Testing H-Store

Andy Pavlo

March 20, 2012

BROWN

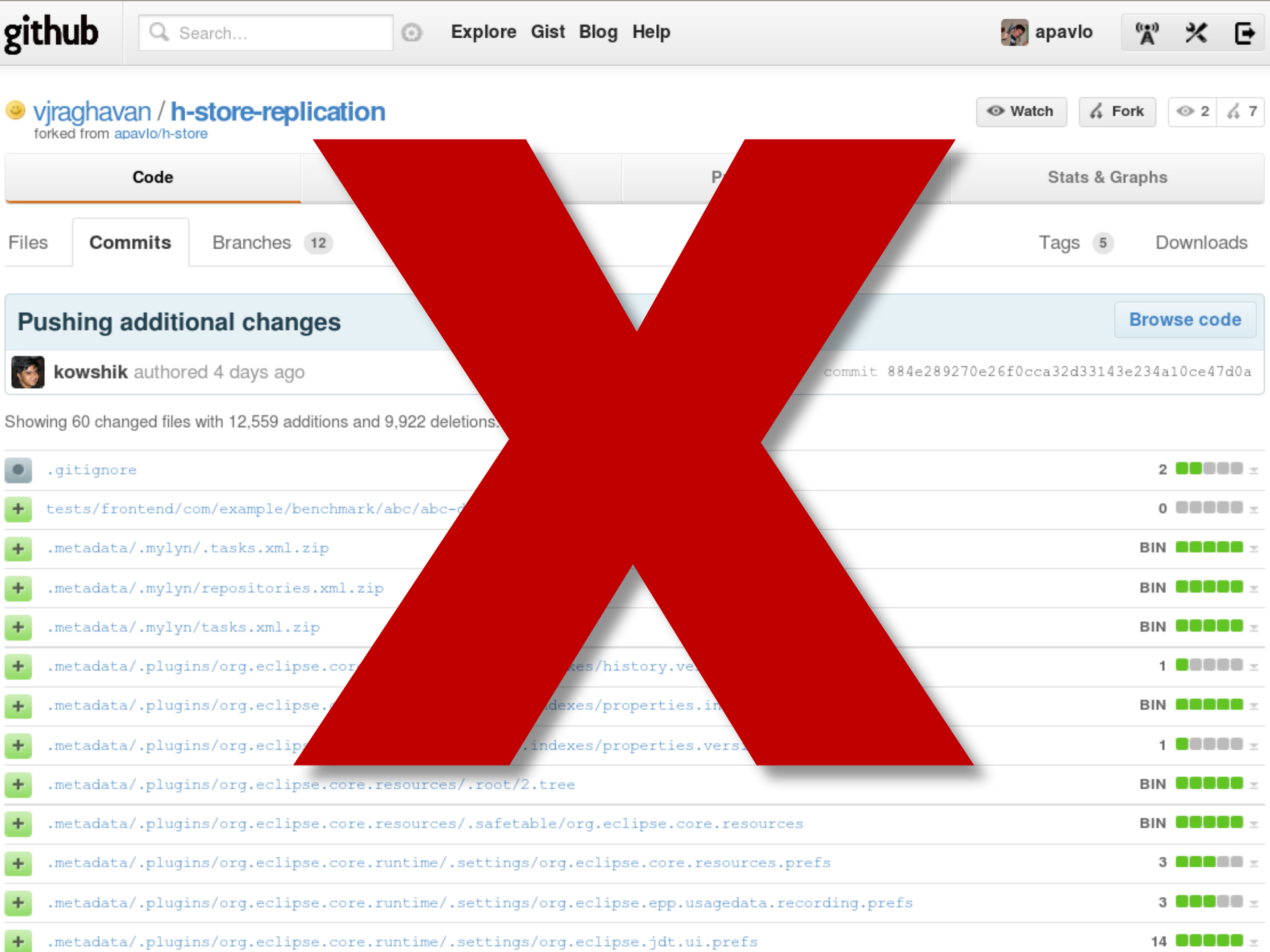
Project Rule #1

- **You must use the "official" Eclipse source code style/format.**
  - ***Easy to download and install***
  - ***4 Space Indentation / No Tabs***
  - ***Automatic checking coming soon…***

