River Trail – Parallel Programming in JavaScript*

Stephan Herhut
Intel Labs
September, 24th 2012

In This Talk

1. Why parallel programming matters for the web
2. How best to add parallel programming to JavaScript*
3. What to do with it once you have it
4. How **You** can get involved

Why Parallel Programming in JavaScript*

The Web Is Evolving

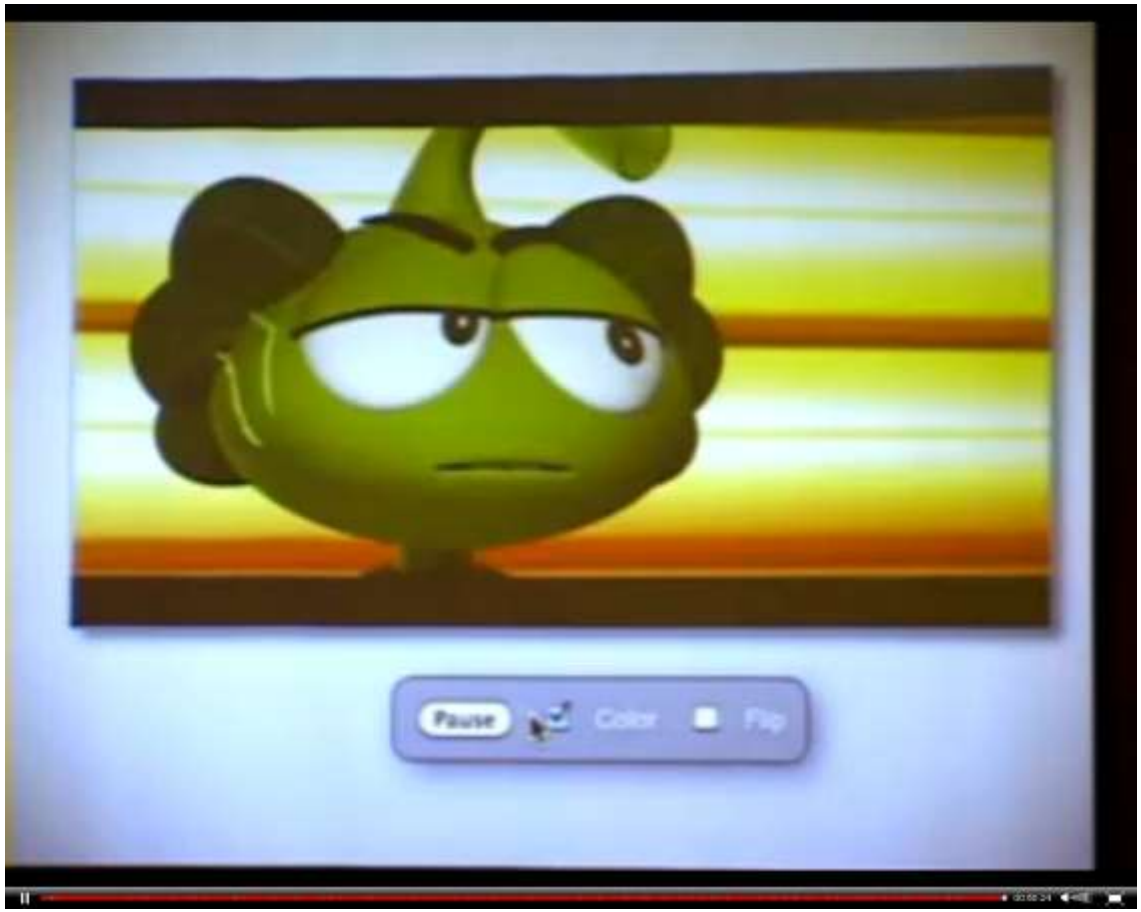
- Storage
- Multimedia
- 3D & Effects
- Device Access
- Connectivity

“very, very powerful stuff”

