Data-independent Sequencing with the Timing Object

MMSys'16, Special Session on Media Synchronization
12. May 2016, Klagenfurt, Austria
Ingar Arntzen, Njål Borch
Norut Northern Research Institute, Tromsø, Norway
EU FP7 MediaScape

Browsers

+

Global Timing


No plugins

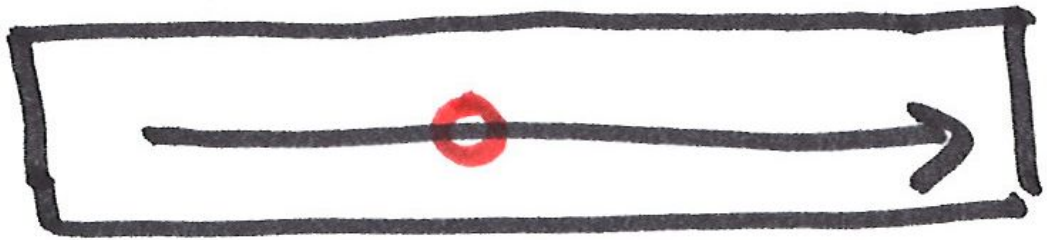
Demo: http://goo.gl/EqN40F

http://mcorp.no/pres/vegas16

Multi-device Timing Community Group

https://www.w3.org/community/webtiming/

Timing Object

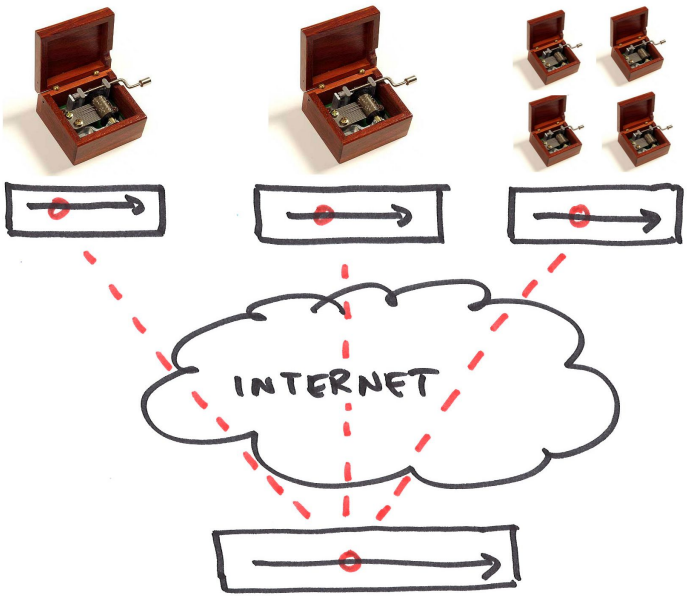


- Represents **motion** through media
  - ... playback, progress, navigation, timer, media clock.
  - Position, velocity, acceleration related to axis/timeline
- Timing & Control
  - timingObject.**update**({position: X, velocity: Y);
  - Play, pause, time-shift, fast-forward, reverse, accelerate ...

Purpose of Timing Object (1/2)

- Temporal interoperability
  - Common interface
- Shared, external timing & control
  - Independent, internal synchronization

Purpose of Timing Object (2/2)



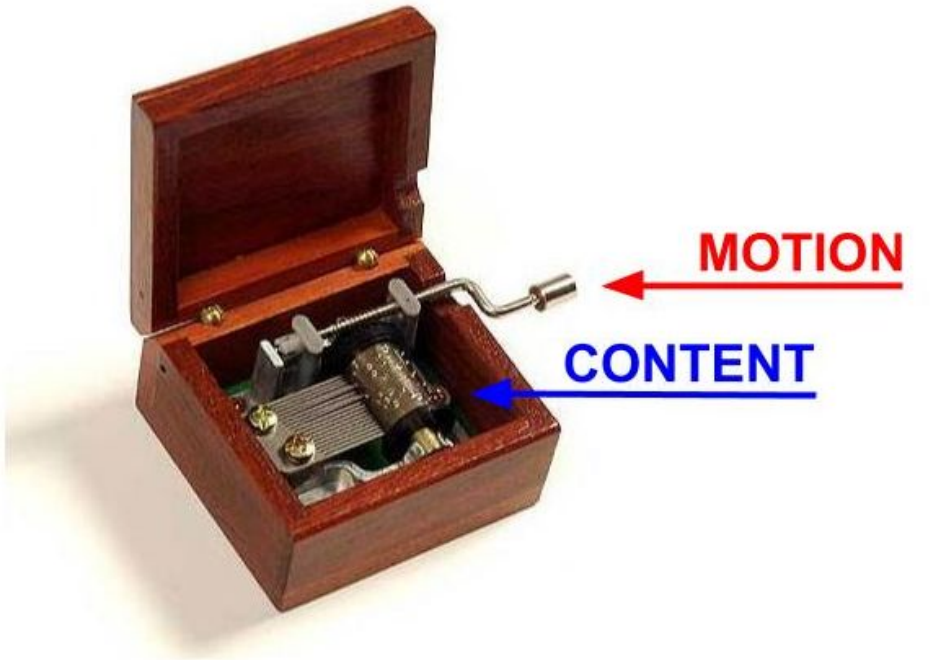
- Gateway to multi-device media!
  - Shared timing & control across Internet
  - Global timing & (remote) control
- Timing Objects
  - Proxies to online timing objects
- Separation of concern
  - Web programmers work with timing objects
  - Timing providers deal with timing
- Temporal interoperability - distributed
  - Reusability, integration, extensibility, flexibility, ...

