1. Apache PHP

1.1. Instalando o Apache2 e o PHP5 no Debian

Para instalar o Apache2 utilizaremos os comandos

```
apt-get update
apt-get install apache2 php5
```

Vamos realizar as principais configurações paraa o apache2, inserindo diretivas em arquivos de configuração em texto puro.

Para testar a instalação de ambos os softwares procederemos com o seguinte:

1. Crie o arquivo index.php

   nano /var/www/html/index.php

```
<?php
    phpinfo();
?>
```

2. Acesse o endereço http://localhost/index.php

1.2. Subdiretórios e arquivos do Apache2

Os arquivos e diretórios da instalação do Apache2, os quais se encontram em /etc/apache2 são:

apache2.conf → principal arquivo de configuração do servidor. Este arquivo servirá para configurações padrão e para ser o ponto comum em que o servidor acessará as configurações mais detalhadas;

ports.conf → esse arquivo é usado para especificar as portas em que os virtual hosts deverão "escutar";

conf.d/ → esse diretório é usado para controlar configurações específicas do Apache. Por exemplo, ele é bastante usado para definir configuração SSL e demais escolhas padrão de segurança;

sites-available/ → diretório que contém todos os arquivos de virtual hosts que definem web sites distintos. Aqui estão as configurações disponíveis e não as configurações ativas;

sites-enabled/ → diretório que estabelece quais definições de virtual hosts estão sendo de fato utilizadas, ou seja, ativas;

mods-enabled/ e *mods-available/* → estes diretórios são parecidos com os diretórios sites-available e sites-enabled, porém neles estão os módulos que podem ser carregados opcionalmente para cada um.

Exercício 1.1

Configure o timeout do Apache de forma que as requisições esperem ser respondidas em no máximo 2 minutos e meio.

1.2.1. Configurando a porta HTTP do servidor Apache2

A porta HTTP default do Apache2 é a 80. Para o caso de desejarmos mudar esta porta devemos editar o arquivo **`/etc/apache2/ports.conf`**, alterando a diretiva **Listen**. Vamos alterar a porta para 8081 (certifique-se de que não exista outro serviço previamente utilizando esta porta com o comando **`netstat -antu | grep 8081`**). Reinicie o serviço apache e tente acessar o localhost pelo navegador acrescentando a porta 8081

```
service apache2 restart
localhost:8081
```

1.2.2. Tunando a performance do Apache2

Um grande problema vivenciado por quem administra serviços web é o servidor de aplicação lotado com centenas ou até milhares de requisições simultâneas. Para diminuir a probabilidade de um servidor entrar em colapso por conta da grande quantidade de acessos, geralmente é necessário modificar a configuração original dos arquivos de configuração do apache.

Para iniciarmos a tunagem, é preciso primeiro, conhecermos os principais parâmetros de execução do servidor. Abra o arquivo /etc/apache2/apache2.conf e verifique os parâmetros a seguir:

Timeout → tempo máximo em segundos em que uma requisição deve ser respondida.

KeepAlive → se estiver permite que cada conexão permaneça aberta para receber múltiplas requisições do mesmo usuário. Pode ester On ou Off.

MaxKeepAliveRequests → quantidade de requisições individuais que cada conexão irá comportar antes de esta conexão ser passada para outro usuário. Deixar o valor 0 (zero), significa número ilimitado de requisições.

KeepAliveTimeout → tempo máximo de espera entre uma requisição e outra do usuário em uma mesma conexão.

MaxClients → quantidade de conexões/clientes simultâneos que o servidor pode atendeer sem que necessite utilizar swap.

StartServers → número de processos de servidor a serem criados logo no startup do Apache.

MinSpareServers → número mínimo de processos de processos de servidor filhos sobressalentes, ou seja, que não estão processando requisições.

MaxSpareServers → número máximo de processos de processos de servidor inativos, ou seja, que não estão processando requisições. Caso o número de processos filhos inativos ultrapasse o valor deste parâmetro, o processo pai os anula (kill).

