Contribuindo para o Git

Lucas Seiki Oshiro

Prós e Contras

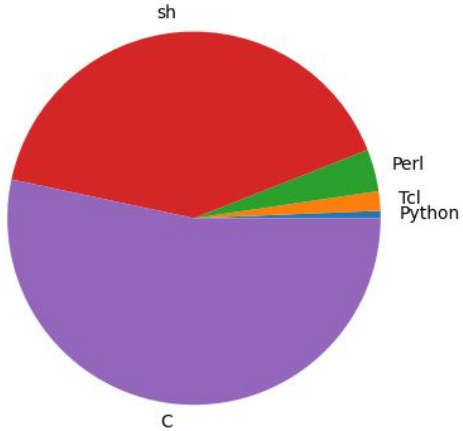
- **PRÓ**: Código bem próximo do **dia-a-dia**
  - Todo mundo usa Git o tempo todo
- **PRÓ**: Funcionamento interno **elegante**
  - Interessante para quem gosta mais de **teoria da computação**
  - DAGs, Merkle Trees, Hash, ordenação topológica, etc
- **PRÓ**: Compilação, execução e escrita de testes **muito fáceis**
- **PRÓ**: Comunidade receptiva e que **responde rápido**
- **CONTRA**: código em **C** e com estrutura de arquivos pouco convencional
  - Apesar disso, não é um código difícil de ler
- **CONTRA**: revisões muito criteriosas, mesmo um patch simples pode levar várias iterações até ser aceito
- **CONTRA**: um pouco difícil de encontrar issues em aberto para contribuição

Requisitos

- **C**
- Noções básicas de **sh** e **testes**
- Fluência em Git:
  - Conceitos **básicos**: commit, branch, staging, merge, tag, …
  - Conceitos de **baixo nível**: objetos, oids, referências, reflog, index, packfile, …
- **Sugestão obrigatória** 1: alguns capítulos do **Pro Git**, disponível online
  - 1.3 What is Git
  - 10. Git Internals, seções “Plumbing and Porcelain”, “Git Objects”, “Git References”
- **Sugestão obrigatória** 2: usar editor de texto ou IDE bem configurado para **navegação no código**
  - Exemplos: Emacs + ETAGS, Vim/Neovim + CTAGS, VSCode, Eclipse, JetBrains CLion

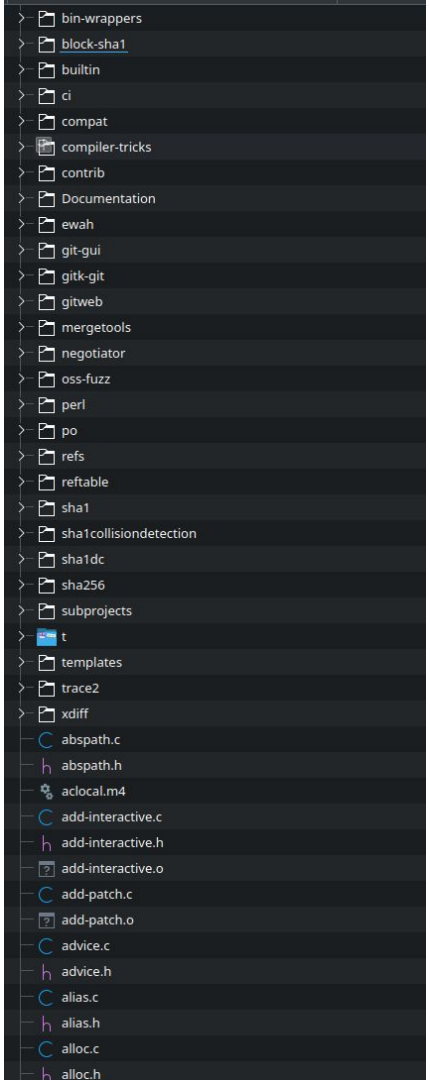
Base de código

- `git clone git@github.com/git/git.git`
  - (sim, 6 vezes a palavra git)
- Código principalmente em **C**, além de:
  - **sh**: principalmente **testes**
  - Python: compatibilidade com Preforce
  - Tcl: interface gráfica (gitk)
  - Perl: alguns scripts (ex: send-email)
- Build: **Makefile** ou **Meson**
  - `make -j $(nproc)`



