Basic elements of regular expression syntax?

In regular expressions, text that is not part of a special syntax simply matches itself. So, the following is also a valid regular expression that would match the above string: “the Justified Ancients of Mu Mu”. (Note that regular expressions are usually case sensitive unless a flag is passed to the engine to disable this).

The following are special syntactic aspects of regular expressions:

.	The dot matches any single character.
\n	Matches a newline character.
\t	Matches a tab.
\d	Matches a digit.
\D	Matches a non-digit.
\w	Matches an alphanumberic character.
\W	Matches a non-alphanumberic character.
\s	Matches a whitespace character.
\S	Matches a non-whitespace character.
\	The backslash escapes special characters. So, if you want to match an actual, literal dot (“.”) you would use \.
^	Matches the start of a string.
$	Matches the end of a string.

Quantifiers

*	Matches the preceding element 0 or more times.
+	Matches the preceding element 1 or more times.
?	Matches the preceding element 0 or 1 times.
{x}	Matches the preceding element x times.
{x,y}	Matches the preceding element between x and y times.
{x,}	Matches the preceding element at least x times.
{,y}	Matches the preceding element between 0 and y times.

Character Groups

[	Begins a character group.
]	Closes a character group.
Any character except ^-]\	Literally matches the character.
\	A backslash on its own literally matches a backslash.
\ followed by ^-]\	Literally matches the character after the backslash.
^	Specifies a *negation* of the character group. This must immediately follow the opening "[" character. It means that the regex will match if the character in the string to be searched is *not* in this character group.
- between two characters	Functions as a character *range*. For instance: a-z means every character between a and z in the alphabet (in lowercase).

Lookahead and Lookbehind

(?=zzz)	Lookahead. True if the next part of the string is "zzz".
(?<=zzz)	Lookbehind. True if the preceding part of the string is "zzz".
(?!zzz)	Negative lookahead. True if the next part of the string is *not* "zzz".
(?<!zzz)	Negative lookbehind. True if the preceding part of the string is *not* "zzz".