Java Programming Language

Object-Oriented Programming

Mario Simaremare, S.Kom., M.Sc.
Program Studi Sarjana Sistem Informasi
Institut Teknologi Del

Objectives

- The objectives of this session is the following:
  - The students are able to describe the basic structure of Java source code.
  - The students are able to compile Java code and run a Java program.

Outlines

1. Java programming language.
2. Java source code anatomy.
3. Java naming convention.
4. Code compilation and program execution.
5. The `main()` entry-point method.
6. Keywords.
7. Troubleshooting.

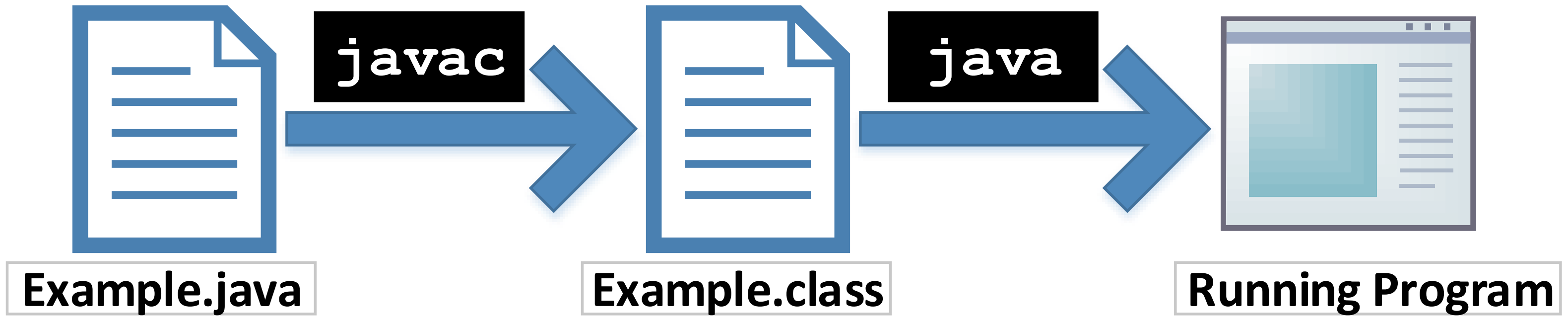
Java Programming Language

Java programming language

- Java is a high-level programming language.
    - It provides abstractions to ease its users (programmers).
- Some core features:
    - Object-orientation: how we perceive a problem,
    - Architecture-neutral (compile once, run anywhere),
    - Dynamic: start small and grow as needed,
    - etc.

Java programming language

- A Java code is written in `.java` source code.
- Later, it is compiled into a Java bytecode in `.class` file.
  - A "half-baked" static program.
  - It is architecture-neutral.
- A Java bytecode can be executed (interpreted).
  - Execution happens on top of Java Runtime Environment.



Java Source Code Anatomy

Java source code anatomy

- It has language specific syntax and structure.
  - Instructions are written in statements.
  - Every statement ends with a semicolon (`;`).
  - Executed in top-down fashion.
  - Occasionally, statements are grouped in blocks.
  - Code blocks are written inside a pair of curly brackets `{}`.

- A program is wrapped in a class with **an entry point**.
  - Program entry point in the form of a static `main()` method.

Java source code anatomy

```
package examples.hello;

import java.lang.*;

public class HelloWorld {

    private static final String DEFAULT_GREETINGS = "Hallo, ";

    public static void main(String[] _args) {
        StringBuilder builder = new StringBuilder(DEFAULT_GREETINGS);

        if (_args.length > 0) {
            builder.append(_args[0]);
        } else {
            builder.append("world");
        }
        builder.append("!");

        System.out.println(builder.toString());
    }

}
```

Package name (optional)

Importing classes form other packages

Class name

Class-level property

Main method, the program entry point

Declaring and instantiating an object

Sending a text to the standard output

Java source code anatomy

- A Java class definition consists of:
  - The class package, an abstract grouping (optional).
    - Affect the artifact structure.
  - A set of import statements (optional).
    - Importing classes outside the class package.
  - A unique class name.
  - Class and instance-level attribute definitions.
  - Class and instance-level method definitions.
  - A `main()` method as the program entry point,

Java Naming Convention

Java naming convention

- Java employ a set of naming convention to make codes more understandable.
  - Packages, classes, interfaces, methods, variables, and constants.

https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html

9 - Naming Conventions

Naming conventions make programs more understandable by making them easier to read. They can also give information about the function of the identifier-for example, whether it's a constant, package, or class-which can be helpful in understanding the code.

