Gutenberg 插件发布

目录
快速参考
时间线
通用发布流程
步骤 1：准备工作
步骤 2：构建发布版本
步骤 3：编辑发布说明
步骤 4：撰写发布公告
额外的候选版本和次要版本 (X.Y.Z) 挑选错误修复
运行发布工作流
详细流程
发布周期
版本管理
准备发布
组织与标记里程碑 PR
运行发布工作流
发布 @wordpress 包到 NPM
查看发布草案
整理发布更新日志
创建候选发布补丁（ cherry-picking）
自动拣选
针对拥有推送权限的 "Gutenberg Core" 成员
非 "Gutenberg Core" 团队成员贡献者的替代流程
手动拣选
发布版本
发布故障排除
记录发布过程
选择版本亮点

发布素材
起草发布公告
发布版本公告
招募下个版本的志愿者
创建次要版本
更新发布分支
运行次要版本发布
何时选择 trunk：
何时选择发布分支：
为先前稳定版本创建次要版本
故障排除

快速参考

时间线

在里程碑日期（通常为周三）发布 RC1 版本
于下周三发布稳定版本

通用发布流程

步骤 1：准备工作

使用模板创建一个 发布议题 。此步骤可选，但有助于你按部就班地完成流程
审查里程碑中的所有 PR 并添加适当的标签（ [Type] Bug、 [Type] Enhancement 等）
测试更新日志： npm run other:changelog -- --milestone="Gutenberg X.Y"

步骤 2：构建发布版本

在 #core-editor Slack 频道中发布公告
前往 GitHub Actions → 构建插件 Zip 工作流
保持 Use workflow from 选项设置为 trunk（默认值）
输入 rc 表示候选发布版，或输入 stable 表示最终发布版
点击 Run workflow

当发布草稿在 GitHub Releases中生成后，将其发布以使工作流继续
仅适用于 RC1 版本 ：获取团队批准以将 @wordpress 包发布到 npm - 如有需要，请在 #core-editor中询问（ 详见 ）
仅适用于稳定版本 ：等待团队批准以上传到 WordPress.org -
这是将插件部署到插件目录的工作流最后一步（ 示例 ）

步骤 3：编辑发布说明

在 GitHub Releases中找到草稿
清理更新日志：修正拼写，移动分类错误的项目，合并相关的 PR
移除仅限移动端的更改和已还原的 PR

步骤 4：撰写发布公告

使用 Google 文档模板
突出展示本次发布的 3-5 个关键特性
在稳定版上线后，发布到 make.wordpress.org/core

额外的候选版本和次要版本 (X.Y.Z)
针对 RC1 后的紧急修复或主要版本间的关键错误修复：

挑选错误修复

对于新的候选版本：使用标记为 Backport to Gutenberg RC 的 PR
对于次要版本：使用标记为 Backport to Gutenberg Minor Release 的 PR
注意： 您必须是 Gutenberg 核心团队 的成员或与其中一员合作，才能挑选到发布分支
检出相应的发布分支： git checkout release/X.Y
运行： npm run other:cherry-pick "[Label Name]"（或请求 Gutenberg 核心团队 的成员运行）
在运行工作流 之前 ，将 PR 重新分配到正确的里程碑（例如，从 12.6 改为 12.5）

运行发布工作流

前往 构建插件压缩包工作流
如果下一版本不存在 RC 版本，请选择 trunk，否则从 Use workflow from
下拉菜单中选择发布分支
在文本输入框中输入 stable
继续执行上述通用发布流程中的步骤 2-4

详细流程

发布 Gutenberg 插件稳定版本的第一步是在 Gutenberg 仓库中 创建工单 。该工单模板名为 "Gutenberg Release"
，其中包含完整的发布流程清单，涵盖从发布候选版本到更新日志整理、代码拣选、稳定版发布以及发布公告
的全过程。 Gutenberg 21.2的工单就是一个很好的范例。

