Best Practices for Building Modern Web Apps

Angel Todorov

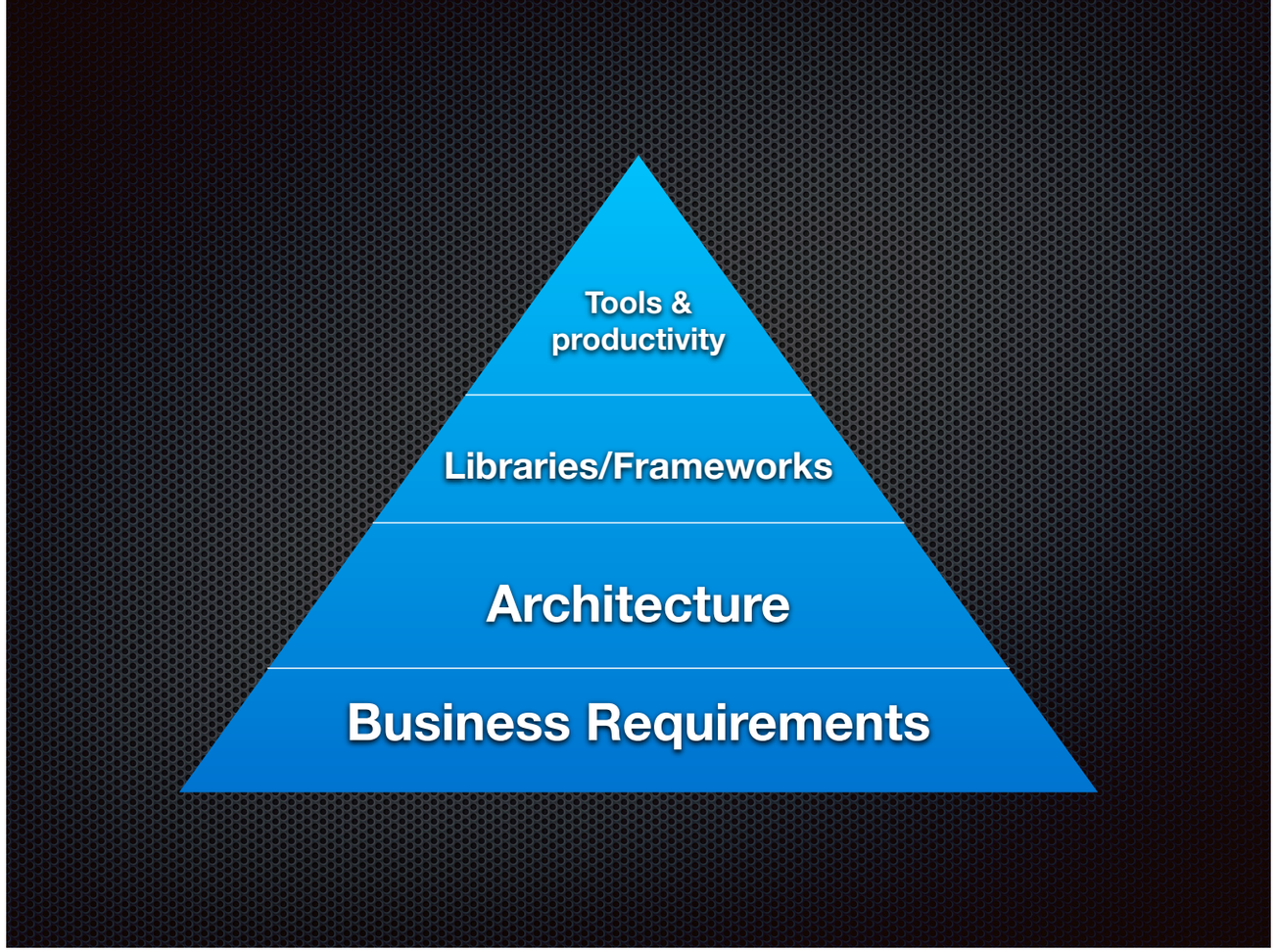
Principal Architect, Infragistics Inc.

About Me

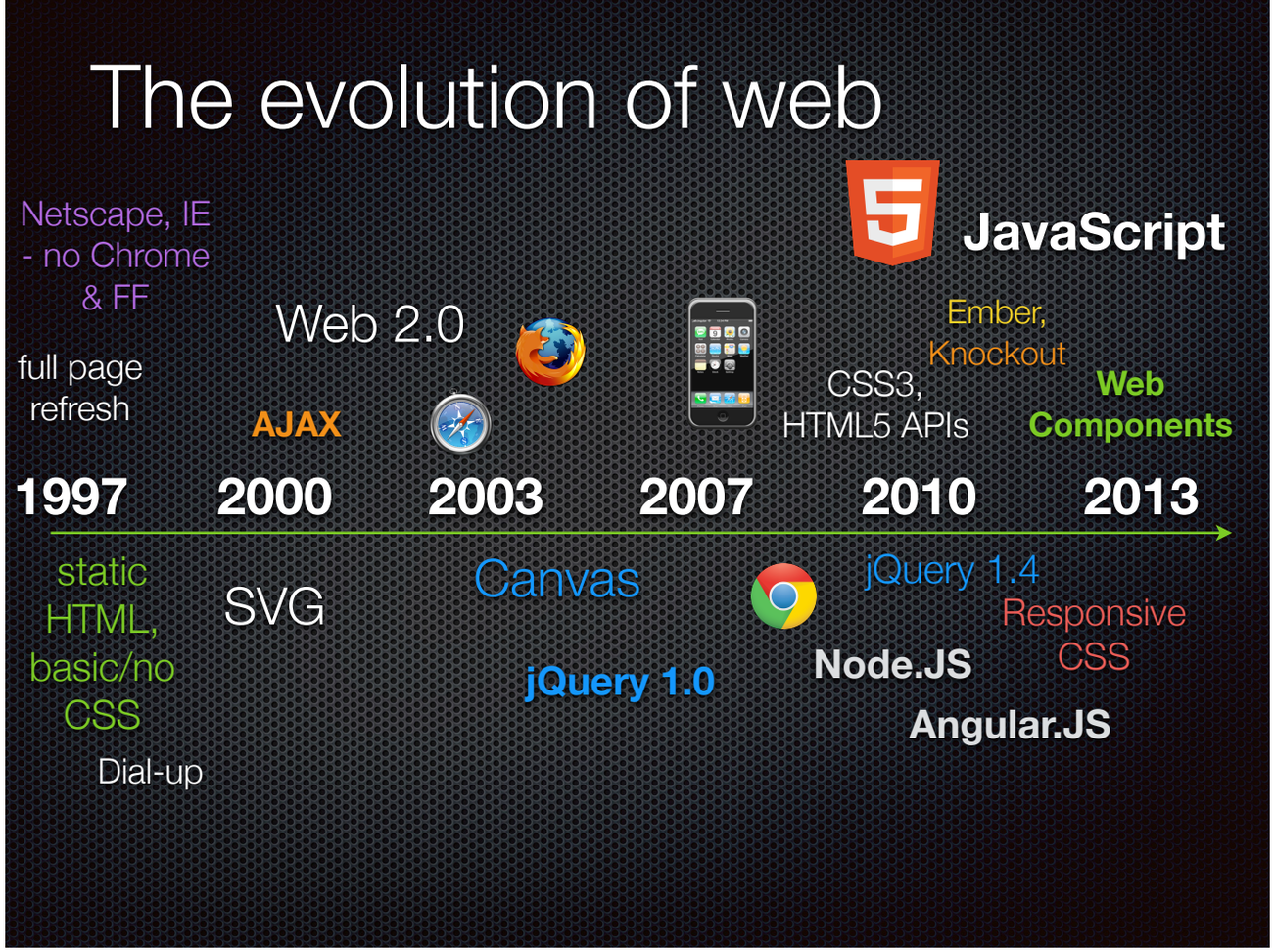
- over 10 years of software development experience
- Proud to be @ Infragistics since 2007
- Love to code for fun
- Have special interests in machine learning and compilers
- Love diving, big movie fan