http://bit.ly/xaF2n7

Project Rule #2

- **Do not commit personal files:**
  - ***Eclipse settings/metadata.***
  - ***Temporary files, logs, tarballs.***
  - ***Pictures of your cat.***


github
Search...
Explore Gist Blog Help
apavlo
vjraghavan / h-store-replication
forked from apavlo/h-store
Watch
Fork
2
7
Code
Stats & Graphs
Files
Commits
Branches 12
Tags 5
Downloads
Pushing additional changes
Browse code
kowshik authored 4 days ago
commit 884e289270e26f0cca32d33143e234a10ce47d0a
Showing 60 changed files with 12,559 additions and 9,922 deletions.
.gitignore 2
tests/frontend/com/example/benchmark/abc/abc-d 0
.metadata/.mylyn/.tasks.xml.zip BIN
.metadata/.mylyn/repositories.xml.zip BIN
.metadata/.mylyn/tasks.xml.zip BIN
.metadata/.plugins/org.eclipse.cor xes/history.ve 1
.metadata/.plugins/org.eclipse. dexes/properties.in BIN
.metadata/.plugins/org.eclip .indexes/properties.vers 1
.metadata/.plugins/org.eclipse.core.resources/.root/2.tree BIN
.metadata/.plugins/org.eclipse.core.resources/.safetable/org.eclipse.core.resources BIN
.metadata/.plugins/org.eclipse.core.runtime/.settings/org.eclipse.core.resources.prefs 3
.metadata/.plugins/org.eclipse.core.runtime/.settings/org.eclipse.epp.usagedata.recording.prefs 3
.metadata/.plugins/org.eclipse.core.runtime/.settings/org.eclipse.jdt.ui.prefs 14

Project Rule #3

- **Avoid using locks in your code:**
  - ***No synchronized, No Semaphores, No ReentrantLocks.***
  - ***Never use Vector or Hashtable.***
  - ***CAS counters are ok (e.g., AtomicInteger)***
  - ***If you think you need a lock, write a clear explanation in the code as to why or ask Andy.***

Additional Suggestions

- **Use primitive objects sparingly.**
  - *Use int/long instead of Integer/Long*
  - *Avoid auto-boxing (Integer.valueOf())*
- **Don't allocate short-lived collections**
  - *For partition/site objects, use array instead of Map*
  - *Consider using an object pool (HStoreObjectPools)*

H-Store Test Harness

- **All test cases for database and projects written using JUnit.**
- **Quick vs. Full**
  - ***Full requires 2GB of space for testing files.***

```
$ ant junit
$ ant junit-full
```

http://bit.ly/FR78mj

Build & Test Server

- **Continuous integration service automatically tests your project when you push changes to Github.**
  - ***Must be using latest version (2012-03-16)***
  - ***Email me your project URL after you merge.***

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

GitHub

Git Polling Log

Build History (trend)

