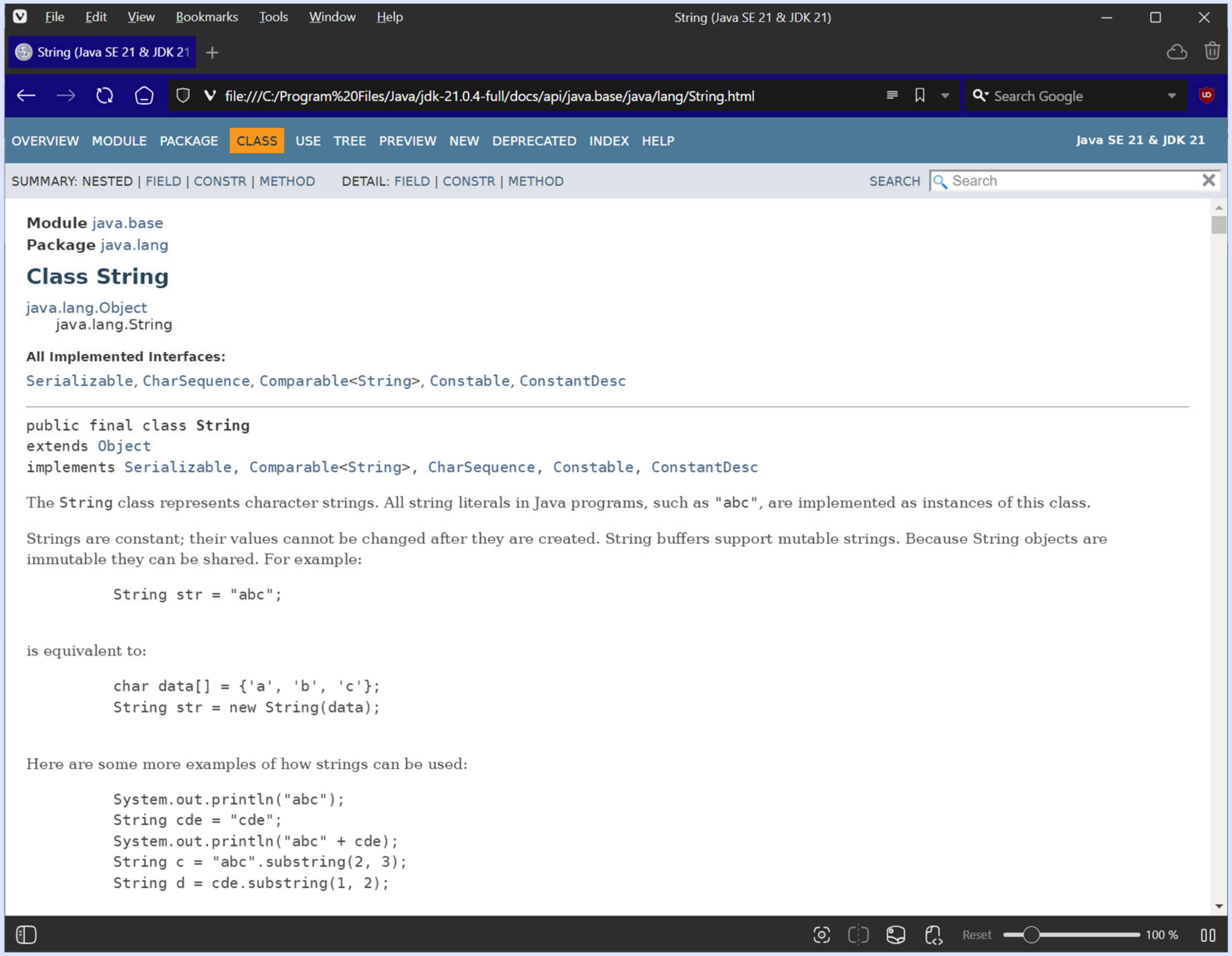
Javadocs

Javadocs

- Java comes with full documentation of all API classes in html format
  - If you followed the installation guide you can find the documentation in a folder named docs inside the folder where you installed Java
    - Either browse the documentation manually (open index.html inside the api folder) or access the documentation from within IntelliJ
- We can document our own classes in the same format as the API classes

Javadoc for class String

Scroll down for info about fields, constructors and methods in the String class


File Edit View Bookmarks Tools Window Help
String (Java SE 21 & JDK 21)
file:///C:/Program%20Files/Java/jdk-21.0.4-full/docs/api/java.base/java/lang/String.html
Search Google
OVERVIEW MODULE PACKAGE CLASS USE TREE PREVIEW NEW DEPRECATED INDEX HELP
Java SE 21 & JDK 21
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset
100 %

Module java.base
Package java.lang

Class String

java.lang.Object
    java.lang.String

All Implemented Interfaces:
`Serializable, CharSequence, Comparable<String>, Constable, ConstantDesc`

```
public final class String
extends Object
implements Serializable, Comparable<String>, CharSequence, Constable, ConstantDesc
```

The `String` class represents character strings. All string literals in Java programs, such as `"abc"`, are implemented as instances of this class.

Strings are constant; their values cannot be changed after they are created. String buffers support mutable strings. Because String objects are immutable they can be shared. For example:

```
     String str = "abc";
```

