Zotero Integration with Google Docs and Git Workflow

the why: to avoid having to go through the pain of redoing citations manually while collaborating with other people

Part 1: Google Docs ↔ Zotero → DOCX Workflow

Initial Setup

This document describes a simple workflow to collaborate in Google Docs using Zotero citations, then convert the final file to a DOCX that your local Zotero/Word plugin can recognize for final referencing and bibliography generation.

1. **Create a Google Doc for collaboration**

- Create a Google Doc and share it with collaborators (Editors as needed).
- Use Google Docs for real-time collaborative drafting and commenting.

2. **Activate and use the Zotero Connector in Google Docs**

- Install the Zotero Connector in your browser and sign into your Zotero account (and make sure your Zotero desktop/client is running if you need local library access).
- In the Google Doc, open the Zotero menu (the connector provides a Zotero toolbar/menu inside the doc).
- Insert citations and generate bibliography using the Zotero menu as you write.

Notes:

- Collaborators who need to add citations should have the Zotero Connector and access to the same Zotero library or shared group library.
- **Collaborators who only edit text do not need Zotero installed.**

Collaboration Process

3. **Collaborators continue to edit as usual**

- All collaborators can continue to edit, comment, and suggest changes in Google Docs.
- Zotero citations in the doc remain active and editable through the Zotero menu while the file stays in Google Docs.

Final Export Process

4. **When the file is final: use "Switch Word Processor" before exporting**

- In the Google Docs Zotero menu choose “Switch Word Processor” option provided by the Zotero menu).
  - This converts Google Docs citation placeholders into DOCX fields that the Zotero Word plugin (desktop) can recognize.
  - The result is a .docx file you can download.
- Download the converted .docx to your Mac.

5. **Open the DOCX locally so Zotero recognizes it**

- Ensure the Zotero desktop app is running.
- Open the downloaded .docx in Microsoft Word (on macOS) with the Zotero Word plugin enabled.
- The Word Zotero plugin should detect the Zotero fields. Use the plugin’s "Document Preferences" or "Refresh" to verify citation style and update the bibliography.
- Save the DOCX; it now acts as a Zotero-managed document (you can continue to update citations via the Word plugin).

Troubleshooting & Tips

- If the DOCX is not recognized:
  - Make sure Zotero desktop is open and the Word plugin is installed/active.
  - In Word, use Zotero → Refresh to re-establish field recognition.
  - If fields show as plain text, re-open the DOCX after restarting Zotero or reinstalling the Word plugin.
- For shared citations, use Group Libraries so all contributors with permission can insert the same references.
- Keep a copy of the original Google Doc if you need further online edits; repeat the conversion when ready for a new final DOCX.
- If collaborators used different Zotero libraries, verify all citations are present in a shared library before switching processors.

This part below is relevant if you want a more secure backup for your Zotero library. The steps below will allow backup of Zotero library via Github automatically everytime you push a commit using Github Actions.
These are quite advanced and you don't need to do these if you have a solid backup for your library.

Part 1.5: Better BibTeX Integration

Setting Up Better BibTeX

1. **Install Better BibTeX for Zotero**
    - Download from Better BibTeX GitHub releases
    - In Zotero: Tools → Add-ons → Install Add-on From File
    - Restart Zotero after installation
2. **Configure Better BibTeX**
    - In Zotero: Edit → Preferences → Better BibTeX
    - Recommended settings:
        - Citation key format: `[auth:lower][year]`
        - Keep keys unique: On
        - Auto-export: On
        - Auto-pin citations: On
3. **Set Up Auto-Export**
    - Select your library or collection
    - Right-click → Export Library
    - Choose Format: "Better BibTeX"
    - Check "Keep updated"
    - Save to `references/zotero_backup/library.bib`

Integration with Git

1. **Citation Key Stability**
    - Better BibTeX generates stable citation keys
    - Keys remain consistent across collaborators
    - Format: `smith2023` or `smith2023title`
2. **Auto-export Configuration**

```
{
  "autoExport": {
    "type": "better-bibtex",
    "path": "references/zotero_backup",
    "automatic": true,
    "interval": "5m"
}
```

