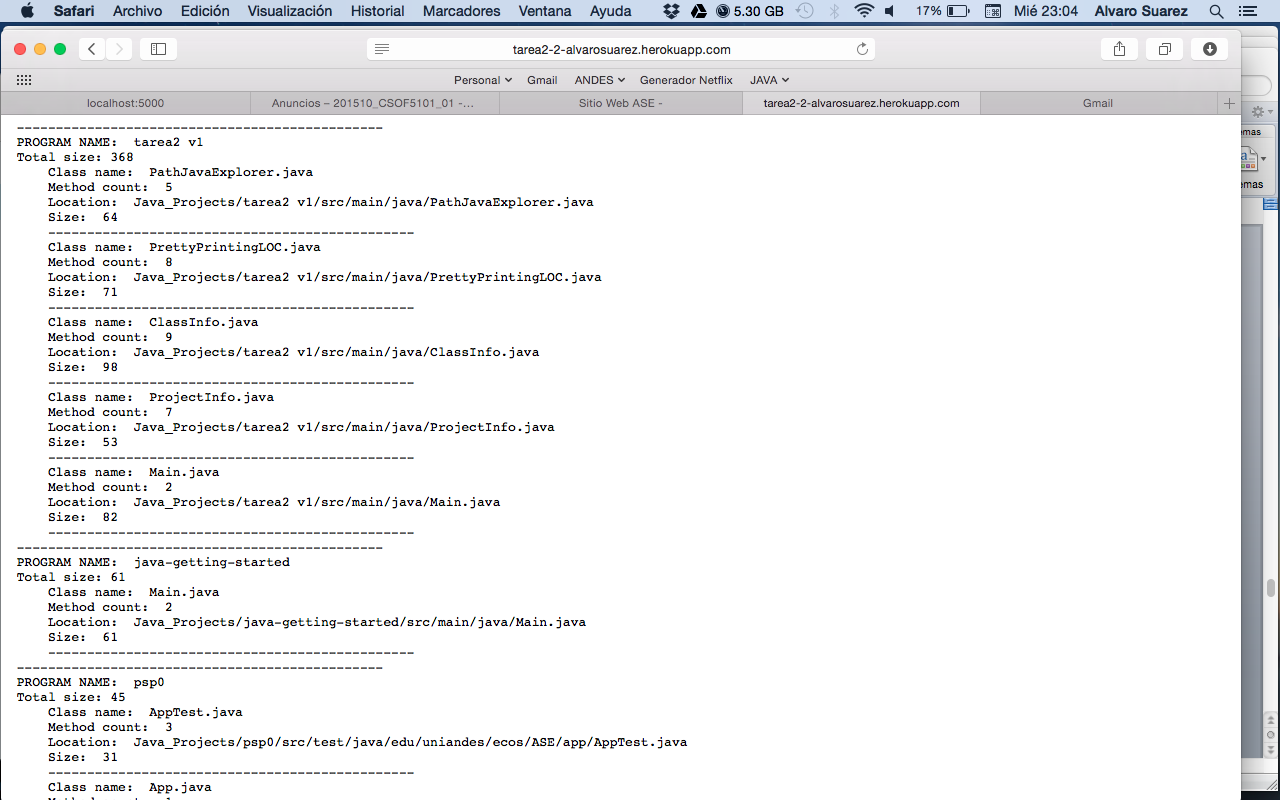
}

Test results

Pruebas Locales:

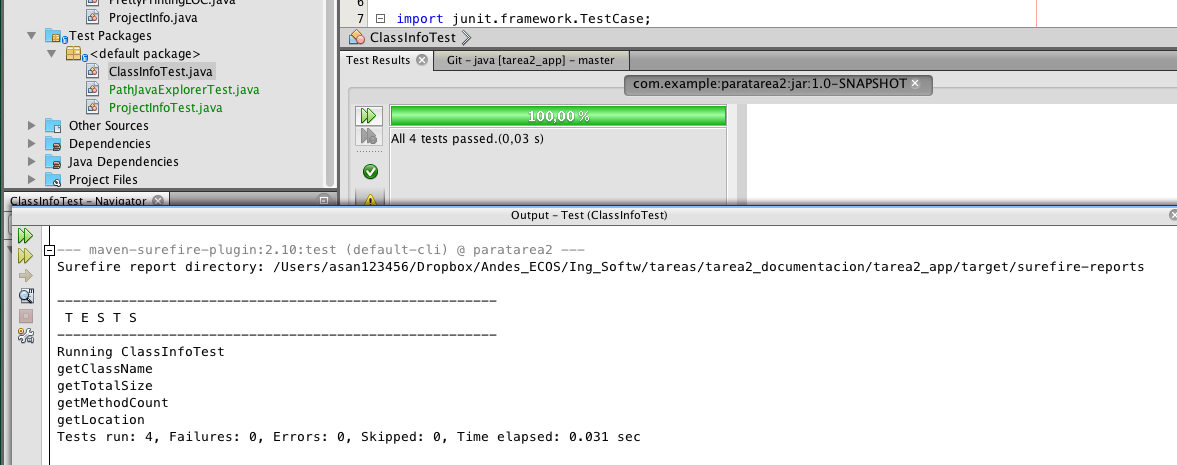


Pruebas en la nube:



Pruebas Unitarias:

ClassInfoTest



* Clase: ClassInfoTest

import junit.framework.TestCase;

/\*\*

\*

\* @author asan123456

\*/

public class ClassInfoTest extends TestCase {

//private ClassInfo instance;

public ClassInfoTest(String testName) {

super(testName);

}

@Override

protected void setUp() throws Exception {

super.setUp();

}

@Override

protected void tearDown() throws Exception {

super.tearDown();

}

/\*\*

\* Test of getClassName method, of class ClassInfo.

