Guía de Uso de Comandos – Bot Señor Fino

Esta guía reúne de forma estructurada y detallada todas las funciones y comandos disponibles en Señor Fino, desde dinámicas interactivas y juegos hasta herramientas avanzadas de administración, registro y auditoría. El objetivo es proporcionar un recurso completo que facilite la correcta implementación y el uso eficiente del bot en entornos que exigen fiabilidad, organización y alto rendimiento.

Índice

Comandos Administrativos

Estos comandos están diseñados para la gestión, moderación y configuración del servidor. Su implementación requiere conocimientos técnicos de Discord y en la administración de roles y permisos. A continuación se documenta cada comando en detalle.

Comando: ?suge <#canal|id>

Funcionalidad:
Configura el canal de sugerencias del servidor, donde los miembros podrán enviar ideas, propuestas o comentarios de manera organizada y con seguimiento.

Funcionamiento:

- Asignación de canal:

Al usar el comando, debes indicar el canal donde quieres que se publiquen las sugerencias, ya sea mencionándolo (#canal) o usando su ID.
Este canal será exclusivo para recibir sugerencias y mantenerlas centralizadas.

- Recepción y publicación automática:

Cada vez que un usuario envía una sugerencia con ?suse, el bot crea automáticamente un mensaje en el canal configurado.

Este mensaje incluye:

- Detalles del usuario que envió la sugerencia.

- Contenido de la sugerencia.

- Un ID único para cada sugerencia, lo que permite un registro ordenado y fácil de consultar.

Se añaden dos botones: uno para aceptar la sugerencia y otro para rechazarla.

- Moderación y debate:

Se recomienda prohibir que otros usuarios envíen mensajes directamente en el canal para mantener el orden.

Se puede permitir la creación de hilos, donde los miembros pueden debatir, preguntar dudas o expresar su opinión sobre cada sugerencia.

- Registro y auditoría:
Todas las sugerencias quedan registradas con su ID único y se almacenan correctamente para futuras referencias, estadísticas o auditoría interna.

Esto asegura que ninguna sugerencia se pierda y facilita la gestión por parte del equipo de administración.

Comando: ?globalban

Funcionalidad:
Activa el sistema de baneos globales gestionado por **Señor Fino**, aplicando sanciones de manera automática en servidores adheridos.

Funcionamiento:

- Gestión centralizada: El bot consulta su base de datos global para identificar usuarios previamente sancionados.

- Bloqueo automático: Si un usuario registrado en la lista global entra al servidor, será baneado de inmediato.

- Sincronización: Los baneos ejecutados en un servidor adherido se propagan al resto.

- Auditoría: Cada acción queda reflejada en los registros del bot.

- Estado por defecto y recomendación:

> Este sistema está **habilitado por defecto**, y se recomienda mantenerlo así en todos los servidores.
>
> Si alguien aparece baneado por Señor Fino, significa claramente que es una persona absolutamente horrible, un verdadero terrorista del respeto y la convivencia, y que no merece ningún tipo de consideración ni respeto. Mantenerlo activo asegura que este tipo de individuos no puedan causar problemas en ningún servidor adherido.

Comando: ?saludos

Funcionalidad:
Activa o desactiva el sistema de respuestas automáticas a saludos dentro del chat.

Funcionamiento:

- Detección de palabras clave: El bot analiza mensajes en busca de saludos comunes.

- Respuesta inmediata: Genera un mensaje automático si la función está habilitada.

- Toggle sencillo: Se alterna entre activado y desactivado al usar el comando.

- Estado por defecto: Está **activado por defecto**, y es completamente normal si deseas desactivarlo en algún momento según las necesidades de tu servidor.

Comando: ?autar

Funcionalidad:
Controla la activación o desactivación del sistema anti-flood del servidor.

Funcionamiento:

- Supervisión constante: Mide la frecuencia de mensajes por usuario.

- Acciones preventivas: Bloquea o limita a quienes excedan los parámetros de flood.

- Gestión rápida: Se activa o desactiva con un solo uso del comando.

- Estado por defecto y recomendación: Este sistema está activado por defecto, y se recomienda configurarlo entre las primeras cosas al añadir el bot o desactivarlo si lo deseas, ya que por defecto puede generar pings a @everyone.

Comando: ?rolid

Funcionalidad:
Configura el rol o mención que será notificado en caso de detección de flood.

Funcionamiento:

- Compatibilidad: Admite @everyone, IDs de rol o menciones específicas.

- Notificación automática: El rol asignado recibe alertas en tiempo real.

- Verificación: El bot comprueba la validez y permisos del rol antes de guardarlo.

Comando: ?banaut

Funcionalidad:
Permite activar o desactivar el baneo automático de usuarios por flood.

Funcionamiento:

- Ejecución directa: Superado el límite, el bot aplica el baneo inmediatamente.

- Personalizable: Adaptable a las políticas de tolerancia de cada servidor.

- Registro: Todos los baneos se guardan con detalles (usuario, fecha, motivo).

Comando: ?botdet

Funcionalidad:
Activa o desactiva la detección de mensajes de bots en el sistema anti-flood.

Funcionamiento:

- Filtro avanzado: Decide si los mensajes de bots cuentan para flood.

- Protección extra: Evita que bots defectuosos saturen el servidor.

Cambio rápido: Alterna entre habilitar o ignorar con un solo comando.

Comando: ?cdab [<n>]

Funcionalidad:
Configura el sistema anti-raid, definiendo el umbral de detección de acciones masivas.

Funcionamiento:

- Uso sin argumento: Alterna entre activado (s) y desactivado (n).

- Uso con número (<n>): Fija el máximo de acciones en un intervalo breve antes de activar medidas de seguridad.

- Respuesta inmediata: Al superar el umbral, ejecuta bloqueos o restricciones automáticas.

- Integración: Se coordina con los sistemas anti-flood y de baneos.

- **Estado por defecto y umbral inicial:** El sistema está **desactivado por defecto**.

El **umbral inicial** es de **3 acciones**, asegurando que el sistema detecte rápidamente comportamientos masivos sospechosos sin afectar el uso normal del servidor.

Comando: ?bie <#canal|id>

Funcionalidad:
Configura el canal donde se enviarán los mensajes automáticos de bienvenida.

Funcionamiento:

- Asignación directa: Se debe indicar el canal por mención o ID.

- Mensaje automático: Cada vez que entra un usuario, el bot envía un saludo.

- Integración: Compatible con imágenes y texto personalizados.

- Control flexible: Puede cambiarse en cualquier momento.

Comando: ?des <#canal|id>

Funcionalidad:
Configura el canal donde se enviarán los mensajes automáticos de despedida.

Funcionamiento:

- Asignación directa: Se establece el canal mediante mención o ID.

- Mensaje automático: Al salir un usuario, se genera un aviso de despedida.