Multi-device Timing Community Group

https://www.w3.org/community/webtiming/

- Web and TV Interest Groups
  - https://www.w3.org/2011/webtv/
- Timing Object : Standard Draft Proposal
  - http://webtiming.github.io/timingobject/
- Timingsrc : Implementation (GitHub)
  - http://webtiming.github.io/timingsrc/
  - Timing Object
  - MediaSync - synchronization of HTML5 Media Elements
  - Sequencer - synchronization of timed data
- Online Timing Provider
  - Motion Corporation http://www.motioncorporation.com

Sequencing



- Activating and deactivating media items at the correct time
- Target : Web
  - Framework or Text Track?

- **CONTENT**
  - **Data-independent sequencing**
  - **Generic programming tool**
  - **Any data any purpose**

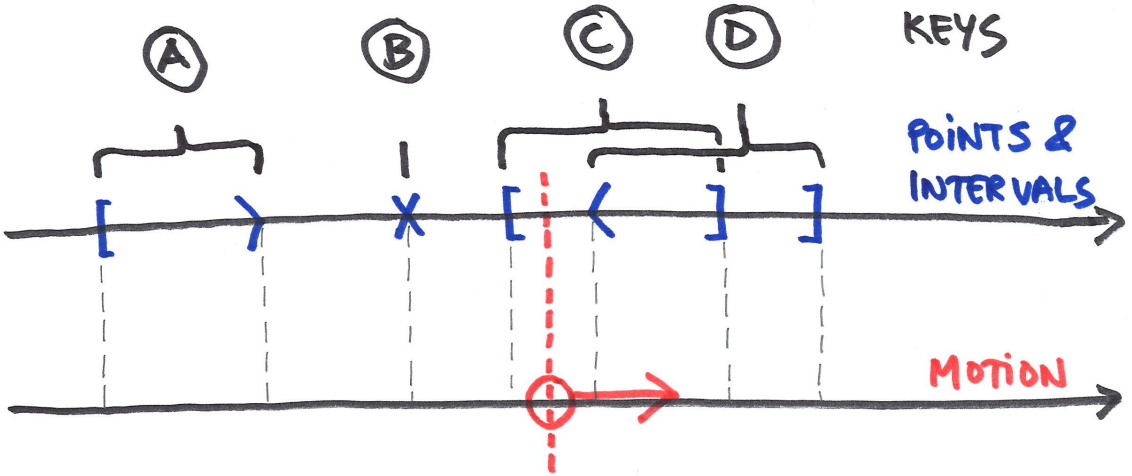
- **MOTION**
  - **Avoid dependence on HTML5 video/audio as motion**
  - **Sequencing driven by Timing Object**

Sequencer

- Generic tool for sequencing discrete media in single and multi-device timed Web applications.
  - Loading and unloading videos?
  - Collaborative viewing of anything Web?
  - Personalized ad-insertions?
  - Secondary device as a Web page?
  - Time-shifting live Web content?
  - Visualizing and replaying system logs or timed user interaction?
  - Timed prefetching of data?
  - Correct timestamping of captured media?

Details

- addCue(); removeCue(); -> (key, interval)
- Emit events -> enter, exit
- Timing Object
- Active keys
- Cue changes during playback



1. Create Sequencer

```
<!DOCTYPE html>
<html>
  <head>
    <script text="javascript" src="http://github.com/webtiming/timingsrc/lib/timingsrc.js"></script>
    <script text="javascript">
      var init = function () {
        // create timing object
        var to = new TIMINGSRC.TimingObject();
        // create sequencer
        var s = new TIMINGSRC.Sequencer(to);
      };
      if (document.readyState === "complete") init();
      else window.onload = init;
    </script>
  </head>
  <body>
  </body>
</html>
```

