IndexTTS容器可执行命令及配置说明

以下是替换后的完整可执行命令（Windows PowerShell），已精准适配你指定的模型目录和n8n映射目录，同时融入--fp16参数优化及命令格式修正，保留所有核心功能：

第一步：先创建映射目录+下载IndexTTS-2模型（核心前置步骤）

Code block

```
# 创建模型目录（如果已创建可跳过）
mkdir -p D:\index_tts2_checkpoints

# 创建n8n映射的IndexTTS输出目录（关键！确保n8n能读取）
mkdir -p D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2
```

第二步：启动IndexTTS容器（核心命令，已替换路径+优化格式+添加--fp16）

说明：--rm 为可选项，可根据需求取舍；-it 默认添加（交互模式），反引号`后无多余空格，确保PowerShell能正常续行

核心启动命令说明（参数含义）：

- -it：交互模式，可查看实时日志，Ctrl+C可停止容器
- --gpus all：启用GPU（无GPU可删除此参数）
- --name index-tts：给容器起固定名称（方便后续管理，避免随机名称）
- -p 7860:7860：映射WebUI端口，访问地址http://127.0.0.1:7860
- -v 映射目录：3个映射分别为①模型目录（D:\index_tts2_checkpoints→/app/checkpoints）、②输出目录（D:\...\outputs→/app/outputs）、③临时目录（D:\...\tmp→/app/tmp），确保模型加载和文件读写正常
- --fp16：启用半精度推理，提升GPU性能（无GPU可删除）

纯命令（可直接复制执行，无注释）：

Code block

```
docker run -it --name index-tts --gpus all -p 7860:7860 ^
    -v D:\index_tts2_checkpoints:/app/checkpoints ^
    -v
D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2\outputs:/app/outputs
```

```
    ^
4       -v D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2\tmp:/app/tmp ^
5       nanaoto/index-tts:latest-pytorch2.8.0-cuda12.8-amd64 ^
6       uv run webui.py --model_dir /app/checkpoints --host 0.0.0.0 --fp16
```

补充：简化版命令（无--rm，默认带-it交互模式，后台+实时日志）

简化版命令说明（无--rm，默认带-it，可直接复制执行）：

Code block

```
1   docker run -it --name index-tts --gpus all -p 7860:7860 -v
    D:\index_tts2_checkpoints:/app/checkpoints -v
    D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2\outputs:/app/outputs
    -v D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2\tmp:/app/tmp
    nanaoto/index-tts:latest-pytorch2.8.0-cuda12.8-amd64 uv run webui.py --
    model_dir /app/checkpoints --host 0.0.0.0 --fp16
```

命名后常用容器管理命令（可直接复制执行）：

- 停止容器：`docker stop index-tts`（无需记容器ID，直接用固定名称停止）
- 重启容器（停止后重新启动）：`docker start index-tts`（启动后可继续访问WebUI）
- 查看容器运行状态：`docker ps -a | findstr index-tts`（判断容器是否在运行/已停止）
- 删除容器（需先停止，避免名称冲突）：`docker rm index-tts`（删除后下次启动会重新创建容器）
- 进入容器内部（排查问题用）：`docker exec -it index-tts sh`（进入后可查看容器内文件结构）

第三步：验证路径映射是否生效

1. 容器启动后，在IndexTTS WebUI（http://127.0.0.1:7860）生成一段测试音频；
2. 打开本地路径 D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2\outputs，能看到生成的音频文件（如 audio_xxxx.wav/mp3）→ 输出目录映射成功；
1. 进入n8n容器，执行以下命令验证是否能访问该音频：
   `# 进入n8n容器（替换为你的n8n容器名/ID）`
   `docker exec -it 你的n8n容器名 sh`

   `# 查看n8n容器内的IndexTTS音频文件`
   `ls /home/node/output/indextts2/` 能看到相同的音频文件 → n8n与IndexTTS目录互通成功。

补充：若用docker-compose管理（可选，长期运行推荐）

说明：若容器已存在（如用“无--rm”的docker run命令启动过），可直接在Docker桌面端点击“启动”重启容器，无需通过此文件；创建 docker-compose.yml 文件（路径可自定义，如 D:\IndexTTS\docker-compose.yml）的核心价值是“便捷管理容器配置和首次/批量启动”，内容如下：

Code block

```
version: '3.8'
services:
  index-tts:
    image: nanaoto/index-tts:latest-pytorch2.8.0-cuda12.8-amd64
    container_name: index-tts
    restart: unless-stopped  # 异常退出自动重启
    ports:
      - "7860:7860"
    volumes:
      # 模型目录映射（替换后的路径）
      - D:\index_tts2_checkpoints:/app/checkpoints
      # 输出目录映射到n8n指定路径（修正为/tmp/gradio，Gradio默认输出目录）
      - D:\N8NCustomDir\n8n_003\Video_Generation\Output\indextts2:/tmp/gradio
    deploy:
      resources:
        reservations:
          devices:
            - capabilities: [gpu]  # 声明需要GPU
    environment:
      - NVIDIA_VISIBLE_DEVICES=all  # 启用所有GPU
    stdin_open: true
    tty: true
    command: uv run webui.py --model_dir /app/checkpoints --host 0.0.0.0 --
fp16  # 添加--fp16，优化GPU推理性能
```

启动/停止命令：

Code block

```
# 后台启动
docker-compose up -d

# 查看日志（排查报错）
docker-compose logs -f index-tts

# 停止容器
docker-compose down
```

关键说明

- 模型目录D:\index_tts2_checkpoints需要提前手动下载IndexTTS-2模型文件，nanaoto/index-tts镜像不支持启动容器时自动下载模型；若未提前下载，容器启动后WebUI会提示“模型文件不存在”。手动下载命令（Windows PowerShell）：

```
# 配置国内镜像加速（避免下载慢，无VPN时使用）
$env:HF_ENDPOINT = "https://hf-mirror.com"

# 安装huggingface工具（若未安装）
pip install huggingface-hub[cli] -i https://pypi.tuna.tsinghua.edu.cn/simple

# 下载IndexTTS-2模型到指定目录
hf download IndexTeam/IndexTTS-2 --local-dir=D:\index_tts2_checkpoints

# 有VPN时简化下载命令（无需镜像）
# pip install huggingface-hub[cli]
# hf download IndexTeam/IndexTTS-2 --local-dir=D:\index_tts2_checkpoints
```

- n8n容器内访问IndexTTS音频的路径调整为 /home/node/output/indextts2/outputs/（因为n8n的/home/node/output 对应本地 D:\N8NCustomDir\n8n_003\Video_Generation\Output），生成的音频会同步到该路径下，n8n可正常读取；
- 若启动时提示“权限不足”，右键本地映射目录 → 属性 → 安全 → 添加“Everyone”并授予“读取/写入”权限即可；
- --fp16参数说明：IndexTTS默认使用fp32精度推理，--fp16仅对GPU生效，可显著提升推理速度、降低显存占用，无GPU时添加此参数无效果（可删除）；
- 容器命名说明：所有启动命令已添加--name index-tts（固定容器名），后续管理更便捷：① 停止容器：docker stop index-tts；② 重启容器：docker start index-tts；③ 若提示“容器名已存在”，先执行docker rm index-tts删除旧容器再启动；
- --rm/-it参数说明：-it 默认添加（交互模式，可查看实时日志，Ctrl+C即可停止容器）；--rm 为可选项（容器停止后自动删除，避免残留），若添加--rm，停止后容器会被自动删除，下次启动需重新创建；
- 端口冲突处理：若7860端口被占用（如与其他Gradio服务冲突），可将-p 7860:7860改为-p 7861:7860，访问地址变为http://127.0.0.1:7861；

- docker-compose.yml适用场景补充：此文件并非必需——容器已存在时，直接在Docker桌面端点击“启动”重启容器即可；其核心作用是“首次启动配置保存”和“长期稳定运行管理”（如自动重启、批量启动），适合需要固定配置、避免重复输入命令的场景；
- 启动日志解读（正常现象，无需担心）：
  1. “CUDA Version 12.8”：说明GPU和CUDA环境正常加载，--fp16参数可正常生效；
  2. “GPT weights restored from: /app/checkpoints/gpt.pth”“s2mel weights restored from: /app/checkpoints/s2mel.pth”：你提前下载的核心模型已成功加载（路径是你挂载的本地目录），这部分是IndexTTS的核心，无需重复下载；
  3. 启动时补充下载的文件（如preprocessor_config.json、model.safetensors、semantic_codec/model.safetensors、bigvgan_generator.pt等）：是IndexTTS运行必需的辅助依赖（包括语义编码模型、声码器、文本归一化工具等），这些文件镜像里未预装，首次启动会自动从Hugging Face拉取；
  4. 补充下载的文件会缓存到容器内（路径如/root/.cache/huggingface），**下次启动容器时不会重复下载**（只要不删除容器，缓存就一直存在）；
  5. “campplus_cn_common.bin”“bigvgan_v2_22khz_80band_256x”：分别是中文语音识别辅助模型、声码器模型，用于提升语音生成的音质和准确性，属于正常依赖；
  6. GPT2InferenceModel相关警告：属于工具类提示，不影响功能使用，可忽略；
  7. 最终“Running on local URL: http://0.0.0.0:7860”：说明容器已完全启动成功，直接访问http://127.0.0.1:7860就能使用WebUI；
  8. 输出目录说明：已更新为你指定的映射配置（本地outputs→/app/outputs、本地tmp→/app/tmp），音频文件会生成在本地outputs目录，适配你的实际使用需求。

（注：文档部分内容可能由 AI 生成）