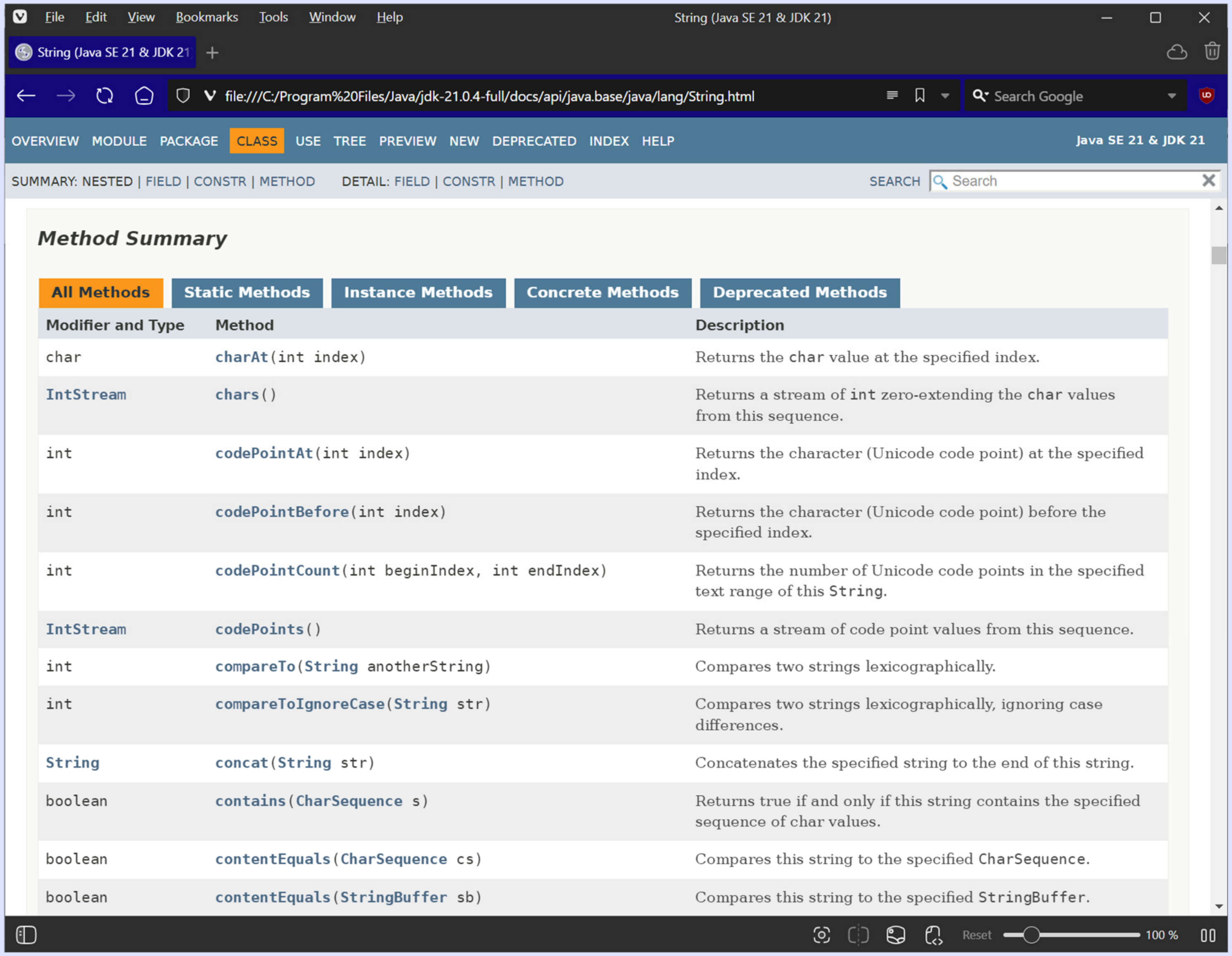
is equivalent to:

```
     char data[] = {'a', 'b', 'c'};
     String str = new String(data);
```

Here are some more examples of how strings can be used:

```
     System.out.println("abc");
     String cde = "cde";
     System.out.println("abc" + cde);
     String c = "abc".substring(2, 3);
     String d = cde.substring(1, 2);
```

Javadoc for class String


File Edit View Bookmarks Tools Window Help
String (Java SE 21 & JDK 21)
file:///C:/Program%20Files/Java/jdk-21.0.4-full/docs/api/java.base/java/lang/String.html
Search Google
OVERVIEW MODULE PACKAGE CLASS USE TREE PREVIEW NEW DEPRECATED INDEX HELP
Java SE 21 & JDK 21
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset
100 %

Method Summary

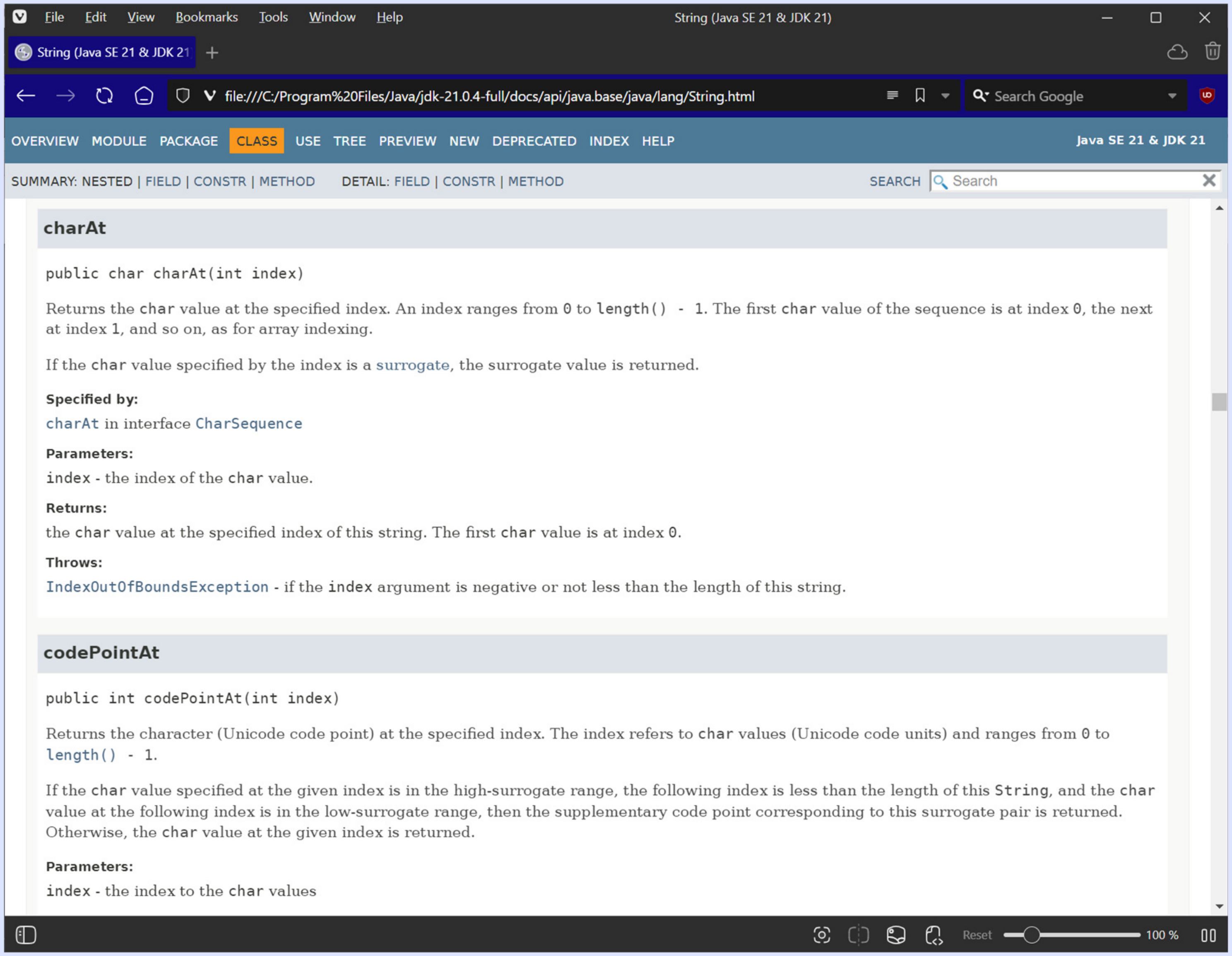
All Methods | Static Methods | Instance Methods | Concrete Methods | Deprecated Methods

