Pregunta:

La salida optimizada standalone ya no se encuentra como tal fichero en las salidas de build de next.js ¿Ha sido sustituida por algún otro fichero actual o carpeta?

Respuesta:

Tienes razón en tu observación. La salida "standalone" de Next.js, habilitada a través de la configuración output: 'standalone' en next.config.js, no se presenta como un único archivo .zip o algo similar en la raíz del build.

En su lugar, cuando configuras output: 'standalone', Next.js crea una carpeta dentro de .next llamada **.next/standalone**.

Esta carpeta .next/standalone contendrá todos los archivos necesarios para desplegar tu aplicación Next.js de forma independiente, sin necesidad de instalar node\_modules en el entorno de producción. Esto es ideal para deployments en entornos como Docker, donde quieres una imagen lo más pequeña posible.

**¿Qué encontrarás dentro de .next/standalone?**

* **server.js**: Este es el archivo del servidor minimalista de Next.js que puedes usar para iniciar tu aplicación en producción. En lugar de usar next start, lo ejecutarías con node .next/standalone/server.js.
* **Archivos de tu aplicación**: Contiene un árbol de directorios con el código compilado de tu aplicación, incluyendo los componentes, páginas, API routes, etc., y sus dependencias (incluso algunas de node\_modules) que Next.js ha rastreado como necesarias para el funcionamiento.
* **.next/ (dentro de standalone)**: Dentro de la carpeta standalone, encontrarás una subcarpeta .next/ que contendrá los archivos estáticos (.next/static) y otros assets que Next.js necesita para servir las páginas.

**Puntos importantes a considerar con la salida standalone:**

* **public y .next/static**: Por defecto, la salida standalone **no copia las carpetas public ni .next/static** directamente dentro de standalone. Next.js asume que estos activos estáticos serán servidos por una CDN o un servidor web dedicado (como Nginx o Apache) para optimizar el rendimiento. Si necesitas que server.js los sirva, debes copiarlos manualmente a standalone/public y standalone/.next/static después del build.
  + Ejemplo de comando para copiar (después de next build):  
    Bash  
    cp -r public .next/standalone/ && cp -r .next/static .next/standalone/.next/
* **Variables de entorno en next.config.js**: Si usas serverRuntimeConfig o publicRuntimeConfig en tu next.config.js, los valores se serializarán en el archivo server.js en el momento del build. Esto significa que si cambias esas variables después del build, el servidor no las reconocerá a menos que reconstruyas la aplicación. Para variables que necesitas leer en runtime (después del build), es mejor usar process.env y asegurarte de que estén disponibles en el entorno donde se ejecuta server.js (como variables de entorno del sistema o del contenedor Docker).
* **next start no funciona**: Con la salida standalone, no debes usar next start. En su lugar, usa node .next/standalone/server.js para iniciar el servidor.

**En resumen:**

La salida optimizada "standalone" **sí existe**, pero no es un archivo único, sino una carpeta (generalmente .next/standalone) que contiene el mínimo necesario para ejecutar tu aplicación de forma autónoma. Next.js ha evolucionado para hacer estos builds lo más eficientes posible para entornos de despliegue modernos.