

Introdução ao XML

Neste tutorial pretendo apresentar o XML de forma simples e precisa. Qualquer programador que tenha um conhecimento básico de HTML compreenderá facilmente esta linguagem por ser baseada em SGML, linguagem que deu origem ao HTML.

O que é?

Você com certeza ouviu falar em XML (Extensible Markup Language) uma tecnologia muito presente hoje em nosso dia-dia. XML é uma forma de representação de documentos estruturados.

Como Surgiu? E qual o Objetivo?

XML foi criado pela W3C (Word Wide Web Consortium) em meados de 1998. O Objetivo do XML era ser uma linguagem boa para utilização na Internet, suportar grande variedades de aplicações, busca por multi-plataforma, compatível com SGML, ser fácil escrever programas que processem documentos XML, legíveis ao ser humanos e razoavelmente claro, os documentos deveriam ser fáceis de serem criados. Então, você pode substituir seus **arquivos de configuração e de comunicação**, que normalmente são binários, incompreensíveis ao ser humano, por arquivos XML. Todas as linguagens dão amplo suporte a leitura de um arquivo desses, e sua forma estruturada e hierárquica facilitam mais ainda. Para que outro programa ou outra pessoa leia o seu XML, basta você falar para ela (existe um arquivo para descrição e validação de XML, chamado DTD, fora do escopo deste tutorial) como é a "cara" do seu XML e o que cada tag indica! Você não precisa mais explicar complicados formatos binário para seus clientes!

Markup Language?

Markup Language ou Linguagens de marca são linguagens onde a sintaxe é baseada em marcas (tags). A Grande vantagem de linguagens de marcas é que são razoavelmente simples e de fácil entendimento. Elas não são linguagens de programação! São uma forma de você ter um texto e marcá-lo, indetificando alguma coisa, ou estruturando este texto.

XML vs HTML.

Ao contrário de que muitos pensam XML não é um substituto do HTML. XML contém informação, HTML exibe a informação. Muitas pessoas tem um XML, e depois que passam um parser ou um transformador geram um HTML.

Isso é muito bom, pois quando você precisa modificar o seu layout, basta modificar esse processo!

Onde aplicar?

XML pode ser a solução para guardar dados de uma pesquisa no banco de dados, guardar dados simples como configurações de um software, guardar textos ou em qualquer situação onde precisamos guardar dados de forma bem estruturada.

Flexibilidade.

As tags XML podem ser criadas por você mesmo, não mantendo um padrão como por exemplo em HTML onde as tags são limitadas. O exemplo abaixo mostra a estrutura básica de um documento XML.

Estrutura de um documento XML.

[view plainprint?](#)

```
1. <?xml version="1.0"?>
2. <!-- Criamos a tag programacao -->
3. <programacao>
4.
5. <!-- Inserimos o dado nome em programacao -->
6. <nome>Java</nome>
7.
8. <!-- Inserimos o dado tipo -->
9. <tipo>00</tipo>
10.
11. <!-- Inserimos o dado nota -->
12. <nota>10</nota>
13.
14. <!-- Fechamos a tag de dados programacao -->
15. </programacao>
```

Declaração XML.

A declaração da tag é importante pois mostra que é um documento XML e a versão em que foi escrita podendo também declarar outros atributos que podem ser importante para o formatador na leitura do documento.

Elementos.

Elementos XML inicia-se com ??. XML assim como Sistemas Unix, XML diferencia letras maiúsculas de minúsculas, ou seja:

[view plainprint?](#)

1. `<dado>` é diferente de `<data>`

Se criarmos uma tag e fechá-la com nosso documento possuirá erros e não será interpretado corretamente por um formatador. XML é muito sensível portanto devemos ser cauteloso na declaração.

[view plainprint?](#)

1. `<autor>`
2. `<nome> Ademir </nome>`
3. `<sobrenome> Constantino Filho </sobrenome>`
4. `</autor>`

Se sobrepormos uma tag o documento possuirá erros:

[view plainprint?](#)

1. `<autor>`
2. `<nome> Ademir`
3. `<sobrenome> Constantino Filho </nome>`
4. `</sobrenome>`
5. `</autor>`

Repare como a estrutura está "quebrada".

Comentários.

[view plainprint?](#)

1. `<!-- Comentários em XML similar a comentários em HTML -->`

Os comentários iniciam-se com `<!--` e terminam com `-->`. Os comentários podem conter qualquer dado, pode-se comentar qualquer declaração em seu documento o que torna muito útil caso este documento seja mantido por outro programador ou até mesmo observações.

Hierarquia.

A hierarquia é baseada na semântica ou a estrutura lógica do documento:

[view plainprint?](#)

1. `<?xml version="1.0"?>`
2. `<bebidas>`
3. `<suco>`
4. `<suco-natural>`
5. `<laranja> Suco de Laranja </laranja>`
6. `<abacaxi> Suco de Abacaxi </abacaxi>`

```
7.         </suco-natural>
8.     </suco>
9. </bebidas>
```

Raiz

Toda hierarquia possui uma raiz, a raiz é o início da hierarquia e geralmente uma abstração. No documento apresentado anteriormente, a raiz é bebidas.

Atributos.

Tags XML podem conter atributos porém diferente de elementos não possuem sub-atributos ou outros elementos. Na declaração de atributos devemos dar seu nome e valor:

[view plainprint?](#)

```
1. <tagxml valor="1"> Ok </tagxml>
```

Podemos também declarar mais de um atributo à uma única tag:

[view plainprint?](#)

```
1. <tagxml valor="1" valor2="1"> Ok </tagxml>
```

[view plainprint?](#)

```
1. <?xml version="1.0"?>
2. <artigos>
3.     <xml>
4.         <nome idioma="pt-br"> Introducao ao XML </nome>
5.         <nome idioma="ing"> XML Introduction </nome>
6.         <nome idioma="es"> Introduccion de XML </nome>
7.         <nome idioma="al"> XML Einfuhrung </nome>
8.         <nome idioma="it"> Introduzione di XML </nome>
9.     </xml>
10. </artigos>
```

Elementos Vazios.

Elementos vazios possuem uma sintaxe modificada. Os elementos vazios são utilizados para marcar uma determinada ação. O Exemplo abaixo mostra a ação do produto esgotado no estoque:

[view plainprint?](#)

```
1. <?xml version="1.0"?>
2. <produtos>
3.     <informatica>
4.         <floppy-drive>
5.             <preco>30</preco>
```

6. <fabricante>nek</fabricante>
7. <esgotado/>
8. </floppy-drive>
9. </informatica>
10. </produtos>

Conclusão

Esperamos que agora você não tenha mais aquele preconceito com XML, e muito menos medo. Você já está apto a criar seus arquivos de configuração e comunicação utilizando XML, ao invés de arquivos binários e de properties. No próprio GUJ existem tutoriais de como parsear um arquivo XML. Chegamos ao fim da Introdução ao XML. Bons estudos e até a próxima.