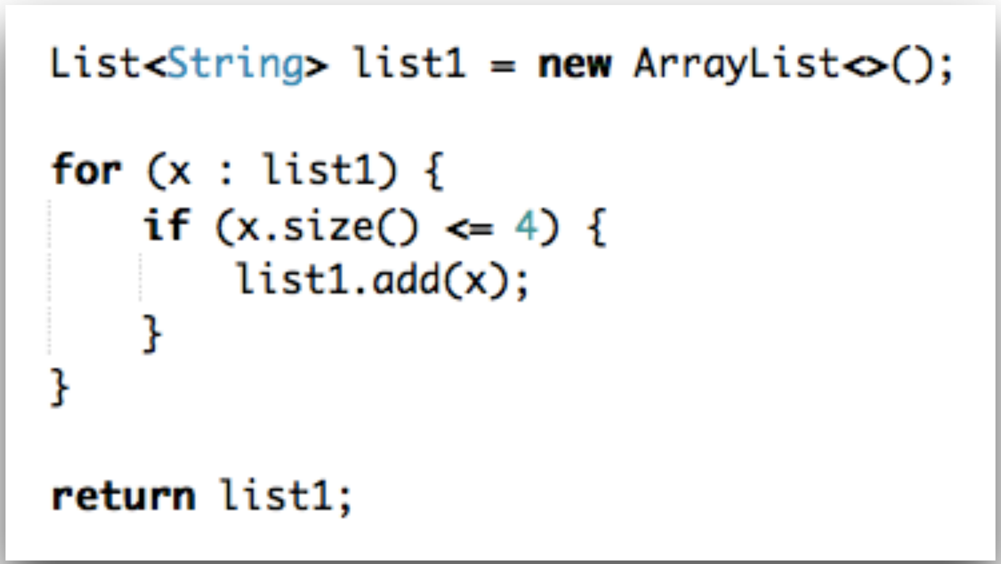
Naming is hard

Dino Kovač

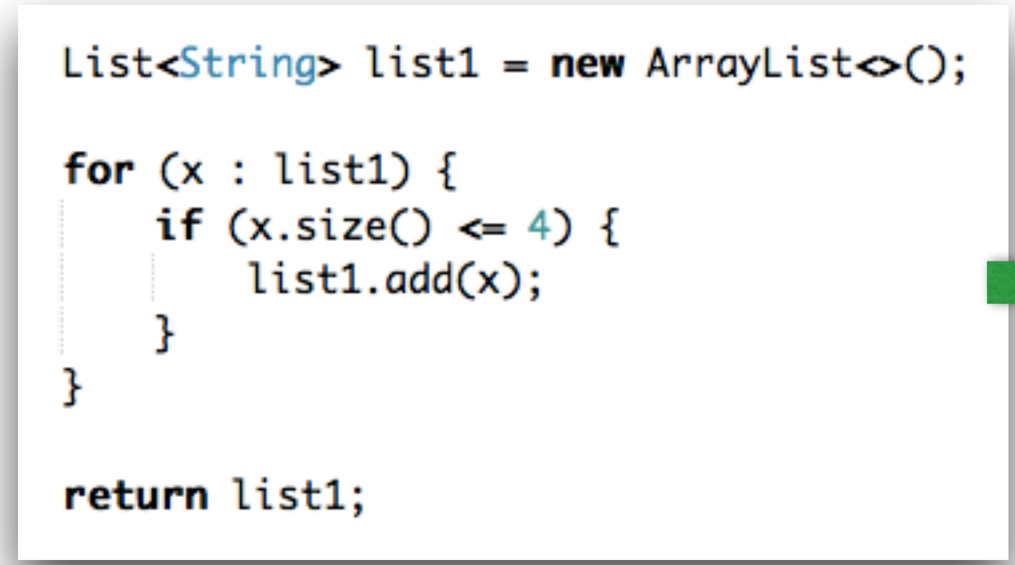
INFINUM

Why is naming important?

