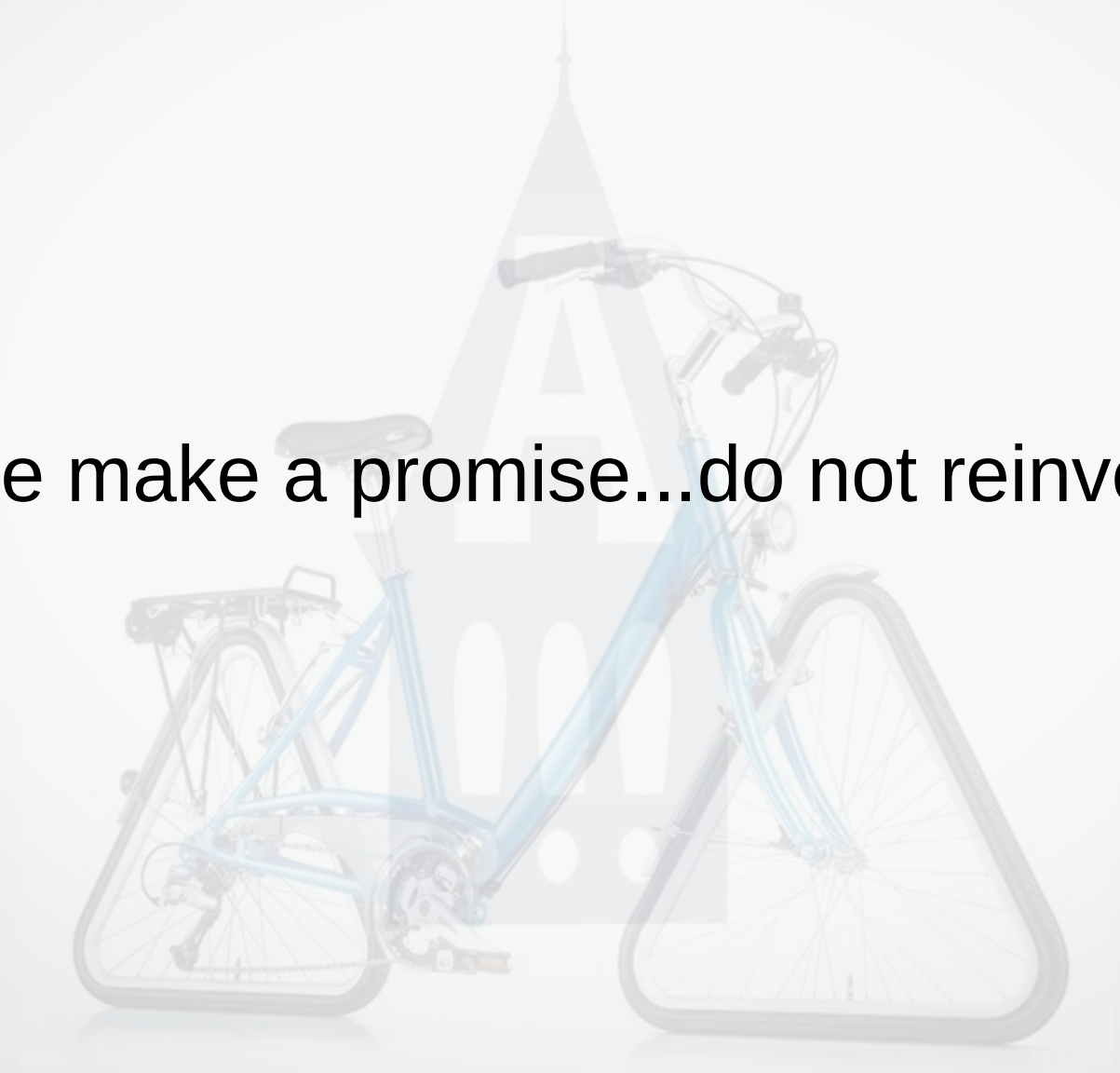
The Game Loop

Please make a promise...

