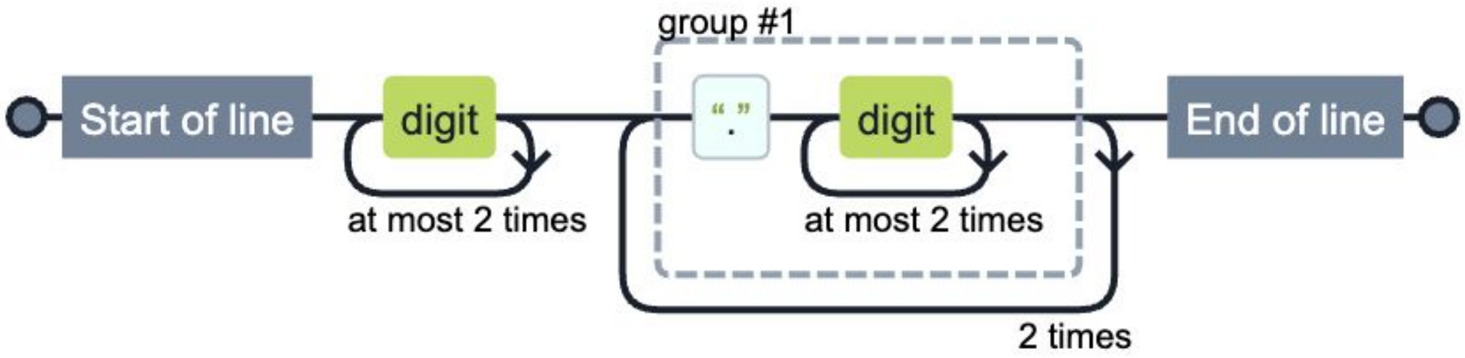
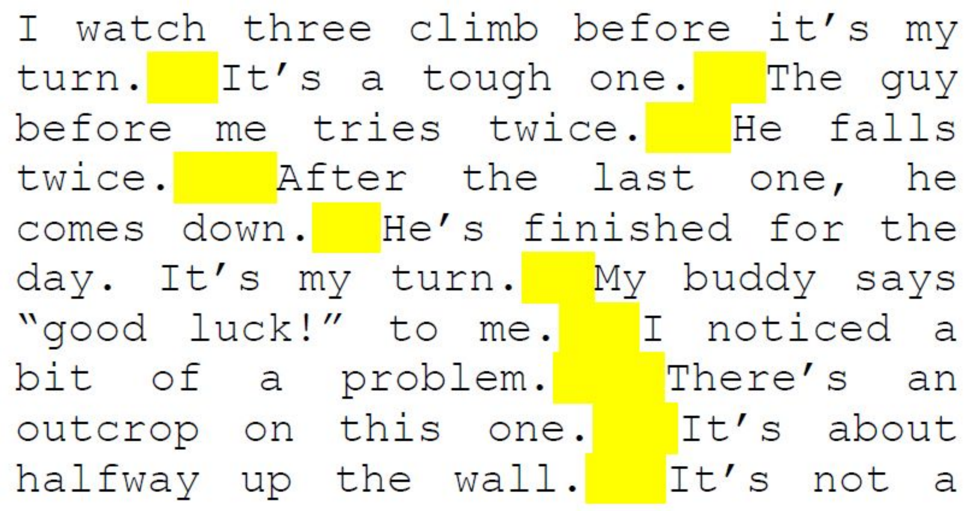
Regular Expressions
Part 1 - Introduction

```
^\d{1,3}(\.\d{1,3}){3}$
```





```
I watch three climb before it's my
turn.  It's a tough one.  The guy
before me tries twice.  He falls
twice.  After the last one, he
comes down.  He's finished for the
day. It's my turn.  My buddy says
"good luck!" to me.  I noticed a
bit of a problem.  There's an
outcrop on this one.  It's about
halfway up the wall.  It's not a
```

```
(?<=\.) {2,}(?=[A-Z])
```

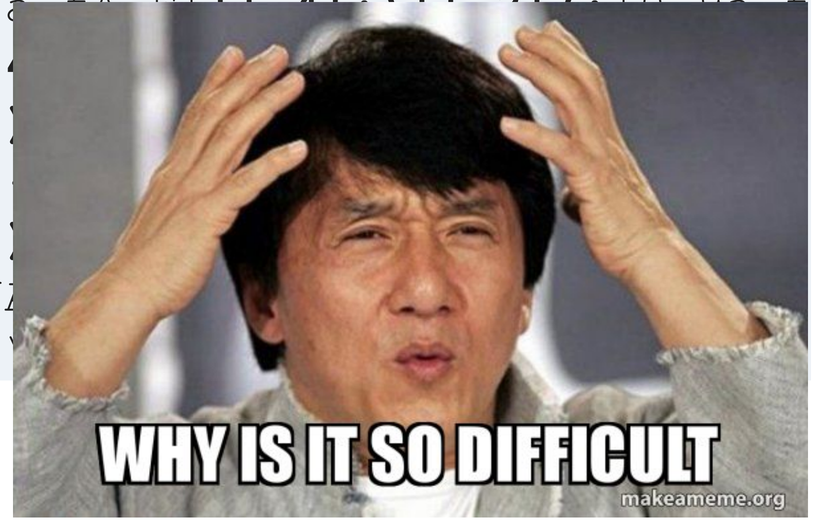
At least two spaces are matched, but only if they occur directly after a period (.) and before an uppercase letter.

