DEV REFERENCE / PYTHON

python regex cheat sheet

Patterns, methods, and flags for Python's `re` module.

BASIC SYNTAX

PATTERN	DESCRIPTION	EXAMPLE
`.`	Any character except newline	`a.c` matches `abc` , `axc`
`^`	Start of string	`^Hello` must start with Hello
`$`	End of string	`world$` must end with world
`*`	0 or more of preceding	`a*` → "", a, aa, ...
`+`	1 or more of preceding	`a+` → a, aa, aaa, ...
`?`	0 or 1 of preceding	`colou?r` → color, colour
`{n}`	Exactly n occurrences	`a{3}` → aaa
`{n,}`	n or more occurrences	`a{2,}` → aa, aaa, ...
`{n,m}`	Between n and m occurrences	`a{2,4}` → aa, aaa, aaaa
`[]`	Any one char inside brackets	`[abc]` → a, b, or c
`[^ ]`	Any char NOT in brackets	`[^abc]` → anything else
`\`	Escape special character	`\.` → literal dot
`|`	Logical OR	`a|b` → a or b
`()`	Capture group	`(ab)+` → ab, abab, ...

SPECIAL SEQUENCES

PATTERN	DESCRIPTION	EXAMPLE
`\d`	Any digit — `[0-9]`	matches 3 in abc123
`\D`	Any non-digit	matches a in a123
`\w`	Alphanumeric + underscore	matches a, 1, _
`\W`	Non-alphanumeric	matches ! in hello!
`\s`	Whitespace (space, tab, newline)	matches space, \t, \n
`\S`	Non-whitespace	matches h in hello world
`\b`	Word boundary	`\bword\b` → word, not sword
`\B`	Non-word boundary	`\Bword` matches sword

RE MODULE METHODS

METHOD	DESCRIPTION	EXAMPLE
`re.match()`	Match at start of string only	`re.match(r"\d+", "123abc")` → 123
`re.search()`	First match anywhere in string	`re.search(r"\d+", "abc123")` → 123
`re.findall()`	List of all matches	`re.findall(r"\d+", "a1b2")` → ['1','2']
`re.finditer()`	Iterator of match objects	`re.finditer(r"\d+", "a1b2")`
`re.sub()`	Replace matches with string	`re.sub(r"\d+", "#", "a1b2")` → a#b#
`re.split()`	Split string by pattern	`re.split(r"\s+", "a b")` → ['a','b']
`re.compile()`	Compile pattern for reuse	`p = re.compile(r"\d+")`

FLAGS

FLAG	DESCRIPTION	EXAMPLE
`re.I`	Case-insensitive matching	`re.search(r"hi", "HI", re.I)`
`re.M`	`^` / `$` match per line	`re.search(r"^hi", "hi\nbye", re.M)`

FLAG	DESCRIPTION	EXAMPLE
re.S	. matches newlines too	re.search(r".+", "a\nb", re.S)
re.X	Allow whitespace & comments in pattern	re.search(r"\d+ # digits", "3", re.X)

USAGE EXAMPLES

python

```python
import re

# Match at start of string
result = re.match(r"^Hello", "Hello World")
print(result.group())  # Hello

# Find first occurrence anywhere
result = re.search(r"\bword\b", "A word in a sentence")
print(result.group())  # word

# Return all matches as a list
numbers = re.findall(r"\d+", "Price: 123 dollars, 456 euros")
print(numbers)  # ['123', '456']

# Replace matches
result = re.sub(r"\s+", "-", "Hello World! How are you?")
print(result)  # Hello-World!-How-are-you?

# Split on pattern
words = re.split(r"\s+", "Hello World! How are you?")
print(words)  # ['Hello', 'World!', 'How', 'are', 'you?']
```

COMMON PATTERNS

- Email: `r'^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$'`
- URL domain: `r'https?://(www\.)?([^/]+)'`
- Phone number: `r'\b\d{3}[-.]?\d{3}[-.]?\d{4}\b'`

TIPS

- Always use raw strings (`r"..."`) to avoid double-escaping backslashes.
- Use `re.compile()` when reusing a pattern — it compiles once and is faster at scale.
- Test patterns interactively at regex101.com before embedding them in code.