该清单能帮助您与开发人员及参与发布流程的其他团队协调工作。它确保所有必要步骤都得以完成，并让每个
人都清楚时间安排和重要里程碑。

发布周期

Gutenberg 的主要新版本大约每两周发布一次。当前版本和下一版本的状态及发布日期均在 GitHub 里程碑中追踪。

在当前里程碑日期 （亦称标记日）， Gutenberg 的首个候选版本（ RC
）将发布。这是插件的预发布版本，供插件开发者和用户测试。若发现任何回归问题，可发布新的候选版本。

候选版本采用递增编号，从 -rc.1 开始，随后是 -rc.2，依此类推。首个候选版本（ RC1
）发布后，即开始准备发布公告。

在 RC1 发布一周后
，将基于最新的候选版本及必要的回归修复创建稳定版本。稳定版本发布后，随即发布版本公告。

若在插件的稳定版本中发现严重错误，可随时发布补丁版本。

版本管理
每个主要的 Gutenberg 版本由一位版本管理员（也称为版本负责人）负责。该个人或小型团队在更广泛的
Gutenberg 开发团队 支持下，负责 Gutenberg 版本的发布。

版本管理员负责启动所有发布活动，任何对发布计划的更改都需要其批准。在紧急情况下或版本管理员无法联
系时，其他团队成员可以采取适当行动，但应随时告知版本管理员。

如果你是 Gutenberg 开发团队 的成员，并且有兴趣主导 Gutenberg 版本的发布，请在 #core-editor Slack 频道中联系我们。

准备发布
插件发布流程基本实现自动化，并在 GitHub
上完成。您无需在本地计算机上执行任何步骤。不过，建议您保留一份 Gutenberg 的本地副本，以便准备更新日志、进行常规测试，以及在需要多个候选版本时使用。具体细节将在后文详述。

这里有一段 11分钟的视频
，演示了插件发布流程。如果您对该流程不熟悉，建议先观看视频。以下段落也记录了该流程，并提供了更详
细的说明。

组织与标记里程碑 PR

快速参考
确保所有 PR 都已正确标记。
每个 PR 必须有一个以 [Type] 为前缀的标签。

准备 Gutenberg 版本的第一步是组织所有分配给当前 里程碑 的 PR，并确保每个 PR 都已正确标记。 标签用于自动生成更新日志，更改 PR 上的标签比之后在发布部分重新组织现有更新日志要快得多。

要测试将作为发布工作流一部分运行的更新日志自动化，您可以在本地的 Gutenberg
副本中使用您正在处理的稳定发布版本的里程碑运行以下命令：

npm run other:changelog -- --milestone="Gutenberg 16.2"

此命令的输出是所提供里程碑的更新日志，在上面的示例中是 Gutenberg 16.2。您可以将输出复制并粘贴到
Markdown 文档中，这将使其更易于查看，并允许您跟踪每个 PR 的链接。

所有 PR 都应有一个以 [Type] 为前缀的标签以及子类别标签。两个最常见的标签是 [Type] Bug 和 [Type] Enhancement 。在查看生成的更新日志时，请密切关注以下内容：

增强功能： 查找没有附加任何子类别的 PR。
错误修复： 同样查找没有附加任何子类别的 PR。
其他： 此部分中的 PR 没有任何标签。

根据需要更新每个 PR
的标签。您可以继续生成更新日志，直到您满意为止。现在您可以开始候选发布工作流了。

您可以在 changelog.js文件中查看如何根据 PR 标签生成更新日志。

运行发布工作流

快速参考

在 #core-editor中宣布你即将开始运行发布工作流。

运行 构建 Gutenberg 插件 Zip工作流。

在你开始之前，请在 #core-editor Slack 频道中宣布你即将开始运行工作流，并说明你是要发布 Gutenberg
的稳定版本还是 RC 版本。

然后转到 Gutenberg 仓库，点击 Actions 标签页，接着找到 构建 Gutenberg 插件 Zip操作。注意写着 “ 此工作流有一个 workflow_dispatch 事件触发器 ”的蓝色横幅。展开其右侧的 “Run workflow”下拉菜单。