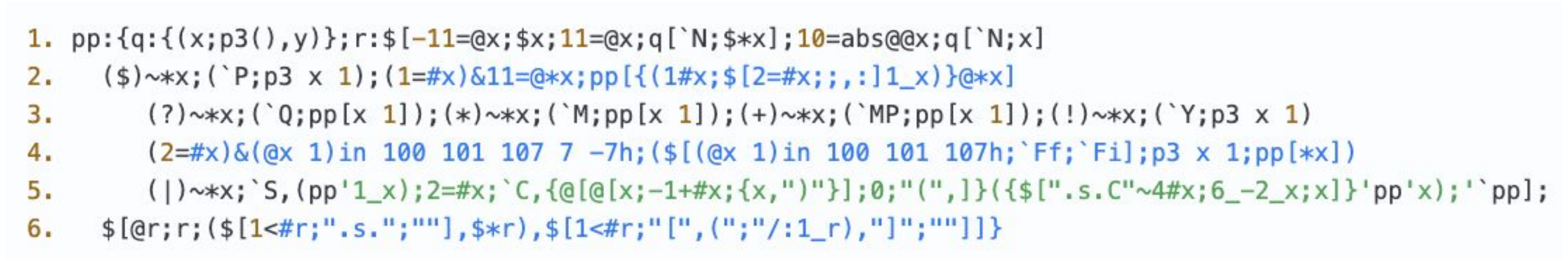
IPv6 RegExp:

```
(([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4
}:){1,7}:|([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|([0-9a
-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|([0-9a-fA-F]{1,
4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|([0-9a-fA-F]{1,4}:){1,3}
(:[0-9a-fA-F]{1,4}){1,4}|([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-f
A-F]{1,4}){1,5}|[0-9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6}
)|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-F]{0,4}){0
,4}%[0-9a-zA-Z]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|
(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{
0,1}[0-9]){0,1}[0-9])|([0-9a-fA-F]{1,4}:){1,4}:((25[0-5]|(
2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{0
,1}[0-9]){0,1}[0-9]))
```

IPv6 RegExp:

```
(([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4
}:){1,7}:|([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|([0-9a
-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|([0-9a-fA-F]{1,
4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|([0-9a-fA-F]{1,4}:){1,3}
(:[0-9a-fA-F]{1,4}){1,4}|([0-9a
A-F]{1,4}){1,5}|[0-9a-fA-F]{1,
)|:((:[0-9a-fA-F]{1,4}){1,7}|:
,4}%[0-9a-zA-Z]{1,}|::(ffff(:0
(2[0-4]|1{0,1}[0-9]){0,1}[0-9]
0,1}[0-9]){0,1}[0-9])|([0-9a-f
2[0-4]|1{0,1}[0-9]){0,1}[0-9])
,1}[0-9]){0,1}[0-9]))
```





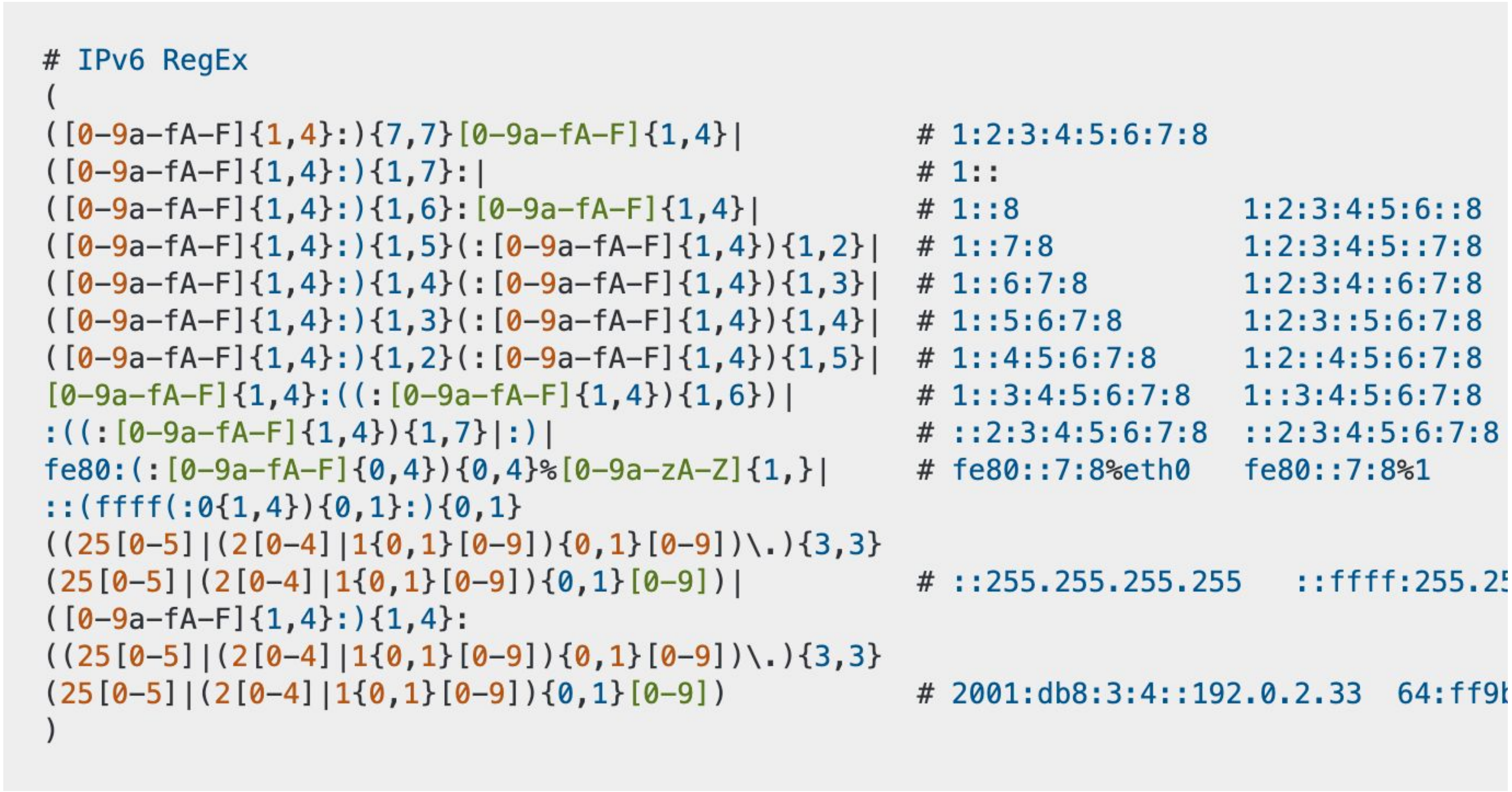
```
pp:{q:{(x;p3(),y)};r:$[-11=@x;$x;11=@x;q[`N;$*x];10=abs@@x;q[`N;x]
  ($)~*x;(`P;p3 x 1);(1=#x)&11=@*x;pp[{(1#x;$[2=#x;;,:]1_x)}@*x]
     (?)~*x;(`Q;pp[x 1]);(*)~*x;(`M;pp[x 1]);(+)~*x;(`MP;pp[x 1]);(!)~*x;(`Y;p3 x 1)
     (2=#x)&(@x 1)in 100 101 107 7 -7h;($[(@x 1)in 100 101 107h;`Ff;`Fi];p3 x 1;pp[*x])
     (|)~*x;`S,(pp'1_x);2=#x;`C,{@[@[x;-1+#x;{x,")"}];0;"(",]}({$[".s.C"~4#x;6_-2_x;x]}'pp'x);'`pp];
  $[@r;r;($[1<#r;".s.";""],$*r),$[1<#r;"[",(";"/:1_r),"]";""]]}
```

```
/**
 * A function in K that
 * implements most of the LL1
 * parser generator for a given
 * grammar
 */
```

```
pp:{q:{(x;p3(),y)};r:$[-11=@x;$x;11=@x;q[`N;$*x];10=abs@@x;q[`N;x]
  ($)~*x;(`P;p3 x 1);(1=#x)&11=@*x;pp[{(1#x;$[2=#x;;,:]1_x)}@*x]
     (?)~*x;(`Q;pp[x 1]);(*)~*x;(`M;pp[x 1]);(+)~*x;(`MP;pp[x 1]);(!)~*x;(`Y;p3 x 1)
     (2=#x)&(@x 1)in 100 101 107 7 -7h;($[(@x 1)in 100 101 107h;`Ff;`Fi];p3 x 1;pp[*x])
     (|)~*x;`S,(pp'1_x);2=#x;`C,{@[@[x;-1+#x;{x,")"}];0;"(",]}({$[".s.C"~4#x;6_-2_x;x]}'pp'x);'`pp];
  $[@r;r;($[1<#r;".s.";""],$*r),$[1<#r;"[",(";"/:1_r),"]";""]]}
```