#93 Mar 18, 2012 9:20:41 PM

#92 Mar 18, 2012 5:04:38 PM

#91 Mar 18, 2012 3:40:36 PM

#90 Mar 18, 2012 3:06:40 PM

#89 Mar 18, 2012 10:58:39 AM

#88 Mar 17, 2012 12:43:25 PM

#87 Mar 16, 2012 6:28:38 PM

#86 Mar 16, 2012 5:42:42 PM

#85 Mar 16, 2012 4:28:38 PM

Project H-Store_Strangelove_Branch

H-Store OLTP Database Management System
http://hstore.cs.brown.edu/

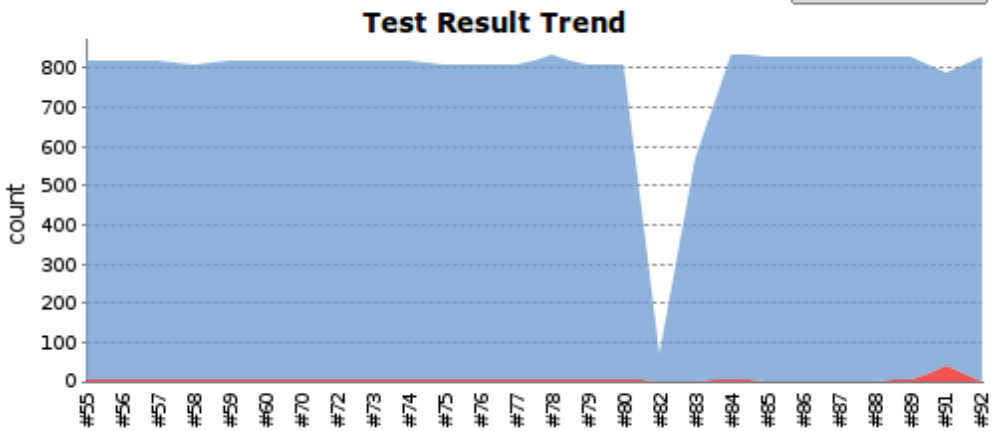
edit description

Disable Project

Workspace

Recent Changes

Latest Test Result (no failures)



(just show failures) enlarge

Permalinks

- Last build (#93), 20 min ago
- Last stable build (#92), 4 hr 36 min ago
- Last successful build (#92), 4 hr 36 min ago
- Last failed build (#91), 6 hr 0 min ago
- Last unsuccessful build (#91), 6 hr 0 min ago

Back to Project
Status
Changes
Console Output [raw]
Edit Build Information
History
Polling Log
Git Build Data
No Tags
Test Result
Previous Build
Next Build

Test Result

39 failures (+38)

788 tests (-38)
Took 14 min.
add description

All Failed Tests

Test Name	Duration	Age
>>> edu.brown.hstore.TestBatchPlannerComplex.testGetPlanGraph	2.7 sec	1
>>> edu.brown.hstore.TestBatchPlannerComplex.testFragmentIds	17 ms	1
>>> edu.brown.hstore.TestBatchPlannerComplex.testBuildWorkFragments	16 ms	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testBatchHashCode	6.3 sec	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testPlanMultiPartition	16 ms	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testMispredict	14 ms	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testMispredictPartitions	55 ms	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testGetStatementPartitions	15 ms	1
>>> edu.brown.hstore.TestBatchPlannerUtil.testReplicatedTableTouchedPartitions	13 ms	1
>>> edu.brown.hstore.TestHStoreCoordinator.testStartConnection	2.6 sec	1
>>> edu.brown.hstore.TestHStoreCoordinator.testStopConnection	67 ms	1
>>> edu.brown.hstore.TestHStoreSite.testSendClientResponse	1.9 sec	1
>>> edu.brown.hstore.TestHStoreSite.testSinglePartitionPassThrough	66 ms	1
>>> edu.brown.hstore.TestPartitionExecutor.testGetProcedure	2.1 sec	1

Writing a Simple Unit Test

- **Create a new class with 'Test' prefix in tests directory:**
  - ***Real Code:***
    src/frontend/edu/brown/hstore/wal/Logger.java
  - ***Unit Test Code:***
    tests/frontend/edu/brown/hstore/wal/TestLogger.java

http://bit.ly/FRkoHw

Writing a Simple Unit Test

- Extend edu.brown.BaseTestCase
- Any method name that starts with '**test**' is automatically executed.
- Can execute in directly Eclipse.

http://bit.ly/FRkoHw

```java
package edu.brown.hstore.wal;

import org.junit.Test;
import org.voltdb.catalog.Procedure;

import edu.brown.BaseTestCase;
import edu.brown.benchmark.tm1.procedures.DeleteCallForwarding;
import edu.brown.utils.ProjectType;

public class TestLogger extends BaseTestCase {

    @Override
    protected void setUp() throws Exception {
        // Passing in a ProjectType will automatically initialize
        // the benchmark catalog
        super.setUp(ProjectType.TM1);
    }

    @Override
    protected void tearDown() throws Exception {
        // TODO: Clean up any files or connections that we made
    }

    @Test
    public void testSimpleTest() {
        // Use BaseTestCase's methods for retrieving catalog objects
        Procedure catalog_proc = this.getProcedure(DeleteCallForwarding.class);
        assertNotNull(catalog_proc.getName());

        // Write asserts to check various conditions
        int x = 1;
        int y = 1;
        assertEquals(2, x + y);
    }
}
```

