

Qualidade e Performance de Sistemas 2.0

- Ivo Nascimento
- iannsp@gmail.com |
ivo.nascimento@ianntech.com.br

O que é web 2.0

"Web 2.0 é a mudança para uma internet como plataforma, e um entendimento das regras para obter sucesso nesta nova plataforma. Entre outras, a regra mais importante é desenvolver aplicativos que aproveitem os efeitos de rede para se tornarem melhores quanto mais são usados pelas pessoas, aproveitando a inteligência coletiva"

Tim O'Reilly

Regras que ajudam a definir sucintamente a Web 2.0

1. O beta perpétuo
2. Pequenas peças frouxamente unidas
3. Software acima do nível de um único dispositivo
4. Lei da Conservação de Lucros, de Clayton Christensen
5. Dados são o novo “Intel inside”

1. O beta perpétuo

não trate o software como um artefato, mas como um processo de comprometimento com seus usuários.

2. Pequenas peças frouxamente unidas

Abra seus dados e serviços para que sejam reutilizados por outros. Reutilize dados e serviços de outros sempre que possível.

3. Software acima do nível de um único dispositivo

Não pense em aplicativos que estão no cliente ou servidor, mas desenvolva aplicativos que estão no espaço entre eles.

4. Lei da Conservação de Lucros, de Clayton Christensen

Lembre-se de que em um ambiente de rede, APIs abertas e protocolos padrões vencem, mas isso não significa que a idéia de vantagem competitiva vá embora.

5. Dados são o novo “Intel inside”

A mais importante entre as futuras fontes de fechamento e vantagem competitiva serão os dados, seja através do aumento do retorno sobre dados gerados pelo usuário, sendo dono de um nome ou através de formatos de arquivo proprietários.

Em linhas de apache | php |
javascript | css | ...

Temos de pensar em cada uma das partes do sistema com mais cuidado.

Servidores Http

- Segurança das transações
- Segurança dos dados
- Escalabilidade

php

- Manutenção das sessões.
- Segurança dos dados.
- Boas praticas de desenvolvimento.(POO, MVC, PDO, SCA SDO,SPL e outras).
- Planejamento antes da codificação.

javascript

- Diminuição do uso do servidor.
- Segurança dos dados.
- Ambiente de aplicação.
- XHR.
- Atenção ao Multi Browser.

CSS

- Dominio sobre o layout(e não o contrário).
- Manipulação de componentes da DOM.
- Personalização por cliente facilitada.

Estudando o Ambiente

Toda aplicação web (sistemas http) necessitara de um browser para interpreta-la e renderiza-la.

Os principais:

- Firefox
- IE
- Safari

Performance browser baseada

em SunSpider JavaScript Benchmark

<http://webkit.org/perf/sunspider-0.9/sunspider.html>

- Firefox 3 Nightly (PGO Optimized): 7263.8ms
- Firefox 3 Nightly (02/25/2008 build): 8219.4ms
- Opera 9.5.9807 Beta: 10824.0ms
- Firefox 3 Beta 3: 16080.6ms
- Safari 3.0.4 Beta: 18012.6ms
- Firefox 2.0.0.12: 29376.4ms
- Internet Explorer 7: 72375.0ms