David Geary at TheStrangeLoop 2011
on <video> and <canvas>

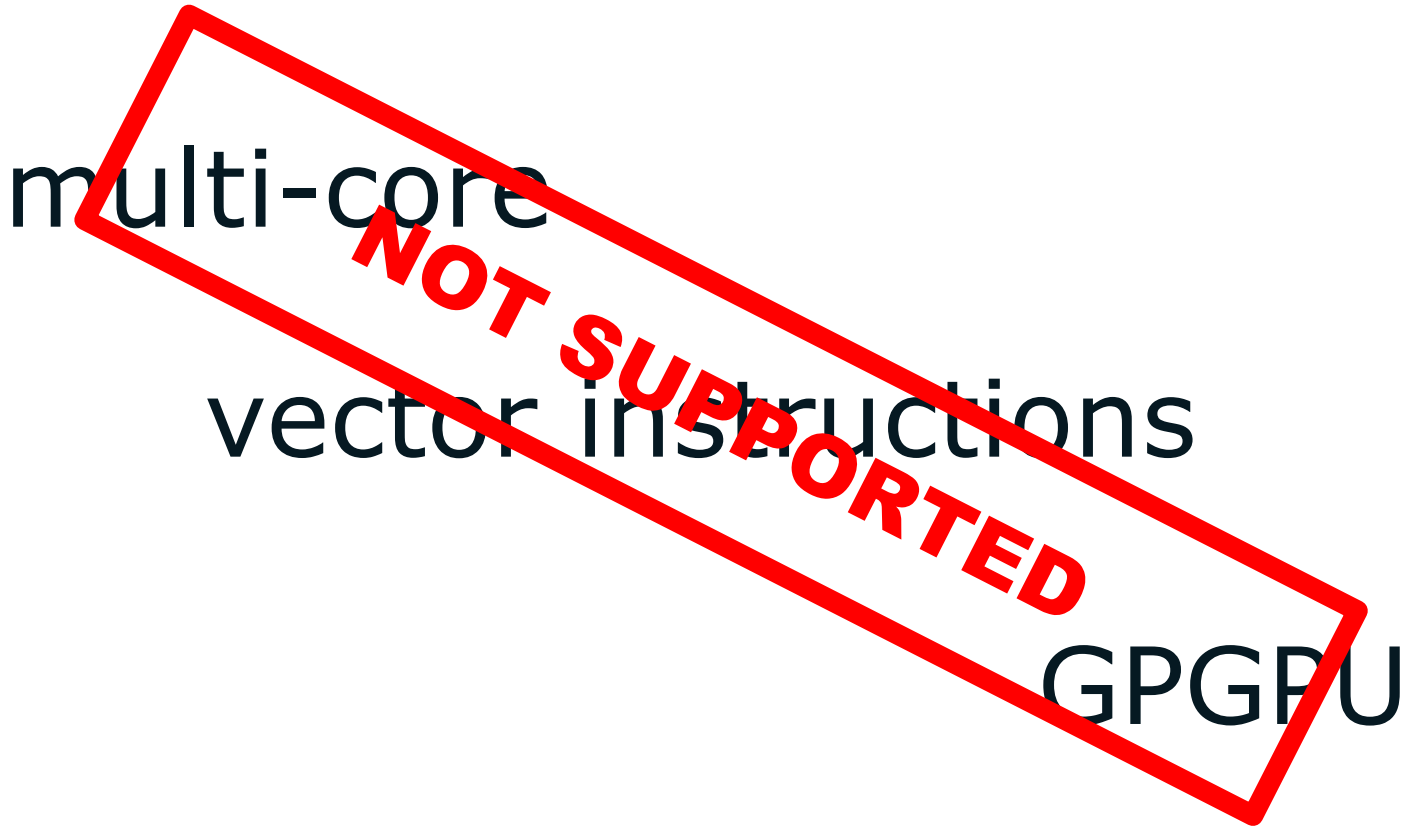
http://www.infoq.com/presentations/Core-HTML5-Canvas

David's <canvas> and <video> Demo



http://www.infoq.com/presentations/Core-HTML5-Canvas

Parallel Hardware Everywhere



Yet Another Programming API

"Meant to be a scripting language […] for the designer, the amateur programmer, the beginner programmer"

