Componentes Web J2EE

Helder da Rocha (helder@acm.org)
www.argonavis.com.br

Sobre este módulo

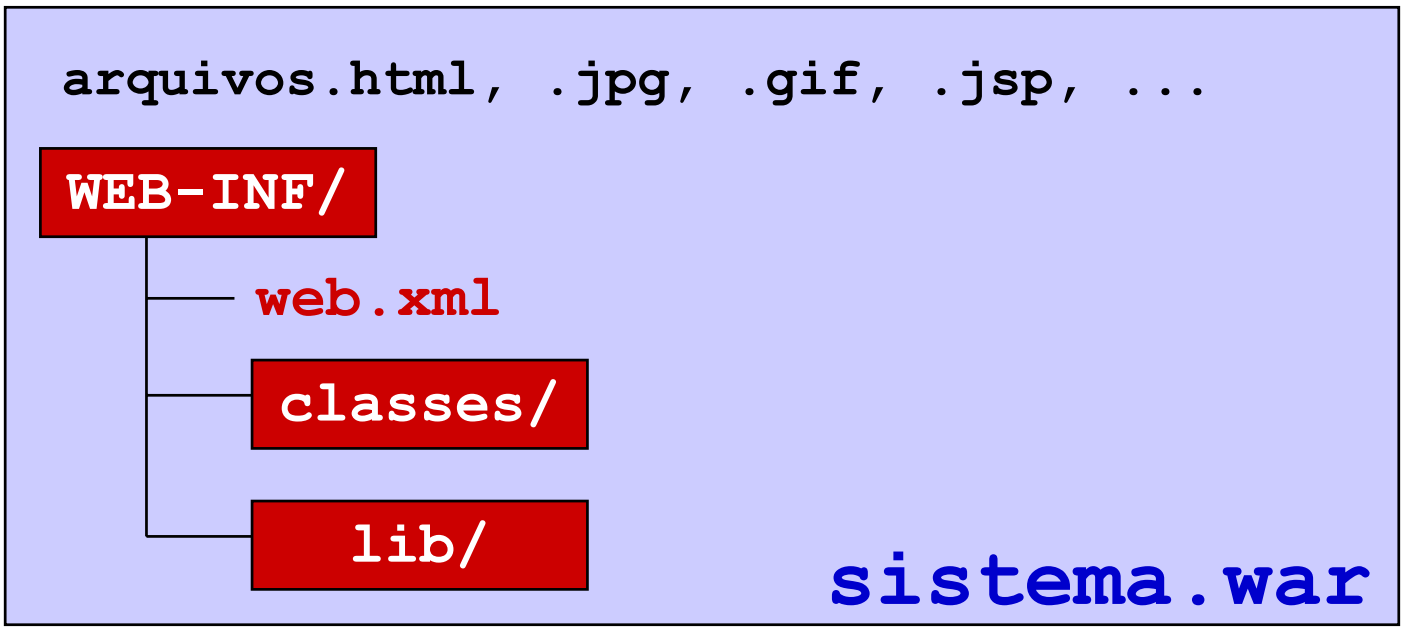
- Neste módulo conheceremos uma nova maneira de fazer deployment: arquivos WAR
  - Com arquivos WAR pode-se colocar uma aplicação no ar, em muitos casos, simplesmente copiando o arquivo para um determinado local do servidor
- A criação de arquivos WAR é basicamente uma tarefa de copiar e compactar
  - Usaremos o Ant para implantar as operações
- Reimplantar um arquivo WAR é simples no JBoss, mas é trabalhoso no Tomcat
  - Requer a remoção do WAR e do diretório criado
  - Requer o reinício do servidor

  por este motivo, continuaremos a copiar os arquivos para o servidor nos próximos módulos, apesar de criar o WAR

Web Archive

- *Utilizável no Tomcat e também em servidores J2EE*
- *Permite criação de novo contexto automaticamente*
- *Coloque JAR contendo estrutura de um contexto no diretório de deployment (webapps, no Tomcat)*
  - *O JAR deve ter a extensão .WAR*
  - *O JAR deve conter WEB-INF/web.xml válido*