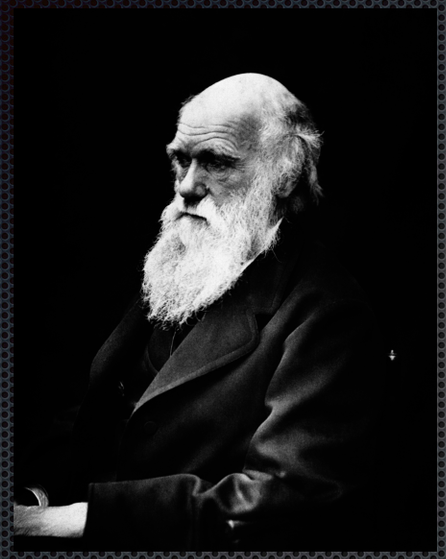
Overview

- Some background
- Know your business requirements
- Anatomy of a web app
- Web App Architectures
- Know your libraries
- Building for multiple devices and platforms
- Know your development tools (Client side)


Tools &
productivity
Libraries/Frameworks
Architecture
Business Requirements

The evolution of web

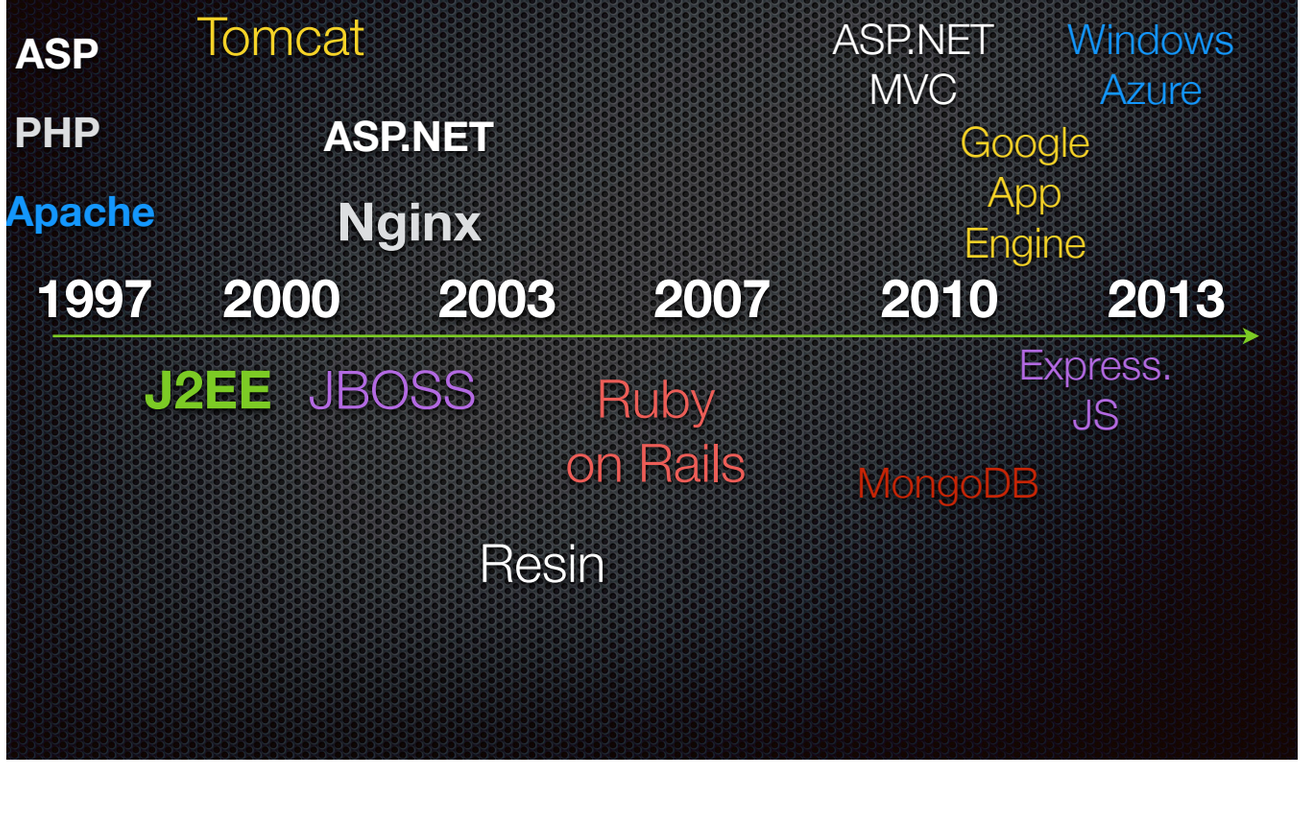




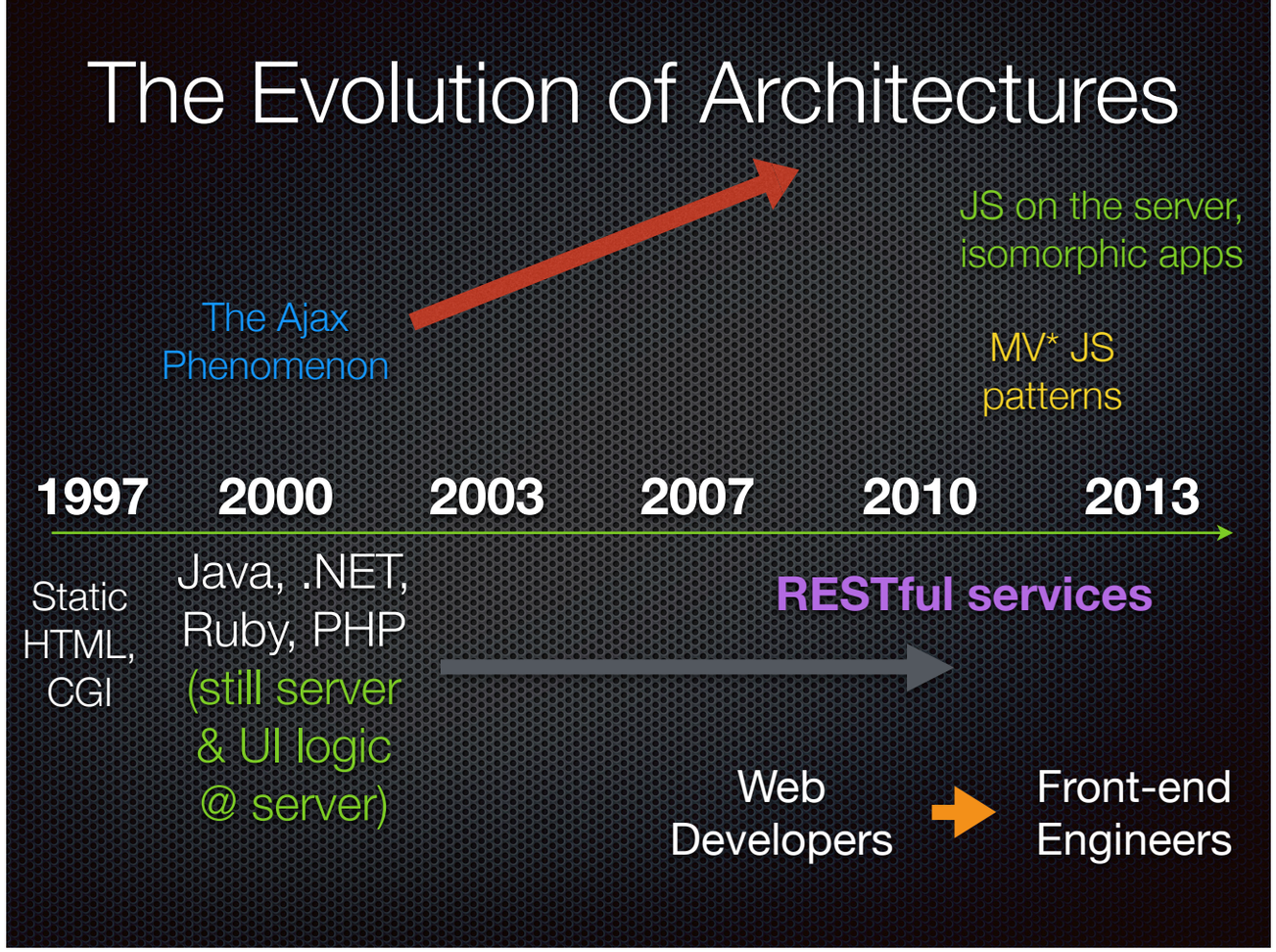
"It is not the strongest or the most intelligent who will survive but those who can best manage change."

