Building a New Docs Release

We use a number of steps to build a new release while also maintaining docs for one or more previous releases:

1. Build the PDF and then the web for the previous version(s).
2. Upload the built previous version web (which includes the PDF) to a NewVersion subdirectory on splicemachine.com/<prev>, where <prev> is the version number (e.g. 2.5).
3. Take that version live, using a script on the server.
4. Build the PDF and then build the web for the current version.
5. Upload the built current version web to a NewVersion subdirectory on splicemachine.com.
6. Take that version live, using a server script.
7. Tell SwifType to re-crawl the web
8. Make sure all is well.

In the next sections, we outline these steps in detail:

- Build the Previous Release First
- Next, Build the Current Release
- Update Search

When creating a new "previous release" directory, you must first prepare that directory with a few files; this is described here

- Build the Previous Release First
- Next, Build the Current Release
- Update Search

A Note About Serve vs Build:

We use Jekyll to build the docs system, and we use a script we've created, `servesplice`, to manage building different versions (doc or docstest or docsdev) and different releases (e.g. 2.7, 2.5). There's an issue you should be aware of:

- You can tell Jekyll to build or serve a docs system. When it serves, Jekyll does the build, then runs the built web locally, on http://127.0.0.1:4000. This is how we examine and test our changes during the development cycle.
- When Jekyll runs the web locally, you need to use Ctrl-C to terminate the run.
- When building a production version, our `servesplice` script needs to rename and/or copy some files after the build completes. However, if you serve the web and then Ctrl-C to terminate the run, those steps are not run, because the script has been cancelled.
- The solution: when building a production version, tell Jekyll to build the system, and when building a development version, tell Jekyll to serve the system.

The `servesplice` script handles this by asking whether you want to serve or build the system.

Build the Previous Release First:

1. In Git:
    a. Switch to the `2.5` branch of the `splicedocs-jekyll` repo

2. Using a text editor such as Atom:
    a. Open the docs project
    b. Update any values that need updating in the `_includes/splicevars.html` file.

3. In Terminal:
    a. `cd` to `splicedocs-jekyll` directory (alias=`splicedocs`)
    b. Use the `./servepdfsuite` script to build the PDF version of `2.5`
    c. Use the `./servesplice` script to **build** the `2.5` version of `doc`: when asked to serve or build, type in `build`.

4. In an FTP client:
    a. Connect to `doc.splicemachine.com/2.5`
    b. Create a directory in there named `NewVersion`
    c. Transfer the contents of `splicedocs-jekyll/_site/` to `NewVersion/`

5. In a browser:
    a. Point to `https://doc.splicemachine.com/2.5/NewVersion` to verify the upload

6. In Terminal:
    a. `ssh` into the splicemachine server (alias=`spliceputty`)
    b. `cd subs/doc/2.5`
    c. Use `ls -d ./*/` to verify that subdirectory `NewVersion` exists
    d. Use `./UpdateVersion.sh` to backup the current version of 2.5 and take the contents of `/NewVersion` live.

7. In a browser:
    a. Point to `https://doc.splicemachine.com/2.5` to verify all is well

8. In Git (if you made any changes to the source):
    a. Commit & push those changes to the repo branch.

NOTE: You must create and structure the previous release dir on the server before using this process. See the final section of this topic for instructions.

Next, Build the Current Release:

1. In Git:
   a. Switch to the `master` branch of the `splicedocs-jekyll` repo

2. Using a text editor such as Atom:
   a. Open the docs project
   b. Update any values that need updating in the `_includes/splicevars.html` file.

3. In Terminal:
   a. cd to `splicedocs-jekyll` directory (alias=`splicedocs`)
   b. Use the `./servepdfsuite` script to build the PDF version of `2.7`
   c. Use the `./servesplice` script to **build** the `2.5` version of `doc`: when asked to serve or build, type in `build`.

4. In an FTP client:
   a. Connect to doc.`splicemachine.com`
   b. Create a directory in there named `NewVersion`
   c. Transfer the contents of `splicedocs-jekyll/_site/` to `NewVersion/`

5. In a browser:
   a. Point to `https://doc.splicemachine.com/NewVersion` to verify the upload

6. In Terminal:
   a. `ssh` into the splicemachine server (alias=`spliceputty`)
   b. `cd subs/doc`
   c. Use `ls -d ./*/` to verify that subdirectory `NewVersion` exists
   d. Use `./UpdateVersion.sh` to backup the current version and take the contents of `/NewVersion` live.

