# Java Coding Standards Cheat Sheet

*Note: Comments are worth much of your mark. If your comments aren't good, your code is not good.*

## 1. JavaDoc Comments Are Critical

Each method must have enough JavaDoc comments that another developer can verify that the method code is correct. Some methods may need a LOT of comments.

## 2. Declare, Then Initialize

```
// declaring altogether

final int x;

final int y;

final int z;

final String a;

// then, later, initializing often together

x = 3;

y = 3;

z = 5;

a = "hi";

// then, later, using

System.out.println(a.toUpperCase());

System.out.println(x + y + z);
```

## 3. All Parameters Must Be final

Wrong:

```
private String getFullName(String first, String last)
```

Right:

```
private String getFullName(final String first, final String last)
```

## 4. Use Only What We've Learned

Use only techniques taught in class. Homework tests your skill with what we've covered, nothing more.

## 5. Explicit Units Always

Wrong: price, weight, date

Right: priceUsd, weightKg, datePublished

## 6. No Magic Numbers (except loop counters)

Use constants instead of hard-coded values, except for loop counters like 'for (int i = 0; ...)'

Example:

```
private static final double TAX_RATE = 0.10;

private static final int PERCENT_CONVERSION = 100;

private static final double BASE_MULTIPLIER = 1.0;

private static final double TAX_MULTIPLIER = BASE_MULTIPLIER + TAX_RATE;

System.out.println("Price has " + (TAX_RATE * PERCENT_CONVERSION) + "% tax added");
```

The rule is that numbers must be declared exactly ONCE: either as a symbolic constant, or centralized in ONE spot (e.g. within the method in which it is used).

## 7. Verbs Are for Methods, Not Variables

| | |
|---|---|
| Wrong: | private final boolean isHappy; |
| Right: | private final boolean happy; |

## 8. Use Proper Packages

Always use a lowercase, reverse domain, meaningful, and unique package name.

Wrong: package WhoCaresNotME;

Right: package ca.bcit.comp2522.petmanager;

## 9. JavaDoc All Public Elements

All non-private classes, constructors, and methods must have JavaDoc comments.

## 10. Instance Variables must be safe.

- private

- final (unless they could change, which is relatively rare)

## 11. Class-Level JavaDoc Required

Each class needs a JavaDoc with a full sentence description, @author, and @version.

## 12. One Method = One Action

Split methods that do more than one thing.

| | |
|---|---|
| Right: | calculateDiscountCad() |
| Wrong: | calculateDiscountCadAndPrintReceipt() |

## 13. Method Naming = camelCase Verbs

Use verbs, no abbreviations unless absolutely well known.

| | |
|---|---|
| Right: | getTotalCad() |
| Wrong: | total() |

## 14. Clear Variable Names

Be descriptive.

| | |
|---|---|
| Right: | emailSubject, weightKg |
| Wrong: | subj, wt, x |

## 15. One Class = One Responsibility

| | |
|---|---|
| Right: | InvoiceCalculator, TaxHelper |
| Wrong: | Utils |

## 16. No Abbreviations (unless well known) or Slang

| | |
|---|---|
| Right: | quantityKg, emailAddress |
| Wrong: | qty, addr, disc |

## 17. Constants Are Capitalized, Unit-Specific

Example:

```
private static final int MAX_USERNAME_LEN = 30;
```

## 18. No System.out.print in Final Code

Use return values or logging instead of print statements. They're only for debugging.

## 19. Always Use Braces

Even for one-line conditions.

Right:

```
if (happy)
{
   doThing();
}
```

## 20. File Name = Public Class Name

Class: PetManager -> File: PetManager.java

## 21. final Applies to Everything

Make methods final if they shouldn't be overridden.

Make classes final if they shouldn't be extended.

Use final for for-each and catch'd variables too.

| | |
|---|---|
| Wrong: | for(String name : names) |
| Right: | for(**final** String name : names) |
| Wrong: | catch(IOException e) |
| Right: | catch(**final** IOException e) |

## 22. Always Code to the Interface

Never declare collections using concrete classes like ArrayList, HashMap, etc.

Declare as List or Map, then initialize separately—and of course, everything must be final unless it truly needs to change.

Right:

```
// declaring

private final List<String> names;

private final Map<String, Integer> scores;

// initializing

names = new ArrayList<>();

scores = new HashMap<>();
```

Wrong:

```
private final ArrayList<String> names = new ArrayList<>();

private final HashMap<String, Integer> scores = new HashMap<>();
```