- Control flexible: Puede cambiarse en cualquier momento.

Comando: ?bieimg setbg <url|none>

Funcionalidad:
Establece la imagen de fondo de la tarjeta de bienvenida.

Funcionamiento:

- URL pública: Permite configurar cualquier imagen online.

- Opción none: Elimina la imagen personalizada y restaura la de defecto.

- Validación: El bot comprueba que la URL sea accesible.

Comando: ?bieimg settext <texto>

Funcionalidad:
Define el texto principal que aparece en la tarjeta de bienvenida.

Funcionamiento:

- Texto dinámico: Soporta variables como {user}, {server} y {membercount}.

- Personalización visual: El texto se muestra con gran tamaño en la imagen.

Comando: ?bieimg setsub <texto>

Funcionalidad:
Configura el subtítulo o texto secundario en la tarjeta de bienvenida.

Funcionamiento:

- Mensajes opcionales: Ideal para frases motivadoras o datos extra.

- Variables soportadas: {membercount} para reflejar el número actual de miembros.

Comando: ?bieimg setmsg <texto>

Funcionalidad:
Personaliza el mensaje que acompaña a la imagen de bienvenida en el chat.

Funcionamiento:

- Texto enriquecido: Soporta {user}, {server}, {membercount}.

- Mención directa: Con {user}, el bot menciona al nuevo miembro.

- Complemento visual: Se envía junto con la imagen o por separado si está desactivada.

Comando: ?bieimg setcolor #RRGGBB

Funcionalidad:
Cambia el color del texto en la tarjeta de bienvenida.

Funcionamiento:

- Formato hexadecimal: Acepta colores en formato #RRGGBB.

- Ejemplo: ?bieimg setcolor #FFD700 aplica dorado.

- Inmediato: Los cambios son visibles en la siguiente previsualización o entrada de usuario.

Comando: ?bieimg enable

Funcionalidad:
Habilita el envío de imágenes personalizadas en el sistema de bienvenida.

Funcionamiento:

- Activación inmediata: Sustituye los mensajes simples por tarjetas gráficas.

- Compatibilidad: Funciona junto al mensaje de texto configurado.

Comando: ?bieimg disable

Funcionalidad:
Desactiva la imagen de bienvenida, enviando únicamente texto plano.

Funcionamiento:

- Cambio rápido: Se alterna entre modo visual y simple.

- Uso recomendado: Para servidores que prefieren ligereza.

Comando: ?bieimg reset

Funcionalidad:
Restaura los valores predeterminados de la plantilla de bienvenida.

Funcionamiento:

- Reinicio completo: Borra personalizaciones de texto, imagen y color.

- Plantilla estándar: Vuelve al diseño base de Señor Fino.

Comando: ?bieimg preview

Funcionalidad:
Genera una previsualización de la tarjeta de bienvenida con la configuración actual.

Funcionamiento:

- Simulación: Usa al propio autor del comando como usuario de prueba.

- Comprobación visual: Permite revisar los cambios antes de aplicarlos definitivamente.

Variables disponibles en los comandos ?bieimg

{user} → Nombre del usuario que entra.

{server} → Nombre del servidor.

{membercount} → Número total de miembros actuales.

Comando: `?contarbaneos <cantidad><horas/dias>`

Funcionalidad:

Permite visualizar la lista de usuarios baneados durante un período de tiempo específico.

Funcionamiento:

- Consulta de registros: Se extraen los datos de la base de datos de baneos del servidor, filtrándolos según el intervalo temporal solicitado.

- Formato de salida: Los resultados se estructuran en un formato tabular o en un listado detallado, mostrando información como el usuario baneado, fecha y motivo del baneo.

- Ejemplos de uso:

?contarbaneos 24h → Muestra los baneos de las últimas 24 horas.

?contarbaneos 2d → Muestra los baneos de los últimos 2 días.

Comando: `?edad <cantidad> <dias/meses>`

Funcionalidad:

Lista las cuentas creadas en el servidor dentro de un rango de tiempo determinado.

Funcionamiento:

- Filtrado temporal: Se realiza una consulta sobre la base de datos de usuarios, filtrando por fecha de creación y comparando con el rango especificado.

- Visualización detallada: Se muestran datos como la fecha de creación, el ID del usuario y otros metadatos relevantes para la auditoría interna.

- Ejemplo de uso:

?edad 3 meses → Muestra las cuentas creadas en los últimos 3 meses.

?edad 10 dias → Muestra las cuentas creadas en los últimos 10 días.

Comando: `?caup <#canal>`

Funcionalidad:

Configura el canal de destino para recibir actualizaciones en tiempo real sobre acciones del servidor y datos financieros (criptomonedas, forex, ETFs e índices).

Funcionamiento:

- Selección de canal: El comando interpreta la mención del canal y verifica la validez de la misma.

- Subscripción a eventos: Se establece una suscripción interna del bot para redirigir eventos y actualizaciones en vivo al canal configurado.

- Manejo de errores: Se incluyen validaciones para asegurar que el canal esté activo y con permisos adecuados.

Comando: `?autorol`

Funcionalidad:

Genera autoroles de forma automática para asignar roles a los usuarios al ingresar al servidor.

Funcionamiento:

- Selección de roles:

Al usar el comando, se despliega un **menú interactivo** que permite seleccionar los roles que deseas asignar automáticamente.

Restricciones: No se pueden seleccionar roles que pertenezcan al autor del comando ni roles con jerarquía superior al tuyo o del bot, para evitar conflictos de permisos.

- Configuración de emojis:

Una vez seleccionados los roles y al darle a **Aceptar**, el bot solicitará ingresar los **emojis asociados a cada rol**, separados por comas.

Esto permite que los usuarios puedan elegir su rol dandole a los emojis correspondientes.

- Mensaje de selección:

Después de ingresar los emojis, el bot pedirá escribir un **mensaje que se mostrará en el mensaje de autorol**, explicando a los usuarios cómo elegir sus roles.

- Implementación final:

Al finalizar la configuración, el bot **borra los mensajes usados durante el proceso** para mantener el canal limpio.

Se coloca un **mensaje con el botón de autorol**, listo para que los usuarios seleccionen su rol mediante las reacciones de emojis configuradas.

Automatización y logs:

El bot asigna automáticamente los roles a los usuarios que entren al servidor según la configuración.

Cada asignación queda registrada para auditoría y futuras revisiones, asegurando un control completo del sistema de autoroles.

Comando: `?ban (@usuario) (razón)`

Funcionalidad:

Permite banear a un usuario del servidor, incorporando en el registro la identidad del administrador que ejecuta la acción y la razón del baneo.

Funcionamiento:

- Validación de permisos: Solo los usuarios con privilegios de administrador pueden ejecutar este comando.

- Extracción de parámetros: Se realiza la extracción del usuario mencionado y se valida la cadena de texto que representa la razón.

