Introdução

Olá! Me chamo Gabriel, e antes de qualquer coisa, seja muito bem-vindo(a) ao curso.

Sou desenvolvedor há 8 anos, apaixonado por tecnologia, infraestrutura, programação e tudo que envolve criação de sistemas e servidores. Atualmente trabalho com desenvolvimento de softwares sob medida, aplicações empresariais, inteligência artificial, infraestrutura e diversas outras áreas da tecnologia.

Ao longo da minha trajetória, sempre gostei muito de ensinar e compartilhar conhecimento, principalmente de uma forma simples, prática e direta. Acredito que qualquer pessoa consegue aprender tecnologia quando o conteúdo é explicado da forma correta, sem complicação desnecessária.

Neste curso, iremos aprender juntos como criar e estruturar um servidor de Minecraft do zero, entendendo não apenas “o que fazer”, mas também o motivo técnico por trás de cada escolha realizada durante o processo.

O objetivo deste material não é apenas ensinar você a copiar um servidor pronto, mas sim construir uma base sólida para que você realmente compreenda como um servidor funciona, podendo evoluir futuramente para projetos muito maiores e mais profissionais.

O que iremos aprender?

Item 1 — Entendendo os tipos de servidores

Antes mesmo de criar um servidor, é importante entender quais tipos existem e qual objetivo cada um possui.

Durante esta etapa, iremos aprender:

- Diferença entre Survival, RankUP, Factions, SkyBlock, RPG, MiniGames e outros;
- Qual tipo de servidor exige mais desempenho;
- Quais são mais fáceis para iniciantes;
- Como escolher um modelo ideal para seu projeto.

Item 2 — Escolhendo a versão ideal do Minecraft

Uma das maiores dúvidas de iniciantes é sobre qual versão utilizar.

Nesta etapa iremos aprender:

- Diferença entre versões antigas e atuais;
- Compatibilidade entre plugins;
- Desempenho de cada versão;
- Como escolher uma versão estável;
- Quais versões são mais utilizadas atualmente.

Item 3 — Escolhendo o software do servidor

Muitas pessoas conhecem apenas o Spigot, porém atualmente existem diversas opções melhores dependendo da necessidade do projeto.

Iremos aprender:

- O que é Paper;
- O que é Purpur;
- Diferença entre Paper, Spigot, Bukkit e outros;
- Qual oferece melhor desempenho;
- Qual software escolher para cada tipo de servidor.

Item 4 — Instalação do Java e ferramentas necessárias

Aqui iremos preparar corretamente o ambiente para o servidor funcionar.

Nesta etapa iremos aprender:

- Como instalar o Java corretamente;
- Qual versão do Java utilizar;
- Como configurar variáveis e caminhos;
- Ferramentas úteis para administração;
- Editores de código recomendados.

Item 5 — Estrutura inicial do servidor

Após preparar o ambiente, iremos criar nosso primeiro servidor.

Iremos aprender:

- Como iniciar o servidor pela primeira vez;
- Como aceitar o EULA;
- Estrutura de arquivos;
- Organização de pastas;
- Arquivos importantes do servidor.

Item 6 — Criando o arquivo .bat e iniciando corretamente

Muitas pessoas simplesmente copiam comandos prontos da internet sem entender o que estão fazendo.

Aqui iremos aprender:

- O que é um arquivo .bat;
- Como criar um inicializador corretamente;
- Como alocar memória RAM;
- Explicação básica dos parâmetros de inicialização;
- Como iniciar o servidor de maneira correta e organizada.

Item 7 — Primeiros plugins e configurações

Após o servidor estar funcionando, começaremos a instalar e configurar plugins básicos.

Iremos aprender:

- Como instalar plugins;
- Como identificar plugins compatíveis;
- Estrutura da pasta plugins;
- Arquivos de configuração;
- Plugins essenciais para administração e segurança.

Item 8 — Noções básicas de otimização

Mesmo servidores simples podem apresentar problemas de desempenho quando mal configurados.

Nesta etapa iremos aprender:

- Noções básicas de otimização;
- Uso correto de memória RAM;
- Impacto de plugins mal otimizados;
- Conceitos básicos de TPS;
- Como evitar erros comuns de iniciantes.

Não importa se você nunca teve contato com servidores antes. Vamos construir tudo passo a passo, do zero, de forma organizada, prática e profissional.

Então prepare seu café, sua curiosidade e vamos começar essa jornada juntos.

Item 1 — Entendendo os Tipos de Servidores

Antes de criar qualquer servidor, precisamos entender uma das decisões mais importantes de todo o projeto: qual será o estilo do servidor.

Muitas pessoas começam instalando plugins aleatórios sem sequer definir um objetivo para o servidor. O resultado normalmente é um servidor desorganizado, pesado, confuso e sem identidade.

Um servidor bem planejado começa pela escolha do seu modelo principal.

Cada tipo de servidor possui:

- uma proposta diferente;
- uma comunidade diferente;
- necessidades diferentes;
- e níveis de complexidade diferentes.

Por isso, antes mesmo de instalar qualquer plugin, precisamos entender como funciona cada modalidade.

Survival

O Survival é o modelo mais tradicional e conhecido dentro do Minecraft.

Neste tipo de servidor, os jogadores possuem liberdade para:

- explorar o mapa;
- construir;
- minerar;
- sobreviver;
- evoluir equipamentos;
- interagir com outros jogadores.

Normalmente é o tipo de servidor mais recomendado para iniciantes, pois sua estrutura inicial costuma ser mais simples.

Vantagens do Survival

- Mais fácil de configurar;
- Grande quantidade de plugins disponíveis;
- Comunidade muito ativa;

- Ótimo para aprender administração de servidores;
- Menor complexidade inicial.

Desvantagens

- Pode exigir otimização conforme o mapa cresce;
- Servidores muito simples podem perder jogadores rapidamente;
- Necessita sistemas de proteção e administração.

RankUP

O RankUP é um dos modelos mais populares no Brasil.

Neste tipo de servidor, os jogadores evoluem através de:

- mineração;
- dinheiro;
- níveis;
- ranks;
- áreas desbloqueáveis.

A ideia principal é criar uma progressão constante.

Normalmente o jogador:

1. minera;
2. ganha dinheiro;
3. sobe de rank;
4. desbloqueia novas minas;
5. continua evoluindo.

Vantagens do RankUP

- Alta retenção de jogadores;
- Fácil monetização;
- Grande comunidade no Brasil;
- Sensação constante de progresso.

Desvantagens

- Requer mais plugins;
- Balanceamento pode ser complicado;
- Economia mal configurada pode destruir o servidor;

- Exige organização maior.

Factions

O Factions é um modelo focado em PvP e guerras entre grupos.

Os jogadores criam facções, dominam territórios, acumulam recursos e entram em conflitos com outros grupos.

É um estilo muito competitivo.

Características principais

- PvP intenso;
- Economia;
- Proteção de territórios;
- Sistema de clãs/facções;
- Guerras e invasões.

Vantagens

- Comunidade extremamente ativa;
- Alto fator competitivo;
- Grande engajamento entre jogadores.

Desvantagens

- Muito mais difícil de balancear;
- Pode exigir hardware melhor;
- Plugins mais complexos;
- Administração mais difícil;
- Maior necessidade de moderação.

SkyBlock

O SkyBlock é um modelo onde os jogadores começam em pequenas ilhas suspensas no vazio.