Um fato que influencia muito na performance de um servidor web é a frequência em que ele faz uso do swap. Por isso devemos configurar o servidor baseado na quantidade de memória RAM.

Configurando o MaxClientes: procedemos com o seguinte cálculo:

$$\text{MaxClients} \approx \frac{\text{RAM total} - \text{memória de outros processos}\_\text{sem Apache}}{\text{média de memória utilizada pelo Apache}}$$

O valor de *MinSpareServers* e o *StartServers* podem ser configurados segundo a tabela a seguir:

Servidor virtual VPS	5
Servidor dedicado com 1 a 2GB RAM	10
Servidor dedicado com 2 a 4GB RAM	20
Servidor dedicado com mais de 4GB RAM	25

O valor de *MaxSpareServers* pode ser setado como o dobro do MinSpareServers.

O *KeepAlive* estando On economiza tempo e processamento, pois diminui a quantidade de sockets e novos processos criados. Contudo, pode consumir bastante memória.

Recomenda-se que o *MaxKeepAliveRequests* fique entre 100 e 150.

Exercício 1.2 Em um servidor Apache com os parâmetros a seguir, calcule o máximo de conexões que este servidor pode atender por minuto.

```
KeepAlive On
KeepAliveTimeout 1
MaxClients 20
StartServers 15
```

1.3. Configurando opções/permissões para as páginas

Ao abrir o apache2.conf você verá algumas linhas semelhantes as que vêm a seguir:

```
<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride None
        Request all granted
</Directory>
```

A diretiva <Directory> é uma das que existem para especificar opções e permissões para as páginas. As diretivas disponíveis são:

Directory → As restrição afetará o diretório no disco especificado, consequentemente a página armazenada nele. Exemplo:

```
<Directory /var/www>
```

```
Order deny,allow
```

```
     deny from all

     allow from 192.168.100.2

<Directory>
```

DirectoryMatch → Funciona como a diretiva <Directory> mas trabalha com expressões regulares como argumento. Exemplo:

```
     <DirectoryMatch "^/www/.*">
           Order deny,allow

          deny from all

     <DirectoryMatch>
```

Files → As restrições afetarão os arquivos do disco que confere com o especificado. É possível usar os coringas ? e * . Também podem ser usadas expressões regulares especificando um "~" após Files e antes da expressão. Exemplo 1:

```
     <Files *.txt>
          Order deny,allow
          deny from all
     </Files>
```

Exemplo 2:

```
     <Files ~ "\.(gif|jpe?g|bmp|png)$">
          Order deny,allow
          deny from all
     </Files>
```

FilesMatch → Permite usar expressões regulares na especificação de arquivos (equivalente a diretiva <Files ~ "expressão">). Exemplo:

```
     <FilesMatch "\.(gif|jpe?g|bmp|png)$">
               Order deny,allow
```

```
            deny from all
            allow from 192.168.100.2
    </FilesMatch>
```

Location→ As restrições afetarão o diretório base especificado na URL e seus sub-diretórios. Por exemplo:

```
  <Location /security>
      Order allow,deny
    </Location>
```

Bloqueia o acesso de todos os usuários ao diretório /security da URL

LocationMatch→ Idêntico a diretiva <Location> mas trabalha com expressões regulares. Por exemplo:

```
  <LocationMatch "/(extra|special)/data">
      Order deny,allow
      deny from all
    </LocationMatch>
```

Bloqueará URLs que contém a substring "/extra/data" ou "/special/data".

Obs: O uso das diretivas <Directory> e <Files> é apropriada quando você deseja trabalhar com permissões a nível de diretórios/arquivos no disco local (o controle do proxy também é feito via <Directory>), o uso da diretiva <Location> é adequado para trabalhar com permissões a nível de URL. A ordem de processamento das diretivas de acesso são processadas é a seguinte:

Normalmente é encontrado a opção *Options* dentro de uma das diretivas acima, a função desta diretiva é controlar os seguintes aspectos da listagem de diretórios:

- All

Todas as opções são usadas exceto a MultiViews. É a padrão caso a opção *Options* não seja especificada.

