microRTS平台使用说明

目录

[安装 2](#_Toc536637890)

[视频教程 2](#_Toc536637891)

[图文版 2](#_Toc536637892)

[新建Java项目 2](#_Toc536637893)

[导入microRTS包 3](#_Toc536637894)

[运行 4](#_Toc536637895)

[游戏平台使用 5](#_Toc536637896)

[左侧控制台 5](#_Toc536637897)

[右侧地图 6](#_Toc536637898)

[Trace界面 8](#_Toc536637899)

[Tournaments 9](#_Toc536637900)

[创建自己的AI 9](#_Toc536637901)

[视频教程 9](#_Toc536637902)

[图文教程 9](#_Toc536637903)

# 安装

## 视频教程

<https://www.youtube.com/watch?v=_jVOMNqw3Qs>

## 图文版

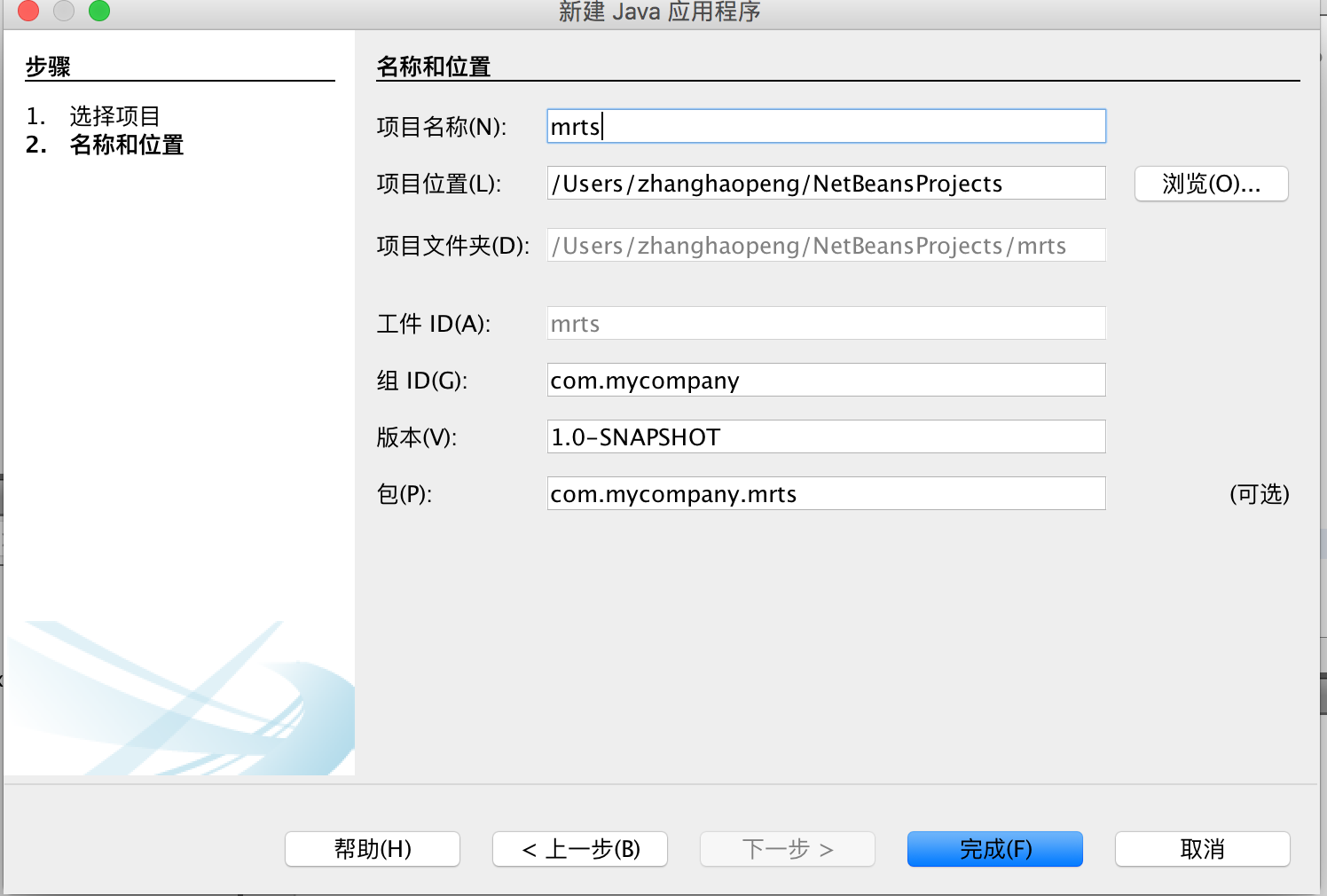
### 新建Java项目

首先通过github下载（<https://github.com/santiontanon/microrts>）

在命令行运行 ： git clone <https://github.com/santiontanon/microrts.git>

或下载zip包。接着通过IDE新建Java项目（以Netbean为例）