A progressão acontece através da expansão da ilha, economia e desafios.

É um modelo muito popular até hoje.

Vantagens

- Excelente retenção;
- Menor consumo de mapa;
- Progressão divertida;
- Fácil combinação com sistemas de economia.

Desvantagens

- Dependência maior de plugins;
- Balanceamento importante;
- Pode exigir configurações mais específicas.

RPG

Servidores RPG possuem foco em:

- história;
- progressão;
- classes;
- habilidades;
- missões;
- exploração.

Normalmente são os servidores mais complexos de desenvolver.

Vantagens

- Experiência extremamente imersiva;
- Grande potencial criativo;
- Comunidades muito fiéis.

Desvantagens

- Alto nível de complexidade;
- Grande quantidade de configuração;
- Desenvolvimento demorado;
- Necessidade maior de organização.

MiniGames

Servidores de MiniGames funcionam como uma coleção de modos rápidos e competitivos.

Exemplos:

- BedWars;
- SkyWars;
- The Bridge;
- BuildBattle;
- PvP Arena.

Vantagens

- Alta rotatividade de jogadores;
- Gameplay rápida;
- Muito populares.

Desvantagens

- Estrutura mais complexa;
- Necessidade de múltiplos servidores;
- Maior consumo de recursos;
- Configuração mais avançada.

Qual tipo de servidor escolher?

Não existe um modelo “melhor”. Existe apenas o modelo mais adequado para:

- seu objetivo;
- seu conhecimento;
- seu hardware;
- seu tempo disponível;
- e sua experiência.

Para iniciantes, normalmente recomendo:

- Survival;
- RankUP simples;
- ou SkyBlock básico.

Esses modelos permitem aprender:

- estrutura;
- plugins;
- administração;
- otimização;
- permissões;
- configuração;
- organização do servidor.

Sem adicionar complexidade desnecessária logo no início.

Um erro muito comum

Muitas pessoas tentam criar:

- Survival;
- RankUP;
- RPG;
- MiniGames;
- PvP;
- SkyBlock;

Tudo ao mesmo tempo.

Isso normalmente cria:

- servidores pesados;
- mal organizados;
- confusos;
- difíceis de administrar.

Um bom servidor não é aquele com mais plugins.

Um bom servidor é aquele que possui:

- organização;
- estabilidade;
- identidade;
- e uma proposta clara.

Conclusão

Agora que entendemos os principais tipos de servidores, já conseguimos tomar decisões muito melhores antes mesmo de iniciar nossa estrutura.

Nos próximos tópicos iremos aprender:

- como escolher a versão ideal do Minecraft;
- qual software utilizar;
- e como preparar corretamente nosso ambiente para criar o servidor.

Item 2 — Escolhendo a Versão Ideal do Minecraft

Após definir qual será o estilo do servidor, o próximo passo é escolher a versão do Minecraft que será utilizada.

Essa escolha é extremamente importante, pois impacta diretamente:

- desempenho;
- compatibilidade;
- quantidade de plugins;
- estabilidade;
- experiência dos jogadores;
- e até mesmo o consumo de hardware.

Muitas pessoas escolhem a versão apenas por ser “a mais nova”, porém isso nem sempre é a melhor escolha.

Cada versão possui vantagens e desvantagens, e entender isso evita muitos problemas futuramente.

Como funcionam as versões do Minecraft?

O Minecraft recebe atualizações constantemente.

Cada atualização pode alterar:

- mecânicas do jogo;
- geração de mapa;
- entidades;
- sistema de blocos;
- comandos;
- desempenho;
- compatibilidade com plugins;
- funcionamento interno do servidor.

Isso significa que um plugin feito para uma versão pode não funcionar corretamente em outra.

Por isso, a escolha da versão deve ser feita com planejamento.

Versões antigas vs versões atuais

Versões antigas

Exemplos:

- 1.7.10
- 1.8.8
- 1.12.2

Essas versões ficaram extremamente populares por muitos anos, principalmente em servidores PvP e Factions.

Vantagens

- Grande quantidade de plugins antigos;
- Menor consumo de hardware;
- PvP clássico muito apreciado pela comunidade;
- Excelente desempenho em máquinas mais simples.

Desvantagens

- Menos recursos modernos;
- Menor compatibilidade com plugins atuais;
- Segurança inferior;
- APIs mais antigas;
- Menor suporte da comunidade atualmente.

Versões atuais

Exemplos:

- 1.20
- 1.20.1
- 1.20.4
- versões mais recentes.

As versões atuais possuem:

- novos blocos;
- novas mecânicas;
- otimizações;
- novos sistemas;
- melhor compatibilidade com plugins modernos.

Vantagens

- Recursos mais modernos;
- Melhor experiência visual;
- Maior suporte da comunidade;
- Melhor compatibilidade futura;
- APIs atualizadas.

Desvantagens

- Maior consumo de hardware;
- Mais entidades e processamento;
- Necessidade maior de otimização;
- Alguns plugins antigos deixam de funcionar.

Qual versão escolher?

A resposta depende totalmente do objetivo do servidor.

Para Survival moderno

Normalmente recomenda-se:

- versões atuais estáveis.

Isso permite:

- novos recursos;
- melhor experiência para jogadores;
- maior compatibilidade futura.

Para PvP e Factions

Muitos servidores ainda utilizam:

- 1.8.8

Isso acontece porque:

- o PvP antigo é extremamente popular;
- a performance costuma ser excelente;
- muitos plugins clássicos foram desenvolvidos para essa versão.

Para projetos modernos e profissionais

O ideal normalmente é utilizar:

- versões atuais estáveis;
- evitando versões recém-lançadas.

Isso porque versões muito novas ainda podem:

- possuir bugs;
- apresentar incompatibilidades;
- ter plugins desatualizados.

Evite versões extremamente recentes no início

Um erro muito comum é instalar imediatamente a versão mais nova lançada.

O problema é que:

- plugins podem não funcionar;
- ferramentas ainda não estão adaptadas;
- bibliotecas podem quebrar;
- otimizações ainda não foram aplicadas.

Por isso, normalmente é melhor utilizar versões:

- estáveis;
- consolidadas;
- já utilizadas pela comunidade.

Compatibilidade entre jogadores

Hoje existem plugins e proxies que permitem múltiplas versões acessando o mesmo servidor.

Exemplo:

- jogador 1.20 entrando em servidor 1.20.1;

- jogador 1.19 entrando em servidor mais recente.

Porém isso exige:

- configuração;
- plugins específicos;
- testes de compatibilidade.

No início, o ideal é manter tudo simples.

Como descobrir se uma versão é boa?

Antes de escolher uma versão, pesquise:

- quantidade de plugins compatíveis;
- estabilidade da versão;
- desempenho;
- popularidade;
- suporte da comunidade.

Servidores profissionais normalmente evitam:

- decisões impulsivas;
- atualizações imediatas;
- mudanças sem testes.

Uma recomendação importante

Se você está começando agora, normalmente recomendo:

Para servidores modernos

- versões atuais estáveis.

Para PvP/Factions clássicos

- 1.8.8 ainda é uma excelente opção.

O mais importante não é usar “a versão mais nova”.

O mais importante é:

- estabilidade;
- compatibilidade;
- desempenho;
- organização.

Conclusão

Agora já entendemos:

- como funcionam as versões do Minecraft;
- diferenças entre versões antigas e modernas;
- vantagens e desvantagens de cada uma;
- e como escolher uma versão adequada para nosso projeto.