插件发布的运行工作流下拉菜单

要发布插件的 RC 版本，请在文本框中输入 rc。要发布稳定版本，请输入 stable。在每种情况下，都按下
“Run workflow”按钮。

这将触发一个 GitHub Actions (GHA) 工作流，该工作流将提升插件版本号、构建 Gutenberg 插件.zip 文件、创建一个发布草稿并附加插件 .zip
文件。这部分过程通常需要大约六分钟。该工作流将出现在列表顶部，紧挨着蓝色横幅下方。一旦完成，工作
流的状态图标将从黄色圆点变为绿色对勾。你可以点击该工作流以获取更详细的视图来跟进进度。

发布 @wordpress 包到 NPM

作为发布流程的一部分，所有 @wordpress 包都会被发布到 NPM。在 构建 Gutenberg 插件 Zip

操作创建了草稿版本后，您可能会看到一条消息，提示 发布 npm 包 操作需要具有适当权限的人员来触发它。

Gutenberg 发布 、Gutenberg 核心 或 WordPress 核心 团队的成员必须 批准部署 。如果需要，请在 #core-editor频道中请求某人批准。此步骤仅适用于 RC1 版本。

查看发布草案

工作流完成后，您可以在 Gutenberg Releases
下找到发布草案。该草案已根据此版本之前的候选版本以及后续精选到发布分支的任何更改，预填了更新日志
条目。因此，在发布系列的第一个稳定版本时，请删除所有仅作参考的候选版本标题，并将最近的更改移至正
确的部分（见下文）。

整理发布更新日志

处理更新日志的最佳时机是在候选发布流程首次创建时。此时更新日志自动化流程被调用，第一版更新日志随
之生成。更新日志流程基本实现自动化，但正如前文所述，它高度依赖于里程碑中 PR的正确标签标注。

稳定版发布流程会收集各候选版本的更新日志并整合至稳定版中。但需特别注意：稳定版仅 "保留 " 首次生成的更新日志版本，即 RC1发布时可用的版本。后续对 RC1更新日志的任何修改都不会被纳入稳定版。

这意味着如果在发布 RC1前完成全部更新日志的整理工作，除后续 RC2或 RC3
版本中少量同样会加入稳定版的内容外，稳定版发布时就不需要再进行处理。

当发布更新日志草稿可用后，请花时间审阅并编辑说明内容，确保其易于阅读且准确无误。此环节不必匆忙完
成。重要的是尽可能保持发布说明的条理性，但无需一次性全部完成。您可以保存草稿稍后继续处理。

若担心在发布前他人无法获取候选版本，可通过 Slack频道 #core-editor
分享发布构件。这样在您完善更新日志期间，他人仍可访问候选版本。

以下是为准备清晰简洁更新日志的补充建议：

将 杂项分类下的所有条目移至更合适的分类
修正拼写错误并优化措辞。表述应易于理解，目标读者包括插件使用者及持续关注开发进展的人员
适时创建新分组，并在分组间移动 PR条目
当多个 PR涉及同一任务时（如后续跟进 PR），尝试合并为单一条目。典型案例如：为性能优化移除
Lodash的 PR、用 Playwright替代 Puppeteer端到端测试的 PR，或将公共组件转换为 TypeScript的 PR
若相关 PR集的子任务内容充实，可考虑采用嵌套列表进行组织
移除同一版本中因代码净变化为零而撤销其他 PR的 PR条目
移除所有仅更新移动端应用的 PR。唯一例外是移动端 PR同时涉及网页端功能更新
若子标题下仅列出一个 PR，则删除该子标题并将 PR移至下一个包含多项条目的匹配子标题下

创建候选发布补丁（ cherry-picking）

快速参考
确保所有需要 cherry-pick 的 PR 都已添加 Backport to Gutenberg RC 标签。
在你的 Gutenberg 仓库本地克隆中，切换到发布分支： git checkout release/X.Y
使用自动化脚本 cherry-pick 所有已合并的 PR： npm run other:cherry-pick "Backport to
Gutenberg RC"

在候选发布版本（ RC）发布后、最终稳定版发布前，一些与发布相关的错误可能会被修复并提交到 trunk 分支。稳定版不会自动包含这些修复。将它们包含进来是一个手动过程，称为 cherry-picking。