Exemplo - aplicação: `http://servidor/sistema/`



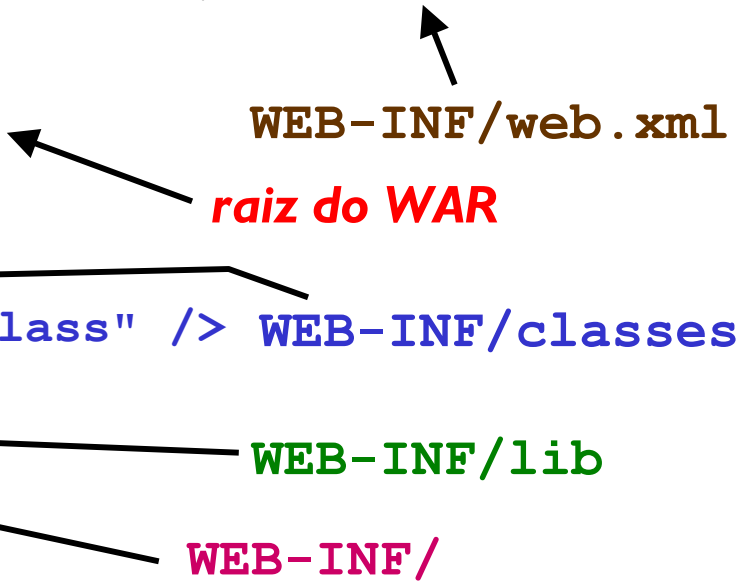
Como criar um WAR

- O mesmo WAR que serve para o Tomcat, serve para o JBoss, Weblogic, WebSphere, etc.
  - Todos aderem à mesma especificação
- Há várias formas de criar um WAR
  - Usando o deploytool do Tomcat.
  - Usando um aplicativo tipo WinZip
  - Usando uma ferramenta JAR:
    jar -cf arquivo.war -C diretorio_base .
  - Usando a ferramenta packager do kit J2EE:
    packager -webArchive <opções>
  - Usando a tarefa <jar> ou <war> no Ant

WAR criado pelo Ant

- Pode-se criar WARs usando a tarefa <jar> ou <war>
  - Com <jar> você precisa explicitamente definir seus diretórios WEB-INF, classes e lib (usando um <zipfileset>, por exemplo) e copiar os arquivo web.xml, suas classes e libs.
  - Com <war> você pode usar o atributo webxml, que já coloca o arquivo web.xml no lugar certo, e outros elementos de um war:

```
<war warfile="bookstore.war" webxml="etc/metainf.xml">
   <fileset dir="web" >
      <include name="*.html />
      <include name="*.jsp" />
   </fileset>
   <classes dir="${build}" >
      <include name="database/*.class" />
   </classes>
   <lib dir="${lib.dir}" />
   <webinf dir="${etc.dir}" />
</war>
```



  - Veja o manual do Ant para outros exemplos e detalhes

Configuração externa do WAR (servidores J2EE)

- Configuração externa ao WAR pode ser feita quando WAR é acrescentado em um arquivo EAR e funciona em servidores J2EE (JBoss, por exemplo)
  - EAR é JAR comum com arquivo application.xml no seu META-INF
- O arquivo application.xml do EAR deve conter

```
<application>
  <module>
    <web>
      <web-uri>mywebapp.war</web-uri>
      <context-root>/myroot</context-root>
    </web>
  </module>
</application>
```

- A aplicação agora é acessada via

```
http://servidor/myroot
```

Deployment e execução

- Depende do servidor
  - No JBoss, copie o WAR ou EAR para o diretório deploy do servidor para implantar o serviço. Remova o WAR para tirar o serviço do ar
  - No Tomcat, copie o WAR para o diretorio webapps e reinicie o servidor. Para remover o serviço, desligue o Tomcat, apague o diretório criado e o WAR.
- Para executar, as regras são as mesmas que uma aplicação comum. Acesse o contexto raiz via Web
  - http://servidor/nome-do-contexto/
  - http://servidor/nome-do-contexto/index.jsp
  - http://servidor/nome-do-contexto/subcontexto/aplicacao
  - http://servidor/nome-do-contexto/servlet/pacote.Classe

Exercícios

- *1. Crie um alvo no Ant que gere WARs para todas as aplicações que você já criou até o momento*
- *2. Crie outro alvo que faça o deployment automático, copiando o WAR para o diretório correto*
  - *Guarde as propriedades de deployment em um arquivo externo build.properties para fácil alteração*
- *3. Crie um alvo que faça o undeploy*
  - *Remova o diretório (com mesmo nome de contexto)*
  - *Remova o WAR*

Esta última tarefa pode falhar caso o Tomcat esteja no ar. Será preciso pará-lo antes (mas não crie uma tarefa para isto)

helder@acm.org

argonavis.com.br