人狼知能世界大会 2019 レギュレーション

2019/05/9 Ver1.2

1. 人狼知能世界大会における人狼のルール

1.1.各役職の人数

本大会では，15 人および 5 人のプレイヤーで村を構成します.

各役職のプレイヤー数は以下の通りです.

・15 人村

・村人	8 人
・占い師	1 人
・霊媒師	1 人
・ボディーガード	1 人
・人狼	3 人
・裏切り者	1 人

・5 人村

・村人	2 人
・占い師	1 人
・人狼	1 人
・裏切り者	1 人

1.2.役職説明

1.2.1. 村人側の役職

村人側の役職は「村人」，「占い師」，「霊媒師」，「ボディーガード」の 4 種類存在します.

村人

何も能力を持たない村人側のプレイヤーです.

占い師

1 日の終わりに 1 人のプレイヤーを占い，そのプレイヤーが人間であるか人狼であるか知ることが出来ます.

霊媒師

あるプレイヤーが追放された際に，そのプレイヤーが人間であるか人狼であるか知ることが出来ま

す.

ボディーガード

1 日の終わりに自分以外の 1 人のプレイヤーを護衛し，そのプレイヤーを人狼の襲撃から守ることが出来ます．本大会では，襲撃が無かった場合に手応え（意図的襲撃ミスなのか，護衛が成功したのか）を得ることは出来ません.

1.2.2. 人狼側の役職

人狼側の役職は「人狼」，「裏切り者」の 2 種類存在します.

人狼

1 日の終わりに各人狼は，人間を 1 人選択して襲撃投票し，最も多く襲撃投票されたプレイヤーを襲撃します．また，人狼だけが聞くことの出来る「囁き」で，村人に隠れて人狼同士会話することが出来ます.

裏切り者

村人と同じく能力は何も無い人間ですが，人狼の勝利を目指して行動します． 占い師や霊媒師の能力では人間と判定されます.

1.3.対話の仕様

1.3.1. ターン制の導入

対話はターン制となります． 各プレイヤーはターンごとに一回発言するチャンスがあります． ただし，発言をしないことも可能です． 各エージェントの発言はまとめて他のエージェントに送られます． 順番はランダムなので，同一ターン内の発言順に意味はありません.

発言は 1 日に 10 回まで行うことが出来ます． ただし，Skip と Over は発言に含まれません.

全てのプレイヤーが Over を発言するか，全てのプレイヤーが Skip をするターンが 3 回連続すると，その日の昼のフェーズは終了します．また，昼のフェーズは最大 20 ターンです．20 ターン経過した時点で昼のフェーズは終了します.

1.3.2. 初日対話

初日に対話はできません.

1.3.3. 人狼の囁き

人狼の囁きは襲撃対象投票前（投票決定後）にのみ行うことが可能です． 昼フェーズの対話と同じようにターン制で行われます． 初日は襲撃がありませんが，対話を行うことは可能です． 偽 CO 等をする際に初日の囁きを使って相談することが可能です.

人狼が一人になると囁きは行われません．whisper メソッド自体が呼ばれなくなりますのでご注意ください．

1.4.投票と再投票

追放のための投票は昼フェーズの最後に行われます．追放者の情報を，夜フェーズの占い，護衛，襲撃決定に利用することが可能となります．

最多得票者が複数いた場合は 1 回のみ再投票となります．このとき対話はできません．二回目の投票でも同点だった場合は最多得票者からランダムに追放者が決定されます．このとき，最多得票者も投票権を持ち，最多得票者以外にも投票可能です．

襲撃のための投票の場合も同様に，1 回のみ再投票となります．再襲撃投票の前にも囁きは行えません．

1.5.特殊能力

1.5.1.　　占い

占い師は夜のフェーズで占い先を指定して，そのエージェントが人狼かどうかを知ることが出来ます．占い前に追放された人を知ることが出来ます．占いは初日にも行うことが出来ます．