3. **Git Workflow**
   - Track `.bib` files in Git
   - Commit changes automatically via GitHub Actions
   - Use citation keys in Markdown/LaTeX files

Part 2: GitHub Integration

Setting Up Git Hooks (Required)

1. **Create Git Hooks Directory**

   ```
   mkdir -p .git/hooks
   ```

2. **Create Pre-Commit Hook**

   Create `.git/hooks/pre-commit` :

```bash
#!/bin/bash

BACKUP_DIR="references/zotero_backup"

# Check if directory exists
if [ ! -d "$BACKUP_DIR" ]; then
  echo "Error: $BACKUP_DIR directory is missing"
  exit 1
fi

# Check for BibTeX files
if ! find "$BACKUP_DIR" -name "*.bib" -type f | grep -q .; then
  echo "Error: No BibTeX files found in $BACKUP_DIR"
  exit 1
fi

# Check for recent updates (within 24 hours)
if ! find "$BACKUP_DIR" -name "*.bib" -mtime -1 | grep -q .; then
  echo "Warning: BibTeX files not updated in last 24 hours"
  echo "Please update Zotero exports before committing"
  exit 1
fi
```

3. **Make Hook Executable**

```bash
chmod +x .git/hooks/pre-commit
```

4. **Share Hooks with Team**

```bash
# Create shared hooks directory
mkdir -p .githooks
cp .git/hooks/pre-commit .githooks/
git add .githooks
git config core.hooksPath .githooks
```

Setting Up GitHub Actions

1. Create the following directory structure in your repository:

```
your-repo/
├── .github/
│   └── workflows/
│       └── zotero-backup.yml
└── references/
    └── zotero_backup/
        ├── your-library.bib
        └── your-library.json
```

2. Configure GitHub Actions by creating the workflow file below
3. Ensure your Zotero is configured to export to the `references/zotero_backup` directory
4. Commit and push your changes
5. Verify the workflow in the GitHub Actions tab

GitHub Actions Workflow File

Create `.github/workflows/zotero-backup.yml` with:

```
name: Zotero Backup Verification
on:
  push:
    branches: [ main, master ]
  pull_request:
    branches: [ main, master ]

jobs:
  verify-backup:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3

    - name: Verify Zotero backup files exist
      run: |
        if [ -d "references/zotero_backup" ]; then
          echo "✅ Zotero backup directory exists"

          if ls references/zotero_backup/*.bib 1> /dev/null 2>&1; then
            echo "✅ BibTeX files found:"
            ls -la references/zotero_backup/*.bib
          else
            echo "❌ No BibTeX files found"
            exit 1
          fi

          if ls references/zotero_backup/*.json 1> /dev/null 2>&1; then
            echo "✅ JSON files found:"
            ls -la references/zotero_backup/*.json
          else
            echo "❌ No JSON files found"
            exit 1
          fi

          # Check if files are recent (modified within last 30 days)
          find references/zotero_backup -name "*.bib" -o -name "*.json" | while read fi
            if [ $(find "$file" -mtime -30 | wc -l) -eq 0 ]; then
              echo "⚠️  Warning: $file is older than 30 days"
            else
              echo "✅ $file is recent"
            fi
          done
        else
    echo "❌ Zotero backup directory not found"
    exit 1
  fi
```

Workflow Behavior

- **Triggers**:
  - Push to main/master branches
  - Pull requests to main/master branches
  - No scheduled checks implemented
- **Checks**:
  - Verifies existence of `references/zotero_backup` directory
  - Checks for presence of .bib files
  - Checks for presence of .json files
  - Monitors file age (warns if > 30 days old)
- **Actions**:
  - Prints status messages with emojis for visibility
  - Exits with error code 1 if required files are missing
  - Shows warnings for outdated files
  - Does not create GitHub issues (not implemented)