Inheritance & Interfaces and Code Layout

The goal of code layout/formatting is to show logical structure

Good layout is shows intention, is consistent, improves readability, and withstands modification.

Slides adapted from Craig Zilles

Inheritance

- **Super-type / Sub-type (extends in Java)**
  - **IsA** relationship; the sub-type isA version of super-type
- **Abstract:**
  - Cannot be instantiated, but describes the interface of what a given type can do.
- **Protected:**
  - Public to my sub-classes (transitively), private to others

Casting in Java

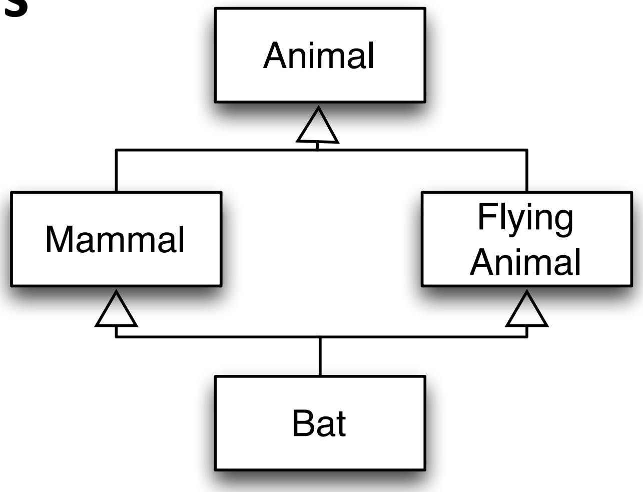
- **What if you have an object reference in a super type and you want to access its sub-type only functionality?**

- **If you _know_ what the sub-type is, just cast it:**
  - SuperType x = new SubType();
  - SubType xAsSubType = (SubType)x; // will except if wrong

- **If you aren't sure, then ask: instanceof**
  - if (x instanceof SubType) {
  -     then cast

Interfaces vs. Abstract Base Classes

- **Java objects can only extend one other class**
  - “single inheritance”
- **Sometimes logical inheritance hierarchies aren’t trees**



- **Java provide Interfaces**
  - You can ‘implement’ any number of interfaces
  - List and Map are interfaces, while ArrayList and HashMap are classes

CourseGrades

- What Section are you on?

A. Section 1
B. Section 2
C. Section 3
D. Section 4
E. All Done

Which is best?

A) `for(int i=0;i<str.length();i++){`

B) `for (int i=0; i<str.length(); i++) {`

C) `for (int i = 0; i < str.length(); i++) {`

D) `for (int i = 0 ; i < str.length() ; i ++) {`

E) `for( int i = 0 ; i < str.length() ; i ++ ){`

Which is better?

A) `for (int i = 0; i < args.length; i++)`

B)
```
for (int i = 0;
        i < args.length;
        i++)
```

C) Both are fine

D) Both are lacking

Which is better?

A) `if (game[i][index] != c)`

B) `if ( game[ i ][ index ] != c )`

C) Both are fine

D) Both are lacking

Which is better?

A) `char [][] game = new char[3][3];`

B) `char [][] game = new char[ 3 ][ 3 ];`

C) `Both are fine`

D) `Both are lacking`

Hmmm…

- I like spacing operands like the following:

```
    int x = a + b + c + d + 17;
```

- But in the below, I personally prefer the second option:

```
    data[i][i] = data[i - 1][i - 1];
    data[i][i] = data[i-1][i-1];

    myStudentIndex = i*3+foo(i);
    myGradeOffset = i+i*i-7;
data[i][i]=data[myStudentIndex][myGradeOff
set];
```

Are you familiar with the ternary operator?

```
if (a) {
  x = b;
} else {
  x = c;
}
```

Are you familiar with the ternary operator?

```
if (a) {
  x = b;
} else {
  x = c;
}
```

```
x = a ? b : c;
```

What is wrong with this?

```
int foo = a + b == 10  ? c : d + e;
```

Which is better?

A)
```
int parenthesis;
parenthesis = 0;
```

B)
```
int parenthesis = 0;
```

C) Both are fine

D) Both are lacking

Which is better?

A)
```
int paren = 0, eqnLength = eqn.legnth();
```

B)
```
int paren = 0;
int eqnLength = eqn.legnth();
```

C) Both are fine

D) Both are lacking

Which is better?

A)

```
if (three) {
    System.out.println("Valid: " + value);
}

else {
    System.out.println("Invalid");
}
```

B)

```
if (three) {
    System.out.println("Valid: " + value);
} else {
    System.out.println("Invalid");
}
```

C) Both are fine D) Both are lacking

Which is better?

A)

```
if (three) {
    System.out.println("Valid");
} else {
    System.out.println("Invalid");
}
```

B)

```
if (three)
    System.out.println("Valid");
else
    System.out.println("Invalid");
```

C) Both are fine

D) Both are lacking

Which is best?

A) `if (prev_type==type&&type!=1&&type!=2) {`

B) `if (prev_type == type && type != 1 && type != 2) {`

C) `if ((prev_type == type) && (type != 1) && (type != 2)) {`

D) All are fine

E) All are lacking

Come teach a class for

CS @ ILLINOIS Sail

April 7th, 2018!

Sail is a one-day event for high school students to learn about computer science and experience a day at the University of Illinois. As part of this experience, they attend classes taught by current students on a broad range of topics.

You can teach about any interesting topic such as:

- Cybersecurity Ninja Training
- Creating Your Own Personal Website
- Hackathons 101: Everything You Need To Know
- Or anything else!

Sign up at

sail.cs.illinois.edu/teacher

SAIL

Come teach a class for

CS @ ILLINOIS Sail

April 7th, 2018!

Sail is a one-day event for high school students to learn about computer science and experience a day at the University of Illinois. As part of this experience, they attend classes taught by current students on a broad range of topics.

You can teach about any interesting topic such as:

- Cybersecurity Ninja Training
- Creating Your Own Personal Website
- Hackathons 101: Everything You Need To Know
- Or anything else!

Sign up at sail.cs.illinois.edu/teacher

SAIL