[redacted]

[redacted] vám jako první přiblížení pomoci [redacted] (ikdyž za chvíli mladší ročníky už [redacted] nástrojů. Regulární výrazy disponují [redacted] regulárních výrazech [redacted]

[redacted]

[redacted] právě jeden **libovolný znak**.

[redacted] (quantifiers) – ty určují kolikrát se [redacted] kvantifikátory se podíváme níže.

[redacted] **skupiny znaků**, uzavřeme takovou [redacted] naopak definovat, že v daném [redacted] znaků, uzavřeme „zakázané" znaky [redacted] `[^ab]`, tak bude znamenat, že se [redacted] znaků `a` a `b`.
[redacted] (například zápis `\d` zastupuje všechny číslice 0-9) – viz tabulku níže.
Pokud chceme definovat skupinu znaků, které následují v abecedě (resp. přesněji v tabulce znaků) za sebou, můžeme je zapsat jako interval, například `[1-5]`, `[a-e]` nebo třeba `[A-Z]`

- Pokud potřebujeme zajistit, opakování určité sekvence znaků (ne jen znaku jednoho), můžeme sekvenci znaků uzavřít do závorek (`(` a `)`) a pokud za pravou kulatou závorku doplníme kvantifikátor, bude se počet opakování vztahovat na celou sekvenci znaků uzavřenou do závorek.

- Pokud chceme dát na výběr několik variant textu (třeba `Petr` nebo `Pavel`), jako oddělovač variant použijeme metaznak `|` (výraz bude tedy `Petr|Pavel`)

- Pokud chceme přikázat, že hledaný textový řetězec se musí nacházet na začátku nebo konci prohledávaného textu, použijeme metaznaky, které nazýváme **hranice** (boundaries) nebo **ukotvení** (anchors) – na různé typy hranic se podíváme níže.

Následující tabulky zobrazují seznam kvantifikátorů a hranic.

Kvantifikátory

Kvantifikátor	Počet opakování
`?`	minimálně 0krát, maximálně 1krát

`*`	minimálně 0krát (maximálně neomezeno)
`+`	minimálně 1krát (maximálně neomezeno)
`{n}`	právě *n*krát
`{m,n}`	minimálně *m*krát, maximálně *n*krát
`{m,}`	minimálně *m*krát (maximálně neomezeno)

Pro pořádek ještě dopním, že existují tzv. **liné kvantifikátory** – ty se od výše zmíněných (tzv. nenasytných) liší v zápisu tak, že výše uvedený kvantifikátor zprava doplníme o otazník (`?`). Líné kvantifikátory tedy budou `??`, `*?`, `+?`, `{m,n}?` a `{m,}?`.

Funkčně se budou líné kvantifikátory (od v tabulce uvedených nenasytných kvantifikátorů) lišit v tom, že pomocí líných kvantifikátorů je zachycen minimální počet znaků, které je třeba zachytit, aby došlo ke shodě s regulárním výrazem. Nenasytné kvantifikátory naopak zachytí co možná největší počet znaků vstupního textu.

Předdefinované skupiny znaků

`\d`	číslice `0-9`
`\D`	jakýkoliv znak kromě číslic `0-9`
`\w`	znaky „slova” (ekvivalentní zápisu `[a-zA-Z0-9_]`)
`\W`	jakýkoliv znak kromě znaků „slova” (ekvivalentní zápisu `[^a-zA-Z0-9_]`)
`\s`	„bílé” znaky (mezera, tabulátor, znaky pro zalomení řádků)
`\S`	jakýkoliv znak kromě „bílých” znaků

Hranice

`^`	začátek řetězce (textu v němž se vyhledává)
`$`	konec řetězce (textu v němž se vyhledává)

Původní význam speciálních znaků (metaznaků)

Možná vás napadlo, že když určité znaky (metaznaky) mají v regulárním výrazu zvláštní význam, jak je možné takový znak zapsat tak, aby nebyl chápán jako metaznak, ale jako obyčejný znak (třeba plus či hvězdička). Řešení je prosté – stačí před inkriminovaný znak dopnit v regulárním výrazu zpětné lomítko `\`. Pokud chcete například pomocí regulárního výrazu popsat rovnici `(a+b)*c=d`, je třeba použít regulární výraz `(a\+b)\*c=d`.

Které znaky je třeba dopnit oním zpětným lomítkem (tzv. escapovat)? Mezi metaznaky patří `\`, `^`, `$`, `.`, `[`, `]`, `|`, `(`, `)`,`?`, `*`, `+`, `{`, `}`.

Příklady

Použití všech zmíněných metaznaků nejlépe pochopíte na několika příkladech.

Regulární výraz	Odpovídá...
`a+`	sekvence písmen `a` (1 a více znaků)
`a*`	sekvence písmen `a` (0 a více znaků)
`o?kov`	`okov` či `kov`
`tel(efon)?`	`tel` či `telefon`
`telef(on|ax)`	`telefon` či `telefax`
`[0-9]|[1-9][0-9]`	čísla 0 až 99
`\d{2}`	sekvence dvou číslic desítkové soustavy (`00`, `01`, ...,`98`, `99`)
`[0-9a-fA-F]|[1-9a-fA-F][0-9a-fA-F]+`	hexadecimální čísla
`(19|20)\d{2}`	letopočty 1900-2099
`\d{2,6}`	sekvence dvou až šesti číslic
`[^ ,.]+`	neprázdná sekvence znaků mezi nimiž nesmí být mezera ( ), čárka (`,`) či tečka (`.`)
`^P.*`	řetězec, který začíná písmenem `P` za nímž následuje libovolný (i nulový) počet libovolných znaků
`\d+0$`	řetězec, který končí znakem `0` (nula), kterému předchází minimálně jedna číslice
`a+b`	`ab`, `aab`, `aaab` atd.
`a\+b`	`a+b`

Poznámky závěrem

- Popsané konstrukce odpovídají regulárním výrazům vycházejícím z Perlu – budou tedy fungovat především v Perlu, .NETu, PHP (při použití Perl-compatible regular expressions funkcí) a Javascriptu.

- V tomto úvodním článku nebyly zmíněny pokročilejší konstrukce regulárních výrazů jako zpětné odkazy (backreferences), modifikátory (modifiers), pokročilejší hranice, komentáře, tvrzení (assertions) nebo podmíněné subvýrazy (conditional subexpressions). Některé pokročilejší konstrukce jsou závislé na konkrétní implementaci regulárních výrazů v daném programovacím jazyce, proto jsou popsány ve zvláštních tutoriálech (seriálech článků).