Modifier and Type	Method	Description
`char`	`charAt(int index)`	Returns the `char` value at the specified index.
`IntStream`	`chars()`	Returns a stream of `int` zero-extending the `char` values from this sequence.
`int`	`codePointAt(int index)`	Returns the character (Unicode code point) at the specified index.
`int`	`codePointBefore(int index)`	Returns the character (Unicode code point) before the specified index.
`int`	`codePointCount(int beginIndex, int endIndex)`	Returns the number of Unicode code points in the specified text range of this `String`.
`IntStream`	`codePoints()`	Returns a stream of code point values from this sequence.
`int`	`compareTo(String anotherString)`	Compares two strings lexicographically.
`int`	`compareToIgnoreCase(String str)`	Compares two strings lexicographically, ignoring case differences.
`String`	`concat(String str)`	Concatenates the specified string to the end of this string.
`boolean`	`contains(CharSequence s)`	Returns true if and only if this string contains the specified sequence of char values.
`boolean`	`contentEquals(CharSequence cs)`	Compares this string to the specified `CharSequence`.
`boolean`	`contentEquals(StringBuffer sb)`	Compares this string to the specified `StringBuffer`.

Just click on a method name for more details

Javadoc for class String

A detailed description of the `charAt` method


File Edit View Bookmarks Tools Window Help
String (Java SE 21 & JDK 21)
file:///C:/Program%20Files/Java/jdk-21.0.4-full/docs/api/java.base/java/lang/String.html
Search Google
OVERVIEW MODULE PACKAGE CLASS USE TREE PREVIEW NEW DEPRECATED INDEX HELP
Java SE 21 & JDK 21
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset
100 %

charAt

```
public char charAt(int index)
```

Returns the `char` value at the specified index. An index ranges from `0` to `length() - 1`. The first `char` value of the sequence is at index `0`, the next at index `1`, and so on, as for array indexing.

If the `char` value specified by the index is a surrogate, the surrogate value is returned.

Specified by:
`charAt` in interface `CharSequence`

Parameters:
`index` - the index of the `char` value.

Returns:
the `char` value at the specified index of this string. The first `char` value is at index `0`.

Throws:
`IndexOutOfBoundsException` - if the `index` argument is negative or not less than the length of this string.

codePointAt

```
public int codePointAt(int index)
```

Returns the character (Unicode code point) at the specified index. The index refers to `char` values (Unicode code units) and ranges from `0` to `length() - 1`.

If the `char` value specified at the given index is in the high-surrogate range, the following index is less than the length of this `String`, and the `char` value at the following index is in the low-surrogate range, then the supplementary code point corresponding to this surrogate pair is returned. Otherwise, the `char` value at the given index is returned.

Parameters:
`index` - the index to the `char` values

Documenting code with Javadocs

- When we install the Java SDK, we also get a Javadoc tool that can generate Javadocs for our own classes
  - All we need to do is add some special comments in our source code
- Professional looking documentation
- Other people can understand what our classes do without looking through all the source code
- A requirement as documentation of your SEP1 Java classes
  - At least for your model classes

Documenting code with Javadocs

- For each of our classes we should write:
  - A comment describing the overall purpose and characteristics of the class
  - The name(s) of the author(s)
  - A version number
  - Documentation for every constructor and public method
    - Usually, Java docs are only generated for public methods, however it's still a good idea to write comments for the private methods in case someone looks at the source code

Documenting code with Javadocs

- The documentation for the constructors and methods should include:
  - A description of the purpose and function of the method
  - A description of each parameter (if there are any)
  - A description of the return value (if there is one)
  - A list of the checked exceptions that the method can throw (if any)

Writing Javadocs

- Javadoc comments are written directly in the source code using a special block comment syntax
  - Javadoc blocks start with `/**` and ends with `*/`
  - Inside the blocks we use special tags, including:

`@author`	The name of the programmer
`@version`	The version of the class
`@param`	Description of parameters for a method
`@return`	Description of what is returned from a method
`@throws`	Description of the exceptions thrown by the method

Javadocs example (1/3)

```
import java.util.ArrayList;

/**
* A class containing a list of Student objects.
* @author Allan Henriksen
* @version 1.0
*/
public class StudentList
{
  private ArrayList<Student> students;

  /**
   * No-argument constructor initializing the StudentList.
   */
  public StudentList()
  {
       students = new ArrayList<Student>();
  }
```

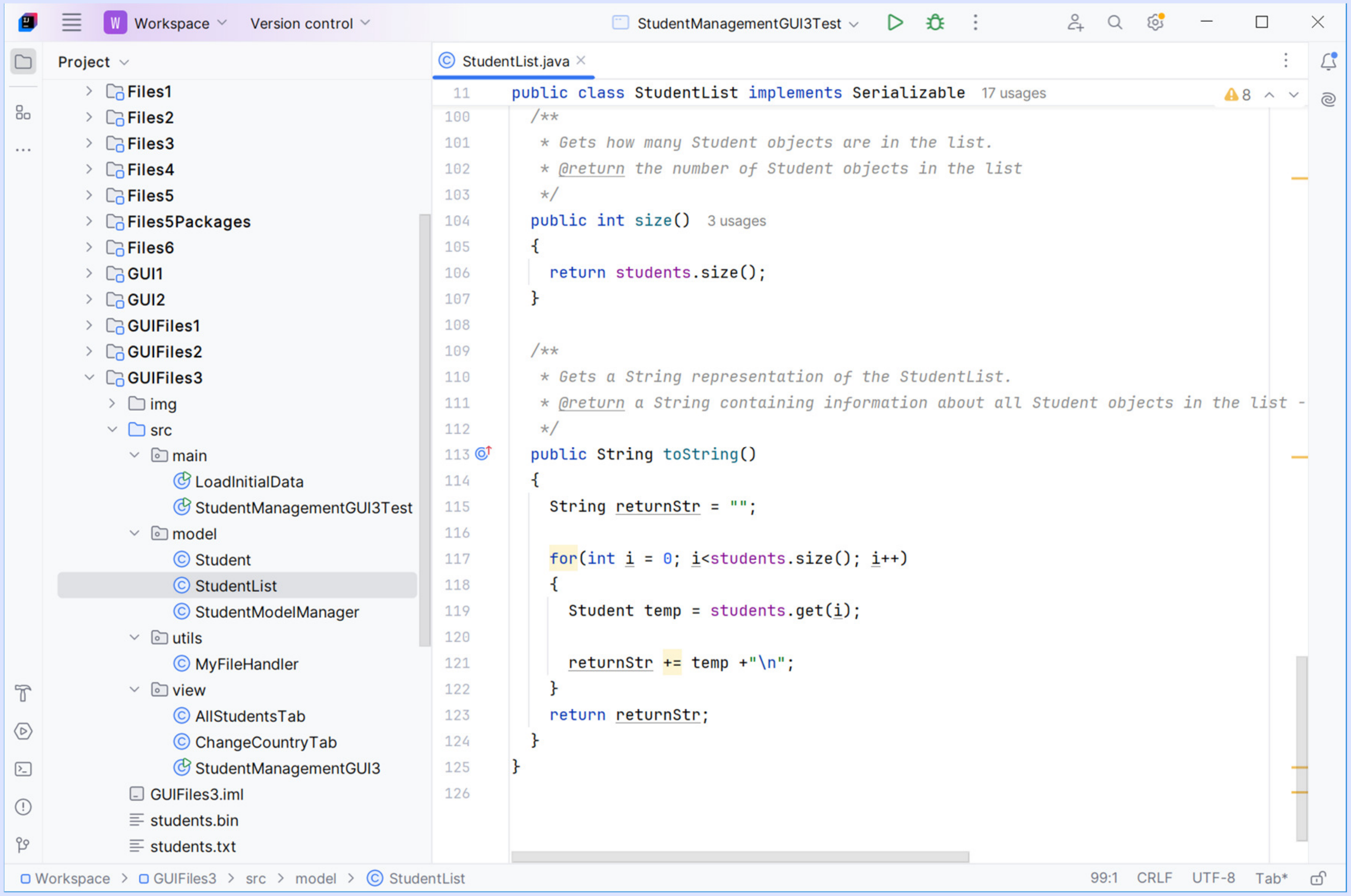
Javadocs example (2/3)

```
/**
 * Adds a Student to the list.
 * @param student the student to add to the list
 */
 public void add(Student student)
 {
    students.add(student);
 }

/**
 * Replaces the Student object at index with student.
 * @param student the student to replace with
 * @param index the position in the list that will be replaced
 */
 public void set(Student student, int index)
 {
      students.set(index, student);
 }
```

