ANT

Rogério Napoleão Júnior (rogerionj@gmail.com)

Agenda

- Histórico
- O que é?
- Tarefas disponíveis
- Conceitos Básicos
- Elementos de um buildfile
- Exemplos / Exercícios

Histórico

- Criado como parte do projeto Tomcat
- Doado ao projeto Apache Software Foundation
- Separado em janeiro de 2000
- Release oficial 1.1 em 2000
- Versão atual 1.7.0

O que é?

- Ferramenta de build semelhante ao make com suporte especial a Java
- Escrito em Java
- Independente de plataforma
- Usado para automatizar tarefas repetitivas e complicadas.
- Ações controladas via arquivos XML.

Tasks do Ant

- Checksum, chmod, concat, copy, delete, filter, FixCRLF, get, mkdir, move, patch
- Bzip2, cab, Ear, gzip, jar, Rpm, Jlink, SignJar, tar, war, zip
- Depend, javac, JspC, RmiC, WljspC
- ServerDeploy, Javadoc, EJB

Tasks do Ant (cont)

- Exec, Java, Parallel, Sequential, Sleep, Waitfor
- Echo, mail, sql, taskdef, tstamp, ftp, telnet
- Cvs, ClearCase, VSS
- Junit, Testlet

Manual ANT

- http://ant.apache.org/manual/index.html

Elementos de um buildfile

- O nome default para um buildfile é build.xml
- O elemento raiz deve ser project
  - O atributo ‘default’ é obrigatório
    - Especifica o alvo default a ser executado
  - Outros atributos estão disponíveos (name, basedir)
- Alvos (target) contém uma ou mais tarefas (tasks)
  - O atributo ‘name’ é obrigatório
- Tarefas são a menor unidade de trabalho
  - A tarefas do Ant estão localizadas no arquivo ant.jar ou no classpath do Ant

Elementos de um buildfile (cont)

- Existe apenas um elemento ‘project’ por buildfile
- Pode existir tantos alvos (targets) quanto necessário (pelo menos um)
- Cada alvo (target) pode conter quantas tarefas (tasks) forem necessárias
- Comentários são permitidos <!--comentário-->
- Propriedades podem ser setadas ou incluídas de um arquivo.

Conceitos Básicos

- Build – processo de compilação e montagem de uma aplicação com o objetivo de gerar uma versão executável.
- Buildfile – scripts XML a partir dos quais o Ant executa os builds do sistema.
- Alvos – são objetivos a serem alcançados durante um build.
- Tarefas – atividades a serem realizadas durante o build.
- Propriedade – são uma forma de tornar o buildfile parametrizável.

Hello World


```
<project default="hello">
   <target name="hello">
        <echo message="Hello, World"/>
   </target>
</project>
```

Elemento Project

Atributo	Descrição
name	Nome do projeto
default	O alvo(target) default a ser executado quando nenhum alvo (target) é especificado
basedir	O diretório base onde todos os cálculos de caminho são feitos.

Elemento Target

Atributo	Descrição
name	Nome do alvo
depends	Uma lista de alvos, separados por vírgula que este alvo depende.
if	O nome da propriedade que precisa estar setada para que este alvo seja executado.
unless	O nome da propriedade que não deve estar setada para que este alvo seja executado.
description	Uma pequena descrição da funcionalidade deste alvo.