作为发布经理，你可能通过以下几种方式了解到这些错误：

贡献者可能会给已关闭的 PR 添加 Backport to Gutenberg RC 标签。在发布最终版本前，
请搜索所有此类 PR。
你可能会在 #core-editor Slack 频道中收到直接消息或提及，通知你需要包含在 RC 中的 PR
。即使在这种情况下，也应始终为 PR 添加 Backport to Gutenberg RC 标签。

自动拣选

拣选流程可通过 Gutenberg 内置的 npm run other:cherry-pick "[Insert Label]"
脚本实现自动化。执行命令时需使用 Backport to Gutenberg RC 标签，并确保所有需要拣选的 PR
都已分配该标签。

执行 PR 拣选操作时，必须克隆（而非分叉） Gutenberg 代码库并具备写入权限。仅 Gutenberg
核心团队 成员拥有直接向发布分支推送的权限。

针对拥有推送权限的 "Gutenberg Core" 成员

将 Gutenberg 仓库克隆到本地开发环境后，首先切换到发布分支：

git checkout release/X.Y

接下来，挑选所有带有相应反向移植标签的已合并 PR：

npm run other:cherry-pick "Backport to Gutenberg RC"

该脚本在后台将执行以下操作：

挑选所有带有 Backport to Gutenberg RC 标签的 PR
将它们添加到发布里程碑
将所有更改 git push 到发布分支
在 PR 上添加一条评论，说明它已被挑选
从 PR 中移除 Backport to Gutenberg RC 标签

以下是该过程的截图：

自动化挑选过程

非 "Gutenberg Core" 团队成员贡献者的替代流程

如果您没有直接推送到发布分支的写入权限，可以 从您的复刻仓库 使用此替代方法：

1. 确保您已复刻 Gutenberg 仓库并添加上游远程：

```
git remote add upstream https://github.com/WordPress/gutenberg.git
git fetch upstream
```

2. 基于上游发布分支创建新分支：

```
git checkout -b backport-fixes-X.Y.Z upstream/release/X.Y
```

3. 手动按时间顺序挑选每个 PR 提交：

```
git cherry-pick [SHA]
```

（自动化脚本在复刻仓库中无法工作，因为它需要主仓库的推送权限）

4. 将您的分支推送到您的复刻仓库：

```
git push origin backport-fixes-X.Y.Z
```

5. 从您的复刻仓库创建拉取请求，目标指向主仓库的 `release/X.Y` 分支，并请求 Gutenberg Core 团队成员进行审核
6. 一旦批准并合并，请与 Gutenberg Core 团队成员或发布负责人协调，以继续发布流程

或者，您可以在 #core-editor Slack 频道中请求 Gutenberg Core 团队成员为您运行 cherry-pick 命令。

手动拣选

若需逐条逐步骤处理拣选操作，可手动按此流程执行。在检出对应发布分支后：

1. 按时间顺序使用 `git cherry-pick [SHA]` 拣选每个 PR
2. 完成后通过 `git push` 推送更改至 GitHub
3. 移除所有已拣选 PR 的 `Backport to Gutenberg RC` 标签，并将其里程碑更新至当前版本

要查找拉取请求的 `[SHA]`，请打开对应 PR，在页面底部附近可见“`[用户名]` 将提交 `[SHA]` 合并至 `trunk`”的提示信息。

手动拣选

若拣选的修复内容值得在稳定版发布前创建新的候选版本，请按上述说明创建新候选版。并在 Slack 频道 #core-editor 告知其他贡献者新候选版已发布。

发布版本

> 快速参考
>
> - 在版本草稿中，点击“发布版本”按钮。
> - 如果发布 RC1，请在构建 Gutenberg 插件 Zip 工作流中批准 npm 发布任务。

如果发布稳定版本，需要获得 Gutenberg 发布 、Gutenberg 核心 或 WordPress 核心
团队中一名成员的批准，才能将新插件版本上传到 WordPress.org 插件仓库 (SVN)。
上传后，请确认可以从 WordPress 插件仪表盘下载和更新到最新版本。