1.5.2.　　霊媒

霊媒師は前日追放されたエージェントが人狼だったかどうかを知ることが出来ます．追放者のいない 0 日目，1 日目は何の情報もありません．

1.5.3.　　護衛

護衛によって，自分以外の生きているエージェントを指定することで襲撃から護ることが出来ます．護衛先には死者を選ぶことも出来ますが，その場合は何も生じません．

護衛先決定の前にその日の追放者を知ることが出来ます．

2. 予選，本戦の試合方式

2.1.予備予選

予選開始まで，登録エージェント同士で対戦する予備予選を行います．一日に 5 人ゲームと 15 人ゲームを各エージェントが最低一回ずつは行うように回します．

なお，登録エージェントが 15 エージェント以下だった場合は，サンプルエージェントが入ります．

予備予選のログは，登録ページからダウンロードすることが可能です．

2.2. 予選

予選は，参加者の全チームのゲーム回数が一定回数以上となるまで以下の試行を繰り返し，平均取得ポイントの上位チームを選出します．予選通過チーム数は，15 チームとなります．
（※ゲームの回数は 100 回としていますが，ゲームの回数は時間の許す限り多く行うこととします．）

1 試合の流れ

全参加者のチームから 5 または 15 チームをランダムに選択して村を構成する．各チームにランダムに役職を振り分けゲームを行う（各役職の人数は 1.1 節参照）．勝利チームの各チームに 1 ポイント与える．これを 1 ゲームとする．
100 ゲームを同一の 5 または 15 チームで行う．

予選に参加する場合は，全役職のチームを実装したソースコードを提出してください（提出方法は「提出方法（3 章）」参照）．また，複数名でのチーム参加も可能です．

2.3. 本戦

予選で選出された 15 チームに対し、ランダムに役職を割り当てながら試合を行います．役職による重みはつけずに勝率によって順位を決定します．なお，各チームのインスタンスは 100 試合ごとに破棄されます．また，100 試合ごとに各チームに割り振られる AgentId はランダムに変更されます．

3. 提出方法

3.1. 参加登録

本大会に参加を希望される方は，人狼知能コンテストエントリーページより，アカウントを作成してください．

http://contest.aiwolf.org/

アカウントの作成を持って参加登録とします．

なお，予備予選のログもこちらからダウンロード可能です．

3.2. エージェント提出

参加登録後，全役職のチームを実装したファイルを同ページより提出して下さい．提出ファイルは実装言語によって異なります．

3.2.1. Java エージェント

提出可能ファイル：jar ファイル

Java エージェントを作成した場合，jar ファイル一つを提出可能です．機械学習などのライブラリなどを利用する場合は jar ファイルの中にライブラリの jar ファイルを含めておいてください．自動的にクラスパスに追加されます．

その他，データファイルを読み込ませたい場合も，jar ファイルの中に含めておくことで，読み込ませることができます．たとえば，jar ファイル内の/data/hoge.dat を読み込みたい場合は，

```
InputStream is = getClass().getClassLoader().getResourceAsStream("data/hoge.txt");
```

と書くことで，ファイルへの InputStream を得ることができます．

なお，aiwolf-client.jar，aiwolf-server.jar，aiwolf-common.jar，aiwolf-viewer.jar，jsonic-xxxx.jar を含めているとコンフリクトが発生し，正しく動作しない可能性があるため，これらのファイルを jar ファイルに含めるのはお控え下さい．

3.2.2. C#エージェント

提出可能ファイル：dll ファイル，zip ファイル

単一 dll ファイルで作成した場合は，dll ファイルを直接ご登録ください．複数 dll ファイルの場合は，zip ファイルに固めてご登録ください．

zip ファイルでエージェントを提出する場合は，プレイヤークラスの入った dll ファイルは，必ず「チーム名.dll」という名前にしてください．例えば，チーム名が tori ならば，「tori.dll」という名前でプレイヤークラスの入った dll ファイルを作成しないと起動しません．単一 dll の場合，従来通りファイル名の制限はありません．なお，zip 内にデータファイルやライブラリファイルを含めておくことはできますが，それ以外のファイルにアクセスしたことが分かった場合，当該エージェントは失格となります．

3.2.3. Python エージェント

Python で提出する方は zip で圧縮して提出してください．コンテストサーバ側で zip 解凍に失敗したり実行が出来なかった場合は大会に参加することが出来ません．

エージェント登録時には起動スクリプト名をご指定ください．

なお，起動スクリプトは zip ファイル内にチームメイト同じディレクトリを作成し，その下に配置してください．例えば，「team_hogehage」というチーム名で登録し，起動スクリプト名が「start_mokemoke.py」だとすると，

```
team_hogehage/start_mokemoke.py
```