Javadocs example (3/3)

```
/**
 * Gets a Student object from position index from the list.
 * @param index the position in the list of the Student object
 * @return the Student at index if one exists, else null
 */
 public Student get(int index)
 {
    if(index<students.size())
    {
        return students.get(index);
     }
     else
     {
        return null;
     }
  }

  //rest of class
```

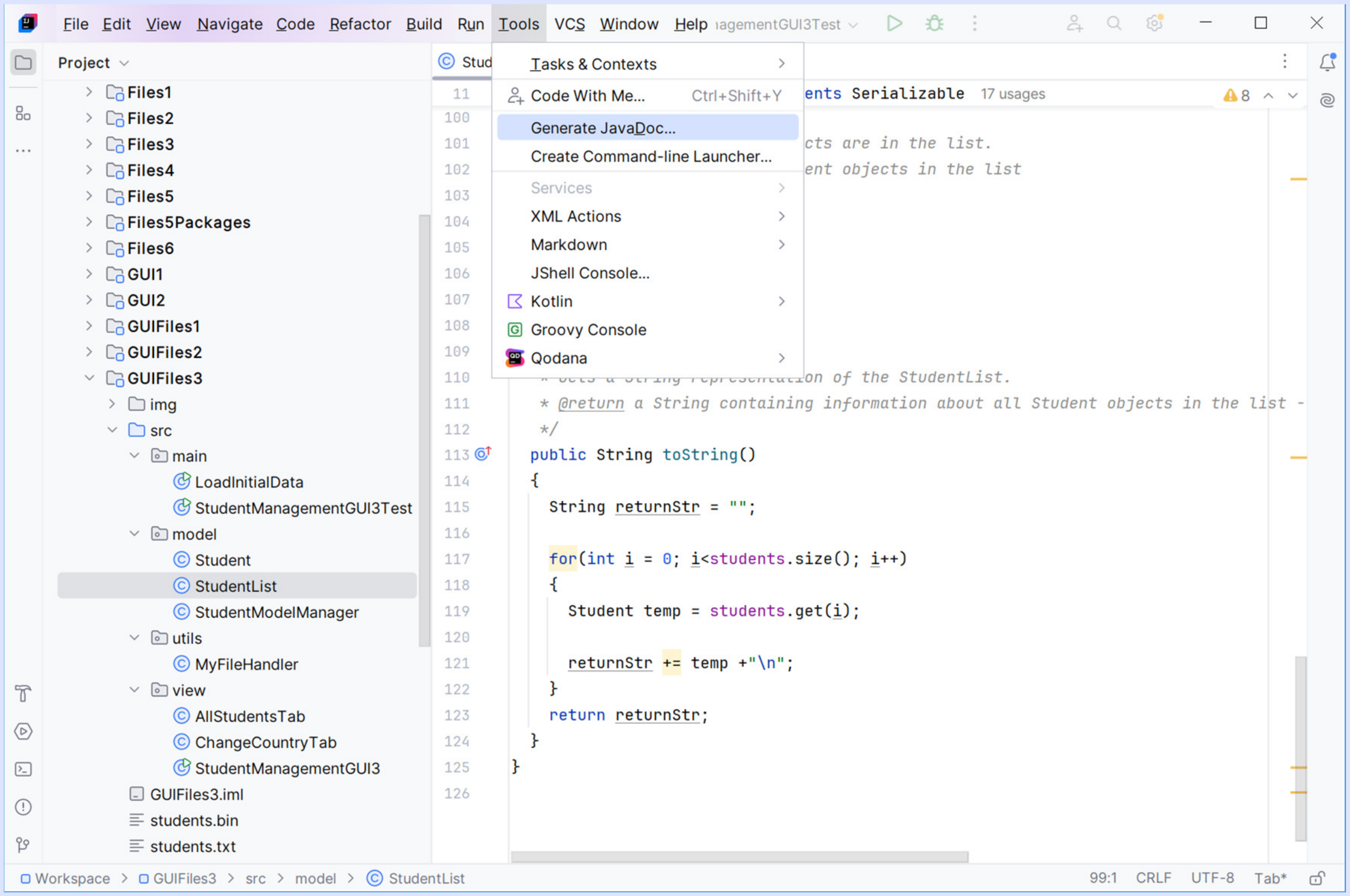
Java code with Javadocs



```java
public class StudentList implements Serializable
  /**
   * Gets how many Student objects are in the list.
   * @return the number of Student objects in the list
   */
  public int size()
  {
    return students.size();
  }

  /**
   * Gets a String representation of the StudentList.
   * @return a String containing information about all Student objects in the list -
   */
  public String toString()
  {
    String returnStr = "";

    for(int i = 0; i<students.size(); i++)
    {
      Student temp = students.get(i);

      returnStr += temp +"\n";
    }
    return returnStr;
  }
}
```

Generating Javadocs

When you are done writing the Javadoc comments, choose “Generate JavaDoc...” from the “Tools” menu in IntelliJ