只有当你对版本草稿中的更新日志内容感到满意后，才能点击 “发布版本 ”按钮。

请注意，你不需要更改按钮上方的复选框。如果你发布的是 RC 版本， “设置为预发布版本 ”
将自动被选中；如果你发布的是稳定版本， “设置为最新版本 ”将被选中。

发布 RC 版本时的复选框

发布版本将为该版本创建一个 git 标签，发布版本，并触发 另一个 GHA 工作流 ，其目的有两个：

1. 使用你刚刚编辑的发布说明来更新 changelog.txt，并且
2. 将新插件版本上传到 WordPress.org 插件仓库 (SVN)（仅当你发布的是稳定版本时）。

最后一步需要 Gutenberg 发布 、Gutenberg 核心 或 WordPress 核心
团队中任一成员批准。当版本准备好批准时，这些团队会收到通知邮件，但如果时间紧迫，你可以在 #core-editor Slack 频道中询问或联系 Gutenberg 发布团队
以加快流程。建议在启动发布流程之前就联系好，以便有人准备好批准。找到新版本的 “将 Gutenberg
插件上传到 WordPress.org 插件仓库 ”工作流 ，并让其 获得批准 。

一旦获得批准，新版古腾堡将面向全球 WordPress 用户开放。上传后，请确认可以从 WordPress
插件仪表盘下载并更新至最新版本。

最后一步是在 make.wordpress.org/core上撰写发布公告。你可以在下方找到相关建议。

发布故障排除

插件已发布到 WordPress.org 插件目录，但工作流失败。 “

这种情况偶尔会发生，例如 这个案例 。

请务必检查：

目录中的插件是否按预期工作
ZIP 文件内容（参见 下载 ）看起来是否正确（没有明显缺失的内容） Gutenberg SVN 仓库 是否有两个新提交（参见 日志 ）：

- `trunk` 文件夹应有提交信息 "Committing version X.Y.Z"
- 存在一个新的 `tags/X.Y.Z` 文件夹，其内容与 `trunk` 相同，其最新提交信息为 "Tagging version X.Y.Z"

最可能的情况是标签文件夹无法创建。这是一个 已知问题，可以 手动修复。

请将 `SVN_USERNAME`、`SVN_PASSWORD` 和 `VERSION` 替换为正确的值，或先将它们设置为全局环境变量：

```
# 检出仓库
svn checkout https://plugins.svn.wordpress.org/gutenberg/trunk --username "$SVN_USERNAME" --password "$SVN_PASSWORD" gutenberg-svn

# 进入本地文件夹
cd gutenberg-svn

# 如果您本地已有仓库
# 且未检出，请确保它已更新
svn up .

# 将当前主干复制到新的标签文件夹
svn copy https://plugins.svn.wordpress.org/gutenberg/trunk https://plugins.svn.wordpress.org/gutenberg/tags/$VERSION -m 'Tagging version $VERSION' --no-auth-cache --non-interactive  --username "$SVN_USERNAME" --password "$SVN_PASSWORD"
```

如果您在任何步骤需要帮助，请向周围询问。

记录发布过程

发布记录由发布经理主导，并在 Gutenberg 开发团队 成员的协助下完成。此过程包含一系列顺序步骤，由于涉及人员众多且需要协调，必须在 RC 版本和稳定版本之间的时间线内完成。 Gutenberg 稳定版本于每周三发布，即首个 RC 版本发布一周后。

> 时间线
>
> 1. 复制 发布文章的 Google 文档模板 – 周三至周五
> 2. 选定发布亮点 – 周五至周一
> 3. 亮点确定后，向设计团队请求发布素材（图片、视频） – 周五至周一
> 4. 起草发布文章并请求同行评审 – 周一至周三
> 5. 稳定版本发布后发布文章 – 周三

选择版本亮点

清理完更新日志后，下一步是挑选一些变更作为发布帖的亮点。这些亮点通常聚焦于新功能和改进，包括性能
和可访问性方面的提升，但也可以包含重要的 API 变更或关键错误修复。

