GitHubでPDF共有・更新手順

使用したPC：Mac（ターミナル / Git）
目的：コマンドのPDFをGitHubに置き、**GitHubページで表示してから任意にダウンロード**してもらう（blobリンク運用）
対象リポジトリ：`yumi-mimimi/practice_pub`

---

0. 前提

- リポジトリは **Public**（URL配布で誰でも見れる運用）
- `git push` は **SSH** で通る状態（Username/Passwordを聞かれない）

> Publicでも、他人が勝手にpushして改ざんはできません（あなたが権限を付与しない限り不可）。
> 見れる・DLできるのは「Publicの仕様」です。

---

1. 初回セットアップ（手元に作業フォルダが無い場合）

1.1 clone

```
cd ~/Desktop
git clone https://github.com/yumi-mimimi/practice_pub.git
cd practice_pub
```

1.2 remoteがSSHか確認（なんか入れない場合など）

```
git remote -v
```

SSHにしたい場合：

```
git remote set-url origin git@github.com:yumi-mimimi/practice_pub.git
git remote -v
```

---

2. PDFを追加（初回アップロード）

2.1 DownloadsのPDFをリポジトリにコピー

```
cd ~/Desktop/practice_pub
mkdir -p pdf
```

```
cp ~/Downloads/"提出したいフォルダ.pdf" pdf/
ls -lh pdf/
```

2.2 add → commit → push

```
git add pdf/
git commit -m "Add SRX PDF"
git status で念の為確認
git push
```

---

3. 共有リンク（任意DL：blobリンク）

3.1 blobリンク（GitHub表示→任意でDownload）

相手に送るリンク形式：

```
https://github.com/yumi-mimimi/practice_pub/blob/main/pdf/<URLエンコード済みファイル名>
```

今回（例：v1.1）のリンク：

- https://github.com/yumi-mimimi/practice_pub/blob/main/pdf/SRX%E6%BC%94%E7%BF%92%E3%82%B3%E3%83%9E%E3%83%B3%E3%83%89_v1.1.pdf

3.2 日本語ファイル名のblobリンクをCLIで生成（再利用）

```
printf "https://github.com/yumi-mimimi/practice_pub/blob/main/pdf/%s\n" "$(python3 -c 'import urllib.parse; print(urllib.parse.quote("SRX演習コマンド_v1.1.pdf"))')"
```

---

4. READMEに最新版リンクを載せる（最新で更新したものを確認ができるため）

4.1 READMEにDownloadsセクションを追加（最新版リンク）

```
cd ~/Desktop/practice_pub

BLOB=$(python3 -c 'import urllib.parse; print("https://github.com/yumi-mimimi/practice_pub/blob/main/pdf/" + urllib.parse.quote("上げたフォルダ.pdf"))')
```

```
printf "\n## Downloads\n- 最新（v1.1）: %s\n" "$BLOB" >> README.md
tail -n 20 README.md
```

4.2 add → commit → push

```
git add README.md
git commit -m "Update README with latest PDF link"
git push
```

---

5. PDF更新手順（更新したい場合。例：v1.0 → v1.1 など）

5.1 事故防止：必ずリポジトリ直下で作業

> `pdf/` の中に入った状態で `pdf/...` を付けると `pdf/pdf/...` になって失敗しがち。

```
cd ~/Desktop/practice_pub
pwd
ls -lh
```

5.2 改訂版PDFを「v1.1」として追加

（Downloadsにある例：`SRX演習コマンド 改訂ver1.1.pdf`）

```
cp ~/Downloads/"SRX演習コマンド 改訂ver1.1.pdf" pdf/"SRX演習コマンド_v1.1.pdf"
ls -lh pdf/
```

5.3 旧ファイルをv1.0にリネーム（履歴を残す）

（旧運用が `SRX演習コマンド.pdf` だった場合）

```
git mv pdf/"SRX演習コマンド.pdf" pdf/"SRX演習コマンド_v1.0.pdf"
ls -lh pdf/
```

5.4 READMEの最新版リンクをv1.1へ（必要に応じて）

READMEにまだリンクが無い場合は「4章」を実施。
すでにDownloadsがある場合は、最新行を編集して v1.1 を指すように更新する。

（追記で増やしたくない場合は、READMEを手動編集が一番安全）

5.5 add → commit → push

```
git add README.md pdf/
git status
git commit -m "Add SRX PDF v1.1 and update README"
git push
```

---

6. 更新したはずなのに `git status` が clean の場合（差分無し）

PDFは「更新日時が変わっても中身が同じ」なら差分として扱われません。

確認（HEADと作業ツリーのハッシュが同じなら中身は同一）：

```
echo "HEAD:" && git rev-parse HEAD:pdf/"SRX演習コマンド_v1.0.pdf"
echo "WORKTREE:" && git hash-object pdf/"SRX演習コマンド_v1.0.pdf"
```

対処：

- “改訂版”として想定しているPDFが本当に違うファイルか確認（別名・別場所を探す）
- PDFを再出力（エクスポートし直し）して差分が出るか確認

---

7. 提出前セルフチェック

- [ ] `~/Desktop/practice_pub` で作業している（`pwd`で確認）
- [ ] `pdf/` に `SRX演習コマンド_v1.0.pdf` / `SRX演習コマンド_v1.1.pdf` がある
- [ ] READMEに「最新（v1.1）」のblobリンクがある
- [ ] `git push` がエラーなく成功している
- [ ] 共有するURLは **blobリンク**（GitHub表示→任意DL）になっている