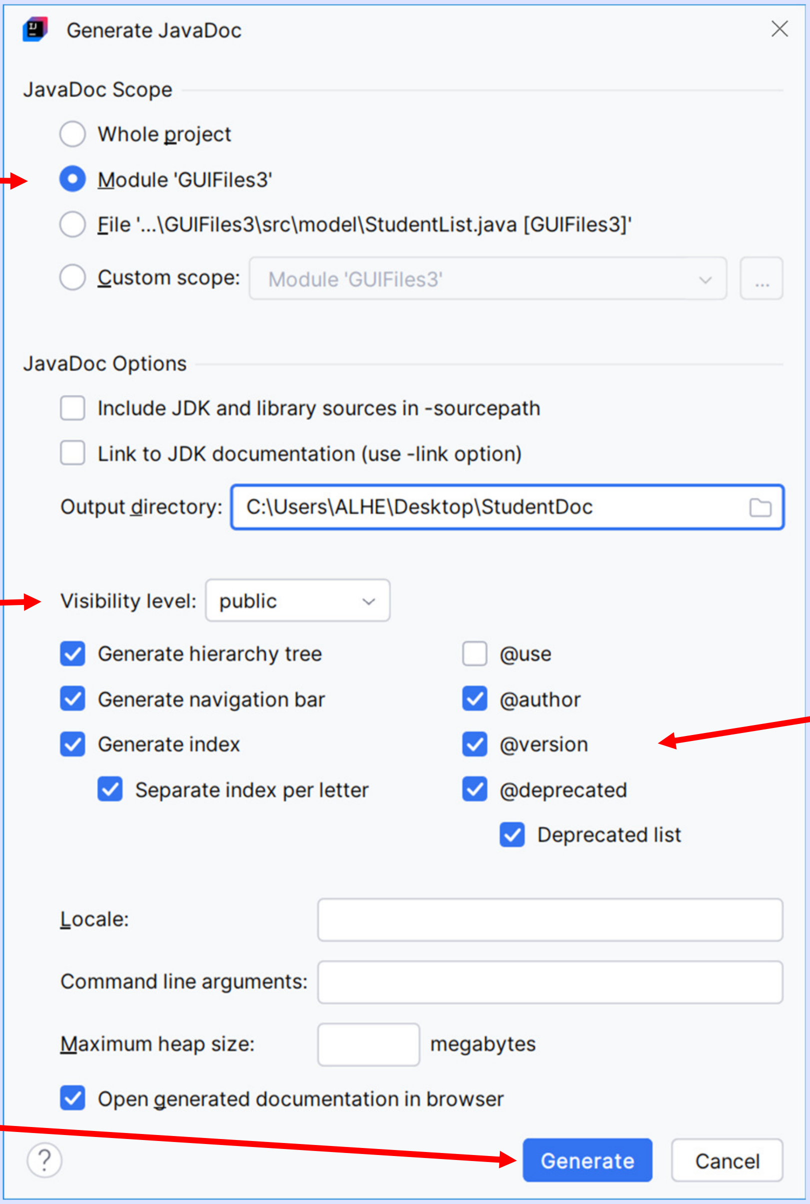
```
ents Serializable
cts are in the list.
ent objects in the list
on of the StudentList.
 * @return a String containing information about all Student objects in the list -
 */
public String toString()
{
  String returnStr = "";

  for(int i = 0; i<students.size(); i++)
  {
    Student temp = students.get(i);

    returnStr += temp +"\n";
  }
  return returnStr;
}
}
```

Generating Javadocs

Choose what to generate JavaDoc for. You'll probably want it for the current module

Choose which access modifier levels to include in the JavaDoc. Usually it's done only for public

Click "OK" to generate the JavaDocs



Generate JavaDoc

JavaDoc Scope

- ○ Whole project
- ● Module 'GUIFiles3'
- ○ File '...\GUIFiles3\src\model\StudentList.java [GUIFiles3]'
- ○ Custom scope: Module 'GUIFiles3'

JavaDoc Options

- [ ] Include JDK and library sources in -sourcepath
- [ ] Link to JDK documentation (use -link option)

Output directory: C:\Users\ALHE\Desktop\StudentDoc

Visibility level: public

- [x] Generate hierarchy tree
- [x] Generate navigation bar
- [x] Generate index
  - [x] Separate index per letter
- [ ] @use
- [x] @author
- [x] @version
- [x] @deprecated
  - [x] Deprecated list

Locale:

Command line arguments:

