

AI による Git コミットメッセージ

この Python スクリプトは、~/bin など、システムの PATH に含まれるディレクトリに配置する必要があります。

```
import subprocess
import os
from openai import OpenAI
from dotenv import load_dotenv
import argparse

load_dotenv()

def gitmessageai(push=True, only_message=False):
    # すべての変更をステージング
    subprocess.run(["git", "add", "-A"], check=True)

    # ステージングされた変更の差分を取得
    diff_process = subprocess.run(["git", "diff", "--staged"], capture_output=True, text=True, check=True)
    diff = diff_process.stdout

    if not diff:
        print(" コミットする変更がありません。 ")
        return

    # AI へのプロンプトを準備
    prompt = f"""
```

以下のコード変更に対して、**Conventional Commits** 形式で簡潔なコミットメッセージを生成してください。
次のいずれかのタイプを使用してください: **feat**, **fix**, **docs**, **style**, **refactor**, **test**, **chore**, **perf**, **ci**, **build**
該当する場合は、影響を受けるコードベースの部分を説明するために括弧内にスコープを含めてください。
コミットメッセージは **70** 文字を超えないようにしてください。

コード変更:

```
{diff}
```

コミットメッセージ:

```
"""
```

```
# DeepSeek API にプロンプトを送信
```

```
api_key = os.environ.get("DEEPSEEK_API_KEY")
```

```
if not api_key:
```

```
    print(" エラー: DEEPSEEK_API_KEY 環境変数が設定されていません。")
```

```
    return
```

```
client = OpenAI(api_key=api_key, base_url="https://api.deepseek.com")
```

```
try:
```

```
    response = client.chat.completions.create(
```

```
        model="deepseek-chat",
```

```
        messages=[
```

```
            {"role": "user", "content": prompt}
```

```
        ],
```

```
        max_tokens=100
```

```
    )
```

```
    if response and response.choices:
```

```
        commit_message = response.choices[0].message.content.strip()
```

```
        commit_message = commit_message.replace('\n', '')
```

```
    else:
```

```
        print(" エラー: API からの応答がありません。")
```

```
        return
```

```
except Exception as e:
```

```
    print(f"API 呼び出し中のエラー: {e}")
```

```
    return
```

```
# コミットメッセージが空かどうかを確認
```

```
if not commit_message:
```

```
    print(" エラー: 空のコミットメッセージが生成されました。コミットを中止します。")
```

```

        return

if only_message:
    print(f" 提案されたコミットメッセージ: {commit_message}")
    return

# 生成されたメッセージでコミット
subprocess.run(["git", "commit", "-m", commit_message], check=True)

# 変更をプッシュ
if push:
    subprocess.run(["git", "push"], check=True)
else:
    print(" 変更はローカルでコミットされましたが、プッシュされていません。")

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="AI でコミットメッセージを生成し、変更をコミットしま
    parser.add_argument('--no-push', dest='push', action='store_false', help='変更をプッシュせずにロー
    parser.add_argument('--only-message', dest='only_message', action='store_true', help='AI が生成した
    args = parser.parse_args()
    gitmessageai(push=args.push, only_message=args.only_message)

```

次に、`~/.zprofile` ファイルに以下を追加します:

```

alias gpa='python ~/bin/gitmessageai.py'
alias gca='python ~/bin/gitmessageai.py --no-push'
alias gm='python ~/bin/gitmessageai.py --only-message'

```