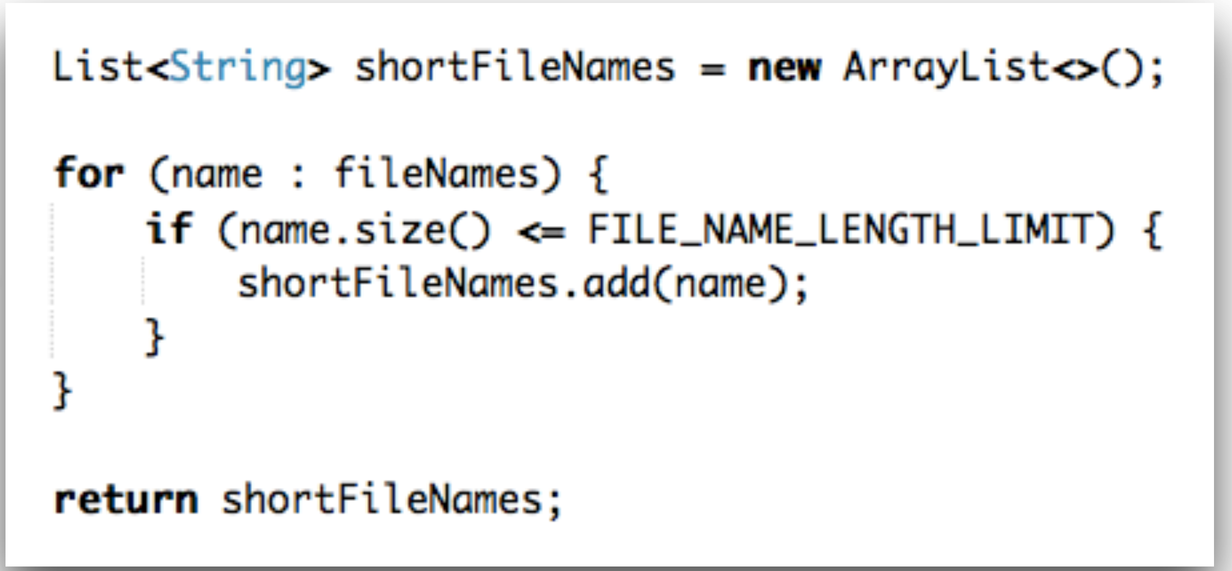


```
List<String> list1 = new ArrayList<>();

for (x : list1) {
    if (x.size() <= 4) {
        list1.add(x);
    }
}

return list1;
```

Why is naming important?

