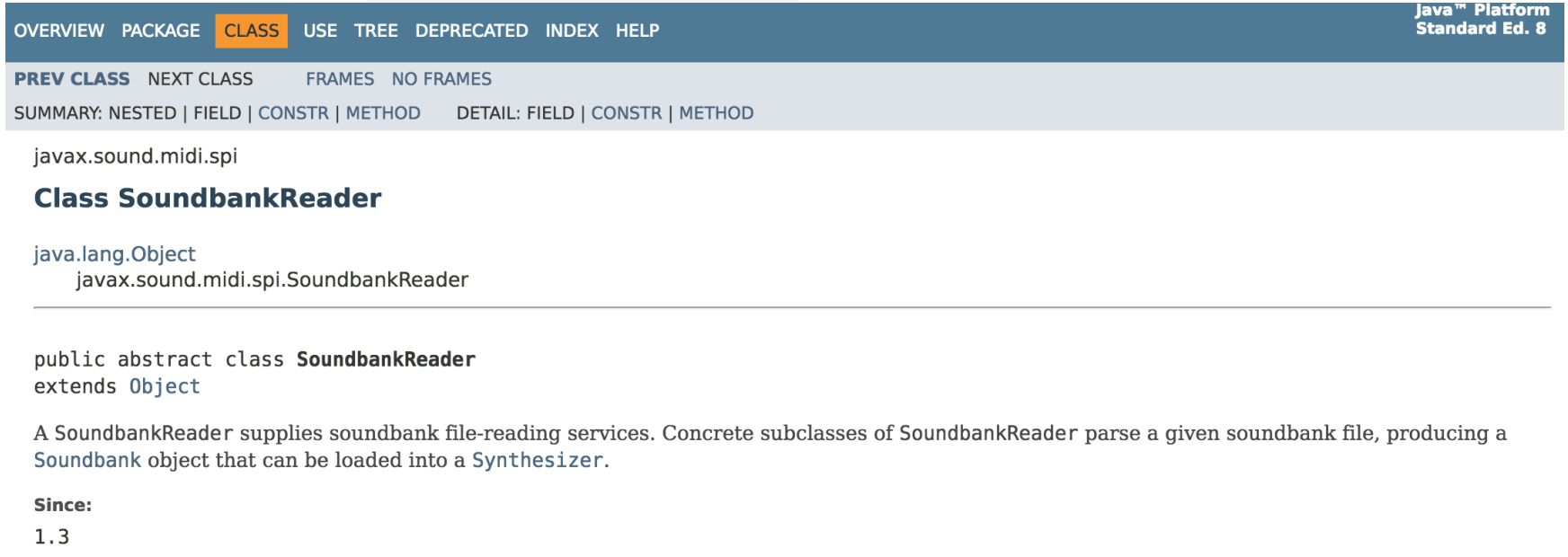
Documentation

We just talked about using Java code written by others.

- What do we import?
- What methods do we call?
- What do those methods do?

What to Import

I want to import `SoundbankReader`. Write the import statement.



OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Java™ Platform Standard Ed. 8

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

javax.sound.midi.spi

Class SoundbankReader

java.lang.Object
    javax.sound.midi.spi.SoundbankReader

```
public abstract class SoundbankReader
extends Object
```

