Productionize ag-docs-to-pdf VS Code Extension

Comprehensive refactoring to improve code quality, performance, reliability, and testability of the Antigravity Docs to PDF extension. This covers all five areas: architecture, performance, correctness, testing, and UX.

User Review Required

> **Important**
>
> **Browser reuse strategy**: During "Export All", the current code launches a new Chromium process per PDF. The plan adds a `BrowserPool` that lazily launches one browser and shares it across all `generate()` calls in a session, disposing on `deactivate()`. This is a significant architectural change.

> **Warning**
>
> **New configuration**: Adding `antigravity.chromePath` setting that lets users override the auto-detected browser. This is a user-facing breaking-ish change — existing users won't be affected since it defaults to auto-detect, but the setting surface changes.

Proposed Changes

Browser Pool (New File)

[NEW] TS browserPool.ts

Singleton-style browser lifecycle manager. Key design:

```
export class BrowserPool {
  private browser: Browser | null = null;

  async acquire(): Promise<Browser>  // lazily launches, returns shared instance
  async dispose(): Promise<void>     // closes browser, called on deactivate()
}
```

- Launch browser once on first `generate()`, reuse for all subsequent calls
- `dispose()` called in `extension.ts#deactivate()` for cleanup
- Guards against concurrent launches with a launch-in-progress promise lock

PdfGenerator Refactoring

[MODIFY] TS PdfGenerator.ts

Changes:

1. **Accept `BrowserPool` via constructor** instead of launching browser internally

2. **Extract header/footer templates** to private constants (remove inline HTML from `page.pdf()` call)
3. **Improve** `waitForMermaid`: Use a smarter wait — `waitForFunction` with a completion flag set via MutationObserver on the client side, falling back to the existing SVG-check approach
4. **Add configurable timeout** via optional `PdfOptions.timeoutMs`

```
 export class PdfGenerator {
   private templateHtml: string;
   private stylesCss: string;
+  private browserPool: BrowserPool;

-  constructor() {
+  constructor(browserPool: BrowserPool) {
+    this.browserPool = browserPool;
     // ... asset loading stays the same
   }

   async generate(options: PdfOptions): Promise<string> {
-    const browserPath = await findBrowser();
-    const browser = await puppeteer.launch({ ... });
+    const browser = await this.browserPool.acquire();
     const page = await browser.newPage();
     try {
       // ... render logic ...
     } finally {
-      await browser.close();
+      await page.close();  // close PAGE, not browser
     }
   }
 }
```

markdownRenderer Type Safety

[MODIFY] TS markdownRenderer.ts

Changes:

1. **Remove** `@ts-ignore` — import Token type properly or use `MarkdownIt.Token`
2. **Remove all** `as any` **casts** — define proper `RenderRule` type signatures using `markdown-it`'s exported types
3. **Fix unused variable** in `mermaidPlugin` (line 151: `content` is assigned but never used)

The `markdown-it` library exports `Token` from `markdown-it/lib/token.mjs`. We'll use the `@types/markdown-it` type for `Token` directly via `import type Token from 'markdown-it/lib/token.mjs'` or use `InstanceType` patterns. The `self` parameter in render rules should be typed as `{ renderToken: (...args: unknown[]) => string }` instead of `any`.

browserFinder Improvements

[MODIFY] TS browserFinder.ts

Changes:

1. **Accept optional** `customPath` parameter (from `antigravity.chromePath` setting)
2. **Add more browser paths**: Brave browser (macOS, Win, Linux), user-installed Chromium on Linux
3. **Better error message**: include platform-specific install instructions

```
-export async function findBrowser(): Promise<string> {
+export async function findBrowser(customPath?: string): Promise<string> {
+  // 0. User-configured path takes priority
+  if (customPath && fs.existsSync(customPath)) {
+    return customPath;
+  }
   // 1. Try chrome-launcher ...
```

