Regular Expressions

Gary Fenstamaker

What are Regular Expressions?

- Pattern to match text
- Consists of two parts, atoms and operators
  - Atoms specifies what we're looking for
  - Operators combine multiple atoms together

Background

- \ is an escape character
  - Meaning anything after it will be escaped
- /…/ will take everything inside as literal
- Special Characters:
  - \n means end line or new line
  - \s means space “ ”

Atoms

- Atoms can be any of the following
  - Single Character
  - Dot
  - Class
  - Anchor
  - Basic Reference

Single Character

- Simplest atom
- Matches a single character to itself

```
/a/
```

```
Mary had a little lamb. And everywhere
that Mary went, the lamb was sure to
go.
```

Dot

- Dot matches every character except the newline character (\n)

```
/./
```

```
Mary had a little lamb.\n
And everywhere that Mary went, the lamb
was sure to go.\n
```

Class

- A set of characters inside brackets
- Matches any of the characters inside

```
/[AML]/
```

```
Mary had a little lamb. And everywhere
that Mary went, the lamb was sure to
go.
```

- Also, regular expressions are case-sensitive

Class

- Classes can also do ranges of characters

```
/[A-z]/
/[A-Z]/
/[a-z]/
/[0-9]/
```

Class

- You can also add ^ to do exclusion
- Will match everything but what is inside

```
/[^AML]/
```

```
Mary had a little lamb. And everywhere
that Mary went, the lamb was sure to
go.
```

Anchors

- Character that line up the pattern to a particular point in the string
- ^ lines up the pattern to the beginning of line
- $ lines up the pattern to the end of line
- \< lines up the pattern to the beginning of a word
- \> lines up the pattern to the end of a word

Anchors

```
/^[AML]/
   Mary had a little lamb.
   And everywhere that Mary went, the lamb
   was sure to go.
/\<t/
   that Mary went.
```

Basic Reference

- Recalls the selected text in one of nine buffers
- More detail in save operator
- \1 recalls whatever text is in the first buffer

Operators

- Operators combine atoms
- Can be any of the following:
  - Sequence
  - Alternation
  - Repetition
  - Group operator
  - Save

Sequence

- Just a series of atoms

```
/Mary/
    Mary had a little lamb. And everywhere
    that Mary went, the lamb was sure to
    go.
```

Alternation

- Defines one or more alternatives
- Essentially OR

  `/a|b/`

  ```
  Mary had a little lamb. And everywhere
  that Mary went, the lamb was sure to
  go.
  ```

Repetition

- Matches repetitions of characters
- `A\{a, b\}`
  - a is the minimal number of repetitions
  - b is the maximum number of repetitions
- Both arguments are optional but one is always required
  - So you can have a minimum and no maximums
  - Or a maximum but no minimum

Repetition

```
/A\{3,5\}/
    AA
    AAA
    AAAA
    AAAAA
    AAAAAA
/A\{4,\}/
    AAA
    AAAA
    AAAAA
    AAAAAA
```

Repetition Short Form

- * matches an atom zero or more times
- + matches an atom one or more times
- ? matches an atom zero or one time only

Repetition Short Form

/BA+D/

BAD

BD

BAAD

BAAAAAAD

/BA?D/

BAD

BD

BAAD

Greedy Operator

- Matches the longest possible string of characters to the pattern

  `/M.*t/`

  ```
  Mary had a little lamb. And everywhere
  that Mary went, the lamb was sure to
  go.
  ```

Save

- Save copies of matched text to a buffer for later use
- `\(.*\)`
- Useful to find double words

  `/\([A-z]+\)\s\1/`

  ```
  Mary Mary had a little lamb. And
  everywhere that Mary went, the lamb was
  sure to go.
  ```