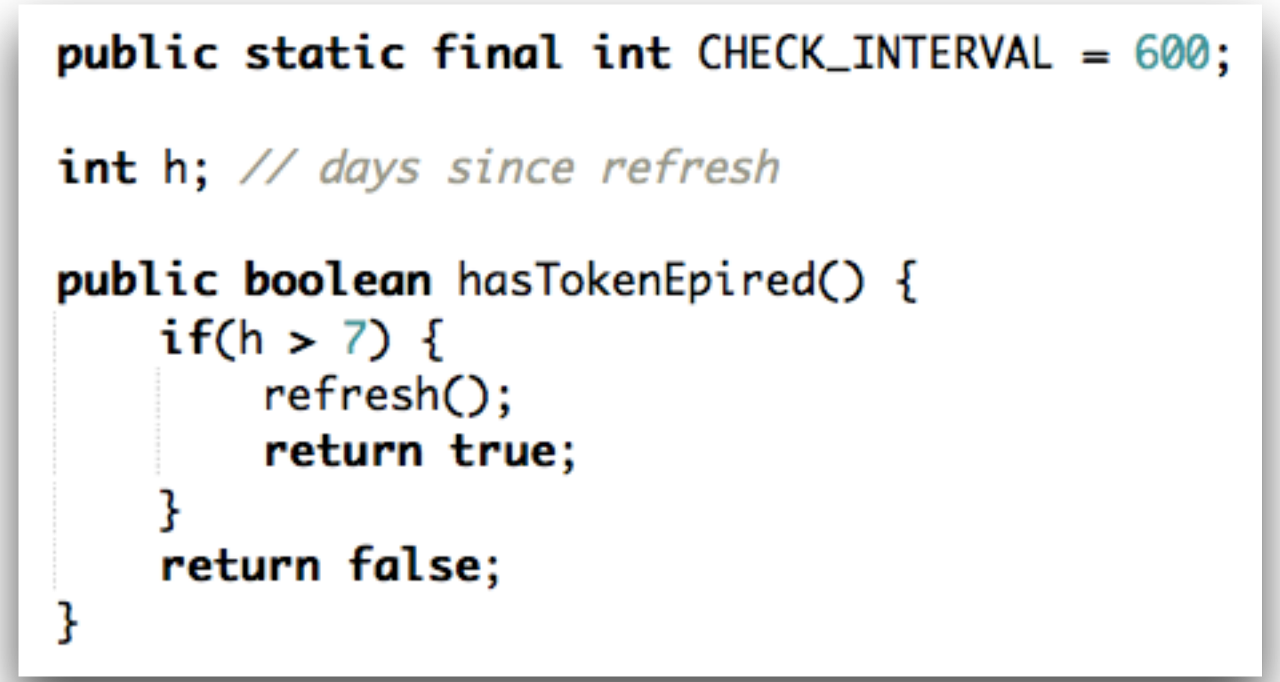


```java
List<String> list1 = new ArrayList<>();

for (x : list1) {
    if (x.size() <= 4) {
        list1.add(x);
    }
}

return list1;
```



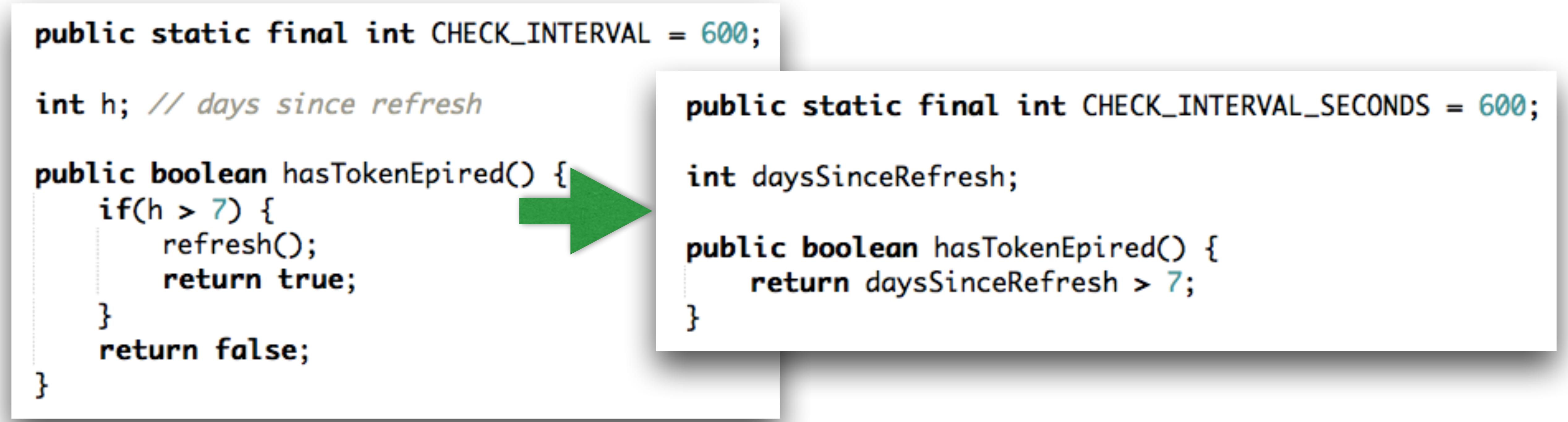
```java
List<String> shortFileNames = new ArrayList<>();

for (name : fileNames) {
    if (name.size() <= FILE_NAME_LENGTH_LIMIT) {
        shortFileNames.add(name);
    }
}

return shortFileNames;
```