–Charles Darwin

The Evolution of Web & App Servers



The Evolution of Architectures



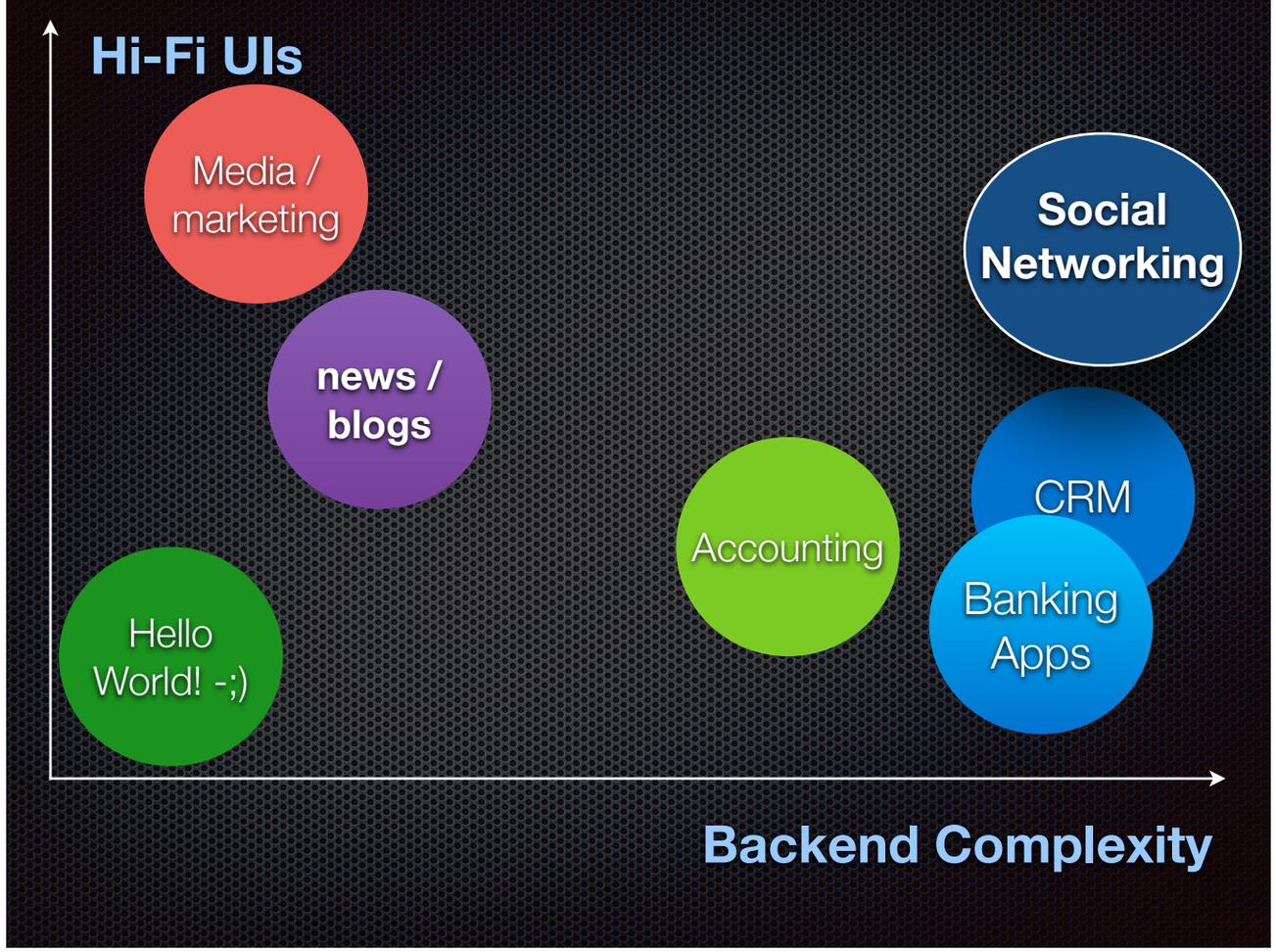
Modern Web Apps

- Fast
- Beautiful
- Interactive
- Responsive and Fluid
- Offline
- REST
- Single Page
- HTML5 / CSS3
- **JavaScript**
- Cloud-enabled
- Touch-enabled

Know your requirements

- Don’t just use what’s new and cool !
- Spend most time on the **core**
- Make sure your customers are **happy**
- Use an **agile** development approach
- Identify potential **bottlenecks** & adjust architecture
