修改文档

读者: 任何愿意为Fabric文档做出贡献的人.

这个简短的指导手册说明了Fabric文档是如何组织，编译及发布的，任何人在修改Fabric文档前，需要了解一些规约。

在本章中，我们会介绍：

文档简介

Fabric文档是由Markdown和reStructuredText的组合编写的，作为一个新的作者，您可以任选一个或者两者都用。我们建议您从Markdown开始，因为它既简单又强大； 如果您有Python基础，您可能会倾向于使用RST。

作为编译过程的一部分，文档源文件会被Sphinx转换为HTML，并发布在这里。 我们为Fabric仓库配置了一个GitHub勾子，任何 `docs/source` 内的新增或修改内容都会触发一个新的文档的编译和发布。

Fabric文档提供各种不同语言的翻译：

- 中文
- 马拉雅拉姆语
- 巴西葡萄牙语 – 即将发布

它们都是在Hyperledger Labs 组的各自语言的仓库中编译的。

例如：

- 中文仓库
- 马拉雅拉姆仓库
- 巴西葡萄牙语仓库 – 即将发布

一旦一个语言仓库接近完成，它就能被贡献到Fabric主发布站上。例如,中文翻译在主文档站.

仓库结构

对于每个仓库,Fabric文档总是将`/docs`作为顶级文件夹并存放在它之下。

```
(docs) bash-3.2$ ls -l docs
total 56
-rw-r--r--     1 user  staff   2107  4 Jun 09:42 Makefile
-rw-r--r--     1 user  staff    199  4 Jun 09:42 Pipfile
-rw-r--r--     1 user  staff  10924  4 Jun 09:42 Pipfile.lock
-rw-r--r--@    1 user  staff    288  4 Jun 14:50 README.md
drwxr-xr-x     4 user  staff    128  4 Jun 10:10 build
drwxr-xr-x     3 user  staff     96  4 Jun 09:42 custom_theme
-rw-r--r--     1 user  staff    283  4 Jun 09:42 requirements.txt
drwxr-xr-x   103 user  staff   3296  4 Jun 12:32 source
drwxr-xr-x    18 user  staff    576  4 Jun 09:42 wrappers
```

顶级文件夹里的文件都是为编译过程准备的配置文件。所有文档文件都存放在`/source`文件夹下:

```
(docs) bash-3.2$ ls -l docs/source
total 2576
-rw-r--r--    1 user  staff   20045  4 Jun 12:33 CONTRIBUTING.rst
-rw-r--r--    1 user  staff    1263  4 Jun 09:42 DCO1.1.txt
-rw-r--r--    1 user  staff   10559  4 Jun 09:42 Fabric-FAQ.rst
drwxr-xr-x    4 user  staff     128  4 Jun 09:42 _static
drwxr-xr-x    4 user  staff     128  4 Jun 09:42 _templates
-rw-r--r--    1 user  staff   10995  4 Jun 09:42 access_control.md
-rw-r--r--    1 user  staff     353  4 Jun 09:42 architecture.rst
-rw-r--r--    1 user  staff   11020  4 Jun 09:42 blockchain.rst
-rw-r--r--    1 user  staff   75552  4 Jun 09:42 build_network.rst
-rw-r--r--    1 user  staff    9115  4 Jun 09:42 capabilities_concept.md
-rw-r--r--    1 user  staff    2851  4 Jun 09:42 capability_requirements.rst
...
```

这些文件和目录被映射在发布文档中。 特别指出，目录以`index.rst`作为根文件并链接到了`/docs/source`下的每一个其它文件。

您可以花些时间来查看一下这些目录和文件，了解一下它们是如何连接在一起的。

要更新这些文档，您可以使用git更新一个或更多文件，在本地编译这些更新来查看是否正常，然后向Fabric主仓库提交一个PR。如果更新被维护者接收了，它就会被合并到Fabric主仓库并成为发布文档的一部分。就这么简单!

