Sounds, Music and Sidescrolling

Game Programming Foundations

Last modified 18/02/15 by Sam Cartwright

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Topics

- Sound in HTML5
- Howler
- Adding BGM / SFX
- Implementing Side-Scrolling

Sound in HTML5

- HTML5 uses the <audio> element
- Manipulate <audio> using methods, properties and events
- Basic methods load(), play() and pause()
- Many problems
  - Playing multiple sounds
  - Mobile browser support
  - Supported audio formats

<audio> Supported Coding Formats

Browser	OS	Ogg Vorbis	WAV PCM	MP3	AAC	WebM Vorbis	Ogg Opus	WebM Opus
Chrome	All	9	Yes	Yes	Yes	Yes	25 (since v31 in Win)	Yes
IE	Win	No	No	9	9	No	No	No
Firefox	All	3.5	3.5	Win (21.0) Linux (24.0) OS X (26.0)	Win (21.0) Linux (24.0) OS X (34.0)	4.0	15.0	28.0
Opera	All	10.50	11.00	14	14	10.60	14	Yes
Safari	OS X	Manual install	3.1	3.1	3.1	No	No	No

Sound in HTML5

Music

```
var music = new Audio("background.ogg");
music.loop = true;
music.play();
```

Sound Effects

```
var isSFxPlaying = false;
var sfx = new Audio("fireEffect.ogg");
sfx.onended = function() {
    isSfxPlaying = false;
};
```

Web Audio API - Challenges

- A bit more complicated to use
  - Create one or more sound sources
  - Connect them to a sound destination
    - Using the AudioContext instance
  - Intermediate AudioNodes act as processing modules for the audio signal

- Needs a web server

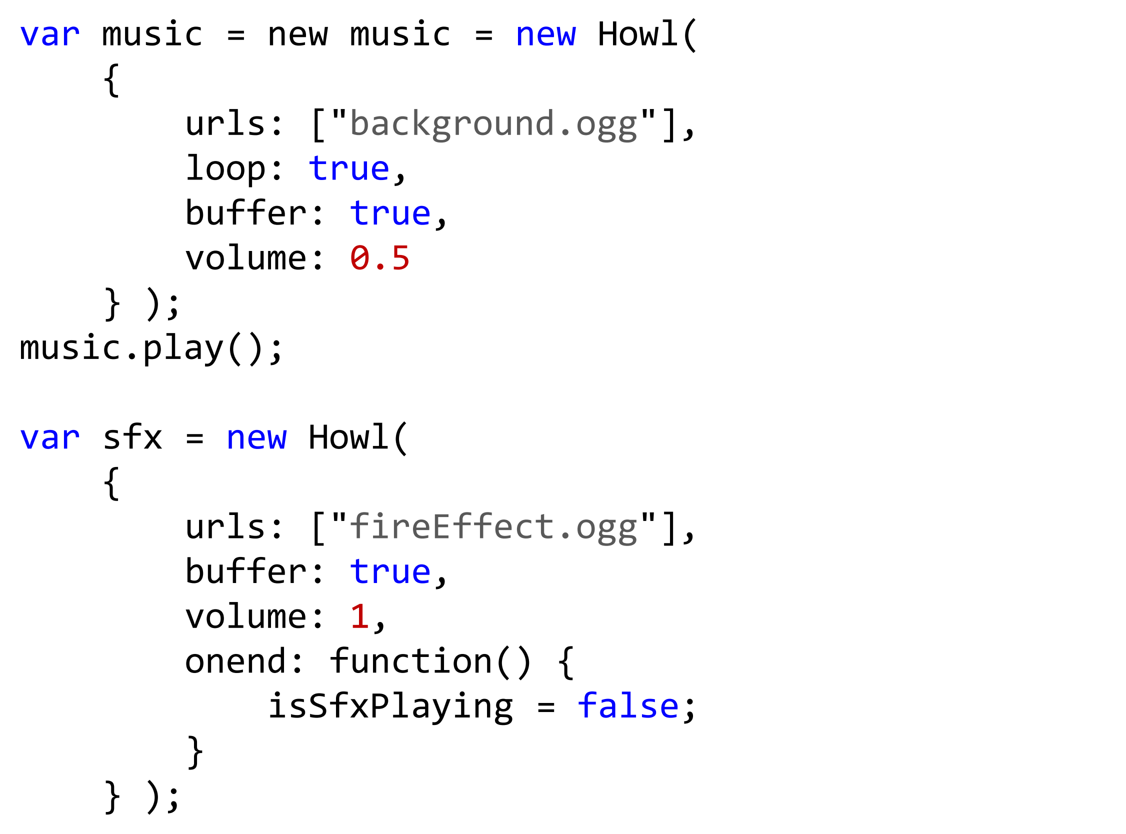
Web Audio API - Advantages

- Overcome limitations with simultaneous playback
- Precisely schedule playback
- Build complex audio graphs to:
  - Cross-fade between sounds
  - Apply filter effects to sounds