- Will your architecture be still valid in 1, 2, 5 years?


beautiful UI on
bad business
logic
is like lipstick
on a pig


Hi-Fi UIs
Media / marketing
news / blogs
Hello World! -;)
Social Networking
CRM
Accounting
Banking Apps
Backend Complexity

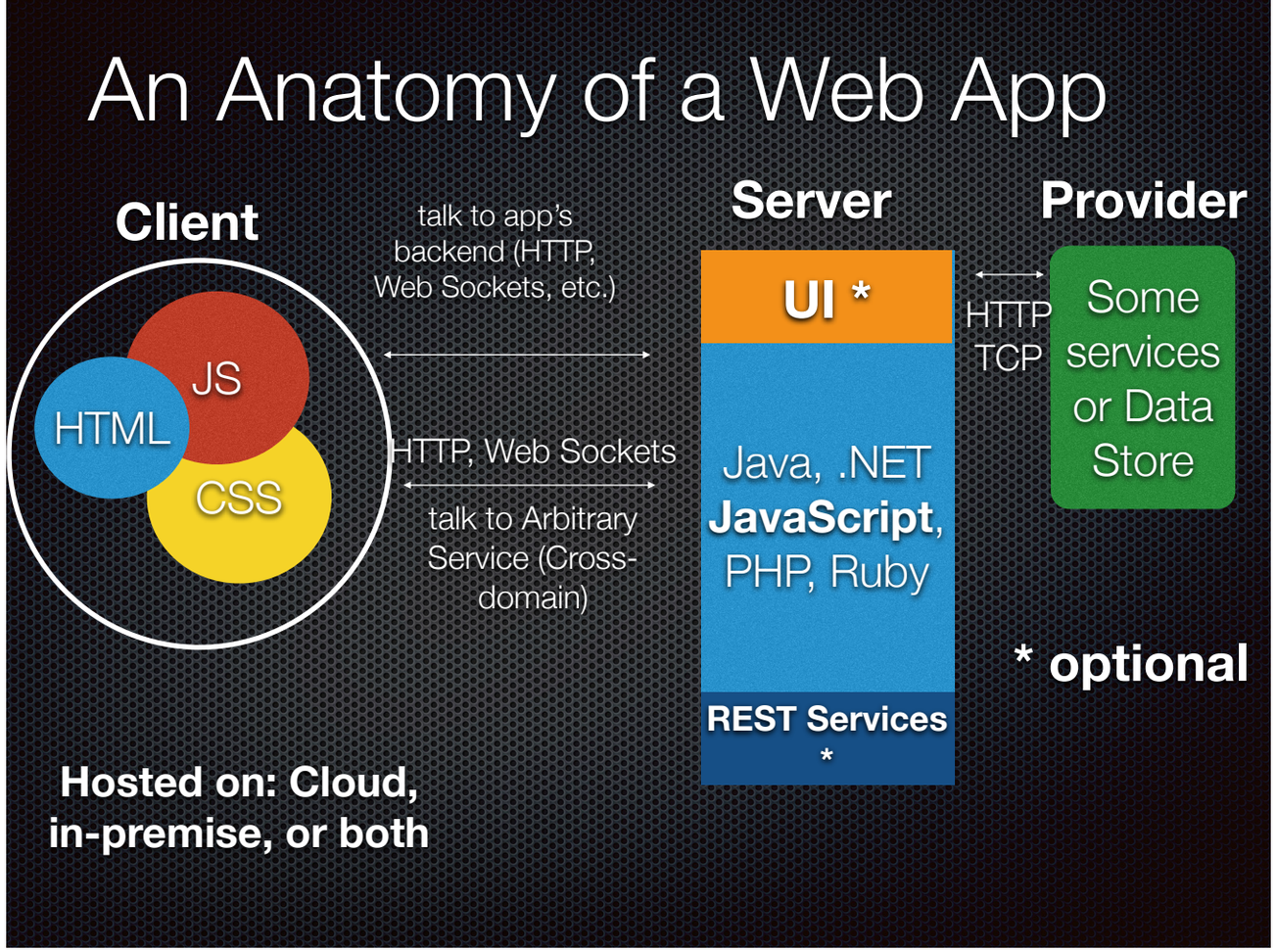
Ways to validate your requirements

- Use a Lean UX Approach
- Provide your customers with access to a demo environment which updates live
- Explore reference apps
- Show customers reference apps
- Always build your UI with responsive design in mind

WHAT HAS BEEN SEEN...