- Registro en base de datos: La acción se registra en un log de moderación, lo que permite rastrear la cadena de eventos para auditorías posteriores.

- Notificación y confirmación: Se envía una confirmación en el canal.

Comando: `?avisar (@usuario) (razón)`

Funcionalidad:

Emite un aviso al usuario mencionado, añadiéndolo a su historial de advertencias.

Funcionamiento:

- Registro incremental: Cada aviso se añade a un registro acumulativo en la base de datos del bot.

- Parámetros adicionales: Se incluye la razón del aviso, lo que permite posteriormente correlacionar la frecuencia y gravedad de las infracciones.

- Interfaz de consulta: Los avisos pueden consultarse posteriormente mediante el comando `?avisos`.

Comando: `?avisos (@usuario)`

Funcionalidad:

Muestra el historial de avisos acumulados para un usuario en particular.

Funcionamiento:

- Consulta a la base de datos: Se recupera el historial de avisos del usuario mencionado.

- Presentación estructurada: La salida incluye detalles como fecha, razón y número de aviso, permitiendo un análisis detallado.

Comando: `?quitaraviso (@usuario) (razón exacta)`

Funcionalidad:

Elimina un aviso específico del historial de un usuario, identificándolo mediante la razón exacta proporcionada.

Funcionamiento:

- Coincidencia exacta: Se realiza una búsqueda en el registro de avisos para identificar la entrada que coincide exactamente con la razón suministrada.

- Actualización del registro: Se elimina el aviso y se actualiza el historial del usuario, emitiendo una confirmación de la acción realizada.

Comando: `?subirnivel (@usuario) [cantidad]`

Funcionalidad:

Incrementa manualmente el nivel de un usuario, sumando la cantidad especificada a su experiencia acumulada.

Funcionamiento:

- Cálculo de XP: Se recalcula la experiencia del usuario en función de la cantidad indicada.

- Actualización del ranking: Se actualiza el ranking global del servidor, reflejando el cambio en la posición del usuario.

- Registro de cambios: Se documenta la modificación para futuras auditorías y análisis de progresión.

Comando: `?bajarnivel (@usuario) [cantidad]`

Funcionalidad:

Reduce el nivel de un usuario restando la cantidad indicada de su experiencia acumulada.

Funcionamiento:

- Ajuste del perfil: El bot descuenta la cantidad especificada, recalculando el nivel y la experiencia del usuario.

- Validación de límites: Se comprueba que la operación no reduzca el nivel por debajo del mínimo permitido.

- Actualización en tiempo real: El cambio se refleja inmediatamente en el ranking y se notifica a los administradores.

Comando: `?cniveles #canal`

Funcionalidad:

Establece un canal específico para anunciar los ascensos de nivel de los usuarios.

Funcionamiento:

- Automatización de notificaciones: Cada vez que un usuario sube de nivel, el bot envía un mensaje al canal configurado, utilizando formatos predefinidos para la notificación.

- Compatibilidad con paginación: Para servidores muy activos, se puede configurar la paginación de notificaciones para evitar saturación del canal.

Comando: `?silenciar (@usuario) [TIEMPO]`

Funcionalidad:

Silencia a un usuario durante un periodo de tiempo especificado, limitando su capacidad de interactuar en el servidor.

Funcionamiento:

- Asignación de rol temporal: Se aplica un rol de "silencio" al usuario, que revoca temporalmente los permisos para enviar mensajes.

- Temporizador interno: Se establece un temporizador que, al expirar, revierte automáticamente la acción.

- Manejo de excepciones: El comando incluye verificaciones para evitar conflictos con otros roles o permisos preexistentes.

Comando: `?juegos`

Funcionalidad:

Registra un canal dedicado donde el bot enviará notificaciones y actualizaciones sobre juegos gratuitos o eventos lúdicos.

Funcionamiento:

- Configuración del canal: Se especifica un canal mediante el comandol.

- Actualizaciones automáticas: El bot se conecta a fuentes de información de juegos y eventos, reenviando actualizaciones en tiempo real.

- Interfaz de usuario: La información se presenta en un formato enriquecido.

Comando: `?invinfo [código de invitación]`

Funcionalidad:

Muestra toda la información disponible sobre un código de invitación específico, incluyendo creador, fecha de creación y estadísticas de uso.

Funcionamiento:

- Análisis del código: El bot valida el formato del código y realiza una consulta.

- Extracción de metadatos: Se recogen datos relevantes y se presentan en un formato detallado.

- Integración con logs: Los datos obtenidos se pueden almacenar para análisis estadísticos y auditoría de invitaciones.

Comando: `?ruletaob (@usuario1) (@usuario2)`

Funcionalidad:

Inicia una partida de ruleta entre dos usuarios, donde ambos se enfrentan en un duelo basado en mecánicas similares al juego de ruleta rusa.

Funcionamiento:

- Configuración del duelo: Se reciben las menciones de dos usuarios y se inicializan las variables

de juego (por ejemplo, vidas, balas reales y de fogueo).

- Algoritmo de turnos: El sistema alterna los turnos, ejecutando comprobaciones aleatorias para determinar el impacto de cada disparo.

- Determinación del ganador: Se aplica la lógica del juego para reducir vidas y, en función del resultado, se declara el ganador.

Comando: `?salas`

Funcionalidad:

Crea diez (10) canales privados para la realización de entrevistas o interrogatorios a usuarios.

Funcionamiento:

- Generación dinámica: El bot crea automáticamente los canales siguiendo una nomenclatura predefinida.

- Control de acceso: Se asignan permisos específicos para que solo los usuarios autorizados puedan acceder a ellos.

Comando: `?revertglrol`

Funcionalidad:

Revierte la asignación del rol GL, restaurando la configuración anterior.

Funcionamiento:

- Registro de cambios: Se mantiene un historial de asignaciones de roles para poder realizar la reversión.
- Ejecución inmediata: El bot elimina el rol GL asignado y reestablece los roles previos, notificando la acción en el canal.

Comando: `?purga (entre 1 y 50)`

Funcionalidad:

Elimina una cantidad específica de mensajes entre 1 y 50 en el canal actual, facilitando la limpieza y moderación del chat.

Funcionamiento:

- Lógica de borrado: Se utiliza un algoritmo que identifica y elimina mensajes en bloque, basándose en parámetros de cantidad.

- Control de errores: Se implementan mecanismos para prevenir la eliminación accidental de mensajes críticos.

- Retroalimentación: Se envía un resumen de la operación realizada, indicando cuántos mensajes fueron eliminados.

Comando: `?borrar (ID o @)`

Funcionalidad:

Borra una cantidad específica de mensajes enviados por un usuario en un intervalo de tiempo determinado.

Funcionamiento:

- Filtrado avanzado: Se extraen los mensajes del usuario indicado dentro del rango temporal establecido.

