BITS ZG629T: Dissertation

GIFT – Groovy In-Memory Framework for Testing

Ajay Kumar S
2013HZ12969

Agenda

- Need for GIFT Approach
- Why TDD?
- Why Groovy?
- Traditional Vs. GIFT Approach
- High Level Design
- Sample GIFT Test case

Need for GIFT Approach

- Unit test is not enough for covering Database connectivity code
- Mocking doesn't solve testing DAO classes in Java
- Needed a in-memory Database framework for testing DAOs which should be running as part of build(similar to Unit tests).

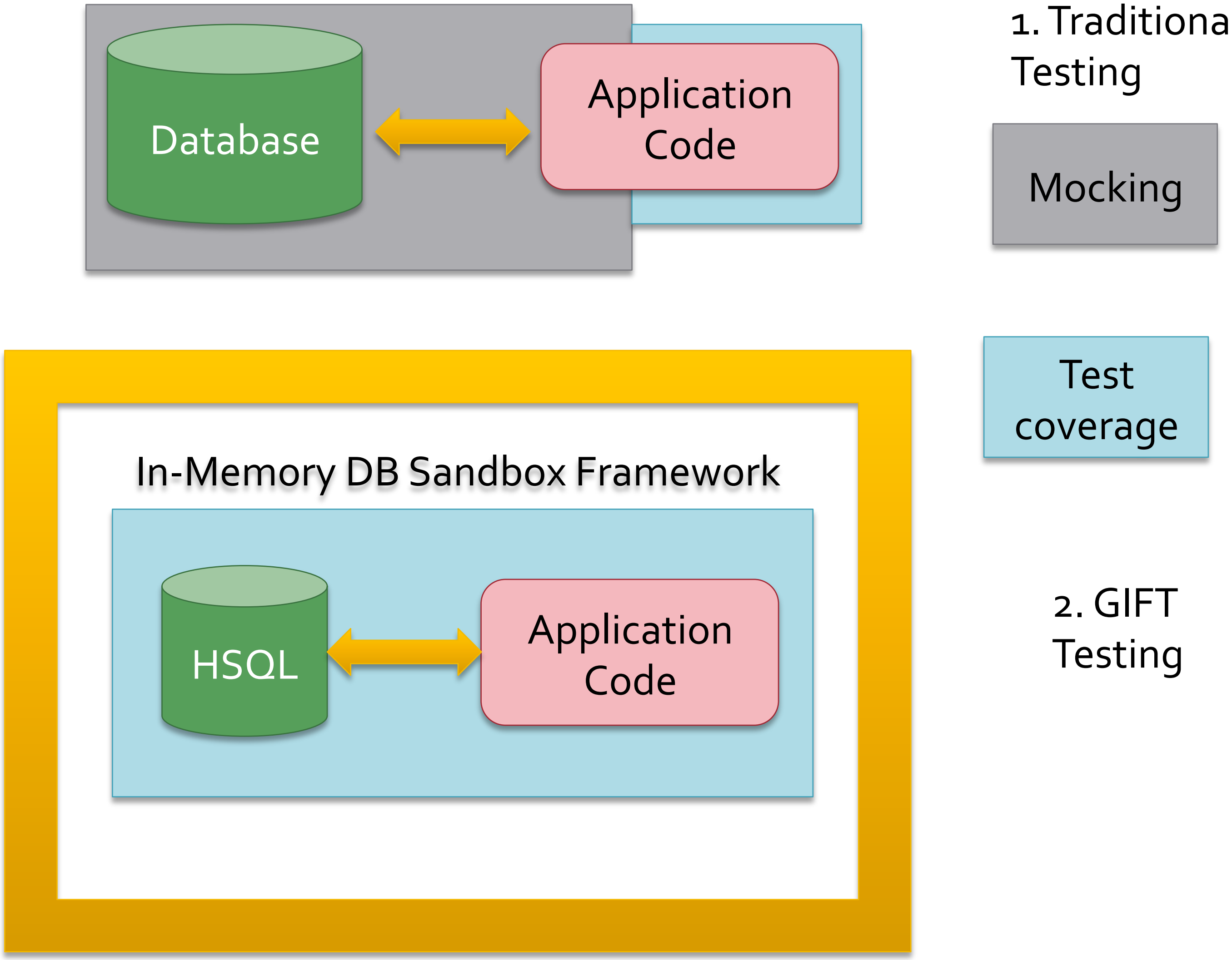
Why TDD?

- TDD, is an evolutionary approach to development which combines test-first development where we write a test before you write just enough production code to fulfill that test and refactoring.
- Makes hassle free code refactoring when development cycle is shorter and codebase is updated daily.