\*/

public void testGetClassName() {

System.out.println("getClassName");

ClassInfo instance = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

String expResult = "Lab1.java";

String result = instance.getClassName();

assertTrue("testGetClassName:fail",expResult.equals(result));

}

/\*\*

\* Test of getTotalSize method, of class ClassInfo.

\*/

public void testGetTotalSize() {

System.out.println("getTotalSize");

ClassInfo instance = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

int expResult = 53;

int result = instance.getTotalSize();

assertTrue("testGetTotalSize:fail",expResult== (result));

// TODO review the generated test code and remove the default call to fail.

//fail("The test case is a prototype.");

}

/\*\*

\* Test of getMethodCount method, of class ClassInfo.

\*/

public void testGetMethodCount() {

System.out.println("getMethodCount");

ClassInfo instance = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

int expResult = 2;

int result = instance.getMethodCount();

assertTrue("testGetMethodCount:fail",expResult== (result));

}

/\*\*

\* Test of getLocation method, of class ClassInfo.

\*/

public void testGetLocation() {

System.out.println("getLocation");

ClassInfo instance = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

String expResult = "Java\_Projects/lab1/src/lab1/Lab1.java";

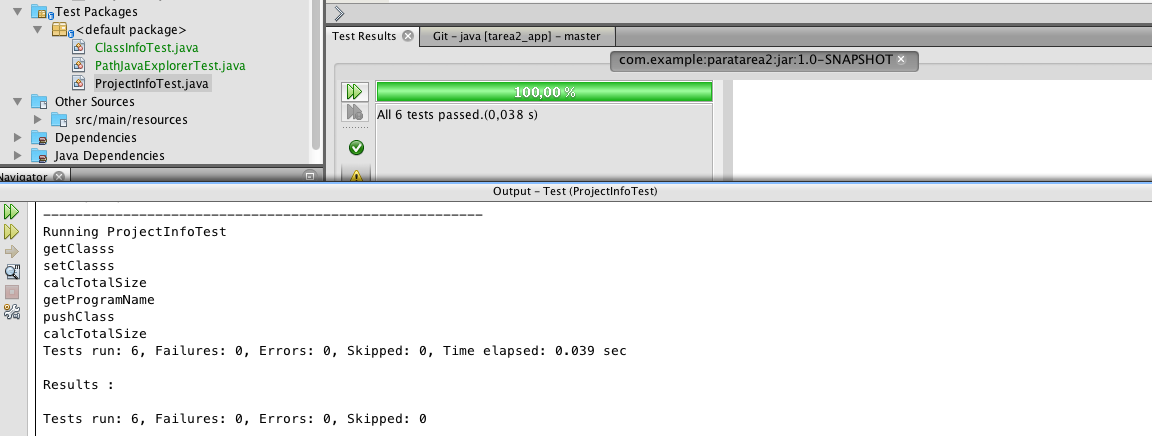
String result = instance.getLocation();

assertTrue("testGetLocation:fail",expResult.equals(result));

}

}

* Clase: ProjectInfoTest



/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.util.LinkedList;

import java.util.List;

import junit.framework.TestCase;

/\*\*

\*

\* @author asan123456

\*/

public class ProjectInfoTest extends TestCase {

public ProjectInfoTest(String testName) {

super(testName);

}

@Override

protected void setUp() throws Exception {

super.setUp();

}

@Override

protected void tearDown() throws Exception {

super.tearDown();

}

/\*\*

\* Test of getClasss method, of class ProjectInfo.

\*/

public void testGetClasss() {

System.out.println("getClasss");

ProjectInfo instance = new ProjectInfo("Prg1");

List<ClassInfo> expResult = new LinkedList<ClassInfo>();

ClassInfo c = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

ClassInfo c2 = new ClassInfo("Java\_Projects/psp0/src/main/java/edu/uniandes/ecos/ASE/app/App.java");

expResult.add(c);

expResult.add(c2);

instance.setClasss(expResult);

List<ClassInfo> result = instance.getClasss();

assertTrue("testGetClasss:fail",expResult.get(1).getClassName().equals(result.get(1).getClassName()));

}

/\*\*

\* Test of setClasss method, of class ProjectInfo.

