River Trail

Lots of folks from lots of places
I’m Rick Hudson, Stephan Herhut is with us also.
Summary

• Intel Labs River Trail project gently extends JavaScript with data-parallel constructs
  – Unlocks vector, multi-core, and GPU from JavaScript
  – Enables new richer browser experiences
  – Preserves simplicity and safety of web programming
  – Leverages improving computation / watt
• Prototype is up and running on Firefox
  – 4-15x performance improvement on 4 core chips
• Proposed spec on wiki and ECMA site
• Various demos floating around
Goal

• Prototype announced last September
  – Firefox extension now on GitHub
  – Prototype leverages OpenCL but lacks close integration with FF JITs

• Intel and Mozilla expect to partner on FF

• Several ISVs working on demos
  – Feedback has helped a lot

• Goal: work with ECMA to create spec in 2012
  – Final approval expected to take longer
  – But this seems like a good first step
Safety and Security: no more, no less

- Performance is important
  - But safety and security are requirements

- Preserves JavaScript safety model
  - No pointers, just object references
  - Automatic memory management
  - Full array bounds checking
  - Nothing more nor less than what is there now
Determinism

• Deterministic execution
  – No race conditions
  – No Deadlock (No Livelock)

• Value non-determinism possible
  – Evaluation order in reduction operations
  – Floating Point effects
Programmers’ Productivity

• Preserve familiar programming model and conventions
  – Parallel kernels written in JavaScript
  – Uses JavaScript’s object oriented model

• Looks like / behaves like JavaScript
  – Follows JavaScript semantics
  – Reference implementation in JavaScript
  – Interoperates with HTML5, WebGL
ParallelArray

• Basic data type for parallel computation
• Created from
  – A JavaScript array
  – Typed array
  – Canvas
  – Comprehension
• Immutable
• Single or multiple dimensions
ParallelArray Methods

- Combine
- Reduce
- Scan
- Scatter
- Filter
- Map
- Plus a constructor and accessor
- Others can be built on top of the above
  - Sum, Max, Add, Gather, Histogram, etc.

Do Few Things Well
Kernel Function

• Methods take kernel function as an argument

  • *This* within kernel function is ParallelArray
    – Orthogonality important
    – Helps composition

• combine and filter arguments
  – index passed as argument
  – get can use the index regardless of depth (dimensionality)

• reduce, scan, and scatter conflict arguments
  – 2 values passed args one value returned
Add 1 to Every Element in A

Sequential

```javascript
var i;
var a = new Array (...);
var b = new Array(a.length);
for(i=0;i<a.length;i++){
    b[i] = a[i] + 1;
}
```

Data parallel

```javascript
var a = new ParallelArray(...);
var b = a.map(
    function(val){return val+1;}
);
```
Add 1 Combine-Style

Sequential

```javascript
var i;
var a = new Array (...);
var b = new Array(a.length);
for (i=0; i<a.length; i++) {
    b[i] = a[i] + 1;
}
```

Data parallel

```javascript
var a = new ParallelArray(...);
var b = a.combine(
    function (i) {
        return this.get(i) + 1;
    }
);
```
Sum Reduce-Style

Sequential

```javascript
var i;
var a = new Array (...);
var sum = 0;
for (i=0; i<a.length; i++) {
    sum += a[i];
}
```

Data parallel

```javascript
var myPA = new ParallelArray(...);
var sum = myPA.reduce((function (a, b) {
    return a + b;
})
);```

_Data Parallelism is Beautiful_
Challenges and Competition

• OpenCL today
  – Useful HW abstraction appropriate for hiding implementation detail
  – Extends C99 in ways appropriate to C programmers
  – Allows ultimate control, performance, and access to HW
• WebCL provides thin layer around OpenCL
• WebCL faces serious security challenges
  – Define the many situations where the OpenCL standard leaves things undefined, for example out of bounds.
  – OpenCL makes these the programmer’s responsibility
  – OpenCL evolving to meet needs of C and HPC programmers
  – Not a reasonable approach for web

• Shared challenge – GPUs do a poor job of context switching and this creates performance hazards
  • River Trail can fall back to JavaScript library or OpenCL CPU execution
  • Current implementations focused on CPU
Goals

• Brought to ECMA today as first step
• Expect many suggestions and revision but we believe this basic approach is the way forward
• Why Intel
  – Watts for parallel instructions is low
  – Must meet the challenge of parallel programming for productivity programmer or HW and SW will diverge or go to a lowest common denominator