Chapter 11.1

0:09

So welcome to Chapter 11.

Chapter 11's kind of a fun chapter because you don't

really need to know regular expressions.

And so, if you just want to skip ahead, or maybe do the assignment or

whatever, but this is kind of fun, regular expressions are a neat little thing.

从 ::25 开始播放视频并学习脚本0:25

They are a real old concept, they're kind of an ancient notion and

having to do with the study of languages,

not really exactly computer programming languages, but languages, and grammars,

and what is in a language and what is not in the language, and

a regular expression is a form of a language, and meaning

it's a way to say that a set of strings match or don't match a regular expression.

And from the 1960s to today, lots of operating systems have

used regular expressions as a more intelligent form of search.

So it's like, look for this expression,

but it's not just like, hello, it's like h, followed by one letter,

followed by ll, followed by a vowel, or something like that.

So that's the idea of a regular expression,

is that instead of just giving characters, you have sort of, it's almost like

a little miniature wildcard programming expression that's kind of cool.

It ends up being a programming language, but it's not like an iteration

programming language, it's a programming language that's like match, match, match,

match, question mark, question mark, question mark.

It's really kind of fun.

So if you think about you search through stuff all the time,

we just search through documents, we search through emails, we search through

things, it just is a really, really smart way to look through lots of text.

从 :1:43 开始播放视频并学习脚本1:43

And they're powerful and cryptic.

从 :1:47 开始播放视频并学习脚本1:47

It's kind of fun, once you know how to use them, because they're very,

very powerful, and you can add a little character here and do these things,

and if you had to do something the same way with

the sophistication of regular expressions, you'd have to write quite a bit of code.

And so there's a way to reduce your code.

But they also aren't something that people are natural.

It's not as easy to learn as, say, an if statement.

But it's old and it's kind of fun, and

here's an xkcd cartoon about just how completely

totally awesome regular expressions are,

and how we sort of think of them as somehow mysterious and powerful

and that those who know regular expressions are somehow special.

So, I have this regular expression quick guide.

The key to regular expressions is that instead of programming with lines,

you're programming actually with characters.

And so the caret character isn't just a caret in regular expressions,

it means beginning of line.

Dollar sign means the end of line.

Dot means any character.

Now this is a very, very cryptic and arcane language.

Bracket means the beginning of a set of characters.

Parenthesis means start extracting.

Other parenthesis means stop extracting.

So these are all like a programming language.

It's a programming language for string matching.

That's why I gave you this handout, you can go grab that,

you can print it out, any time you want to look at it, you can save yourself.

There's actually a lot more stuff, and if you look at the Python documentation,

it'll tell you that it turns out that lots of things have

regular expressions in them.

Lots of languages.

Java, JavaScript, and there's overlap and most of them are kind of the same,

but every once in a while, there will be some feature that's a little bit weird and

different.

从 :3:35 开始播放视频并学习脚本3:35

So inside Python, there's regular expressions are sort of not

built into the base language, like strings or lists or dictionaries.

And so we have to import at the beginning with the import statement that says,

pull in the regular expression library for this program.

And there's a couple things we're going to play with.

One is re.search,

which is the search capability from the regular expression library,

and that tells you yes or no, did it work, and it's like a really smart find.

And then findall is like an extraction.

It's way more powerful than slicing, but it is the idea of

finding the beginning and end of a bit of stuff, and pulling that out.

So we'll talk about both of these things.

So I'll start by showing you some code.

And show you sort of before/after.

从 :4:17 开始播放视频并学习脚本4:17

And so this is just showing you how you use the search capability from

a regular expression like a find operation.

So this, again, is our little thing we're reading through and we're saying,

if this line.find from is greater than or equal to 0.

So, this is the position, and if it's greater than or equal to 0,

then we're going to print it.

So we're searching, some of the lines we're going to skip,

some of the lines we're going to print.

Most of the lines we're going to skip and then we're going to print once in a while.

Same kind of thing.

从 :4:42 开始播放视频并学习脚本4:42

Everything here is the same, except because we're using regular expressions,

we have to import the regular expression library.

We open the file,

we're going to loop through, and that if we're going to do is say re.search.

Now, this is kind of the object oriented pattern where we take a variable name

.method.

Here we have library name.function, and then we have to pass in

the line that we're searching within and then the string that we're searching for.

