Balanced Annihilation

Downloads

Intern

- Spring 104.0 Engine Installer für Windows
- Spring 104.0 Engine Portable für Windows
- Spring 104.0 Engine für Linux, vorkompiliert x64 static
- SpringLobby 0.264 Installer für Windows
- SpringLobby 0.264 Portable für Windows
- SpringLobby 0.264 Sources für Linux
- Balanced Annihilation 10.10 - Mod für Spring
- Balanced Annihilation - Mappack

Extern

- Spring-Engine
- SpringLobby
- Balanced Annihilation 10.10 - Mod für Spring

Spiel-Prinzip

Basierend auf der Spielmechanik von Total Annihilation, ist BA im Prinzip dasselbe, nur mit einer neuen Spiel-Engine realisiert. Der Techtree ist natürlich auch etwas anders, aber das Gameplay wurde entschieden verbessert. Das Spiel ist ein Mod der Spring-Engine. Diese erlaubt viele Spiele. Ein weiteres Spiel mit der TA-Mechanik ist Tech Annihilation.

Installation Client

Windows

1. Spring muss als erstes installiert werden. Aktuelle Installer enthalten offenbar auch bereits die SpringLobby. Allerdings sind noch keinerlei Spiele (die die Engine nutzen) installiert und ebenfalls keine Maps. Theoretisch könnte man jetzt bereits zum Punkt Multiplayer-Spiele weiter gehen, da Spiele und Maps auch von anderen Spielern in SpringLobby herunter geladen werden

können. Möchte man aber erst mal im Single-Player das Spiel kennenlernen, braucht man Spiel-Mod und Maps.
2. Deshalb muss man ins Verzeichnis `c:\Dokumente und Einstellungen\<Benutzername>\Eigene Dokumente\My Games\spring\games` den Balanced Annihilation-Mod speichern. Erst mit diesem hat man auch ein Spiel, aber noch keine Maps.
3. Wenn man jetzt noch eine gutes Interface haben möchte, mit dem man z. B. Online-Spiele starten bzw. beitreten will, bietet sich SpringLobby an.
4. Für die SpringLobby muss man im Normalfall noch die richtigen Pfade zu Spring setzen:
    1. Unter `Bearbeiten` → `Preferences` → `neuen Hinzufügen` anklicken
    2. Spring-Ordner raussuchen → `unitsync.dll` auswählen.
5. Nun sollte man noch das Mappack ins Verzeichnis `c:\Dokumente und Einstellungen\<Benutzername>\Eigene Dokumente\My Games\spring\` entpacken (es sollte am Ende also einen Order `[…]\spring\maps` geben). Für Linux sehen die Pfade ähnlich aus: `<homefolder>/.spring/[…]`.
6. Als letzten Schritt noch in SpringLobby über `Werkzeuge` → `Reload maps/games` die Liste der Maps und Spiele aktualisieren.

Linux

Unter Linux lässt sich das Spiel üblicherweise einfach über die Paketverwaltung installieren. Allerdings kommt es hier oft vor, dass man eine veraltete Version erhält. Es ist daher notwendig, andere Paketquellen hinzuzufügen, wenn man auf dem aktuellen Stand sein will. Wie man das macht, bzw. woher man diese kriegt, steht hier. Wie immer ist es natürlich Geschmacksache, ob man Fremdquellen im eigenen System zulassen will, oder nicht. Danach sollte man über eine Aktualisierung des Systems auch eine neuere Version von Springlobby erhalten.

Alternativ kann man auch Spring selbst im User-Space ablegen, mit der gewünschten Version. Spring kriegt man hier, um genauer zu sein, die portable-Versionen. Springlobby selbst hat ein ähnliches Problem mit veralteten Versionen. Es ist aber nicht so wichtig, dass man davon die neuste Version hat. Bei BA selbst sieht das auch wieder anders aus, aber hier ist es ja ohnehin so, dass das Spiel im User-Space liegt.

Installation Lobby-Server

Der Lobby-Server ist eine private Instanz des auch offiziell verwendeten Servers »Uberserver«. Die nachfolgende Anleitung bezieht sich auf ein Ubuntu Linux.