- ExecCGI

Permite a execução de scripts CGI.

- FollowSymLinks

O servidor seguirá links simbólicos neste diretório (o caminho não é modificado). Esta opção é ignorada caso apareça dentro das diretivas <Location>, <LocationMatch> e <DirectoryMatch>.

- Includes

É permitido o uso de includes no lado do servidor.

- IncludesNOEXEC

É permitido o uso de includes do lado do servidor, mas o comando #exec e #include de um script CGI são desativados.

- Indexes

Se não existir um arquivo especificado pela diretiva <DirectoryIndex> no diretório especificado, o servidor formatará automaticamente a listagem ao invés de gerar uma resposta de acesso negado.

- MultiViews

Permite o uso da Negociação de conteúdo naquele diretório. A negociação de conteúdo permite o envio de um documento no idioma requisitado pelo navegador do cliente.

- SymLinksIfOwnerMatch

O servidor somente seguirá links simbólicos se o arquivo ou diretório alvo tiver como dono o mesmo user ID do link. Esta opção é ignorada caso apareça dentro das diretivas <Location>, <LocationMatch> e <DirectoryMatch>.

2. Apache Tomcat

O Apache Tomcat é um web server e servlet container open source que implementa as especificações para Servlets Java, páginas JSP (Java Server Pages), Java Unified Expression Languages (JUEL) e o WebSocket Java, e provê um servidor web onde o código Java será executado. O Apache Tomcat é desenvolvido pela Apache Software Foundation (http://www.apache.org/).

Vamos proceder com a instalação do Tomcat no Debian/Ubuntu:

1. Primeiro verificar se o Oracle Java está instalado na máquina.:

```
java -version
```

Se o texto da imagem abaixo for exibido, significa que o Oracle Java já está instalado. Caso contrário este deverá ser instalado com os procedimento a seguir (instalabdo o Java versão 8):

1.1 Adicionando o repositório onde se encontra o Oracle Java 8[1]

```
apt-get install python-software-properties

su -

echo "deb
http://ppa.launchpad.net/webupd8team/java/ubuntu trusty
main" | tee
/etc/apt/sources.list.d/webupd8team-java.list

echo "deb-src
http://ppa.launchpad.net/webupd8team/java/ubuntu trusty
main" | tee -a
/etc/apt/sources.list.d/webupd8team-java.list

apt-key adv --keyserver hkp://keyserver.ubuntu.com:80
--recv-keys EEA14886
```

1.2 Realizando o apt update e instalando o Oracle Java 8

```
apt-get update

apt-get install oracle-java8-installer
```

2. Instalando o Tomcat:

2.1. Baixando e extraindo os pacotes Tomcat:

```
cd /opt
```

[1] Até certo tempo, o Java fazia parte das distribuições Linux, mas após mudança na licença do Oracle Java isso não é mais permitido. Por conta disso, é necessário adicionar o repositório do Oracle Java e baixá-lo manualmente. Para facilitar a instalação do Oracle Java (JDK) no Ubuntu o site Web Upd8 criou um repositório PPA.

```
wget
http://www.apache.org/dist/tomcat/tomcat-8/v8.0.versaoat
ual/bin/
apache-tomcat-8.0.versaoatual.tar.gz

tar -xvf apache-tomcat-8.0.versaoatual.tar.gz
```

2.2. Configurando a variável de ambiente CATALINA_HOME:

```
nano ~./bashrc
```

Acrecente o conteúdo

export CATALINA_HOME="/opt/apache-tomcat-8.0.versaoatual"

Salve e feche o arquivo ~/.bashrc e utilize o comando source para tornar as alterações válidas

```
source ~/.bashrc
```