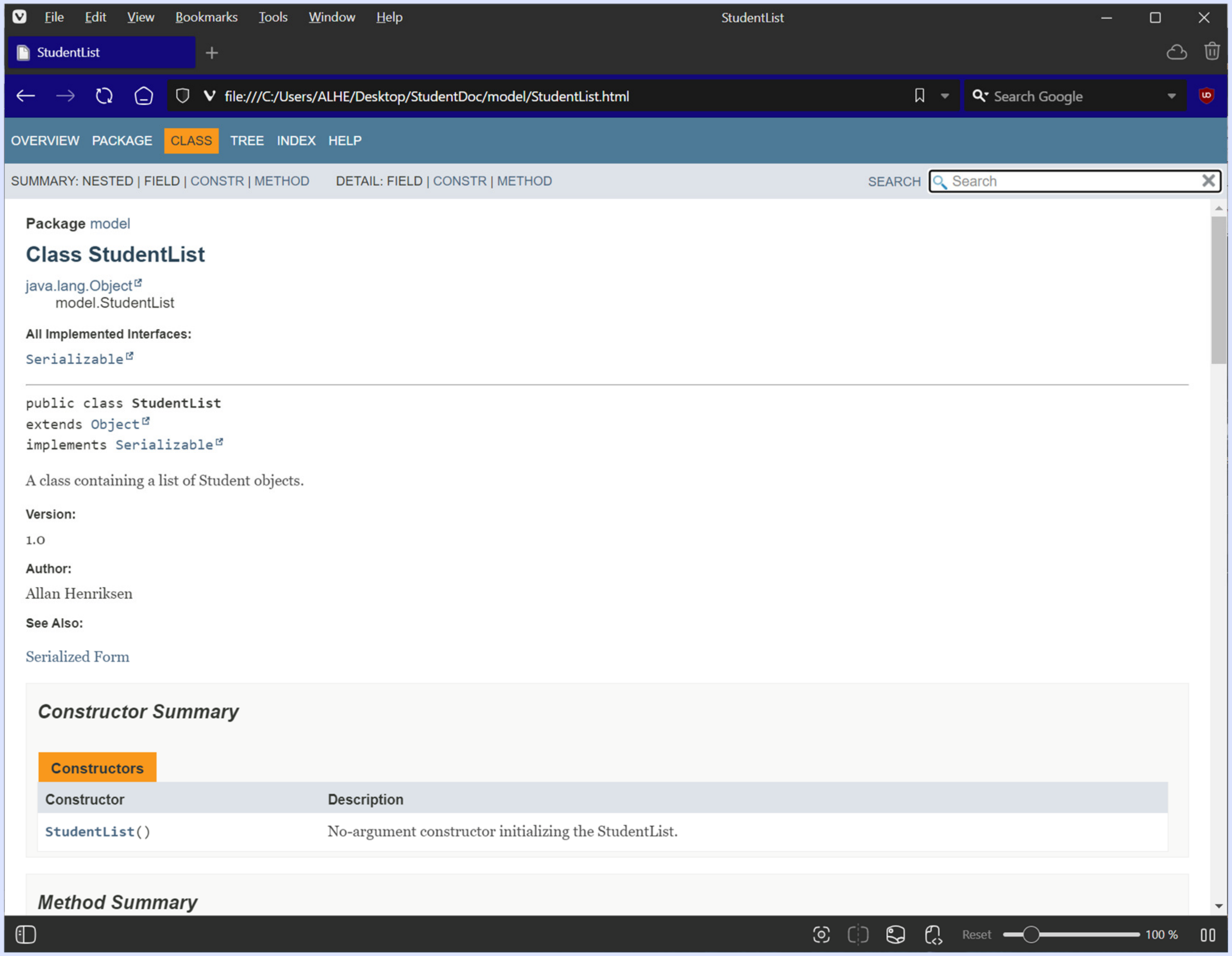
Maximum heap size: megabytes

- [x] Open generated documentation in browser

Choose the folder where the generated files will be stored

Choose which tags to include in the JavaDoc

The generated Javadocs



File Edit View Bookmarks Tools Window Help
StudentList
file:///C:/Users/ALHE/Desktop/StudentDoc/model/StudentList.html
Search Google
OVERVIEW PACKAGE CLASS TREE INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset
100 %

Package model

Class StudentList

java.lang.Object
 model.StudentList

All Implemented Interfaces:

`Serializable`

```
public class StudentList
extends Object
implements Serializable
```

A class containing a list of Student objects.

Version:

1.0

Author:

Allan Henriksen

See Also:

Serialized Form

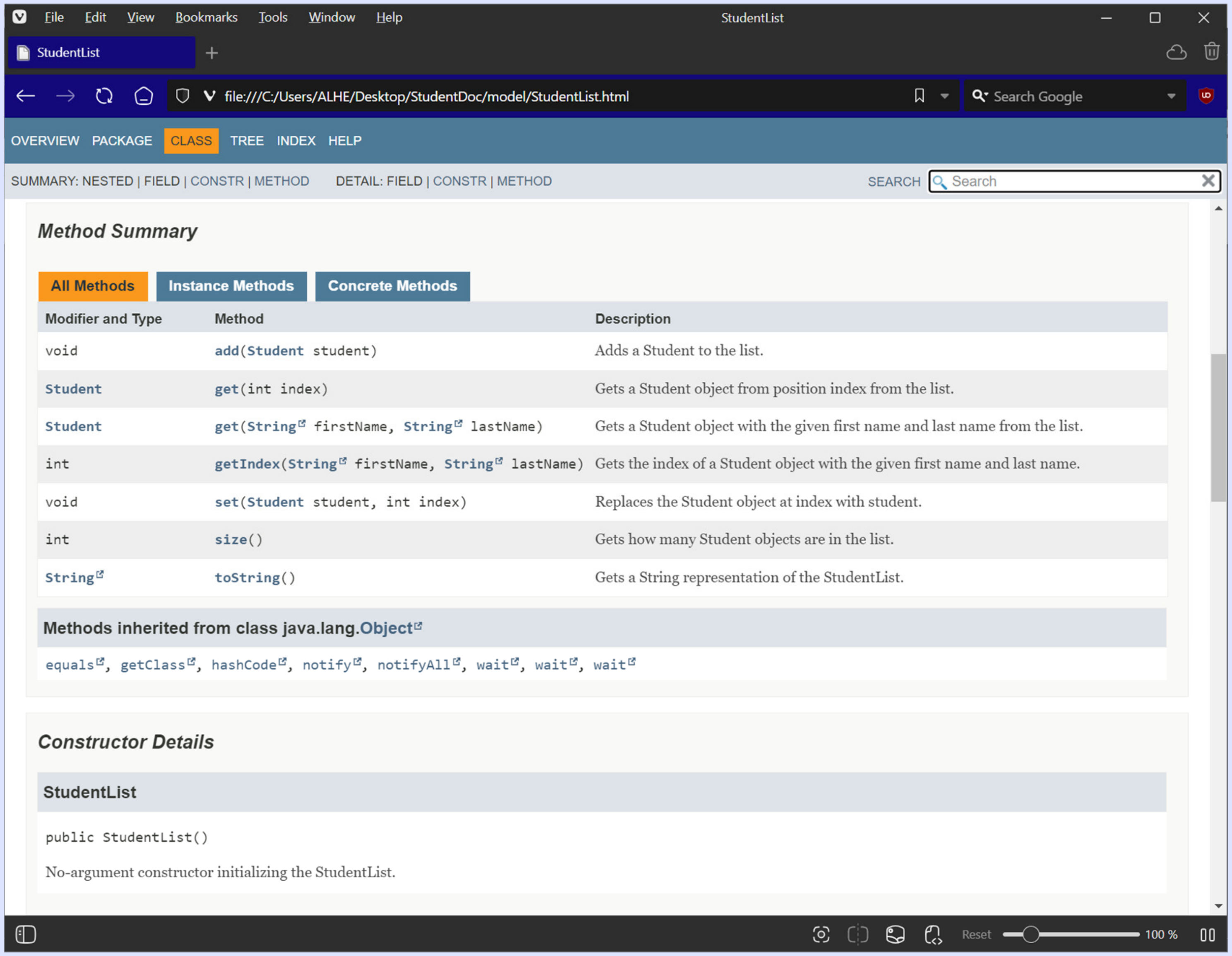
Constructor Summary

Constructors

Constructor	Description
`StudentList()`	No-argument constructor initializing the StudentList.

Method Summary

The generated Javadocs


File Edit View Bookmarks Tools Window Help
StudentList
file:///C:/Users/ALHE/Desktop/StudentDoc/model/StudentList.html
Search Google
OVERVIEW PACKAGE CLASS TREE INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset 100 %

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
`void`	`add(Student student)`	Adds a Student to the list.
`Student`	`get(int index)`	Gets a Student object from position index from the list.
`Student`	`get(String firstName, String lastName)`	Gets a Student object with the given first name and last name from the list.
`int`	`getIndex(String firstName, String lastName)`	Gets the index of a Student object with the given first name and last name.
`void`	`set(Student student, int index)`	Replaces the Student object at index with student.
`int`	`size()`	Gets how many Student objects are in the list.
`String`	`toString()`	Gets a String representation of the StudentList.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, wait, wait, wait`

