Strings

Introduction

- Each and every text is represented using a `String` object.
- The `String` object holds all the characters of the text it represents.
- Once the `String` class was instantiated it isn't possible to change the text it represents.

Creating Strings

- There are two common ways for creating a `String` object. We can either use the constructors that `String` already includes, or simply writing the text.

```
String str = "abc";
String str = new String("abc");
```

- The `String` class has various constructors:

```
public String(String str)
public String(byte vec[])
public String(char vec[])
```

Comparing strings

- When using the == operator we actually compare the references.

```
String str1 = new String(“Haim”);
String str2 = new String(“Haim”);
if(str1==str2)
     ...
```

Comparing strings

- In order to compare the strings themselves we should use the equals method.

```
String str1 = new String(“Haim”);
String str2 = new String(“Haim”);
if(str1.equals(str2))
     ...
```

The `toString()` method

- Each and every object in Java is also an `Object`.

  The `toString()` method was defined in the `Object` class.
- We can override the `toString` method in every class we define.

The `StringBuffer` Class

- Unlike `String`, when having a `StringBuffer` object, we can change the text it represents.

```
StringBuffer sb = new StringBuffer(“haim“);
sb.append(“ “);
sb.append(“michael”);
```

The `StringBuffer` Class

- When adding one string to another, we might improve the performance by using the `StringBuffer` class.

```
StringBuffer sb = new StringBuffer();
String vec[];
...
for(int i=0; i<vec.length; i++)
{
        sb.append(vec[i]);
}
```

Using a `StringBuffer` Object

The `StringBuffer` Class

```
String str = new String("");
String vec[];
...
for(int i=0; i<vec.length; i++)
{
     str = str + vec[i];
}
```

Without Using The `StringBuffer` Class

The `StringBuilder` Class

- `StringBuilder` is very similar to `StringBuffer`.
- Unlike `StringBuffer`, `StringBuilder` is not thread safe. On the other hand, `StringBuilder` is much fatster.

The `main` Method Arguments

- When executing a java application it is possible to pass over arguments to the main method. Each argument will be represented by a `String` object.
- The `args[]` parameter receives a reference to array that holds all references for these `String` objects.