The Game Loop

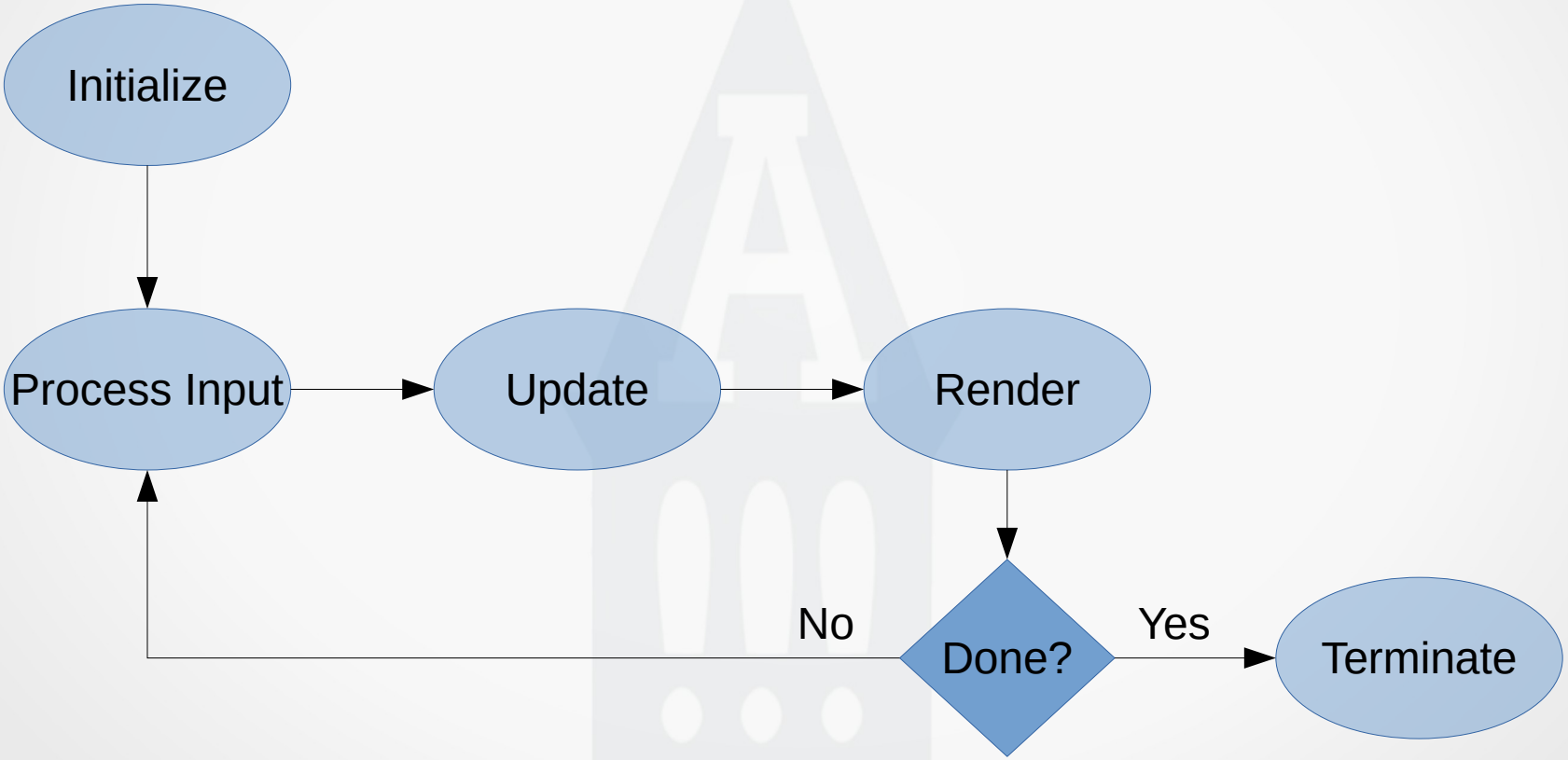
Please make a promise...do not reinvent this!



The Game Loop - Step-by-Step

1. Initialize Game
   - Load graphics, models, animations, etc
   - Take initial time-stamp; call it previous time-stamp
2. Process Input
3. Update Game Logic
   - Take current time-stamp; compute elapsed time
   - Update based on elapsed time
4. Render Game State
5. Move current time-stamp to previous time-stamp
6. (if fixed rate) use spin-lock to wait until frame-time expires
7. If done, move to Step 8, otherwise return to Step 2
8. Termination

The Game Loop - Visualized



The Game Loop - JavaScript

```
function gameLoop(timeStamp) {
  elapsedTime = timeStamp - prevTime;
  prevTime = timeStamp;
  processInput(elapsedTime);
  update(elapsedTime);
  render();

  requestAnimationFrame(gameLoop);
}
```

The Game Loop - JavaScript

```
function processInput(elapsedTime) {
  keyboard.update(elapsedTime);
}
```

```
function gameLoop(timeStamp) {
  elapsedTime = timeStamp - prevTime;
  prevTime = timeStamp;
  processInput(elapsedTime);
  update(elapsedTime);
  render();

  requestAnimationFrame(gameLoop);
}
```

The Game Loop - JavaScript

```
function processInput(elapsedTime) {
  keyboard.update(elapsedTime);
}
```

```
function update(elapsedTime) {
  gameModel.update(elapsedTime);
}
```

```
function gameLoop(timeStamp) {
  elapsedTime = timeStamp - prevTime;
  prevTime = timeStamp;
  processInput(elapsedTime);
  update(elapsedTime);
  render();

  requestAnimationFrame(gameLoop);
}
```

The Game Loop - JavaScript

```
function processInput(elapsedTime) {
  keyboard.update(elapsedTime);
}
```

```
function update(elapsedTime) {
  gameModel.update(elapsedTime);
}
```

```
function render() {
  gameModel.render();
}
```

```
function gameLoop(timeStamp) {
  elapsedTime = timeStamp - prevTime;
  prevTime = timeStamp;
  processInput(elapsedTime);
  update(elapsedTime);
  render();

  requestAnimationFrame(gameLoop);
}
```

Timing

- Frame Rate: Measured in Hz; frames per second (fps)
- Frame Time: Amount of time within a frame Δt
  - The entire game simulation and rendering must take place in this amount of time
  - If 30 fps, each frame has 33.33ms for everything!

Which Time?

- Wall-Clock Time: real-world elapsed time
- Simulation Time: How much game-play time has passed

- These two might be the same, but don’t have to be
  - Consider Bullet-Time.
    - Game frame-rate stays the same
    - Game simulation slows down
    - Player continues to react in real-time (MOL)

Moving Objects

- Bad Idea: Move some number of (virtual) meters per frame
- Not as Bad Idea: Move some number of (virtual) meters based on running frame-rate average
  - $x_2 = x_1 + v\ \Delta t_{ave}$
- Best Idea
  - $x_2 = x_1 + v\ \Delta t_{frame}$
- Fixed Frame Rate
  - $x_2 = x_1 + v\ 1/fps$