Spring Stack Testing: Continuous integration, Continuous Agitation

Massimiliano Dessì CRS4
http://wiki.java.net/bin/view/People/MassimilianoDessi

Lo speaker

- Consulente Java, Spring e Jetspeed per Società e Università
- Co-fondatore e consigliere Java User Group Sardegna (2002)
- Fondatore e coordinatore degli User Group:

  Spring Framework Italian User Group (70 utenti)

  Jetspeed Italian User Group (27 utenti)
- Jug Avis Web (Spring) Lead
- Vari talk tecnici e articoli dal 2003: Mokabyte, Dev, Java Journal, Programmazione.it, Jug Sardegna, Java Conference
- Utilizza Spring da Luglio 2004, ad oggi 12 progetti: Turismo, Banche, Open Source, Pubblica Amministrazione

Spring Framework

Leading full-stack Java/J2EE application framework, Spring delivers significant benefits for many projects, reducing development effort and costs while improving test coverage and quality.

We believe not only that J2EE development should be much simpler than the mixture of drudgery and complexity it's often made out to be, but that developing J2EE applications should be ***fun***.

Non sono tutte rose e fiori...

Purtroppo utilizzare software Open Source non significa automaticamente che il proprio progetto sia di ottima qualità. Se è vero che usando Spring si scrive meno codice, non è detto che “magicamente” le nostre applicazioni diventino migliori. Spring ha già di suo molte caratteristiche “sotto il cofano”, come l'uso di variabili ThreadLocal, molte classi Template per l'utilizzo di Api JEE, l'uso di default di singole istanze di oggetti condivise sul modello delle Servlet, astrazioni, programmazione completamente ad Interfacce, programmazione ad aspetti, eppure tutto ciò può essere usato male.

Dove si può sbagliare ?

Il primo rischio è usare Spring ma scrivere il codice alla maniera “ortodossa” JEE, Spring diventa solo un jar in più nel classpath, in questo modo Spring diventa solo una sigla per il marketing, e si perde il valore aggiunto che può portare.

Il secondo è usare di usare Spring, ma non scrivere codice OO.

Il terzo è di farsi generare interamente il codice da tool appositi, lo sviluppo non è più nelle mani del team di sviluppo, ma di chi ha fatto il tool di generazione, magari vengono generati pure i test, lo sviluppo guidato dai wizard...

Ho cambiato un jar e il tool di generazione è esploso !

Ho fatto un commit e non funziona più nulla !

Continuous Stress...

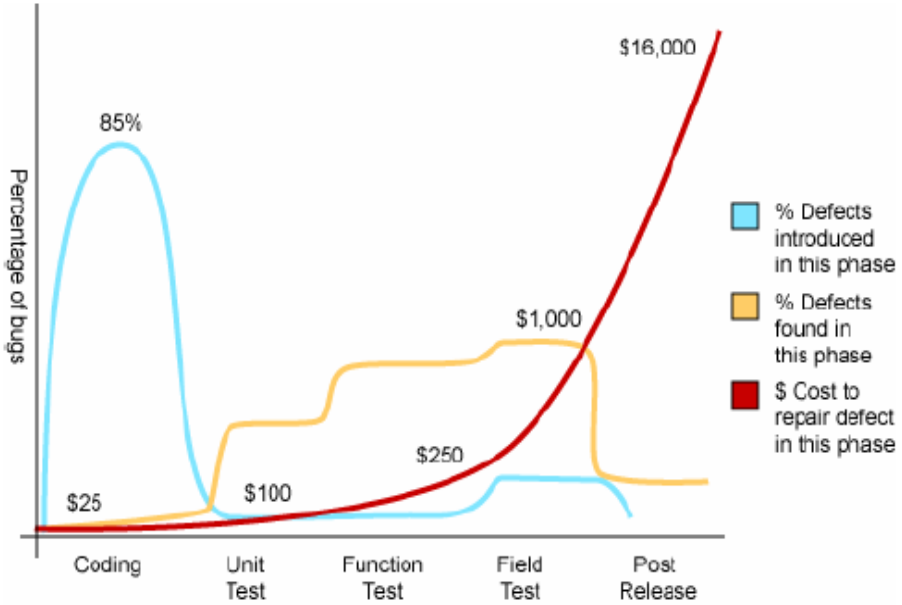
Test: No Fear !

Aiutano a scrivere codice migliore, perchè i difetti vengono individuati prima.