Identifier Type	Rules for Naming	Examples
Packages	The prefix of a unique package name is always written in all-lowercase ASCII letters and should be one of the top-level domain names, currently com, edu, gov, mil, net, org, or one of the English two-letter codes identifying countries as specified in ISO Standard 3166, 1981. Subsequent components of the package name vary according to an organization's own internal naming conventions. Such conventions might specify that certain directory name components be division, department, project, machine, or login names.	com.sun.eng com.apple.quicktime.v2 edu.cmu.cs.bovik.cheese

Classes	Class names should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML).	class Raster;
		class ImageSprite;

Interfaces	Interface names should be capitalized like class names.	interface RasterDelegate;
		interface Storing;

Methods	Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.	run();
		runFast();
		getBackground();

Variables	Except for variables, all instance, class, and class constants are in mixed case with a lowercase first letter. Internal words start with capital letters. Variable names should not start with underscore _ or dollar sign $ characters, even though both are allowed.
	
	Variable names should be short yet meaningful. The choice of a variable name should be mnemonic- that is, designed to indicate to the casual observer the intent of its use. One-character variable
		`int i;`
		`char c;`
		`float myWidth;`

```
int             i;
char            c;
float           myWidth;
```

https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html

Compilation and Execution

Code compilation and program execution

- To compile a Java source code the `javac` command is used.
  - The output of a compilation process is Java bytecode.
  - Requires Java Development Kit (JDK).
- To run a Java bytecode the `java` command is used.
  - The static program is loaded, interpreted and turned into a running program.
  - Requires Java Runtime Environment (JRE).

```
javac <source-code>.java
java  <class name without extension>
```

The `main()` Method

The `main()` method

- Every program has an entry point, where the program starts.
  - In Java, `main()` method is the program entry point.
  - The method must be:
    - Defined in class level (`static`).
    - Publicly available (`public` access modifier).
    - Without any return value (`void`).
  - It accepts an array of `String` object to read the given arguments.

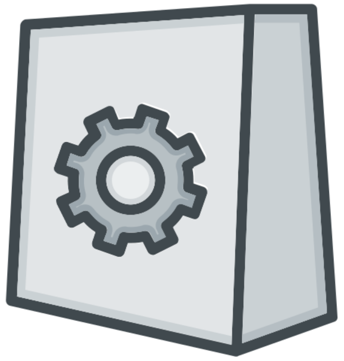
- Should every class has a `main()` method?
  - No, only a program entry class has it.

