KAI

Reguläre Ausdrücke (Regular Expressions, kurz **RegEx**) werden häufig verwendet, um **HTML-Tags** zu erkennen, zu extrahieren, zu verarbeiten oder zu validieren. Im Folgender Beispiele, die zeigen, wie reguläre Ausdrücke für HTML-Tags genutzt werden können, sowie Hinweise auf ihre Grenzen.

---

1. Einfache Erkennung von HTML-Öffnungstags

Dieser reguläre Ausdruck erkennt **öffnende HTML-Tags** (z. B. `<div>`, `<p>`, `<h1>`), unabhängig von den konkreten Tag-Namen.

RegEx:

```
<\s*[a-zA-Z]+\s*>
```

Erklärung:

- `<`: Start mit einem `<`-Zeichen (Tag-Start).
- `\s*`: Erlaubt beliebige Leerzeichen nach dem `<`.
- `[a-zA-Z]+`: Erfasst den Namen des Tags (z. B. `div`, `h1`, `p`), bestehend aus Buchstaben.
- `\s*`: Optionale Leerzeichen dürfen innerhalb des Tags stehen (z. B. `< div >`).
- `>`: Schließt das Tag mit einem `>`.

Beispiele:

- **Erkannt:** `<div>`, `<h1>`, `<span>`
- **Nicht erkannt:** `</div>` (Schließen), `<img src="x.jpg">` (enthält Attribute).

---

2. Erkennung von kompletten HTML-Tags (mit oder ohne Attribute)

Dieser reguläre Ausdruck erkennt **vollständige HTML-Tags**, einschließlich solcher mit Attributen.

RegEx:

```
<\s*[a-zA-Z]+(\s+[a-zA-Z-]+\s*=\s*".*?")*\s*>
```

Erklärung:

- `<\s*[a-zA-Z]+`: Erkennung des Tag-Namens, wie oben beschrieben.
- `(\s+[a-zA-Z-]+\s*=\s*".*?")*`: Optionaler Attributteil:
  - `\s+[a-zA-Z-]+`: Ein Attributname (z. B. `class`, `id`, `src`).
  - `\s*=\s*`: Optional Leerzeichen um das Gleichheitszeichen.
  - `".*?"`: Gültiger Attributwert in Anführungszeichen.
- `\s*>`: Schließt das Tag.

Beispiele:

- **Erkannt:** `<div class="container">`, `<img src="image.jpg">`, `<input type="text" value="Hello">`
- **Nicht erkannt:** `<div >some text<div>` (enthält geschachtelte Inhalte, für die andere Ansätze notwendig sind).

---

3. Erkennung von HTML-Schließtags

Dieser reguläre Ausdruck erkennt **schließende Tags** wie `</div>`, `</p>` usw.

RegEx:

```
</\s*[a-zA-Z]+\s*>
```

Erklärung:

- `</`: Ein Schließen beginnt mit `</`.
- `[a-zA-Z]+`: Der Name des Tags (wie `div`, `h1`, `body`).
- `\s*>`: Optional Leerzeichen und das Zeichen `>` schließen das Tag.

Beispiele:

- **Erkannt:** `</div>`, `</p>`, `</body>`
- **Nicht erkannt:** `<div>` (öffnendes Tag).

---

4. Erkennung von Inhalten innerhalb von HTML-Tags

Dieser reguläre Ausdruck erkennt Inhalte zwischen einem **öffnenden und einem schließenden Tagpaar**, beispielsweise den Text zwischen `<p>` und `</p>`.

RegEx:

```
<\s*[a-zA-Z]+\s*>(.*?)</\s*[a-zA-Z]+\s*>
```

Erklärung:

- `<\s*[a-zA-Z]+\s*>`: Öffnendes HTML-Tag.
- `(.*?)`: Erfasst alles innerhalb der Tags (inklusive Leerzeichen); `?` stellt sicher, dass der "greedy mode" deaktiviert wird und der Reguläre Ausdruck nicht zu viel "schluckt"
- `</\s*[a-zA-Z]+\s*>`: Passendes schließendes HTML-Tag.