No próximo tópico iremos aprender algo extremamente importante:
qual software utilizar para o servidor, entendendo as diferenças entre:

- Bukkit;
- Spigot;
- Paper;
- Purpur;
- e outras opções modernas.

Item 3 — Escolhendo o Software do Servidor

Agora que já entendemos os tipos de servidores e aprendemos como escolher uma boa versão do Minecraft, chegou o momento de escolher o software que irá gerenciar nosso servidor.

Essa é uma das decisões mais importantes de toda a estrutura.

Muitas pessoas acreditam que “Minecraft Server” é apenas um único programa, porém atualmente existem diversos softwares diferentes, cada um com:

- características próprias;
- desempenho diferente;
- sistemas específicos;
- vantagens e desvantagens.

Escolher o software correto impacta diretamente:

- desempenho;
- estabilidade;
- compatibilidade;
- otimização;
- e experiência dos jogadores.

O que é o software do servidor?

O software do servidor é basicamente o núcleo responsável por:

- iniciar o servidor;
- processar jogadores;
- carregar mundos;
- gerenciar entidades;
- executar plugins;
- controlar eventos do jogo.

Sem ele, o servidor simplesmente não funciona.

Atualmente, os principais softwares utilizados são:

- Vanilla;
- Bukkit;
- Spigot;
- Paper;

- Purpur.

Vanilla

O Vanilla é o servidor oficial da Mojang.

Ele possui apenas os recursos básicos do Minecraft, sem suporte nativo para plugins.

Vantagens

- Servidor oficial;
- Estrutura simples;
- Ideal para experiências totalmente puras.

Desvantagens

- Sem suporte para plugins;
- Menor capacidade de personalização;
- Poucas opções de otimização;
- Desempenho inferior comparado a softwares modernos.

Bukkit

O Bukkit foi um dos primeiros grandes projetos focados em plugins para Minecraft.

Ele revolucionou os servidores durante muitos anos e serviu como base para diversos softwares atuais.

Hoje em dia, praticamente ninguém utiliza Bukkit puro em projetos modernos.

Vantagens

- Histórico extremamente importante;
- Grande compatibilidade com plugins antigos;
- Estrutura leve.

Desvantagens

- Projeto ultrapassado;

- Poucas otimizações modernas;
- Menor suporte atualmente.

Spigot

O Spigot surgiu como uma evolução do Bukkit.

Ele trouxe:

- melhorias de desempenho;
- otimizações;
- gerenciamento melhor de entidades;
- e maior estabilidade.

Durante muitos anos foi o software mais utilizado no mundo.

Vantagens

- Muito estável;
- Excelente compatibilidade com plugins;
- Bom desempenho;
- Grande comunidade.

Desvantagens

- Menos otimizações comparado ao Paper;
- Algumas limitações modernas.

Mesmo hoje, o Spigot ainda é muito utilizado.

Paper

O Paper atualmente é um dos softwares mais populares e recomendados.

Ele é baseado no Spigot, porém adiciona:

- otimizações avançadas;
- melhorias de desempenho;
- correções;
- recursos extras;
- APIs modernas.

Hoje, grande parte dos servidores profissionais utiliza Paper.

Vantagens

- Excelente desempenho;
- Melhor gerenciamento de entidades;
- Menor consumo de recursos;
- Compatibilidade enorme com plugins;
- Atualizações constantes;
- Grande estabilidade.

Desvantagens

- Alguns plugins extremamente antigos podem apresentar incompatibilidade;
- Algumas mecânicas Vanilla podem sofrer pequenas alterações por otimização.

Purpur

O Purpur é baseado no Paper, porém adiciona ainda mais personalização.

Ele é muito popular em servidores que desejam:

- maior controle;
- configurações extras;
- modificações mais avançadas;
- recursos adicionais.

Vantagens

- Altíssima personalização;
- Excelente desempenho;
- Grande quantidade de opções de configuração;
- Base extremamente otimizada.

Desvantagens

- Quantidade maior de configurações pode confundir iniciantes;
- Alterações excessivas podem causar desequilíbrio se mal configuradas.

Então, qual software escolher?

A resposta depende do objetivo do servidor.

Quando usar Vanilla?

Use apenas se:

- quiser uma experiência totalmente pura;
- não precisar de plugins;
- estiver criando um servidor extremamente simples.

Na maioria dos casos modernos, não é a melhor escolha.

Quando usar Spigot?

O Spigot ainda é uma excelente opção para:

- servidores mais simples;
- projetos leves;
- quem deseja estabilidade tradicional.

Porém hoje, muitos administradores preferem diretamente o Paper.

Quando usar Paper?

Atualmente, para a maioria dos projetos, o Paper costuma ser a melhor escolha.

Principalmente para:

- Survival;
- RankUP;
- SkyBlock;
- servidores modernos;
- servidores profissionais.

Isso porque ele oferece:

- excelente desempenho;
- ótima compatibilidade;
- grande estabilidade;
- e diversas otimizações.

Quando usar Purpur?

O Purpur é excelente para:

- servidores mais personalizados;
- projetos avançados;
- administradores que desejam maior controle.

Porém, para iniciantes, recomendo primeiro aprender Paper antes de avançar para configurações mais complexas.

Um erro muito comum

Muitas pessoas instalam:

- plugins pesados;
- dezenas de sistemas;
- modificações aleatórias;

E depois culpam o software pelos problemas de desempenho.

Na prática, o desempenho do servidor depende de:

- organização;
- plugins;
- otimização;
- hardware;
- quantidade de entidades;
- gerenciamento do mapa;
- e configuração correta.

O software ajuda muito, mas sozinho não faz milagres.

Minha recomendação

Se você está começando agora, normalmente recomendo:

Para iniciantes

- Paper.

Para projetos mais avançados

- Purpur.

Para servidores extremamente específicos

- avaliar outras soluções conforme a necessidade.

O importante é:

- manter estabilidade;
- organização;
- e compatibilidade.

Conclusão

Agora já entendemos:

- o que é o software do servidor;
- diferenças entre Vanilla, Bukkit, Spigot, Paper e Purpur;
- vantagens e desvantagens de cada um;
- e como escolher a melhor opção para nosso projeto.

No próximo tópico iremos preparar corretamente nosso ambiente de trabalho, aprendendo:

- como instalar o Java;
- quais ferramentas utilizar;
- e quais programas são recomendados para administração e desenvolvimento do servidor.

Item 4 — Instalando o Java e Preparando o Ambiente

Agora que já entendemos:

- os tipos de servidores;
- as versões do Minecraft;
- e os principais softwares utilizados;

Chegou o momento de preparar corretamente nosso ambiente de trabalho.

Antes de iniciar qualquer servidor Minecraft, precisamos instalar algumas ferramentas essenciais para garantir:

- estabilidade;
- compatibilidade;
- organização;
- e facilidade durante o desenvolvimento e administração.

Muitas pessoas simplesmente instalam arquivos aleatórios da internet sem entender o que estão fazendo. O resultado normalmente são:

- erros de inicialização;
- incompatibilidade;
- problemas de desempenho;
- plugins quebrando;
- e dificuldade para resolver problemas futuramente.

Por isso, nesta etapa iremos preparar tudo corretamente desde o início.

O que é o Java?

O Minecraft Java Edition funciona utilizando a linguagem Java.

Isso significa que o servidor precisa obrigatoriamente do Java instalado para funcionar.

