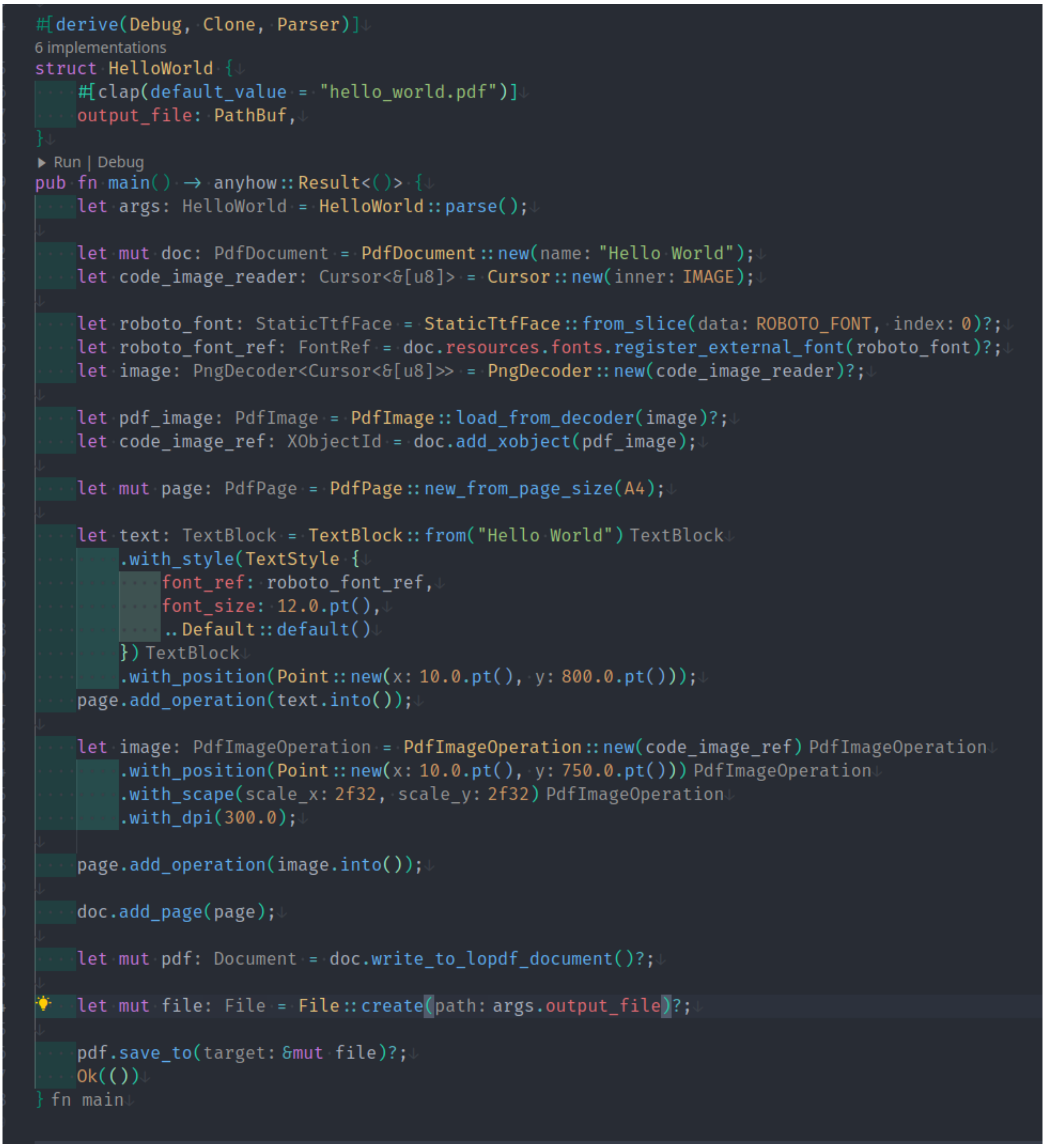
Hello World



```
#[derive(Debug, Clone, Parser)]
6 implementations
struct HelloWorld {
    #[clap(default_value = "hello_world.pdf")]
    output_file: PathBuf,
}
▸ Run | Debug
pub fn main() → anyhow::Result<()> {
    let args: HelloWorld = HelloWorld::parse();

    let mut doc: PdfDocument = PdfDocument::new(name: "Hello World");
    let code_image_reader: Cursor<&[u8]> = Cursor::new(inner: IMAGE);

    let roboto_font: StaticTtfFace = StaticTtfFace::from_slice(data: ROBOTO_FONT, index: 0)?;
    let roboto_font_ref: FontRef = doc.resources.fonts.register_external_font(roboto_font)?;
    let image: PngDecoder<Cursor<&[u8]>> = PngDecoder::new(code_image_reader)?;

    let pdf_image: PdfImage = PdfImage::load_from_decoder(image)?;
    let code_image_ref: XObjectId = doc.add_xobject(pdf_image);

    let mut page: PdfPage = PdfPage::new_from_page_size(A4);

    let text: TextBlock = TextBlock::from("Hello World") TextBlock
        .with_style(TextStyle {
            font_ref: roboto_font_ref,
            font_size: 12.0.pt(),
            ..Default::default()
        }) TextBlock
        .with_position(Point::new(x: 10.0.pt(), y: 800.0.pt()));
    page.add_operation(text.into());

    let image: PdfImageOperation = PdfImageOperation::new(code_image_ref) PdfImageOperation
        .with_position(Point::new(x: 10.0.pt(), y: 750.0.pt())) PdfImageOperation
        .with_scape(scale_x: 2f32, scale_y: 2f32) PdfImageOperation
        .with_dpi(300.0);

    page.add_operation(image.into());

    doc.add_page(page);

    let mut pdf: Document = doc.write_to_lopdf_document()?;

    let mut file: File = File::create(path: args.output_file)?;

    pdf.save_to(target: &mut file)?;
    Ok(())
} fn main
```