Codice difficile da testare avrà costi maggiori quando i difetti verranno trovati successivamente.

Aumenta la percezione della qualità effettiva del codice scritto.

Diminuisce il costo degli errori.



Source: Applied Software Measurement, Capers Jones, 1996

Buoni valori, Buone Abitudini

- Programmazione OO
- Evitare complessità non necessaria, la cosa più semplice che possa funzionare (Xp)
- Facile da testare (aumenta la conoscenza del codice scritto facilitando i miglioramenti e il refactoring)

Cosa si può fare per avere buone abitudini ?

- Pensare le applicazioni completamente ad oggetti, classi con una sola responsabilità.

  Una classe con un metodo da 500 righe anche se funziona mi presenterà il prezzo del suo mantenimento in futuro...
- Programmare per interfacce, oltre a poter migliorare singole porzioni di applicazione cambiando l'implementazione, verrà più semplice anche scrivere i test
- Lasciare che i test guidino lo sviluppo, non i wizard dei tool.

TDD

- Scrivo meno codice perchè scrivo solo quello necessario
- I bachi non ritornano
- Le nuove funzionalità non rompono il funzionamento del sistema
- Feedback immediato
- L'applicazione avrà la forma più semplice possibile

Tradotto: Lavori meglio e sei più produttivo

Applicazioni Risorse Sistema 800 MHz ven 27 ott, 15.30

Java - jugavis-data.xml - Eclipse SDK

File Edit Navigate Search Project Run Tomcat Window Help

Package Explorer | Hierarchy | JUnit

Finished after 17,037 seconds

Runs: 23/23 Errors: 0 Failures: 0

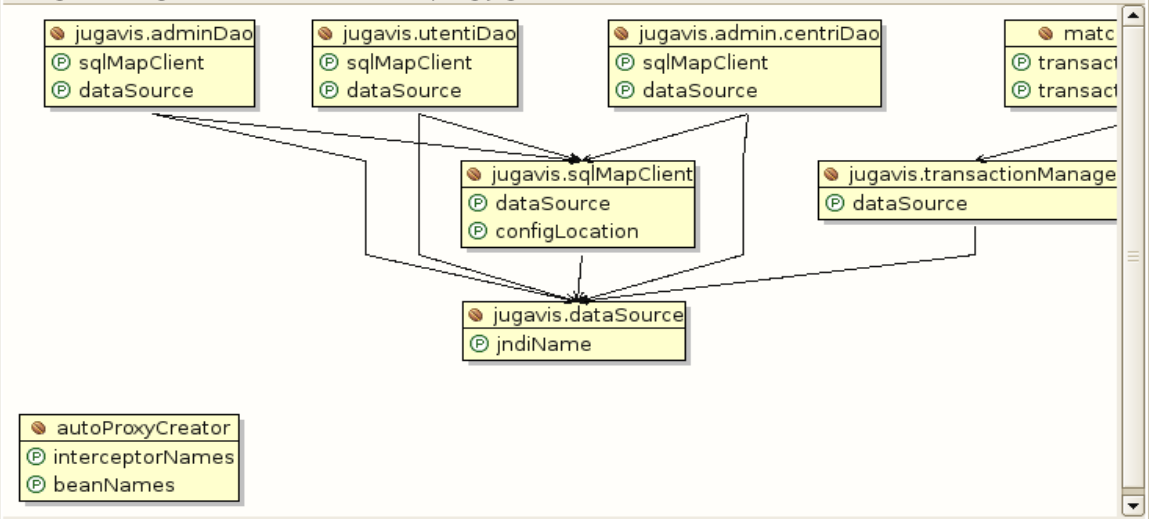
Test for Jug Avis [Runner: JUnit 3]

- test.org.jugsardegna.avis.web.admin.dao.ibatis.SqlMapCentriTest
- test.org.jugsardegna.avis.web.admin.dao.ibatis.SqlMapUtentiTest
- test.org.jugsardegna.avis.web.admin.controller.CentroControllerTest
- test.org.jugsardegna.avis.web.admin.validator.UtenteValidatorTest
- test.org.jugsardegna.avis.web.admin.validator.CentroValidatorTest
- test.org.jugsardegna.avis.web.admin.service.UtentiManagerTest

Failure Trace

build.xml | jugavis-data.xml | jugavis-servlet.xml | jugavis-web.xml

Config File: /magic-box/www/WEB-INF/conf/spring/jugavis-data.xml



Problems | Console | Ant | Metrics - AllTest.java | Spring Beans | Coverage

type filter text

- magic-box
  - www/WEB-INF/conf/spring/jugavis-data.xml
  - www/WEB-INF/conf/spring/jugavis-services.xml
  - www/WEB-INF/conf/spring/jugavis-web.xml
  - www/WEB-INF/jugavis-servlet.xml

ld2k6Dessi - Ope... | Java - jugavis-dat...

