# Strings

- String Literals
- Adding Strings
- String Equality
- String Methods
- StringBuilder
- String Algorithms

# See Also

http://www.docjar.com/html/api/java/lang/String.java.html

http://www.javacodegeeks.com/2010/09/string-performance-exact-string.html

https://www.youtube.com/watch?v=4l50UaPca7Y

# Character Array Manager

```
public final class String {

    private final char value[];
    private final int offset;
    private final int count;

    public String() {
        this.offset = 0;
        this.count = 0;
        this.value = new char[0];
    }

    public String(char value[]) {
        int size = value.length;
        this.offset = 0;
        this.count = size;
        this.value = Arrays.copyOf(value, size);
    }

    public int length() {
        return count;
    }

}
```

- String is not a primitive type
- You could write one based on char []
- This code is from the real String class (see the first link in See Also)

# String Literals

```
public class Tinker {

    public static void main(String [] args) {

        String s = "Hello world";

        String s = constObject22;

        System.out.println(s);

        int i = "Hello world".length();
        int i = constObject22.length();


    }

}
```

- String literals become special global-objects
- When you use a literal you are really using one of these special objects that is created when your program runs

# Changing a String

```java
public static void main(String [] args) {

    String s = "Hello there world";

    s.toUpperCase();

    System.out.println(s);

}
```

Problems @ Javadoc Declaration Console

```
<terminated> Tinker (1) [Java Application] C:\Program Files\Java\j
Hello there world
```

```java
public static void main(String [] args) {

    String s = "Hello there world";

    String t = s.toUpperCase();

    System.out.println(t);

    System.out.println(s);

    s = s.toUpperCase();

}
```

Problems @ Javadoc Declaration Console

```
<terminated> Tinker (1) [Java Application] C:\Program Files\Jav
HELLO THERE WORLD
Hello there world
```

- Strings are immutable – the characters can never be changed
- The methods return new string objects

# Adding Strings

```java
public static void main(String [] args) {

    String s = "Hello";
    String t = "World";

    String u = s + t;

    System.out.println(u);

}
```

```java
public static void main(String [] args) {

    String s = "Hello";
    String t = "World";

    String u = s.concat(t);

    System.out.println(u);

}
```

# String Equality

```
String s = "Hello";
String t = "World";
String u = s + t;

if(u=="HelloWorld") {
    System.out.println("Same");
}

String x = "Hello";
if(x==s) {
    System.out.println("SAME");
}
```

- The == compares pointer addresses
- Sometimes it works as you expect (with literals)
- Most of the time it doesn't

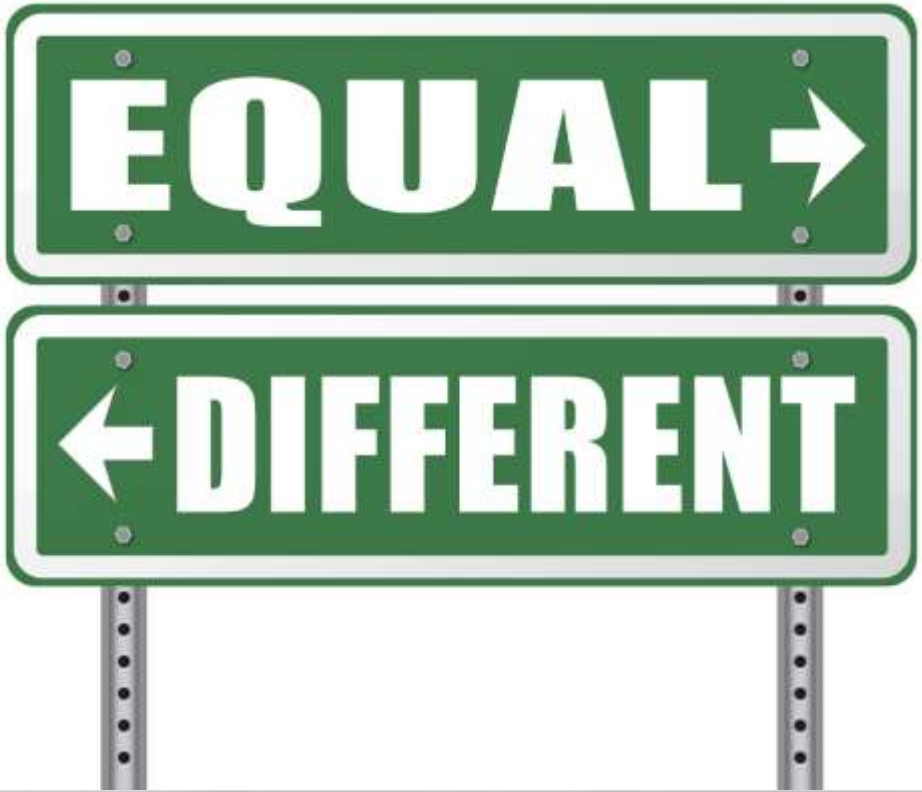
# String Equality

```
String s = "Hello";
String t = "World";
String u = s + t;

if(u.equals("HelloWorld")) {
    System.out.println("Same");
}

String x = "Hello";
if(x.equals(s)) {
    System.out.println("SAME");
}
```