...cannot be unseen

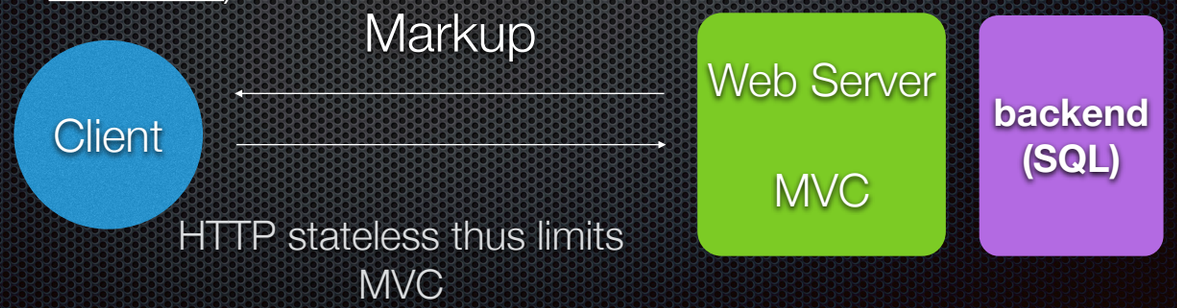
An Anatomy of a Web App



Common stack recipes

The Traditional

- all UI (or parts of it) rendered on server
- Some JavaScript & AJAX
- MVC on the server (traditional RoR, Spring, etc.)
- no REST services
- Multi-threading on the server
- notion of server-side UI tree & **state** mgmt, auth (JSF, ASP.NET)



Common stack recipes

The Modern

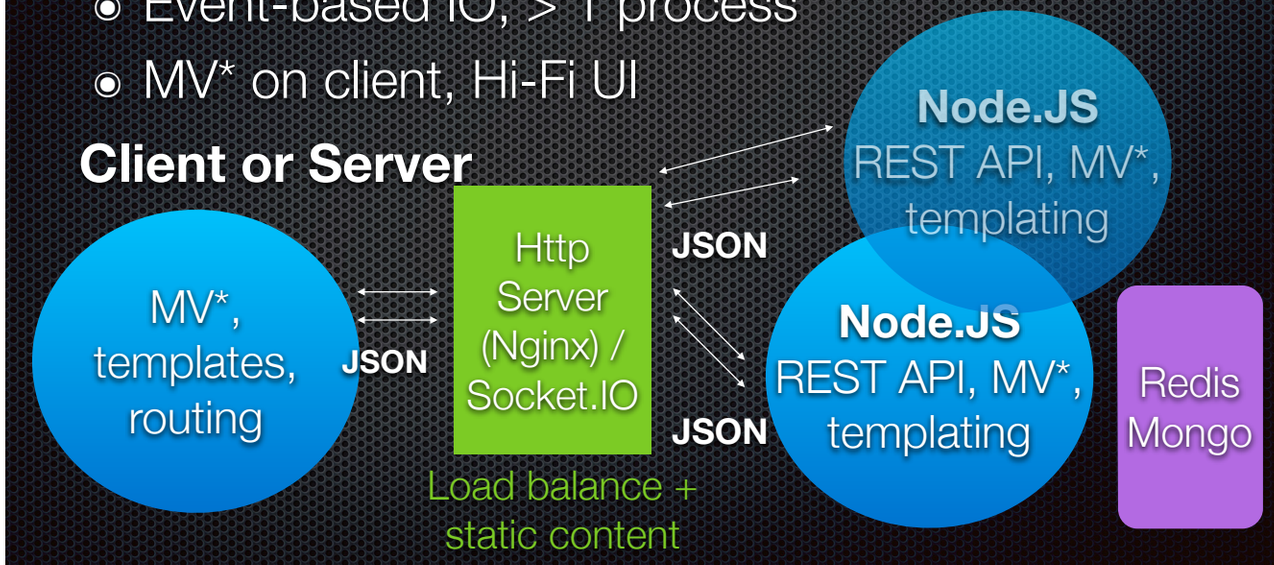
- Single Page apps , Hi-Fi UIs
- RESTful JSON API (RFC 2616, HTTP/1.1)
- HTTP Basic, OAuth
- MV* JavaScript frameworks & jQuery (Knockout, Angular.JS, Ember.JS)
- Client-Side Templates

Common stack recipes

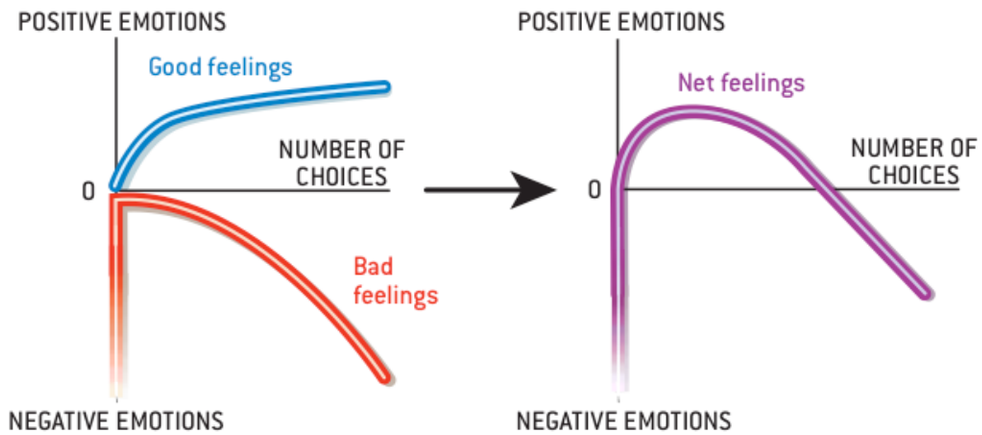
The Future ?

- JavaScript
- Code can run on both server & client
- Event-based IO, > 1 process
- MV* on client, Hi-Fi UI

Client or Server



REACTIONS TO INCREASING CHOICE



Choices - event-based vs multi-threading

Event-based

- great for real-time communication & handling LOAD
- suffers when we run **blocking** code
- Very small footprint. HTTP server integrated (Node.JS)

Multi-threaded

- Not easy to handle huge load
- Good when doing blocking processing
- **Not intuitive for a Front-End JavaScript developer**

The cost of I/O

L1-cache	3 cycles
L2-cache	14 cycles
RAM	250 cycles
Disk	41 000 000 cycles
Network	240 000 000 cycles

2 GHz CPU = 2,000,000,000 cycles every second

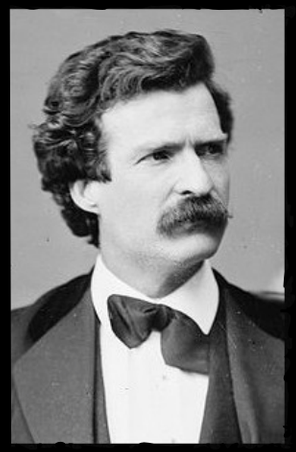
Event-based example

```
http.createServer(function (req, res) {
    database.getInformation(function (data) {
        res.writeHead(200);
        res.end(data);
    });
});
```

Put Nginx in front and have one Node per CPU core !

