02561 COMPUTER GRAPHICS   DTU COMPUTE

Worksheet 2: Input devices and interaction

Reading	Angel: Sections 3.2-3.11
Purpose	The purpose of this set of exercises is to start interacting with the graphics elements that we draw using input devices. You will make a small 2D drawing program where primitive shapes of different colors can be added using the mouse. Although it is possible to draw 2D shapes using a 2D canvas context, you will do it with a WebGL context as it is really not much harder, and WebGL enables later extension to full 3D. The following screenshot is an example of what the final web application could look like. Your version doesn't have to be exactly like it. The important point is that it has the features we want.
Features	In your drawing program, you should be able to: • Use the mouse to draw different primitive shapes (point, triangle, and circle). • Select the color of a shape while adding it. • Draw points at the first two vertices of a triangle and replace the points by a triangle for every third vertex. • Draw a point in the circle center and replace the point by a circle for every second mouse click (the second point sets the circle radius). • Use color interpolation between triangle vertices and radially in circles. • Clear the canvas to the selected color (delete all shapes). In the following parts, we outline a suggested work flow that you can follow.

02561 COMPUTER GRAPHICS   DTU COMPUTE

Worksheet 2: Input devices and interaction

Part 1	• Start from your solution to Part 2 of Worksheet 1: A web application that clears the canvas and then draws three points. [Angel 2.8] • Attach an event handler to the mouse click event and draw points on the canvas where the mouse was clicked. [Angel 3.7] • Points are placed at the center of the mouse cursor, but we want to use the upper left corner of the mouse cursor as the point that we are clicking. Get the bounding rectangle of the mouse cursor using `event.target.getBoundingClientRect()` and correct the mouse position using the `left` and `top` coordinates of this rectangle.
Part 2	• Add a button that clears the canvas. [Angel 3.6.2] • Add a color selection menu where you can choose the color to be used when clearing the canvas. [Angel 2.5, 3.6.3, 3.10] • Use the same color selection menu to set the color of the drawn point when the mouse is clicked. This requires updating your shaders to work with colors as in Part 3 of Worksheet 1.
Part 3	• We would now like to have two different drawing modes. One where we draw points as before and one where we interactively build a triangle by placing three points. Add a button for each drawing mode. [Angel 3.6.2] • (Hint) The following is closely related to the textbook CAD example [Angel 3.10], where a polygon is built interactively. The main difference is that we have two drawing modes. • Create an empty array of point indices and an empty array of triangle indices. When drawing in point mode, we add vertex buffer indices of the points to the array of point indices. We also do this in triangle mode for the first two vertices, but with the third mouse click, we add the index of the first triangle vertex to the array of triangle indices and remove the indices of the first two vertices from the array of point indices. Use `push` and `pop` when working with the arrays. • When rendering, draw the vertices in the array of point indices as points and draw the vertices in the array of triangle indices as triangles. Try to call `gl.drawArrays` as few times as possible.
Part 4	• Add a button for a circle drawing mode. [Angel 3.6.2] • Create an array for circle indices. When drawing in circle mode, add a point on the first click. On the second click add vertices for the circumference as in Part 5 of Worksheet 1 using the position of the second click to set the circle radius. As with the triangle, add the index of the center vertex to the array of circle indices and remove this index from the array of point indices. • Draw each circle as a triangle fan when rendering.