Soltoon: Bokosh Bokosh

Technical documentation

V1.0.4

December, 2017

Authors:

Payam Mohammadi (payam.int@gmail.com)
Amirkasra Jalaldoust (amirkasraj@gmail.com)

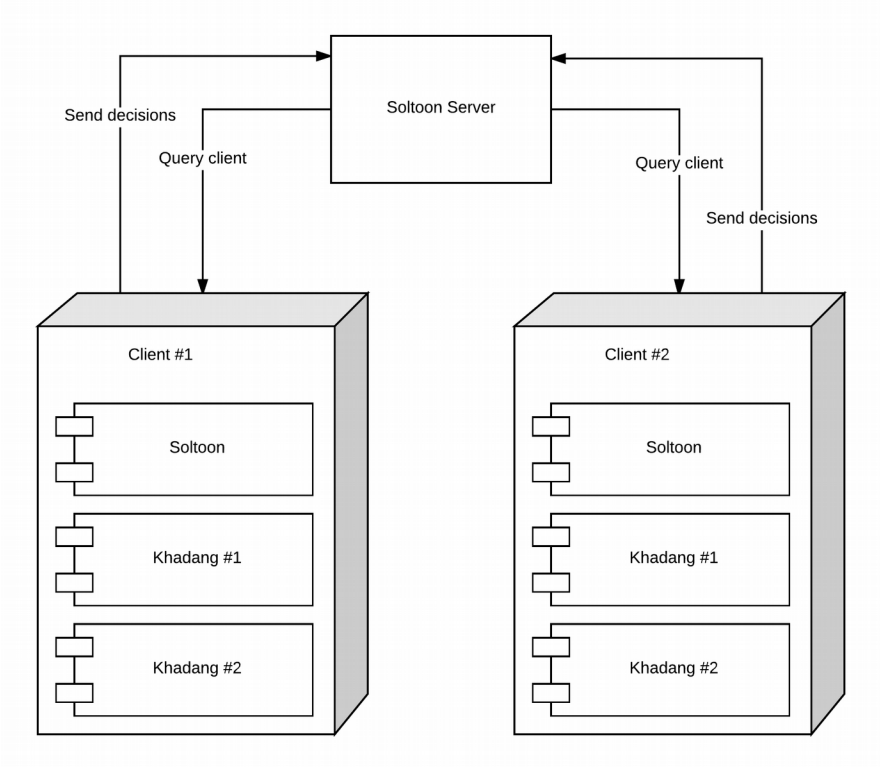
آموزش سریع بازی سلطون

بازی سلطون چگونه انجام می‌شود ؟

هر بازی سلطون یک یا چند شرکت کننده (کلاینت) دارد. هر کلاینت برنامه‌ای دارد که تصمیمات سلطون و خدنگ‌های سلطون را مشخص می‌کند. هنگامی که کلاینت‌ها آماده بازی هستند برنامه سلطون‌سرور به برنامه های شرکت‌کنندگان متصل می‌شود. هر بازی در چند دور انجام می‌شود.

در هر دور:

1. سلطون‌سرور از کلاینت‌ها می‌خواهد تصمیمات سلطون خود را اعلام کند.
2. سلطون‌سرور از کلاینت‌ها می‌خواهد تصمیمات خدنگ های خود را اعلام کند.



به خاطر داشته باشید همیشه ابتدا کلاینت‌ها را اجرا کنید سپس سلطون سرور را اجرا کنید.

وظیفه شما

تمام مسائل مربوط به ارتباط با سرور و مدیریت بازی از قبل انجام شده است. تنها وظیفه شما پیاده سازی کلاس دو Player و Fighter است. این دو کلاس تصمیم گیری‌های سلطون و خدنگ‌ها را مدیریت میکنند.

سلام جهان !

مرحله اول: دانلود آخرین نسخه بازی

قبل از شروع به کار باید آخرین نسخه بازی سلطون را دانلود کنید و به پروژه‌تان اضافه کنید. برای دانلود به لینک زیر بروید و آخرین نسخه سلطون را دانلود کنید:

https://soltoon.github.io/

در این آموزش از نسخه فایل *soltoon-game-1.0.4-jar-with-dependencies.jar* استفاده می‌شود.