Writing a Regression Test

- **Regression test suite allow you to test the full system.**
- **Automatically setup and deploy cluster configurations directly in tests.**
  - ***Compiles catalog and provides client handle.***
  - ***Only on localhost***

http://bit.ly/FRkoHw

```java
public class TestLoggerSuite extends RegressionSuite {

    /**
     * JUnit / RegressionSuite Boilerplate Constructor
     * @param name The name of these test suite
     */
    public TestLoggerSuite(String name) {
        super(name);
    }

    static public junit.framework.Test suite() {
        MultiConfigSuiteBuilder builder =
                new MultiConfigSuiteBuilder(TestLoggerSuite.class);
        VoltProjectBuilder project = new VoltProjectBuilder("logger");
        VoltServerConfig config = null;

        // Schema + Table Partitions
        project.addSchema(TestLoggerSuite.class.getResource("logger-ddl.sql"));
        project.addTablePartitionInfo("P1", "ID");
        project.addTablePartitionInfo("P2", "ID");

        // Single Statement Procedures
        project.addStmtProcedure("TestInsert", "INSERT INTO P1 VALUES (?, ?)");
        project.addStmtProcedure("TestSelect", "SELECT * FROM P1 WHERE ID = ?");

        // CLUSTER CONFIG #1
        // One site with two partitions running in this JVM
        config = new LocalSingleProcessServer("logger-twoPart.jar", 2,
                                              BackendTarget.NATIVE_EE_JNI);
        config.compile(project);
        builder.addServerConfig(config);

        // CLUSTER CONFIG #2
        // Two sites, each with two partitions running in separate JVMs
        config = new LocalCluster("logger-twoSiteTwoPart.jar", 2, 2, 1,
                                  BackendTarget.NATIVE_EE_JNI);
        config.compile(project);
        builder.addServerConfig(config);

        return builder;
    }
```

```java
public class TestLoggerSuite extends RegressionSuite {

    /**
     * JUnit / RegressionSuite Boilerplate Constructor
     * @param name The name of these test suite
     */
    public TestLoggerSuite(String name) {
        super(name);
    }

    static public junit.framework.Test suite() {
        MultiConfigSuiteBuilder builder =
                new MultiConfigSuiteBuilder(TestLoggerSuite.class);
        VoltProjectBuilder project = new VoltProjectBuilder("logger");
        VoltServerConfig config = null;

        // Schema -
        project.add
        project.add
        project.add

        // Single S
        project.add
        project.add

        // CLUSTER
        // One site
        config = ne

        config.comp
        builder.add

        // CLUSTER
        // Two sites, each with two partitions running in separate JVMs
        config = new LocalCluster("logger-twoSiteTwoPart.jar", 2, 2, 1,
                                  BackendTarget.NATIVE_EE_JNI);
        config.compile(project);
        builder.addServerConfig(config);

        return builder;
    }
```

```java
@Test
public void testInsert() throws IOException, ProcCallException {
    int id = 1234;
    String val = "ABCD";

    Client client = getClient();
    ClientResponse cr = client.callProcedure("TestInsert", id, val);
    assertEquals(Status.OK, cr.getStatus());

    cr = client.callProcedure("TestSelect", id);
    assertEquals(Status.OK, cr.getStatus());
    VoltTable result = cr.getResults()[0];
    assertTrue(result.advanceRow());
    assertEquals(id, (int)result.getLong(0));
    assertEquals(val, result.getString(1));
}
```

Writing a Regression Test

- If you get an **UnsatisfiedLinkError**, make sure you have the "VM Arguments" configured properly:
  - *Run Configurations → JUnit → Arguments*

```
-ea -Djava.library.path=obj/release/nativelibs
```

http://bit.ly/FRkoHw