And then this returns a true or a false, whether or not this match happened.

Now this is a very simple regular expression, and

I've carefully not used any of the special characters for regular expressions.

So it really functions exactly the same as find, and this gets us started.

I would probably not use regular expressions for something this simple,

but as you'll see, we can do a lot more powerful things with regular expression

than we can with the string library.

从 :5:33 开始播放视频并学习脚本5:33

So, we can also use search like the startswith function, and so that time

we're using find.

And now we're going to use startswith.

And so we say oh, it's this prefix, does it match a prefix,

because that's what startswith does for strings.

And so this code is pretty straightforward.

We know how this works.

And if starts with from, we're going to print it, otherwise,

we're going to skip it.

To do that with regular expressions, it's a little different.

We look inside strings for

a different function does a different thing for us, right?

But in regular expressions, what we do is we tweak the matching string,

so the only thing we do to indicate that we want this search to match

the beginning of the line, is we add a special character,

we add this caret character, and that caret character is not really a caret.

What it says is, I want the F to be the first character of the line,

so caret F means F at the beginning of the line.

So that's a way of matching, not from anywhere in the line, and so

this returns a true, if the From is at the beginning of the line, and a false,

if there is no From at the beginning of the line.

From can be somewhere else in the line, you'll still get a false, right?

And so that's the trick to regular expressions, is we begin to write

code in between these double quotes, and this stuff gets increasingly complex, and

then we achieve that what we want to do by more and more complex things.

And so we are going to change these wildcard expressions.

从 :6:51 开始播放视频并学习脚本6:51

So, just as an example, here's another one that may want to represent a bunch

of things, and we're introducing the dot character and the dot is any character.

It is any character.

Star, in the wildcards like the dir.\*.\*,

that means any character but star means zero or more times, zero or more.

zero or more.

Zero or more times. So that basically says, if we translate this little regular

expression with English, is I'm looking for lines that have an X at the beginning,

capital X, followed by any number of characters, followed by a colon.

So a colon's not a special character, X is not a special character, but

caret is a special character, dot and star are.

There, if you go back to that little cheat sheet, that's what those things do.

And so the kinds of things that this is going to match, the kinds of lines is

lines that start with X, followed by some number of characters, followed by a colon.

X, followed by some number of characters, followed by a colon.

Lines that don't match this, and so

this is quite a powerful little thing because it matches all of those.

It's not like saying, starts with X.

It's like X, followed by some number of characters, followed by a colon.

True or false?

And it would skip lines or give you false on lines that did not meet that, right?

And so that's the idea that caret matches the start of the line,

the dot is any character, and star means as many times as you like.

Now, you may want to narrow this down just a little bit, right?

And so, starting with the X, that might not be good enough, okay?

And so, we might not want to include, for example, this line right here,

because we really want these to be not spaces, right?

So, this X-Plane is behind schedule.

This doesn't really look like what we intended for it to look like.

This is a mail header that starts with X-Sieve, X-something, X-this,

but the X-Plane behind schedule is not actually a match.

Now it will match, because technically, it starts with an X,

it has any number of characters, and it has a colon.

But we want to be a little more precise, and so we fine tune it.

And so we say, I want matches that start with an X, followed by a dash,

followed by, and this is a special character, any non-whitespace character,

plus, it's like asterisk, but it's one or more times, so that's greater than or

equal to one time, so greater than this bit here says greater than or

equal to one non-blank character, followed by a colon.

So, this is starts with X dash, greater than or

equal to one non-blank character, followed by a colon, that's a true.

This one is X dash, followed by greater than one non-blank character,

blanks don't count.

And then that one's a true.

This one is starts with X dash, followed by one or

more, oop, no colon, because it has to match the whole thing.

So it's like taking this is like a template and applying it to this line.

This is a blank character,

which violates this rule of must be non-blank characters up to the colon,

and so it's really searching for a colon because colon is also required.

So this does not meet the requirements because of these blanks, and so

that gives us a false.

And so all I'm saying here is you can kind of add a little more detail and

hit on lines that you are sure that you want to hit on.

So up next, what we're going to do is we're going to not just look for data and

give us true/falses, whether we found it, we're actually going to start pulling

pieces out, extracting data, and so that's what's up next.