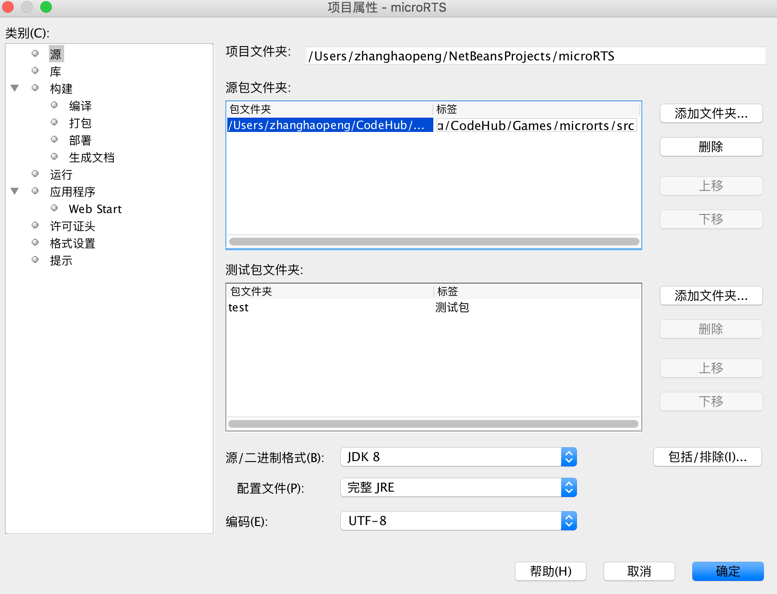


### 导入microRTS包

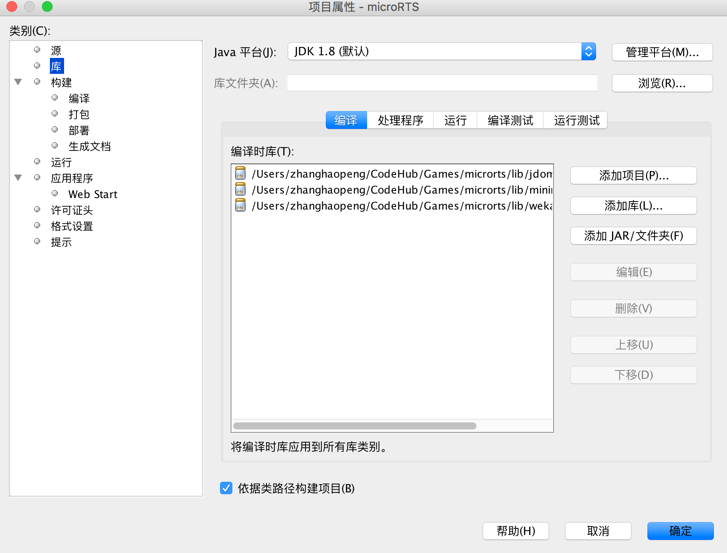
1. 右键点击项目，点击属性



1. 将原来的Src文件删除，倒入microRTS项目中的src文件。

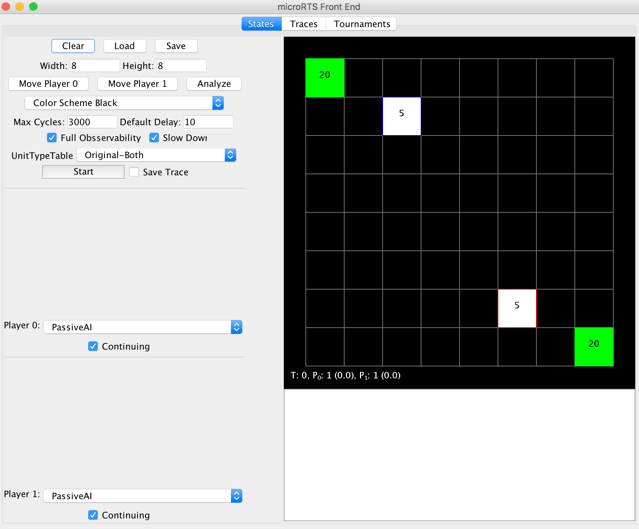


1. 将microts中lib文件下的三个jar包也都导入到项目中。



### 运行

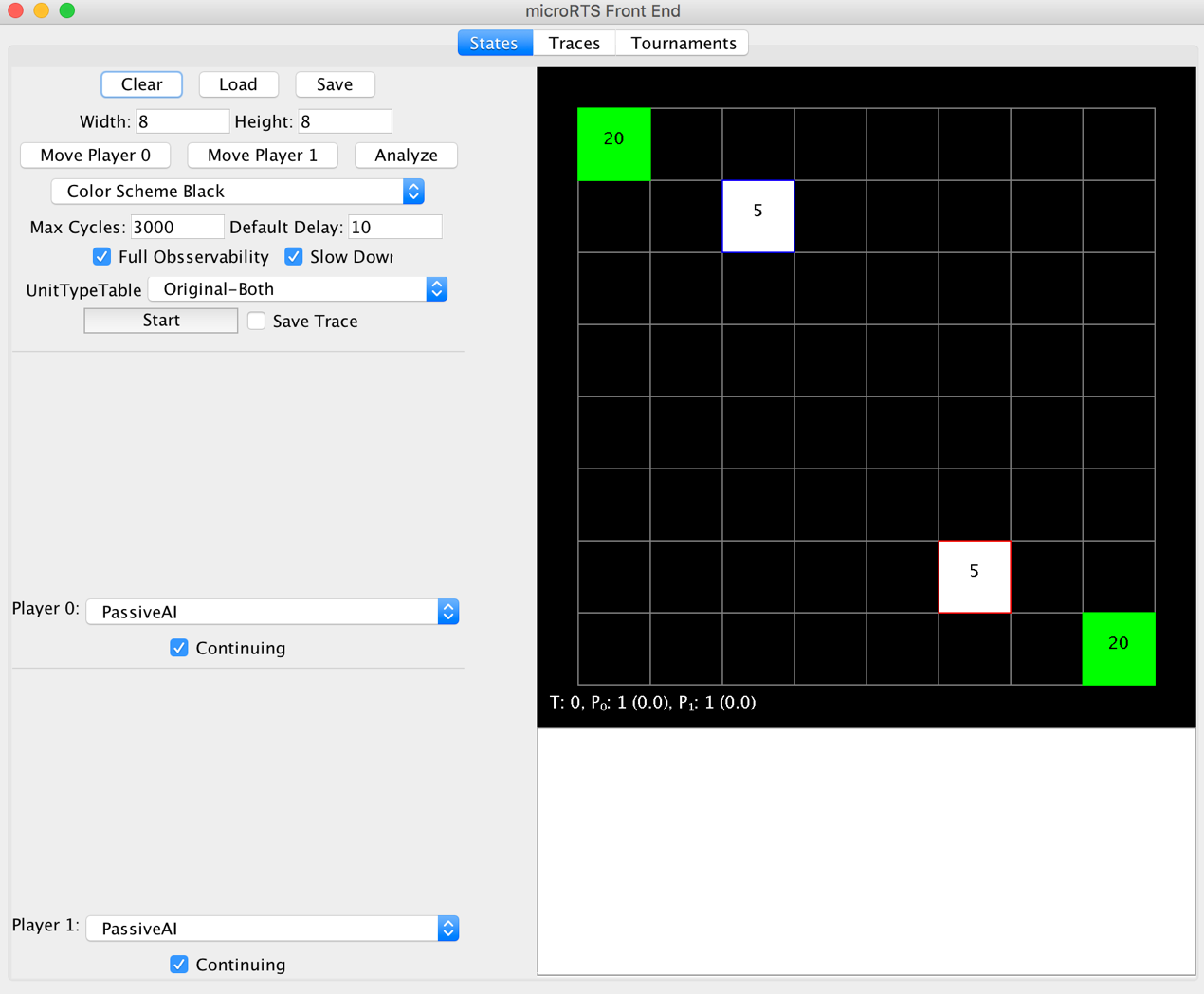
完成上述步骤，运行gui.fronend包下的FrontEnd.java。成功运行后进入游戏的GUI界面。