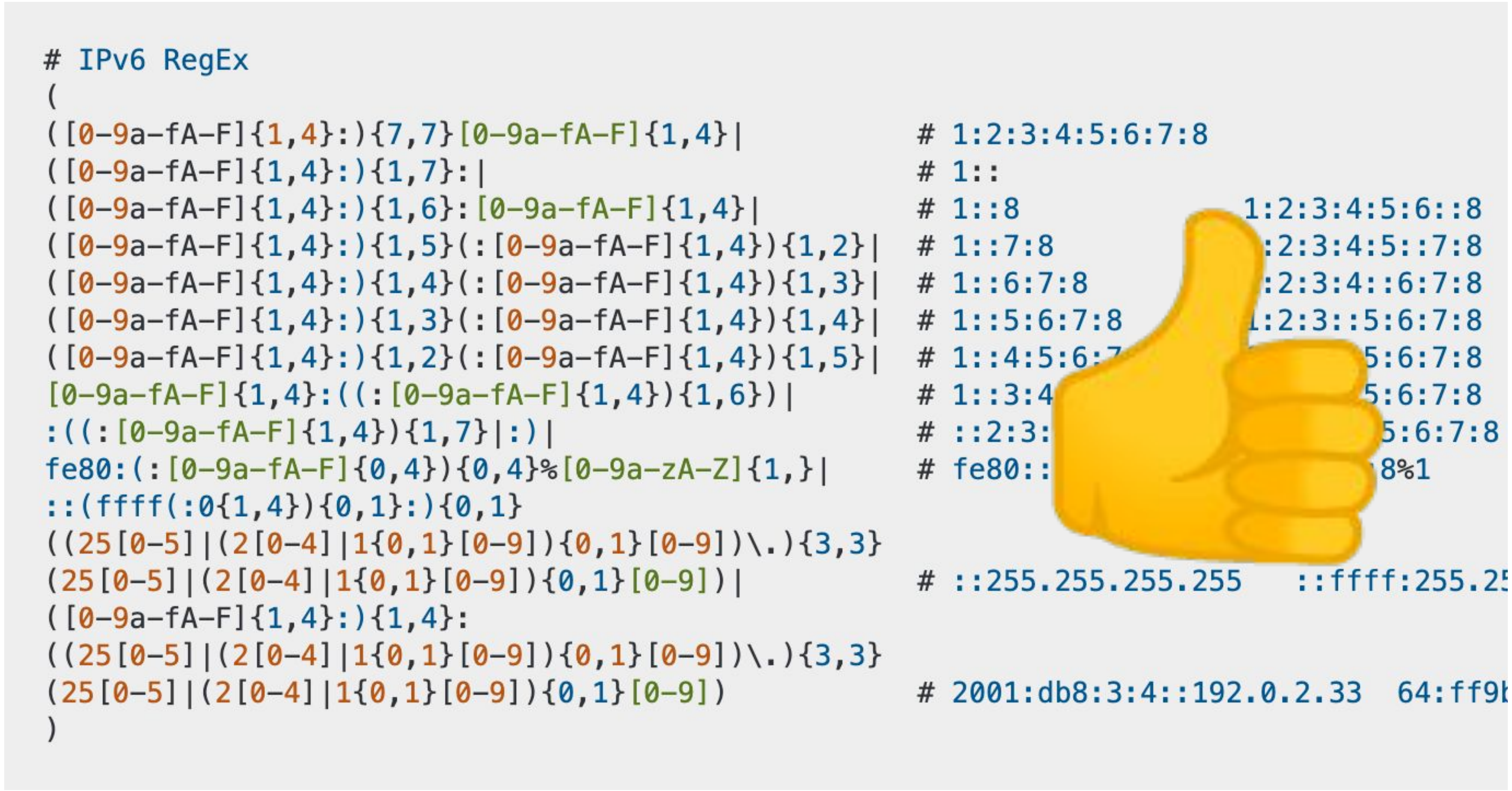
```
/**
 * A function in K that
 * implements most of the LL1
 * parser generator for a given
 * grammar
 */
```

IPv6 RegExp:



```
# IPv6 RegEx
(
([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|          # 1:2:3:4:5:6:7:8
([0-9a-fA-F]{1,4}:){1,7}:|                         # 1::
([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|         # 1::8             1:2:3:4:5:6::8
([0-9a-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|  # 1::7:8           1:2:3:4:5::7:8
([0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|  # 1::6:7:8         1:2:3:4::6:7:8
([0-9a-fA-F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|  # 1::5:6:7:8       1:2:3::5:6:7:8
([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|  # 1::4:5:6:7:8     1:2::4:5:6:7:8
[0-9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|       # 1::3:4:5:6:7:8   1::3:4:5:6:7:8
:((:[0-9a-fA-F]{1,4}){1,7}|:)|                     # ::2:3:4:5:6:7:8  ::2:3:4:5:6:7:8
fe80:(:[0-9a-fA-F]{0,4}){0,4}%[0-9a-zA-Z]{1,}|     # fe80::7:8%eth0   fe80::7:8%1
::(ffff(:0{1,4}){0,1}:){0,1}
((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}
(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|          # ::255.255.255.255   ::ffff:255.25
([0-9a-fA-F]{1,4}:){1,4}:
((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}
(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])           # 2001:db8:3:4::192.0.2.33  64:ff9
)
```