Reveal your intent



```
public static final int CHECK_INTERVAL = 600;

int h; // days since refresh

public boolean hasTokenEpired() {
    if(h > 7) {
        refresh();
        return true;
    }
    return false;
}
```

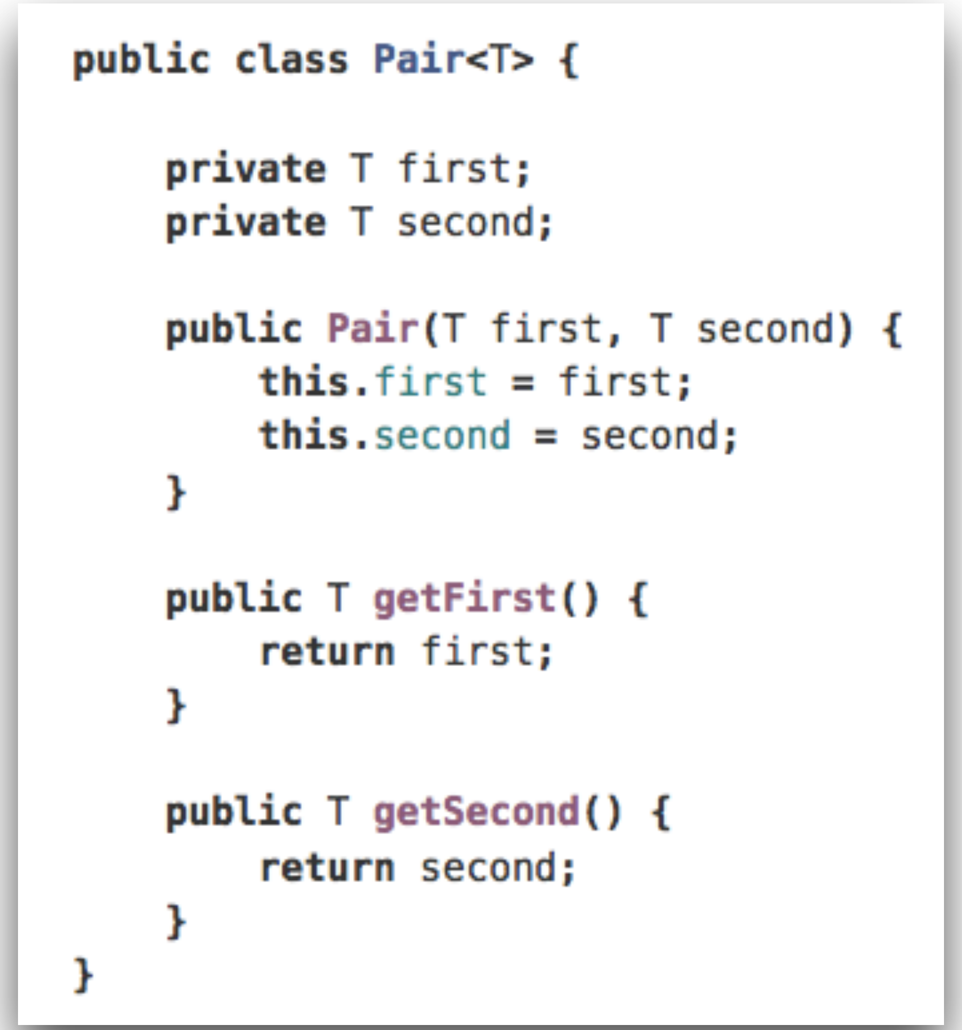
Reveal your intent



```
public static final int CHECK_INTERVAL = 600;

int h; // days since refresh

public boolean hasTokenEpired() {
    if(h > 7) {
        refresh();
        return true;
    }
    return false;
}
```