# 游戏平台使用

游戏平台界面如图，右侧是控制台，可以导入地图，选择AI等。左侧是游戏初始化设置展示。

控制台



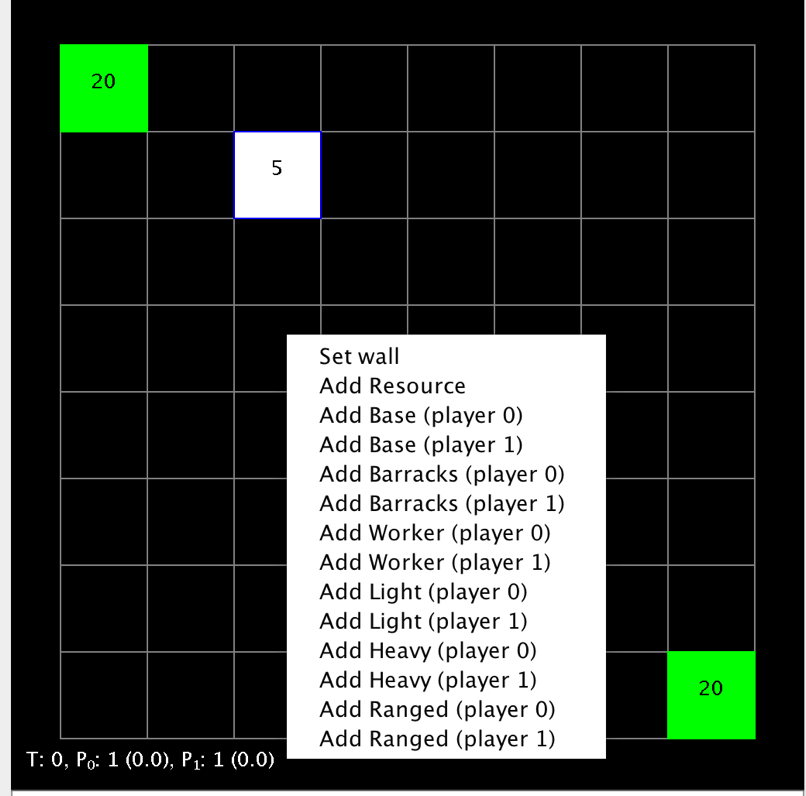
游戏初始设置

## 左侧控制台

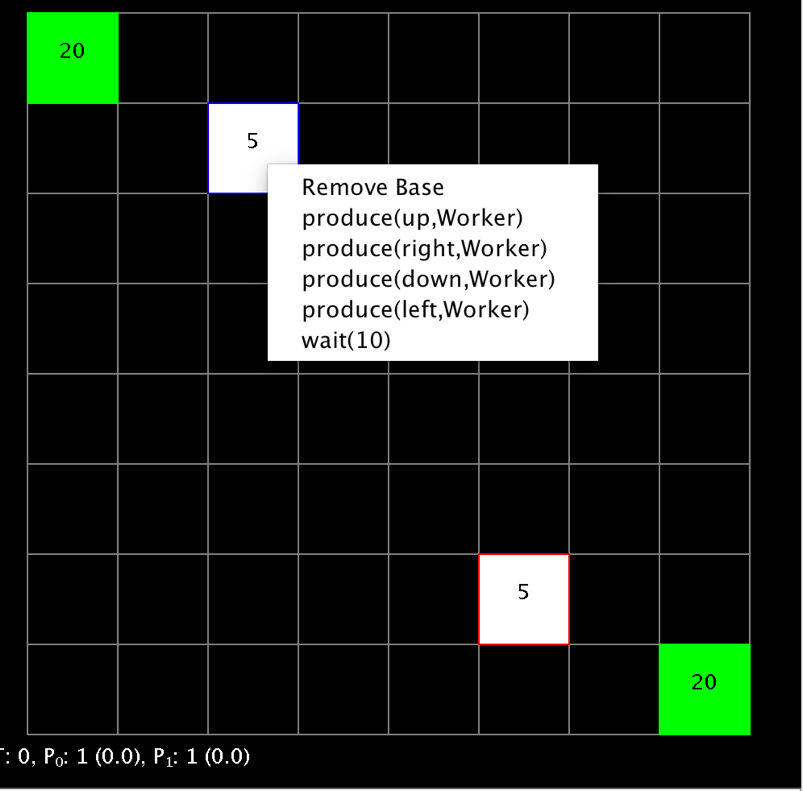
* **Clear:**将游戏的初始化设置清空
* **Load:**导入游戏设置（map下的xml文件）
* **Save：**将自定义的设置保存。
* **Max Cycles：**游戏最大的循环次数，即游戏时长
* **Defalut delay：**游戏延迟，涉及到展示时的游戏速度
* **Full Observalibity:** 即一方对于另一方是否可见
* **Slow down：**涉及在可视化时的展示速度。
* **Player 0 & Player 1:**游戏内置了不同的AI类型，可供选择。其中特别的是。PassiveAI是不会行动的AI。MouseController是人类操控AI。
* **Start：**开始游戏
* **Save Trace:** 开启该功能会将一局游戏的记录保存下来。