Use Nginx, Apache or IIS for static content!

There are alternatives in Java & .NET (e.g. Grizzly)



There are three kinds of lies — lies, damned lies and statistics.

(Mark Twain)

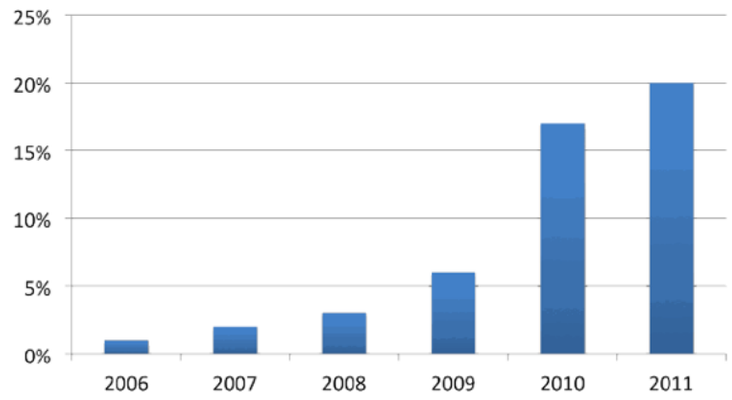
Choices - NoSQL vs. SQL

- Use a RDBMS when you need reporting (e.g. JOINS)
- Use a RDBMS when you need **transactions**
- NoSQL (MongoDB) plays nicely with JSON (BSON) & JavaScript
- NoSQL - best for storing **unstructured** data
- Use an in-memory DB (**Redis**) for **volatile** data

Choices - JSON vs XML

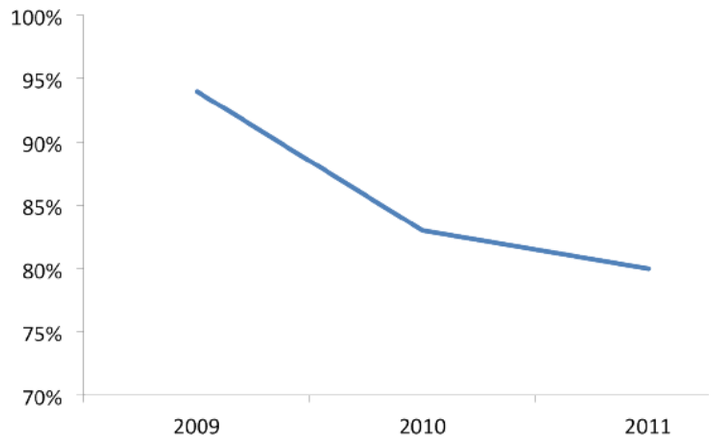
- XML is verbose
- JSON is native to JavaScript & natural to use
- JSON - fast & small
- XML has namespaces
- **API reliance on XML is going down very fast**

programmableweb.com



*Percentage of new APIs with **only** JSON support*

programmableweb.com



Percentage of APIs with XML support

Choices - MVC vs MVVM vs MVWhatever works for you

- The View Model - specialized controller that acts as a data converter (Knockout)
- MVVM Falls short when it comes to things like routing
- MVVM reduces glue code
- MVC *appears* to be flexible for more complex apps, **dynamic views**
- **TodoMVC - same app written in many MV* fw**

Example: Airbnb

- Hybrid approach - Isomorphic Javascript apps - can run on both the client-side and the server-side
- Use Node.JS
- REST services (Rails App)
- Isomorphic Features:
  - Routing
  - getting/persisting data
  - view rendering
  - building & packaging (Grunt & Browserify)

Example: LinkedIn

- many web apps
- started with JSPs and Servlets
- Ruby on Rails for some apps
- Difficult to write reusable front-end features
- Start using client templates & serve JSON
- Use Node on the server
- **rebuilt mobile apps using HTML5 & Node.JS**

The Genius of the AND

The Tyranny of the OR

Structure of a web page

```
<!DOCTYPE HTML>
<html>
  <head>
      [CSS]
      [Scripts]
      <script type="text/javascript">
          // when the document is ready
      </script>
  </head>
  <body>
      [basic static layout]
      [Scripts!]
  </body>
</html>
```

Mind your requests!

Know your libs - **jQuery**

Cache Selectors

Bad

```
$("#parent").find(".list").doSomething();
$("#parent").find(".list").doSomethingElse();
```

Good

```
var list = $("#parent").find(".list");
list.doSomething();
list.doSomethingElse();
```

Know your libs - **jQuery**

Even better... Use Chaining where applicable

Bad

```
$("#elem").hide();
$("#elem").html("hello");
$("#elem").sthElse();
```

Good

```
$("elem")
  .hide()
  .html("hello")
  .sthElse();
```

Know your libs - **jQuery**

Append Once

Very Bad!

```
$.each(arr, function (index, val) {
   list.append("<li>" + val + "</li>");
});
```

Good

```
$.each(arr, function (index, val) {
   str += "<li>" + val + "</li>";
});
list.append(str);
```

Know your libs - **jQuery**

Use Event Delegation

Bad

```
$("#hugelist li").on("click", function () {
  // do something
});
```

Good

```
$("#hugelist").on("click", "li", function () {
  // do something
});
```

Know your libs - **jQuery**

Use Promises

Bad

```
function getData() {
  $.ajax({
    url: “mydata.php",
    success: function (data) {
      // do something
    }
  });
}
```

Good

```
function getData() {
  return $.ajax({
    url: "mydata.php"
  });
}
getData().done(function (data) { });
```

Know your libs - **jQuery**

Don’t Overuse !

