ConvertX

A desktop file converter built with **Flutter + Rust**.

`status` `active development` `platform` `Windows` `license` `MIT`

[!WARNING] **Work in Progress**

This project is under active development. Features and APIs may change. Not recommended for production use yet. Feedback and issues are welcome.

Table of Contents

Overview

ConvertX is a desktop file conversion tool:

- A modern desktop UI built with Flutter
- High-performance conversion engine built with Rust
- Type-safe FFI integration via `flutter_rust_bridge`

Current focus areas:

- Image conversion
- Text/document conversion
- EPUB -> PDF (via external tools)
- Clear conversion progress and error feedback

Features & Supported Formats

Implemented

Category	Input Formats	Output Formats	Notes
Image	`png/jpg/jpeg/webp/bmp/ico/gif`	`png/jpg/jpeg/webp/bmp/ico/gif`	Basic image conversion
Text Document	`txt/md/html/htm`	`txt/html`	Plain text and markup conversion
Office (via pandoc)	`md`	`docx/pptx`	Markdown to Word/PowerPoint
Office (via pandoc)	`docx/pptx`	`md`	Word/PowerPoint to Markdown
PDF (via Chromium)	`md`	`pdf`	Markdown to PDF
Ebook	`epub`	`pdf`	Requires `pandoc` or `ebook-convert`

Planned

- ☐ Generic document -> PDF (currently only `epub -> pdf`)

- ☐ Audio/video conversion (planned with FFmpeg)
- ☐ PDF input parsing/conversion
- ☐ More batch processing and advanced options

UI Behavior

- Output formats are filtered dynamically based on selected input files
- In-progress tasks clearly show `Converting...`
- Long-running jobs (like EPUB -> PDF) show external-tool stage hints
- Failed tasks support detailed error viewing

Tech Stack

- **Frontend**: Flutter (Dart)
- **Backend**: Rust
- **Bridge**: flutter_rust_bridge

Quick Start

1) Prerequisites (Windows)

1. Install Flutter SDK: https://docs.flutter.dev/get-started/install/windows
2. Install Rust: https://www.rust-lang.org/tools/install
3. Install Visual Studio with the **Desktop development with C++** workload

Verify installation:

```
flutter doctor
rustc --version
cargo --version
```

2) Clone and Run

```
git clone <your-repo-url>
cd ConvertX

# Initialize Flutter platform files (first time only)
flutter create . --platforms=windows

# Install dependencies
flutter pub get

# Build Rust backend
cd rust
cargo build --release
```

```
cd ..

# Run app
flutter run -d windows
```

EPUB -> PDF Dependencies

`epub -> pdf` requires one of the following external tools:

- `pandoc`
- `calibre` (provides `ebook-convert`)

Verify Installation

```
pandoc --version
ebook-convert --version
```

Windows Installation (choose one or both)

```
winget install Pandoc
```

```
winget install calibre.calibre
```

Markdown <-> Office Dependencies

`md <-> docx` and `md <-> pptx` are implemented via **pandoc**.

Verify Installation

```
pandoc --version
```

Windows Installation

```
winget install Pandoc
```

Markdown -> PDF Dependencies

`md -> pdf` uses **Chromium** (via headless_chrome) for high-quality output.

Recommended Setup (Chromium)

Requirements:

- **Chrome** or **Chromium** browser installed on your system

Verify Installation:

```powershell
# Check if Chrome is installed (Windows)
Get-Command chrome -ErrorAction SilentlyContinue
# or check for Chromium
Get-Command chromium -ErrorAction SilentlyContinue
```

Why Chromium?

- Better HTML/CSS rendering for professional PDF output
- Superior styling and layout compared to LaTeX-based output
- Better support for complex Markdown with embedded HTML

Project Structure

```
ConvertX/
├── lib/                    # Flutter frontend
│   ├── main.dart           # App entry
│   └── src/
│       ├── screens/        # Screens
│       ├── widgets/        # UI components
│       ├── providers/      # State management
│       └── rust/           # Generated Rust bindings
├── rust/                   # Rust backend
│   ├── src/
│   │   ├── api.rs          # FFI API
│   │   └── converters/     # Conversion logic
│   └── Cargo.toml
└── pubspec.yaml
```

Roadmap

- ☑ Base project structure
- ☑ Image and document conversion
- ☑ EPUB -> PDF via external tools
- ☑ Progress and error feedback improvements

- ☐ Full flutter_rust_bridge workflow docs
- ☐ Audio/video conversion (FFmpeg)
- ☐ Automated testing and release pipeline

Contributing

Contributions are welcome:

1. Fork the repository
2. Create a branch: `feat/xxx` or `fix/xxx`
3. Commit your changes and open a Pull Request
4. In your PR, describe purpose, scope, and test results

If you find a bug, please open an Issue with minimal reproduction steps.

License

MIT