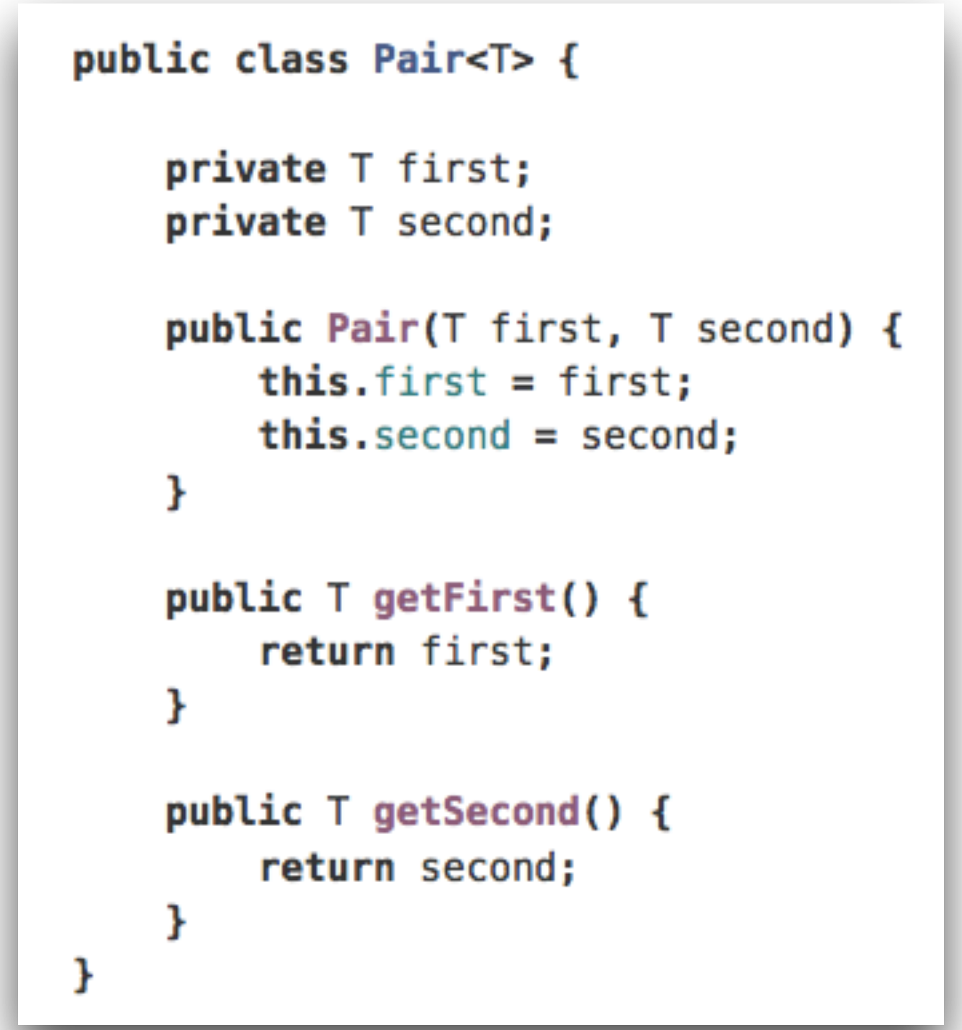
```
public static final int CHECK_INTERVAL_SECONDS = 600;

int daysSinceRefresh;

public boolean hasTokenEpired() {
    return daysSinceRefresh > 7;
}
```

