

Line War AI

Customization Manual

Last updated 2024-09-13
Basantos

Disclaimer

Some parts of the manual and the configuration system itself is not easy to understand nor adequately explained. Line War AI customization at this point is mostly intended for people who have at least some experience in programming, or are just generally very curios and stubborn! 😊

If you have any question, please let me know by jumping in to our [Discord](#). There, you can DM me (@Basantos) directly. Good luck!

Getting started

Customization is currently a **single-player only** feature. You can customize all AIs in the game using the same configuration system that the game uses internally: A JSON configuration file (look [here](#) if you are unfamiliar with this format).

To customize all AIs, first make a copy of the game's [original configuration file](#) named *AiConfiguration.json*. Now put this in your game's personal directory under this folder:

`C:\Users\<MY_NAME>\AppData\LocalLow\Studio Centurion AB\Line War\Ai`

To customize individual AIs, put another copy of the file in the same folder, adding the AIs player ID at the end. So, for example, to customize AI 3 in a game, put a *AiConfiguration3.json* in the folder.

AIs can be customized while playing! You will see messages printed in-game that confirms the presence, modification or removal of each config file. To revert back to using the original, just delete the file|s.

Understanding the AI scoring system

1. All AI construction and production decisions start with the maximum **score**: 1.00.
2. Aspects are then applied in order of appearance
 - a. First, the **relevance** is calculated. It's a value ranging from 0 ("zero relevance", "unimportant", "irrelevant", "not applicable" or whatever you want call it) to 1 ("maximum relevance").
 - b. The relevance is combined with the aspect **scale** (how heavy the aspect should be "weighed").
 - c. Relevance and scale is then applied to the current score, using a specified **operation** (see below)

Now, let's go through an example case study of FactoryConstruction, the logic that determines when and where a factory should be constructed:

```
{
  "Name": "FactoryConstruction",
  "Description": "Construction of factories.",
  "Aspects": [
    {
      "Type": "SurfaceThreatVicinity",
      "Scale": 0.5
    },
    {
      "Type": "AirStrikeThreat",
      "Scale": 0.5
    },
    {
      "Type": "ProducerToContinentSaturation",
      "Scale": 0.2
    },
    {
      "Type": "ConquestValueContinent",
      "Scale": 0.6,
      "Inverted": true
    },
    {
      "Type": "IdealProducerDistribution",
      "Scale": 0.6,
      "Inverted": true
    },
    {
      "Type": "SurfaceThreatContinent",
      "Scale": 1.3
    },
    {
      "$type": "Assets.Server.AI.Configuration.TacticalAspectTypeCount, Server",
      "Type": "TypeMaskCount",
      "Scale": 1.3,
      "Types": "AirWeaponized",
      "RegionType": "Continent",
      "Stance": "Hostile",
      "Ceil": 10
    }
  ]
}
```

Each entry under the “Aspects” object correspond to a *TacticalAspect*, the most important building block for analyzing a particular game situation. Now, take a look at the [logs](#) to find detailed calculation printed at regular intervals:

Construction Factory failed at s:0.43 <= **0.75 (9 at (-67.6, 0.0, -31.5) (AI 2))**

r:0.00 => w:0.60 => c:1.00 * v:1.00 => s:1.00 [SurfaceThreatVicinity]

r:0.00 => w:0.60 => c:1.00 * v:1.00 => s:1.00 [AirStrikeThreat]

r:0.19 => w:0.30 => c:1.00 * v:0.87 => s:0.87 [ProducerToContinentSaturation]

r:0.31 => w:0.60 => c:0.87 * v:0.73 => s:0.63 [ConquestValueContinent | Inverted]

r:0.21 => w:0.60 => c:0.63 * v:0.68 => s:0.43 [IdealProducerDistribution | Inverted]

r:0.00 => w:1.30 => c:0.43 * v:1.00 => s:0.43 [SurfaceThreatContinent]

r:0.00 => w:1.30 => c:0.43 * v:1.00 => **s:0.43** [TypeMaskCount(AirWeaponized)]

See how each aspect is calculated one by one in order of appearance, until we arrive at **final score**? That's how it works! The info indicated in **bold** is the site (the center location of a context-specific cluster of entities) that is being evaluated. The abbreviations stand for

- **r**: Relevance - How relevant is this aspect given the site, game state, AI sight vs FoW (including memorized positions of enemy entities), etc?