Sem o Java:

- o servidor não inicia;
- plugins não funcionam;
- e o sistema não consegue executar os arquivos `.jar`.

O que é um arquivo `.jar`?

Grande parte do ecossistema do Minecraft utiliza arquivos `.jar`.

Esses arquivos normalmente representam:

- servidores;
- plugins;
- ferramentas;
- aplicações Java.

Basicamente, o `.jar` é um pacote executável utilizado pelo Java.

Quando iniciamos um servidor Minecraft, normalmente estamos executando um arquivo `.jar`.

Escolhendo a versão correta do Java

Uma das maiores dúvidas de iniciantes é:
“qual versão do Java devo instalar?”

A resposta depende da versão do Minecraft utilizada.

Versões diferentes do Minecraft podem exigir versões diferentes do Java.

Exemplos

- Minecraft 1.8 → normalmente Java 8;
- Minecraft 1.12 → normalmente Java 8;
- Minecraft 1.18+ → Java 17;
- versões mais recentes → geralmente Java 17 ou superior.

Por isso, antes de instalar o Java, sempre verifique:

- compatibilidade do servidor;
- compatibilidade dos plugins;
- recomendação oficial do software utilizado.

Onde baixar o Java?

Sempre utilize fontes oficiais e confiáveis.

Evite:

- sites desconhecidos;
- instaladores modificados;
- versões suspeitas encontradas aleatoriamente na internet.

Atualmente, algumas das distribuições mais utilizadas são:

- Oracle Java;
- Temurin;
- OpenJDK.

Para a maioria dos usuários, recomendo utilizar:

- Temurin/OpenJDK;
- ou versões oficiais confiáveis.

Instalando o Java

A instalação normalmente é simples:

1. baixar o instalador;
2. executar;
3. concluir a instalação.

Porém, após instalar, é importante verificar se tudo foi configurado corretamente.

Verificando se o Java está funcionando

Após instalar o Java:

1. abra o terminal ou prompt de comando;
2. digite:

java -version

Se tudo estiver correto, o sistema exibirá:

- versão do Java;
- distribuição instalada;

- informações do ambiente Java.

Caso apareça erro, normalmente significa:

- Java não instalado corretamente;
- variáveis do sistema não configuradas;
- caminho do Java não reconhecido.

O que é terminal ou prompt de comando?

O terminal é uma ferramenta que permite executar comandos diretamente no sistema operacional.

Muitas pessoas têm medo do terminal no início, porém ele é extremamente importante para:

- administração;
- servidores;
- programação;
- automação;
- infraestrutura.

Grande parte da administração de servidores acontece através dele.

Por isso, é importante começar a se familiarizar desde cedo.

Escolhendo um bom editor de código

Mesmo que inicialmente não vamos criar plugins, um bom editor ajuda muito na:

- organização;
- leitura de arquivos;
- edição de configurações;
- visualização de logs;
- produtividade.

Visual Studio Code

O Visual Studio Code é atualmente uma das melhores opções para iniciantes e profissionais.

Vantagens

- Gratuito;
- Leve;
- Muito organizado;
- Grande quantidade de extensões;
- Excelente suporte para Java;
- Ótimo para editar arquivos de plugins e configurações.

IntelliJ IDEA

O IntelliJ IDEA é extremamente popular entre desenvolvedores Java.

Ele é mais avançado e muito utilizado para:

- desenvolvimento de plugins;
- aplicações Java;
- projetos profissionais.

Vantagens

- Ferramentas avançadas;
- Excelente suporte ao Java;
- Recursos profissionais;
- Muito utilizado no mercado.

Desvantagens

- Mais pesado;
- Pode assustar iniciantes no começo.

Organização de pastas

Uma boa organização evita muitos problemas futuramente.

Recomendo criar:

- uma pasta principal do servidor;
- pasta para plugins;
- backups;
- arquivos antigos;
- e materiais auxiliares.

Evite:

- deixar tudo espalhado;
- misturar servidores diferentes;
- salvar arquivos importantes na área de trabalho.

Organização é extremamente importante em qualquer projeto.

Aprendendo a ler logs

Um dos maiores erros de iniciantes é ignorar completamente os logs do servidor.

Os logs mostram:

- erros;
- plugins com problema;
- incompatibilidades;
- falhas;
- e informações importantes do servidor.

Aprender a ler logs desde cedo ajuda muito na resolução de problemas.

Um erro muito comum

Muitas pessoas:

- instalam dezenas de programas;
- plugins aleatórios;
- modificações desconhecidas;

Sem sequer entender o ambiente básico do servidor.

O ideal é:

1. preparar corretamente o ambiente;
2. entender as ferramentas;
3. aprender o básico;

4. e somente depois avançar para estruturas mais complexas.

Conclusão

Agora já temos nosso ambiente praticamente preparado para iniciar o servidor.

Aprendemos:

- o que é Java;
- como escolher a versão correta;
- como verificar a instalação;
- ferramentas recomendadas;
- organização básica;
- e conceitos importantes do ambiente de administração.

No próximo tópico iremos finalmente:

- iniciar nosso primeiro servidor;
- entender a estrutura inicial;
- aceitar o EULA;
- e conhecer os primeiros arquivos gerados automaticamente pelo sistema.

Item 5 — Criando e Iniciando o Primeiro Servidor

Agora que já:

- instalamos o Java;
- preparamos o ambiente;
- escolhemos a versão do Minecraft;
- e entendemos qual software utilizar;

Finalmente chegou o momento de criar nosso primeiro servidor.

Nesta etapa iremos:

- iniciar o servidor pela primeira vez;
- entender os arquivos gerados;
- aceitar o EULA;
- organizar nossa estrutura;
- e conhecer os primeiros elementos importantes do servidor.

Mesmo sendo uma etapa relativamente simples, ela é extremamente importante, pois muitos iniciantes acabam cometendo erros logo no primeiro contato com o servidor.

Por isso, iremos fazer tudo de maneira organizada e correta.

Criando a pasta do servidor

Antes de iniciar qualquer arquivo, o primeiro passo é criar uma pasta exclusiva para o servidor.

Exemplo:

- `ServidorMinecraft`
- `MeuServidor`
- `ServidorSurvival`

Evite:

- utilizar nomes confusos;
- espaços excessivos;
- caracteres especiais;
- misturar vários servidores na mesma pasta.

Organização desde o início evita muitos problemas futuramente.

Baixando o software do servidor

Agora precisamos baixar o software escolhido anteriormente.

Neste curso iremos utilizar como exemplo o:

- Paper.

O Paper atualmente é uma das melhores opções para:

- desempenho;
- estabilidade;
- compatibilidade;
- e otimização.

Sempre baixe o servidor através do site oficial do projeto.

Evite:

- links aleatórios;
- versões modificadas;
- downloads desconhecidos.

Colocando o arquivo na pasta

Após baixar o arquivo `.jar`, coloque-o dentro da pasta criada anteriormente.

Exemplo:

```
ServidorMinecraft/
└── paper.jar
```

Você pode renomear o arquivo para algo mais simples como:

- `server.jar`
- `paper.jar`

Isso ajuda bastante futuramente na inicialização do servidor.

Primeira inicialização do servidor

Agora iremos iniciar o servidor pela primeira vez.

Inicialmente, ainda não iremos utilizar o arquivo `.bat`.

Primeiro precisamos permitir que o servidor gere seus arquivos iniciais.