Avoid disinformation

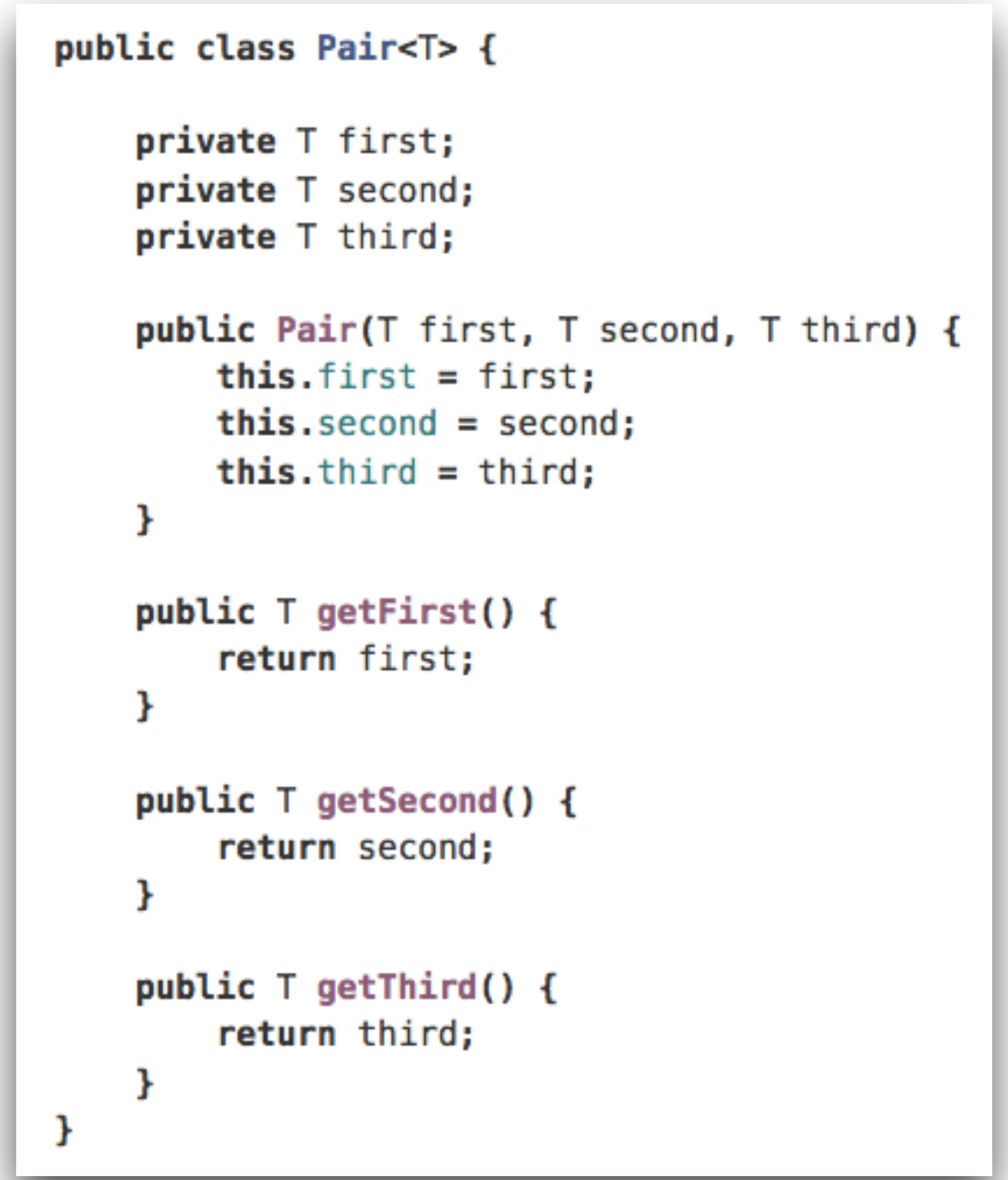


```java
public class Pair<T> {

    private T first;
    private T second;

    public Pair(T first, T second) {
        this.first = first;
        this.second = second;
    }

    public T getFirst() {
        return first;
    }

    public T getSecond() {
        return second;
    }
}
```