- Zu beziehen ist dieser über https://github.com/spring/uberserver. Dort das `*.zip`-File herunterladen (*nicht* nur die `server.py`!) oder mit `git clone` aus dem Repo ziehen.
- Es werden einige wenige Pakte für Linux benötigt (Python 3 wird benötigt, was aber bei aktuellen Distries normalerweise bereits installiert ist), der Rest kann über Python selbst installiert werden:
    1. `sudo apt-get install python3-pip geoip-database python-geoip libgeoip-dev`
    2. `sudo pip3 install Twisted sqlalchemy pyopenssl service_identity GeoIP`
- Gestartet wird der Server normalerweise via `./server.py --latestspringversion 98.0`,

oder `/home/ba/uberserver/server.py --latestspringversion 103.0` bzw. in unserem Fall über ein Init-Skript oder Systemd. Es gibt aber noch einige andere Start-Optionen, die man festlegen kann. Dazu ein Auszug aus der Help-Rückgabe:

<spoiler|Commandline-Help>

```
Usage: server.py [OPTIONS]...
Starts uberserver.
()
Options:
  -h, --help
      { Displays this screen then exits }
  -p, --port number
      { Server will host on this port (default is 8200) }
  -n, --natport number
      { Server will use this port for NAT transversal (default is 8201) }
  -g, --loadargs filename
      { Reads additional command-line arguments from file }
  -o, --output /path/to/file.log
      { Writes console output to file (for logging) }
  -u, --sighup
      { Reload the server on SIGHUP (if SIGHUP is supported by OS) }
  -v, --latestspringversion version
      { Sets latest Spring version to this string. Defaults to "*" }
  -m, --maxthreads number
      { Uses the specified number of threads for handling clients }
  -s, --sqlurl SQLURL
      { Uses SQL database at the specified sqlurl for user, channel, and ban
storage. }
  -c, --no-censor
      { Disables censoring of #main, #newbies, and usernames (default is to
censor) }
  --proxies /path/to/proxies.txt
     { Path to proxies.txt, for trusting proxies to pass real IP through
local IP }
   -a --agreement /path/to/agreement.txt
     { sets the pat to the agreement file which is sent to a client
registering at the server }
   -r --redirect "hostname/ip port"
     { redirects connecting clients to the given ip and port
SQLURL Examples:
  "sqlite:////absolute/path/to/database.txt"
     { uses a database in the file specified }
  "sqlite:///relative/path/to/database.txt"
     { note sqlite is slower than a real SQL server }
  "mysql://user:password@server:port/database?charset=utf8&use_unicode=0"
     { requires the MySQLdb module }
  "oracle://user:password@server:port/database"
     { requires the cx_Oracle module }
  "postgres://user:password@server:port/database"
     { requires the psycopg2 module }
```

```
  "mssql://user:password@server:port/database"
     { requires pyodbc (recommended) or adodbapi or pymssql }
  "firebird://user:password@server:port/database"
     { requires the kinterbasdb module }
()
Usage example (this is what the test server uses at the moment):
 server.py -p 8200 -n 8201
()
```

</spoiler> Die Version für `--latestspringversion` sollte natürlich ungefähr dem entsprechen, was gerade die tatsächlich aktuelle ist.

Multiplayer-Spiele

Für Multiplayer-Spiele sollte es es für ein bequemes Spiel einen Lobby-Server geben. Andernfalls müsste man das alles händisch einrichten, was viel Arbeit macht. Auf der mobilen Infanterie läuft bereits ein Lobby-Server, siehe auch die Status-Seite. Zum Beitreten benutzt man SpringLobby. Unter `Server` → `Verbinden…` kommt man in die Anmeldemaske. Dort muss die Server-Adresse der mobilen Infanterie (`www.mobile-infanterie.de`) bzw. `192.168.0.1` angegeben werden. Port ist der Standard-Port `8200`, da muss man nichts extra angeben. Bei erstmaliger Anmeldung ist die Erstellung eines Accounts notwendig, den man sich ohne Angabe weiterer persönlicher Daten anlegen lassen kann (Reiter `Registrieren`). Ab da an kann man sich mit den angegebenen Zugangsdaten auf dem Lobby-Server anmelden (Reiter `Anmelden`). Via `Battle-Room` kann man dann Server erstellen oder in `Battlelist` einem Server beitreten. Es kann nützlich sein, beim Hosten von Servern die Option `Hole Punching` im Kasten `NAT traversal` zu aktivieren, wenn man keine Port-Freigaben im Router setzen kann oder möchte. Wichtig für die Erstellung von Spielen ist ganz allgemein, dass man keine zwei Spieler ins selbe Team setzt. Anders sieht das beim Bündnis aus. Wollen zwei oder mehr Spieler zusammen gegen eine andere Gruppe spielen, sind sie in einem Bündnis, nicht in einem Team. Setzt man zwei Spieler oder mehr (oder auch Bots) in ein Team, startet das Spiel nicht richtig.