Abra o terminal ou prompt de comando dentro da pasta do servidor e execute:

java -jar paper.jar

Caso tenha utilizado outro nome no arquivo, substitua corretamente.

O que acontece na primeira inicialização?

Ao iniciar o servidor pela primeira vez:

- arquivos serão gerados;
- pastas serão criadas;
- configurações iniciais aparecerão;
- e o sistema irá preparar toda a estrutura do servidor.

Porém, logo após isso, o servidor irá fechar automaticamente.

E isso é completamente normal.

O que é o EULA?

O EULA significa:

- End User License Agreement.

Basicamente, é o contrato de uso da Mojang/Minecraft.

Antes de utilizar o servidor, precisamos aceitar esse contrato.

Quando iniciamos o servidor pela primeira vez, um arquivo chamado:

eula.txt

é criado automaticamente.

Aceitando o EULA

Abra o arquivo `eula.txt`.

Você verá algo parecido com:

eula=false

Altere para:

eula=true

Salve o arquivo.

Pronto. Agora o servidor poderá iniciar normalmente.

Arquivos gerados automaticamente

Após a primeira inicialização, diversas pastas e arquivos serão criados.

Isso pode assustar iniciantes no começo, porém é completamente normal.

Alguns arquivos importantes são:

---

server.properties

Esse é um dos arquivos mais importantes do servidor.

Nele podemos configurar:

- porta;
- quantidade máxima de jogadores;

- dificuldade;
- PvP;
- modo online;
- distância de renderização;
- e diversas outras opções.

Mais para frente iremos estudar esse arquivo com mais profundidade.

---

plugins/

A pasta `plugins` será utilizada para:

- instalar plugins;
- armazenar configurações;
- organizar sistemas adicionais do servidor.

Inicialmente ela pode estar vazia.

---

logs/

A pasta `logs` armazena:

- registros do servidor;
- erros;
- mensagens;
- inicializações;
- informações importantes.

Ela é extremamente útil para:

- identificar problemas;
- corrigir falhas;
- entender erros de plugins.

---

worlds/

Após iniciar o servidor corretamente, os mundos do Minecraft serão gerados automaticamente.

Normalmente:

- mundo principal;
- Nether;
- End.

Dependendo da versão e do software utilizado, os nomes podem variar.

Iniciando o servidor novamente

Após aceitar o EULA, execute novamente:

java -jar paper.jar

Agora o servidor deverá iniciar corretamente.

Durante a inicialização:

- o mundo será gerado;
- plugins internos serão carregados;
- configurações serão aplicadas.

A primeira inicialização normalmente demora um pouco mais.

Isso é completamente normal.

Como saber se o servidor iniciou corretamente?

Se tudo estiver funcionando corretamente, o terminal exibirá mensagens semelhantes a:

Done!

ou:

For help, type "help"

Isso significa que o servidor está online.

Entrando no servidor

Se o servidor estiver rodando na mesma máquina:

1. abra o Minecraft;
2. clique em Multiplayer;
3. Adicionar servidor;
4. utilize:

localhost

ou:

127.0.0.1

Pronto. Você estará conectado ao próprio servidor.

Comandos básicos do console

O console permite controlar o servidor diretamente pelo terminal.

Alguns comandos úteis:

Parar o servidor

stop

Nunca feche o servidor simplesmente apertando o X da janela.

Sempre utilize:

stop

Isso evita:

- corrupção de arquivos;
- perda de dados;
- problemas no mundo.

Listar jogadores online

list

Salvar o mundo

save-all

Um erro muito comum

Muitas pessoas:

- instalam dezenas de plugins imediatamente;
- alteram configurações sem entender;
- modificam arquivos aleatórios;
- copiam tutoriais sem aprender.

O ideal é:

1. iniciar o servidor corretamente;
2. entender sua estrutura;
3. aprender como tudo funciona;
4. e somente depois avançar para personalizações maiores.

Conclusão

Agora já conseguimos:

- criar o servidor;
- iniciar corretamente;
- aceitar o EULA;
- entender os principais arquivos;
- acessar o servidor;
- e utilizar comandos básicos do console.

Nos próximos tópicos iremos:

- criar o arquivo `.bat`;
- aprender a iniciar o servidor corretamente;
- entender memória RAM;
- e melhorar nossa estrutura de inicialização e organização.

CORREÇÃO 15/05/2026 - ADICIONADO “ARQUIVO START.BAT”, ANTES ESQUECIDO.

Arquivo start.bat

Até agora iniciamos o servidor manualmente utilizando comandos diretamente no terminal.

Porém, fazer isso toda vez pode ser:

- cansativo;
- desorganizado;
- pouco prático.

Por isso, normalmente utilizamos um arquivo `.bat` para automatizar a inicialização do servidor.

O `.bat` é basicamente um arquivo executável do Windows que permite executar comandos automaticamente.

Com ele, conseguimos:

- iniciar o servidor mais facilmente;
- definir memória RAM;
- organizar comandos;
- automatizar processos;
- e deixar nossa estrutura muito mais profissional.

Criando o arquivo start.bat

Dentro da pasta do servidor:

1. clique com o botão direito;
2. crie um novo arquivo de texto;
3. renomeie para:

start.bat

Muito importante:
o arquivo deve terminar em `.bat`.

Ele não pode ficar como:

start.bat.txt

Caso as extensões dos arquivos não estejam aparecendo:

- abra o explorador de arquivos;
- clique em “Exibir”;
- habilite “Extensões de nomes de arquivos”.

Estrutura básica do start.bat

Agora abra o arquivo `start.bat` e adicione:

java -Xms2G -Xmx4G -jar paper.jar nogui

pause

Explicando cada parte do comando

Agora vamos entender exatamente o que esse comando faz.

---

java

java

Esse comando chama o Java instalado no sistema.

Sem ele, o servidor não consegue iniciar.

-Xms

-Xms2G

Define a memória RAM inicial do servidor.

Neste exemplo:

- o servidor iniciará utilizando 2GB de RAM.

-Xmx

-Xmx4G

Define o limite máximo de memória RAM.

Neste exemplo:

- o servidor poderá utilizar até 4GB de RAM.

-jar

-jar paper.jar

Informa qual arquivo `.jar` será executado.

No nosso caso:

- `paper.jar`.

Caso o nome do arquivo seja diferente, altere corretamente.

Exemplo:

-jar server.jar

---

nogui

nogui

Desativa a interface gráfica do servidor.

Isso é extremamente recomendado, pois:

- reduz consumo de memória;
- melhora desempenho;
- deixa o servidor mais leve.

Hoje praticamente todos os servidores utilizam `nogui`.

---

pause

pause

Mantém a janela aberta caso aconteça algum erro.

Isso ajuda muito na identificação de problemas.

Sem o `pause`, a janela fecharia instantaneamente ao ocorrer um erro.

Como iniciar o servidor

Agora basta:

1. salvar o arquivo;

2. clicar duas vezes em `start.bat`.

Pronto.

O servidor iniciará automaticamente utilizando os parâmetros configurados.

Quanta memória RAM devo usar?

Essa é uma dúvida extremamente comum.

Mais memória RAM nem sempre significa:

- mais desempenho;
- mais estabilidade;
- melhor otimização.

O ideal depende:

- da quantidade de jogadores;
- plugins;
- entidades;
- tamanho do mapa;
- e versão utilizada.

Recomendação básica

Servidores simples

- 2GB a 4GB.