مرحله دوم: اضافه کردن فایل jar به پروژه

اگر از *IDE* دیگری غیر از *Intellij Idea* استفاده میکنید می‌توانید به لینک های معرفی شده در لینک بالا مراجعه کنید در غیراینصورت:

1. در محل پروژه‌تان دایرکتوری lib را ایجاد کنید.
2. فایل soltoon-game-1.0.4-jar-with-dependencies.jar را در پوشه lib قرار دهید.
3. در نرم افزار Intellij Idea در پنجره Project، پوشه lib، روی فایل راست کلیک کرده گزینه Add as library را انتخاب کنید.

مرحله سوم: نوشتن اولین کد

حال می‌خواهیم اولین کدمان را بنویسیم. قصد داریم برنامه ای بنویسیم که در نقطه (1,1) بازی یک توپ جنگی ایجاد کند. این توپ جنگی وظیفه دارد به ن.

قدم اول: نوشتن کلاس مربوط به توپ جنگی‌مان

برای این‌کار باید کلاس جدیدی ایجاد کنیم که کلاس Fighter را extend می‌کند. همه‌ی خدنگ‌های بازی باید کلاس Fig hter را Extend کنند. در Constructor تابع باید نوع جنگنده را مشخص کنیم.

هر کلاس فرزند Fighter باید ۳ متد در این تابع را پیاده سازی کند. متد init هنگامی صدا زده می‌شود که خدنگ ایجاد می‌شود. متد lastThingsToDo وقتی صدا زده می‌شود که خدنگ از بین می‌رود. متد getAction در هر نوبت بازی صدا زده می‌شود و تصمیم خدنگ را دریافت می‌کند. این سه متد بعنوان ورودی پارامتر gameBoard را دریافت می‌کنند که حاوی اطلاعات زمین بازی در آن لحظه است.

در مورد این کلاس ها در مرجع بطور کامل توضیح داده شده است. فعلا به همین اندازه اکتفا کنید.

با توجه به توضیحات داده شده باید متد getAction را پیاده سازی کنیم. این متد یک آبجکت از نوع Action را برمی‌گرداند. کلاس Shoot فرزند کلاس Action است بنابراین یک آبجکت از کلاس Shoot که مشخص میکند به چه نقطه ای شلیک کنیم میسازیم و برمی‌گردانیم:

WiseAttacker.java

```java
import ir.pint.soltoon.soltoongame.shared.data.Fighter;
import ir.pint.soltoon.soltoongame.shared.data.action.Action;
import ir.pint.soltoon.soltoongame.shared.data.action.Shoot;
import ir.pint.soltoon.soltoongame.shared.data.map.FighterType;
import ir.pint.soltoon.soltoongame.shared.data.map.GameBoard;

public class WiseAttacker extends Fighter {
    public WiseAttacker() {
        super(FighterType.CANNON);
    }

    @Override
    public void init(GameBoard gameBoard) {

    }

    @Override
    public void lastThingsToDo(GameBoard gameBoard) {

    }

    @Override
    public Action getAction(GameBoard gameBoard) {
        return new Shoot(0, 0);
    }
}
```

قدم دوم: نوشتن کلاس مربوط به سلطون‌مان

قرار است سلطون در نقطه (1,1) یک توپ جنگی بسازد که این توپ جنگی هر نوبت به نقطه (0,0) حمله می‌کند. در قدم قبلی کد مربوط به توپ جنگی را نوشتیم. حال باید سلطون این توپ را در نقطه (1,1) ایجاد کند.

برای ساختن سلطون باید کلاس Player را extend کنیم. کلاس Player متدهای مشابه‌ای دارد که باید پیاده سازی شوند. متد getAction تصمیمات سلطون در هر دور بازی را مشخص می‌کند.

برای ساختن یک توپ جنگی در نقطه مربوطه مانند زیر عمل می‌کنیم. کلاس AddFighter شی مربوط به خدنگ و موقعیت ایجاد آن را دریافت میکند و در صورتی که ممکن باشد آن را می‌سازد.

WisePlayer.java

```java
import ir.pint.soltoon.soltoongame.shared.data.Player;
import ir.pint.soltoon.soltoongame.shared.data.action.Action;
import ir.pint.soltoon.soltoongame.shared.data.action.AddFighter;
import ir.pint.soltoon.soltoongame.shared.data.map.GameBoard;
```

```java
public class WisePlayer extends Player {
   @Override
   public void init(GameBoard gameBoard) {

   }

   @Override
   public void lastThingsToDo(GameBoard gameBoard) {

   }

   @Override
   public Action getAction(GameBoard gameBoard) {
       return new AddFighter(new WiseAttacker(), 1, 1);
   }
}
```

حال ما وظایف خود را انجام داده‌ایم. برای اجرای بازی لازم است ابتدا کد شرکت کنندگان بازی و سپس سلطون‌سرور را اجرا کنیم.