Tipi di Test :Unitari

- Unitari: Testo le classi senza Spring o altri container,
- usando dei MockObject dove necessario.

Controller:

```
public void testDettaglio() throws Exception {

    mockCentriManager.expects(once())
    .method("getCentro").with(eq(new Long(13))).will(returnValue(MioMockCentro.getMioCentroMock()));

    req.setMethod("GET");
    req.addParameter("id", "13");
    ModelAndView mav = controller.dettaglio(req, res);

    assertNotNull(mav);
    assertEquals(mav.getViewName(),"admin/dettaglioCentro");
    assertTrue(mav.getModel().containsValue(MioMockCentro.getMioCentroMock()));
    assertTrue(mav.getModel().containsKey(Costanti.CENTRO));
    mockCentriManager.verify();
}
```

Tipi di Test : integrazione

Utilizzando Spring, esattamente come a runtime

```
public void setUp() throws Exception {
    String[] paths = { "/test/org/jugsardegna/avis/web/jugavis-data.xml",
            "/test/org/jugsardegna/avis/web/jugavis-services.xml" };
     ctx = new ClassPathXmlApplicationContext(paths);
     utentiManager = (IUtentiManager)  ctx.getBean("jugavis.utentiService");
     centriManager = (ICentriManager)  ctx.getBean("jugavis.admin.centriService");
     mioMockCentro = MioMockCentro.getMioCentroMock();
     mioMockUtente = MioMockUtente.getMioUtenteMock();
}

public void testInsertUtente() {
    long idCentro = (long)  centriManager.insertCentro( mioMockCentro);
     utentiManager.insertUtente( mioMockUtente, idCentro);

    IUtente recuperato =   utentiManager.getUtente( mioMockUtente.getTessera(),idCentro);

    assertNotNull(recuperato);
    assertEquals( mioMockUtente, recuperato);

     utentiManager.deleteUtente( mioMockUtente.getTessera(),idCentro);
     centriManager.deleteCentro(idCentro);
}
```

Code Coverage

- Una volta scritti i test, in maniera automatica devo avere indicazione sulla percentuale della applicazione coperta dai test.

- Ho il feedback su quali oggetti devo ancora testare, o quali metodi ancora non testati.

Applicazioni Risorse Sistema 800 MHz ven 27 ott, 15.32

Java - CentroControllerTest.java - Eclipse SDK

File Edit Source Refactor Navigate Search Project Run Tomcat Window Help

Package Explorer | Hierarchy | JUnit

Finished after 18,529 seconds

Runs: 23/23 Errors: 0 Failures: 0

Test for Jug Avis [Runner: JUnit 3]

- test.org.jugsardegna.avis.web.admin.dao.ibatis.SqlMapCe
- test.org.jugsardegna.avis.web.admin.dao.ibatis.SqlMapUt
- test.org.jugsardegna.avis.web.admin.controller.CentroCon
- test.org.jugsardegna.avis.web.admin.validator.UtenteValic
- test.org.jugsardegna.avis.web.admin.validator.CentroValic
- test.org.jugsardegna.avis.web.admin.service.UtentiManag

Failure Trace

jugavis-data.xml | jugavis-servlet.xml | jugavis-web.xml | CentroController...

```
public void testDettaglio() throws Exception {

        mockCentriManager.expects(once())
        .method("getCentro").with(eq(new Long(13))).will(returnValue(MioMockCentro.getMioCentro

        req.setMethod("GET");
        req.addParameter("id", "13");
        ModelAndView mav = controller.dettaglio(req, res);

        assertNotNull(mav);
        assertEquals(mav.getViewName(),"admin/dettaglioCentro");
        assertTrue(mav.getModel().containsValue(MioMockCentro.getMioCentroMock()));
        assertTrue(mav.getModel().containsKey(Costanti.CENTRO));
        mockCentriManager.verify();
    }
```

Problems | Console | Ant | Metrics - CentroControllerTest | Spring Beans | Coverage