Servidores médios

- 6GB a 10GB.

Servidores maiores

- acima disso, dependendo da estrutura.

Um erro MUITO comum

Muitas pessoas utilizam:

-Xmx32G

Mesmo sem necessidade.

Isso pode:

- desperdiçar memória;
- piorar o garbage collector;
- causar instabilidade;
- aumentar consumo desnecessário.

O importante não é “colocar muita RAM”.

O importante é:

- organização;
- otimização;
- plugins bem configurados;
- e hardware equilibrado.

Organização recomendada

Uma boa prática é deixar:

- o `start.bat`;
- o `.jar`;
- e os arquivos principais;

sempre organizados na pasta principal do servidor.

Evite:

- múltiplos arquivos aleatórios;
- dezenas de `.jar`;
- backups misturados;
- versões duplicadas espalhadas.

Conclusão

Agora já aprendemos:

- o que é um arquivo `.bat`;
- como criar o `start.bat`;
- como iniciar o servidor corretamente;
- como configurar memória RAM;
- e como deixar nossa estrutura muito mais organizada e profissional.

A partir deste ponto, já temos uma base sólida para começar a:

- instalar plugins;
- configurar sistemas;
- e evoluir nosso servidor corretamente.

Item 6 — Instalando Plugins e Entendendo sua Estrutura

Agora que:

- nosso servidor já está funcionando;
- aprendemos a criar o `start.bat`;
- configuramos corretamente a inicialização;
- e organizamos nossa estrutura básica;

Chegou o momento de entrar em uma das partes mais importantes de qualquer servidor Minecraft:
os plugins.

Os plugins são responsáveis por adicionar:

- sistemas;
- comandos;
- permissões;
- economia;
- proteção;
- menus;
- ranks;
- otimizações;
- e praticamente qualquer funcionalidade extra dentro do servidor.

Grande parte da identidade de um servidor vem justamente dos plugins utilizados.

Porém, antes de sair instalando dezenas de plugins aleatórios, precisamos entender:

- como eles funcionam;
- como instalá-los corretamente;
- e como manter uma estrutura organizada.

O que é um plugin?

Um plugin é uma extensão criada para adicionar funcionalidades ao servidor.

Basicamente:

- o servidor fornece a base;
- e os plugins adicionam recursos extras.

Exemplos:

- sistema de permissões;
- economia;
- proteção de terrenos;
- menus personalizados;
- ranks;
- teleportes;
- NPCs;
- anti-cheat;
- e muito mais.

Onde os plugins ficam?

Todos os plugins devem ser colocados dentro da pasta:

plugins/

Exemplo:

ServidorMinecraft/

└── plugins/

  ├── EssentialsX.jar

  ├── LuckPerms.jar

  └── Vault.jar

Cada plugin normalmente é um arquivo `.jar`.

Como instalar um plugin

A instalação normalmente é extremamente simples:

1. baixar o plugin;
2. colocar o `.jar` na pasta `plugins`;
3. reiniciar o servidor.

Após isso, o plugin:

- será carregado;
- criará sua pasta;
- e gerará arquivos de configuração automaticamente.

Onde baixar plugins?

Sempre utilize fontes confiáveis.

Atualmente, os principais locais são:

- Hangar;
- Modrinth;
- SpigotMC;
- páginas oficiais dos desenvolvedores.

Evite:

- plugins piratas;
- versões modificadas;
- downloads suspeitos;
- arquivos encontrados aleatoriamente na internet.

Além de riscos de segurança, plugins maliciosos podem:

- roubar dados;
- destruir mundos;
- instalar backdoors;
- consumir recursos excessivos;
- comprometer totalmente o servidor.

O que acontece quando o plugin inicia?

Quando o servidor liga:

1. ele lê os arquivos da pasta `plugins`;
2. carrega cada plugin;
3. verifica compatibilidade;
4. inicializa os sistemas;
5. gera arquivos de configuração.

Por isso, quanto mais plugins:

- maior o consumo de recursos;
- maior a complexidade;
- e maior a necessidade de organização.

Pastas de configuração

Após iniciar o servidor, os plugins normalmente criam suas próprias pastas.

Exemplo:

plugins/

├── Essentials/

├── LuckPerms/

├── Vault/

└── WorldEdit/

Dentro dessas pastas ficam:

- configurações;
- bancos de dados;
- logs;
- arquivos internos;
- mensagens;
- permissões;
- sistemas personalizados.

Nunca edite arquivos sem backup

Antes de alterar configurações:

- faça backup;
- copie arquivos importantes;
- evite mudanças impulsivas.

Muitos iniciantes quebram o servidor simplesmente:

- apagando arquivos;

- removendo linhas;
- alterando configurações sem entender.

Plugins essenciais para praticamente qualquer servidor

Agora vamos conhecer alguns plugins extremamente populares.

EssentialsX

O EssentialsX adiciona diversos comandos úteis.

Exemplos:

- `/spawn`
- `/home`
- `/warp`
- `/tp`
- sistema de kits;
- mensagens;
- economia básica.

É um dos plugins mais utilizados do Minecraft.

LuckPerms

O LuckPerms é um sistema moderno de permissões.

Com ele conseguimos:

- criar grupos;
- administrar cargos;
- definir permissões;
- controlar comandos;
- organizar administração.

Hoje ele é praticamente padrão em servidores profissionais.

Vault

O Vault funciona como uma ponte entre plugins.

Ele normalmente é utilizado para:

- economia;
- permissões;
- integração entre sistemas.

Muitos plugins dependem dele.

WorldEdit

O WorldEdit permite:

- editar mapas rapidamente;
- copiar estruturas;
- criar regiões;
- modificar terrenos.

Extremamente útil para construção e administração.

WorldGuard

O WorldGuard é utilizado para:

- proteger áreas;
- controlar PvP;
- impedir destruição;
- administrar regiões.

Muito importante para servidores Survival e públicos.

Um erro MUITO comum

Muitas pessoas acreditam que:

> mais plugins = servidor melhor.

Na prática:

- plugins demais podem causar conflitos;
- aumentar consumo de memória;
- reduzir TPS;
- gerar instabilidade;
- dificultar administração.

Um bom servidor normalmente utiliza:

- apenas o necessário;
- sistemas organizados;
- plugins bem configurados.

Qualidade sempre vale mais que quantidade.

Como saber se um plugin é bom?

Antes de instalar um plugin, analise:

- atualizações recentes;
- compatibilidade;
- avaliações;
- documentação;
- suporte do desenvolvedor;
- impacto no desempenho.

Nunca instale plugins apenas porque:

> “alguém recomendou”.

Reiniciar ou usar reload?

Muitos iniciantes utilizam:

reload

para recarregar plugins.

Isso NÃO é recomendado.

O `reload` pode:

- quebrar plugins;
- gerar vazamento de memória;
- causar conflitos;
- criar erros invisíveis.

O ideal normalmente é:

- desligar corretamente;
- iniciar novamente o servidor.

Conclusão

Agora já entendemos:

- o que são plugins;
- como instalá-los;
- como funciona sua estrutura;
- quais são os plugins essenciais;
- e como manter o servidor organizado e estável.

Nos próximos tópicos iremos começar a:

- configurar plugins;
- criar permissões;
- organizar administração;
- e transformar nosso servidor em algo muito mais profissional.

