JT acceptance tests v1

Lars Wirzenius / The Ick project

Contents

1 Introduction

JT is a command line tool for writing entries in my personal journal. It works by adding files to the git repository with the journal, for formatting by the Ikiwiki website compiler. This document presents its automated acceptance tests, using a (for now hypothetical) language similar to the Gherkin langauge implemented by Cucumber.

All of these scenarios start with an empty directory and with JT configured to put drafts in the `drafts` directory, topics in `topics`, and finished journal entries in `notes`.

2 Happy path scenarios

2.1 Add a journal entry

We start off with an empty journal repository. We only test that it's empty here; all other scenarios assume this works.

> *given* an empty journal repository
> *then* there is only ***.git*** in the journal repository

We create a draft journal entry.

> *when* I run `jt new "my title"`
> *then* ***drafts/0.mdwn*** exists
> *and* it contains ***"my title"***

When we finish the entry, the draft is moved to the notes directory.

> *given* the date is ***2019-09-01***
> *when* I run `jt finish`
> *then* ***notes/2019/09/01/my_title.mdwn*** exists
> *and* it contains ***"my title"***

2.2 Add a journal entry with an attachment

This is similar to adding a journal entry, except it adds an attachment to the new entry. The attachment could be anything, such as a photo.

> *given* an empty journal repository
> *when* I run `jt new "my title"`
> *then* ***drafts/0.mdwn*** exists
> *and* it contains ***"my title"***

The attachment file needs to exist. It gets copied to the drafts directory.

> *given* the file `photo.jpg` exists
> *when* I run `jt attach 0 photo.jpg`
> *then* ***drafts/0/photo.jpg*** exists

When we finish the entry, the draft is moved to the notes directory, with the attachment.

> *given* the date is ***2019-09-01***
> *when* I run `jt finish`
> *then* ***notes/2019/09/01/my_title.mdwn*** exists
> *and* it contains ***"my title"***
> *and* ***notes/2019/09/01/my_title/photo.jpg*** exists

2.3 Add a topic

JT allows "topic", which are meant to be pages that collect entries of specific topics. The formatted journal will have a page for each topic shows all entries that reference that topic. Similar to tags, but

another dimension of metadata. It's up to the user to use tags or topics as they wish.

> *given* an empty journal repository
> *when* I run `jt new-topic 2019/my-project`
> *then* ***topics/2019/my-project.mdwn*** exists

One can add an entry for a topic, but it must exist.

> *when* I run `jt new --topic nonexistent "my title"`
> *then* it fails *when* I run `jt new --topic 2019/my-project "my title"`
> *then* **drafts/0.mdwn**_ exists
> *and* it contains ***"2019/my-project"***

3 Unhappy path scenarios

3.1 Finishing when there's more than one draft

JT will pick the draft to act on if there's only one current draft. If there's more, it will fail with an error message saying "too many drafts".

> *given* an empty journal repository
> *when* I run `jt new "first"`
> *then* **drafts/0.mdwn**_ exists
> *when* I run `jt new "second"`
> *then* **drafts/1.mdwn**_ exists
> *when* I run `jt finish`
> *then* it fails with `"too many drafts"`

3.2 Attaching when there's no such draft

Attaching to a draft that doesn't exist doesn't work.

> *given* an empty journal repository
> *and* the file `photo.jpg` exists
> *when* I run `jt attach 0 photo.jpg`
> *then* it fails with `"no such draft"`

3.3 Attaching when there's no such file

Attaching a file that doesn't exist doesn't work.

> *given* an empty journal repository
> *when* I run `jt new "first"`
> *and* I run `jt attach 0 photo.jpg`
> *then* it fails with `"no such file"`