Avoid disinformation

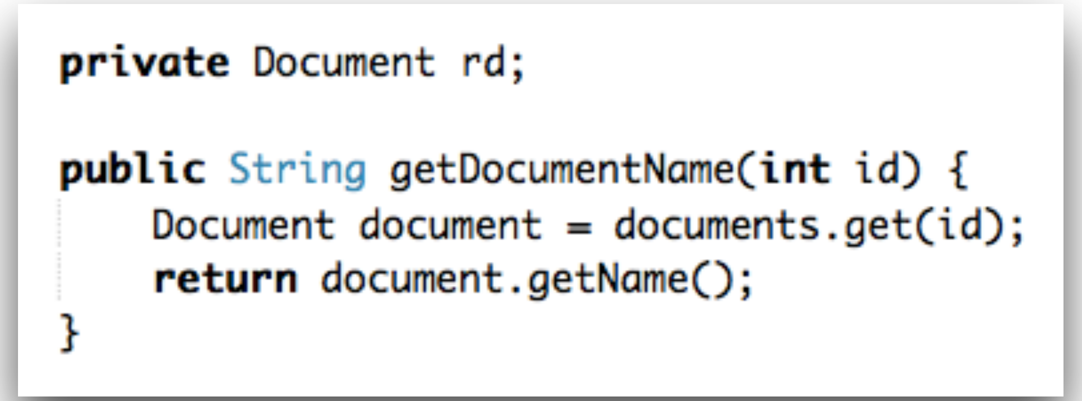


```java
public class Pair<T> {

    private T first;
    private T second;

    public Pair(T first, T second) {
        this.first = first;
        this.second = second;
    }

    public T getFirst() {
        return first;
    }

    public T getSecond() {
        return second;
    }
}
```



```java
public class Pair<T> {

    private T first;
    private T second;
    private T third;

    public Pair(T first, T second, T third) {
        this.first = first;
        this.second = second;
        this.third = third;
    }

    public T getFirst() {
        return first;
    }

    public T getSecond() {
        return second;
    }

    public T getThird() {
        return third;
    }
}
```