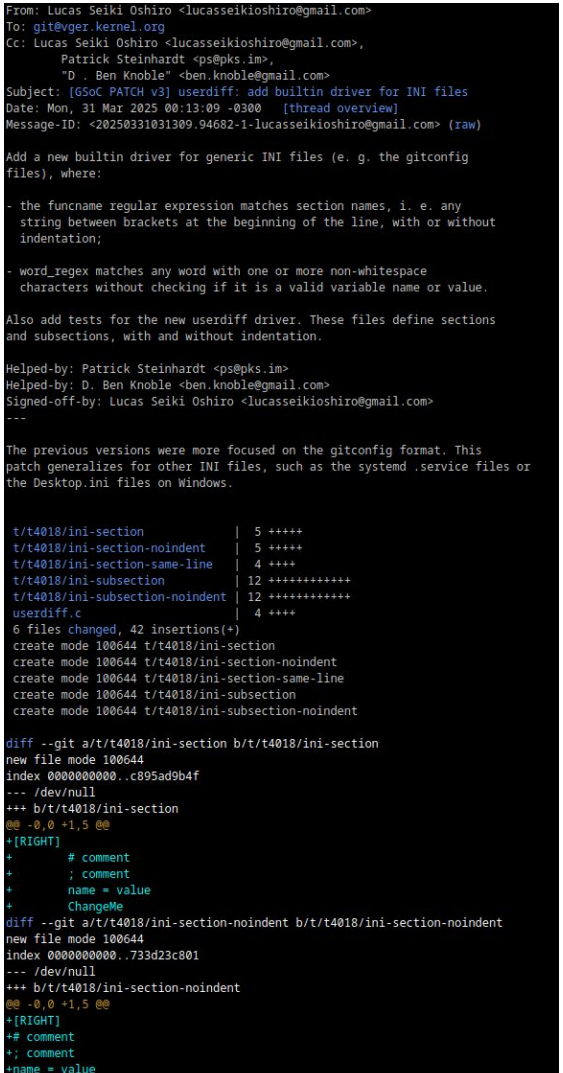
Estrutura

- Não muito convencional… Arquivos objeto (.o) são gerados nos mesmos diretórios dos fontes (.c/.h)
- Funcionalidades mais básicas ficam na raiz
- **Comandos** ficam dentro de `builtin`
- **Testes** ficam dentro de `t`
- **Documentação** dentro de `Documentation`



Fluxo de contribuição

- Igual ao kernel:
  - Patches por **e-mail** (usando git-send email)
  - Alternativamente, por PR no GitHub (**GitGitGadget**)
  - Revisões na lista de e-mail
  - Mesmas guidelines para **mensagens de commit** e **estilo de código**
  - Interação na lista de email e IRC



```
From: Lucas Seiki Oshiro <lucasseikioshiro@gmail.com>
To: git@vger.kernel.org
Cc: Lucas Seiki Oshiro <lucasseikioshiro@gmail.com>,
       Patrick Steinhardt <ps@pks.im>,
       "D . Ben Knoble" <ben.knoble@gmail.com>
Subject: [GSoC PATCH v3] userdiff: add builtin driver for INI files
Date: Mon, 31 Mar 2025 00:13:09 -0300   [thread overview]
Message-ID: <20250331031309.94682-1-lucasseikioshiro@gmail.com> (raw)

Add a new builtin driver for generic INI files (e. g. the gitconfig
files), where:

- the funcname regular expression matches section names, i. e. any
  string between brackets at the beginning of the line, with or without
  indentation;

- word_regex matches any word with one or more non-whitespace
  characters without checking if it is a valid variable name or value.

Also add tests for the new userdiff driver. These files define sections
and subsections, with and without indentation.

Helped-by: Patrick Steinhardt <ps@pks.im>
Helped-by: D. Ben Knoble <ben.knoble@gmail.com>
Signed-off-by: Lucas Seiki Oshiro <lucasseikioshiro@gmail.com>
---

The previous versions were more focused on the gitconfig format. This
patch generalizes for other INI files, such as the systemd .service files or
the Desktop.ini files on Windows.


 t/t4018/ini-section             |  5 +++++
 t/t4018/ini-section-noindent    |  5 +++++
 t/t4018/ini-section-same-line   |  4 ++++
 t/t4018/ini-subsection          | 12 ++++++++++
 t/t4018/ini-subsection-noindent | 12 ++++++++++
 userdiff.c                      |  4 ++++
 6 files changed, 42 insertions(+)
 create mode 100644 t/t4018/ini-section
 create mode 100644 t/t4018/ini-section-noindent
 create mode 100644 t/t4018/ini-section-same-line
 create mode 100644 t/t4018/ini-subsection
 create mode 100644 t/t4018/ini-subsection-noindent

diff --git a/t/t4018/ini-section b/t/t4018/ini-section
new file mode 100644
index 0000000000..c895ad9b4f
--- /dev/null
+++ b/t/t4018/ini-section
@@ -0,0 +1,5 @@
+[RIGHT]
+       # comment
+       ; comment
+       name = value
+       ChangeMe
diff --git a/t/t4018/ini-section-noindent b/t/t4018/ini-section-noindent
new file mode 100644
index 0000000000..733d23c801
--- /dev/null
+++ b/t/t4018/ini-section-noindent
@@ -0,0 +1,5 @@
+[RIGHT]
+# comment
+; comment
+name = value
```

Referências úteis

- Posts do Matheus Tavares sobre Git: https://matheustavares.dev/tags/git/
  - **IMPORTANTE:** https://matheustavares.dev/posts/first-steps-contributing-to-git
- Hacking Git: https://git.github.io/Hacking-Git/
- Sugestões de microprojetos: https://git.github.io/Hacking-Git/