AllTest (27-ott-2006 15.31.49)

Element	Coverage	Covered Instructi	Total Instructi
test.org.jugsardegna.avis.web.admin.controller	100,0 %	281	
CentroControllerTest.java	100,0 %	281	
CentroControllerTest	100,0 %	281	
setUp()	100,0 %	34	
testConferma()	100,0 %	56	
testDettaglio()	100,0 %	56	
testElenco()	100,0 %	52	
testElimina()	100,0 %	80	
test.org.jugsardegna.avis.web.admin.dao.ibatis	98,9 %	518	

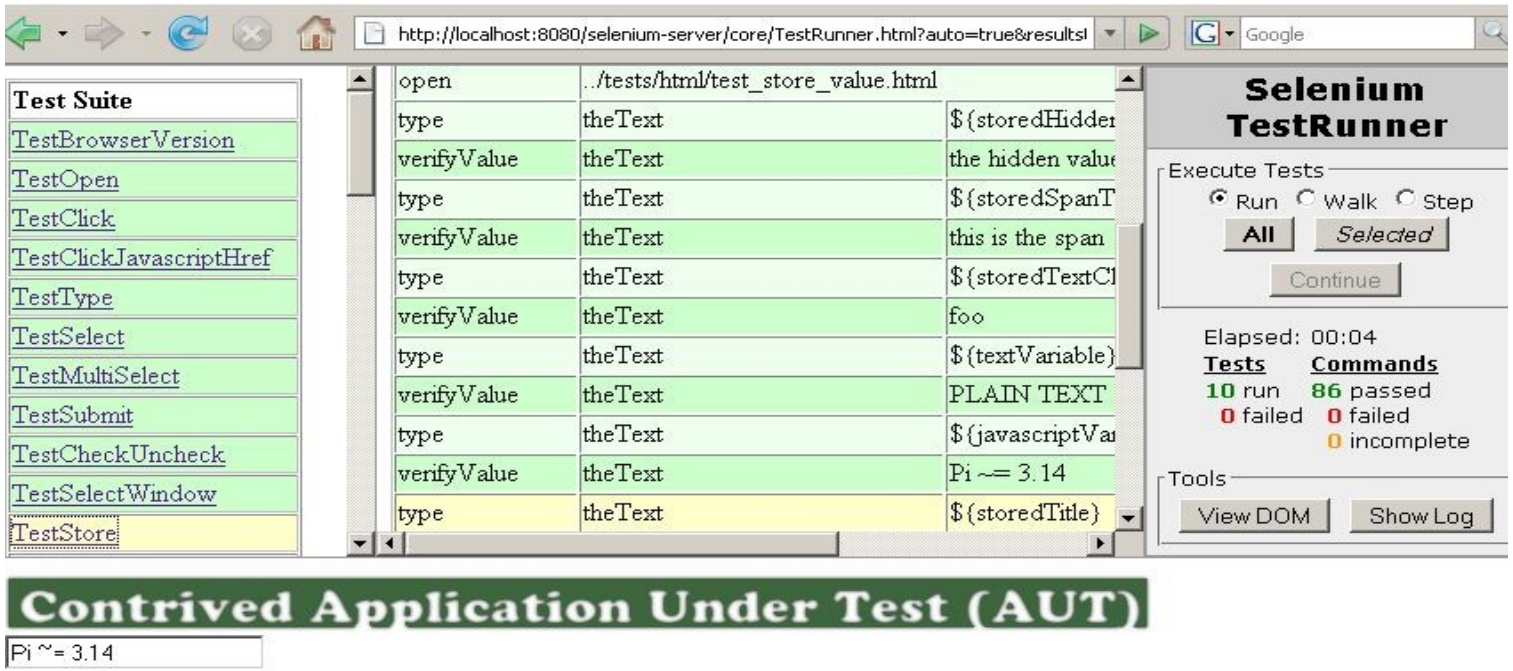
ld2k6Dessi - OpenOffi... | Java - CentroControlle...

Test di Accettazione

- Sono quelli che il cliente chiede perchè ai suoi occhi il sistema soddisfi i requisiti richiesti.