という形で，team_hogehage ディレクトリ内に start_mokemoke.py を配置することになり

ます．なお，起動スクリプトは，start_mokemoke.py と登録してください．
すなわち，すべてのスクリプトはチーム名以下のディレクトリに配置するものと考えてください．
なお，zip 内にデータファイルやライブラリファイルを含めておくことはできますが，それ以外のファイルにアクセスしたことが分かった場合，当該エージェントは失格となります．

4. 禁止事項

本大会では，以下の項目を禁止します．以下の項目に反したプログラムと実行中にエラーを出力するプログラムは予選不戦敗，もしくは個別に連絡を差し上げた上で対処致します．

- ContentBuilder で生成不可能な発言の利用
- ファイルへの書き込み（ファイルの読み込みは条件付きで可）
- ネットワークへの接続
- スレッドの立ち上げ
- 別プロセスでのプログラムの実行
- サーバからのリクエストに対して 100ms 以上の計算時間を要するプログラム（多少の誤差は大目に見ます）

ファイル読み込みについて

本大会においては，ファイルの読み込みはアップロードしたファイルのみ可とします．
Java の場合，アップロードした jar ファイル内に含まれるファイルにアクセス可能です．
.NET，Python の場合は zip ファイル内に含まれるファイルにのみアクセス可能です．
なお，zip ファイルは実行時に展開されるため，通常のファイルアクセスによってアクセス可能です．

5. プレイヤーのプログラム実装について

人狼知能大会は，http://www.aiwolf.org/server/
内の aiwolf-ver0.5.x を利用します．ゲームの起動方法は同ページ内の「AIWolf ゲームサーバ起動方法」を参照してください．
AIWolfCommon.jar 内の org.aiwolf.common.data.Player インターフェースを継承したプログラムがプレイヤーとしてゲームに参加することが出来ます．
また，それと同様の通信が可能な.NET，Python 等のプログラムも参加することが出来ます．

5.1. Player インターフェースの実装すべきメソッド（JAVA 版）

Player インターフェースを継承したクラスは 11 個のメソッドを実装する必要があります．これらのメ

ソッドは以下の 4 種類に分類されます.

- 情報処理メソッド： initialize, update, dayStart, finish
- 対象指定メソッド： vote, attack, guard, divine
- 会話メソッド： talk, whisper
- 命名メソッド： getName

5.1.1.情報処理メソッド（initialize, update, dayStart, finish）

これらは情報を処理するためのメソッドであり，何も戻り値を返す必要がありません.

initialize(GameInfo, GameSetting)

ゲーム開始時に一度だけ呼ばれます．引数として現在のゲーム状態を表す GameInfo とゲームの設定（各役職の人数等）を表す GameSetting が与えられます．（GameInfo と GameSetting については○○参照）

update(GameInfo)

initialize 以外の全てのメソッドの前に呼ばれます．引数としてゲーム内の最新の情報を含んだ GameInfo が与えられます．finish()の前に呼ばれる時のみ，全プレイヤーの役職情報を含んだ GameInfo が与えられます.

dayStart()

毎日の始めに一度だけ呼ばれます.

finish()

ゲーム終了時に呼ばれます.

5.1.2.対象指定メソッド（vote, attack, guard, divine）

対象となる Agent を戻り値として返すメソッドです．attack，guard，divine は特定の役職のプレイヤーの場合のみ呼ばれるメソッドです.

vote()

その日に投票する対象プレイヤーを返します.

attack()

人狼のプレイヤーのみ呼ばれるメソッドです．その日に襲撃投票する対象プレイヤーを返します.

guard()

狩人のプレイヤーのみ呼ばれるメソッドです．その日に護衛する対象プレイヤーを返します.

divine()

占い師のプレイヤーのみ呼ばれるメソッドです．その日に占う対象プレイヤーを返します.

5.1.3.会話メソッド（talk, whisper）

発言する内容（String 型）を返すメソッドです．whisper は人狼のプレイヤーの場合のみ呼ばれます．

talk()

村全体に対して発言する内容を返します．本大会で用いる発言は org.aiwolf.client.lib.ContentBuilder で生成出来る発言のみです．（詳しくは「発言可能な内容（4.3 章）」を参照）

whisper()

人狼のプレイヤーのみ呼ばれるメソッドです．人狼だけに対して発言する内容を返します．この発言内容は人狼以外のプレイヤーに公開されることはありません．本大会で用いる発言は org.aiwolf.client.lib.ContentBuilder で生成出来る発言のみです．（詳しくは「発言可能な内容（4.3 章）」を参照）

5.1.4.命名メソッド（getName）

getName()