\*/

public void testSetClasss() {

System.out.println("setClasss");

List<ClassInfo> classs = new LinkedList<ClassInfo>();

ClassInfo c = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

ClassInfo c2 = new ClassInfo("Java\_Projects/psp0/src/main/java/edu/uniandes/ecos/ASE/app/App.java");

classs.add(c);

classs.add(c2);

ProjectInfo instance = new ProjectInfo("Prg1");

instance.setClasss(classs);

assertTrue("testSetClasss:fail",instance.getClasss().size() == 2);

}

/\*\*

\* Test of getTotalSize method, of class ProjectInfo.

\*/

public void testGetTotalSize() {

System.out.println("calcTotalSize");

List<ClassInfo> classs = new LinkedList<ClassInfo>();

ClassInfo c = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

ClassInfo c2 = new ClassInfo("Java\_Projects/psp0/src/main/java/edu/uniandes/ecos/ASE/app/App.java");

classs.add(c);

classs.add(c2);

ProjectInfo instance = new ProjectInfo("Prg1");

instance.setClasss(classs);

instance.calcTotalSize();

assertTrue("testGetTotalSize:fail",instance.getTotalSize()== 67);//67

}

/\*\*

\* Test of getProgramName method, of class ProjectInfo.

\*/

public void testGetProgramName() {

System.out.println("getProgramName");

ProjectInfo instance = new ProjectInfo("Prg1");

String expResult = "Prg1";

String result = instance.getProgramName();

assertTrue("testGetProgramName:fail",expResult.equals(result));

}

/\*\*

\* Test of pushClass method, of class ProjectInfo.

\*/

public void testPushClass() {

System.out.println("pushClass");

ClassInfo c = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

ProjectInfo instance = new ProjectInfo("Prg1");

instance.pushClass(c);

assertTrue("testPushClass:fail",instance.getClasss().get(0).getClassName().equals("Lab1.java"));

}

/\*\*

\* Test of calcTotalSize method, of class ProjectInfo.

\*/

public void testCalcTotalSize() {

System.out.println("calcTotalSize");

List<ClassInfo> classs = new LinkedList<ClassInfo>();

ClassInfo c = new ClassInfo("Java\_Projects/lab1/src/lab1/Lab1.java");

ClassInfo c2 = new ClassInfo("Java\_Projects/psp0/src/main/java/edu/uniandes/ecos/ASE/app/App.java");

classs.add(c);

classs.add(c2);

ProjectInfo instance = new ProjectInfo("Prg1");

instance.setClasss(classs);

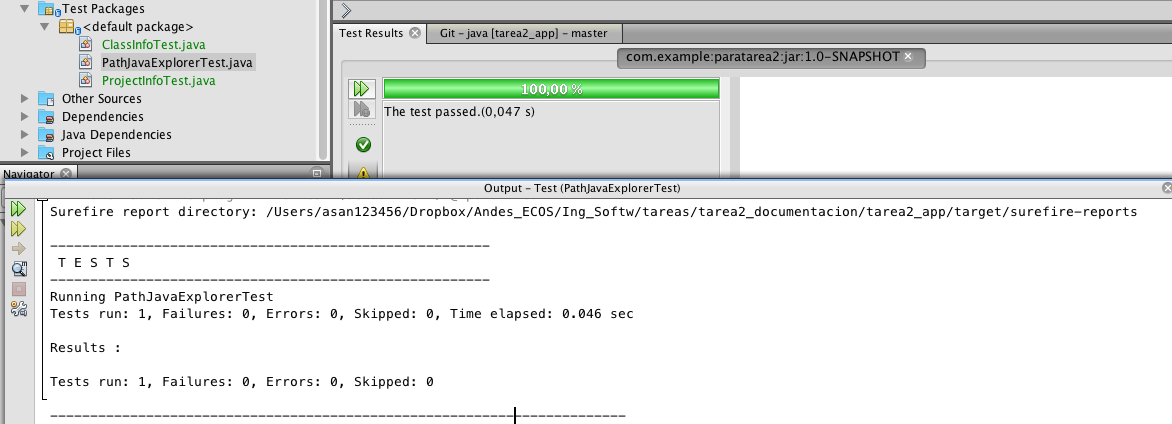
instance.calcTotalSize();

assertTrue("testCalcTotalSize:fail",instance.getTotalSize()== 67);//67

}

}

* Clase PathJavaExplorerTest



/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.io.File;

import junit.framework.TestCase;

/\*\*

\*

\* @author asan123456

\*/

public class PathJavaExplorerTest extends TestCase {

public PathJavaExplorerTest(String testName) {

super(testName);

}

@Override

protected void setUp() throws Exception {

super.setUp();

}

@Override

protected void tearDown() throws Exception {

super.tearDown();

}

/\*\*

\* Test of lsDirectory method, of class PathJavaExplorer.

\*/

public void testFullPathJavaExplorer() {

String name = "tarea2 v1";

PathJavaExplorer instance = new PathJavaExplorer();

//System.out.println("lsDirectory:::"+ instance.getPje().getProjects().get(0).getProgramName().equals(name));

assertTrue("testFullPathJavaExplorer:fail",name.equals( instance.getPje().getProjects().get(3).getProgramName()));

}

}