



<http://shevdon.com/before-you-hit-save>

DOCUMENTOS

Documentos



- Aplicações necessitam formatos comuns para a troca de dados
 - ▣ Formatos adequados ao caso de utilização
- 3 famílias de formatos possuem uma grande popularidade
 - ▣ CSV
 - ▣ JSON
 - ▣ XML



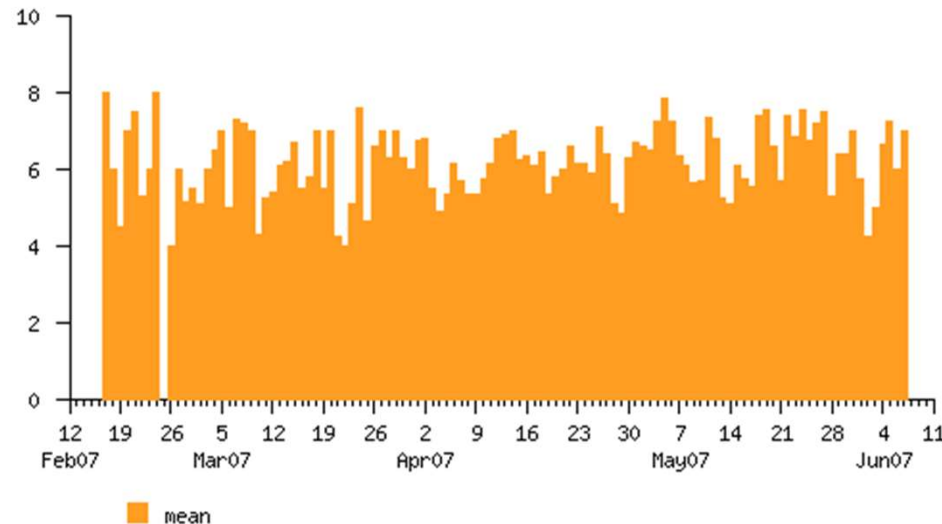
CSV

Comma Separated Values

or (Character Separated Values)

CSV: Comma Separated Value

- Representa séries de valores tabulares
- Origem nos anos 60-70
 - ▣ Muito comum nos dias de hoje
- Formato textual
 - ▣ Fácil de interpretar por humanos



CSV: Comma Separated Value



- Uma série de valores por linha
 - ▣ Valores na mesma linha estão relacionados
- Utiliza um carácter para separar valores
 - ▣ Tipicamente: ,
 - ▣ Outros: ; | <tab> <space>
- Pode possuir um cabeçalho indicando o nome dos campos

