**War of Robotcraft**

**Test Plan**

Team: A3

Team Members:

|  |  |  |
| --- | --- | --- |
| **Name** | **NSID** | **Student ID** |
| Fu, Chen | chf354 | 11183491 |
| He, Jiahuan | jih889 | 11183346 |
| Wang, Shisong | shw940 | 11157916 |
| Xie, Ruida | rux793 | 11194258 |
| Yang, Chen | chy202 | 11183550 |

Date: Nov 01, 2016

|  |  |  |
| --- | --- | --- |
| **Version Number** | **Description of Changes** | **Approved Date** |
| 0.1 | First Draft | 2016-11-01 |
| 0.2 |  |  |
| 0.3 |  |  |
| 0.4 |  |  |
| 0.5 |  |  |
| 0.6 |  |  |
| 0.7 |  |  |
| 1.0 |  |  |

Revision History:

# Introduction

# Initialize game interface

# Human player interface

## Unit test

### Class Robot

#### isDead(): bool

Summary: This unit test is to test Robot.isDead() whether it can return true if a robot is dead or false otherwise.

robot 🡨 new Robot()

declare expectedValue

declare actualValue

* test case 1: the robot is not dead

expectedValue = false

robot.healthPoint = 1

actualValue = robot.isDead()

assert if actualValue and excpectdValue are equal

* test case 2: the robot is dead

expectedValue = true

robot.healthPoint = 0

actualValue = robot.isDead()

assert if actualValue and excpectdValue are equal

* test case 3: the value is out of range

expectedValue = true

robot.healthPoint = -1

actualValue = robot.isDead()

assert if actualValue and excpectdValue are equal

#### turn()

#### move()

Summary: This unit test is to test Robot.move(coorfinate coor) whether it can change the robot position correctly.

robot 🡨 new Robot()

robot.rangPoint 🡨 1

robot.coor.x 🡨 2

robot.coor.y 🡨 3

robot.coor.z 🡨 1

coor 🡨 new coordinate()

coor.x 🡨 3

coor.y 🡨 4

coor.z 🡨 2

declare excpectedValueMovePoint

declare expectedValueCoor

declare actualValueMovePoint

declare actualValueCoor

* test case 1: the robot has full of movementPoint, then movemetPoint minus 1, the coor will changed at new position

robot.movementPoint 🡨 3

robot.move(coor)

excpectdValue1 🡨 2

actualValue1 🡨 robot.movementPoint

assert if actualValue1 and excpectdValue1 are equal

expectValue2 🡨 coor

actualValue2 🡨 robot.coor

assert if actualValue1 and excpectdValue1 are equal

* test case 2: the robot has no movementPoint, then robot cannot move, nothing to be changed

robot.movementPoint 🡨 0

robot.move(coor)

excpectdValue1 🡨 0

actualValue1 🡨 robot.movementPoint

actualValue =

excpectdValue = true

assert if actualValue and excpectdValue are equal

#### shoot(int distance): Coordinate coor

Summary: This unit test is to test Robot.shoot() which receives a distance to shoot and returns the target coordinate.

robot\_1 🡨 new Robot()

robot\_2 🡨 new Robot()

robot\_3 🡨 new Robot()

robot\_4 🡨 new Robot()

declare expectedValue

declare actualValue

* test case 1:

#### demaged()

## Functional test

# AI player interface

# Game interface

# Changes

# Summary

Unit test

Class XXXX

method()

test case 1: balabalabala

input: balabalabala

expected output: balabalabala

test case 2:

test case ….

Functional test

Start game: actions and event