Performance baseada em acesso a sites populares

yahoo.com

msn.com

hi5.com

google.com

rapidshare.com

fotolog.net

facebook.com

live.com

youtube.com

microsoft.com

myspace.com

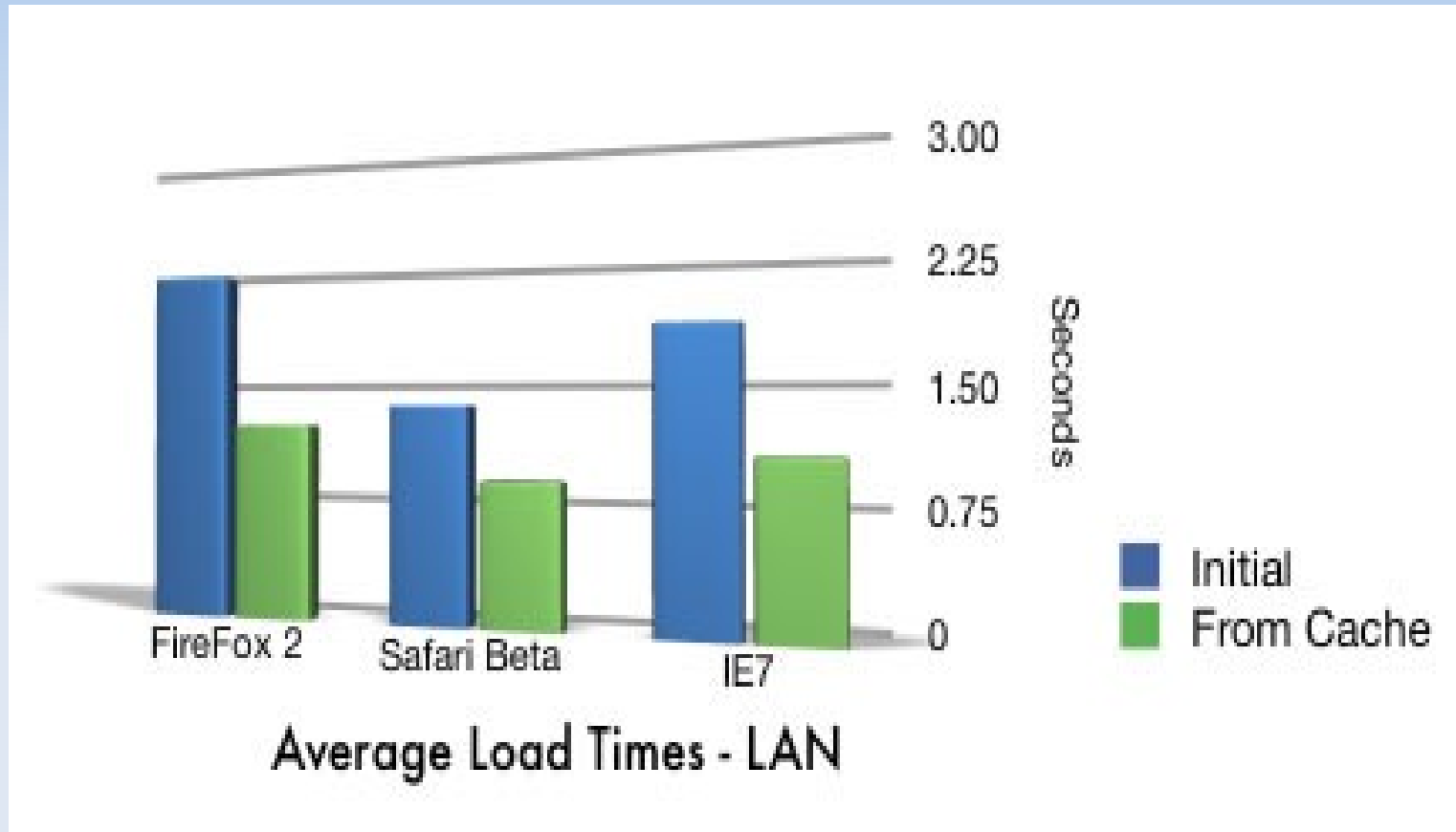
blogger.com

medaupload.com

wikipedia.org

friendster.com

orkut.com



XHR – 1a Preocupação

- **Dados** - fatos em sua forma primária.
- **Informação** - estruturas significantes com a intenção de gerar conhecimento no indivíduo e seu espaço.

XHR transporta dados e não informação.

O contexto da aplicação gera a informação.

XHR (retorno)

Caso 1)

```
Retorno = "<div  
id='titulo'>Exemplo 1 </div>  
<div id='corpo'>  
Este e o corpo de um exemplo  
de retorno XHR ja formatado.  
</div>";
```

```
var ret =  
Retorno.responseText;  
$('texto').innerHTML = ret;
```

Caso 2)

```
Retorno = " titulo:'Exemplo 2',  
corpo:'Este e o corpo de um  
exemplo de XHR sem  
formatacao'";
```

```
var ret = Retorno.responseJSON;  
$('titulo').innerHTML = ret.titulo;  
$('corpo').innerHTML = ret.corpo;  
document.title = "Texto sobre: "+  
ret.titulo;
```

Renderização – 2a Preocupação

Inserir objetos na página via html e via DOM diretamente.

```
With( espaço.appendChild(document.createElement('ID')) ){  
    title=rec.titulo;  
    innerHTML=rec.corpo;  
}  
ou
```

```
rec = "<div title='"+rec.titulo+"'> "+rec.corpo+"</div>";  
espaço.innerHTML = rec;
```

Sincronismo – 3a Preocupação

Partes XHR da Aplicação precisam estar serializadas, executando uma ação depois de outra.

Asynchronous | Synchronous

Código manutenível – 4a

Preocupação

A evolução:

- a) Coleção de **functions espalhadas** pelo código da pagina.
- b) Coleção de **functions separadas** em arquivos(bibliotecas).
- c) Coleção de **classes** bem definidas.

Prototype

Todo Objeto Javascript implementa prototype, que permite extensão do objeto.

```
String.prototype.isIvoNome = function(){  
    if (this=='Ivo')  
        return true  
    else  
        return false;  
}  
var a = "Ivo";  
if(a.isIvoNome()) alert('Foi escrito Ivo');
```

Json Syntax

```
var modelo = {  
  nome :",  
  setNome : function(n){  
    this.nome = n;  
  },  
  getNome:function(){  
    return this.nome;  
  }  
}  
m = modelo;  
m.setNome('TR');  
alert(m.getNome());
```

Provendo conexão a outros sistemas.

Webservice
API's Javascript
RSS