This is what we made previously
This is what we want to make
What’s the big deal?
There are 3 triangles
There are 3 triangles

Vertex count: 3
There are 3 triangles

Vertex count: 3+3
There are 3 triangles

Vertex count: $3 + 3 + 3 = 9$
Let’s count again…

“Real” vertex count = 6
These are used more than once!
And, practically speaking...

Vertex count: 3+3+3+3 = 12

vs.

6 “real” vertices
WHY THIS IS A BIG DEAL

- Inefficient
  - Takes up more memory
    - Each vertex can be 48 bytes
      - Position: 3*4 = 12 bytes
      - Color: 4*4 = 16 bytes
      - Normal: 3*4 = 12 bytes
      - Texture coordinate: 2*4 = 8 bytes
  - Takes up more processing!
    - Must translate/rotate/scale/skew vertices
    - Redundant computations!
Enter the Index Buffer
Enter the Index Buffer
Specify Triangle 1

\{0, 1, 2\}
Specify Triangle 2

{0, 1, 2, 1, 3, 4}
Specify Triangle 3

0 1 2
3 4 5

{0, 1, 2, 1, 3, 4, 2, 4, 5}
And, practically speaking...

Index buffer!

{0, 1, 2, 1, 3, 4, 2, 4, 5, 1, 4, 2}
Index Buffer size

\{0, 1, 2, 1, 3, 4, 2, 4, 5, 1, 4, 2\} = 48 bytes
COMPARISON

• Without Index Buffers
  • Each vertex takes up 48 bytes
  • Each triangle has 3 vertices
  • 4 triangles have 12 vertices
  • Total size = 4 triangles * 3 vertices * 48 bytes each = 576 bytes

• With Index Buffers
  • Each vertex takes up 48 bytes
  • Have only 6 of them
  • Index buffer is 48 bytes
  • Total size = 6 vertices * 48 bytes each + 48 bytes = 336 bytes

• In this case, ½ the number of vertices to process!
This is what we *wanted* to make
This is what we made
Why?

Only 1 color per vertex
RENDERING OPTIONS

• We typically create a single batch of vertices to draw
  • We generally set up all “features” before drawing anything
  • State machine mentality
• We always have vertices, but we can render them in 7 different ways
  • GL_POINTS
  • GL_LINES
  • GL_LINE_STRIP
  • GL_LINE_LOOP
  • GL_TRIANGLES**
  • GL_TRIANGLE_STRIP
  • GL_TRIANGLE_FAN
RENDERING OPTIONS
(GL_POINTS)

- Can change the point sizes, but not important right now
- Points are always square unless anti-aliased
**RENDERING OPTIONS**

*(GL_LINES)*

- Connects in **pairs** (line segments), so should have an **even** number of points
- Change line width with `glLineWidth (GLfloat width);`
RENDERING OPTIONS
(GL_LINE_STRIP)

• In a connect-the-dots fashion, draw from one vertex to another
• How would you do this with GL_LINES?
RENDERING OPTIONS
(GL_LINE_LOOP)

- Closes the loop
- Is typically what you would use for outlines/tracing
RENDERING OPTIONS

(OTHERS)

• Some of these are allowed...
RENDERING OPTIONS
(OTHERS)

- Some of these aren’t…
- NO QUADS! Triangles only…