2. Register cues

```
// Timed data
var array = [
    { data: 'A', start: 0, end: 1 },
    { data: 'B', start: 2, end: 3 },
    { data: 'C', start: 4, end: 5 },
    { data: 'D', start: 6, end: 7 },
    { data: 'E', start: 8, end: 9 },
    { data: 'F', start: 10, end: 11 },
    { data: 'G', start: 12, end: 13 },
    { data: 'H', start: 14, end: 15 },
    { data: 'I', start: 16, end: 17 }
];

// Load timed data, use array indexes as keys into Sequencer
for (var i=0; i<array.length; i++) {
    var obj = array[i];
    s.addCue(i.toString(), new Interval(obj.start, obj.end));
}
```

3. Make a (simple) viewer

```
1  var v = document.getElementById("viewer");
2  s.on("enter", function (e) {
3    var i = parseInt(e.key);
4    v.innerHTML = array[i].data;
5  });
6  s.on("exit", function (e) {
7    v.innerHTML = "";
8  });
```

4. You’re done! Start playback

```
1  document.getElementById('playButton').onclick = function () {
2    timingObject.update({velocity:1.0});
3  };
4  document.getElementById('pauseButton').onclick = function () {
5    timingObject.update({velocity:0.0});
6  };
7  document.getElementById('resetButton').onclick = function () {
8    timingObject.update({position: 0.0});
9  };
```

Shared Motion Timing Provider

```
1  var app = MCorp.app("your_appid");
2  app.run = function () {
3    var timingProvider = app.motions["your_motion_name"];
4    var to = new TimingObject({provider:timingProvider});
5    // good to go!
6  };
7  app.init();
```

- Not limited to Web - anything IP
- And scalable too :)

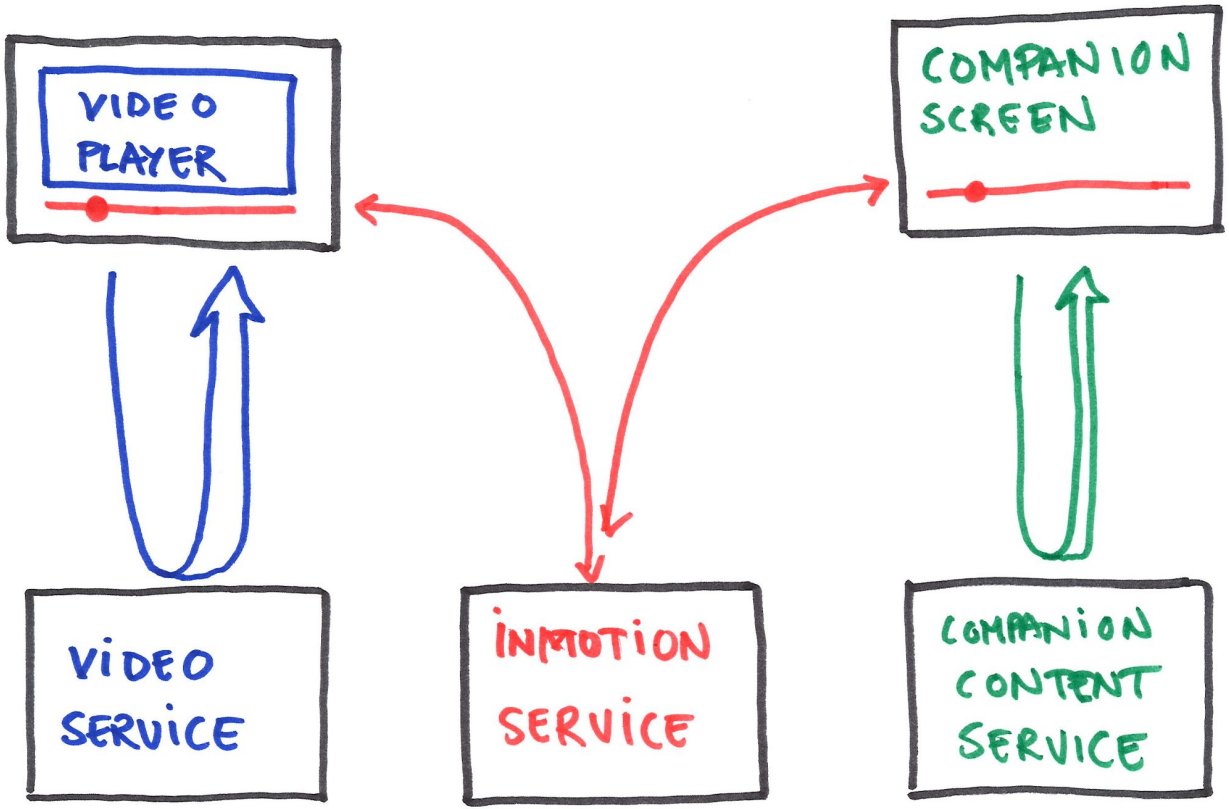
Summary

- Web already excellent platform for precisely timed multimedia!
- With standardization - even better!
- Consider joining the Multi-device Timing CG