- Eliminación selectiva: El comando procesa la eliminación en función de filtros adicionales, asegurando que no se borren mensajes fuera del criterio.

- Registro de la acción: Se documenta la operación para futuras auditorías.

Comando: `?tickets`

Funcionalidad:

Genera un botón interactivo en el canal para abrir un ticket de soporte o consulta.

Funcionamiento:

- Interfaz gráfica: Se despliega un botón con un identificador único que activa el proceso de generación de un ticket.

- Gestión de solicitudes: La acción se integra con el sistema interno de tickets, enviando alertas al equipo de soporte.

- Persistencia de datos: Cada ticket se registra en la base de datos para seguimiento y análisis posterior.

Comando: `?confgtick`

Funcionalidad:

Configura un canal específico donde se enviarán los logs y actualizaciones de los tickets cerrados.

Funcionamiento:

- Selección de canal: Se asigna y almacena el canal para la recepción de logs.

- Actualización en tiempo real: Los cambios en el estado de los tickets se comunican de forma inmediata al canal configurado.

- Validación de permisos: Solo administradores pueden configurar o modificar este parámetro.

Comando: ?adboton <nombre del botón> <color>

Funcionalidad:
Permite añadir un botón personalizado al mensaje de tickets, con el cual los usuarios podrán abrir tickets con el motivo indicado.

Funcionamiento:

- Parámetros:

<nombre del botón> → El texto que se mostrará en el botón y definirá el motivo del ticket.

<color> → El color del botón. Opciones disponibles: **Primary, Secondary, Success o Danger**.

- Límite: Se pueden configurar hasta **10 botones diferentes** en el mensaje de tickets.

Resultado: Al hacer clic en el botón, el bot creará un canal de ticket privado con el motivo correspondiente en el nombre.

Comando: ?reboton <nombre del botón>

Funcionalidad:
Elimina un botón previamente añadido al mensaje de tickets, impidiendo que se puedan generar más tickets con ese motivo.

Funcionamiento:

- Se debe indicar exactamente el nombre del botón que se quiere eliminar.

- Una vez eliminado, ya no estará disponible en el mensaje de tickets.

- Si aún quedan otros botones configurados, estos seguirán funcionando con normalidad.

Comando: `?autruleta`

Funcionalidad:

Activa o desactiva la funcionalidad de la ruleta en el servidor.

Funcionamiento:

- Alternancia de estado: Se invierte un flag interno que controla la disponibilidad del juego de ruleta.

- Notificación de estado: Se emite un mensaje informativo que indica si la ruleta se encuentra activada o desactivada.

- Integración con otros sistemas: El estado se guarda en la configuración global para mantener la coherencia.

Comando: `?glrol`

Funcionalidad:

Configura el rol de verificación que se aplicará a los usuarios para acceder a los canales restringidos.

Funcionamiento:

- Asignación de rol:

Se especifica un rol particular mediante su ID, el cual será asignado automáticamente a los usuarios una vez completado el proceso de verificación.

- Panel de configuración de canales:

Tras asignar el rol, el bot despliega un **panel interactivo** en el que puedes seleccionar qué canales privados **no quieres que el bot modifique**.

Esto sirve para proteger canales especiales y evitar que la configuración de permisos los afecte.

- Automatización de accesos:

Para el resto de canales restringidos, el bot ajustará los permisos automáticamente para que **solo los usuarios verificados puedan verlos**, manteniendo invisibles los canales al resto.

- Registro de cambios:

Cada modificación realizada por el bot queda documentada en el sistema de auditoría interna, facilitando el seguimiento de ajustes en los permisos.

Comando: `?verfmult`

Funcionalidad:

Activa la detección de multicuentas en el servidor, para identificar usuarios que puedan estar utilizando más de una cuenta.

Funcionamiento:

- Monitoreo de actividad: El sistema implementa algoritmos de detección que analizan patrones de comportamiento y datos de conexión.

- Integración con verificación: Requiere que el sistema de verificación esté activado, ya que utiliza los datos de validación de usuarios.

- Acciones preventivas: En caso de detección, se pueden activar medidas correctivas o notificar a los administradores.

Comando: `?verificar`

Funcionalidad:

Genera un botón interactivo que permite a los usuarios iniciar el proceso de verificación en el servidor.

Funcionamiento:

- Despliegue interactivo: Se muestra un botón en el canal que, al pulsarlo, inicia una secuencia de verificación.
- Proceso de autenticación: El sistema valida la identidad del usuario a través de respuestas a retos o integración con sistemas externos.

- Asignación automatizada: Una vez verificado, el usuario es automáticamente asignado para ser verificado.

Comando: `?rolasig`

Funcionalidad:

Define y asigna el rol que recibirán los usuarios al completar correctamente el proceso de verificación, otorgándoles acceso a las áreas restringidas del servidor.

Funcionamiento:

- Rol de verificación:

Este comando establece el **rol que se aplicará automáticamente** a los usuarios que logren verificarse.

Dicho rol será el que determine qué canales o funciones exclusivas podrán ver y utilizar.

Secuencia post-verificación:

Una vez que el usuario finaliza con éxito el proceso de verificación, el bot asigna de inmediato el rol configurado.

Notificación de éxito:

Tras la asignación, el bot envía una confirmación en el canal correspondiente, indicando que el rol fue aplicado correctamente.

Comando: `?copias`

Funcionalidad:

Genera una copia de todos los mensajes del servidor en los canales de texto, útil para auditorías o respaldos de información.

(Advertencia: Puede fallar si hay más de 300,000 mensajes)

Funcionamiento:

- **Recorrido masivo:** Se implementa un algoritmo iterativo que recorre la totalidad de los mensajes del servidor, segmentándolos en lotes.

- Optimización de recursos: Se utilizan técnicas de paginación y almacenamiento temporal para gestionar grandes volúmenes de datos.

- Limitaciones: Se notifica al usuario en caso de alcanzar límites de API o restricciones en la cantidad de mensajes procesados.

Comandos Divertidos

Estos comandos están orientados a crear experiencias lúdicas e interactivas en el servidor. Además de promover la participación, su implementación incorpora algoritmos de aleatoriedad y gestión de estados en tiempo real.

Comando: ?tirar

Funcionalidad:
Inicia el juego interactivo de **tira y afloja** dentro del servidor, donde los usuarios compiten por llevar la cuerda hacia su lado.

Funcionamiento:

- Inicio del juego: Al ejecutar el comando, el bot abre una partida de tira y afloja disponible para los usuarios del canal.

- Mecánica: Los jugadores interactúan con el mensaje del bot para “tirar” de la cuerda hacia su lado.

- Competencia dinámica: La fuerza acumulada de cada equipo determina el avance de la cuerda.