Bots

Im Prinzip hat man eine gewisse Auswahl unterschiedlicher KIs, die unterschiedliche Spielweisen an den Tag legen. Aber nicht alle Bots sind auch für BA brauchbar. Manche tun einfach gar nichts, andere lassen Spring abstürzen. Hier also mal eine kurze Übersicht über die verfügbaren Bots:

- **AAI 0.9**: Eine relativ leichte KI. Taugt allerdings in der Version 0.9 nur bis zu 3 Stück davon. Nimmt man mehr, dauert das Spiel meist zu lange und Spring stürzt irgendwann einfach ab.
- **E323AI 3.25.0**: Fängt langsam an, aber wehe er hat mal Wind davon bekommen, dass du auch auf der Map bist…
- **KAIK 0.13**: Recht saftiges Spiel…
- **RAI 0.601**: Ziemlich nervig, weil sie recht früh anfängt anzugreifen, aber machbar. Kackt auch irgendwann nach längerem Spiel ab.
- **Shard dev**: Harter Gegner, fängt sehr früh an zu beharken, scheint aber keinen Finish zu machen (oder hat's bei mir halt nicht hin bekommen).
- **CppTestAI 0.1**: Tut genau gar nichts (bringt also nichts, außer man will seine Builddauer unter

Laborbedingungen testen)
- **NullAI 0.1**: Macht gar nichts.
- **HughAI**: Macht gar nichts, Spring stürzt aber beim Tod des Commanders ab.
- **NullJavaAI 0.1**: Macht gar nichts, Spring stürzt aber beim Tod des Commanders ab.
- **NullOOJavaAI 0.1**: Macht gar nichts, Spring stürzt aber beim Tod des Commanders ab.

(Letzte KI-Tests mit Spring 100.00 und BA 9.07.)

SpringLobby kompilieren

Da für Windows i. A. ohnehin schon kompilierte Binaries angeboten werden, hier nur ein kurzer Abriss, wie das unter Ubuntu Linux läuft:

1. Die Sourcen von http://springlobby.info/landing/index.php runterladen und entpacken.
2. Die Schritte in der INSTALL ausführen:
    1. `cmake .` konfiguriert das Source-Paket vor, sodass der nachfolgende Schritt auf das eigene System zugeschnitten ablaufen kann. (Hinweis: Es wird CMake 3.1 oder neuer benötigt (SpringLobby Version 0.264))
    2. `make` führt die eigentliche Kompilierung aus.
    3. `sudo make install` installiert SpringLobby, braucht daher gewöhntlich root-Rechte. (Das ist ggf. nicht als root notwendig, wenn man mit `ccmake .` den Installationspfad ins `$home` gelegt hat.)

So oder sehr ähnlich sollte es auch auf allen anderen Linux-Distributionen laufen. Für Ubuntu 17.04 sind folgende Pakete vorher nachzuinstallieren (sofern man sie nicht ohnehin schon auf irgend einem Wege installiert hat): `build-essential`, `cmake`, `libwxgtk3.0-dev`, `libcurl4-nss-dev`, `zlib1g-dev`, `libpng-dev`, `libalure-dev` und `libboost-all-dev`. In einem knackigen Statement:

```
sudo apt-get install build-essential cmake libwxgtk3.0-dev libcurl4-nss-dev
zlib1g-dev libpng-dev libalure-dev libboost-all-dev
```

Für andere Versionen von Ubuntu kann es ggf. notwendig sein, andere Versionen der Pakete zu installieren. Auf anderen Distributionen heißen sie ggf. etwas anders, oder sind bereits von Haus aus installiert.

Weblinks

- BA-Seite
- BA-Forum

Zurück zur Games-Datenbank

From:
https://www.mobile-infanterie.de/wiki/ - **mwohlauer.d-n-s.name / www.mobile-infanterie.de**

Permanent link:
https://www.mobile-infanterie.de/wiki/doku.php?id=games:balanced_annihilation&rev=1560684390

Last update: **2019-06-16-11-26**