CSV: Exemplo



id,time,timestamp,temperature ← **Cabeçalho**

1,15/03/2014 18:07:24,1394903244.0,2.3

1,15/03/2014 18:08:24,1394903304.0,1.8

1,15/03/2014 18:09:24,1394903364.0,1.2

1,15/03/2014 18:10:24,1394903424.0,1.6

CSV: Exemplo

id,time,timestamp,temperature ← **Cabeçalho**

1,15/03/2014 18:07:24,1394903244.0,2.3

1,15/03/2014 18:08:24,1394903304.0,1.8

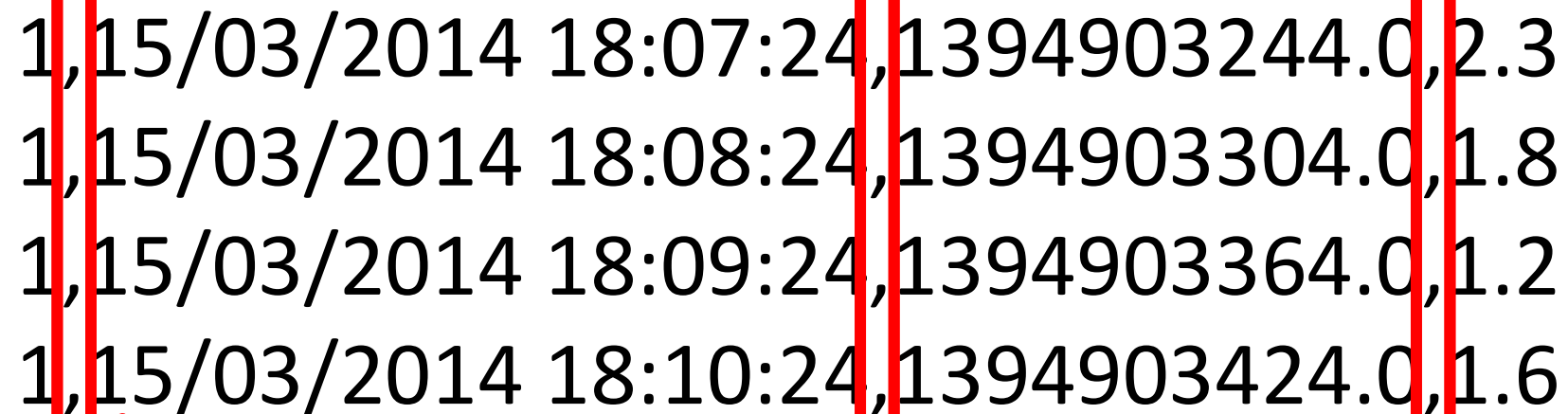
1,15/03/2014 18:09:24,1394903364.0,1.2

1,15/03/2014 18:10:24,1394903424.0,1.6

Séries de Valores

CSV: Exemplo

id,time,timestamp,temperature ← **Cabeçalho**



1,15/03/2014 18:07:24,1394903244.0,2.3
1,15/03/2014 18:08:24,1394903304.0,1.8
1,15/03/2014 18:09:24,1394903364.0,1.2
1,15/03/2014 18:10:24,1394903424.0,1.6

Separador de Valores

CSV: Colisão de Separador

- Usando Português, como codificar valores reais?

- Ex: 2,3

1,15/03/2014 18:07:24,1394903244,2,3

- Solução:

- Usar outro delimitador

- Limitar campos por aspas

1;15/03/2014 18:07:24;1394903244;2,3

1,15/03/2014 18:07:24,1394903244,"2,3"

CSV: Espaços e Colisão com NL

- Usar aspas:
 - ▣ Se o valor possuir \n (New Line)
 - ▣ Se o valor possuir espaços nos extremos

1,"DETI

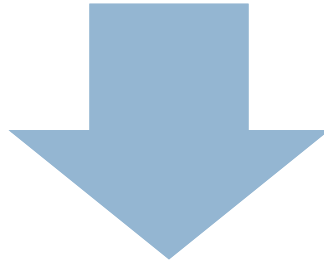
UA",3810,Aveiro

1," DETI\n UA\n ",3810,Aveiro

CSV: Aspas

- Usar aspas duplas caso existam aspas dentro dos valores

1, "DETI "LABI"", UA, Aveiro



1, "DETI ""LABI""", UA, Aveiro



JSON

JavaScript Object Notation

JSON: JavaScript Object Notation



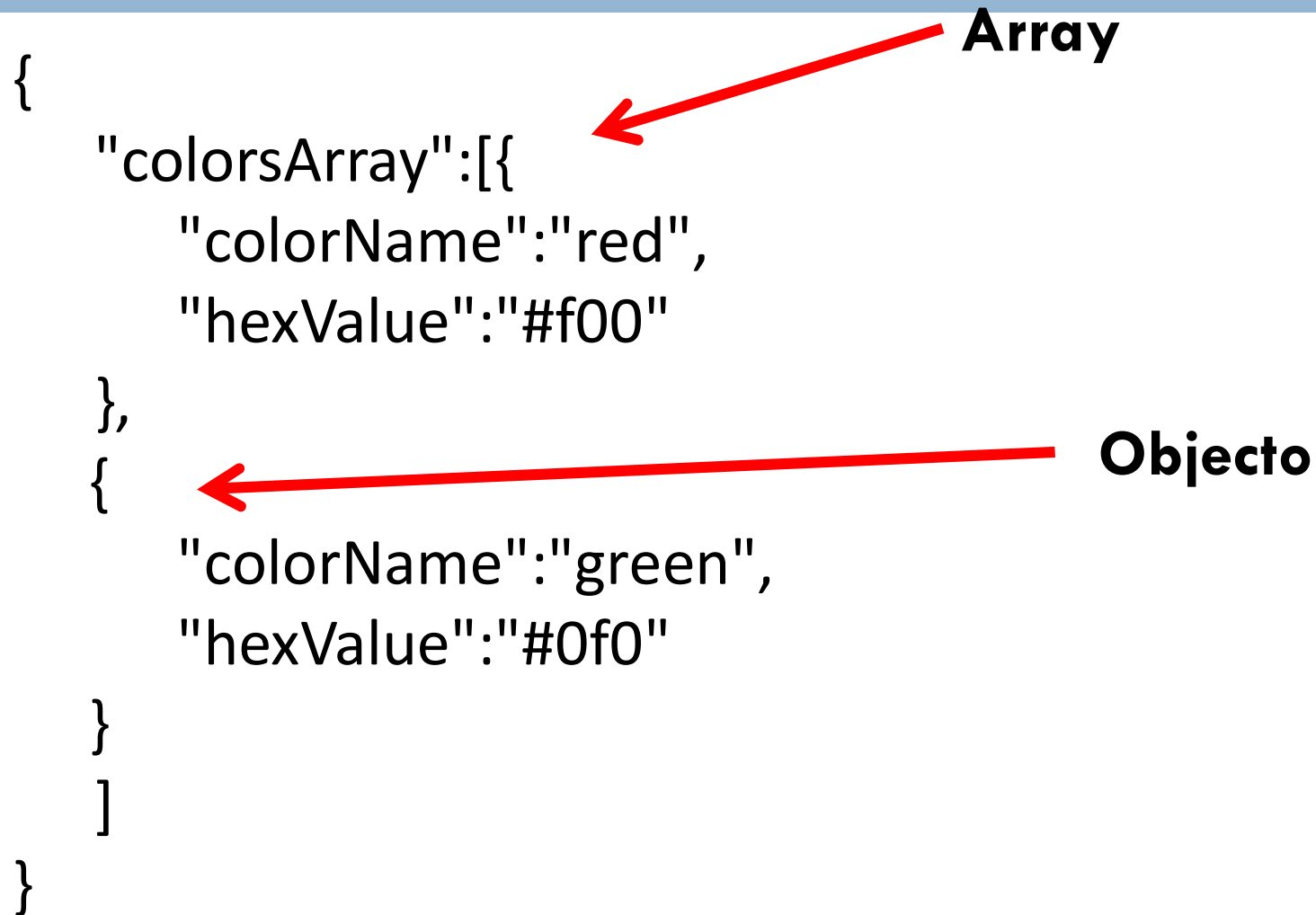
- Formato originário na ling. JavaScript
 - ▣ Evoluiu para um formato independente
- Representação textual
 - ▣ Facilmente interpretada/gerada por humanos
- Muito utilizado na comunicação entre aplicações
 - ▣ Especialmente Aplicações Web
 - ▣ Também utilizado em documentos

