Historical Versions and Patches of the Removed `rgb.txt` File

GPT-4

May 27, 2023

This repository contains the historical versions and patches of the `rgb.txt` file which was removed from the Vim Git repository. The purpose of this repository is to allow users to understand the evolution of the file over time and to recreate any previous version of the file.

Structure

The repository primarily consists of the `dist` directory, the build scripts (`build_files.sh`, `build_patches.sh`), a `GNUmakefile` and the `README.tex`:

- `dist:` A directory that contains all the generated version and patch files. Each version file is timestamped with the time of the commit from which it was generated. Correspondingly, each patch file contains the changes introduced in a specific commit.
- `build_files.sh, build_patches.sh:` The scripts used to generate the version and patch files.
- `GNUmakefile:` Defines the tasks for building the project.
- `README.tex:` This document, explaining the repository.

Usage

Simply clone this repository to get access to all the historical versions and patches of the `rgb.txt` file. Navigate to the `dist` directory to find all the files. The version files are named as `rgb.txt.[timestamp]` and patch files are named in the format `0001-patch-[log entry].patch`.

Building

The building process involves running scripts to generate different versions and patches for the `rgb.txt` file from the Vim repository, which is included as a submodule. The process is managed by a `GNUmakefile`.

Here are the steps involved in the building process:

1. Run the `build_files.sh` script: This Bash script generates distinct versions of the `rgb.txt` file. It checks out each commit that modified the file, and creates a version of the file timestamped with the commit time. The output files are moved into the `dist` directory.

2. Run the `build_patches.sh` script: This Bash script generates Git patch files for each commit that changed the `rgb.txt` file. These patches can be applied to the file to recreate the version from each commit. The output patch files are moved into the `dist` directory.

3. Generate the `README.pdf`: The `GNUmakefile` also compiles the `README.tex` into a `README.pdf` using `xelatex`.

Note that the `GNUmakefile` includes a `clean` command to remove the generated files.

Dependencies

These scripts depend on a working installation of Git, `parallel`, and `xelatex`, which is used to compile the `README.tex` into a PDF.