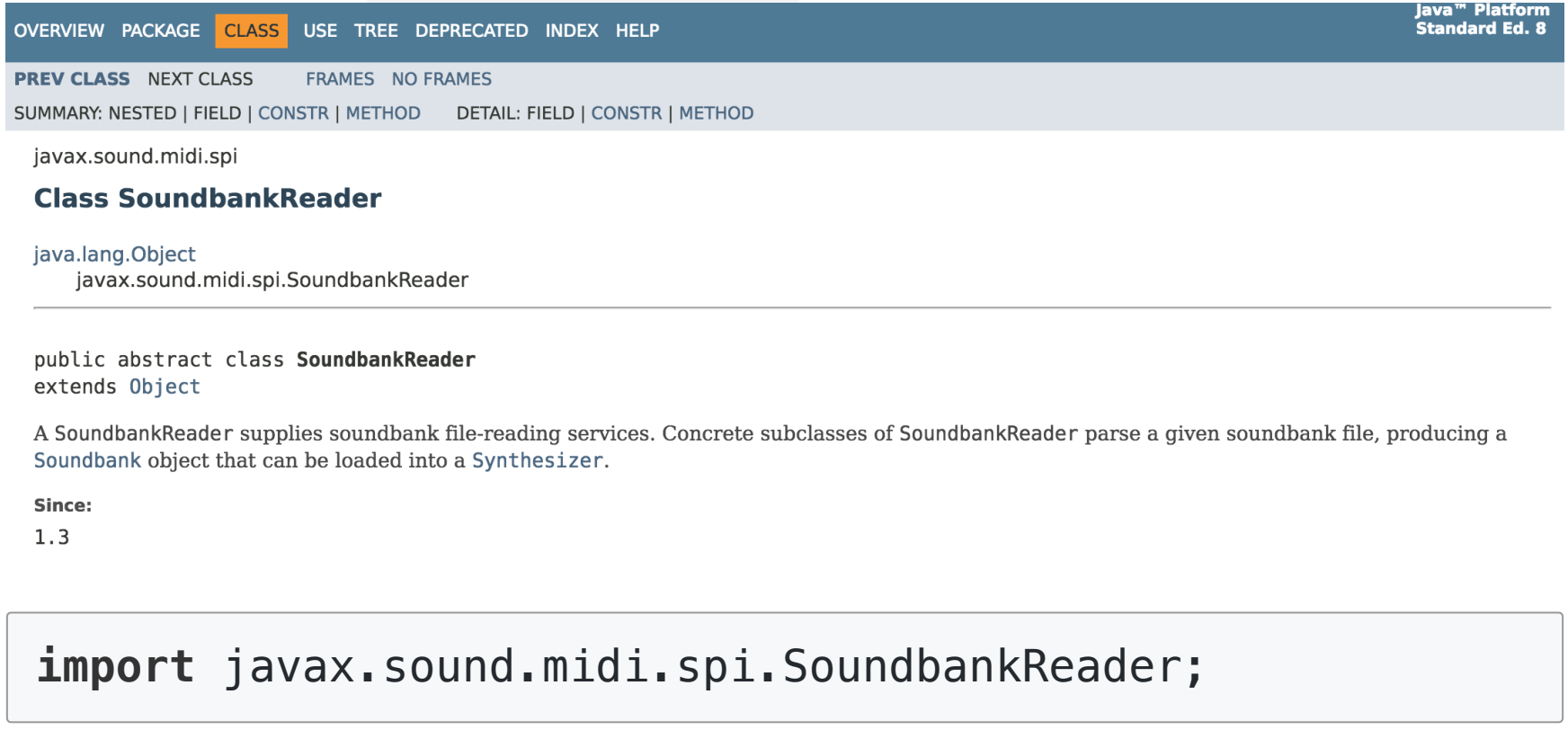
A SoundbankReader supplies soundbank file-reading services. Concrete subclasses of SoundbankReader parse a given soundbank file, producing a Soundbank object that can be loaded into a Synthesizer.

Since:

1.3

What to Import: Answer

I want to import `SoundbankReader`. Write the import statement.



OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Java™ Platform Standard Ed. 8

PREV CLASS NEXT CLASS FRAMES NO FRAMES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

javax.sound.midi.spi

Class SoundbankReader

java.lang.Object
    javax.sound.midi.spi.SoundbankReader

```
public abstract class SoundbankReader
extends Object
```

A SoundbankReader supplies soundbank file-reading services. Concrete subclasses of SoundbankReader parse a given soundbank file, producing a Soundbank object that can be loaded into a Synthesizer.

Since:

1.3

```
import javax.sound.midi.spi.SoundbankReader;
```

Integer Class

What is this class good for?

```
public final class Integer
extends Number
implements Comparable<Integer>
```

The `Integer` class wraps a value of the primitive type `int` in an object. An object of type `Integer` contains a single field whose type is `int`.

In addition, this class provides several methods for converting an `int` to a `String` and a `String` to an `int`, as well as other constants and methods useful when dealing with an `int`.

Implementation note: The implementations of the "bit twiddling" methods (such as `highestOneBit` and `numberOfTrailingZeros`) are based on material from Henry S. Warren, Jr.'s *Hacker's Delight,* (Addison Wesley, 2002).

Public Fields

How would you print out the largest possible Integer?

Field Summary

Fields

Modifier and Type	Field and Description
`static int`	**`BYTES`** The number of bytes used to represent a `int` value in two's complement binary form.
`static int`	**`MAX_VALUE`** A constant holding the maximum value an `int` can have, $2^{31}-1$.
`static int`	**`MIN_VALUE`** A constant holding the minimum value an `int` can have, -2^{31}.
`static int`	**`SIZE`** The number of bits used to represent an `int` value in two's complement binary form.
`static Class<Integer>`	**`TYPE`** The `Class` instance representing the primitive type `int`.

(Reminder: These are static fields.)

(By default this class is already imported being in `java.lang`.)

Public Fields

How would you print out the largest possible Integer?