- The equals method compares the strings character by character



# String Operations

```
String s = "Hello There World";
System.out.println( s.length() );
System.out.println( s.toUpperCase() );
System.out.println( s.toLowerCase() );
System.out.println( s.concat("Earth") );

s = s + "Other";    // s = s.concat("Other")
s = "Score: "+25;   // Integer.toString(25)

s = "Hello   There World";
String [] a = s.split(" ");
System.out.println(Arrays.toString(a));  →  [Hello, , , There, World]

s = "   Hello   \n\n   \n";
s = s.trim(); // "Hello"
```

# Index and Substrings

```
//          01234567890123456
String s = "Hello There World";

String t = s.substring(2,8); // 2 through 7

System.out.println(":"+t+":");
```

```
:llo Th:
```

```
t = s.substring(4); // 4 to the end

System.out.println(":"+t+":");
```

```
:o There World:
```

# Index and Substrings

```
//          01234567890123456
String s = "Hello There World";

int i = s.indexOf("er");
System.out.println(i);

i = s.indexOf("e",4);
System.out.println(i);

i = s.lastIndexOf("e");
System.out.println(i);

i = s.lastIndexOf("e",6);
System.out.println(i);
```

Problems @ Javadoc Declaration Console

```
<terminated> Tinker (1) [Java Application] C:\Program Files\Jav
8
8
10
1
```

# Many More

- equals(Object anObject) : boolean - String - 0.68%
- length() : int - String - 0.41%
- equalsIgnoreCase(String anotherString) : boolean - String - 0.2%
- charAt(int index) : char - String - 0.11%
- indexOf(int ch) : int - String - 0.1%
- hashCode() : int - String - 0.06%
- indexOf(String str) : int - String - used
- indexOf(String str, int fromIndex) : int - String - used
- lastIndexOf(String str) : int - String - used
- lastIndexOf(String str, int fromIndex) : int - String - used
- chars() : IntStream - CharSequence
- codePointAt(int index) : int - String
- codePointBefore(int index) : int - String
- codePointCount(int beginIndex, int endIndex) : int - String
- codePoints() : IntStream - CharSequence
- compareTo(String anotherString) : int - String
- compareToIgnoreCase(String str) : int - String
- concat(String str) : String - String
- contains(CharSequence s) : boolean - String
- contentEquals(CharSequence cs) : boolean - String
- contentEquals(StringBuffer sb) : boolean - String
- endsWith(String suffix) : boolean - String
- getBytes() : byte[] - String
- getBytes(Charset charset) : byte[] - String
- getBytes(String charsetName) : byte[] - String
- getBytes(int srcBegin, int srcEnd, byte[] dst, int dstBegin) : void - String
- getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) : void - String
- getClass() : Class<?> - Object
- indexOf(int ch, int fromIndex) : int - String
- intern() : String - String
- isEmpty() : boolean - String
- lastIndexOf(int ch) : int - String
- lastIndexOf(int ch, int fromIndex) : int - String
- matches(String regex) : boolean - String
- notify() : void - Object