## 右侧地图

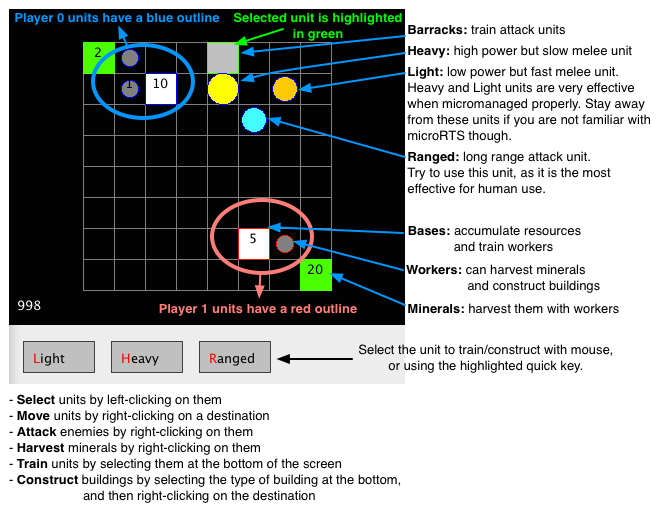
在地图上左键点击可以直接创建单位。



点击单位会有相应的操作：

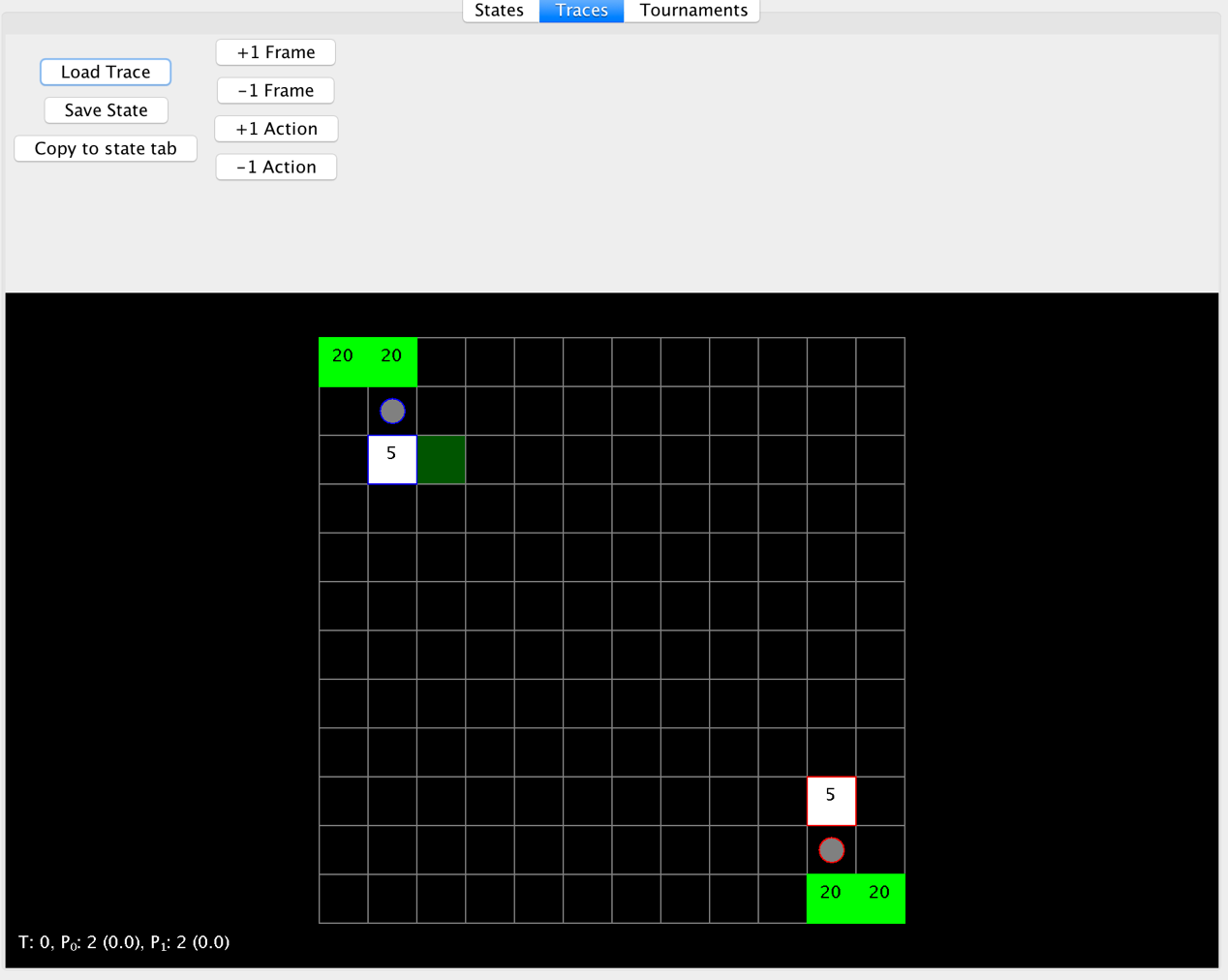


地图更详细的介绍如下：