- **w**: Weight - What weight (called “Scale” in config file, sorry!) is given to this particular aspect, i.e. how strong will the penalty (or reward, if $w > 1$) be, when combined with the relevance?
- **c**: Current score - Simply the current score value at this stage of the calculation
- **v**: Value - The value that will be added | subtracted | multiplied to the score, depending on relevance, inversion and other aspect configuration
- **s**: Score - Result score at this stage. This will be passed on to the next aspect until the final score has been calculated, or aborted earlier if it has already fallen below the **minimum score**. The minimum score (or “threshold”) is initialized as *GeneralMinimumConstructionScore* or *GeneralMinimumProductionScore* (see config file), then elevated for each preceding construction/production evaluation that got a higher score. It is also modified by context-sensitive factors such as capital availability forecast, economic side-effects of constructing a particular structure, etc. For production specifically, the final minimum score is actually an *average score* among production candidates, reduced by 10% to allow for closely competing scores to still qualify.

The Tactical Aspect

Here’s an example of a standard TacticalAspect configuration. You can add, remove and rearrange these, as long as they stay underneath the “Aspects” container (be careful with the commas!):

```
{
  "Type": "IdealProducerDistribution",
  "Scale": 0.6,
  "Inverted": true,
  "Operation": "Multiply"
},
```

- **Type**: The name of the aspect that you want to configure
- **Scale**: The base value that will be either added, subtracted, multiplied or “restored” to the current score, after being weighed against situational relevance.
- **Inverted**: Set this to true if you want the scale instead be fully applied at zero relevance. The default behaviour is for the scale to be fully applied at *max* (1.0) relevance.
- **Operation**: Set to one of
 - Add: Scale (weighed by relevance) will be added to the current score
 - Subtract: Scale (weighed by relevance) will be subtracted from the current score
 - Multiply: Scale will be weighed by relevance from 1 (zero relevance) to Scale (max relevance), then multiplied to current score.
 - Restore: Hmmmm, how can I explain this? It will serve to “restore” score back towards 1, based on scale and relevance. A scale of 0.5 combined with a max relevance of 1 would mean that a current score of, let’s say, 0.1, would “LERP” half way back to being fully restored, i.e. 0.55. Or, if you prefer this

```
case ScoreOperation.Restore:
  value = (current < 1 ? (1 - current) * relevanceFinal * Scale : 0);
  score = current + value;
  sign = '^';
  break;
```

Here’s a list of all currently available aspects:

Aspect name	Description
ArtilleryRatioVicinity	The ratio between artillery : non-artillery land units in the vicinity.
SurfaceThreatVicinity	The level of threat from enemy surface units in the vicinity (in relation to strength of comrades).
SurfaceThreatContinent	The level of threat from enemy surface units on the continent (in relation to strength of comrades).
ConquestValueContinent	Income difference obtained by conquering the continent/sea. Relevance depends on actual need of more reinforcements.
ConquestValueTransContinent	Income difference obtained by conquering the sea body's neighboring continents. Relevance depends on actual need of more reinforcements.
ConquestValueShore	Income difference obtained by conquering and exploiting all shores connected to a sea body.
UnitToContinentSaturation	Should the continent's unit count be checked against it's total worth?
ProducerToContinentSaturation	Producer count visavi continent worth: Should the continent's producer count be checked against it's total worth?
IdealProducerDistribution	Military producer distribution: Should the amount of this particular producer be checked against an ideal distribution on the continent and in the world?
IdealSeaUnitDistribution	Sea unit distribution: Should the amount of this particular sea unit be checked against an ideal distribution in the sea body?
IdealAirUnitDistribution	Should the amount of this particular air unit be checked against an ideal distribution of all air units in the world?
EnergyNeed	Do we need more energy now or in the future?
RefineryProspects	Estimation of refinery construction prospects, i.e. the likelihood of being able to grab (or already possessing) a reasonably safe energy well on which to begin construction of more refineries.
KnownEnemyStructuresWithinRange	Are there known enemy structures within long-range? How close are they?
SpecialSitesWithinRange	Are there special sites (exploited or not) within long-range? How close are they?
NearWorldEquatorOrCenter	Is the site situated in the center of the world or at least near the equator?
AntiAirSurroundingThreat	How many enemy AA units are known near the site and to what extent do they surround it?
AirStrikeThreat	What is the level of threat from an air strike at this site?
PathSafe	Preferred safety of the path taken, in terms of harmful, guarding enemies along the stretch.
PathLength	What path length should be preferred for the mission? Higher values means generally greater distances travelled.
DestinationViableProxyOrSabotage	Destination is viable proxy or sabotage: Must the destination be a viable proxy site -or- house precious enemy destructibles?