- Finalización: El juego concluye cuando uno de los bandos logra llevar la cuerda totalmente a su lado, proclamando un ganador.

- Registro opcional: El resultado puede integrarse en sistemas de estadísticas o logs internos del servidor.

Comando: `?ruleta`

Funcionalidad:

Inicia una partida de "Ruleta Rusa" entre el usuario y el bot, en la que ambos toman turnos de disparo.

Funcionamiento:

- Inicialización del juego: Se establecen parámetros como el número de vidas (ambos inician con 5) y se define la cantidad de balas (12 en total) con una distribución aleatoria entre reales y de fogueo.

- Turnos y recarga: El juego alterna turnos entre el usuario y el bot, utilizando un algoritmo de extracción aleatoria de balas. Cuando se agotan, se recarga el arma automáticamente.

- Condiciones de victoria: El juego termina cuando el usuario o el bot pierden todas sus vidas. En caso de que el usuario se quede sin vidas, se ejecuta una acción de baneo.

- Control de tiempo: Se establece un límite de 15 minutos para la respuesta del usuario, implementando un timeout para finalizar la partida en caso de inactividad.

Comando: `?trivialidades`

Funcionalidad:

Genera una trivia aleatoria con preguntas sobre temas variados, en la que el usuario debe reaccionar con emojis para responder.

Funcionamiento:

- Selección aleatoria: El sistema utiliza un arreglo de preguntas con probabilidades asignadas, asegurando la distribución equitativa entre las opciones.

- Interfaz interactiva: Se envía la pregunta al canal y se espera la reacción del usuario con emojis predefinidos ("1" o "2") dentro de un límite de 60 segundos.

- Verificación de respuestas: El bot valida la respuesta y edita el mensaje original para indicar si la respuesta fue correcta o incorrecta.

- Gestión de tiempo: En caso de inactividad, se notifica que el tiempo ha expirado.

Comando: `?aleatorio`

Funcionalidad:

Permite a los usuarios obtener un resultado aleatorio entre dos opciones ("cara" o "cruz").

Funcionamiento:

- Distribución equitativa: Se define una lista con dos opciones, asignando un 50% de probabilidad a cada una.

- Selección aleatoria: El bot utiliza una función de randomización para seleccionar la opción y responde con la imagen o el texto correspondiente.

- Simplicidad operativa: El comando se ejecuta de forma inmediata, siendo ideal para decisiones rápidas.

Comando: `?diputado`

Funcionalidad:

Muestra una imagen aleatoria de un diputado o figura política, seleccionada en función de probabilidades predefinidas.

Funcionamiento:

- Lista de opciones: Se mantienen varias imágenes, cada una con una probabilidad específica.

- Selección probabilística: Se utiliza un algoritmo para seleccionar la imagen basada en el valor de probabilidad, ofreciendo una distribución controlada.

- Respuesta visual: El bot envía la imagen resultante al canal, permitiendo su visualización directa.

Comando: `?chistes`

Funcionalidad:

Permite a los usuarios recibir un chiste aleatorio al ejecutar el comando, con un enfoque en generar humor mediante respuestas predefinidas.

Funcionamiento:

- Chistes: Se utiliza una lista que contiene múltiples chistes, cada uno preparado para ser mostrado de forma independiente.

- Selección aleatoria: Se aplica una función de randomización para elegir el chiste, asegurando una experiencia variada.

- Simplicidad de uso: El comando se activa únicamente con el mensaje `?chistes`, sin requerir parámetros adicionales.

Comando: `?ppt`

Funcionalidad:

Inicia una partida de "Piedra, Papel o Tijera" entre el usuario y el bot.

Funcionamiento:

- Interacción directa: El bot invita al usuario a elegir entre "piedra", "papel" o "tijera" y espera la respuesta durante 15 segundos.

- Lógica de comparación: Una vez recibida la elección, se ejecuta un algoritmo que compara las opciones y determina el resultado del juego.

- Validación y notificación: En caso de respuesta inválida o inactividad, se cancela el juego y se notifica al usuario.

Comando: `?8ball`

Funcionalidad:

Simula el comportamiento de la bola 8 mágica, ofreciendo respuestas aleatorias a las preguntas formuladas por el usuario.

Funcionamiento:

- Base de respuestas: Se define un conjunto de respuestas preestablecidas, incluyendo opciones afirmativas, negativas y neutrales.

- Selección aleatoria: El comando utiliza un algoritmo de selección para responder con una opción aleatoria.

- Formato de respuesta: La respuesta se envía directamente al canal, permitiendo interacciones rápidas y dinámicas.

Comando: `?buscaminas`

Funcionalidad:

Inicia una partida interactiva de Buscaminas en Discord, generando un tablero de 10x10 celdas con 20 minas distribuidas aleatoriamente.

Funcionamiento:

- Generación del tablero: Se crea una matriz de 10x10 en la que se asignan minas y se calculan los números indicativos de las minas adyacentes.

- Uso de spoilers: Cada celda se representa con emojis ocultos (spoilers) para mantener la intriga y el desafío del juego.

- Segmentación de mensajes: El tablero se divide en fragmentos de 15 celdas para facilitar su envío en mensajes separados.

Comando: `?buscaminas2`

Funcionalidad:

Permite jugar una versión avanzada de Buscaminas, generando una cuadrícula de 24 filas y 30 columnas con 256 minas.

Funcionamiento:

- Evitar la pérdida inmediata: Se asegura que las primeras 3x3 celdas estén libres de minas para brindar una mejor experiencia inicial al jugador.

- Algoritmo de colocación: Se generan las minas de forma aleatoria, incrementando el contador en celdas adyacentes para calcular los números indicativos.

- Representación visual: La cuadrícula se envía en fragmentos, utilizando spoilers para ocultar el contenido hasta que el jugador lo descubra.

- Limitaciones de interacción: El código muestra el estado inicial del tablero sin implementar la interacción directa para descubrir celdas a través de comandos.

Comando: `?buscar <pregunta>`

Funcionalidad:

Realiza una consulta en Google para responder preguntas de los usuarios, extrayendo el primer fragmento relevante de la búsqueda.

Funcionamiento:

- Extracción de la consulta: El bot elimina el comando `?buscar` y toma el texto restante como consulta.

- Llamada a la API: Se envía una solicitud HTTP a la API de búsqueda para obtener resultados actuales.

- Selección de fragmento: Se extrae el primer snippet relevante y se envía como respuesta en el canal.

- Manejo de errores: En caso de fallos en la conexión o ausencia de resultados, se notifica al usuario.

Comando: `?alerta <mensaje>`

Funcionalidad:

Permite a los administradores enviar un mensaje personalizado a todos los miembros del servidor mediante mensajes directos (DM).

Funcionamiento:

- Verificación de contexto y permisos: Se valida que el comando se ejecute en un servidor y que el usuario tenga permisos de administrador.