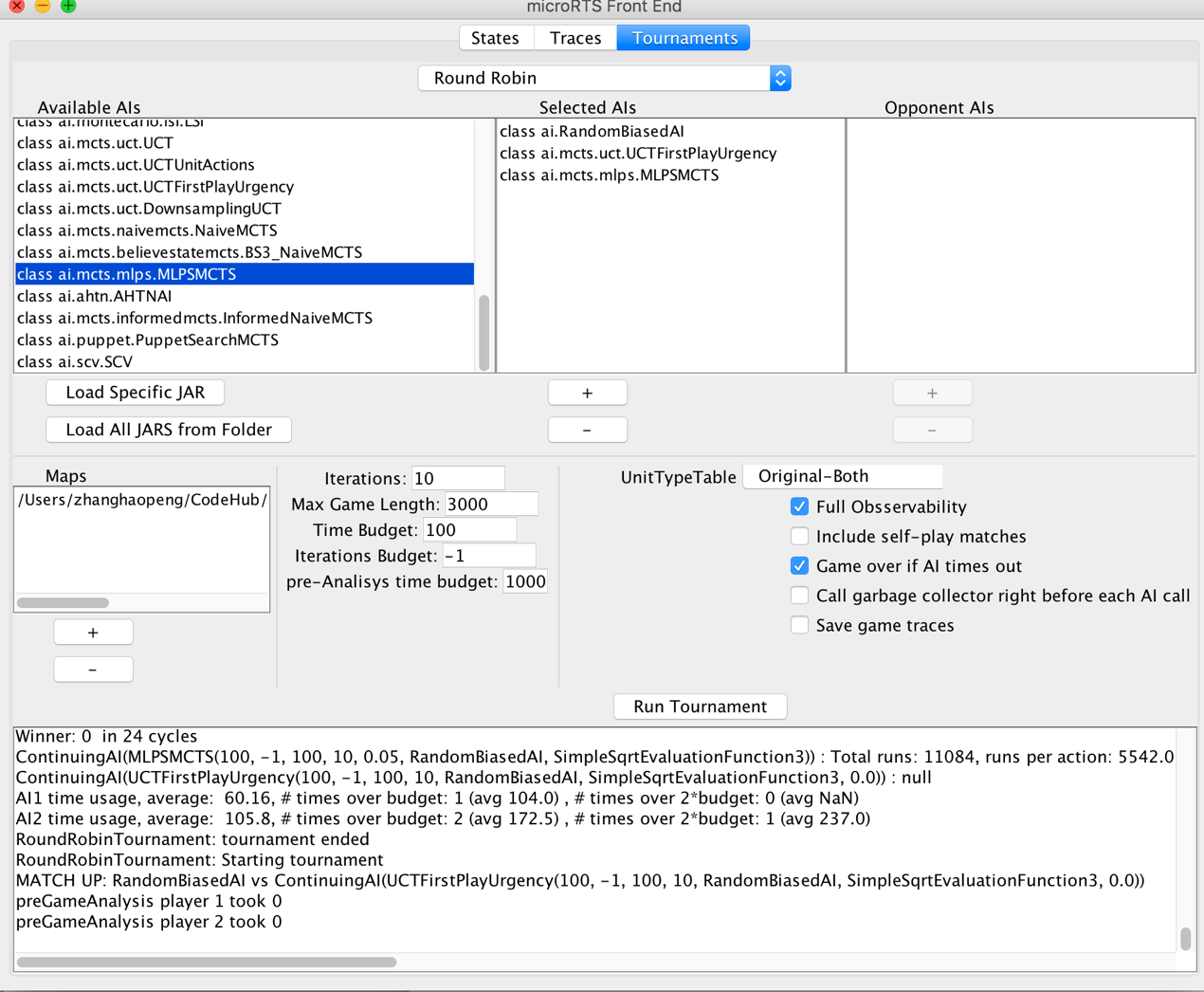
Heavy，light，Ranged是攻击单位，从Barracks中训练而得，训练需要消耗矿物。Workes是建造和收集矿物的单位，矿物从Bases中收集，建造Barrack也需要矿物。攻击单位可以移动并攻击对手单位，从而消灭对手。