Howler

- One of many custom APIs to simplify sound
- Defaults to Web Audio API, falls back to HTML5 Audio
- Per sound and global volume control
- Fade in/out
- Easy audio sprite support
- http://goldfirestudios.com/blog/104/howler.js-Modern-Web-Audio-Javascript-Library

Howler - Adding BGM / SFX



```
var music = new music = new Howl(
    {
        urls: ["background.ogg"],
        loop: true,
        buffer: true,
        volume: 0.5
    } );
music.play();

var sfx = new Howl(
    {
        urls: ["fireEffect.ogg"],
        buffer: true,
        volume: 1,
        onend: function() {
            isSfxPlaying = false;
        }
    } );
```

Howler - Tips

- Always set buffer to true
  - Ensures use of HTML5 audio
  - Because we can’t use a live web server

- Optimize by using a Sound Sprite (all sounds in 1 file)

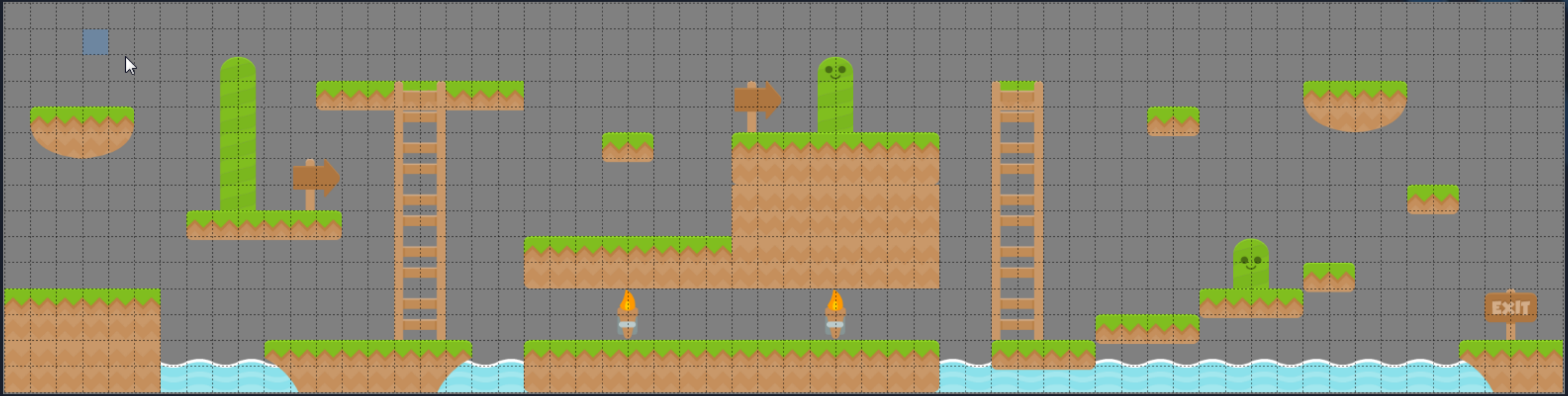
- Use events callbacks to detect audio events
  - onend, onload, onloaderror, onpause, onplay

- Refer to https://github.com/goldfire/howler.js for more samples

Sidescrolling

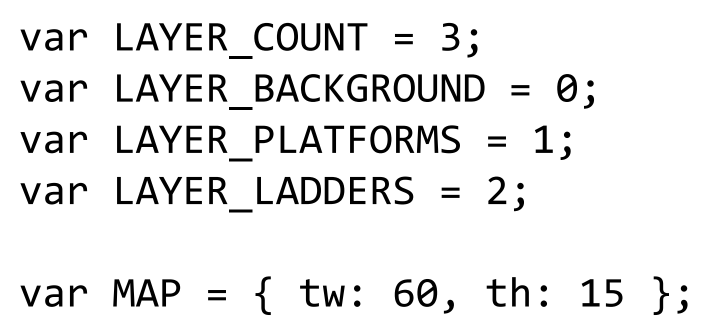
- Reuse code from previous samples

- Update game level 60x20 tiles
  - 1 new layer for background images (water, signs, etc)



Sidescrolling

- Update constant values



```
var LAYER_COUNT = 3;
var LAYER_BACKGROUND = 0;
var LAYER_PLATFORMS = 1;
var LAYER_LADDERS = 2;

var MAP = { tw: 60, th: 15 };
```

- Collision map initialization remains the same
  - Optionally optimize by removing background layer collisions

Sidescrolling

- Update drawMap function

- Always keep player in centre of screen
  - Except where map can’t scroll anymore
  - Means x position of map based on player’s position