Bad

```
$(this).attr("id");
```

Good

```
this.id
```

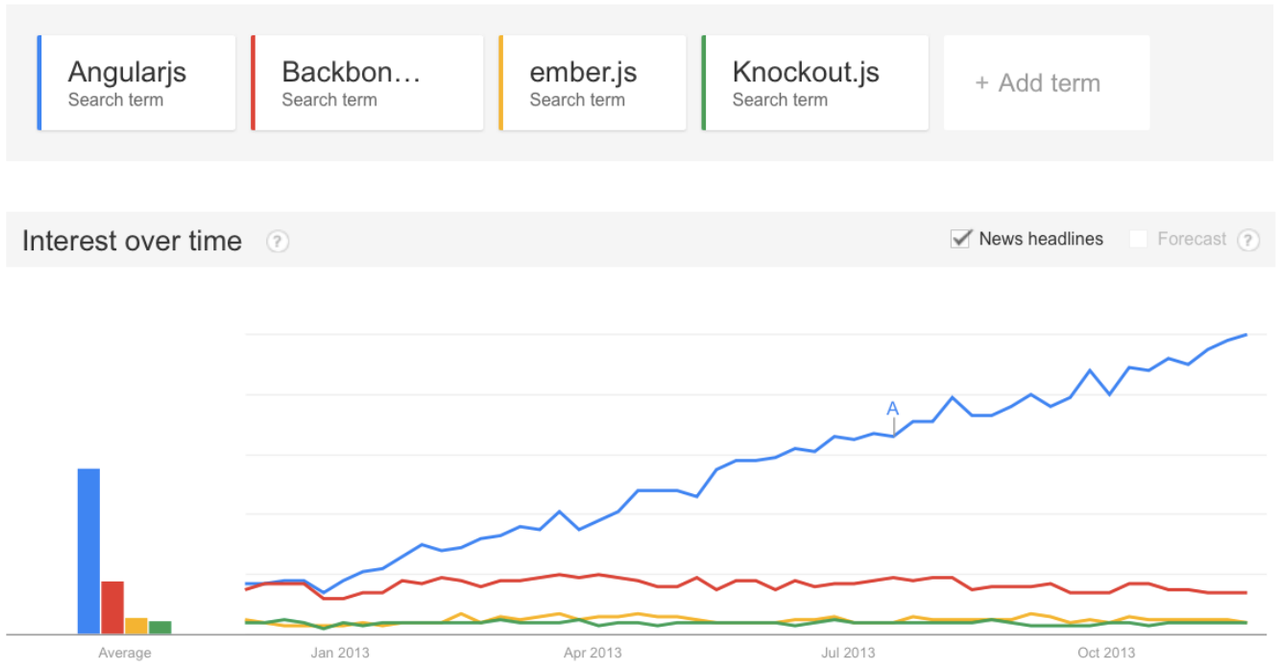
Know your libs - **MV***

	two-way bindings	navigation & routing	templates	lib size	custom tags
Knockout	Yes - accesors	**No**	**Yes** *	46 (16) Kb	No
Backbone	Yes - accessors	**No**	Yes *	20 (6) Kb	No
Angular	**Yes - dirty checking**	Yes	**Yes** *	99 (37) Kb	**Yes**
Ember	Yes - accessors	Yes	**Yes**	**249 (64) Kb**	No

* DOM-based

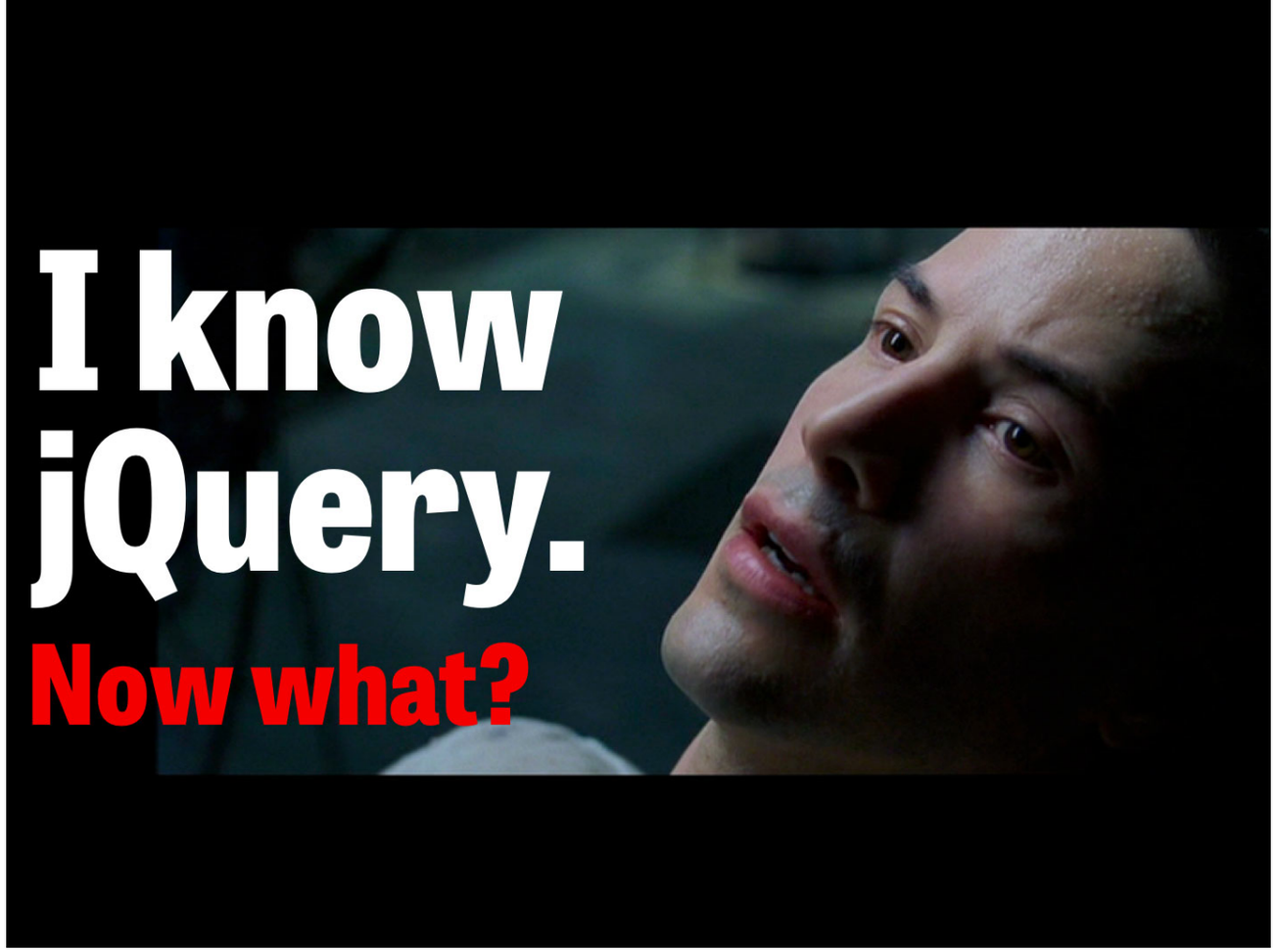
* String-based

Some statistics - use responsibly



jQuery has other friends, too…

- **RequireJS**
- Modernizr
- jQuery UI
- Underscore.js
- **Simple JavaScript Inheritance (John Resig)**


I know
jQuery.
Now what?