Elemento Task

- Um pedaço de código que pode ser executado
- Todas as tarefas (tasks) tem uma estrutura em comum:
  <nome atributo1=“valor1“ attirubo2=“valor2" ... />
  - atributoN é o nome do enésimo atributo
  - valorN é o valor do enésimo atributo
- O Ant vem com uma série de tarefas (tasks) básicas pré-definidas e tarefas (tasks) opcionais
  - Uma lista completa pode ser encontrada em http://ant.apache.org/manual

Propriedades

- Um par de nome e valor
- Imutáveis quando setados
- Exemplo

  <property name="src" value="C:\fontes" />

  <property file="build.properties" />

Exemplo 1

Hello.java:

```
public class Hello {
    public static void main( String[] args )
    {
        System.out.println( "Hello World" );
    }
}
```

Exemplo 1

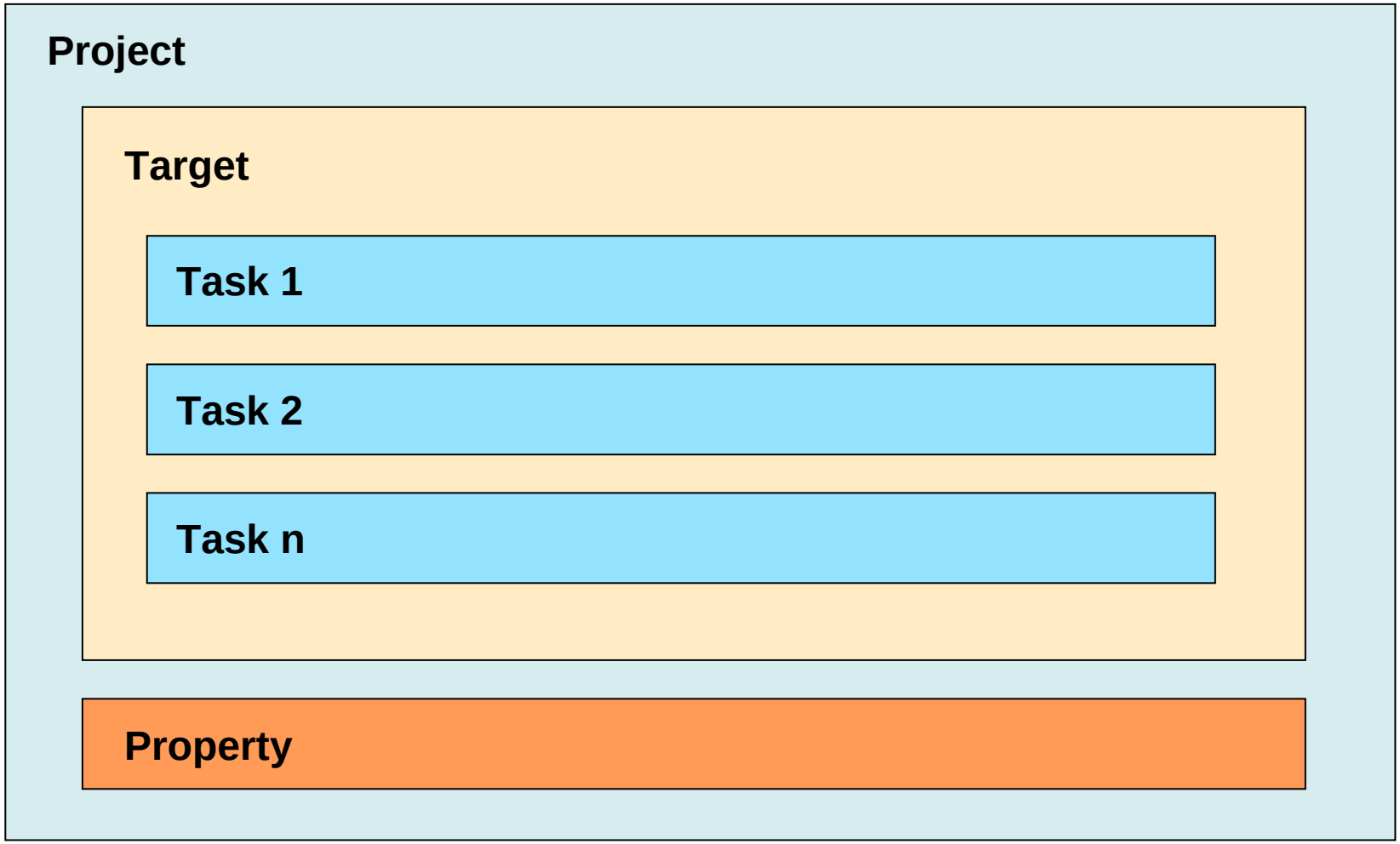
build.xml

```
<project name=“HelloWorld” default="compile">
  <target name="compile">
    <javac srcdir=“.” />
  </target>
  <target name="jar" depends="compile">
     <jar destfile=“hello.jar” basedir=“.”
          includes=“**/*.class” />
  </target>
</project>
```