Press 'Ctrl+Space' to show Template Proposals

- notify() : void - Object
- notifyAll() : void - Object
- offsetBy CodePoints(int index, int codePointOffset) : int - String
- regionMatches(int toffset, String other, int ooffset, int len) : boolean - String
- regionMatches(boolean ignoreCase, int toffset, String other, int ooffset, int len) : boolean - Stir
- replace(char oldChar, char newChar) : String - String
- replace(CharSequence target, CharSequence replacement) : String - String
- replaceAll(String regex, String replacement) : String - String
- replaceFirst(String regex, String replacement) : String - String
- split(String regex) : String[] - String
- split(String regex, int limit) : String[] - String
- startsWith(String prefix) : boolean - String
- startsWith(String prefix, int toffset) : boolean - String
- subSequence(int beginIndex, int endIndex) : CharSequence - String
- substring(int beginIndex) : String - String
- substring(int beginIndex, int endIndex) : String - String
- toCharArray() : char[] - String
- toLowerCase() : String - String
- toLowerCase(Locale locale) : String - String
- toString() : String - String
- toUpperCase() : String - String
- toUpperCase(Locale locale) : String - String
- trim() : String - String
- wait() : void - Object
- wait(long timeout) : void - Object
- wait(long timeout, int nanos) : void - Object
- CASE_INSENSITIVE_ORDER : Comparator<java.lang.String> - String
- copyValueOf(char[] data) : String - String
- copyValueOf(char[] data, int offset, int count) : String - String
- format(String format, Object... args) : String - String
- format(Locale l, String format, Object... args) : String - String
- join(CharSequence delimiter, CharSequence... elements) : String - String
- join(CharSequence delimiter, Iterable<? extends CharSequence> elements) : String - String
- valueOf(boolean b) : String - String
- valueOf(char c) : String - String
- valueOf(char[] data) : String - String
- valueOf(double d) : String - String
- valueOf(float f) : String - String
- valueOf(int i) : String - String
- valueOf(long l) : String - String
- valueOf(Object obj) : String - String
- valueOf(char[] data, int offset, int count) : String - String

Press 'Ctrl+Space' to show Template Proposals

# String Builder

```
StringBuilder b = new StringBuilder("Hello World");

b.append("Earth");
System.out.println(b); // Hello WorldEarth

b.delete(2, 4);
System.out.println(b); // Heo WorldEarth

b.insert(6, 1234);
System.out.println(b); // Heo Wo1234rldEarth

b.reverse();
System.out.println(b); // htraEdlr4321oW oeH
```

# String Algorithms

```
//          0123456789012
String s = "age    =    27";

int i = s.indexOf("="); // 6

String key = s.substring(0,i); // "age   "
key = key.trim(); // "age"

String value = s.substring(i+1).trim();

System.out.println(":"+key+":"+value+":");
```

| key | value |
|---|---|
| firstName | Bugs |
| lastName | Bunny |
| location | Earth |

```
Age      = 27
Name     = Chris
Height   = 8

socket: 8080
host: localhost
```

Problems @ Javadoc Declaration Console

```
<terminated> Tinker (1) [Java Application] C:\Program Files\Java
:age:27:
```

# String Algorithms

```java
public static void main(String [] args) {

    String s = "This AA is AA a test";

    String t = replaceAll(s,"AA","ummm");

    System.out.println(t);

}
```

```java
public static String replaceAll(String s, String key, String value) {

    while(true) {
        int i = s.indexOf(key);
        if(i<0) {
            return s;
        }
        String bef = s.substring(0,i);
        String aft = s.substring(i+key.length());
        s = bef + value + aft;
    }

}
```

# Tinkering

- Code up the "replaceAll" algorithm and test drive it. Try passing in a string with a match at the very front … at the very end … test all cases.
- What if you pass in a replacement that includes the match? How can you make method perform just one pass?