File Attachments & Directory Context

Overview

TermAI now supports comprehensive file attachment and directory context features in both inline and chat modes. You can attach images, PDFs, text files, and code to your prompts for AI analysis.

Supported File Types

Images

- `.jpg`, `.jpeg`, `.png`, `.gif`, `.webp`
- Processed as base64-encoded data URLs
- Compatible with vision models (e.g., GPT-4 Vision, Claude with vision)

PDFs

- `.pdf`
- Text content is automatically extracted
- Appended to the prompt as text

Text Files

- `.txt`, `.md`, `.markdown`
- Content read directly and appended to prompt

Code Files

- `.go`, `.py`, `.js`, `.ts`, `.tsx`, `.jsx`, `.java`, `.c`, `.cpp`, `.h`, `.rs`, `.rb`, `.php`, `.sh`, `.bash`
- `.yaml`, `.yml`, `.json`, `.xml`, `.html`, `.css`, `.sql`
- Content read directly with proper formatting preserved

Inline Mode

Basic Usage

```
# Single file
termai -file document.pdf "Summarize this document"

# Multiple files
termai -file image.png -file code.go "Analyze these files"

# Short flag
termai -f report.md -f data.txt "Compare these documents"
```

Features

- Files are processed before sending the prompt
- Images are sent as base64 data URLs to vision models
- Text/PDF/Code content is appended to the prompt
- Progress feedback shows file processing status

Example Output

```
Processing files... ✓ Processed 2 file(s)
  • Image: screenshot.png
  • PDF: report.pdf

User: Analyze the report and the screenshot

(with 2 attachment(s))


## Chat Mode

### Starting Chat with Files
```bash
# Attach files when starting chat
termai chat -file document.pdf -file image.png

# Multiple files with profile
termai chat -p openai -file code.go -file readme.md
```

Directory Context

```
# Load all supported files from a directory as context
termai chat --dir ./project

# Short flag
termai chat -d ./src

# Combined with file attachments
termai chat -d ./docs -file screenshot.png
```

Note: Directory scanning only processes top-level files (no subdirectories) for safety.

In-Chat Commands

/attach - Attach Files

```
/attach document.pdf
/attach file1.txt file2.md
/attach path/to/image.png
```

Attaches one or more files to be included in the next message.

/files - Show Attached Files

```
/files
```

Displays a list of currently attached files with their types.

/clear-files - Clear Attachments

```
/clear-files
```

Removes all attached files. Useful if you want to send a message without the previously attached files.
```

/context - Show Context Files

```
/context
```

Shows files loaded from the directory context (via `--dir` flag).

/context-add - Add to Context

```
/context-add newfile.txt
/context-add module.py config.yaml
```

Adds additional files to the context.

/context-remove - Remove from Context

```
/context-remove oldfile.txt
```

Removes a specific file from the directory context.

UI Indicators

Header

When directory context is active, the header shows:

```
✨ TermAI  | profile_name | model_name | 📁 src (12 files) | ● Ready
```

Input Area

When files are attached, they’re shown above the input:

```
📎 Attached: 📷 screenshot.png, 📄 report.pdf, 💻 code.go

You ▸ Your message here...
```

File type icons:
- 📷 Images
- 📄 PDFs
- 💻 Code files
- 📝 Text files

Configuration

Add file handling settings to your `~/.termai/config.yaml`:

```yaml
files:
  max_file_size: 10485760          # 10MB (in bytes)
  auto_clear_after_send: true      # Clear attached files after sending
  include_context_in_every_msg: false  # Include context files in every message
```

Configuration Options

- **max_file_size**: Maximum file size in bytes (default: 10MB)

- **auto_clear_after_send**: When `true`, attached files are cleared after sending a message (default: `true`)
- Context files (from `--dir`) are NOT cleared automatically
- **include_context_in_every_msg**: When `true`, context files are included in every message (default: `false`)
- When `false`, context files are only included in the first message

How It Works

File Processing Flow

1. Files are validated (existence, size, type)
2. Images are encoded as base64 data URLs
3. PDFs have text content extracted
4. Text/code files are read directly
5. Files are added to the message:
   - Images → `Message.Images[]` array
   - Text content → Appended to `Message.Content`

Message Construction

When you send a message with attachments:

```
Your prompt text

--- Content from document.pdf ---
[PDF text content]
--- End of document.pdf ---

--- Content from code.go ---
[Code content]
--- End of code.go ---
```

Images are sent separately in the `Images` array for vision model processing.

Examples

Example 1: Code Review

```
termai chat -d ./src
```

In chat:

```
You: Review the architecture and suggest improvements
```

The AI will have access to all supported files in `./src` as context.

Example 2: Document Analysis

```
termai -file report.pdf -file data.csv "Summarize key findings"
```

Example 3: Image Analysis with Context

```
termai chat -d ./project
```

In chat:

```
/attach screenshot.png
What does this UI screenshot show in the context of our codebase?
```

Example 4: Multi-File Comparison

```
termai chat
```

In chat:

```
/attach version1.py version2.py
Compare these two implementations and suggest the better approach
```

Best Practices

1. **File Size**: Keep files under the configured `max_file_size` (default 10MB)
2. **Context Management**: Use directory context for related files in a project
3. **Selective Attachments**: Use `/attach` for specific files you want to discuss
4. **Clear When Done**: Use `/clear-files` when changing topics
5. **Vision Models**: Use images with models that support vision (GPT-4 Vision, Claude with vision)

Troubleshooting

“File does not exist”

- Check the file path is correct
- Use absolute paths or paths relative to your current directory

“Unsupported file type”

- Check if your file extension is in the supported list
- Rename files to have proper extensions if needed

“Failed to process PDF”

- Ensure the PDF is not corrupted
- Some PDFs may be image-based and won’t extract text well

Files not showing in context

- Verify files are in the top level of the directory (subdirectories are skipped)
- Check if file types are supported
- Use `/context` command to see what’s loaded

Technical Notes

Security

- Files are processed locally before being sent to the AI
- No files are uploaded to external services except as part of your API request
- Directory scanning is limited to top-level to prevent accidental exposure of sensitive subdirectories

Performance

- Large files may take time to process
- PDF text extraction can be slow for large documents
- Consider splitting very large files

Compatibility

- Vision features require vision-capable models (GPT-4 Vision, Claude 3, etc.)
- Image attachments are sent as base64 data URLs
- Text content is appended directly to prompts