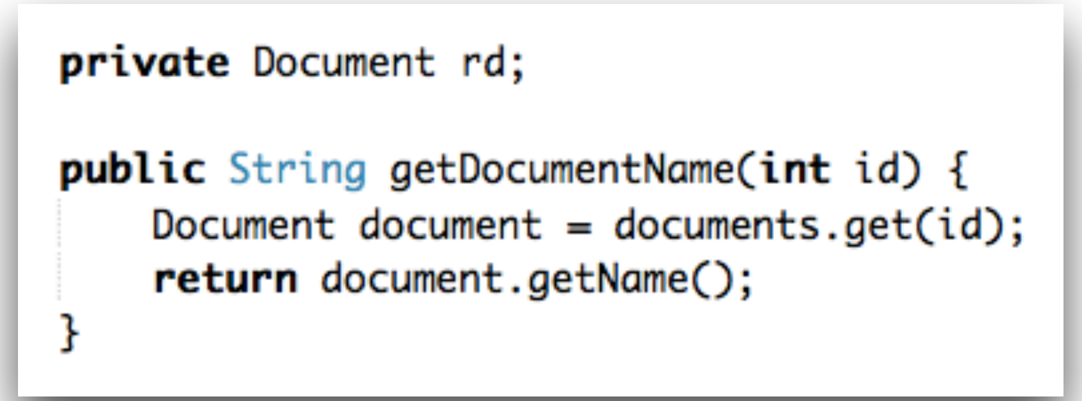
The Scope Rule - variables



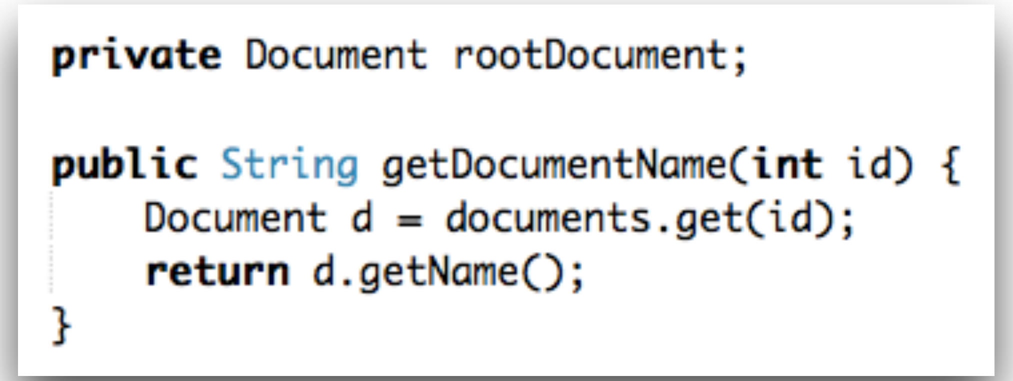
```
private Document rd;

public String getDocumentName(int id) {
    Document document = documents.get(id);
    return document.getName();
}
```

The Scope Rule - variables

- the length of the variable name increases with its scope



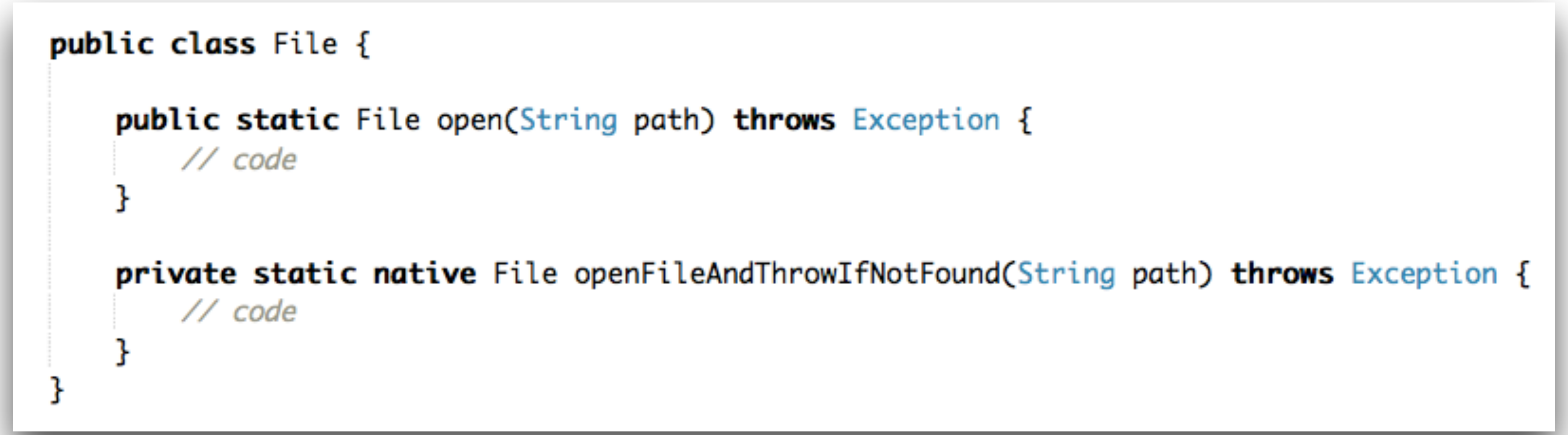
```java
private Document rd;

public String getDocumentName(int id) {
    Document document = documents.get(id);
    return document.getName();
}
```



```java
private Document rootDocument;

public String getDocumentName(int id) {
    Document d = documents.get(id);
    return d.getName();
}
```

The Scope Rule - methods

- public functions tend to be general so they should have short general names
- nobody likes to use a function called openFileAndThrowIfNotFound



```java
public class File {

    public static File open(String path) throws Exception {
        // code
    }

    private static native File openFileAndThrowIfNotFound(String path) throws Exception {
        // code
    }
}
```

Resources

- https://class.stanford.edu/c4x/Engineering/CS144/asset/Naming.pdf
- http://cleancoders.com/episode/clean-code-episode-2/show