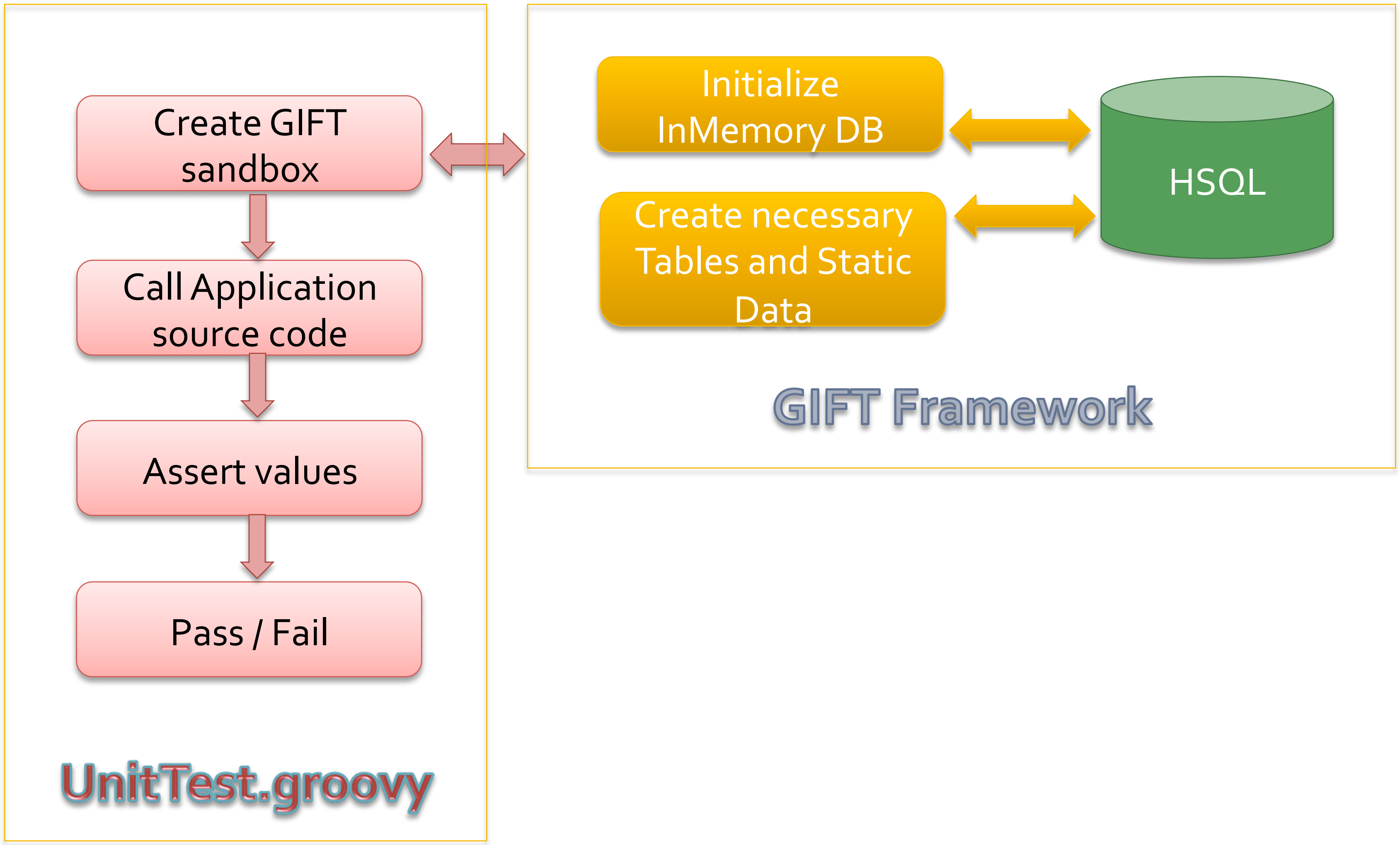
Why Groovy?

- Code faster than Java
- Same Java Syntax can be used
- Readable code using DSL
- Closure support (even with Java 6)
- Meta Object Class support

Traditional Vs. GIFT Approach



High Level Design



Sample GIFT Test case

```
public class EmployeeTest {
   def database
   enum Sandbox {
      Employee
   }
   @Before
   void setUp() {
      database = create(Sandbox)
   }
   @After
   void tearDown() {
      database.shutDown()
   }
   @Test
   void 'test Employee DAO Table'() {

      //Get Employee details from Employee DB.
      EmployeeDAO employeeDAO = new EmployeeDAO()
      Employee employee = employeeDAO.get("Ajay");
      assert employee.id == '2013HZ12969'
      assert employee.name == 'Ajay'
      assert employee.location == 'Bangalore'
   }
}
```

Queries

Thank you!