鉴于 Gutenberg 项目范围庞大且每个里程碑合并的 PR
数量众多，有时会忽略值得强调的重要变更；因此，这一步需要版本管理员与其他 Gutenberg 开发团队成员协作完成。如果您不确定如何选择，请向团队中的其他成员寻求帮助。

发布素材

这篇发布文章包含若干需要整理的视觉素材。文章的特色图片请沿用上一版本使用的相同图片，该图片应已存
在于媒体库中，名为 'gb-featured'。

文章正文中还有一个横幅，可通过名为 'Gutenberg What's New Banner'
的同步模式添加。插入此模式并将文本更新为正确的版本号。

重点功能也需要视觉素材。对于重要功能，你可以向设计团队申请视觉素材；对于其他功能，如果你有把握也
可以自行制作。如需向设计团队申请素材，请在 #design Slack 频道中提出， 15.8 版本的示例文章可参考
此处 。素材将提供在分配给特定版本的 Google Drive 文件夹 中。

为 WordPress 版本创建视觉素材时，请使用动画（视频或 GIF）或静态图片来展示亮点。可参考 往期发布文章作为指南，注意动画更适合演示工作流程，而更直接的亮点则可用图片展示。创建素材时请避免使用受版权保
护的内容，并禁用浏览器画布中可能显示的插件。

起草发布公告

发布经理负责根据 谷歌文档模板
起草发布公告。也就是说，鉴于发布公告内容的性质，如果事先达成一致，可以将职责划分并委派给其他团队
成员。草案完成后，请进行同行评审。

发布版本公告

当公告内容准备就绪后，拥有 make.wordpress.org/core 发布权限的作者将创建新草稿并导入内容。公告应包含以下标签：

#block-editor
#core-editor

#gutenberg

#gutenberg-new

随后，作者应启用公告的公开预览功能，并请求最终同行评审。这符合 make/core 发布指南 的鼓励做法。

最后，公告应在稳定版本发布并可在 WordPress.org
上获取后发布。这将有助于外部媒体传播和扩大版本影响力。

招募下个版本的志愿者
完成当前版本发布后，请在 #core-editor Slack 频道中发布消息，招募志愿者负责下一个 Gutenberg
版本的发布工作。

可参考 此处的示例 。

创建次要版本
有时需要为插件创建次要版本（即 X.Y.Z）。这通常是为了快速修复严重的回归问题或错误。 Backport to Gutenberg Minor Release 标签通常用于标识需要包含在次要版本中的 PR，但作为发布协调员，您也可能通过
Slack 收到非正式通知。即便如此，最好确保所有相关 PR 都已正确标记。

在执行以下流程时，值得注意的是，此类次要版本不会作为独立分支创建（例如 release/12.5.0），而仅仅是

标签 。

次要版本的发布方法与主要插件发布流程几乎相同（见上文），但有一些显著例外。请务必在继续之前阅读
完整 指南。

更新发布分支

次要版本发布应仅包含 必需的特定提交 。为此，你需要在本地检出上一个 主要 稳定（即非 RC
）发布分支（例如 release/12.5），然后精选所需提交到该分支。

如果新版本已存在 RC，你 必须
在相应的发布分支中精选相同的提交，因为它们不会自动包含。例如：如果你即将为 12.5
发布新的次要版本，并刚刚精选到 release/12.5，但 12.6.0-rc.1
已经发布，那么你需要将相同的提交精选到 release/12.6 分支，否则它们将不会包含在 12.6
的后续发布中！通常最好与下一个版本的发布协调员协调此过程。

精选过程可以通过 npm run cherry-pick 脚本自动化，但运行脚本时务必使用 Backport to Gutenberg Minor Release 标签。

你还必须确保所有包含的 PR 都已分配到次要版本所基于的 GitHub 里程碑。请注意，当 PR 被 合并
时，它们会自动分配到下一个 稳定 版本的里程碑。因此，你需要返回 GitHub 中的每个 PR
并重新分配里程碑。

