

# Sprachunterstützung: Schriftarten und Text-zu-Sprache

Mein Blog unterstützt jetzt neun Sprachen: Japanisch (`ja`), Spanisch (`es`), Hindi (`hi`), Chinesisch (`zh`), Englisch (`en`), Französisch (`fr`), Deutsch (`de`), Arabisch (`ar`) und traditionelles Chinesisch (`hant`). Sie finden die Website unter <https://lzwjava.github.io>.

Bei der Handhabung mehrerer Sprachen in einer Computerumgebung müssen mehrere Aspekte berücksichtigt werden.

## Schriftartenbehandlung

Verschiedene Sprachen erfordern spezifische Schriftarten für eine korrekte Darstellung, insbesondere bei der Generierung von PDFs mit LaTeX. Der folgende Python-Code zeigt, wie geeignete Schriftarten basierend auf dem Betriebssystem und der Sprache ausgewählt werden:

```
if platform.system() == "Darwin":
    if lang == "hi":
        CJK_FONT = "Kohinoor Devanagari"
    elif lang == "ar":
        CJK_FONT = "Geeza Pro"
    elif lang in ["en", "fr", "de", "es"]:
        CJK_FONT = "Helvetica"
    elif lang == "zh":
        CJK_FONT = "PingFang SC"
    elif lang == "hant":
        CJK_FONT = "PingFang TC"
    elif lang == "ja":
        CJK_FONT = "Hiragino Sans"
    else:
        CJK_FONT = "Arial Unicode MS"
else:
    if lang == "hi":
        CJK_FONT = "Noto Sans Devanagari"
    elif lang == "ar":
        CJK_FONT = "Noto Naskh Arabic"
    elif lang in ["en", "fr", "de", "es"]:
        CJK_FONT = "DejaVu Sans"
    elif lang == "zh":
        CJK_FONT = "Noto Sans CJK SC"
    elif lang == "hant":
        CJK_FONT = "Noto Sans CJK TC"
```

```

elif lang == "ja":
    CJK_FONT = "Noto Sans CJK JP"
else:
    CJK_FONT = "Noto Sans"
command = [
    'pandoc',
    input_markdown_path,
    '-o', output_pdf_path,
    '-f', 'markdown',
    '--pdf-engine', 'xelatex',
    '-V', f'romanfont={CJK_FONT}',
    '-V', f'mainfont={CJK_FONT}',
    '-V', f'CJKmainfont={CJK_FONT}',
    '-V', f'CJKsansfont={CJK_FONT}',
    '-V', f'CJKmonofont={CJK_FONT}',
    '-V', f'geometry:{GEOMETRY}',
    '-V', 'classoption=16pt',
    '-V', 'CJKoptions=Scale=1.1',
    '-V', 'linestretch=1.5'
]

```

Es ist wichtig zu beachten, dass diese Lösung nicht perfekt ist. Zum Beispiel könnte Hindi-Text in Kommentaren von Codeblöcken nicht wie erwartet dargestellt werden.

## Text-zu-Sprache

Ich nutze Google Text-to-Speech, um Audio-Versionen meiner Blogbeiträge zu erstellen. Der folgende Codeausschnitt zeigt, wie ich den entsprechenden Sprachcode für die Text-zu-Sprache-Engine auswähle:

```

if filename.endswith('-zh.md'):
    language_code = "cmn-CN"
    voice_language_code = "cmn-CN"
else:
    language_code = "en-US"
    voice_language_code = "en-US"

text_to_speech(
    text=article_text,
    output_filename=output_filename,
    task=task,

```

```
        language_code=language_code,  
        dry_run=dry_run,  
        progress=progress  
    )
```

Derzeit wird Audio für chinesische und englische Inhalte generiert. Um die Unterstützung auf andere Sprachen auszudehnen, müssen die entsprechenden Sprachcodes konfiguriert werden.