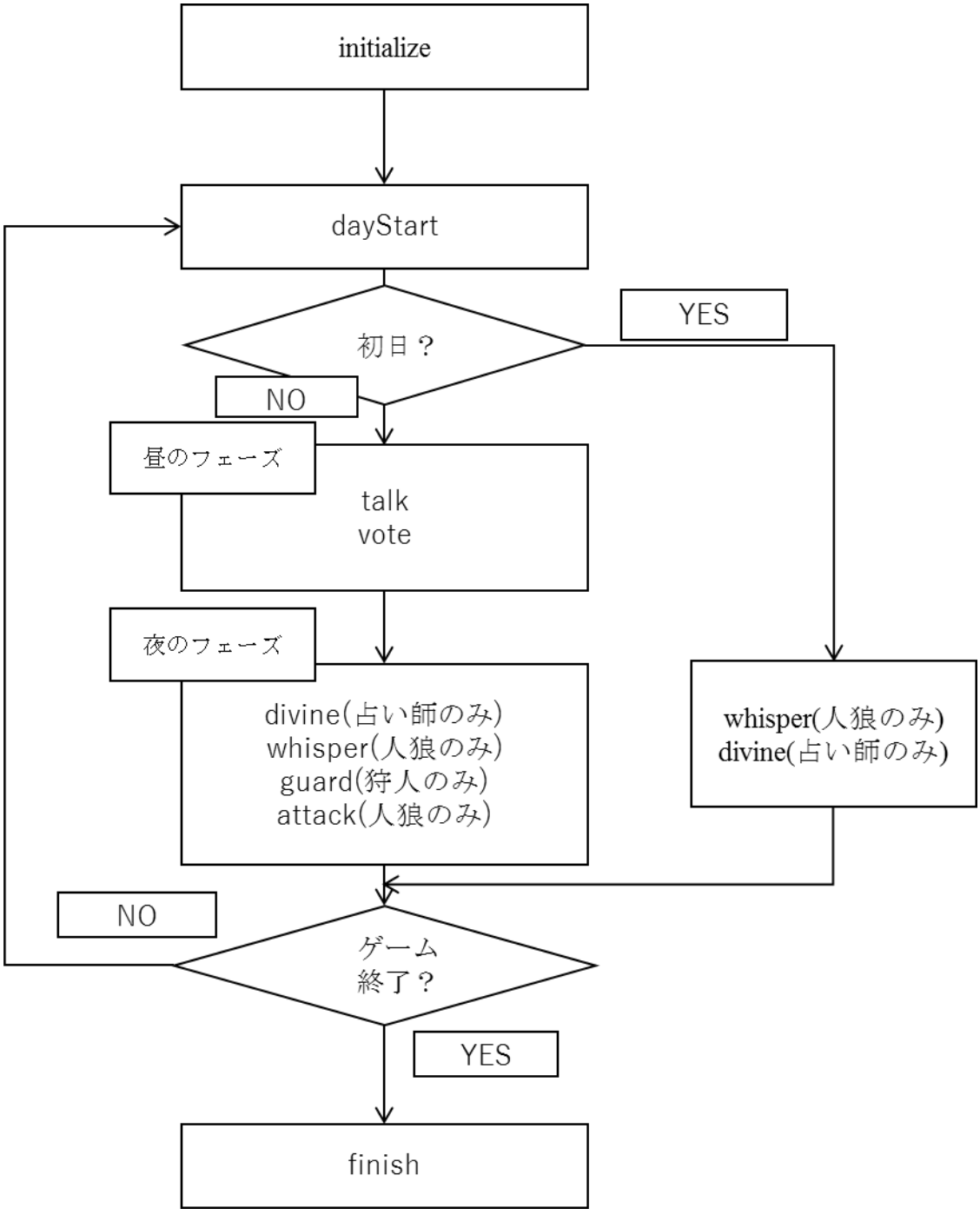
プレイヤーの名前（String 型）を返します．ゲーム実行時のログに名前が反映されます．

getName では人狼知能コンテストページで登録した「アカウント名」を返すようにしてください．それ以外の文字列を返した場合，決勝進出の対象とならない場合があります．

5.2.各メソッドの呼ばれるタイミング

4.1.節で挙げた 11 個のメソッドは以下のフローで呼び出されます（getName は除く）．update は省略していますが，initialize 以外の全てのメソッドの前に呼び出されます． また，会話の詳しい流れは 3.2.1 節を参照してください．