Item 7 — Configurando Permissões e Organizando a Administração

Agora que já:

- aprendemos a instalar plugins;
- entendemos sua estrutura;
- e conhecemos alguns plugins essenciais;

Chegou o momento de configurar uma das partes mais importantes de qualquer servidor: as permissões.

Muitos iniciantes acreditam que permissões servem apenas para:

- administrador;
- moderador;
- ou dono do servidor.

Porém, na prática, sistemas de permissões controlam praticamente tudo dentro do servidor.

São eles que definem:

- quais comandos cada jogador pode utilizar;
- quais áreas podem acessar;
- quais sistemas podem utilizar;
- e como a administração será organizada.

Sem um sistema de permissões organizado, o servidor rapidamente se torna:

- bagunçado;
- inseguro;
- difícil de administrar;
- e vulnerável a erros.

O que são permissões?

Permissões são regras que controlam o acesso a funcionalidades dentro do servidor.

Exemplo:

- um jogador comum não pode utilizar `/gamemode`;
- um moderador pode utilizar comandos de moderação;
- um administrador possui acesso total.

Tudo isso funciona através de permissões.

O plugin LuckPerms

Neste curso iremos utilizar o:

- LuckPerms.

O LuckPerms atualmente é um dos melhores sistemas de permissões disponíveis para Minecraft.

Ele é:

- moderno;
- extremamente organizado;
- muito rápido;
- amplamente utilizado;
- e compatível com praticamente qualquer servidor profissional.

Instalando o LuckPerms

Caso ainda não tenha instalado:

1. baixe o plugin;
2. coloque o `.jar` na pasta `plugins`;
3. reinicie o servidor.

Após iniciar, a pasta do LuckPerms será criada automaticamente.

Entendendo grupos

O LuckPerms funciona principalmente através de grupos.

Exemplo:

- Jogador;
- VIP;
- Moderador;
- Admin;

- Dono.

Cada grupo pode possuir:

- permissões específicas;
- prefixos;
- comandos;
- restrições.

Criando um grupo

No console do servidor, podemos criar um grupo utilizando:

lp creategroup Moderador

Pronto.

Agora o grupo:

- Moderador;
  foi criado.

Adicionando permissões

Agora podemos adicionar permissões ao grupo.

Exemplo:

lp group Moderador permission set essentials.kick true

Isso permite que o grupo:

- Moderador;
  utilize o comando `/kick`.

Adicionando um jogador ao grupo

Agora podemos adicionar um jogador:

lp user NomeDoJogador parent add Moderador

Substitua:

- `NomeDoJogador`
  pelo nick correto.

Pronto.

Agora esse jogador pertence ao grupo Moderador.

O que significa “node”?

As permissões normalmente são chamadas de:

- permission nodes.

Exemplo:

essentials.fly

essentials.gamemode

worldedit.*

Essas “nodes” definem exatamente o que um jogador pode fazer.

O que significa o `*`?

O símbolo:

*

normalmente representa:

- todas as permissões.

Exemplo:

worldedit.*

significa:

- todas as permissões do WorldEdit.

Porém, utilize isso com cuidado.

Dar permissões excessivas pode:

- comprometer segurança;
- causar abuso;
- quebrar sistemas do servidor.

Organizando cargos corretamente

Um erro muito comum é criar:

- dezenas de cargos inúteis;
- permissões bagunçadas;
- grupos sem organização.

O ideal é manter:

- poucos grupos;
- funções claras;
- hierarquia organizada.

Exemplo simples:

Jogador

- comandos básicos.

VIP

- benefícios extras.

Moderador

- comandos de moderação.

Administrador

- gerenciamento completo.

Prefixos e aparência no chat

O LuckPerms pode trabalhar junto com plugins de chat para exibir:

- cargos;
- prefixos;
- cores;
- tags.

Exemplo:

[Admin] Gabriel

Isso ajuda muito na:

- organização;
- identificação;
- profissionalismo do servidor.

Nunca dê OP para todos

Um dos maiores erros de iniciantes é utilizar:

op NomeDoJogador

em excesso.

O OP concede praticamente acesso total ao servidor.

Isso pode:

- comprometer segurança;
- permitir abuso;
- quebrar plugins;
- causar problemas administrativos.

O ideal é utilizar:

- permissões organizadas;
- grupos;
- cargos bem definidos.

Segurança é extremamente importante

Muitos servidores pequenos acabam sofrendo:

- abuso de permissões;
- grief;
- destruição;
- roubo de arquivos;
- problemas administrativos.

Por isso:

- nunca entregue acesso sem necessidade;
- nunca compartilhe arquivos importantes;
- mantenha backups;
- e organize corretamente os cargos.

Organização administrativa

Mesmo servidores pequenos devem possuir:

- regras;
- hierarquia;
- organização;
- controle.

Isso evita:

- confusão;
- abuso;
- favoritismo;
- e problemas futuros.

Um erro MUITO comum

Muitas pessoas:

- instalam dezenas de plugins;
- criam sistemas enormes;
- mas esquecem completamente da administração.

Um servidor pode ter:

- plugins incríveis;
- mapas lindos;
- sistemas avançados;

Mas se a administração for desorganizada, o servidor dificilmente irá durar.

Conclusão

Agora já aprendemos:

- o que são permissões;
- como funciona o LuckPerms;
- como criar grupos;
- como adicionar permissões;
- como organizar cargos;
- e como estruturar corretamente a administração do servidor.

Nos próximos tópicos iremos começar a:

- otimizar desempenho;
- entender TPS;
- reduzir lag;
- e melhorar a estabilidade do servidor de forma profissional.

Item 8 — Noções Básicas de Otimização e Desempenho

Agora chegamos em uma das partes mais importantes de qualquer servidor Minecraft:
a otimização.

Não importa:

- quantos plugins você possui;
- qual mapa utiliza;
- ou quão bonito o servidor está.

Se o desempenho for ruim:

- jogadores irão reclamar;
- travamentos acontecerão;
- comandos ficarão lentos;
- entidades irão bugar;
- e a experiência do servidor será prejudicada.

Por isso, entender otimização desde o início é extremamente importante.

E aqui existe algo muito importante:
otimização não significa apenas:

> “colocar mais memória RAM”.

Na prática, desempenho depende de:

- organização;
- hardware;
- plugins;
- configuração;
- quantidade de entidades;
- geração de mapa;
- e administração correta.

O que é TPS?

TPS significa:

Ticks Per Second

O Minecraft funciona através de “ticks”.

O servidor tenta processar:

- 20 ticks por segundo.

Isso significa que:

- o TPS ideal é 20.

Quando o TPS começa a cair:

- o servidor começa a apresentar lag.

Como o lag aparece?

Quando o desempenho piora, normalmente surgem problemas como:

- entidades travando;
- blocos demorando para quebrar;
- comandos lentos;
- mobs congelando;
- teleportes falhando;
- atraso em sistemas.

Muitas vezes os jogadores chamam isso apenas de:

“lag”.

O que mais causa lag em servidores?

Os principais problemas normalmente são:

- plugins mal otimizados;
- excesso de entidades;
- farms exageradas;
- mapas muito carregados;
- renderização excessiva;
- geração de chunks;
- máquinas fracas;
- configurações ruins.

Mais plugins nem sempre é melhor

Um erro extremamente comum:

- instalar dezenas de plugins desnecessários.

Cada plugin:

- consome memória;
- utiliza processamento;
- adiciona eventos;
- executa tarefas constantemente.