Brendan Eich at FluentConf 2012

https://www.youtube.com/watch?v=Rj49rmc01Hs

River Trail Design Goals

1. Ease of use

http://www.flickr.com/photos/jcestnik/5666428672

River Trail Design Goals

1. Ease of use

2. Code reuse and mash-up coding


Video Share
California Christmas
Rate
Share
Favorite
Playlist
Flag


09:45
Incoming call
Unknown
menu

River Trail Design Goals

1. Ease of use
2. Code reuse and mash-up coding
3. Performance portability

http://www.flickr.com/photos/carbonnyc/2294144289/

River Trail Design Goals

1. Ease of use
2. Code reuse and mash-up coding
3. Performance portability
4. Safety and Security

Challenge: meet these criteria and get good performance

The Design of River Trail

Concurrency in JavaScript* Today

- Cooperative multi-tasking
  - Scripts compete with the browser for computing resources
  - Event driven execution model
- Concurrent programming mindset
  - Asynchronous call-backs for latency hiding
- Fully deterministic
  - Run-to-completion semantics
  - No concurrent side effects
- **No support for concurrent execution**
  - Single threaded evaluation of JavaScript

Design Decisions

- Deterministic execution model
  - No race conditions, no dead lock (no live lock)
- Looks and behaves like JavaScript
  - All code still written purely in JavaScript
- Uses high-level parallel patterns
  - Express parallel code without tying it down to a specific implementation
- Maintains JavaScript's Safety and Security
  - Still use managed runtime, still no pointers

Three Pillar Approach

- Data structure: **`ParallelArray`**
  - Immutable, dense and homogeneous
- Six Methods: **`map`**, **`combine`**, **`reduce`**, **`scan`**, **`filter`**, **`scatter`**
  - Provide the basic skeletons for parallel computing
  - Typically creates a freshly minted ParallelArray object
- Elemental functions
  - Specify the actual workload
  - Written purely in JavaScript*
  - Side effect free

ParallelArray construction

```
new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


new ParallelArray(4, function (i) { return i+1; })

//-> [1, 2, 3, 4]


new ParallelArray([2,2],
  function (iv) { return iv[0] + iv[1]; })

//-> [0, 1, 1, 2]                  // [[0, 1], [1, 2]]
```

Map Method

```
var source = new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


source.map(function inc(v) { return v+1; })

//-> [2, 3, 4, 5]
```


http://rivertrail.github.com/interactive/

Combine Method

```
var source = new ParallelArray([1,2,3,4]);

source.combine(
  function inc(iv) { return this.get(iv)+1; }
)

//-> [2, 3, 4, 5]


source.combine(
  function rev(iv) {
    return this.get(this.length - iv[0] - 1); }
)

//-> [4, 3, 2, 1]
```

Combine Method in 2D

```
var source = new ParallelArray([2,2],
  function (iv) { return iv[0] + iv[1]; })

//-> [0, 1, 1, 2]                    [[0, 1], [1, 2]]


source.combine( 2,
  function rev(iv) {
    return this.get([this.getShape()[0]-iv[0]-1,
                     this.getShape()[1]-iv[1]-1]); }
)

//-> [2, 1, 1, 0]                    [[2, 1], [1, 0]]
```

Reduce Method

```
var source = new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


source.reduce(
  function plus(a, b) { return a + b; }
)

//-> 10
```


http://rivertrail.github.com/interactive/

Scan Method

```
var source = new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


source.scan(
  function plus(a, b) { return a + b; }
)

//-> [1, 3, 6, 10]
```


http://rivertrail.github.com/interactive/

Filter Method

```
var source = new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


source.filter(
  function isOdd(iv) { return this.get(iv) % 2; }
)

//-> [1, 3]
```

Scatter Method

```
var source = new ParallelArray([1,2,3,4]);
//-> [1, 2, 3, 4]

source.scatter([3,0,2,1]);
//-> [2, 4, 3, 1]

source.scatter([2,0,2,1]);
//-> RangeError : Duplicate indices in scatter
```