Extension Lifecycle Updates

[MODIFY] TS extension.ts

Changes:

1. **Create shared** `BrowserPool` in `activate()`, pass to `PdfGenerator`
2. **Dispose browser pool** in `deactivate()`
3. **Read** `antigravity.chromePath` setting and pass to `findBrowser`
4. **Remove startup notification** (`showInformationMessage`) — noisy for prod
5. **Fix progress reporting**: increment should happen *after* successful generation, not before

---

Configuration Addition

[MODIFY] {} package.json

Add new configuration property and test dependencies:

```
"antigravity.chromePath": {
  "type": "string",
  "default": "",
  "description": "Custom path to a Chromium-based browser executable. Leave empty for auto-detect
}
```

Add dev dependencies:

- `mocha`, `@types/mocha` — test runner
- `@vscode/test-electron` — VS Code integration test host

Add scripts:

- `"test": "node ./out/test/runTest.js"`
- `"pretest": "tsc -p tsconfig.test.json"`

Test Infrastructure

[NEW] {} tsconfig.test.json

Separate TypeScript config for tests — compiles `src/` to `out/` for Mocha (esbuild handles the main extension, but tests run through `tsc` for simplicity with `@vscode/test-electron`).

[NEW] TS src/test/runTest.ts

Bootstraps `@vscode/test-electron` to download VS Code and run the test suite.

[NEW] TS src/test/suite/index.ts

Mocha test runner entry point — globs `**/*.test.js` and runs them.

Unit Tests

[NEW] TS src/test/suite/markdownRenderer.test.ts

Comprehensive tests for `markdownRenderer.ts`:

Test Group	Cases
Alerts	Each of 5 alert types renders correct CSS class and icon; nested alerts work
Mermaid	```` ```mermaid ```` block becomes `<div class="mermaid">`
Checkboxes	`[x]`, `[/]`, `[ ]` render as correct SVG spans
File links	`file:///path/to/foo.ts` renders with TS icon badge
Diff blocks	`+` / `-` / context lines get correct CSS classes
Carousel	Slides split on `<!-- slide -->`
Basic rendering	Headers, lists, tables, inline code

[NEW] TS src/test/suite/browserFinder.test.ts

Tests for `browserFinder.ts`:

- Returns custom path when provided and exists
- Falls through to chrome-launcher
- Throws descriptive error when no browser found

[NEW] TS **src/test/suite/pdfGenerator.test.ts**

Integration tests for `PdfGenerator.ts`:

- Generates a valid PDF from a simple markdown string (check file exists and size > 0)
- Handles Mermaid-containing markdown without errors
- Respects output directory creation
- Handles empty markdown gracefully

Verification Plan

Automated Tests

1. **Compile check** — ensure no TypeScript errors:

```
npx tsc --noEmit
```

2. **Unit + Integration tests** — run the full Mocha suite:

```
npm test
```

This runs `@vscode/test-electron` which downloads a VS Code instance and executes all `*.test.ts` files. Expected: all tests pass.

3. **Build check** — ensure esbuild bundles successfully:

```
npm run build
```

Manual Verification

1. **Single file export**: Open any `.md` file → click "Export PDF" status bar button → verify PDF opens correctly with all formatting (alerts, code blocks, checkboxes)
2. **Batch export**: Run "Antigravity: Export All" command → verify progress bar increments correctly and all PDFs generate in the export directory
3. **Browser reuse**: During "Export All" with 3+ files, verify only one Chrome process spawns (check Activity Monitor / Task Manager)
4. **Missing browser**: Temporarily set `antigravity.chromePath` to a non-existent path with no Chrome installed — verify a clear error message appears
5. **Custom Chrome path**: Set `antigravity.chromePath` to the correct Chrome path — verify exports work

> **Note**
>
> The PDF integration tests (`pdfGenerator.test.ts`) require Chrome/Edge installed on the test machine. They will be skipped in CI environments missing a browser via a guard check.