- Extracción de parámetros: Se toma el mensaje posterior al comando para enviarlo a cada miembro.

- Iteración sobre miembros: El bot recorre la lista de miembros, omitiendo bots, y envía el mensaje a cada uno.

- Reporte de resultados: Se informa en el canal sobre el número de mensajes enviados correctamente y los fallos ocurridos (por ejemplo, usuarios con DM deshabilitados).

Comando: `?reportar <messageLink> <reason>`

Funcionalidad:

Permite a los usuarios reportar mensajes específicos al Consejo a través de mensajes directos, asegurando la privacidad del proceso.

Funcionamiento:

- Restricción de contexto: El comando solo se acepta en mensajes directos con el bot.

- Extracción de IDs: Se extraen los identificadores del servidor, canal y mensaje a partir del enlace proporcionado.
- Construcción del reporte: Se crea un mensaje detallado que incluye el contenido reportado, el autor, el motivo y un enlace directo al mensaje.

- Envío del reporte: El mensaje se envía a un administrador designado mediante MD.

- Confirmación y manejo de errores: Se notifica al usuario sobre el estado del reporte y se manejan errores en caso de enlaces inválidos o problemas de permisos.

Comando: `?permrol <roleId>`

Funcionalidad:

Proporciona un listado detallado de los permisos de un rol específico, mostrando cada permiso con indicadores de habilitado o deshabilitado.

Funcionamiento:

- Consulta interna: Se accede a la configuración de roles del servidor utilizando el ID proporcionado.

- Formato de salida: Se utiliza un formato visual para indicar el estado de cada permiso.

- Restricción de seguridad: Por motivos de privacidad y seguridad, el comando no permite consultar permisos de tu propio rol ni de roles superiores en la jerarquía.

- Uso en auditoría: Este comando es fundamental para la revisión y auditoría de roles en entornos con alta exigencia de seguridad.

Comando: `?rolc <permNumber>-s/n <roleId>`

Funcionalidad:

Permite modificar los permisos de un rol específico, habilitando o deshabilitando permisos a partir de un número predefinido que identifica cada permiso.

Funcionamiento:

- Formato de comando: Se debe ingresar un número de permiso seguido de `-s` (habilitar) o `-n` (deshabilitar), junto con el ID del rol.

- Validación y aplicación: El bot verifica el formato y aplica el cambio en la configuración interna de permisos.

- Restricción jerárquica: Por seguridad, no es posible modificar roles iguales o superiores al rol del usuario que ejecuta el comando.

- Confirmación: Se notifica al usuario sobre el éxito o fallo en la modificación, registrando el cambio para futuras auditorías.

Comando: `?ruletapvp`

Funcionalidad:

Desafía a otro usuario a participar en una ruleta PvP, donde ambos compiten en un duelo interactivo.

Funcionamiento:

- Invitación al duelo: Se mencionan dos usuarios y se inicializan variables de juego (por ejemplo, balas reales, y vacias).

- Algoritmo de turnos: La función intercala disparos entre ambos participantes, aplicando probabilidades y reduciendo balas según el resultado del disparo.

- Determinación del ganador: El juego concluye cuando uno de los participantes es baneado.

Comandos de Utilidad

Estos comandos están diseñados para facilitar tareas de consulta, suscripción y manejo de información dentro del servidor, permitiendo una interacción más rica y personalizada.

Comando: ?suse <texto de la sugerencia>

Funcionalidad:
Permite a los usuarios **enviar sugerencias** directamente al canal configurado con ?suge.

Funcionamiento:

- Envío automatizado: El bot toma el texto ingresado y lo publica en el canal de sugerencias previamente configurado.

- Formato estructurado: Cada mensaje incluye información del autor y la fecha para facilitar la gestión y el seguimiento.

- Compatibilidad con administración: Los moderadores pueden revisar, responder o destacar sugerencias relevantes directamente desde el canal.

- Registro opcional: Se puede mantener un historial de sugerencias para auditoría o análisis interno del servidor.

Comando: ?icono

Funcionalidad:
Muestra el **icono del servidor** en alta resolución, compatible con imágenes estáticas y animadas (GIF).

Funcionamiento:

- Extracción directa: Obtiene el icono actual del servidor desde la configuración principal.

- Resolución máxima: El bot envía la versión de mayor calidad disponible.

- Compatibilidad con GIF: Si el icono es animado, se enviará en formato dinámico.

- Transparencia: Incluye la URL directa del recurso para su descarga o uso externo.

Comando: ?avatar (@usuario | ID | respuesta)

Funcionalidad:
Muestra el **avatar de un usuario** en alta resolución, incluyendo soporte para GIF si el usuario tiene avatar animado.

Funcionamiento:

- Parámetros flexibles: Puede usarse con mención directa, ID del usuario o respondiendo a un mensaje.

- Extracción personalizada: Obtiene el avatar actual en la mejor resolución disponible.

- Compatibilidad con animaciones: Si el usuario posee un avatar animado, se mostrará en formato GIF.

- Acceso directo: El bot comparte la URL de la imagen para su descarga o integración en otros usos.

Comando: `?getregistro`

Funcionalidad:

Muestra el registro de sorteos realizados en el servidor.

Funcionamiento:

- Consulta de historial: Se extraen los datos de sorteos anteriores de la base de datos.

- Formato detallado: Se presenta la información en un formato estructurado, con detalles sobre fechas, participantes y ganadores.

- Utilidad en auditorías: Este comando es útil para revisar la transparencia y gestión de los eventos de sorteos.

Comando: `?crearsorteo`

Funcionalidad:

Crea un sorteo configurable como privado o público, gestionando la participación y el proceso de selección del ganador.

Funcionamiento:

- Parámetros de configuración: El administrador define la modalidad del sorteo, duración, premios y condiciones de participación.

- Automatización: El bot organiza la inscripción de usuarios y, al finalizar, selecciona un ganador de forma aleatoria o en base a criterios predefinidos.

- Registro y notificación: Se registra el evento en la base de datos y se notifica el resultado en el canal configurado.

Comando: `?nivel [@usuario | ID]`

Funcionalidad:

Muestra el nivel y la experiencia (XP) acumulada de un usuario, ya sea el que se menciona o el que ejecuta el comando.

Funcionamiento:

- Consulta personalizada: Se accede al perfil del usuario para extraer datos de experiencia y nivel.

- Visualización: La información se presenta en un formato gráfico o numérico, permitiendo comparar el progreso con el ranking general del servidor.

- Integración en sistemas de gamificación: Este comando es fundamental en entornos que utilizan sistemas de niveles y recompensas.

Comando: `?clasificacion`

Funcionalidad:

Muestra el ranking de niveles del servidor, con paginación para facilitar la visualización en servidores de gran tamaño.

Funcionamiento:

- Ordenamiento: Se organiza la lista de usuarios en función de la experiencia acumulada.

