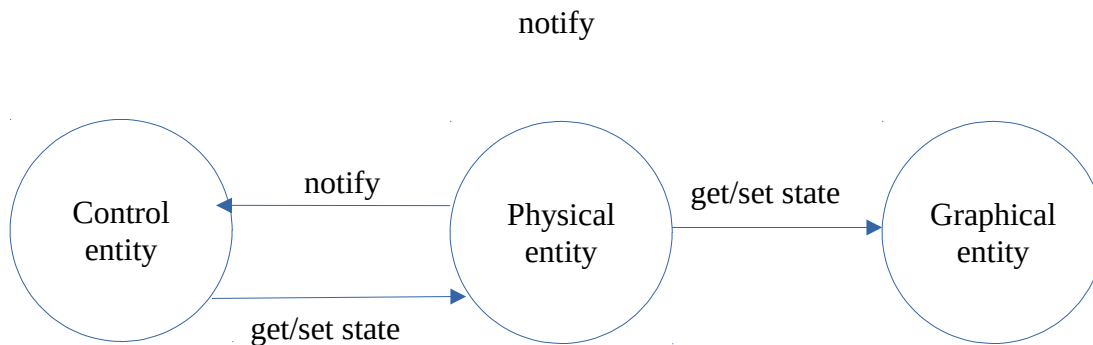


WebGL Sandbox/Game Engine

Common terms

1. The system consists of minimum 3 **layers**: **graphical**, **physical** and **control**;
2. The function of **graphical layer** is about drawing and animating pictures: **graphical primitives** such as circle, rectangle, polygon etc. with color, texture etc. (to be described later);
3. The function of **physical layer** is about calculating physical state of objects (form, size, position, rotation) that affects appearance of **graphical entity(ies)**;
4. The function of **control layer** is about maintaining state that's defined by game programmer/designer, and this can affect both **graphical** and **physical** layers;
5. Flow of data between those layers is one-directional, MS. State of object can be changed either by this object, or by its controller. Controllables can notify their controllers about changing their state:



get/set state

Graphical layer

1. **WebGL program** can be divided in two phases: **initialization** and **drawing frame**;

2. **Initialization phase** requires compiling **shaders (vertex and fragment)**, retrieving **attribute/uniform locations**, buffer initialization for each shape;

// TODO: texture initialization;

NB: **vertex shader** is called per vertex and its goal is to make clip space (i.e. boundary polygons for filling pixels), and **fragment shader** is called per pixel after applying vertex shader and its goal is to draw pixels in grid;

3. **Drawing frame phase** requires following steps:

3.1. Clear viewport;

3.2. Bind **GLSL program** to **WebGL Context** that is associated with current shape;

3.3. For each attribute of the program enable vertex attribute array, bind buffer, setup vertex attribute pointer, initialize bufferData;

3.4. For each uniform of the program set uniform with its location;

3.5. Call drawArrays;

3.6. Repeat process from 3.2. if there's next shape;

3.7. For each shape modify the state of attributes and uniforms, specified by **client program** (physics engine, user-defined controllers);

3.8. Repeat 3.1.

4. **shape** is an geometric object, that's initialized with form (a method to draw a shape, bound to shaders or native program), position point, rotation angle, scale (bounding rectangle with respect to viewport bounds), flip state (with respect to horizontal and vertical axes), color/gradient/texture (TODO). They can be modified during **drawing frame phase**. These parameters define attribute and uniform values of shape;

5. **viewport (canvas)** is main object of graphical layer, that holds all shapes in scene and has particular size (to which dimensions of shapes are normalized);