7. In a browser:
   a. Point to `https://doc.splicemachine.com` to verify all is well

8. In Git (if you made any changes to the source):
   a. Commit & push those changes to the repo branch.

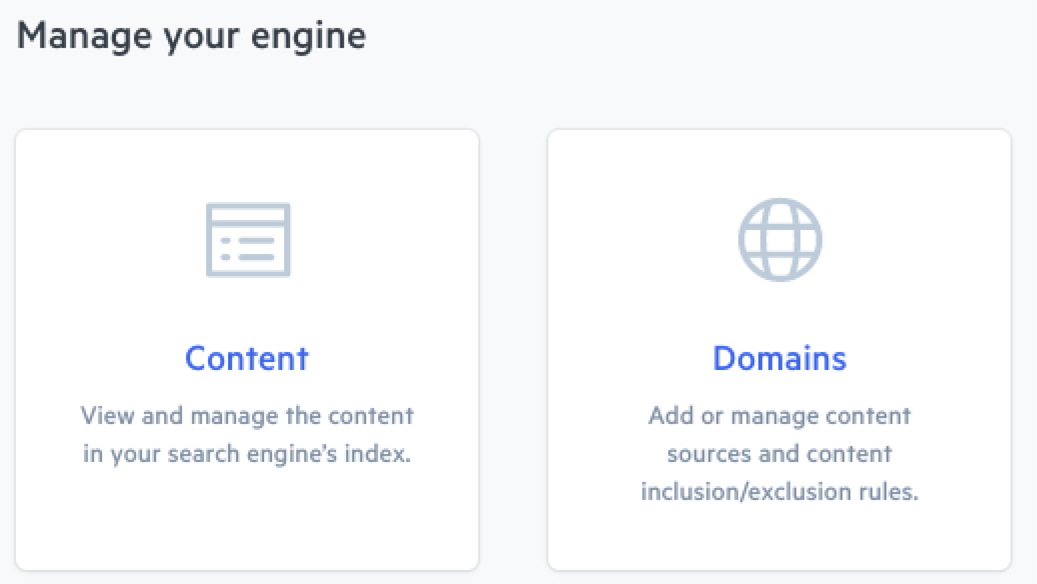
NOTE: You may want to create a new branch for the new release.

Update Search

After updating the docs on the server, you must tell Swiftype Site Search to re-index the site:

1. Point your browser to the Splice Machine Swiftype Dashboard: https://app.swiftype.com/engines/splicedocs/overview

2. Click the large Domains button in the Dashboard:



Manage your engine

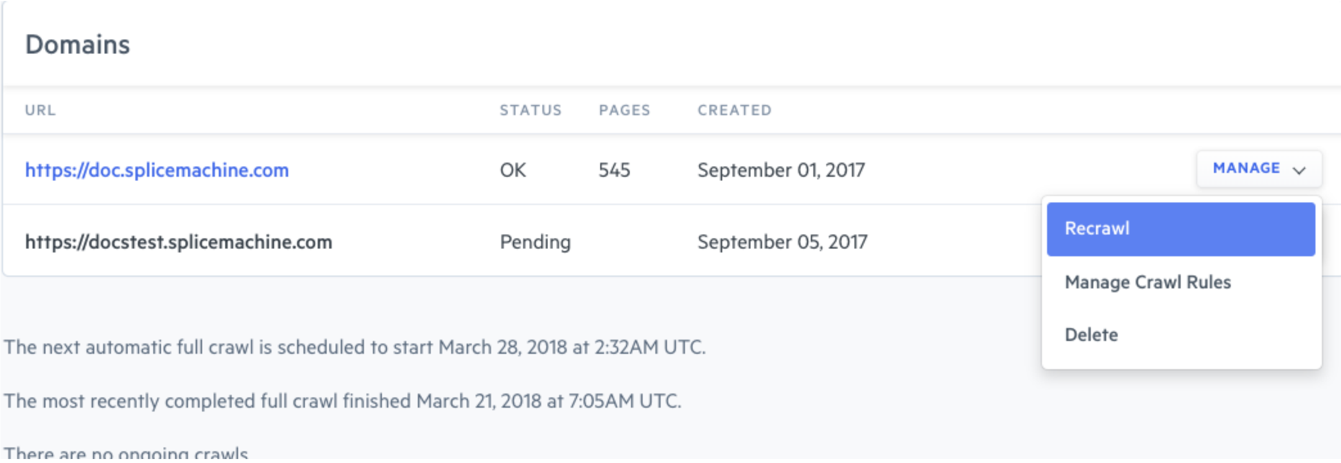
Content

View and manage the content in your search engine's index.

Domains

Add or manage content sources and content inclusion/exclusion rules.

3. Click the Manage down-arrow and select ReCrawl:



Domains

URL	STATUS	PAGES	CREATED
https://doc.splicemachine.com	OK	545	September 01, 2017
https://docstest.splicemachine.com	Pending		September 05, 2017

The next automatic full crawl is scheduled to start March 28, 2018 at 2:32AM UTC.

The most recently completed full crawl finished March 21, 2018 at 7:05AM UTC.

There are no ongoing crawls

4. Check back on this page after 4-8 hours. It will show the most recently created crawl time.

5. Once the crawl has completed, navigate to the site and verify that search is working.

NOTE: Re-indexing the doc.splicemachine.com site can require 6-12 hours; this time increases as additional previous-release subdirectories need to be indexed. While Swiftype is recrawling the site, search results may be suspects. We usually initiate these recrawls after 6pm.

Preparing a New "Previous Release" Directory

If you need to create a new "previous release" directory on the server, that directory must be pre-populated with several files before you upload the docs there.

For example, if Splice Machine releases version 2.8, and wants to maintain the documentation for customers on 2.5 and 2.7, we would need to create and pre-populate a 2.7 directory.

You can prepare the directory by simply following these steps:

1. Ssh into doc.splicemachine.com

2. Issue these shell commands:

   ```
   a. mkdir 2.7
   b. shopt -s extglob
   c. cp -r !(2.7) 2.7/
   d. shopt -u extglob
   ```

3. Edit the `UpdateVersion.sh` script, removing any lines (near the top) that move previous release versions into the `Temp` directory. For example, you would remove this line:

   ```
   mv 2.5 Temp/
   ```

You must also make one change in the UpdateVersion.sh script that's located in the top-level doc.splicemachine.com:

1. Edit the `UpdateVersion.sh` script, adding a line to move your new prev-release directory to `Temp`. For example, you would add a line for 2.7, so the UpdateVersion.sh script now would contain:

   ```
   mv 2.5 Temp/

   mv 2.7 Temp/
   ```

Now you'll be able to FTP updated versions of the 2.7 docs to your new directory and use the UpdateVersion.sh script to take that update live.