您可以在这里学习如何提交一个PR，不过在此之前，请首先继续读下去来学习如何在本地编译您的更改。此外，如果您是git和GitHub的新手，Gitbook将会非常有用。

测试您的修改

在提交一个PR前，我们强烈建议您测试您的修改。您应该先在您的机器上编译文档，然后将您的修改推送到您自己的GitHub阶段仓库中，它们可以迁移到您的发布网站ReadheDocs上。一旦您对您的修改感到满意，您就可以通过发起一个PR把它纳入到Fabric主仓库中。

下述章节包含了如何在本地编译文档，并使用您自己的Github分支发布在ReadTheDocs上。

本地编译文档

一旦您将Fabric repository克隆到您的本机，使用这简单的几步来在本机编译Fabric文档。注意：您可能需要根据您的操作系统来做一些调整。

前置需求:

- Python 3.7
- Pipenv

```
cd fabric/docs
pipenv install
pipenv shell
make html
```

这会在 `docs/build/html` 中生成所有Fabric文档的html文件，您可以在您自己的浏览器中查看；您可以从 `index.html` 进行导航。

对一个文件做些小修改，然后重新编译文档来确认您的修改是否已在本地编译。当然，您的对文档的每一次修改都需要重新运行 `make html` 。

另外，如果您愿意的话，您可以使用如下命令来启动一个本地web服务器（或者根据您的操作系统使用其它替代）：

```
sudo apt-get install apache2
cd build/html
sudo cp -r * /var/www/html/
```

然后您可以通过 `http://localhost/index.html` 访问html文件。

在GitHub上编译文档

使用您自己的Fabric仓库分支编译Fabric文档总是有用的，它是公开的，可以被其它人看见。下述方法将告诉您如何使用ReadTheDocs做到这些。

1. 到http://readthedocs.org注册一个账号。
2. 创建一个工程。您的用户名非常适合作为URL地址，并且您可以在后面添加 `-fabric` 以将其和其它工程的文档做出区分。例如： `YOURGITHUBID-fabric.readthedocs.io/en/latest` 。
3. 点击 `Admin` ,点击 `Integrations` ,点击 `Add integration` ,选择 `GitHub incoming webhook` ,然后点击 `Add integration` 。
4. Fork Fabric on GitHub。
5. 在您的分支中，在右上角找到 `Settings` 。
6. 点击 `Webhooks` 。
7. 点击 `Add webhook` 。
8. 将ReadTheDocs的 URL地址添加到 `Payload URL` 。
9. 选择 `Let me select individual events` : `Pushes` 、 `Branch or tag creation` 、 `Branch or tag deletion` 。
10. 点击 `Add webhook``Add webhook` 。

现在一旦您在您的分支上修改或新增了任何文档内容，这个URL地址都会自动更新您的改动！

更新命令参考

更新命令参考中的文件需要额外的步骤。因为命令参考章节中的内容是生成出来的，您不能仅仅更新关连的markdown文件。

- 您需要更新`src/github.com/hyperledger/fabric/docs/wrappers`下的`_preamble.md`或`_postscript.md`文件。
- 更新命令帮助需要您编辑命令关连的`.go`文件，在`/fabric/internal/peer`文件夹下。
- 然后，对于`fabric`文件夹，您需要运行`make help-docs`来生成更新过的markdown文件，它们在`docs/source/commands`文件夹下。

记住，当您将变更push到GitHuB时，您需要将修改过的`_preamble.md`，`_postscript.md`或`_.go`文件如同生成的markdown文件一样包含在内。

这个过程仅适用于英语版本翻译。命令参考翻译目前不支持国际语言。

增加一个新CLI命令

添加一个新的CLI命令，请执行如下步骤：

- 在`/fabric/internal/peer`下为新命令创建一个新的文件夹并添加帮助文本。您可以将`internal/peer/version`作为一个简单的例子来帮助您开始。
- 在`src/github.com/hyperledger/fabric/scripts/generateHelpDoc.sh`中为您的CLI命令增加一个章节。
- 在`/src/github.com/hyperledger/fabric/docs/wrappers`中创建如下两个新文件:
  - `<command>_preamble.md` (命令名和语法)
  - `<command>_postscript.md` (样例)
- 运行`make help-docs`生成markdown内容然后将所有变更的文件push到GitHub。

这个过程仅应用于英语版本翻译。CLI命令翻译目前不支持国际语言。