- Calculate the start/end X tile coord based on player pos

- Calculate X offset for tiles based on player pos

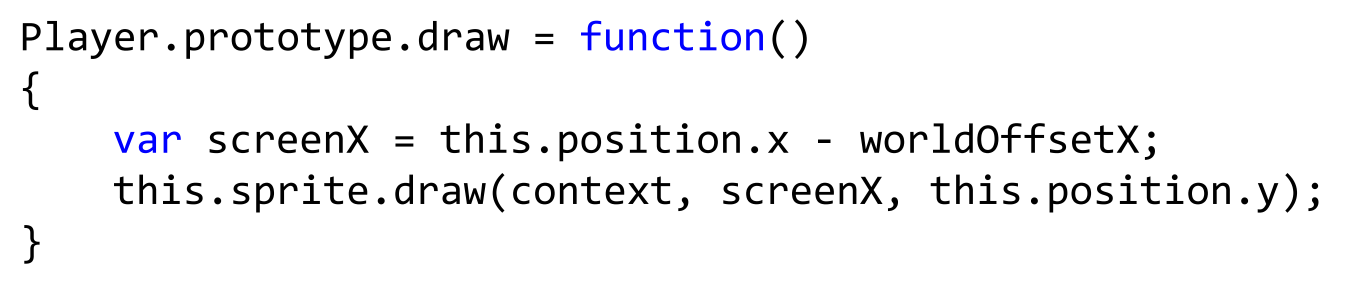
- Draw only the tiles that should be on-screen

```
var worldOffsetX = 0;
function drawMap()
{
    var startX = -1;
    var maxTiles = (SCREEN_WIDTH / TILE) + 2;
    var tileX = pixelToTile(player.position.x);
    var offsetX = TILE + (player.position.x - tileToPixel(tileX));

    startX = tileX - ((SCREEN_WIDTH / TILE) / 2);

    if(startX < -1)  {
        startX = -1;
        offsetX = TILE;
    }
    if(startX > MAP.tw - maxTiles)   {
        startX = MAP.tw - maxTiles + 1;
        offsetX = TILE;
    }

    worldOffsetX = startX * TILE + offsetX;

    for(var layerIdx=0; layerIdx<LAYER_COUNT; layerIdx++)
    {
        for(var y=0; y<level1.layers[layerIdx].height; y++)
        {
            var idx = y * level1.layers[layerIdx].width + startX;
            for(var x = startX; x < startX+maxTiles; x++)
            {
                if(level1.layers[layerIdx].data[idx] != 0)
                {
// the tiles in the Tiled map are base 1 (meaning a value of 0
// means no tile), so subtract one from the tileset id to get
// the correct tile
                    var tileIndex = level1.layers[layerIdx].data[idx] - 1;
                    var sx = TILESET_PADDING +
                                  (tileIndex % TILESET_COUNT_X) *
                                  (TILESET_TILE + TILESET_SPACING);
                    var sy = TILESET_PADDING +
                                  (Math.floor(tileIndex / TILESET_COUNT_Y)) *
                                  (TILESET_TILE + TILESET_SPACING);

                    context.drawImage( tileset,
                                           sx,  sy, TILESET_TILE, TILESET_TILE,
                                           (x - startX) * TILE - offsetX,  (y-1)*TILE,
                                           TILESET_TILE, TILESET_TILE);
                }
                idx++;
            }
        }
    }
}
```

Sidescrolling

- worldOffsetX tells us the current scrolling of the world

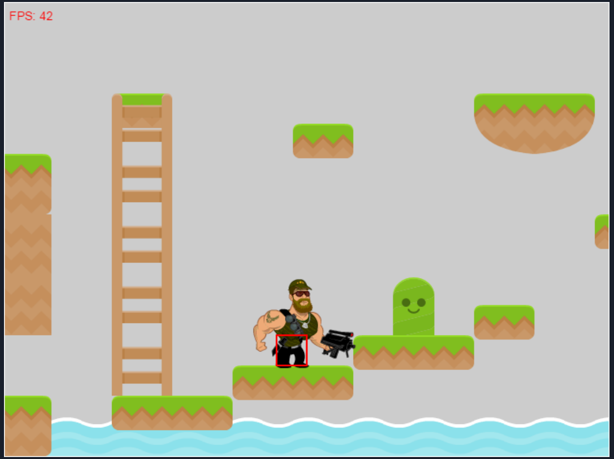
- To keep player in centre of screen:
  - sceenX = position.x – worldOffsetX



```
Player.prototype.draw = function()
{
    var screenX = this.position.x - worldOffsetX;
    this.sprite.draw(context, screenX, this.position.y);
}
```

Sidescrolling

- Collision detection works without modification
- All other code shouldn’t need modification
- Y-axis scrolling will work the same way



Summary

- Sound in HTML5 can be challenging
- Using a Third Party API can simplify development
- Howler.js is a small, efficient library for sound
  - Using either HTML5 audio or Web Audio API
- Sidescrolling requires calculating which tiles to draw based on player’s position
  - Player stays in centre of screen where possible

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Questions?

References

- www.w3schools.com. 2014. *HTML Audio/Video DOM Reference*. [ONLINE] Available at:http://www.w3schools.com/tags/ref_av_dom.asp. [Accessed 24 December 14].

- James Simpson. 2013. *howler.js - Modern Web Audio Javascript Library*. [ONLINE] Available at:http://goldfirestudios.com/blog/104/howler.js-Modern-Web-Audio-Javascript-Library. [Accessed 24 December 14].

- Boris Smus. 2013. *Getting Started with Web Audio API*. [ONLINE] Available at:http://www.html5rocks.com/en/tutorials/webaudio/intro/. [Accessed 29 December 14].