DoctorPepper README

jnorthr

Version 1.1, Updated 17 Apr.2017

Table of Contents

Status

Travis C/I Project Health - [Build Status (Travis CI)] [Apache License]

Purpose

This is an open-source tool to continuously translate any asciidoctor file that is changed within a defined folder.

DropBox or Google Drive Enabled ?

This trick works nicely if you have a shared drive service like **Dropbox**, or **Google Drive** etc. and all your internal systems use the same **Dropbox**,etc. account.

I have six systems in my place. All of them hooked to same remote drive account. This way i can work on any of my internal systems and still use/keep documentation and code safely and visibly on **all** my systems.

On one of my least-used systems, did the following:

Gradle Continuous Doctor Translation

```
cd ~/Dropbox
git clone https://github.com/jnorthr/DoctorPepper.git
cd ~/Dropbox/DoctorPepper
gradlew -t asciidoctor
```

The **-t** option keeps gradle running in continuous mode so now any changes i make to any asciidoctor files are near-instantly translated for me.

If you don't have a **git** client click here to install one

Update from A Different System

Went to a second system in my place. Updated this **Dropbox/DoctorPepper/src/docs/asciidoc/sample.adoc** file with:

More stuff here. Did this `sample.html` update from another machine show up on my **Dropbox** account?

Will a new **sample.html** show up on this machine too ?

NO ! Why not ? Because screen saver on my gradle system put it to sleep !

Don’t Allow Sleeping on The Job

If the server or desktop running your `gradlew -t asciidoctor` continuous process has it’s screen saver turned on, then you will only have continuous doctor translation while it does not sleep.

With full-time running, my asciidoctor translation process will nicely convert any/every change i make to my **.adoc**,etc. documents within the **src/docs/asciidoc** folder.

Goal

What does this achieve ? Well anywhere on any of my six systems, i can modify my documentation and the *kind-of* remote server will produce a new `**.html** set of results. Then i can have one or several browsers viewing the results of files i’m editing, say in folder **DoctorPepper/build/docs** for my revised **sample.html**.

While doing this page, i had my text editor open, fixed some spelling mistakes and only did an editor save, like **Ctrl-S** and toggled over to my window with the browser open to this **sample.html** page and did a browser re-fresh! **Bang!**

Near-instant Translation

Then i only click browser refresh to see the new view and this saves me the bother of copying everything up to my CloudFoundry target. **Nice ;-)**

Just copying new **.adoc** files into **src/docs/asciidoc** will appear in the output folder too !

Table of Contents / Summary Page

While working on this tool, decided to write some groovy to generate an asciidoctor file named **toc.adoc** and this is written into the same folder that the asciidoctor task is watching. The next time this task runs, the **toc.adoc** is magically turned into an **toc.html** page with links to every `*.html` within the watched folder or sub-folders.

Surprise !

Was **surprised** when i accidentally ran **groovydoc** task to produce some API doc.s for my `Walker.groovy` code, and low-and-behold the next cycle of our **asciidoctor** task included all my **groovydoc**, **javadoc**, and test reports in this **toc.adoc** summary !

Even More

Since i've added **Jacoco** code coverage tool, to see a full-blown example of `walker` in action try this:

Generate a Bunch of HTML, Writing Table of Contents for It and Convert to HTML

```
gradlew build  jacocoTestReport groovydoc test walker asciidoctor
```

After this, there should be a fully-loaded `/Users/jimnorthrop/Dropbox/Projects/DoctorPepper/build/docs/toc.html`

`Walker` Task in Gradle

Several ways can be used to execute **Walker**. To cause the automatic generation of a *table of contents*, have included a `walker` task. Run it using **gradle** like this:

Generating a Table Of Contents as a One-Time Event

```
gradlew walker
```

- or -

We can generate a *Table Of Contents* as a step after the `asciidoctor` task completes

Continuous Running

```
gradlew -t asciidoctor walker
```

Change gradle's `defaultTasks` to do both tasks in continuous mode.

In continuous mode, gradle notices the arrival of **toc.adoc** in the `DoctorPepper/src/docs/asciidoc` folder and converts it to **HTML**. This approach causes a tight-loop as each time `asiidoctor` ran followed by `walker`, then `walker` would produce a new **toc.adoc** which then caused `asciidoctor` to run, which caused ...

So added some timimg logic in `walker` to only produce a new **toc.adoc** file if 1) it's missing or 2) it is older than about 50 seconds.

PDF Generation

As it was so easy to do, have caused the `asciidoctor` task to produce a **PDF** file for each **.html** file generated. To improve performance or if **PDF** files are not needed, change the closure in **gradle** like this:

```
backends = ['html5','pdf']    // if you don't want PDFs delete 'pdf'
```

Enjoy **;-D**