  Può scrivere lui stesso i requisiti pur non essendo un tecnico

- I test vengono "scritti" anche con un plugin per firefox



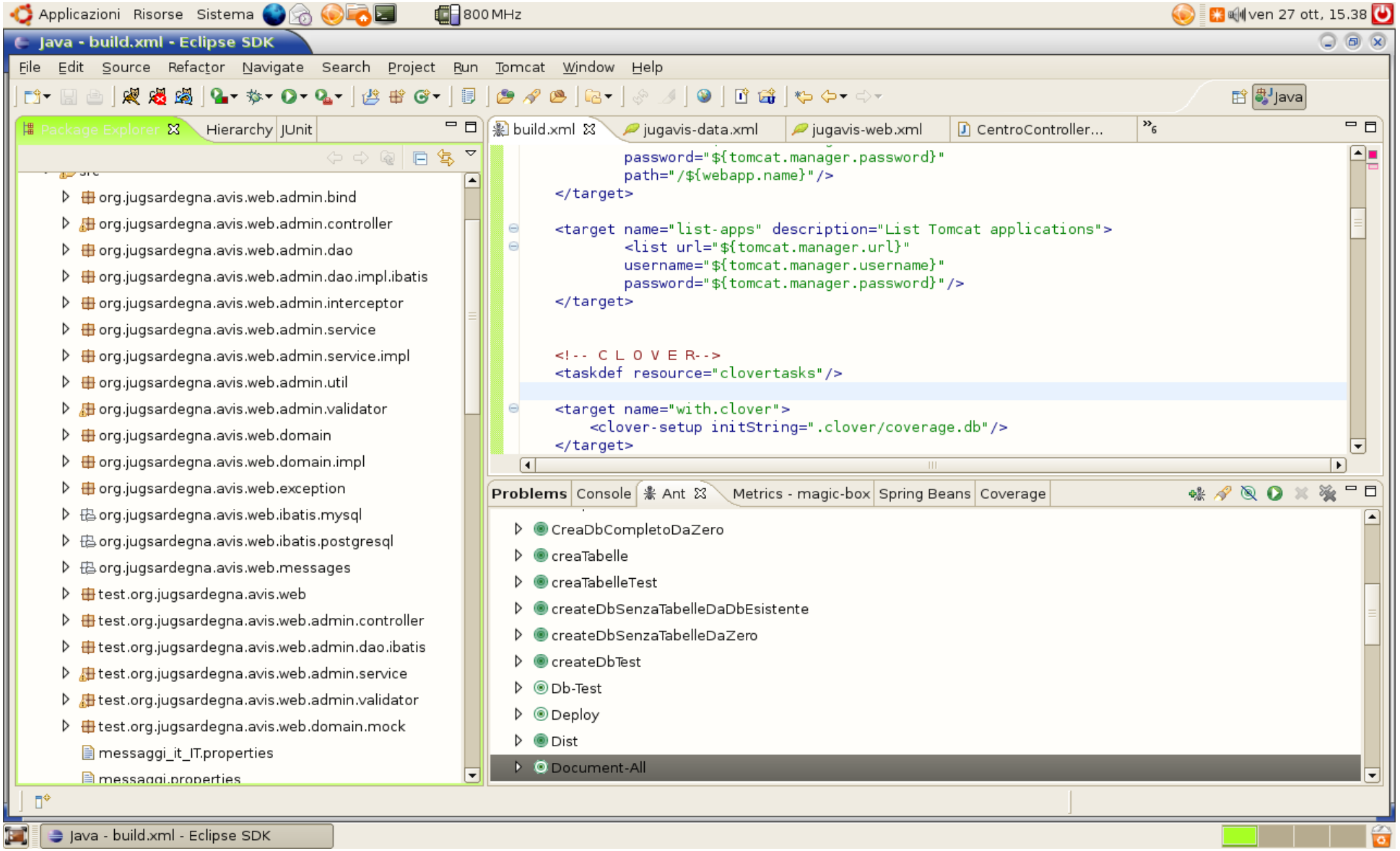
Integrazione Continua

- Integro spesso il mio lavoro con quello del team
- Ho un feedback rapido sui cambiamenti (report, indicatori)
- Vengono mitigati i costi degli errori
- Ad intervalli regolari, il sistema di build continuo, verificata immediatamente i cambiamenti, se il build e i test hanno successo il codice viene “taggato”.