例如，如果你正在发布版本 12.5.4，那么为该发布精选的所有 PR 必须从 12.6
里程碑取消分配，并重新分配到 12.5 里程碑。

精选完成后，你还可以从 PR 中移除 Backport to Gutenberg Minor Release 标签。

一旦稳定发布分支准备就绪且 PR 分配了正确的里程碑，你就可以 将分支推送到 GitHub，并使用 GitHub 网站
GUI 继续发布流程。

运行次要版本发布

插件发布的运行工作流下拉菜单

前往 Gutenberg 的 GitHub 仓库的 Actions 标签页，找到 "构建 Gutenberg 插件 Zip" 操作 。

重要提示： 在 "Use workflow from"
下拉菜单中选择的分支决定了工作流将使用哪个发布分支来创建次要版本。

何时选择 trunk：

如果 上一个发布版本是稳定版 （例如 12.5.0、 12.5.1）且 下一个主要版本不存在 RC 版本 （例如没有 12.6.0-rc.1 ）：

保持 Use workflow from 为 Branch: trunk
在文本输入框中输入 stable
工作流将自动递增补丁版本号（例如 12.5.1 → 12.5.2）

何时选择发布分支：

如果 下一个主要版本已存在 RC 版本 （例如，存在 12.6.0-rc.1 而你需要发布 12.5.1）：

从 Use workflow from 下拉菜单中选择稳定发布分支（例如 release/12.5）
在文本输入框中输入 stable
否则工作流将发布下一个主要稳定版本而非次要版本

为先前稳定版本创建次要版本

即使已发布更新的稳定版本，仍可为任何发布分支创建次要版本。此操作适用于 所有
先前的发布分支，从而为用户提供更灵活的更新交付方式。过去用户可能需要等待数天才能获得下一个稳定版
本，现在则可根据需要快速将修复推送到任何先前的发布分支。

该流程与上文所述存在已发布候选版本时完全相同：选择先前的发布分支，输入 stable，点击 "运行工作流 "
。版本将发布至 Gutenberg的GitHub发布页面，同时以 tag形式推送至 WordPress核心代码库 SVN的
https://plugins.svn.wordpress.org/gutenberg/tags/。SVN的 trunk目录将保持不变。

重要提示： 发布由 "构建插件压缩包 "工作流 创建的草稿时，请务必 取消勾选 "设为最新版本 "

复选框。若意外保持勾选状态， "上传 Gutenberg插件至 WordPress.org插件库 "工作流 仍会正确将其
作为标签（且不会替换 trunk版本） 上传至 WordPress插件仓库 SVN——
该工作流将通过版本号运算确定插件交付方式 ——但您仍需在 GitHub的发布页面 将正确版本设为 latest
以修复状态！

故障排除

发布草稿已创建但内容为空 /包含错误信息 “

如果您忘记为精选的 PR 分配正确的里程碑，则变更日志可能无法按预期生成。

务必始终手动验证变更日志中显示的 PR 是否与精选到发布分支的 PR 一致。

此外，如果发布仅包含单个 PR，则未能将该 PR
分配到正确的里程碑将在生成变更日志时导致显示错误。在这种情况下，您可以编辑发布说明以包含缺失 PR 的详细信息（手动复制先前发布的格式）。

如果由于任何原因里程碑已关闭，您可以为了发布目的重新打开它。

发布草稿仅包含 1 个资产文件。其他发布有 x3 个。 “

这是正常现象。发布草稿将仅包含插件 zip 文件。只有在发布后，其余资产才会生成并添加到发布中。

我是否需要将点版本发布到 WordPress.org？ “

是的。方法与主插件发布过程完全相同。您需要 Gutenberg 核心团队或 Gutenberg 发布团队的成员来批准发布工作流。

发布过程未能将版本更新提交精选到主干分支。 “

首先，通过检查 trunk
上的最新提交是否不包含版本更新提交来确认该步骤失败。然后在发布分支上还原版本更新提交 - git revert
- - no-edit {commitHash}。最后，推送更改并重新启动发布过程。

版本号 #2
由 壳壳虫 创建于 2026-03-27 01:08:43 UTC
由 壳壳虫 更新于 2026-03-27 01:08:43 UTC