Java Cheatsheet

Basic Structure and Execution

A Java program is composed of classes and a `main` method.

```
// File name must match the class name (Main.java)
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

- Compilation and Execution:
  - `javac Main.java`: Compile (generates `Main.class` bytecode).
  - `java Main`: Run the compiled class.
- Package: Used for organizing and avoiding name conflicts. `package com.example;`

Variables and Data Types

- Primitive Types: `byte`, `short`, `int`, `long`, `float`, `double`, `char`, `boolean`.
- Reference Types: Classes, Arrays, Interfaces (e.g., `String`, `Integer`).
- Casting:
  - Implicit (Widening): `double d = 10;`
  - Explicit (Narrowing): `int i = (int) 3.14;`
- Final: Used to declare constants. `final double PI = 3.14;`

Control Flow

- `if-else` / `switch`:

```
if (condition) { ... } else { ... }

switch (variable) {
    case 1 ->
System.out.println("one"); // Java 12+ switch expressions
    default ->
System.out.println("others");
}
```

- Loops:
  - `for (int i = 0; i < 5; i++) { ... }`
  - Enhanced `for`: `for (String s : list) { ... }`
  - `while` / `do-while` loops.

Object-Oriented Programming (OOP)

- Class and Constructor:

```
class Person {
    private String name;
    public Person(String name)
{ this.name = name; }
}
```

- Inheritance: `class Student extends Person { ... }`
- Interface: `interface Drivable { void drive(); }`
- Enum: `enum Level { LOW, MEDIUM, HIGH }`
- Access Modifiers: `public`, `protected`, `private`, (default/package-private).

Exception Handling

- `try-catch-finally`:

```
try {
    // Code that might throw an exception
} catch (Exception e) {
    // Handle exception
} finally {
    // Always executed
}
```

- `throw` vs `throws`: `throw` triggers an exception; `throws` declares that a method might throw an exception.
- Try-with-resources: Automatically closes resources. `try (BufferedReader br = new BufferedReader(...)) { ... }`

Collections Framework

- `List`: `ArrayList`, `LinkedList`
- `Set`: `HashSet`, `TreeSet` (sorted)
- `Map`: `HashMap`, `TreeMap` (sorted key-value pairs)
- `Queue` / `Stack`: `PriorityQueue`, `Deque`

Generics

```
public class Box<T> {
    private T t;
    public void set(T t) { this.t = t; }
}
```

Lambda and Stream API (Java 8+)

- Lambda: `(parameters) -> expression`
- Stream API:

```
List<Integer> list = Arrays.asList(1, 2, 3, 4, 5);
int sum = list.stream()
              .filter(n -> n % 2 == 0)
              .mapToInt(n -> n)
              .sum();
```

Input and Output (I/O)

- Console: `Scanner sc = new Scanner(System.in);`
- File I/O (NIO.2): `Path path = Paths.get("file.txt");`

Advanced Concepts

- Reflection: Inspect classes, methods, fields at runtime.
- Annotations: Metadata for code. `@Override`, `@Deprecated`, `@SuppressWarnings`.
- Multithreading: `Thread` class, `Runnable` interface, `ExecutorService`.

Google Style Guide

Naming Conventions

- Package: `com.example.project` (all lowercase, concatenated)
- Class/Interface: `UpperCamelCase` (nouns)
- Method: `lowerCamelCase` (verbs)
- Constant: `UPPER_SNAKE_CASE` (static final)
- Local Variables/Fields: `lowerCamelCase`
- Type Variables: `T`, `E` (single capital letters) or `RequestT` (T suffix)

Formatting

- Indentation: 2 spaces (no tabs)
- Line Length: Max 100 characters
- Braces: K&R style (no newline before the opening brace {)
- Empty Blocks: Can be expressed concisely as {}, except for multi-block statements.
- Statements: Write only one statement per line.

Programming Practices

- Wildcard Imports: Forbidden (`import java.util.*;` is not allowed).
- Override: The `@Override` annotation must be used.
- Exceptions: Do not ignore exceptions (if you must, specify why in a comment).
- Static Members: Access via class name (`MyClass.staticMethod()`).
- Modifiers: Follow JLS recommended order (`public protected private abstract ...`).

Documentation (Javadoc)

- Required for all public classes and members. The first sentence should be a summary fragment, not a full sentence.
- Block Tags: Write in the order `@param`, `@return`, `@throws`, `@deprecated`.

Annotations

- Write a single annotation on its own line (except for annotations without arguments).

Variables and Literals

- Local Variables: Declare near the point of first use to minimize scope.
- Long Literals: Use `L` suffix (avoid lowercase `l`).