- Paginación: La salida se divide en varias páginas, permitiendo una consulta eficiente sin saturar el canal.

- Actualización en tiempo real: El ranking se actualiza dinámicamente conforme los usuarios acumulan experiencia.

Comando: `?mbuscar`

Funcionalidad:

Busca música en línea y muestra una lista de opciones que luego pueden reproducirse en el canal de voz.

Funcionamiento:

- Integración con APIs externas: Se realiza una consulta en servicios de streaming o bases de datos de música.

- Filtrado de resultados: Se presentan resultados relevantes basados en la consulta del usuario.

- Interfaz interactiva: Permite seleccionar y reproducir música, integrándose con el sistema de audio del bot.

Comando: `?subs`

Funcionalidad:

Permite a los usuarios suscribirse para recibir actualizaciones y notificaciones sobre eventos o novedades del servidor.

Funcionamiento:

- Registro de suscriptores: Se almacena el ID del usuario en una lista de suscriptores para futuras notificaciones.

- Automatización de alertas: El bot envía mensajes automáticos a los suscritos cuando se publica nueva información.

- Configuración personalizada: Los administradores pueden ajustar la frecuencia y el contenido de las actualizaciones.

Comando: `?decir`

Funcionalidad:

Permite al bot repetir un mensaje específico proporcionado por el usuario, eliminando el mensaje original para mantener la limpieza del canal.

Funcionamiento:

- Extracción y validación: Se separa el comando del contenido, verificando que el mensaje no esté vacío.

- Reenvío y eliminación: El bot envía el mensaje reconstruido al canal y elimina el mensaje original para evitar redundancias.

- Uso moderado: Este comando se utiliza para crear mensajes y para enfatizar comunicaciones importantes.

Sistemas Avanzados de Registro y Auditoría

La gestión de eventos y actividades en el servidor se complementa con sistemas avanzados de registro y auditoría. Estos permiten la supervisión detallada y la trazabilidad de acciones críticas, mejorando la seguridad y la transparencia en el entorno.

Registro de Mensajes Eliminados

Funcionalidad:

Cada vez que se elimina un mensaje, el sistema genera un registro que incluye:

- Canal de eliminación.
- Autor original.
- Usuario que ejecutó la eliminación (si es determinable).
- Contenido del mensaje eliminado.
- Información de archivos adjuntos asociados.

Detalles:

- Integración en tiempo real: Se intercepta eventos de eliminación y almacena los datos en una base de datos segura.

- Acceso restringido: Solo los administradores pueden consultar estos registros mediante comandos específicos, garantizando la privacidad y la integridad de la información.

- Uso en auditorías: Los registros permiten revisar incidentes y detectar patrones de abuso o errores en la moderación.

Seguimiento de Mensajes Editados

Funcionalidad:

Registra todas las ediciones realizadas en los mensajes, notificando al canal configurado y almacenando los cambios.

Detalles:

- Comparación de versiones: Se guarda tanto el contenido original como el editado, permitiendo comparaciones históricas.

- Enlaces directos: Se genera un enlace al mensaje editado, facilitando su revisión.

- Notificaciones automáticas: Las ediciones se envían a un canal específico para una rápida verificación por parte de los moderadores.

Registro de Reacciones Eliminadas

Funcionalidad:

Cada vez que un usuario elimina una reacción de un mensaje, se genera un registro que incluye:

- Canal donde ocurrió.
- Mensaje afectado.
- Emoji eliminado.
- Usuario que realizó la reacción.

Detalles:

- Intercepción de eventos: Se captura el evento de eliminación de reacciones y se almacena en el sistema de auditoría.

- Uso para análisis: Permite detectar comportamientos inusuales o manipulaciones en la interacción de los usuarios.

Seguimiento de Cambios en Canales

Funcionalidad:

Detecta modificaciones en permisos o configuraciones de canales y genera un informe detallado.

Detalles:

- Comparación de configuraciones: Se registran los permisos anteriores y nuevos, permitiendo identificar cambios precisos.

- Notificación al administrador: El sistema envía un reporte automatizado con los cambios, facilitando la intervención en caso de configuraciones erróneas.

- Auditoría de accesos: Permite correlacionar cambios con actividades sospechosas en el servidor.

Registro de Creación y Eliminación de Canales

Funcionalidad:

Genera registros automáticos cada vez que se crea o elimina un canal.

Detalles:

- Metadatos del canal: Se almacena el nombre, tipo, ID único y usuario responsable de la acción.

- Integración con logs: Estos registros se integran en el sistema general de auditoría para seguimiento a largo plazo.

- Notificaciones: Se pueden configurar alertas en tiempo real para administradores cuando se realicen estas acciones.

Notificación de Nuevos Miembros y Registro de Salidas

Funcionalidad:

Registra la entrada y salida de miembros, notificando a los administradores y almacenando los datos para análisis estadísticos.

Detalles:

- Datos de ingreso: Se guarda el nombre, etiqueta, ID único y la fecha/hora exacta de ingreso.

- Datos de salida: Similarmente, se documenta la salida de miembros, permitiendo analizar las tendencias de abandono.

- Integración en sistemas de bienvenida: Las notificaciones pueden ser utilizadas para activar mensajes de bienvenida o encuestas de salida.

Configuración del Sistema de Registro y Auditoría

Funcionalidad:

Permite a los administradores configurar un canal específico donde se enviarán todos los registros y logs de actividad.

Detalles:

- Uso del comando `?configlogs`: Se establece el ID del canal destino y se activa el sistema de logging.

- Restricción de permisos: Solo usuarios con privilegios elevados pueden modificar esta configuración, asegurando la integridad del sistema.

- Persistencia y análisis: Los logs se almacenan de forma persistente, permitiendo su consulta y análisis histórico mediante herramientas externas o comandos internos del bot.

Preguntas Frecuentes (FAQ Técnica)

¿Debo verificarme en cada servidor donde esté el bot?

No. El sistema de verificación de **Señor Fino** funciona de manera **centralizada y persistente**. Esto significa que solo necesitas **verificarte una vez** utilizando el botón de verificación. A partir de ese momento, en cualquier otro servidor que utilice el sistema oficial de verificación del bot, únicamente deberás ejecutar el comando correspondiente y tu verificación será **aceptada automáticamente** sin necesidad de repetir el proceso.

¿Dónde está la página web oficial?

La página web oficial de **Señor Fino** es:
https://senorfino.pro:7020/inicio/

En ella se publican:

- Actualizaciones técnicas del bot.
- Documentación extendida sobre su funcionamiento.
- Servicios asociados (como consultas administrativas).
- Recursos adicionales para administradores y usuarios avanzados.

¿Qué usuarios pueden ser baneados globalmente?