Quanto mais plugins:

- maior a complexidade;
- maior o consumo;
- maior a chance de problemas.

Um bom servidor normalmente possui:

- poucos plugins;
- plugins úteis;
- plugins bem configurados.

Cuidado com plugins antigos

Plugins muito antigos podem:

- causar vazamentos de memória;
- reduzir TPS;
- criar incompatibilidades;
- consumir CPU excessivamente.

Sempre verifique:

- atualizações;
- compatibilidade;
- avaliações;
- suporte do desenvolvedor.

Distância de renderização

Uma configuração extremamente importante é a distância de renderização.

Ela define:

- quantos chunks serão carregados ao redor dos jogadores.

Quanto maior:

- maior o consumo de processamento;
- maior o uso de memória;
- maior a carga no servidor.

Em servidores públicos, normalmente valores moderados funcionam melhor.

Entidades consomem MUITO desempenho

Animais, mobs, itens no chão e sistemas automáticos podem consumir uma quantidade enorme de processamento.

Exemplos:

- farms gigantes;
- centenas de villagers;
- itens acumulados;
- mobs excessivos;
- redstone exagerada.

Tudo isso impacta diretamente no TPS.

Hardware importa, mas não faz milagres

Ter um hardware forte ajuda muito.

Porém:

- um servidor mal configurado continuará problemático;
- plugins ruins continuarão ruins;
- má administração continuará causando lag.

O ideal é equilíbrio.

O processador é extremamente importante

Minecraft depende muito de desempenho por núcleo.

Isso significa que:

- processadores com bom single-core normalmente performam melhor.

Muitas pessoas olham apenas:

- quantidade de núcleos;
- memória RAM;

Mas ignoram:

- clock;
- desempenho por núcleo;
- arquitetura.

Memória RAM

A memória RAM ajuda:

- no carregamento;
- armazenamento temporário;
- gerenciamento do servidor.

Porém:

> colocar RAM excessiva não resolve tudo.

Inclusive, em alguns casos:

- RAM exagerada pode piorar o garbage collector.

O importante é:

- quantidade equilibrada;
- configuração correta;
- otimização geral.

Evite usar “reload”

Como já vimos anteriormente:

- o comando `reload`
  não é recomendado.

Ele pode:

- quebrar plugins;
- gerar vazamentos;
- criar erros invisíveis;
- causar instabilidade.

Sempre prefira:

- desligar corretamente;
- iniciar novamente o servidor.

Faça backups frequentemente

Um servidor sem backup é um risco enorme.

Problemas acontecem:

- corrupção de mundo;
- plugins quebrando;
- erros humanos;
- falhas de hardware;
- atualizações problemáticas.

Por isso:

- mantenha backups;
- organize versões;
- salve arquivos importantes regularmente.

Aprenda a analisar logs

Os logs são uma das ferramentas mais importantes para administração.

Eles ajudam a identificar:

- plugins problemáticos;
- erros;
- conflitos;
- falhas;
- e gargalos de desempenho.

Administradores experientes passam muito tempo:

- lendo logs;
- analisando erros;
- entendendo comportamento do servidor.

Não tente criar um servidor gigante imediatamente

Esse é provavelmente um dos maiores erros de iniciantes.

Muitas pessoas querem começar criando:

- network;
- múltiplos modos;
- dezenas de sistemas;
- estruturas enormes.

O ideal é:

1. aprender a base;
2. criar algo estável;
3. entender administração;
4. aprender otimização;
5. evoluir gradualmente.

Servidores grandes normalmente começaram pequenos.

Organização vale mais que exagero

Um servidor organizado normalmente performa melhor que:

- servidores gigantes;
- cheios de plugins;
- mal configurados.

Muitas vezes:

- simplicidade;
- estabilidade;
- organização;
- e boa administração;

valem mais do que dezenas de sistemas exagerados.

Conclusão

Agora já aprendemos:

- o que é TPS;
- como funciona o desempenho do Minecraft;
- causas comuns de lag;
- otimização básica;
- impacto de plugins;
- importância do hardware;
- organização;
- e boas práticas administrativas.

Com isso, já temos uma base extremamente sólida para:

- criar;
- administrar;
- otimizar;
- e evoluir servidores Minecraft de forma correta e profissional.

Este foi apenas o começo.

A partir daqui, você já possui conhecimento suficiente para:

- explorar novas possibilidades;
- instalar novos sistemas;
- aprender desenvolvimento;
- estudar infraestrutura;
- e evoluir cada vez mais dentro do universo de servidores Minecraft.

Aviso Importante

Todo o conteúdo apresentado neste material possui como objetivo criar uma base sólida e teórica sobre a estrutura de servidores Minecraft.

Aqui aprendemos:

- conceitos;
- organização;
- funcionamento dos sistemas;
- estrutura básica;
- boas práticas;
- e fundamentos extremamente importantes para qualquer administrador de servidor.

Porém, a parte prática também é essencial.

Por isso, os vídeos disponíveis no canal complementam este material didático de forma completa, mostrando:

- instalações passo a passo;
- criação do servidor em tempo real;
- configuração de plugins;
- organização da estrutura;
- permissões;
- otimização;
- resolução de erros;
- e diversas outras etapas importantes durante o desenvolvimento do servidor.

Durante os vídeos, iremos literalmente construir o servidor juntos, explicando cada etapa de maneira prática e organizada.

A recomendação ideal é:

- utilizar este material como apoio teórico;
- e acompanhar os vídeos para visualizar toda a aplicação prática acontecendo em tempo real.

Tecnologia é algo que se aprende muito na prática.

Então não tenha medo de:

- testar;
- errar;
- modificar;
- explorar novas possibilidades;
- e evoluir constantemente.

Com dedicação e prática, você ficará surpreso com o quanto consegue evoluir em pouco tempo.

Agradecimentos

Antes de finalizar este material, quero deixar meu agradecimento sincero a você que dedicou parte do seu tempo para estudar e aprender comigo.

Hoje em dia existem milhares de conteúdos rápidos e superficiais espalhados pela internet, então saber que existem pessoas interessadas em realmente aprender, entender e evoluir dentro da tecnologia é algo que me motiva muito a continuar criando conteúdo.

Ensinar sempre foi algo que gostei de fazer, e poder compartilhar conhecimento com tantas pessoas é algo extremamente gratificante para mim.

Espero de verdade que este material tenha conseguido te ajudar:

- a entender melhor os servidores Minecraft;
- a perder o medo de começar;
- e principalmente a enxergar que tecnologia não é algo impossível quando aprendemos da forma correta.

E isso é apenas o começo.

Ainda teremos muitos conteúdos, vídeos, projetos, configurações, sistemas e aprendizados pela frente, e pretendo continuar ensinando e compartilhando conhecimento por muito tempo.

Se este material te ajudou de alguma forma, uma das melhores maneiras de apoiar meu trabalho é:

- se inscrevendo no canal;
- curtindo os vídeos;
- compartilhando com amigos;
- e enviando para pessoas que também possuem interesse em tecnologia e servidores Minecraft.

Isso ajuda não apenas o canal a crescer, mas também permite que mais pessoas tenham acesso a conteúdos gratuitos e de qualidade.

Muito obrigado pela sua confiança, pelo seu tempo e por fazer parte dessa jornada.

Nos vemos nos próximos conteúdos.

LOGS:

- CRIAÇÃO: 02/05/2026
- ÚLTIMA ATUALIZAÇÃO: 15/05/2026