JSON

- Baseado em pares Chave : Valor
 - ▣ Dicionário em Python
- Chave: um identificador (String), delimitador por aspas (")
- Valor: Número, String, Array, Boolean, Outro Objecto

```
{  
  "time" : 1394984189,  
  "name" : "cpu",  
  "value": 12  
}
```

JSON



The diagram illustrates a JSON structure with two annotations:

- Array**: A red arrow points to the opening square bracket of the `colorsArray` value.
- Objecto**: A red arrow points to the opening curly brace of the second object in the array.

```
{  
  "colorsArray": [  
    {"colorName": "red",  
     "hexValue": "#f00"  
    },  
    {"colorName": "green",  
     "hexValue": "#0f0"  
    }  
  ]  
}
```

JSON



- Menos compacto do que CSV
 - ▣ Devido às chaves, aspas e outros caracteres
- Mais estruturado do que CSV
 - ▣ Cada campo é identificado
 - ▣ Estrutura hierárquica e não tabular
 - Conceito de lista e dicionário



XML

Extended Markup Language

XML: Extended Markup Language

- Formato baseado nas tecnologias Web
 - ▣ Plataforma genérica para outros formatos

- Possui muitos dialectos
 - ▣ **HTML**: Páginas Web (HTML5 já não é XML!)
 - ▣ **MathML**: Representação de fórmulas matemáticas
 - ▣ **ODF**: Documentos Open Office
 - ▣ **OpenXML**: Documentos Microsoft Office
 - ▣ Etc..

XML: Extended Markup Language



- Baseia-se em Marcas e atributos
 - ▣ Marcas têm de ser terminadas
- Estrutura Hierárquica
 - ▣ Marcas dentro de marcas

```
<marca atributo="valor">  
    conteúdo  
</marca>
```

Documento XML

- Iniciado por um identificador da versão e codificação
- Pode conter outros metadados (DOCTYPE)
- Possui um elemento raíz

```
<?xml version="1.0" encoding="utf-8">
```

```
<fridge>
```

```
  <reading time="1394903244">
```

```
    <id>1</id>
```

```
    <temperature>2.3</temperature>
```

```
  </reading>
```

```
</fridge>
```

XML Schema e DTD

- XML Schema define estrutura de um documento
 - ▣ Que marcas podem ser utilizadas, onde e como
- Documentos indicam qual a Schema aplicável
 - ▣ HTML usa um formato mais simples: DTD
- Possível validar documentos antes do processamento
 - ▣ Detectar erros de construção/transmissão
 - ▣ Evita processar documentos corrompidos

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

XML vs JSON vs CSV

XML = 157 carateres

```
<?xml version="1.0" encoding="utf-8">
<fridge>
  <reading time="1394903244">
    <id>1</id>
    <temperature>2.3</temperature>
  </reading>
</fridge>
```

JSON = 97 carateres

```
{"fridge":[
  {
    "time":1394903244,
    "id":1,
    "temperature":2.3
  }
]}
```

CSV = 17 carateres

```
1,1394903244,2.3
```