Building for multiple devices

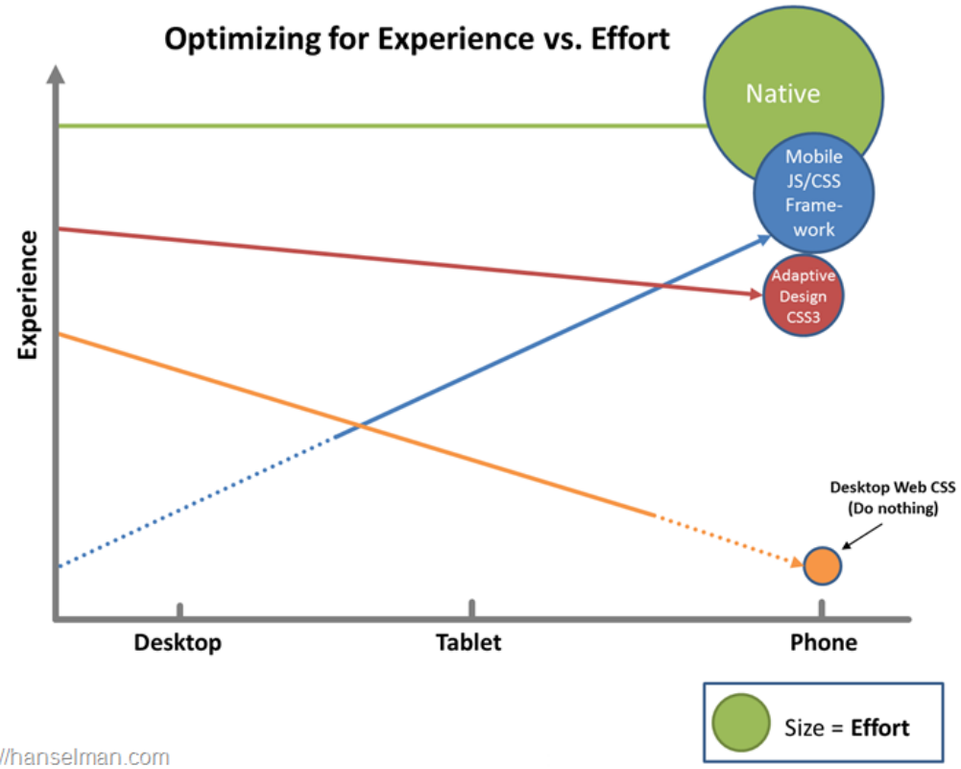
- Mobile First
- Choose your media queries (common device sizes)
- Use a grid system - Bootstrap, ZURB Foundation
- Use percentage widths & heights
- Support Touch Events
- But also … be fast for clicks! - Fastclick
- Use scalable/fluid images (**max-width:100%**)
- **Go Beyond CSS - Semantic responsiveness**

Semantic Responsiveness

Company Name	Alfreds Futterkiste
Contact Name	Maria Anders
Contact Title	Sales Representative
Address	Obere Str. 57
City	Berlin
Country	Germany
Company Name	Ana Trujillo Emparedados y helados
Contact Name	Ana Trujillo
Contact Title	Owner
Address	Avda. de la Constitución 2222
City	México D.F.
Country	Mexico

Table data rendered vertically on a phone

Hide data if it's too much!



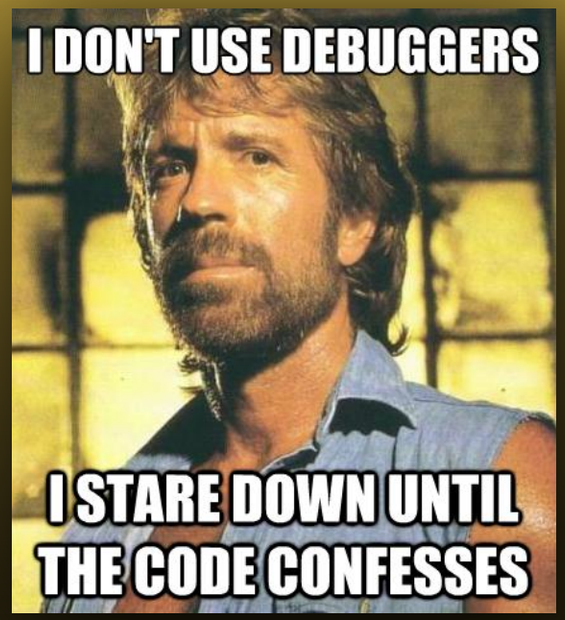
http://hanselman.com

"Getting information off the Internet is like taking a drink from a fire hydrant."

–Mitch Kapor

Know your tools - Debugging

- **Firebug**
- **Chrome Dev Tools**
- IE Dev Tools
- Use Code Maps
- Node.JS - node debugger, Node Inspector, Chrome Dev Tools


I DON'T USE DEBUGGERS
I STARE DOWN UNTIL
THE CODE CONFESSES

Know your tools - Performance

- **Chrome Dev Tools**
- **Firebug**
- Spy.js
- Speed Tracer (Google)
- John Resig's plugin for profiling jQuery apps
- Drip (IE Leak Detector)
- **Nodetime**
- Leak Finder for JavaScript (Chrome)

Know your tools - Security

- **JSLint**
- dom-xss-scanner (DOM-based XSS scanner)

Know your tools - CSS

- Gridpak (responsive grid generator) - gridpak.com
- gridsystemgenerator.com
- gridinator.com
- Zurb CSS Grid Builder

Know your tools - Testing & LINT

- **QUnit**
- Jasmine
- JSLint / JSHint
- **Blanket.js**
- Selenium WebDriver

Know your tools - Build & CI

- **Grunt**
- Bower
- Browserify
- **UglifyJS**
- **YUI Compressor (both Js and CSS)**
- Yeoman
- Jenkins

"There is no single development, in either technology or management technique, which by itself **promises** even one order of magnitude [tenfold] improvement within a decade in productivity, in reliability, in simplicity."

-Fred Brooks

"No Silver Bullet - Essence and Accidents of Software Engineering"


KEEP
CALM
AND
BE
CREATIVE

Q & A

Thank You!

atodorov@infragistics.com

Thanks to our Sponsors:

Diamond Sponsor:

Gold Sponsors:

Silver Sponsors:

Technological Partners:

Bronze Partners:

JobTiger.bg

Swag Sponsors:

Media Partners: