What is this perl statement all about? `"Hello World"` is a simple double quoted string. `World` is the regular expression and the `//` enclosing `/World/` tells perl to search a string for a match. The operator `=~` associates the string with the regexp match and produces a true value if the regexp matched, or false if the regexp did not match. In our case, `World` matches the second word in `"Hello World"`, so the expression is true. Expressions like this are useful in conditionals:

```
    if ("Hello World" =~ /World/) {
        print "It matches\n";
    }
    else {
        print "It doesn't match\n";
    }
```

There are useful variations on this theme. The sense of the match can be reversed by using `!~` operator:

```
    if ("Hello World" !~ /World/) {
        print "It doesn't match\n";
    }
    else {
        print "It matches\n";
    }
```

The literal string in the regexp can be replaced by a variable:

```
    $greeting = "World";
    if ("Hello World" =~ /$greeting/) {
        print "It matches\n";
    }
    else {
        print "It doesn't match\n";
    }
```

If you're matching against the special default variable `$_`, the `$_ =~` part can be omitted:

```
    $_ = "Hello World";
    if (/World/) {
        print "It matches\n";
    }
    else {
        print "It doesn't match\n";
    }
```

And finally, the `//` default delimiters for a match can be changed to arbitrary delimiters by putting an `'m'` out front:

```
    "Hello World" =~ m!World!;   # matches, delimited by '!'
    "Hello World" =~ m{World};   # matches, note the matching '{}'
    "/usr/bin/perl" =~ m"/perl"; # matches after '/usr/bin',
                                 # '/' becomes an ordinary char
```

`/World/`, `m!World!`, and `m{World}` all represent the same thing. When, e.g., `""` is used as a delimiter, the forward slash `'/'` becomes an ordinary character and can be used in a regexp without trouble.

Let's consider how different regexps would match `"Hello World"`:

```
    "Hello World" =~ /world/;  # doesn't match
    "Hello World" =~ /o W/;    # matches
    "Hello World" =~ /oW/;     # doesn't match
    "Hello World" =~ /World /; # doesn't match
```