Beispiele:

- **Erkannt:** Hello World , <h1>Title</h1>
  - Erfasst dabei: „Hello World“, „Title“.
- **Nicht erkannt:** Ungeschlossene Tags wie Hello.

---

5. Extrahieren nur bestimmter Tags (z. B. <a>-Tags)

Dieser reguläre Ausdruck extrahiert nur <a>-Tags (Hyperlinks), einschließlich ihrer Inhalte und Attribute.

RegEx:

```
<\s*a\b[^>]*>(.*?)</\s*a\s*>
```

Erklärung:

- <\s*a\b: Sucht explizit nach <a>-Tags (\b = Wortgrenze, um Tags wie <ab> auszuschließen).
- [^>]*: Erfasst alle Zeichen innerhalb des a-Tags, außer >, um Attribute mitzunehmen.
- (.*?): Erfasst die Inhalte (Text oder HTML) zwischen den Öffnungs- und Schließtags.
- </\s*a\s*>: Beendet das <a>-Tag.

Beispiele:

- **Erkannt:** Click Me
  - Erfasst den Inhalt: „Click Me“.
- **Nicht erkannt:** Andere Tags (z. B. oder).

---

6. Erkennung von selbstschließenden HTML-Tags

Dieser reguläre Ausdruck erkennt **selbstschließende HTML-Tags**, z. B. <img />,
.

RegEx:

```
<\s*[a-zA-Z]+\s*[^>]*\/?>
```

Erklärung:

- <\s*[a-zA-Z]+: Erkennung des Tag-Namens.
- [^>]*: Optionale Attribute innerhalb des Tags.
- /?>: Abschließendes > oder selbstschließendes />.

Beispiele:

- **Erkannt:** ,
, <input type="text" />
- **Nicht erkannt:** Öffnungsschließ-Paare wie ... .

---

7. Komplettes HTML-Dokument finden (Tags und Inhalte ignorieren)

Dieser reguläre Ausdruck matcht den groben Aufbau von **HTML-Dokumenten**, beginnend vom <html>-Tag bis zum </html>.

RegEx:

```
<\s*html\b[^>]*>(.*?)<\s*/\s*html\s*>
```

Erklärung:

- <\s*html\b[^>]*>: Öffnendes <html>-Tag mit optionalen Attributen.
- (.*?): Alles innerhalb des <html>-Tags (HTML-Inhalt).
- <\s*/\s*html\s*>: Schließendes </html>-Tag.

Beispiele:

- **Erkannt:**

```
<html>
  <head><title>Test</title></head>
  <body>Hello</body>
</html>
```

- **Nicht erkannt:** Dokumente ohne schließendes </html>.

---

8. Erkennung aller HTML-Tags im Dokument

Dieser RegEx matcht **alle** HTML-Tags, unabhängig davon, ob sie öffnend, schließend oder selbstschließend sind.

RegEx:

```
<[^>]+>
```

Erklärung:

- <: Start eines HTML-Tags.
- [^>]+: Erfasst alle Zeichen bis zu einem >.
- >: Schließen des Tags.

Beispiele:

- **Erkannt:** `<div>`, `<img src="x.png" />`, `</body>`.
- **Nicht erkannt:** Text außerhalb von Tags, wie „Hello World“.

HINWEIS: Grenzen von RegEx bei HTML

Während Reguläre Ausdrücke nützlich für schnelle, einfache Aufgaben wie **Erkennen und Extrahieren von Tags** oder einfachen **Validierungen** sind, sollten komplexe Aufgab mit spezialisierten **HTML-Parsern** (z. B. BeautifulSoup in Python oder DOM-Parser in JavaScript) erledigt werden. HTML ist rekursiv und erlaubt verschachtelte Strukturen, die RegEx zu verarbeiten sind.