Strings and Text

Elements of Graphics
CS324e
Spring 2019
Characters

- Primitive data types in Processing
- Assigned to variables with single quotes
  ```
  char letter = 'A';
  ```
- Characters have corresponding ASCII value
  ```
  char letter = 'A'; //letter has value A
  int number = letter; //number has value 65
  ```
ASCII Table

- American Standard Code for Information Interchange
- Provides standard for mapping characters to computer-understood numbers
- [http://www.asciitable.com/](http://www.asciitable.com/)
- ASCII encodes 128 characters (8-bits)
- Unicode allows for 16 and 32 bit encodings
Strings

- Data type in Processing contains words and sentences
- Assigned to variables with double quotes
  ```java
  String s = "A string";
  ```
- Strings can be concatenated with the + operator
  ```java
  String s1 = "A ";
  String s2 = "string";
  String s3 = s1 + s2;  //s3 = "A string"
  ```
String Objects

- Objects have variables (fields) and functions (methods)
- Fields and methods accessed using the dot operator
- Fields do not have parentheses, but methods do
- `length()` is a String function

```java
String str = "Hello World";
int length = str.length();  //length is 11
```
String Methods

- `startsWith()` and `endsWith()` check whether the String begins or ends with the provided parameter:

  ```
  String s = "Hello";
  bool isTrue = s.startsWith("He");
  isTrue = s.endsWith("Lo");
  ```
• `charAt()` returns the character at a given index
  ```java
  String s = "Hello World";
  char x = s.charAt(4);  // x now has value 'o'
  ```

• `substring()` returns a String within the provided indices
  ```java
  String s = "Hello World";
  String s1 = s.substring(0, 5);
  String s2 = s.substring(6);
  ```
toLowercase() and toUpperCase() return a copy of the String in either all lower or upper case

String s = "Hello World"
String s_lower = s.toLowerCase();
//s_lower is "hello world"
s = s.toLowerCase();

equals() allows the String to be compared to the provided parameter

bool isTrue = s.equals(s_lower);
isTrue = (s == s_lower);
Splitting and Joining Strings

- `split()` separates String into an array of Strings separated by the delimiter
  
  ```java
  String s = "Hello World";
  String[] subwords = split(s, ' ');
  //subwords = [Hello, World]
  ```

- `splitTokens()` allows for splitting along multiple delimiters

- `join()` can join multiple Strings

- `+` operator can also join Strings
Reading Files

- Processing can read from .txt, .csv and .xml file types
  - For now, we’ll focus on reading from .txt

1. Move the .txt file into sketch’s data directory
2. Call `loadStrings()` to break each line into its own String

   ```java
   String[] lines = loadStrings("textfile.txt");
   ```
Writing Files

- Writing to a file can be done all at once or continuously appended
- `saveStrings()` writes an array of Strings to a file (one line per String)
- The `PrintWriter` class appends print statements to a file
PrintWriter Example

PrintWriter output;
void setup() {
    output = createWriter("outputPositions.txt");
}
void draw() {
    output.println("(" + mouseX + ", " + mouseY + ")");
}
void keyPressed() {
    output.flush();
    output.close();
}
Hands-on: Using Strings

❖ Today’s activities:

1. Create a small text file “mytext.txt” to read into Processing
2. Count the length of each line and print it to the console
3. Split each line into individual words and count the number of words. Print this number to the console
4. Use the PrintWriter class to print out each word on its own line to the file “words.txt”