El sistema de **baneos globales** está diseñado para proteger a la comunidad. Generalmente, los baneos globales se aplican a:

- Cuentas utilizadas en **raids masivos** (entrada de bots, spam, destrucción de canales, etc.).
- Usuarios que incurren en **abuso grave de la plataforma**, como spam automatizado, fraudes o estafas.
- Individuos involucrados en actividades ilícitas que utilizan Discord como herramienta.

El proceso de inclusión en la lista global se lleva a cabo bajo criterios técnicos y verificaciones internas, minimizando al máximo los falsos positivos.

¿El bot expone información privada de los usuarios?

No. **Señor Fino** no expone ni hace públicos datos privados de los usuarios.
Los rumores sobre supuestas filtraciones o exposiciones carecen de fundamento: la seguridad y privacidad de las comunidades es una prioridad fundamental en el diseño y funcionamiento del bot.

¿Para qué están las páginas de contacto?

Las páginas de contacto oficiales están destinadas **exclusivamente** a gestiones serias y necesarias, como:

- Reporte de fallos técnicos o errores graves.
- Solicitudes de soporte administrativo.
- Comunicaciones relacionadas con la seguridad o el mal uso del bot.

No se recomienda utilizarlas para consultas triviales, sugerencias informales o temas ajenos a la administración.

¿Se añadirán comandos de contenido NSFW o inapropiado?

No. Por diseño, **Señor Fino** no incluirá comandos de carácter sexual, explícito o inapropiado. El objetivo del bot es ser una **herramienta seria, confiable y multiuso**, centrada en la administración, auditoría y seguridad de servidores, complementada con funciones interactivas sanas y de entretenimiento.

Glosario de Términos

- Flood: Enviar muchos mensajes en un corto periodo de tiempo.

- Raid: Se refiere a cualquier acción masiva, coordinada o automatizada cuyo objetivo es interrumpir, saturar o dañar un servidor de Discord. Un raid puede incluir:

- Entrada repentina de muchos usuarios o bots falsos.
- Envío masivo de mensajes (spam/flood) en múltiples canales.
- Creación excesiva de canales, roles o categorías en poco tiempo.
- Eliminación en masa de canales, roles o mensajes.
- Uso de bots de raid configurados para ejecutar ataques automáticos.
- Menciones masivas a todos los miembros (@everyone / @here) de forma abusiva.
- Reacciones masivas en mensajes para generar caos o lag.

En resumen, un raid no se limita solo a la entrada de bots o usuarios, sino a cualquier intento de sabotaje en el servidor mediante acciones desproporcionadas o automatizadas.

- RoleID: Identificador único de un rol en Discord, utilizado para configuraciones internas.

- Toggle: Acción de activar o desactivar una función con un solo comando.

- Logs: Archivos o registros donde el bot guarda información sobre eventos importantes (baneos, cambios, auditorías).

Casos de Uso

Moderación y Seguridad:

- Control de raids y spam:

?cdab [<n>] → Activa el sistema antiraid.

?autar, ?banaut, ?botdet → Controlan flood, banean automáticamente y detectan mensajes de bots.

?verfmult y ?verificar → Detección y prevención de multicuentas.

?ban (@usuario) y ?avisar (@usuario) → Aplicación de sanciones rápidas.

?silenciar (@usuario) [TIEMPO] → Mantiene el orden en casos de spam o discusiones.

- Protección avanzada contra raids (acciones maliciosas como bots de raid, creación/borrado masivo de canales, spam de menciones, etc.):

?globalban → Bloquea a usuarios considerados peligrosos por Señor Fino en todos los servidores.

?rolid → Define qué rol será notificado en situaciones de flood o raid.

?alerta → Enviar un aviso masivo a toda la comunidad durante emergencias.

Configuración del Servidor:

- Sistemas de bienvenida y despedida:

?bieimg ... y ?bie, ?des → Configuración de imágenes, textos y mensajes automáticos para entradas/salidas.

- Sugerencias y reportes:

?suge <#canal|id> → Define canal de sugerencias.

?suse <texto> → Los usuarios envían sugerencias.

?reportar → Canaliza reportes de mal uso del bot.

- Tickets de soporte:

?tickets y ?confgtick → Creación de sistema de tickets con logs automáticos.

?adboton y ?reboton → Personalización de botones para tickets.

- Roles y permisos:

?rolasig, ?autorol, ?permrol, ?rolc → Configuración flexible de roles y permisos sin necesidad de usar la interfaz de Discord.

?glrol y ?revertglrol → Gestión de roles de verificación.

- Logs y control:

?configlogs → Registro de acciones del servidor.

?contarbaneos y ?edad → Estadísticas y rastreo de usuarios baneados o cuentas recientes.

Dinámica y Entretenimiento:

- Juegos y diversión:

?ruleta, ?ruletapvp, ?ruletaob, ?tirar → Juegos de azar con riesgo real.

?ppt, ?8ball → Juegos simples de interacción.

?buscaminas, ?buscaminas2 → Entretenimiento clásico dentro del chat.

?trivialidades, ?chistes, ?diputado → Contenido ligero y de humor.

- Sorteos y actividades:

?crearsorteo, ?getregistro → Creación y control de sorteos.

?subs → Sistema de suscripciones a anuncios/eventos.

- Música y multimedia:

?musica, ?mbuscar → Reproducción de música en canales de voz.

?icono → Mostrar icono del servidor en HD.

Utilidades y Comunidad:

- Sistema de niveles:

?nivel, ?clasificacion, ?subirnivel, ?bajarnivel, ?cniveles → Seguimiento de progreso de miembros.

- Notificaciones externas:

?caup → Canal de actualizaciones de mercados financieros.

?juegos → Avisos de juegos gratis.

- Mensajes y herramientas:

?decir, ?buscar, ?aleatorio → Utilidades rápidas para dinamizar el chat.

Conclusión

Esta guía exhaustiva está diseñada para administradores y usuarios avanzados que buscan aprovechar al máximo las capacidades de Señor Fino. Incluye tanto los comandos originales —que van desde juegos interactivos hasta funciones de búsqueda y reproducción— como los nuevos comandos que amplían las funcionalidades administrativas y de utilidad.

Cada comando se documenta con detalle, abarcando lógica de funcionamiento, validaciones, manejo de errores y aspectos técnicos clave, lo que garantiza una implementación robusta, facilita la auditoría y optimiza la gestión del servidor en entornos complejos y de alta demanda.

Se recomienda a los administradores familiarizarse con cada comando y realizar pruebas en entornos controlados antes de aplicarlos en servidores de producción, asegurando así la correcta configuración y el funcionamiento óptimo del bot. La integración de estos sistemas avanzados de registro y auditoría proporciona una capa adicional de seguridad y transparencia, permitiendo una respuesta eficaz ante incidentes y manteniendo un entorno seguro, ordenado y confiable para toda la comunidad.

Edición del 25/08/2025