قدم سوم: اجرای کد خودمان

برای اجرای کد خودمان بعنوان بازیکن اول لازم است مطابق زیر عمل کنیم. لازم است به ورودی متد ClientRunner.run کلاس سلطون‌مان را ورودی بدهیم.

Client1.java

```java
import ir.pint.soltoon.soltoongame.client.ClientRunner;

public class Client1 {
   public static void main(String[] args) {
       ClientRunner.run(WisePlayer.class);
   }
}
```

قدم چهارم: اجرای کد رقیب‌مان

برای اجرای کد رقیب‌مان بعنوان بازیکن دوم کافی است مانند زیر عمل کنیم. البته در حال حاضر ما رقیب خاصی نداریم. بازی سلطون برای راحتی شما چند رقیب آزمایشی آماده کرده است. ما در اینجا از رقیب آزمایشی RandomPlayer استفاده می‌کنیم. این رقیب آزمایشی بطور تصادفی در نقاطی از زمین خدنگ ایجاد می‌کند و این خدنگ ها در صورت امکان جابجا می‌شوند.

Client2.java

```java
import ir.pint.soltoon.soltoongame.client.ClientRunner;
import ir.pint.soltoon.soltoongame.client.implementations.RandomPlayer;
```

```java
public class Client2 {
   public static void main(String[] args) {
       ClientRunner.runPlayerTwo(RandomPlayer.class);
   }
}
```

قدم نهایی: اجرای سرور

برای اجرای سرور دو نفره از دستور زیر استفاده میکنیم.

Server.java

```java
import ir.pint.soltoon.soltoongame.server.ServerRunner;

public class Server {
   public static void main(String[] args) {
       ServerRunner.runTwoPlayers();
   }
}
```

سناریوهای بازی

سناریوی ۱-: نگاه کن !

یک سلطون طراحی کنید که هنگام صدا زدن متد getAction اطلاعات زیر را به ازای هر خدنگ روی زمین بازی بنویسد:

- شناسه خدنگ (id)
- شناسه سلطون صاحب خدنگ
- مکان خدنگ
- نوع خدنگ
- شناسه خدنگ های که در محدوده شلیک این خدنگ

معیار عملکرد شما

- درستی توصیف شما از بازی

سناریوی صفر: سلام دنیا !

در این سناریو شما پول بسیار زیادی دارید و اجازه دارید هنگام شروع سلطون‌سرور طول و عرض زمین بازی را مشخص کنید. وظیفه شما این است که با پر کردن خانه های بازی از خدنگ ها نام خودتان را روی صفحه بازی رسم کنید. برای این کار باید سلطونی طراحی کنید که این کار را انجام دهد.

معیار عملکرد شما

- صحت و زیبایی کارتان

سناریوی ۱: ماجراجو

در شروع سناریو شما باید یک خدنگ متحرک بسازید و بعد منتظر باشید تا یک خدنگ بی‌دفاع حریف روی یک نقطه‌ی تصادفی زمین بازی ظاهر شود، بعد باید هرچه سریع‌تر به او نزدیک شوید و او را بکشید و همین اتفاق به طور پیوسته تکرار می‌شود. تضمین می‌شود که در هر لحظه حداکثر یک خدنگ حریف روی زمین است.

جزئیات بیشتر این سناریو در نسخه بعدی متنشر خواهد شد.

معیار عملکرد شما

- بهینه بودن حرکت‌هایتان

سناریوی ۲: مدافع با بودجه‌ی اولیه‌ی معلوم

در شروع سناریو به شما مقدار مشخصی پول داده شده است. شما باید در ابتدای بازی تعدادی خدنگ ثابت در نقاط مختلف زمین ایجاد کنید. سپس حریفتان که پولش هر لحظه زیادتر می‌شود، با ساختن سرباز و حمله به خدنگ‌های شما سعی در نابود کردن تمامی آنها دارد و خدنگ‌های ثابت شما باید از خودشان دفاع کنند.

جزئیات بیشتر این سناریو در نسخه‌های بعدی متنشر خواهد شد.

معیار عملکرد شما

- معیار عملکرد شما مدت زمانی است که دوام می‌آورید.