## Trace界面



在GUI在界面中切换到Traces界面。通过load Trace读取Trace的xml文件。可以通过界面一帧一帧或者一个行动一个行动的观看局势。

## Tournaments



用户可在该GUI下导入AI，选择地图以及参数，让它们自行对决。

# AI模版

## 视频教程

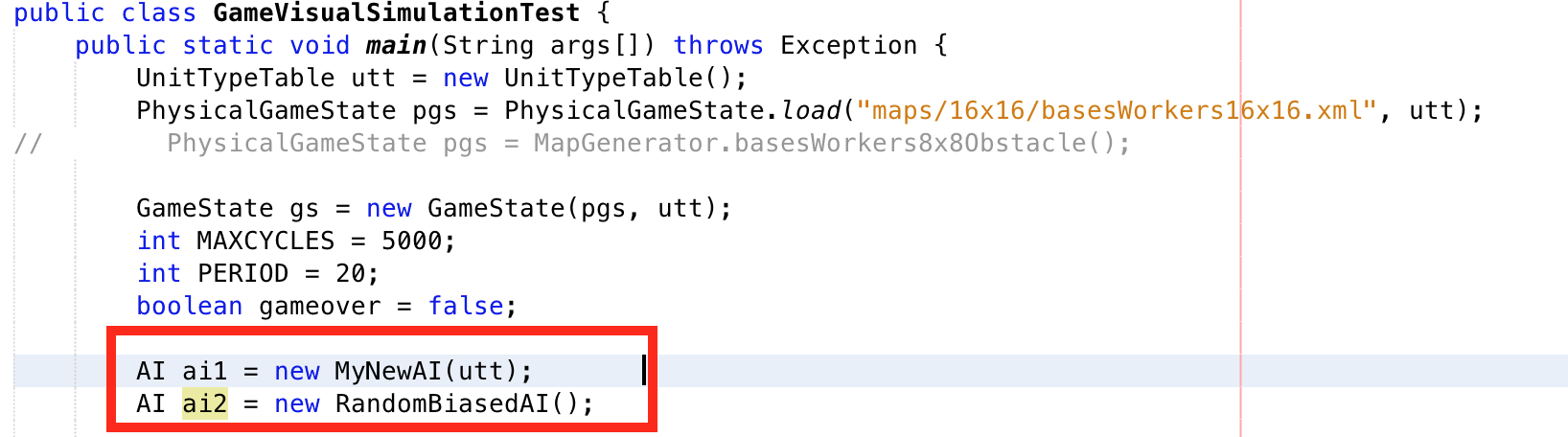
<https://www.youtube.com/watch?v=BikqC5gZWhU>