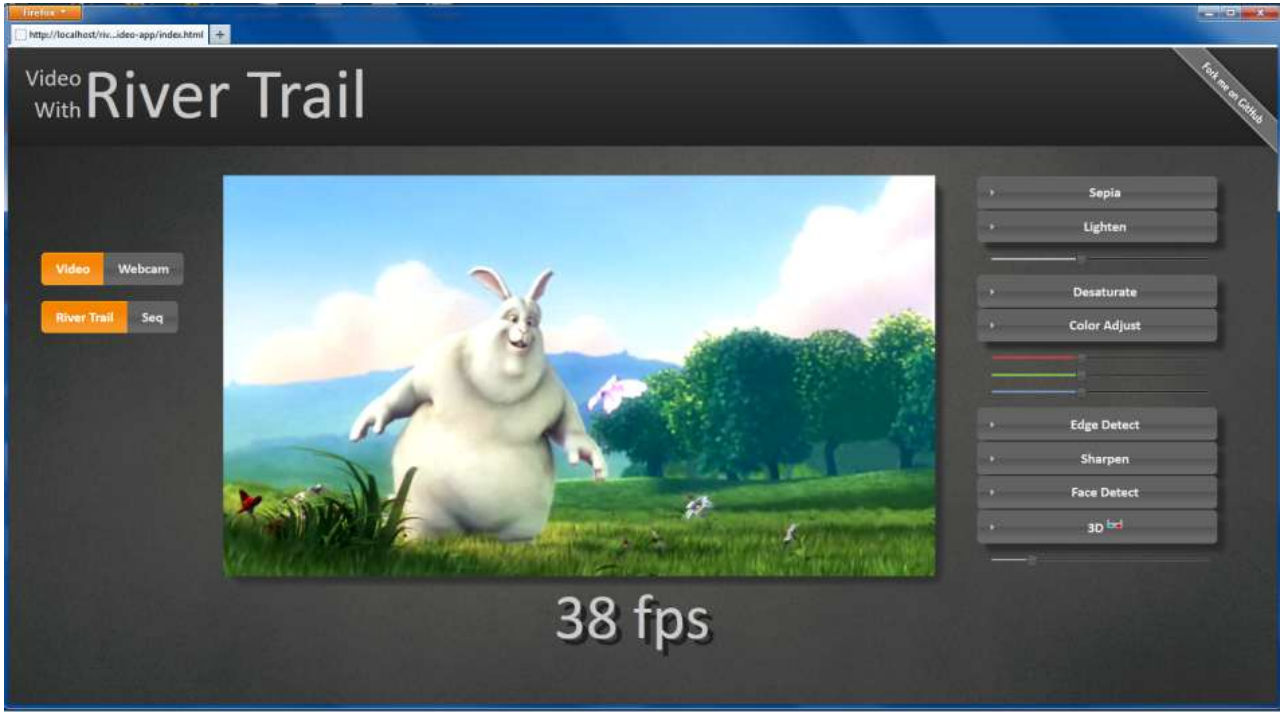
Scatter Method With Conflict

```
var source = new ParallelArray([1,2,3,4]);

//-> [1, 2, 3, 4]


source.scatter([2,0,2,1], 9,
               function plus (a, b) { return a+b; });

//-> [2, 4, 4, 9]


source.scatter([2,0,2,1], 9,
               function plus (a, b) { return a+b; },
               3);

//-> [2, 4, 4]
```