IPv6 RegExp:

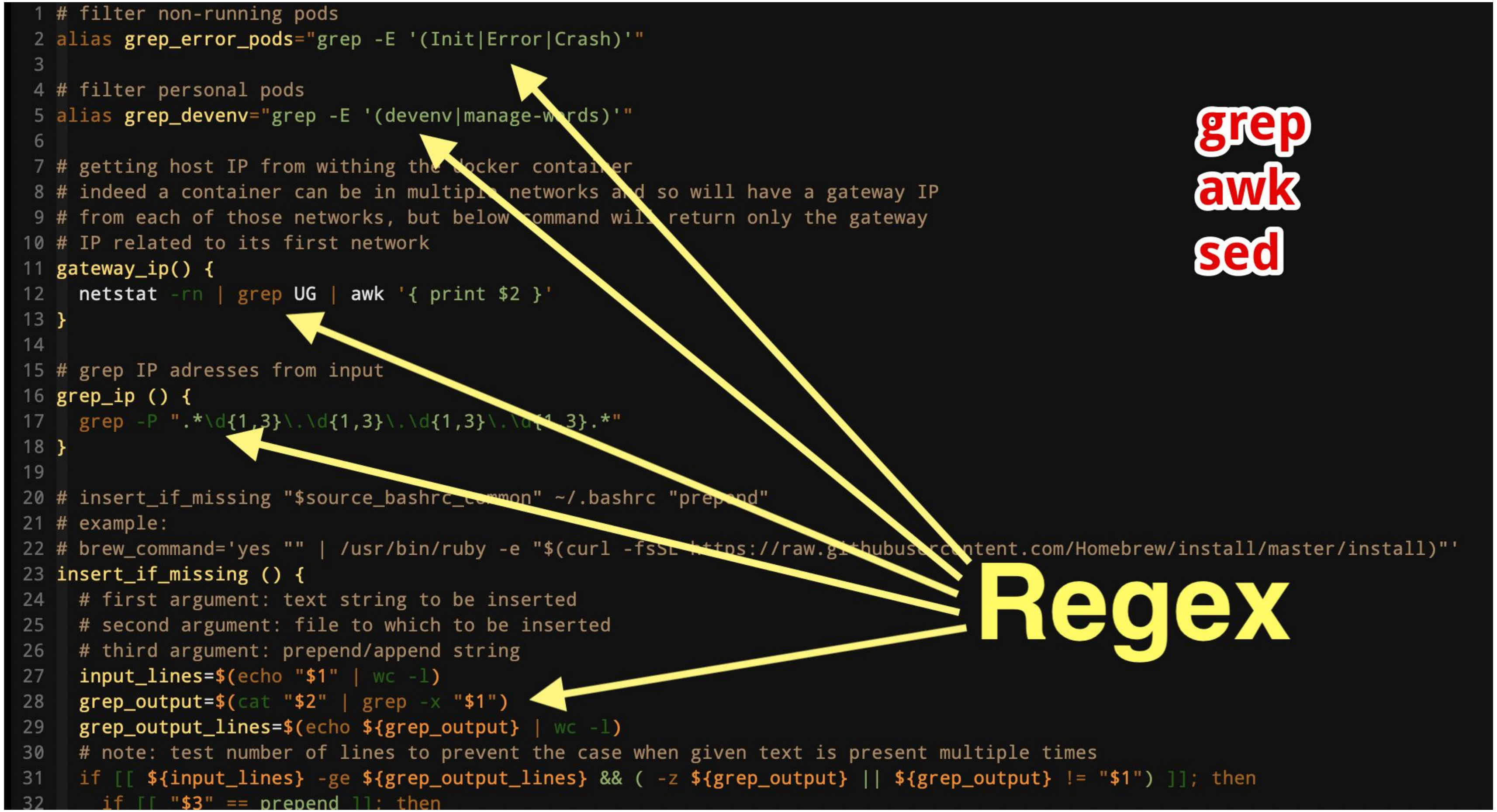


```
# IPv6 RegEx
(
([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|          # 1:2:3:4:5:6:7:8
([0-9a-fA-F]{1,4}:){1,7}:|                         # 1::
([0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|         # 1::8               1:2:3:4:5:6::8
([0-9a-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|  # 1::7:8              :2:3:4:5::7:8
([0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|  # 1::6:7:8            :2:3:4::6:7:8
([0-9a-fA-F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|  # 1::5:6:7:8          1:2:3::5:6:7:8
([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|  # 1::4:5:6:7                 5:6:7:8
[0-9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|       # 1::3:4                     5:6:7:8
:((:[0-9a-fA-F]{1,4}){1,7}|:)|                     # ::2:3:                     5:6:7:8
fe80:(:[0-9a-fA-F]{0,4}){0,4}%[0-9a-zA-Z]{1,}|     # fe80::                    8%1
::(ffff(:0{1,4}){0,1}:){0,1}
((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}
(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|          # ::255.255.255.255   ::ffff:255.25
([0-9a-fA-F]{1,4}:){1,4}:
((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}
(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])           # 2001:db8:3:4::192.0.2.33  64:ff9
)
```

Why learn RegExp?

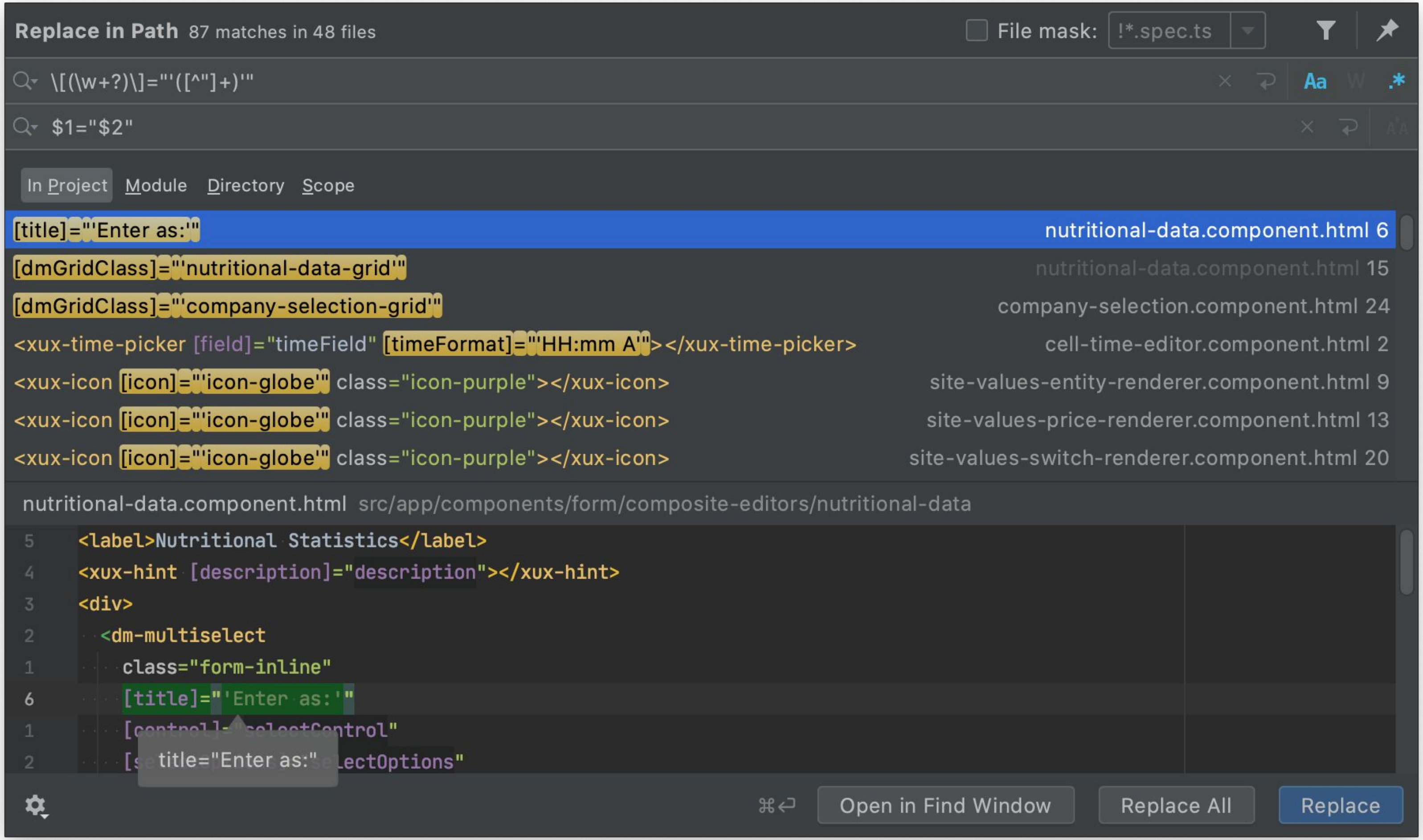
1. RegExp are everywhere

2. Automate tasks using bash



```
# filter non-running pods
alias grep_error_pods="grep -E '(Init|Error|Crash)'"

# filter personal pods
alias grep_devenv="grep -E '(devenv|manage-words)'"

# getting host IP from withing the docker container
# indeed a container can be in multiple networks and so will have a gateway IP
# from each of those networks, but below command will return only the gateway
# IP related to its first network
gateway_ip() {
  netstat -rn | grep UG | awk '{ print $2 }'
}

# grep IP adresses from input
grep_ip () {
  grep -P ".*\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}.*"
}

# insert_if_missing "$source_bashrc_common" ~/.bashrc "prepend"
# example:
# brew_command='yes "" | /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"'
insert_if_missing () {
  # first argument: text string to be inserted
  # second argument: file to which to be inserted
  # third argument: prepend/append string
  input_lines=$(echo "$1" | wc -l)
  grep_output=$(cat "$2" | grep -x "$1")
  grep_output_lines=$(echo ${grep_output} | wc -l)
  # note: test number of lines to prevent the case when given text is present multiple times
  if [[ ${input_lines} -ge ${grep_output_lines} && ( -z ${grep_output} || ${grep_output} != "$1") ]]; then
    if [[ "$3" == prepend ]]; then
```

3. Refactoring



Replace in Path 87 matches in 48 files

File mask: !*.spec.ts

```
\[(\w+?)\]="'([^"]+)'"
```

```
$1="$2"
```

In Project Module Directory Scope

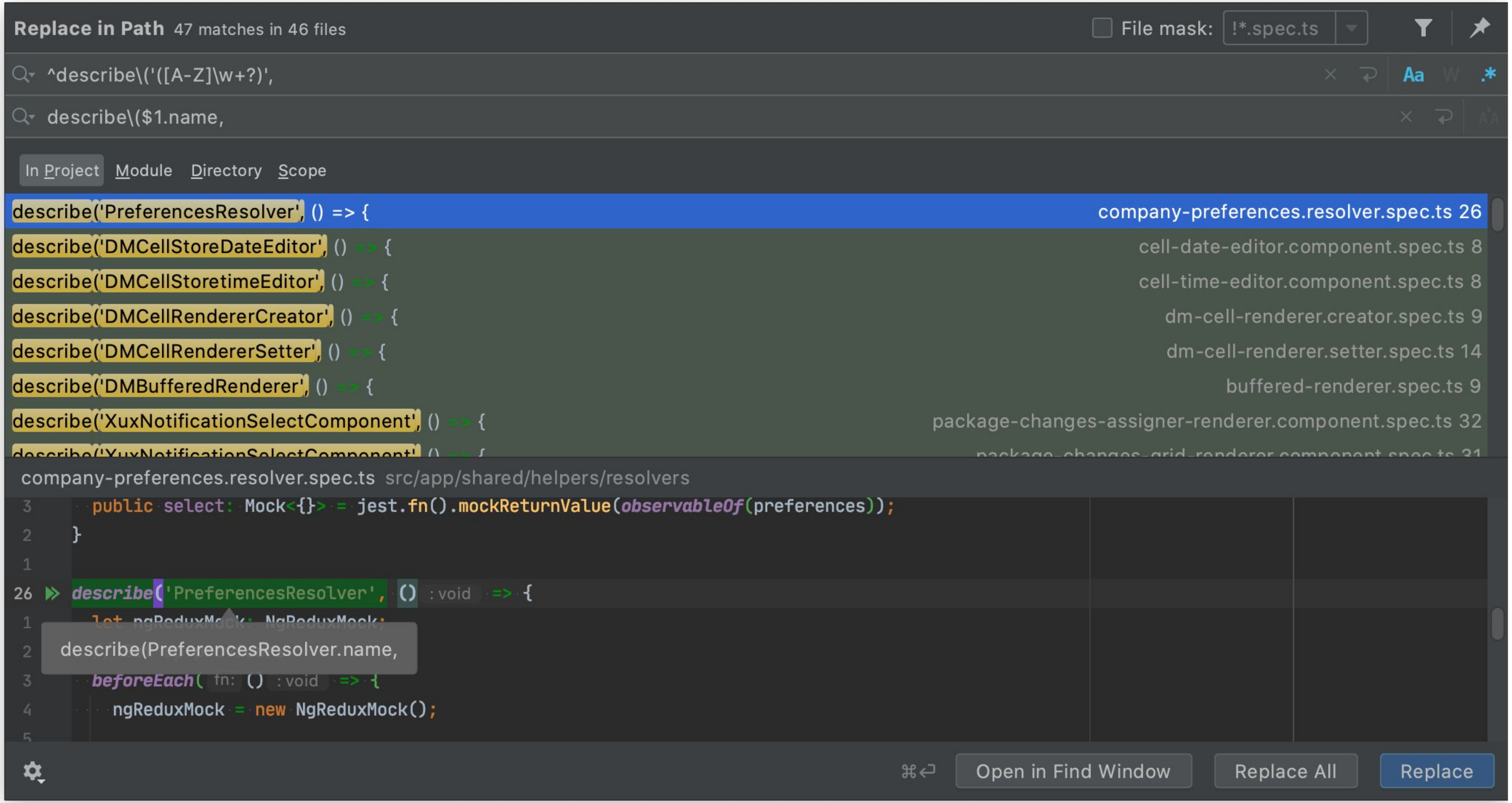
```
[title]="'Enter as:'"                                                          nutritional-data.component.html 6
[dmGridClass]="'nutritional-data-grid'"                                        nutritional-data.component.html 15
[dmGridClass]="'company-selection-grid'"                                       company-selection.component.html 24
<xux-time-picker [field]="timeField" [timeFormat]="'HH:mm A'"></xux-time-picker>  cell-time-editor.component.html 2
<xux-icon [icon]="'icon-globe'" class="icon-purple"></xux-icon>                site-values-entity-renderer.component.html 9
<xux-icon [icon]="'icon-globe'" class="icon-purple"></xux-icon>                site-values-price-renderer.component.html 13
<xux-icon [icon]="'icon-globe'" class="icon-purple"></xux-icon>                site-values-switch-renderer.component.html 20
```

nutritional-data.component.html src/app/components/form/composite-editors/nutritional-data

```
<label>Nutritional Statistics</label>
<xux-hint [description]="description"></xux-hint>
<div>
  <dm-multiselect
    class="form-inline"
    [title]="'Enter as:'"
    [control]="selectControl"
    [selectOptions]"
```

title="Enter as:"

3. Refactoring



Replace in Path 47 matches in 46 files

File mask: !*.spec.ts

```
^describe\('([A-Z]\w+?)',
```

```
describe\($1.name,
```

In Project | Module | Directory | Scope

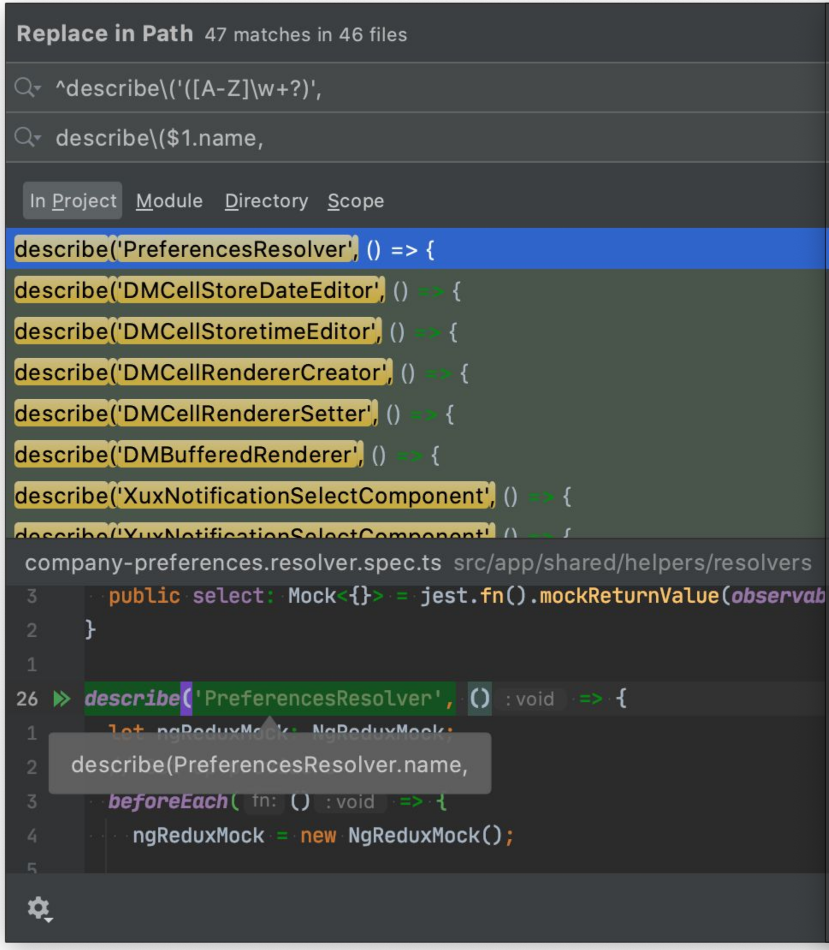
```
describe('PreferencesResolver', () => {                  company-preferences.resolver.spec.ts 26
describe('DMCellStoreDateEditor', () => {                cell-date-editor.component.spec.ts 8
describe('DMCellStoretimeEditor', () => {                cell-time-editor.component.spec.ts 8
describe('DMCellRendererCreator', () => {                dm-cell-renderer.creator.spec.ts 9
describe('DMCellRendererSetter', () => {                 dm-cell-renderer.setter.spec.ts 14
describe('DMBufferedRenderer', () => {                   buffered-renderer.spec.ts 9
describe('XuxNotificationSelectComponent', () => {       package-changes-assigner-renderer.component.spec.ts 32
describe('XuxNotificationSelectComponent', () => {       package-changes-grid-renderer.component.spec.ts 31
```

company-preferences.resolver.spec.ts src/app/shared/helpers/resolvers

```
  public select: Mock<{}> = jest.fn().mockReturnValue(observableOf(preferences));
}

describe('PreferencesResolver', () :void => {
  let ngReduxMock: NgReduxMock;

  beforeEach( fn: () :void => {
    ngReduxMock = new NgReduxMock();
```

describe(PreferencesResolver.name,

3. Refactoring



Replace in Path 47 matches in 46 files

```
^describe\('([A-Z]\w+?)',
describe\($1.name,
```

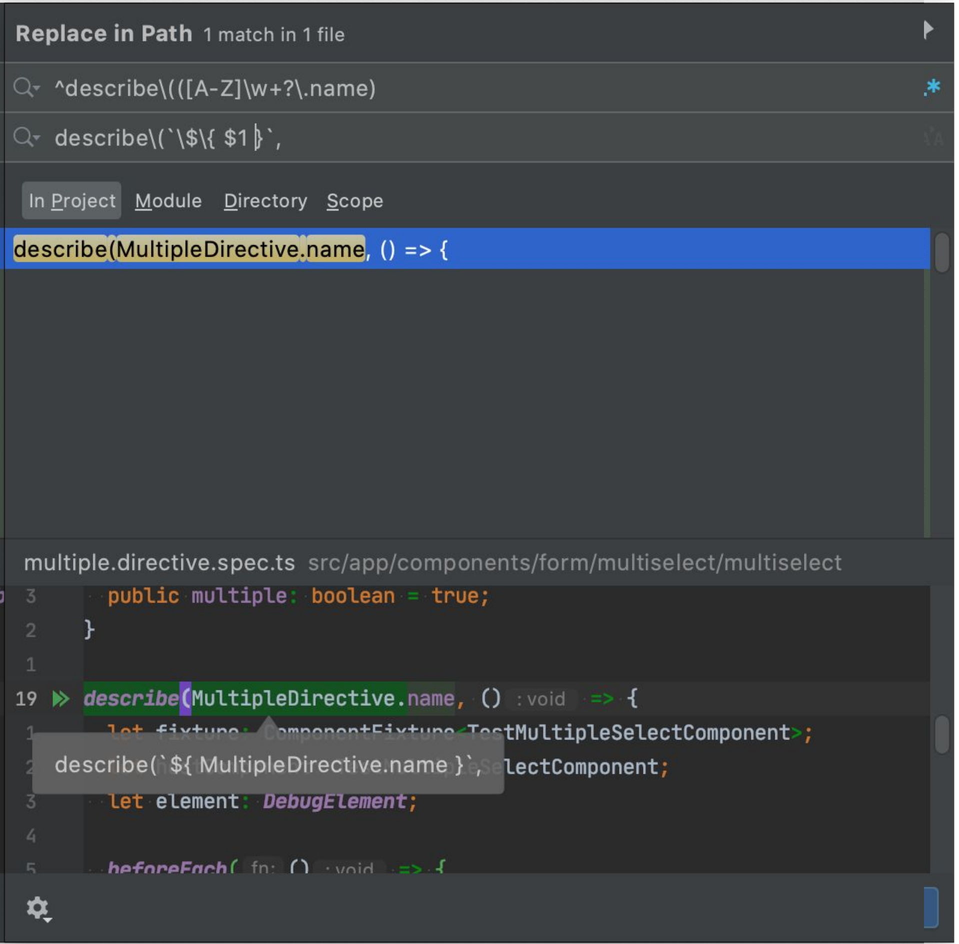
In Project | Module | Directory | Scope

```
describe('PreferencesResolver', () => {
describe('DMCellStoreDateEditor', () => {
describe('DMCellStoretimeEditor', () => {
describe('DMCellRendererCreator', () => {
describe('DMCellRendererSetter', () => {
describe('DMBufferedRenderer', () => {
describe('XuxNotificationSelectComponent', () => {
```

company-preferences.resolver.spec.ts src/app/shared/helpers/resolvers

```
  public select: Mock<{}> = jest.fn().mockReturnValue(observab
}

describe('PreferencesResolver', () :void => {
  let ngReduxMock: NgReduxMock;

  beforeEach( fn: () :void => {
    ngReduxMock = new NgReduxMock();
```

describe(PreferencesResolver.name,



Replace in Path 1 match in 1 file

```
^describe\(([A-Z]\w+?\.name)
describe\(`\$\{ $1 }`,
```

In Project | Module | Directory | Scope

```
describe(MultipleDirective.name, () => {
```

multiple.directive.spec.ts src/app/components/form/multiselect/multiselect

```
  public multiple: boolean = true;
}

describe(MultipleDirective.name, () :void => {
  let fixture: ComponentFixture<TestMultipleSelectComponent>;
  ...lectComponent;
  let element: DebugElement;

  beforeEach( fn: () :void => {
```

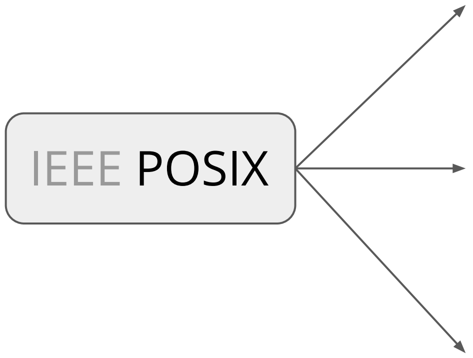
describe(`${ MultipleDirective.name }`,

4. If your search is simple, regular expression syntax is simple.
5. Regular expressions can help you write short code.
6. Regular expressions save time. ⏳
7. Regular expressions can match just about anything. 🧪
8. Regular expressions are fast. *(not always).*
9. Regular expressions can match just about anything.
10. Regular expression mastery can help you stand out from the crowd.💎
11. Regular expressions are fun. 😃

Standards:

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:



BRE - Basic Regular Expressions

SRE - Simple Regular Expressions

ERE - Extended Regular Expressions-

- adds `?`, `+`, and `|`
- removes the need to escape the metacharacters `( )` and `{ }`

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:

- IEEE POSIX
  - BRE - Basic Regular Expressions
  - SRE - Simple Regular Expressions
  - ERE - Extended Regular Expressions-
    - adds `?`, `+`, and `|`
    - removes the need to escape the metacharacters `( )` and `{ }`
- PCRE
  - **P**erl **C**ompatible **R**egular **E**xpressions

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:

IEEE POSIX

→ BRE - Basic Regular Expressions 😱🤯👎👎👎

→ SRE - Simple Regular Expressions

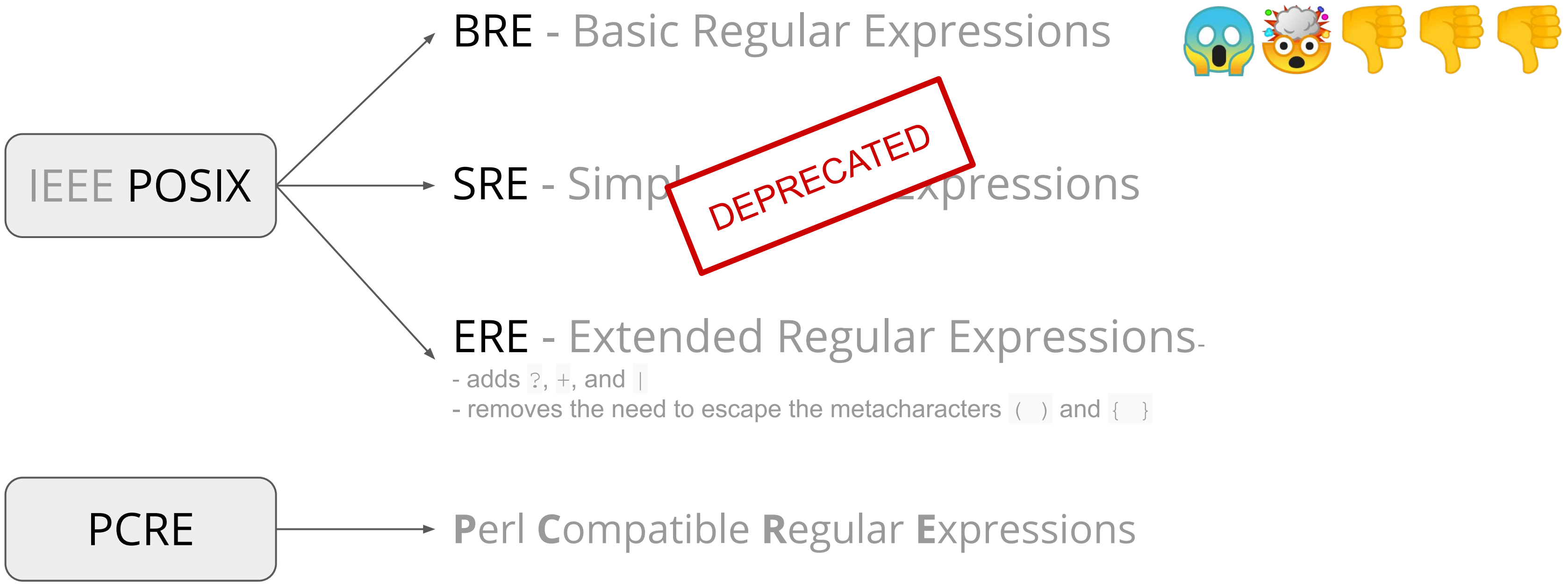
→ ERE - Extended Regular Expressions-
- adds `?`, `+`, and `|`
- removes the need to escape the metacharacters `( )` and `{ }`

PCRE

→ **P**erl **C**ompatible **R**egular **E**xpressions

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:



IEEE POSIX

BRE - Basic Regular Expressions 😱🤯👎👎👎

SRE - Simpl DEPRECATED xpressions

ERE - Extended Regular Expressions-

- adds `?`, `+`, and `|`
- removes the need to escape the metacharacters `( )` and `{ }`

PCRE

Perl **C**ompatible **R**egular **E**xpressions

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:

IEEE POSIX

BRE - Basic Regular Expressions 😱🤯👎👎👎

SRE - Simpl DEPRECATED xpressions

ERE - Extended Regular Expressions 👍👎

- adds `?`, `+`, and `|`
- removes the need to escape the metacharacters `( )` and `{ }`

PCRE

Perl **C**ompatible **R**egular **E**xpressions

https://en.wikipedia.org/wiki/Regular_expression#Standards

Standards:

IEEE POSIX

- BRE - Basic Regular Expressions 😱🤯👎👎👎
- SRE - Simpl DEPRECATED xpressions
- ERE - Extended Regular Expressions 👍👎
  - adds `?`, `+`, and `|`
  - removes the need to escape the metacharacters `( )` and `{ }`

PCRE

- **P**erl **C**ompatible **R**egular **E**xpressions 👍💪

https://en.wikipedia.org/wiki/Regular_expression#Standards

Examples…

How to move on …

The best online books:

- https://www.regular-expressions.info/tutorial.html
- https://www.rexegg.com

Books:

- Regular Expressions Cookbook by Jan Goyvaerts
- https://www.regular-expressions.info/books.html

Cheat Sheet:

- https://cheatography.com/davechild/cheat-sheets/regular-expressions

Interactive Tutorials:

- https://regexone.com

Editors:

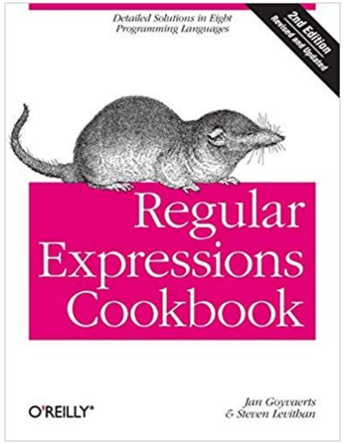
- RegexBuddy: https://www.regexbuddy.com *(The best RegExp IDE!)* 💪
- Online editor: https://regex101.com

Visualizers:

- https://ihateregex.io/playground
- https://extendsclass.com/regex-tester.html

Other:

- http://xregexp.com/xregexp/api


Detailed Solutions in Eight Programming Languages
2nd Edition Revised and Updated
Regular Expressions Cookbook
O'REILLY
Jan Goyvaerts & Steven Levithan