Imperfect tests, run frequently, are much better than perfect tests that are never written at all.
Martin Fowler

- Con ant viene automatizzata:
- la creazione del db, l'inserimento dei dati di test,
- l'esecuzione della suite di test dell'intera applicazione
- Il code Coverage con Clover o Emma,
- la generazione dei report di Junit di Clover e di Emma
- L' esecuzione dei test di accettazione con Selenium

  , e la generazione dei report con i risultati
- La documentazione sulla configurazione (beandoc) e delle classi (Javadoc)
- Il deploy della applicazione


Applicazioni Risorse Sistema
800 MHz
ven 27 ott, 15.38
Java - build.xml - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Tomcat Window Help
Java
Package Explorer
Hierarchy
JUnit
org.jugsardegna.avis.web.admin.bind
org.jugsardegna.avis.web.admin.controller
org.jugsardegna.avis.web.admin.dao
org.jugsardegna.avis.web.admin.dao.impl.ibatis
org.jugsardegna.avis.web.admin.interceptor
org.jugsardegna.avis.web.admin.service
org.jugsardegna.avis.web.admin.service.impl
org.jugsardegna.avis.web.admin.util
org.jugsardegna.avis.web.admin.validator
org.jugsardegna.avis.web.domain
org.jugsardegna.avis.web.domain.impl
org.jugsardegna.avis.web.exception
org.jugsardegna.avis.web.ibatis.mysql
org.jugsardegna.avis.web.ibatis.postgresql
org.jugsardegna.avis.web.messages
test.org.jugsardegna.avis.web
test.org.jugsardegna.avis.web.admin.controller
test.org.jugsardegna.avis.web.admin.dao.ibatis
test.org.jugsardegna.avis.web.admin.service
test.org.jugsardegna.avis.web.admin.validator
test.org.jugsardegna.avis.web.domain.mock
messaggi_it_IT.properties
messaggi.properties
build.xml
jugavis-data.xml
jugavis-web.xml
CentroController...
password="${tomcat.manager.password}"
path="/${webapp.name}"/>
</target>
<target name="list-apps" description="List Tomcat applications">
<list url="${tomcat.manager.url}"
username="${tomcat.manager.username}"
password="${tomcat.manager.password}"/>
</target>
<!-- C L O V E R-->
<taskdef resource="clovertasks"/>
<target name="with.clover">
<clover-setup initString=".clover/coverage.db"/>
</target>
Problems
Console
Ant
Metrics - magic-box
Spring Beans
Coverage
CreaDbCompletoDaZero
creaTabelle
creaTabelleTest
createDbSenzaTabelleDaDbEsistente
createDbSenzaTabelleDaZero
createDbTest
Db-Test
Deploy
Dist
Document-All
Java - build.xml - Eclipse SDK

Luntbuild

Ora è necessario che i task di ant vengano eseguiti, in maniera automatica e schedulata, per poter creare un ambiente di integrazione continua con cui si può avere un feedback immediato sulla intera applicazione

LUNTBUILD 1.3.2

Home > project - Jug Avis Web REFRESH IS OFF system log logout

Basic | VCS adaptors | Builders | Schedules | Login mapping [luntbuild]

Ant	
Builder type	Ant builder
Command to run Ant	C:/java_home/apache-ant-1.6.5/bin/ant.bat
Build script path	D:/workspace/magic-box/build.xml
Build targets	
Build properties	buildVersion="${build.version}" artifactsDir="${build.artifactsDir}" buildDate="${build.startDate}" junitHtmlReportDir="${build.junitHtmlReportDir}"

LUNTBUILD 1.3.2

Home REFRESH IS OFF system log logout

Builds | Projects | Users | Properties | Administration [luntbuild]

This page shows build information for all projects. Status of a schedule and build is denoted using icon: GREEN icon means success, the animation gear icon means building, and RED icon means build failed. Schedule status is different from build status, and it means whether or not the schedule has been successfully triggered. Trigger of the schedule may or may not generate a new build, it depends on the current build strategy and repository changes. The system log and build log contain detail information about the execution of a schedule.

Build filter:

Project	Schedule	When to trigger	Latest build
Jug Avis Web	Orario	manually	luntbuild-1.0.1 [2006-10-27 22:20]

Per la Demo Live ci vediamo al Javaday a Cagliari il 25 novembre 2006

Have a lot of Fun !

Ringraziamenti.

Spring

Spring Framework Team

Grazie del lavoro !

Grazie per l'attenzione

http://www.jugsardegna.org/vqwiki/jsp/Wiki?MassimilianoDessi

http://wiki.java.net/bin/view/People/MassimilianoDessi

http://www.jroller.com/page/desmax