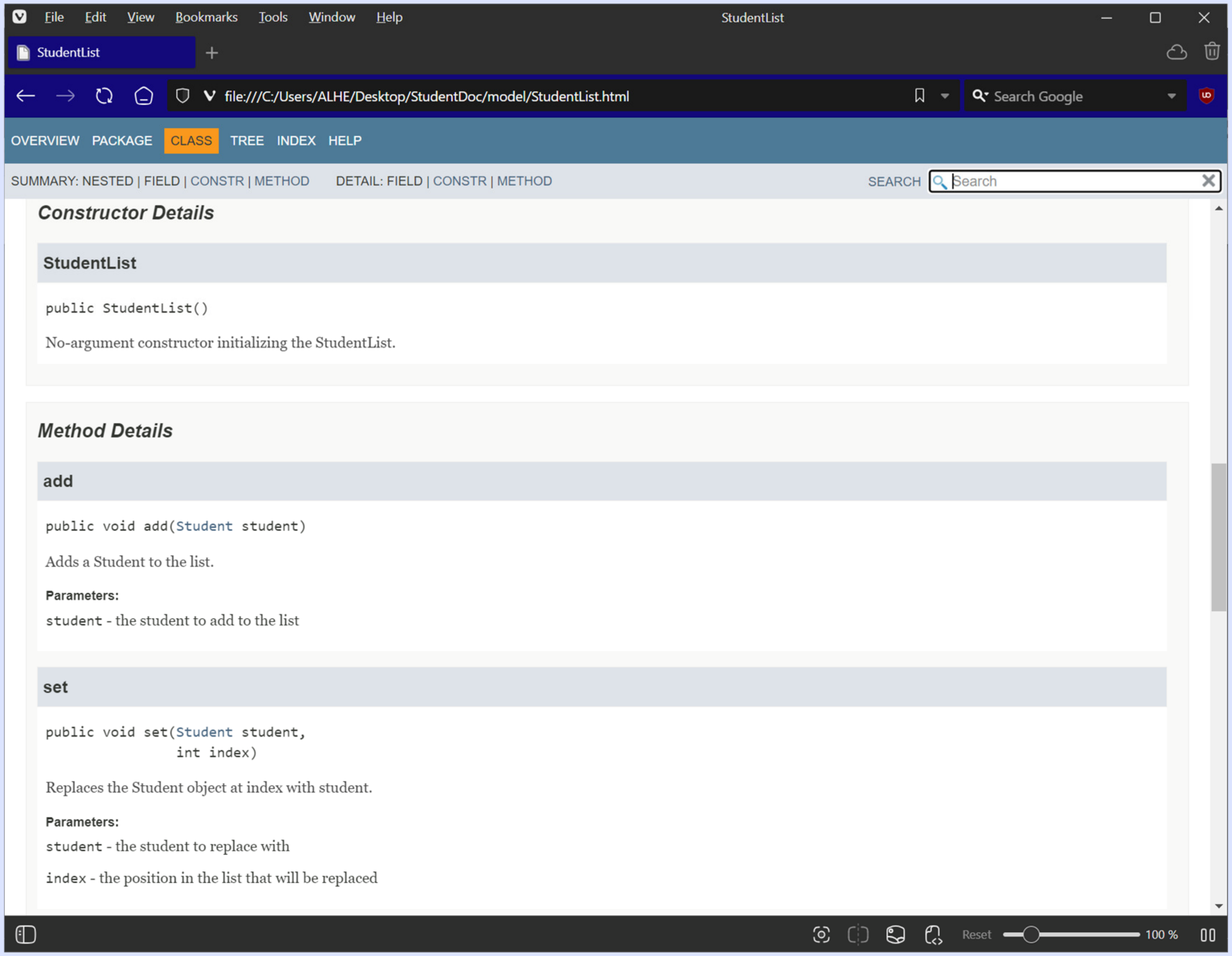
Constructor Details

StudentList

```
public StudentList()
```

No-argument constructor initializing the StudentList.

The generated Javadocs



File Edit View Bookmarks Tools Window Help
StudentList
file:///C:/Users/ALHE/Desktop/StudentDoc/model/StudentList.html
Search Google
OVERVIEW PACKAGE CLASS TREE INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset 100 %

Constructor Details

StudentList

```
public StudentList()
```

No-argument constructor initializing the StudentList.

Method Details

add

```
public void add(Student student)
```

Adds a Student to the list.

Parameters:

`student` - the student to add to the list

set

```
public void set(Student student,
                int index)
```

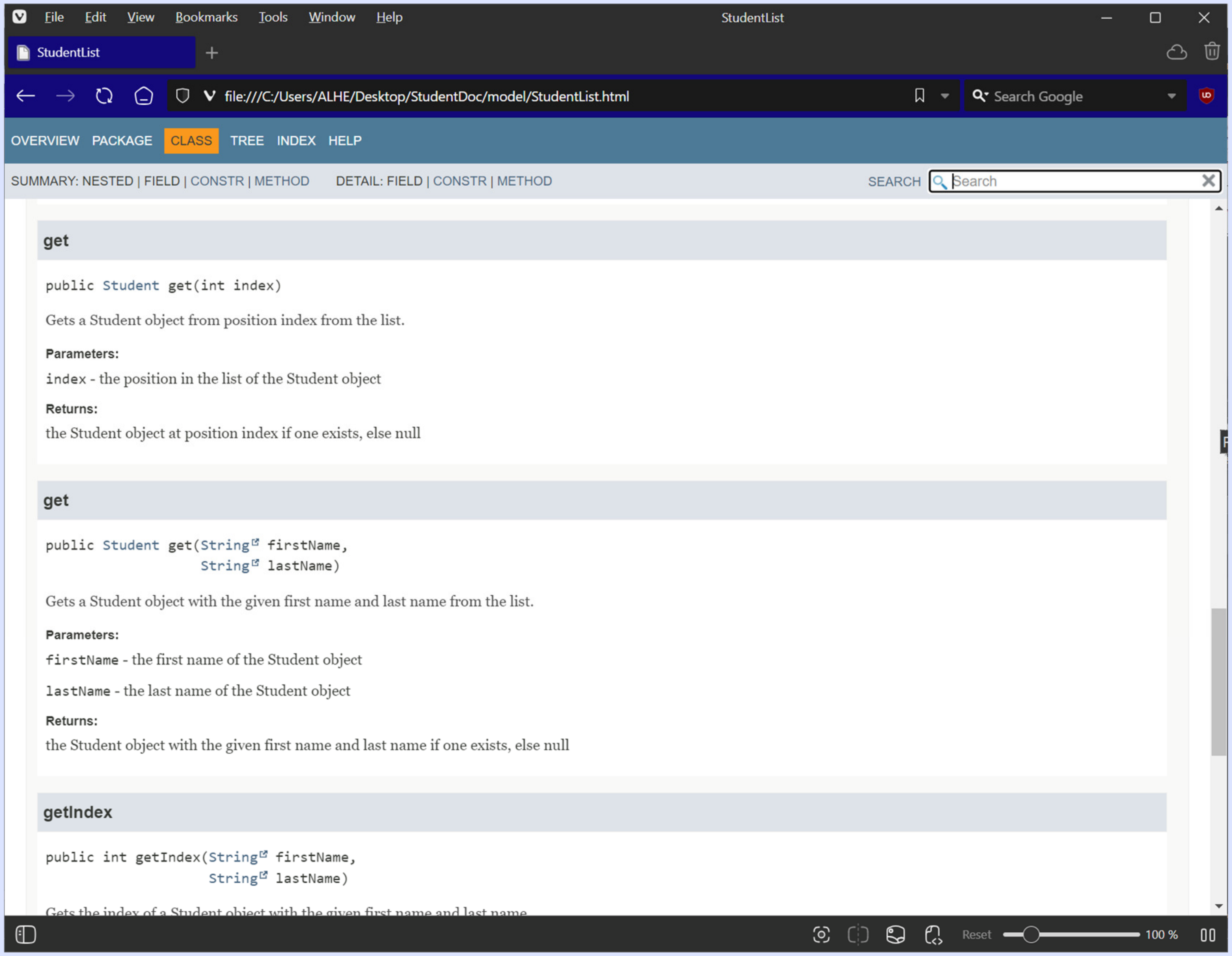
Replaces the Student object at index with student.

