

## Double Buffering

- To improve visual quality
  - Video cards provide two frame buffers
  - Easily swappable
  - Swap is delayed to next refresh cycle
- Simple to use
  - Draw in the non-displayed (back) buffer
  - Swap the buffers

1

---

---

---


---

---

---

---

---



## Double Buffering Under QGLWidget [<http://doc.trolltech.com/qglwidget.html>]

- Enabled by default, like depth buffer
  - Buffers swapped when paintGL finishes
- Is a mode that must be chosen when rendering context is created
- If not wanted
  - Pass QGLFormat argument to QGLWidget ctor
    - In client class
      - `QGLFormat qglf;`
      - `qglf.setDoubleBuffer(false);`
      - `myQGLWidget wid(qglf);`
    - In myQGLWidget ctor initializer list
      - `QGLWidget(myQGLReference);`
  - Swap the buffers in paintGL
    - `swapBuffers();`

2

---

---

---


---

---

---

---

---



## Coding Animation

- It is **not** acceptable for redraw to loop
  - Loop for all frames
    - Display frame
    - Swap buffers
    - Delay as needed
  - This will "hog" the processor
    - Event routines should be "quick"

3

---

---

---

---

---

---

---

---

### Animations: GLUT Method 1 – idle function

- **Register** an idle function to run whenever there is not a higher priority task.
  - Idle function then posts a redisplay event (puts a paintGL in the event queue, in Qt terms)
  - Display function uses wall clock time,
    - or moves to next frame in sequence
  - Drawbacks: no timing control, jerkiness, variable framerate

4

---

---

---

---

---

---

---

---

### Animation Timing

- We really want a periodic event
  - Trigger a redraw at regular intervals (frame rate)
- Good idea
  - Provide a mechanism to start, stop, and pause animation
  - i.e., Turn the periodic event off

5

---

---

---

---

---

---

---

---

### Animations: GLUT Method 2: Timer Callbacks

- Create an animation function `void animate(int)`
  1. Update simulation time
  2. Queue a new frame
  3. Schedule the next callback
- Schedule the first callback before entering the GLUT main loop
  - `glutTimerFunc(msecs, animate, arg);`
- The display function still does the drawing.

6

---

---

---

---

---

---

---

---

## Animations: Qt: Timer Callbacks

- Create an animation function
 

```
void timer()
1. Update simulation time (simStart.elapsed())
2. Draw a new frame (updateGL)
3. Schedule the next callback
   1. QTimer::singleShot(msecs, this,
      SLOT(timer()));
```
- Schedule the first callback – override polish()
  - `QGLWidget::polish(); // do not disable base behavior`
  - `singleShot` as above
- The display function still does the drawing.

7

---

---

---

---

---

---

---

---

## Timer Event Concerns

- Timer events are queued
  - They are not processed immediately
  - Only minimum interval is specified
- O/S restrictions may limit the min. time
  - 1 ms looks possible, but it may not be
- Keep in mind 24 fps is very good
  - Video refresh rate is a factor too

8

---

---

---

---

---

---

---

---

## Hierarchical Representations (1)

- Useful articulation and compound objects
- Conform well to C++ object ideas
- Base class Object
- Class data members of Object
  - Orientation transform (from OpenGL)
  - List of children (Object\*)
  - Passed to constructor or hard coded

9

---

---

---

---

---

---

---

---

## Hierarchical Representations (2)

- Main drawing code (in base class!)
  1. Push state information
  2. Transform
  3. Draw yourself (use a separate virtual)
  4. Ask children to draw themselves
  5. Pop state information

10

---

---

---

---

---

---

---

---

## Miscellaneous: Getting OpenGL Implementation Info.

```
printf("OpenGL Information...\n");  
printf("  Vendor: %s\n", glGetString(GL_VENDOR));  
printf("  Renderer: %s\n", glGetString(GL_RENDERER));  
printf("  Version: %s\n", glGetString(GL_VERSION));
```

OpenGL Information...  
Vendor: ATI Technologies Inc.  
Renderer: Radeon 7200 LE DDR x86  
Version: 1.3.3064 Win2000 Release

11

---

---

---

---

---

---

---

---