## 图文教程

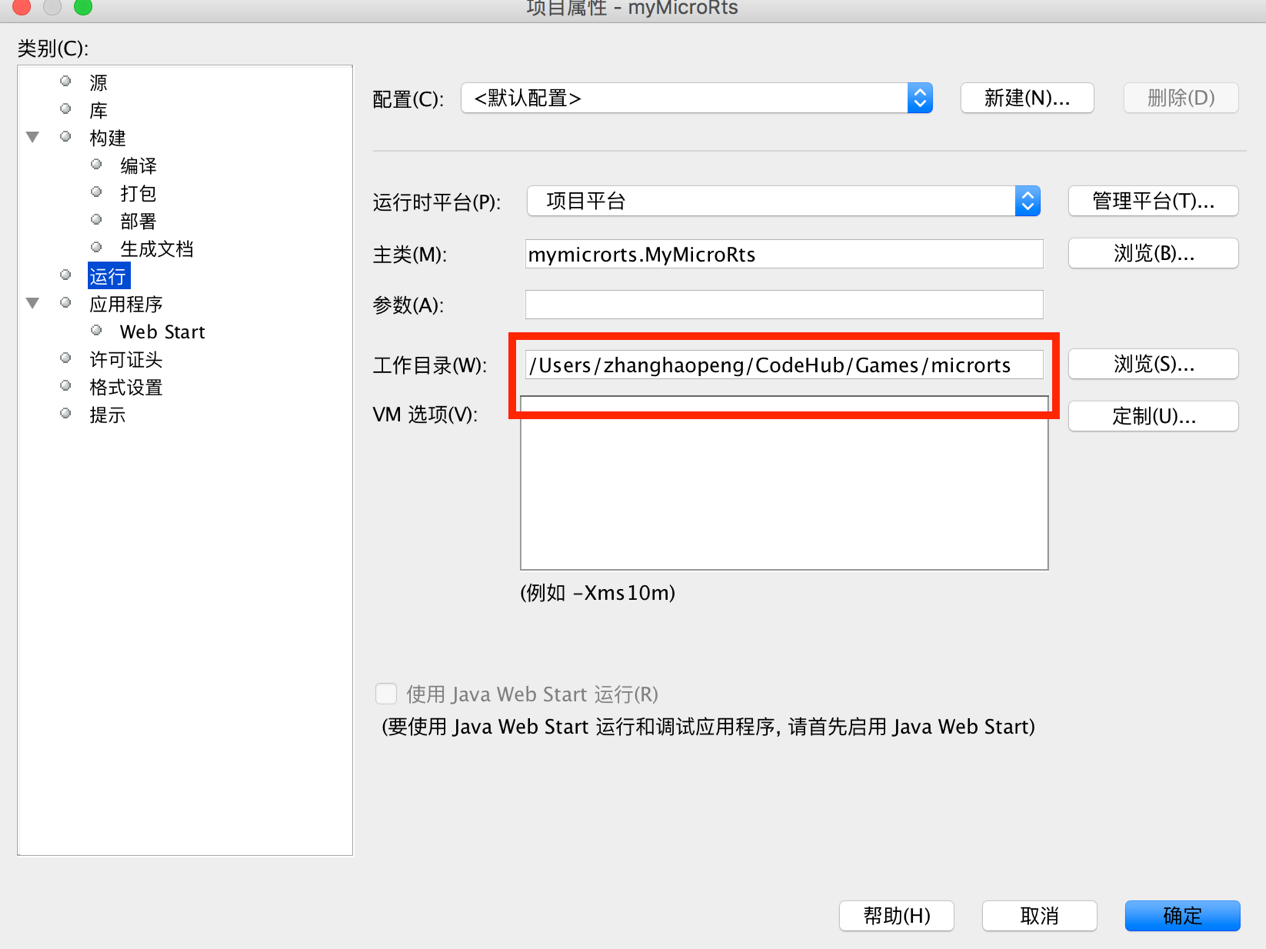
新创建一个Java项目。并创建一个用于构建AI的类。令该类继承 AIWithComputationBudget，即创建好了一个AI类。模版代码如下：

|  |
| --- |
| */\**  *\* To change this license header, choose License Headers in Project Properties.*  *\* To change this template file, choose Tools | Templates*  *\* and open the template in the editor.*  *\*/*  package **mymicrorts**;  import **ai.core.AI**;  import **ai.core.AIWithComputationBudget**;  import **ai.core.ParameterSpecification**;  import **java.util.ArrayList**;  import **java.util.List**;  import **rts.GameState**;  import **rts.PlayerAction**;  import **rts.units.UnitTypeTable**;  **public** **class** MyNewAI **extends** AIWithComputationBudget {  **UnitTypeTable** m\_utt = null;  *// This is the default constructor that microRTS will call:*  **public** MyNewAI(**UnitTypeTable** utt) {  super(-1,-1);  m\_utt = utt;  }  *// This will be called by microRTS when it wants to create new instances of this bot (e.g., to play multiple games).*  **public** **AI** clone() {  return new MyNewAI(m\_utt);  }    *// This will be called once at the beginning of each new game:*  **public** **void** reset() {  }    *// Called by microRTS at each game cycle.*  *// Returns the action the bot wants to execute.*  **public** **PlayerAction** getAction(**int** player, **GameState** gs) {  **PlayerAction** pa = new PlayerAction();  pa.fillWithNones(gs, player, 10);  return pa;  }    *// This will be called by the microRTS GUI to get the*  *// list of parameters that this bot wants exposed*  *// in the GUI.*  **public** **List**<**ParameterSpecification**> getParameters()  {  return new **ArrayList**<>();  }  } |

其中核心函数是 getAction 。返回的是AI每轮的行动。

之后建立一个test包。将原项目中的tests下GameVisualSimulationTest的类文件导入。将文件中的AI替换成自己的AI。运行。

运行时若报错，则可能是工作路径问题。



将项目属性运行的工作目录切换到原项目的工作目录应该即可成功运行。