Parameters:

`student` - the student to replace with

`index` - the position in the list that will be replaced

The generated Javadocs


File Edit View Bookmarks Tools Window Help
StudentList
file:///C:/Users/ALHE/Desktop/StudentDoc/model/StudentList.html
Search Google
OVERVIEW PACKAGE CLASS TREE INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Reset 100 %

get

```
public Student get(int index)
```

Gets a Student object from position index from the list.

Parameters:

`index` - the position in the list of the Student object

Returns:

the Student object at position index if one exists, else null

get

```
public Student get(String firstName,
                   String lastName)
```

Gets a Student object with the given first name and last name from the list.

Parameters:

`firstName` - the first name of the Student object

`lastName` - the last name of the Student object

Returns:

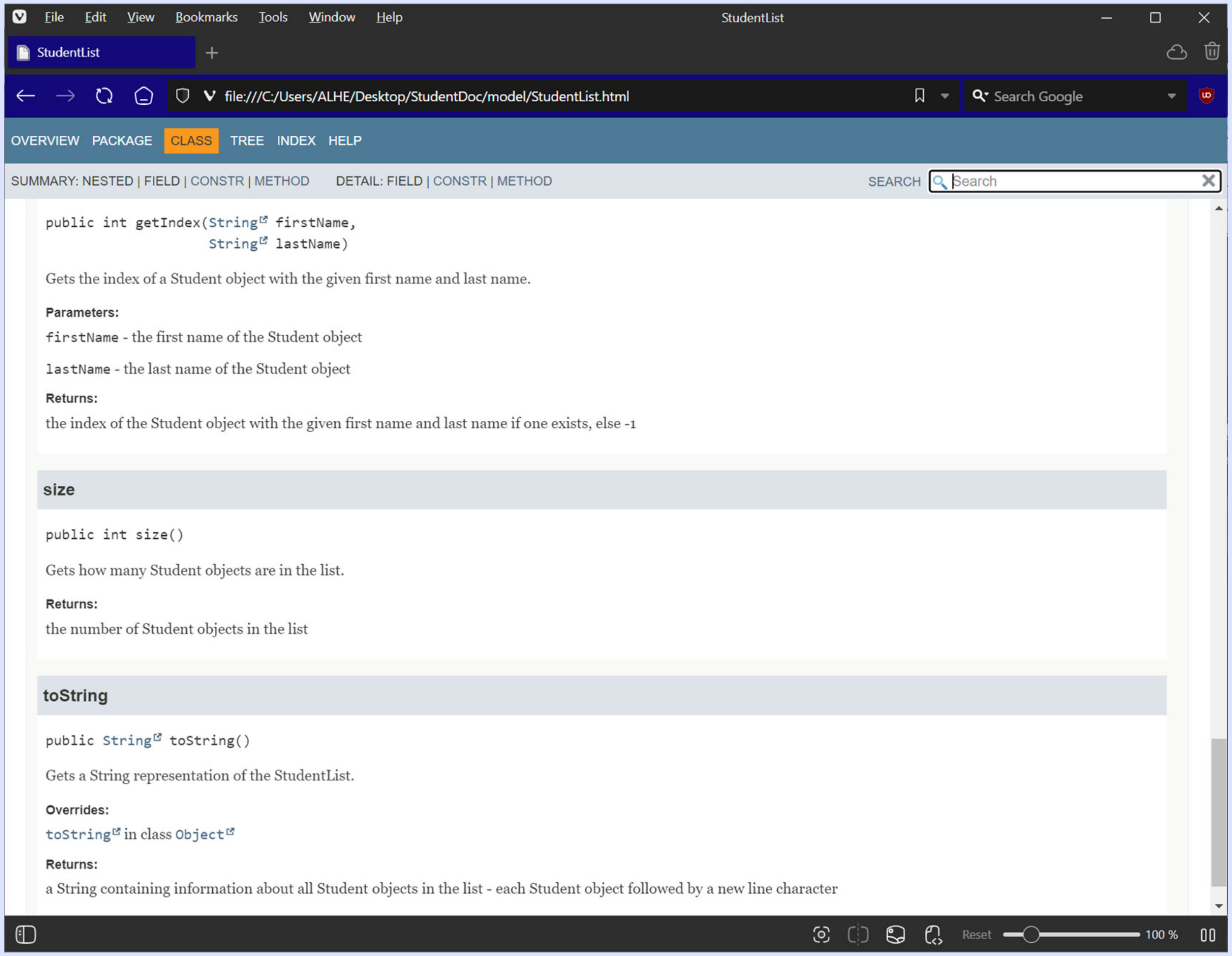
the Student object with the given first name and last name if one exists, else null

getIndex

```
public int getIndex(String firstName,
                    String lastName)
```

Gets the index of a Student object with the given first name and last name

The generated Javadocs


File Edit View Bookmarks Tools Window Help
StudentList
StudentList
file:///C:/Users/ALHE/Desktop/StudentDoc/model/StudentList.html
Search Google
OVERVIEW PACKAGE CLASS TREE INDEX HELP
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
SEARCH
Search
Reset
100 %

```
public int getIndex(String firstName,
                    String lastName)
```

Gets the index of a Student object with the given first name and last name.

Parameters:

`firstName` - the first name of the Student object

`lastName` - the last name of the Student object

Returns:

the index of the Student object with the given first name and last name if one exists, else -1

size

```
public int size()
```

Gets how many Student objects are in the list.

Returns:

the number of Student objects in the list

toString

```
public String toString()
```

Gets a String representation of the StudentList.

Overrides:

`toString` in class `Object`

Returns:

a String containing information about all Student objects in the list - each Student object followed by a new line character