PR #30308 完整报告

sgl-project/sglang

docs(install): add nightly install + docker tag guidance, and auto-bump version on release tag

合并时间：2026-07-08 03:10

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30308

执行摘要

- 一句话：补充 nightly 安装指南和 Docker 标签说明，自动 bump 文档版本
- 推荐动作：建议发布流程负责人和维护者阅读本 PR，特别是 bump_docs_install_version.py 的版本匹配策略和多文件同步机制。其设计思路（复用 utils、验证后更新、对 mutable tags 放行）可推广到其他自动化场景。

功能与动机

docs_new 安装文档中的 git clone -b vX.Y.Z 和 lmsysorg/sglang:vX.Y.Z 版本号只能手工更新，容易过时；且缺少 nightly 安装方法和 Docker 标签可变的说明。本 PR 通过自动化脚本和工作流消除版本同步滞后，同时补齐缺失的文档。

实现拆解

1. 新增 scripts/release/bump_docs_install_version.py，定义两个正则 CLONE_RE 和 DOCKER_RE 分别匹配 git clone 和 Docker image 的版本；通过 replace_version 函数替换指定文件中的版本号；main 函数解析参数、验证版本、循环更新并最终验证。
2. 新增 .github/workflows/bot-bump-docs-version.yml，监听 v* tag 推送和 workflow_dispatch 事件。先解析版本号，然后 checkout main 分支，运行 bump 脚本，若无变更则退出，否则调用 commit_and_pr.sh 创建 PR。
3. 编辑 docs_new/docs/get-started/install.mdx，在 Method 1 下插入 Nightly builds 小节，给出使用 --prerelease=allow --index-strategy unsafe-best-match --extra-index-url 的命令；在 Method 3 Docker 部分添加 <Note> 说明 latest/dev 为可变标签，建议按版本固定。
4. 更新 scripts/release/README.md，添加 bump_docs_install_version.py 的用法说明和更新文件列表。

关键文件：

- scripts/release/bump_docs_install_version.py（模块 发布脚本；类别 source；类型 core-logic；符号 read_current_version, stale_versions, replace_version, main）：核心自动化脚本，定义版本替换逻辑，是自动化流程的关键。
- .github/workflows/bot-bump-docs-version.yml（模块 CI 工作流；类别 infra；类型 infrastructure）：触发自动化版本 bump 的 CI 工作流，是自动化的关键基础设施。

- docs_new/docs/get-started/install.mdx（模块 安装文档；类别 other；类型 core-logic）：用户可见的安装文档，新增 nightly 构建指南和 Docker 标签说明。
- scripts/release/README.md（模块 发布脚本；类别 docs；类型 documentation）：记录版本 bump 脚本用法，便于团队了解和使用。

关键符号：read_current_version, stale_versions, replace_version, main

关键源码片段

scripts/release/bump_docs_install_version.py

核心自动化脚本，定义版本替换逻辑，是自动化流程的关键。

```
# scripts/release/bump_docs_install_version.py
# 核心替换函数：匹配并替换两种模式中的版本号

def replace_version(file_path: Path, new_version: str) -> bool:
    if not file_path.exists():
        print(f"Warning: {file_path} does not exist, skipping")
        return False

    content = file_path.read_text()
    # 替换 git clone -b v<old> 为 v<new>
    new_content = CLONE_RE.sub(rf"\g<1>v{new_version}\g<3>", content)
    # 替换 lmsysorg/sglang:v<old> 为 v<new>，保留后缀
    new_content = DOCKER_RE.sub(rf"\g<1>v{new_version}", new_content)

    if content == new_content:
        print(f"No changes needed in {file_path}")
        return False

    file_path.write_text(new_content)
    print(f"✓ Updated {file_path}")
    return True


def main():
    parser = argparse.ArgumentParser(
        description="Bump the 'install from source' release-branch version in the docs"
    )
    parser.add_argument("new_version", help="New version (e.g., 0.5.13, 0.5.13rc0)")
    args = parser.parse_args()

    new_version = normalize_version(args.new_version)
    if not validate_version(new_version):
        print(f"Error: Invalid version format: {new_version}")
        sys.exit(1)

    repo_root = get_repo_root()
    primary = repo_root / FILES_TO_UPDATE[0]
```

```
    old_version = read_current_version(primary)
    print(f"Current docs install version: {old_version}")
    print(f"New docs install version: {new_version}")

    comparison = compare_versions(new_version, old_version)
    if comparison == 0:
        print("Docs are already at this version; nothing to do.")
        return
    elif comparison < 0:
        print(f"Warning: new version ({new_version}) is older than the docs version "
              f"({old_version}); proceeding anyway.")

    updated_count = 0
    for file_rel in FILES_TO_UPDATE:
        file_abs = repo_root / file_rel
        if replace_version(file_abs, new_version):
            updated_count += 1

    print(f"\nSuccessfully updated {updated_count} file(s)")
    print(f"Docs install version bumped from {old_version} to {new_version}")

    # 最后验证所有文件不再包含旧版本
    print("\nValidating version updates...")
    for file_rel in FILES_TO_UPDATE:
        file_abs = repo_root / file_rel
        if not file_abs.exists():
            continue
        stale = stale_versions(file_abs, new_version)
        if stale:
            print(f"✗ {file_rel} still pins v{', v'.join(sorted(set(stale)))}")
        else:
            print(f"✓ {file_rel} validated")
```

评论区精华

本 PR 只有 1 条自动评论（来自 gemini-code-assist 的配额提醒），无 review 讨论，变更直达合并。

- 暂无高价值评论线程

风险与影响

- 风险：风险低。脚本使用正则替换特定模式，若文档格式变化可能匹配失败，但由于对方文件已知且流程最终会开放 PR 供人工审查，危险可控。脚本明确排除了 mutable 标签（latest/dev），不会错误覆盖。CI 工作流使用专门的 GH_PAT_FOR_PULL_REQUEST 令牌，权限隔离。
- 影响：用户侧：安装文档新增 nightly 选项，方便尝试最新特性；Docker 标签说明有助于用户选择正确部署版本。维护侧：发布新版本后不再需要手动修改文档版本，减少遗漏和手工

操作。系统侧：新增工作流仅在有 tag 推送时触发，几乎无额外运行成本。

- 风险标记：低风险自动化，依赖正则匹配，需人工审查 PR

关联脉络

- 暂无明显关联 PR