Viewport

- The rectangular region in the screen for displaying the graphical objects defined in the world window

- Defined in the screen coordinate system

```c
glViewport(int left, int bottom, int (right-left), int (top-bottom));
```

call this function before drawing (calling glBegin() and glEnd() )
To draw in 2D

- Two tasks need to be done
  - Define a rectangular **world window**
    (call an OpenGL function)
  - Define a viewport (call an OpenGL function)
  - Perform **window to viewport mapping**
    (OpenGL internals will do this for you)
A simple example

DrawQuad()
{
    glViewport(0,0,300,200);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-1,1,-1,1);
    glBegin(GL_QUADS);
    glColor3f(1,1,0);
    glVertex2f(-0.5,-0.5);
    glVertex2f(+0.5,-0.5);
    glVertex2f(+0.5,+0.5);
    glVertex2f(-0.5,+0.5);
    glEnd();
}
Window to viewport mapping

- The objects in the world window will then be drawn onto the viewport.
Window to viewport mapping

How to calculate \((sx, sy)\) from \((x, y)\)?
you don’t need to do it by yourself. OpenGL will do it for you

You just need to define the viewport (with `glViewport()`), and a world window (with `gluOrtho2D()`)

Define a world window

```c
glMatrixMode(GL_PROJECTION);
gluOrtho2D(Left, Right, Bottom, Top);
```

```c
glmatrixmode(GL_PROJECTION);
glLoadIdentity();
```
Window to viewport mapping

- Things that are given:
  - The world window \((W_L, W_R, W_B, W_T)\)
  - The viewport \((V_L, V_R, V_B, V_T)\)
  - A point \((x, y)\) in the world coordinate system
- Calculate the corresponding point \((sx, sy)\) in the screen coordinate system
Window to viewport mapping

- Basic principle: the mapping should be proportional

\[
\frac{x - W_L}{W_R - W_L} = \frac{sx - V_L}{V_R - V_L}
\]
\[
\frac{y - W_B}{W_T - W_B} = \frac{sy - V_B}{V_T - V_B}
\]
Window to viewport mapping

\[
\frac{x - W_L}{W_R - W_L} = \frac{sx - V_L}{V_R - V_L}
\]

\[
\frac{y - W_B}{W_T - W_B} = \frac{sy - V_B}{V_T - V_B}
\]

\[
sx = x \frac{V_R - V_L}{W_R - W_L} - W_L \frac{V_R - V_L}{W_R - W_L} + V_L
\]

\[
sy = y \frac{V_T - V_B}{W_T - W_B} - W_B \frac{V_T - V_B}{W_T - W_B} + V_B
\]
Non-distorted viewport setup

- Distortion happens when …
- World window and display window have different aspect ratios
- Aspect ratio?
- \( R = \frac{W}{H} \)
Compare aspect ratios

World window
Aspect Ratio = R

Display window
Aspect Ratio = W / H

R > W / H
Match aspect ratios

World window

Aspect Ratio = R

Display window

Aspect Ratio = W / H

R > W / H
Match aspect ratios

World window

Aspect Ratio = R

R > W / H

Display window

Aspect Ratio = W / H

glViewport(0, 0, W, W/R)
Compare aspect ratios

World window
AspectRatio = R

Display window
AspectRatio = W / H

R < W / H
Match aspect ratios

World window

Aspect Ratio = R

Display window

Aspect Ratio = W / H

R < W / H
Match aspect ratios

World window

Aspect Ratio = R

Display window

Aspect Ratio = \( \frac{W}{H} \)

\( R < \frac{W}{H} \)

\text{glViewport}(0, 0, H*R, H)
When to call glViewport()?

Two places:

- Initialization
  - Default: same as the window size
- When the user resizes the display window
void resize () – a function provided by you. It will be called when the window changes size.
Resize (reshape) window

Void resize(int W, int H)
{
    glViewport(0, 0, W, H);
}

This is done by default in GLUT

You can use the call to make sure the aspect ratio is fixed that we just discussed.
How big is the quad?

DrawQuad()
{
    glViewport(0,0,300,200);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-1,1,-1,1);
    glBegin(GL_QUADS);
    glColor3f(1,1,0);
    glVertex2f(-0.5,-0.5);
    glVertex2f(+0.5,-0.5);
    glVertex2f(+0.5,+0.5);
    glVertex2f(-0.5,+0.5);
    glEnd();
}