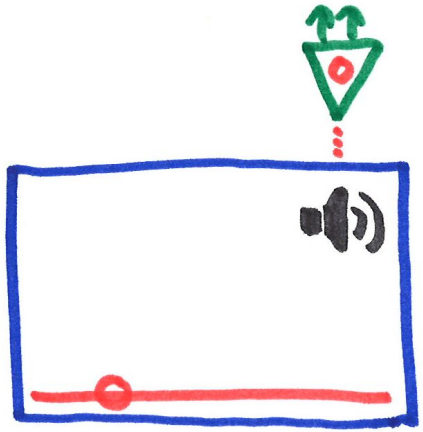
Thank you!

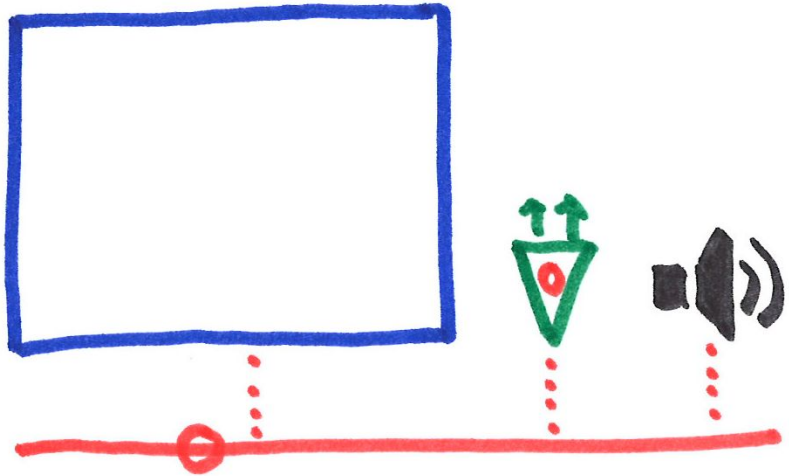
Appendix

Integration

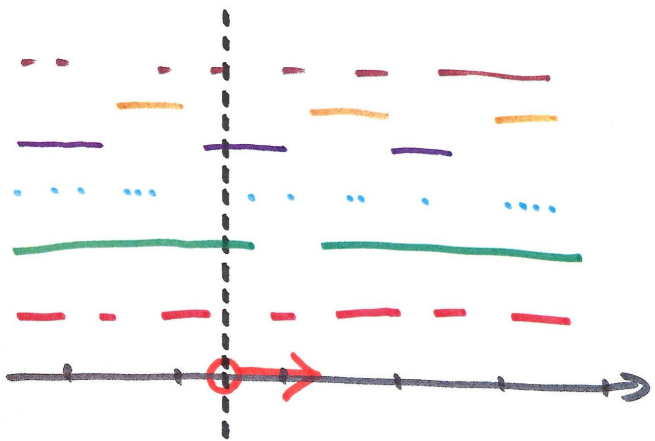


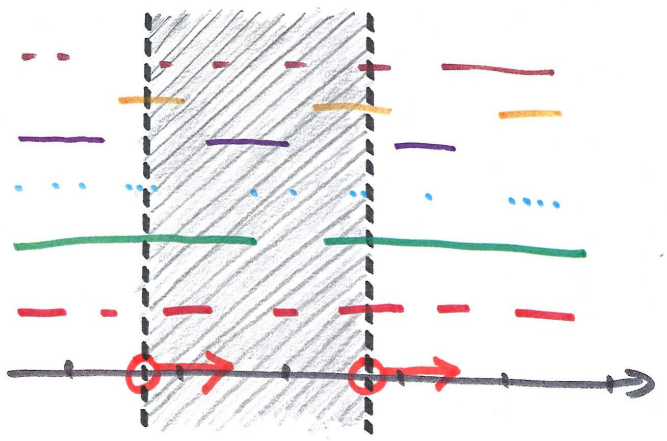
Media-centric -> Timing-centric





Default sequencing - Window sequencing





Implementation

- Goals
  - Precision, low resource consumption, correctness
- setTimeout
  - Precision typically down to a few milliseconds
  - Push for improvements in native setTimeout

Limitations

- Goals
  - simple concept - generic programming tool
- No particular support for
  - Relative timing statements
  - Repetitive patterns
  - Structured key-space
- Avoid - one sequencer that does it all.
- Instead - different sequencers suitable for different problems.