DestinationViableDestruction	Does the destination house precious enemy destructibles?
Greed	A multiplier who's relevance is dictated by the Greed _parameter_.
ContinentScoutingNeed	The need for scouting on the continent, based on general visibility of it's territories (especially forests and mountains) and sightings of stealthy units.
Constant	Always returns full 1.0 relevance. Used to set an unconditional multiplier.
Randomize	A randomized value between 0 and 1 at every request.
Composite	An aspect that takes the max (best) score from any of it's children. Special configuration options • Method: Only current supported value is "Max". • Aspects: An array containing child tactical aspects.
TypeMaskCount	A simple counting of how many entities matching the specified criteria exist in the specified region. Special configuration options • Types: Any combination of these types, separated by pipes " ". • RegionType: One of Territory, Vicinity, Continent, World. • Stance: One of Comrade, Enemy, Neutral, Ally • Ceil: The value corresponding to a 1.0 relevance.
HazardousSite	Has a structure previously built on this special site been destroyed by the enemy recently?
TerrainDistribution	The ratio of the specified terrains, within some specified radii. Special configuration options • Terrain: Any combination of Plain, Mountain, Forest, Beach, Shallow, Sea, Polar, separated by pipes " ". • Radius: How far from the evaluated site should terrain be checked? Don't set higher than 75. • Ceil: The value corresponding to a 1.0 relevance.
ViableRocketBlastClusterWithinRange	Is a viable cluster of enemy entities within range?

Caveats

- Never change the number type of a value in the config file. If it's an integer, keep it as an integer. If it's a floating point, keep it as a floating point.
- Be on the lookout for errors appearing in the logs. There should be zero errors during a game. If one appears, it is most likely because of an error you made in the config file.

The original AI configuration

This is named *AiConfiguration.json* and can be found in the [application directory](#). It is advisable to compare your changes with this original from time to time. You can use a diff tool such as [WinMerge](#) for this purpose.