Field Summary

Fields

Modifier and Type	Field and Description
`static int`	**`BYTES`** The number of bytes used to represent a `int` value in two's complement binary form.
`static int`	**`MAX_VALUE`** A constant holding the maximum value an `int` can have, $2^{31}-1$.
`static int`	**`MIN_VALUE`** A constant holding the minimum value an `int` can have, -2^{31}.
`static int`	**`SIZE`** The number of bits used to represent an `int` value in two's complement binary form.
`static Class<Integer>`	**`TYPE`** The `Class` instance representing the primitive type `int`.

```
System.out.println(Integer.MAX_VALUE);
```

Constructors

How is each constructor called?

Constructor Summary

Constructors

Constructor and Description
`Integer(int value)` Constructs a newly allocated `Integer` object that represents the specified `int` value.
`Integer(String s)` Constructs a newly allocated `Integer` object that represents the `int` value indicated by the `String` parameter.

Constructors

How is each constructor called?

Constructor Summary

Constructors

Constructor and Description
`Integer(int value)` Constructs a newly allocated `Integer` object that represents the specified `int` value.
`Integer(String s)` Constructs a newly allocated `Integer` object that represents the `int` value indicated by the `String` parameter.

```
Integer intNum = new Integer(3);
Integer intStr = new Integer("12");
Integer fail = new Integer("fail"); // What will happen?
```

Methods

How would you get the float value from an `Integer` (without casting)?

`float`	**`floatValue`**`()` Returns the value of this `Integer` as a `float` after a widening primitive conversion.

Methods

How would you get the float value from an `Integer` (without casting)?

`float`	**`floatValue`**`()` Returns the value of this `Integer` as a `float` after a widening primitive conversion.

```
Integer intObj = new Integer(45);
float f = intObj.floatValue();
```

Methods

How is `parseInt` called? What type is returned?

`static int`	**`parseInt`**`(`**`String`** `s)` Parses the string argument as a signed decimal integer.

Methods

How is `parseInt` called? What type is returned?

`static int`	**`parseInt(String s)`** Parses the string argument as a signed decimal integer.

```
int i = Integer.parseInt("3");
```

Creating Javadoc Documentation

Next I will show you how to generate documentation that looks just like Java's documentation.

Class Description

```
/**
 * This is an example class.
 * @author Me
 */
public class Example {
```

Notes:

- Precedes the class.
- Always starts with `/**` and ends with `*/`.
- There are a number of useful tags. I will only show the most common ones.

Field Description

```
        /** Mighty fine constant */
        public static final String CONST = "Unchanging";
```

Javadoc comments can just be a single line.

Often they are when it is a field.

Constructor/Method Description

```
        /**
         * Real constructor
         * @param parameter Give the example an int
         */
        public Example(int parameter) {
```

- Starts with general description of the method/constructor.
- Use the `@param` annotation to mark descriptions of each parameter.
- Form: `@param [name] comment`

Method Description with Return

```
        /**
         * Useful method
         * @param speed in knots
         * @return fun factor as an integer
         */
        public int fly(int speed) {
```

- The `@return` annotation allows for specific comments about what is returned.

Javadoc Command

```
> javadoc Example.java
```

- You can run javadoc on a specific file(s) or a list of packages.
- Will auto-generate all of the HTML and formatting.
- Mimics Oracle's Java API.

Javadoc Command

```
> javadoc Example.java
Loading source file Example.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_144
Building tree for all the packages and classes...
Generating ./Example.html...
Generating ./package-frame.html...
Generating ./package-summary.html...
Generating ./package-tree.html...
Generating ./constant-values.html...
Building index for all the packages and classes...
Generating ./overview-tree.html...
Generating ./index-all.html...
Generating ./deprecated-list.html...
Building index for all classes...
Generating ./allclasses-frame.html...
Generating ./allclasses-noframe.html...
Generating ./index.html...
Generating ./help-doc.html...
```

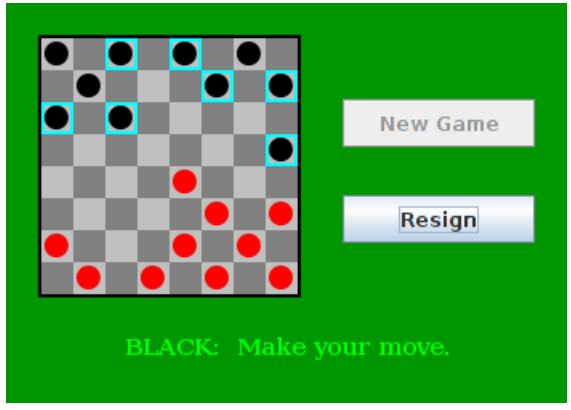
Google It!

There is more information than just documentation from Oracle:

- Example code
- Additional explanations
- Tutorials
- Libraries

Consider the possibilites of programming

Visual Game.



Artificial Intelligence.