Exemplo 1

- Executar o exemplo através da ferramenta ant do eclipse

Estrutura



Tipos

- **FileSet**
  - É um agrupamento de arquivos, esses arquivos são encontrados em um diretório base e são identificados conforme os padrões estabelecidos.
- **PatternSets**
  - Filtros em um conjunto de arquivos, pode ser referenciado em mais de um ponto por um id.
- **Selectors**
  - São critérios aplicados em um FileSet para selecionar os arquivos.

FileSet

Diretório base

```
<fileset dir="${dir}" casesensitive="yes">
  <include name="**/*.java"/>
  <exclude name="**/*Test*"/>
</fileset>
```

PatternSet

PatternSet


```
<patternset id="identificador">
  <include name="**/*.java"/>
  <exclude name="**/*Test*"/>
</patternset>
```

Selectors


```
<fileset dir="${doc.dir}" includes="**/*.html">
    <contains text="script" casesensitive="no"/>
</fileset>
```

Tasks

- BuildNumber
- Concat
- Copy/Move
- Jar
- Javac
- Javadoc
- Manifest
- Record

Exercício 1

- Criar uma pasta chamada WEB em um projeto no eclipse (Manual)
- Criar arquivos html dentro da pasta WEB (Manual)
- Fazer um target para criar um diretório através do ant chamado “Sequencial” (Ant)
- Fazer um target que compacte (zip) os arquivos html (Ant)
- Copiar o arquivo .zip para nossa pasta criada (Ant)
- Usar targets com “depends”
- Utilizar arquivo properties em todas as possíveis variantes

Exercício 2

- Criar um novo projeto no eclipse chamado Exercício 2 (Manual)
- Criar uma pasta chamada “Arquivos”, nela deve ser criado 2 arquivos .xml, 3 arquivos .html, 2 arquivos .jsp, 1 arquivo.sql e 2 arquivos .tmp (Manual)
- Faça targets que criem os seguintes diretórios web, conf, java, doc
- Após isso copie os arquivos .html para pasta web, .xml para pasta conf, .jsp para java e .sql para doc. Ignore os arquivos .tmp
- Compacte a pasta Arquivos sem os arquivos .tmp
- Crie uma pasta escrito backup e copie para dentro dela a pasta Arquivos compactada
- Crie uma target para apagar todas as pastas web, conf, java, doc e backup a criacao das pasta no inicio deve depender deste target.
- Usar build.properties

Exercício 3

- Criar um novo projeto com classes java, essas classes java deve conter Javadoc
- Criar uma target para criar um diretorio doc
- Com ANT gere o javadoc das classes

Exemplo 2 – Criar base

- Exemplo task:

```
<target name="criar.base">
  <sql driver="org.postgresql.Driver" password="admin"
     url="jdbc:postgresql://localhost:5432/postgres" userid="postgres"
     src="doc/Script.sql" classpathref="lib.banco" />
  </target>
```

- Path Reference

```
<path id="lib.banco">
     <fileset dir="lib">
          <include name="org.postgresql.jar" />
     </fileset>
</path>
```

- É necessário ter toda estrutura do ant, lib do postgre e script SQL

Requisitos – Exemplo 2

- Instalar PostgreSQL e PGAdmin
- Criar script SQL
- www.w3schools.com.br (Learn SQL)
- Colocar a lib do postgresql no projeto