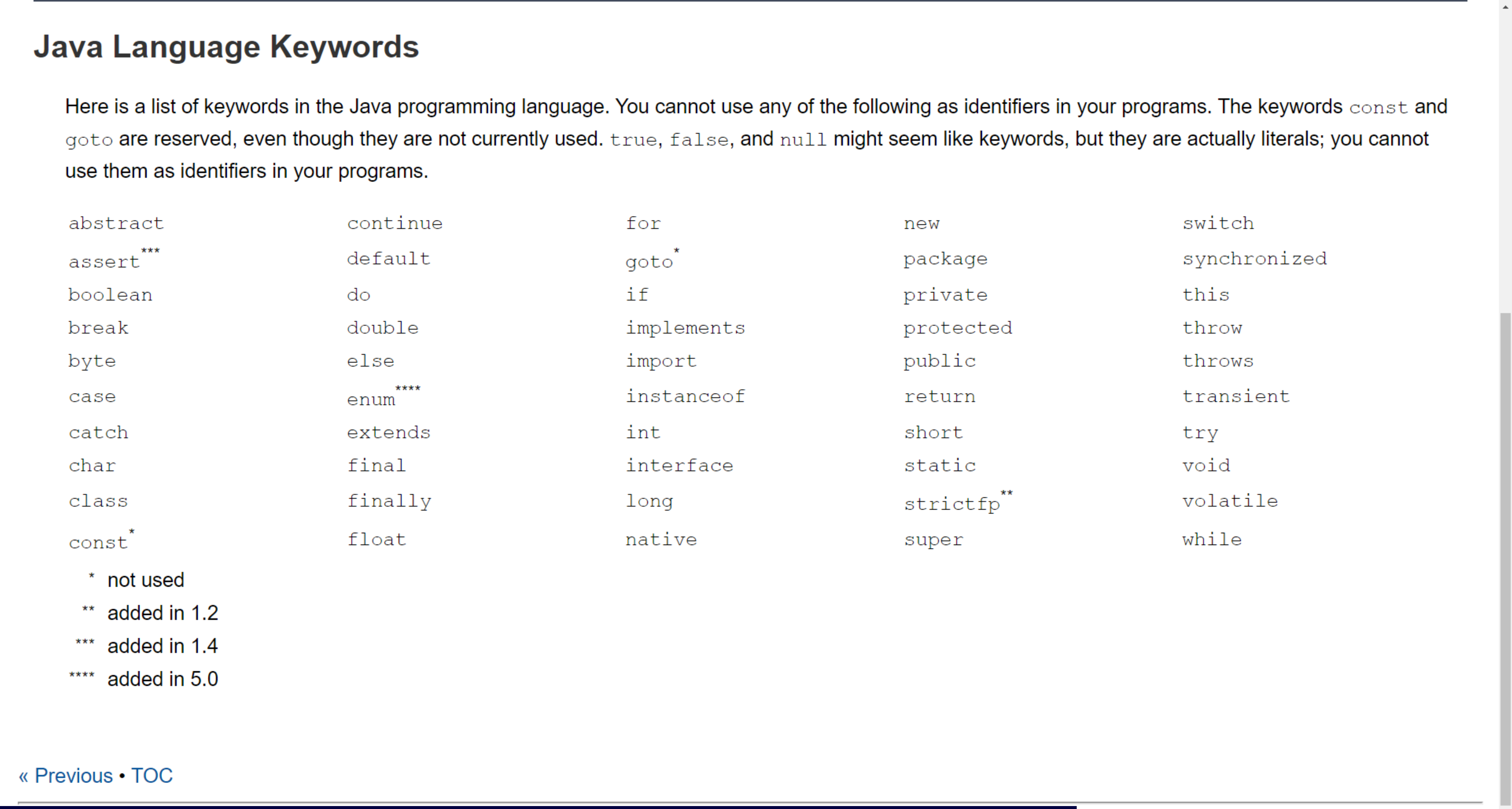
https://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html

Keywords

Keywords

- Keywords are **reserved** words of the language.
  - These keywords have special meaning to the compiler.
  - None of them can be used as identifier.
  - E.g. `class`, `new`, `int`, `if`, `for`, `case`, etc.





Java Language Keywords

Here is a list of keywords in the Java programming language. You cannot use any of the following as identifiers in your programs. The keywords `const` and `goto` are reserved, even though they are not currently used. `true`, `false`, and `null` might seem like keywords, but they are actually literals; you cannot use them as identifiers in your programs.

`abstract`	`continue`	`for`	`new`	`switch`
`assert`***	`default`	`goto`*	`package`	`synchronized`
`boolean`	`do`	`if`	`private`	`this`
`break`	`double`	`implements`	`protected`	`throw`
`byte`	`else`	`import`	`public`	`throws`
`case`	`enum`****	`instanceof`	`return`	`transient`
`catch`	`extends`	`int`	`short`	`try`
`char`	`final`	`interface`	`static`	`void`
`class`	`finally`	`long`	`strictfp`**	`volatile`
`const`*	`float`	`native`	`super`	`while`

