Regular Expressions

• A regular expression is an object that describes a pattern of characters.
• The JavaScript RegExp class represents a regular expression.
• Both String and RegExp define methods that use regular expressions to perform pattern matching and search-and-replace function on text.

Defining Regular Expression Literals

• Regular-expression pattern specifications consist of a series of characters.
• The characters can be alphabets, digits and special characters.
• Regular expression literals are specified as characters within a pair of slash (/)

```javascript
var pattern = /JavaScript/;
// matches any string that contains substring “JavaScript”.
```

Regular Expression Literal Characters

• `/cs-3710/`: alphanumeric characters
• `/cs-3710\t/`: \t is tab
• `/cs-3710\n/`: \n is newline
• `/\(cs-3710\)/`: characters that have special meaning are prefixed with backslash (\) for its literal meaning.
• Special characters:
  ^ $ . * + ? = ! : | / ( ) [ ] { }

Character Classes

• Individual literal characters can be grouped into character classes using square brackets [].
• A character class matches any one character that is contained within it.
• /[abc]/ matches any of the letter a, b or c.
• /[^abc]/ matches any of letter other than a, b or c.
• /[a-z]/ matches any of the lower-case alphabet a to z. //”123”?
• /[^a-zA-Z0-9]/ matches any alphanumeric characters
• //”123”? “Java”? “cs-3710”?

Other Character Classes

• \w: [a-zA-Z0-9_] //”--”?
• \W:[^a-zA-Z0-9_] //”/”
• \s: white space //”java”
• \S: anything except white space
• \d:[0-9] //”abc”?
• \D:[^0-9] //”ABC”?

Repetition

• /\{0,3\}/: match slash (/) 0, 1, 2 or 3 times. //”http://”?
• /\{0,\}/: match slash(/) 0 or more times. //”http://”?
• /\{1\}/: match slash(/) exactly once. //”http://”? 
• /\d?: an optional digit //”cs-3710”?
• /\d+/: match any of the digits one or more times. //”cs-3710”?
• /\d*/: match any of the digits 0 ore more times //”cs-3710”?

Alternation, grouping, referencing

• Alternation: |
  – /ab|cd|ef/: match the string ab, cd or ef.
  – /\d\{3\}|[a-z]\{4\}/: match 3 digits or 4 lowercase letters. //”www.123.com”?
• Grouping: ()
  – /ab|cd)+ef/: math either ef or one or more ab or cd. //”cs-3710”?
  – /\{(’ “’ “’)*\}: match zero or more characters within single or double quotes. \1 refers to the matched quote in () //”course ”cs-3710”
Matching Position and Flags

- ^: beginning of the string
- $: end of the string
- `/^JavaScript$/i`: match the word “JavaScript” on a line by itself. `//JavaScript book?`
- `/^JavaScript$/g`: case insensitive
- `/^JavaScript$/g`: global match multiple times, don’t stop when the first match is found. This flag is mostly used for global replacement.

String Methods for Pattern Matching

- `String.search(RegExp)`: returns the character position of the first matching substring or -1 if no matching.
  
  “JavaScript”.search(/script/i); //4
  “JavaScript”.search(/script/); //?
- `String.replace(RegExp, String)`: replaces the first matching string with the string argument.
  
  text.replace(/java/gi, “Java”); //text is a String variable

String Methods - Match

- `String.match(RegExp)`: returns an array that contains the result of the match; null if no match.
- If there is a match, array[0] contains the entire match, array[i] contains the substring that matches the ith parenthesized expression.

  ```javascript
  var url = /(\w+)://(\[(\w.)]+)/(\S*)/;
  var text = "Visit my blog at http://www.example.com/~david";
  var result = text.match(url);
  if(result != null){
      document.writeln(result[0]);
      document.writeln(result[1]);
      document.writeln(result[2]);
      document.writeln(result[3]);
  }
  ```

String Method - Split

- `String.split(String)`: returns an array that contains the substring of the String using the string argument as a separator.
  
  "1,2,3,4,5,6".split(\',\');
  //return [1,2,3,4,5,6]
- `String.split(RegExp)`: using RegExp as the separator.
  
  "1,2;3 4,5 6".split(/\[\s,;\t]/);
  //return [1,2,3,4,5,6]
RegExp Methods

- `RegExp.exec(String)` gives the same result as `String.match(RegExp)` when the search is not global (RegExp has no g flag);
- `RegExp.test(String)` returns true if the string contains a match for the RegExp.

```javascript
var pattern=/java/i;
pattern.test("JavaScript"); //true
```