第二回大会とはタイミングが一部異なりますので，ご注意ください．



5.3.発言可能な内容

本大会では，org.aiwolf.client.lib.ContentBuilder を継承したクラスで生成可能な発言のみで会話を行います．生成可能な発言は以下の 23 種類です．

- estimate：□□が△△の役職は○○だと思う．(EstimateContentBuilder)

- comingout：□□が△△の役職を○○と明かす．(ComingoutContentBuilder)
- divination：□□が△△を占う．(DivinationContentBuilder)
- divined ： □□ が △△ を 占 っ た 結 果 ， ○○ （ 人 間 or 人 狼 ） だ っ た ． (DivinedResultContentBuilder)
- identified：□□の霊能行使の結果，△△は○○（人間 or 人狼）だった．(IdentContentBuilder)
- guard：□□が△△を護衛する．(GuardCandidateContentBuilder)
- guarded：□□が△△を護衛した．(GuardedAgentContentBuilder)
- vote：□□が△△に投票する．(VoteContentBuilder)
- voted ：□□が△△に投票した．(VotedContentBuilder)
- attack：□□が△△に襲撃投票する．（AttackContentBuilder）
- attacked ：□□が△△を襲撃した．(AttackedContentBuilder)
- agree：□□が発言 T に同意する．(AgreeContentBuilder)
- disagree：□□が発言 T に反対する．(DisagreeContentBuilder)
- request：□□が△△に～を要請する．(RequestContentBuilder)
- inquire：□□が△△に～を照会する．(InquiryContentBuilder)
- because：□□が◇◇の理由で～を主張する．(BecauseContentBuilder)
- and：□□が A, B, … を同時に主張する．(AndContentBuilder)
- or：□□が A, B, … の少なくとも一つを主張する．(OrContentBuilder)
- xor：□□が A, B のどちらかを主張する．(XorContentBuilder)
- not：□□が～を否定する．(NotContentBuilder)
- day：□□が x 日目に～であったと主張．(DayContentBuilder)
- over：もう話すことは無い． （全プレイヤーが OVER なら会話フェーズ終了） (OverContentBuilder)
- skip：様子見（他のプレイヤーが全員 OVER でも会話フェーズが終了しない） (SkipContentBuilder)

5.4. Player クラスのパッケージについて

Player クラスは独自の物を作成してください．デフォルトである
org.aiwolf.sample.player.SampleRoleAssignPlayer
などを直接書き換えることは避けてください．

Player クラスは，独自パッケージに配置するようにしてください．通常パッケージはドメイン名また

はメールアドレスを逆から利用して作成します．

たとえば，gm@aiwolf.org というメールアドレスをお持ちの方が MyPlayer という Player クラスを作成する場合，パッケージとして org.aiwolf.gm を指定し，

```
package org.aiwolf.gm;
import org.aiwolf.sample.lib.AbstractRoleAssignPlayer;

public class MyPlayer extends AbstractRoleAssignPlayer {

}
```

という書き方になります．

また，ソースコード提出時の「class path:」の欄には，

```
org.aiwolf.gm.MyPlayer
```

と指定してください．

なお，AbstractRoleAssignPlayer の場所が

org.aiwolf.sample.lib.AbstractRoleAssignPlayer

に変更になっていますのでご注意ください．

6. 他の言語で参加する場合

Java 以外の言語は Python，.NET を想定しています．これらの言語で参加する場合は，それぞれのライブラリを参照して下さい．独自のライブラリでも TCP-IP 接続が出来，エージェントが正しく動けば参加することが出来ますので事前に運営側にご相談(gm@googlegroups.com)ください．

なお，試合は Linux マシンで動きますので，その点を注意して特殊な環境に特化したプログラムはご遠慮下さい．ゲームサーバ上で動かなかったエージェントは自動的に失格となります．第 2 回大会の際に，Python の特定の JSON ライブラリが動かなかったことが確認されていますので，ご注意ください．事前の予備予選へのご参加を強く推奨いたします．

7. 更新

レギュレーションは予告なく変更される可能性があります．変更された場合は

プロジェクトページ http://aiwolf.org

開発者用 Slack https://aiwolfen.slack.com

で告知いたしますので，登録あるいはフォローをしておいてください.

8. 更新履歴

2019 年 5 月 8 日 Ver.1.2 作成
2019 年 3 月 31 日 Ver.1.1 作成
2019 年 3 月 28 日 Ver.1.0 作成