

Today's Agenda

- **Black Box Testing – Monopoly Test Cases**
- **Automated Test**
- **FIT**

Test Automation Manifesto* Part One

- **Concise** – Test should be as simple as possible but no simpler
- **Self-checking** – Test should report its results such that no human interpretation is necessary
- **Repeatable** – Test can be run repeatedly without human intervention
- **Robust** – Test produce the same result now and forever
- **Sufficient** – Test verify all the requirements of the software being used
- **Necessary** – Everything in each test contributes to the specification of desired behavior

*Meszaros, G., Smith, S. M., et al. (2003) The Test Automation Manifesto, Extreme Programming and Agile Methods – XP/Agile Universe 2003, Lecture Notes in Computer Science 2753, Springer.

Test Automation Manifesto* Part Two

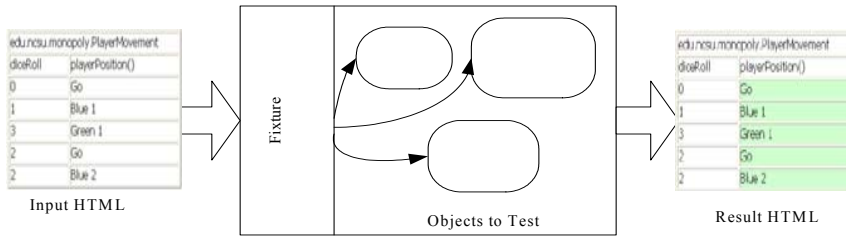
- **Clear** – Every statement is easy to understand.
- **Efficient** – Tests run in a reasonable amount of time.
- **Specific** – Each test failure points to a specific piece of broken functionality (e.g. each test case tests one possible point of failure)
- **Independent** – Each test can be run by itself or in a suite with an arbitrary set of other tests in any order.
- **Maintainable** – Tests should be easy to modify and extend.
- **Traceable** – Tests should be traceable to requirements; requirements should be traceable to tests
- **Incremental** – Tests should be developed during development; tests can be (near) impossible to automate when code is complete, {added by L.W.}

Automated Tests

➤ **Pros:**

➤ **Cons:**

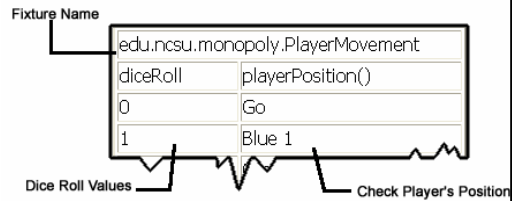
Framework for Integrated Test (FIT)



See resources posted on Open Seminar

- Book chapter
- Tutorial
- Monopoly example

Column Fixture



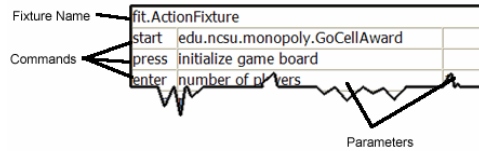
```
public class PlayerMovement extends ColumnFixture {
    public int diceRoll;
    private GameMaster gameMaster;

    public PlayerMovement() {
        gameMaster = GameMaster.instance();
        gameMaster.reset();
        gameMaster.setGUI(new MockGUI());
        gameMaster.setGameBoard(new SimpleGameBoard());
        gameMaster.setNumberOfPlayers(1);
    }

    public void execute() throws Exception {
        gameMaster.movePlayer(0, diceRoll);
    }

    public String playerPosition() {
        return gameMaster.getCurrentPlayer().getPosition().getName();
    }
}
```

Action Fixture



start: Starts a fixture. This command has one parameter: the name of the fixture class.

enter: This command simulates a user entering a value in a text field or text box. It has two parameters. The first parameter is name of the field, and the second is the value to be put in this field. *[invoke method with one parameter]*

press: This command simulates a user clicking a button. It has one parameter: the name of the button. *[invoke method with no parameters]*

check: This command is used to check whether the state of the system is as expected. It has two parameters. The first is the name of the state, and the second is the expected value of this state. *[invoke method with no parameters that returns a value.]*

Action Fixture (cont'd)



fit.ActionFixture		
start	edu.ncsu.monopoly.GoCellAward	
press	initialize game board	
enter	number of players	1
enter	player dice roll	7
check	player money	1700
enter	player dice roll	5
check	player money	1900
enter	player dice roll	2
check	player money	1900

Action Fixture (cont'd again)

```
public class GoCellAward extends ActionFixture {
    private GameMaster gameMaster;
    public void initializeGameBoard() {
        gameMaster = GameMaster.instance();
        gameMaster.reset();
        gameMaster.setGUI(new MockGUI());
        gameMaster.setGameBoard(new SimpleGameBoard());
    }
    public void numberOfPlayers(int number) {
        gameMaster.setNumberOfPlayers(number);
    }
    public void playerDiceRoll(int diceRoll) {
        gameMaster.movePlayer(0, diceRoll);
    }
    public int playerMoney() {
        return gameMaster.getCurrentPlayer().getMoney();
    }
}
```

fit.ActionFixture		
start	edu.ncsu.monopoly.GoCellAward	
press	initialize game board	
enter	number of players	1
enter	player dice roll	7
check	player money	1700
enter	player dice roll	5
check	player money	1900
enter	player dice roll	2
check	player money	1900

NC STATE UN