*	not used
**	added in 1.2
***	added in 1.4
****	added in 5.0

« Previous • TOC

See https://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html

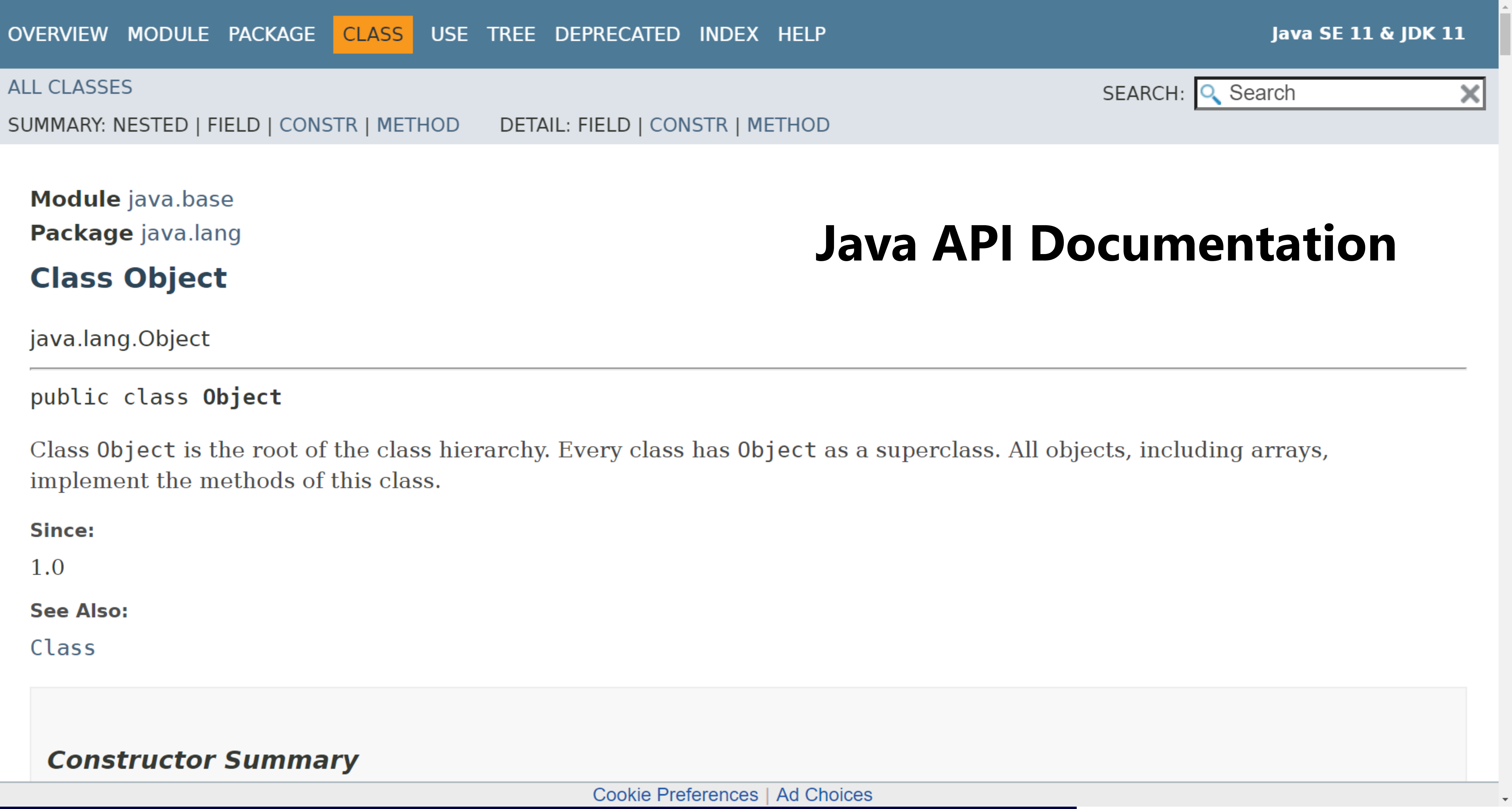
Troubleshooting

Troubleshooting

- Learning a new language and perspective is fun.
  - Problems come and go along the way.

- Steps to troubleshoot:
  - Understand the problem, read the error message carefully.
  - Similar problems might be posted in the Internet.
    - Find it and learn from it!
  - Read the API documentation ← very encouraged.

Java API Documentation



OVERVIEW MODULE PACKAGE CLASS USE TREE DEPRECATED INDEX HELP   Java SE 11 & JDK 11

ALL CLASSES   SEARCH: Search

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

Module java.base
Package java.lang

Class Object

java.lang.Object

```
public class Object
```

Class `Object` is the root of the class hierarchy. Every class has `Object` as a superclass. All objects, including arrays, implement the methods of this class.

Since:

1.0

See Also:

`Class`

Constructor Summary

Cookie Preferences | Ad Choices

See https://docs.oracle.com/en/java/javase/11/docs/api/index.html

References

- Cay Horstman. Core Java.
- Matt Weisfeld. The Object-Oriented Thought Process.

Thank you