**Inline Singleton**

**Motivation**

Using a Singleton is not always necessary when some code needs access to an object. Singletons should only be used when a class has only one instance and can provide a global point of access to it. It is usually simpler to pass object resources as a reference to objects that need it. This ensures that classes are easier to read as similar functionality is grouped together, rather than having to reference the singleton every time.

**Methods**

1. Declare the Singleton’s methods in the class that uses them.
2. Change all code references to the singleton to references within the class.
3. Use Move Method and Move Field to move functionalities from the Singleton to the class.
4. Delete the Singleton.

**Sample Code to refactor**

A casino uses a virtual blackjack game. The Blackjack class references a singleton that returns the hit or stay response of a player. This functionality can be implemented within the Blackjack class itself

public class Console {

static private HitStayResponse hitStayResponse =

new HitStayResponse();

private Console() {

super();

}

public static HitStayResponse obtainHitStayResponse(BufferedReader input) {

hitStayResponse.readFrom(input);

return hitStayResponse;

}

public static void setPlayerResponse(HitStayResponse newHitStayResponse) {

hitStayResponse = newHitStayResponse;

}

}

public class ScenarioTest extends TestCase...

public void testDealerStandsWhenPlayerBusts() {

**Console.setPlayerResponse(new TestAlwaysHitResponse());**

int[] deck = { 10, 9, 7, 2, 6 };

Blackjack blackjack = new Blackjack(deck);

blackjack.play();

assertTrue("dealer wins", blackjack.didDealerWin());

assertTrue("player loses", !blackjack.didPlayerWin());

assertEquals("dealer total", 11, blackjack.getDealerTotal());

assertEquals("player total", 23, blackjack.getPlayerTotal());

}

public class Blackjack...

public void play() {

deal();

writeln(player.getHandAsString());

writeln(dealer.getHandAsStringWithFirstCardDown());

HitStayResponse hitStayResponse;

do {

write("H)it or S)tay: ");

**hitStayResponse = Console.obtainHitStayResponse(input);**

write(hitStayResponse.toString());

if (hitStayResponse.shouldHit()) {

dealCardTo(player);

writeln(player.getHandAsString());

}

}

while (canPlayerHit(hitStayResponse));

// ...

}

The code in bold indicates references to the singleton class.