

# The Bleeding Edge of Web Technology

Eric Shepherd



# Why Am I Talking About This?

- What I do at Mozilla.
- The goal: a plugin-free Web.
- The future is **awesome**.



# New Technologies

- HTML5
- CSS3
- Server-Sent Events
- WebSockets
- WebGL



# HTML5

- `<progress>` element.
- `<input>` validation.
- So much awesomeness I'm not going to talk about it, because everyone else already has.



# CSS3 Animations

- Animate content without scripting.
- Most CSS styling can be animated, such as:
  - Position, size, color, style, backgrounds, opacity, and so forth.
- [https://developer.mozilla.org/en/CSS/CSS\\_animations](https://developer.mozilla.org/en/CSS/CSS_animations)



# A Simple Example

```
<style type="text/css">
  h1 {
    -moz-animation-duration: 3s;
    -moz-animation-name: slidein;
  }

  @-moz-keyframes slidein {
    from {
      margin-left: 100%;
      width: 300%
    }

    to {
      margin-left: 0%;
      width: 100%;
    }
  }
</style>
```



# CSS Transitions

- Lets you change the style of an element over time instead of instantly.
- Great for visual effects in user interfaces, for example.
- [https://developer.mozilla.org/en/CSS/CSS\\_transitions](https://developer.mozilla.org/en/CSS/CSS_transitions)



# Server-Sent Events

- Let servers send messages to browsers without polling.
- So new, I don't even have a demo, but...



# Using SSEs

```
var source = new EventSource('updates.cgi');
source.onmessage = function (event) {
    alert(event.data);
};
```

- Server sends messages with the text/event-stream MIME type.
- See <http://dev.w3.org/html5/eventsource/>



# WebSockets

- True bidirectional, real-time communication between browser and server.
- [https://developer.mozilla.org/en/  
WebSockets](https://developer.mozilla.org/en/WebSockets)
- Let's look at some code and an example!



# WebGL

- Use OpenGL ES from web content.
- Works in Firefox and Google Chrome.



# Initializing WebGL

```
function start() {  
  var canvas = document.getElementById("glcanvas");  
  
  initWebGL(canvas);    // Initialize the GL context  
  
  // Only continue if WebGL is available and working  
  
  if (gl) {  
    gl.clearColor(0.0, 0.0, 0.0, 1.0);  
    gl.clearDepth(1.0);  
    gl.enable(gl.DEPTH_TEST);  
    gl.depthFunc(gl.LEQUAL);  
    gl.clear(gl.COLOR_BUFFER_BIT|gl.DEPTH_BUFFER_BIT);  
  }  
}
```



# Creating a GL Context

```
function initWebGL(canvas) {  
    gl = null;  
  
    try {  
        gl = canvas.getContext("experimental-webgl");  
    }  
    catch(e) {  
    }  
  
    // If we don't have a GL context, give up now  
  
    if (!gl) {  
        alert("Unable to initialize WebGL. Your browser may not support it.");  
    }  
}
```



# Demo Hawtness!

- Let's see some fun demos!