http://rivertrail.github.com/interactive/

River Trail in Action

Video With River Trail

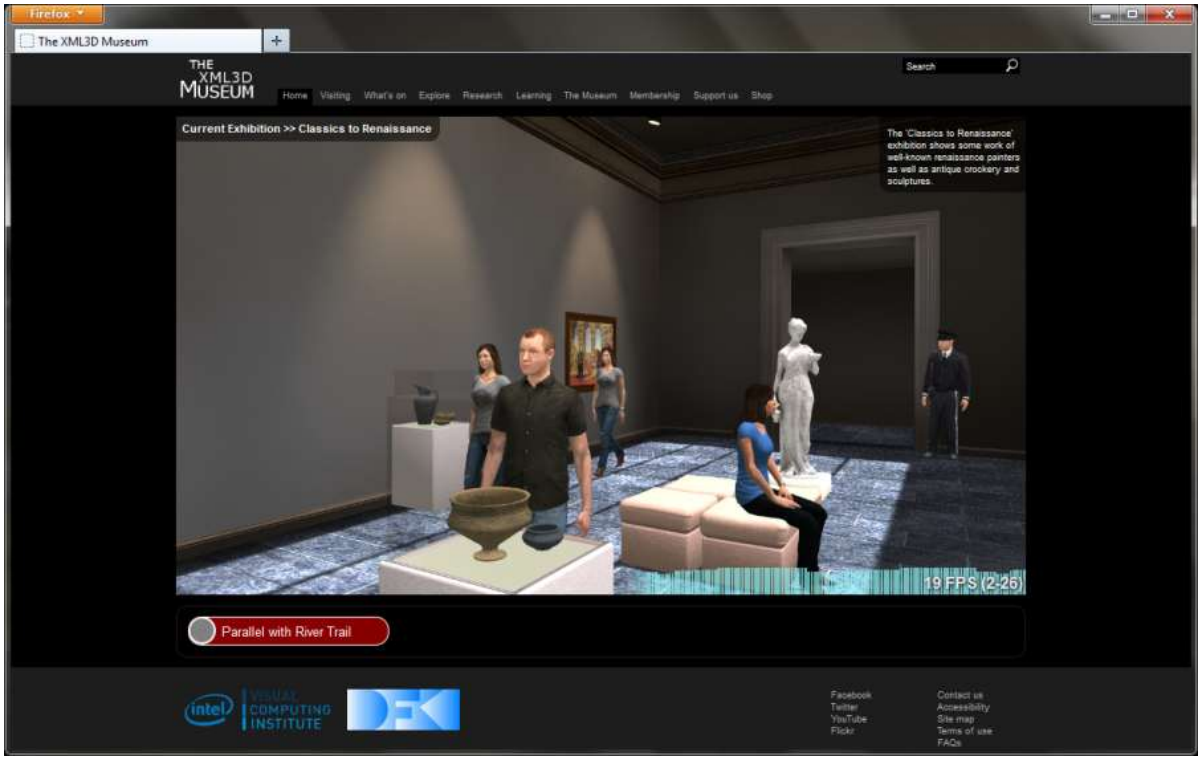


http://github.com/RiverTrail/RiverTrail/tree/master/tutorial

Physics Based Gaming in the Browser

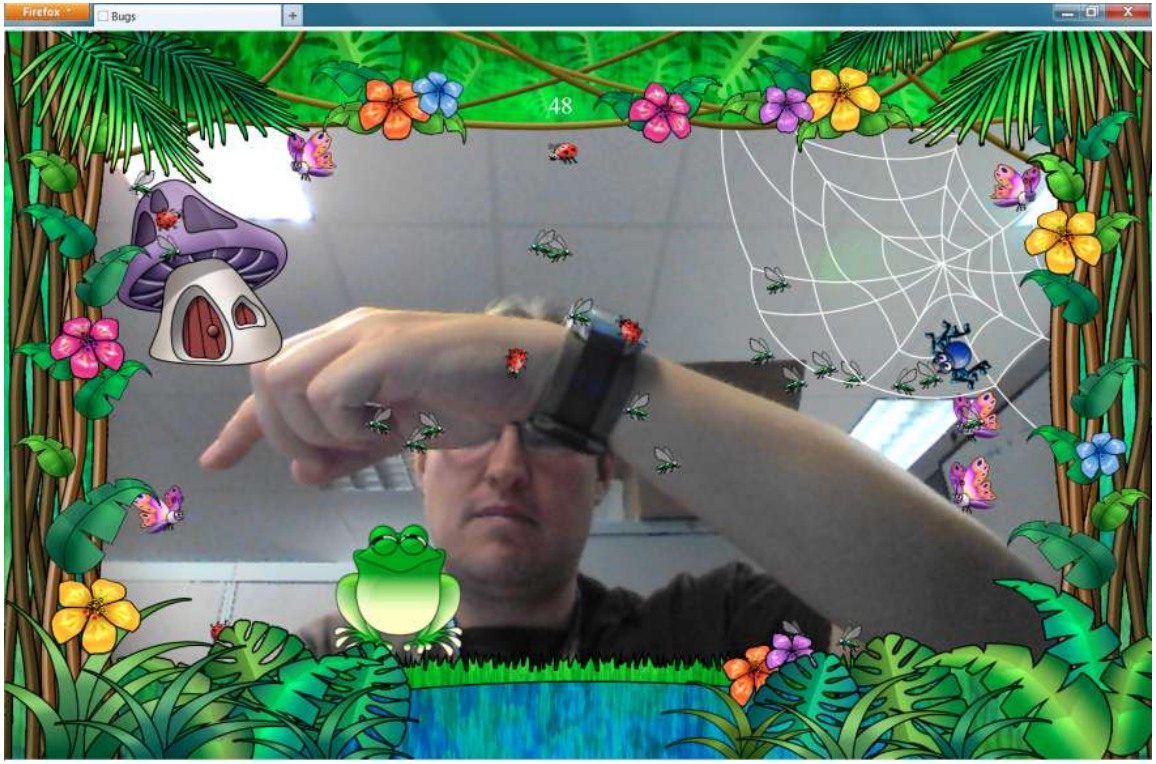
River Trail is used to **accelerate the physics computations** (collision detection)

3D Avatar Animation in the Browser



River Trail is used to **accelerate the animation of characters** (skinning)

Visual Computing in the Browser



River Trail **powers the computer vision algorithms** (optical flow) for motion tracking

Getting Involved

Try River Trail Today

- Open source Mozilla* Firefox* prototype available on GitHub*
  - Pre-built binary extension for current Firefox release
  - Sequential library fall back for other browsers

  http://github.com/RiverTrail/RiverTrail/wiki

- REPL to play with the API
  - Works in Chrome* and Firefox

  http://rivertrail.github.com/interactive/

- Full "Video with River Trail" tutorial
  - A fun way to learn River Trail

  http://rivertrail.github.com/RiverTrail/tutorial/

Help Shape The Future

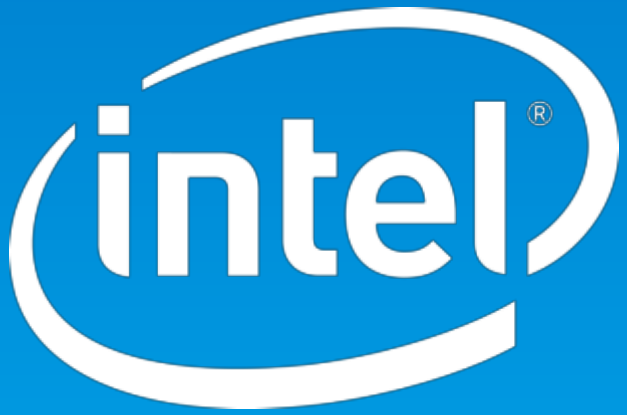
- ECMA proposal for Parallel JavaScript*
  - Full API specification published

  http://wiki.ecmascript.org/doku.php?id=strawman:data_parallelism

- First implementation in Firefox* Nightly
  - Implements full API
  - Work in progress

  http://nightly.mozilla.org/

- Join the discussion at es-discuss@mozilla.org
- Get in touch: here, @herhut, stephan.a.herhut@intel.com
- Go, Play, Create!


intel

Legal Disclaimer

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.
- Intel may make changes to specifications and product descriptions at any time, without notice.
- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
- Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- River Trail and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user
- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark* and MobileMark*, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.
- Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: http://www.intel.com/products/processor_number
- Intel product plans in this presentation do not constitute Intel plan of record product roadmaps. Please contact your Intel representative to obtain Intel's current plan of record product roadmaps.
- Intel, Core and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
-