سناریوی ۳: مهاجم با بودجه‌ی رو به رشد و مواضع مشخص حریف

در شروع سناریو حریفتان تعدادی برجک در نقاط از پیش مشخص شده ایجاد می‌کند. شما باید با ساختن سربازان و حمله به او استحکاماتش را نابود کنید. گفتنی است که در سناریوهای مهاجم با گذشت زمان پولتان زیاد می‌شود.
جزئیات بیشتر این سناریو در نسخه‌های بعدی متنشر خواهد شد.

معیار عملکرد شما

- معیار عملکرد شما سرعت‌تان در تخریب استحکامات حریف است.

سناریوی ۴: مدافع با بودجه‌ی اولیه‌ی نامعلوم

تفاوت این سناریو با سناریوی ۲ این است که مقدار اولیه‌ی پولتان معلوم نیست و برنامه‌ی شما به طور خودکار باید بودجه‌بندی کند و استحکامات بسازد و آماده‌ی دفاع شود.
جزئیات بیشتر این سناریو در نسخه‌های بعدی متنشر خواهد شد.

معیار عملکرد شما

- معیار عملکرد شما مدت زمانی است که دوام می‌آورید.

سناریوی ۵: مهاجم با بودجه‌ی رو به رشد و مواضع نامشخص حریف

تفاوت این سناریو با سناریوی ۳ این است که شما از قبل نمی‌دانید استحکامات حریفتان به چه شکل خواهد بود.
جزئیات بیشتر این سناریو در نسخه‌های بعدی متنشر خواهد شد.

معیار عملکرد شما

- معیار عملکرد شما سرعت‌تان در تخریب استحکامات حریف است.

سناریوی ۶: جنگ

- شما و حریفتان با بودجه‌ی اولیه‌ی مساوی شروع میکنید و پولتان با گذشت زمان و با نرخ ثابتی زیاد میشود.
- هرکدامتان میتوانید در لحظات مختلف خدنگ‌هایی ایجاد کنید و گاهی دفاع کنید و گاهی حمله.
- در صورتی که بازی با درگیری ناچیز خدنگ‌ها همراه باشد برای هر دو سلطون یک عملکرد بد تلقی می‌شود.

جزئیات بیشتر این سناریو در نسخه‌های بعدی متنشر خواهد شد.

معیار عملکرد شما

- معیار عملکرد شما در پایان بازی، امتیازتان است.

جنگ جهانی سلطون

بعد از اجرای کامل سناریوی ۶ یک دوره مسابقه میان سلطون‌ها برگزار خواهد شد. این مسابقات ابتدا به طور گروهی - حذفی انجام می‌شود. معیار عملکرد شما در این بازی ها امتیاز شماست.

جزئیات بیشتر در نسخه‌های بعدی متنشر خواهد شد.

مرجع

مطابق با نسخه 1.0.4

https://soltoon.github.io/releases/1.0.4/api-docs/

اجرای کلاینت

کلاس ClientRunner	
نام متد	**عملکرد**
`run(Class<? extends Player> player)`	اجرای کلاینت بعنوان بازیکن اول. ورودی کلاس سلطون شماست.
`runPlayerTwo(Class<? extends Player> playerClass)`	اجرای کلاینت بعنوان بازیکن دوم. ورودی کلاس سلطون شماست.
`run(Class<? extends Player> player, ComRemoteConfig remoteConfig)`	اجرای کلاینت. ورودی اول کلاس سلطون شماست. ورودی دوم مشخصات اتصال شماست.

اجرای سرور

کلاس ServerRunner	
نام متد	**عملکرد**
`run()`	اجرای سناریوی ۱ با یک بازیکن
`runTwoPlayers()`	اجرای سناریوی ۶ با دو بازیکن
`runFirstScenario()`	اجرای سناریوی ۱ و اتصال خودکار به بازیکن اول
`runFirstScenario(ComRemoteInfo comRemoteInfo)`	اجرای سناریوی ۱. ورودی مشخصات شبکه بازیکن است.
`runSecondScenario(int width, int height)`	اجرای سناریوی ۲. ورودی طول و عرض صفحه بازی است.
`runSecondScenario(ComRemoteInfo remoteInfo, int width, int height)`	اجرای سناریوی ۲. ورودی مشخصات بازیکن و طول و عرض صفحه بازی است.

زمان اجرا

در نسخه‌های آینده منتشر خواهد شد.

خروجی موقت

در نسخه‌های آینده منتشر خواهد شد.

تغییر در ثابت های بازی

در نسخه‌های آینده منتشر خواهد شد.