Enable verbose logging

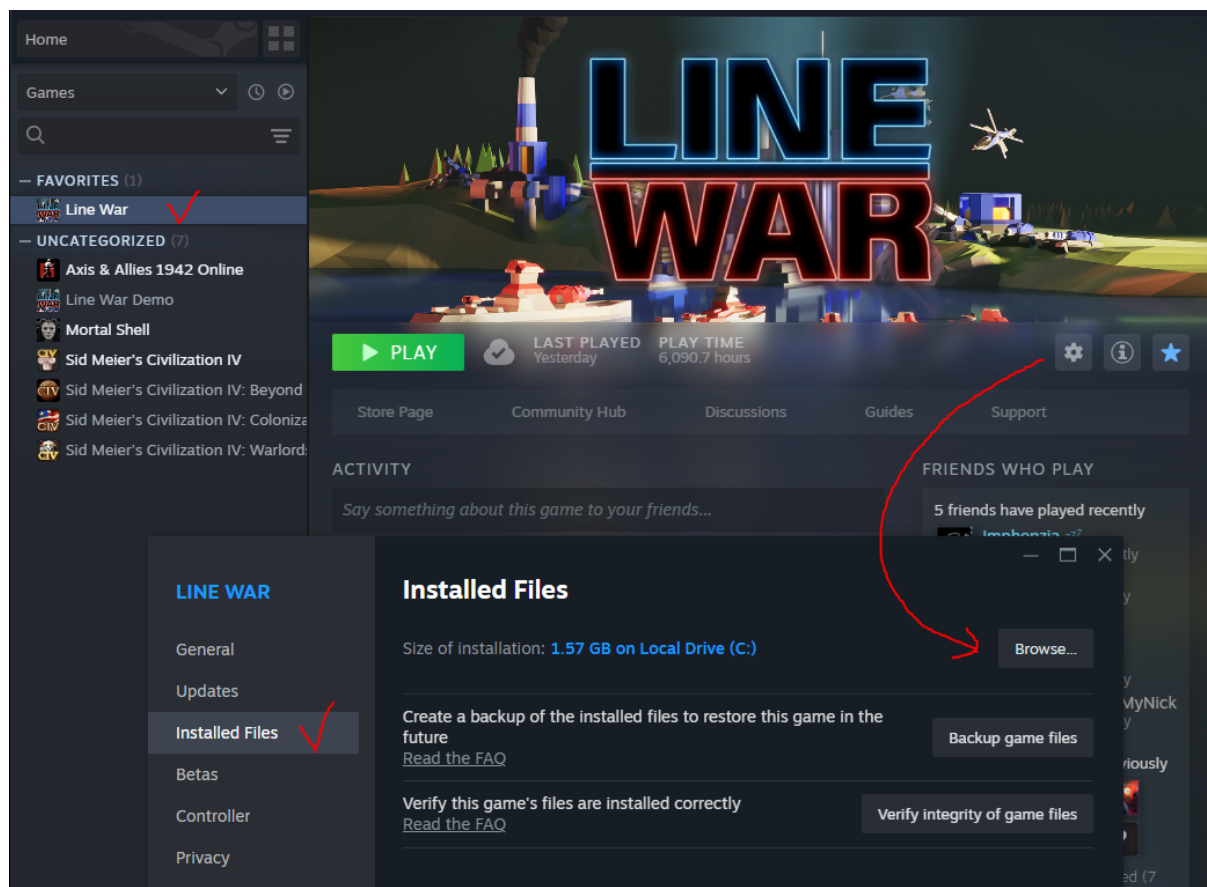
Just observing an AI's gameplay responses to a change in configuration is often not enough to fully understand the effects of a change. It can also be helpful to start the game with full logging. This option is **only supported in the beta environment**. Be aware that full logging has negative effects on the game's performance.

To start a full logging session, do the following:

1. Start the command prompt and go to the [application directory](#).
2. Run **LineWarClient.exe -logTags AI -logLevel Verbose**
3. Start a single-player game.
4. Watch 2 new log files appear in the [log directory](#). One of them is for the local server. This is the file to which AI decision making is printed.

Locate the Line War application directory

1. Click the cog wheel ⇒ Properties...
2. Click Browse...



Locate log files

1. Go to the [application directory](#).
2. Got to /Logs.
3. Delete old files if you want.
4. The 2 last modified files are the one's currently in use, if the game is running. One is for the local server, the other is for the client.

> Program Files (x86) > Steam > steamapps > common > Line War > Logs			
Name	Date modified	Type	Size
 Main_1.log	18/03/2024 17:48	Text Document	2 KB
 Main_353.log	18/03/2024 19:22	Text Document	5 KB

Strict entity type names

Infantry, Commando, Tank, Artillery, SAM, Interceptor, Striker, Helicopter, Bomber, TransportJet, Rocket, Submarine, Destroyer, MissileShip, LandingCraft, AircraftCarrier, Cruiser, MerchantShip, Territory, Barracks, Factory, Shipyard, SubPen, Airport, Airstrip, Refinery, Powerplant, Town, City, Depot, Port, Launchpad, ResearchCenter, IntelligenceCenter, SpaceCenter, Infrastructure, Fortification, Exploitation, PointCommand, LineCommand, PathCommand, TradeRouteCommand, AmphibiousCommand, EmbarkCommand, TargetCommand, DisbandCommand, InterAirbaseTransportCommand, TownSite, CoastalTownSite, DockSite, CitySite, CoastalCitySite, PortSite, EnergyWellSite, MineSite, LumberMillSite, FarmSite, Mine, LumberMill, Farm, TradeRoute, Entrenchment, Pipeline, PipelineSegment