2.3. Execute o Tomcat:

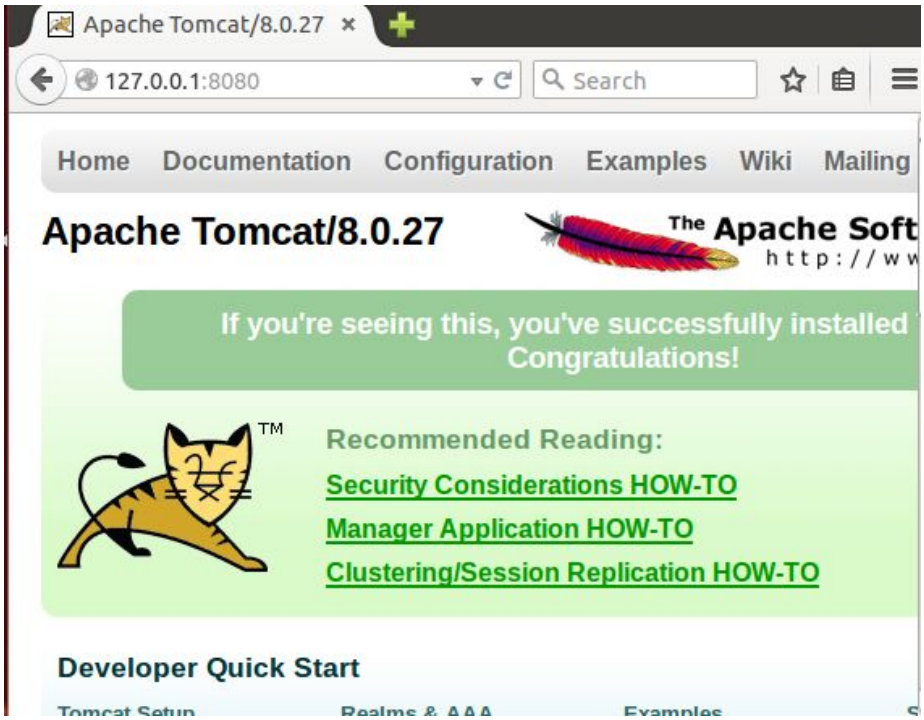
```
cd /opt/apache-tomcat-8.0.versaoatual/bin

./startup.sh
```

Se tudo estiver certo, você verá o texto a seguir

```
Using CATALINA_BASE:   /opt/apache-tomcat-8.0.27
Using CATALINA_HOME:   /opt/apache-tomcat-8.0.27
Using CATALINA_TMPDIR: /opt/apache-tomcat-8.0.27/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /opt/apache-tomcat-8.0.27/bin/bootstrap.
jar:/opt/apache-tomcat-8.0.27/bin/tomcat-juli.jar
Tomcat started.
```

6. Testando a instalação, acessando pelo navegador o endereço do servidor na porta 80. Ex: http://192.168.100.1:8080.



Parabéns, o Apache Tomcat foi instalado com sucesso.

Atenção! Para parar o serviço do Tomcat execute o script **`/opt/apache-tomcat-8.0.27/bin/.shutdown`**

No caso de reiniciarmos a máquina onde se encontra o Tomcat, verificamos que o mesmo não é startado. Para isso precisamos criar um arquivo que seja executado assim que a máquina for inicializada.

1. Criando o arquivo executável na pasta /etc/init.d
   `nano /etc/init.d/tomcat8`

2. Inserir o código a seguir
   case $1 in
   start)
   /opt/tomcat-latest/bin/startup.sh;;
   stop)
   /opt/tomcat-latest/bin/shutdown.sh;;
   esac
   exit 0

   Salve o arquivo

3. Execute os comandos a seguir:
   `chmod +x tomcat8`

   `service tomcat start`

   `update-rc.d tomcat8 defaults`

Diretórios e subdiretórios do Tomcat

No diretório /opt/tomcat/ estão todos os subdiretórios necessários para o funcionamento do Tomcat, os quais seguem:

/bin

Neste local encontram-se os arquivos binários do sistema, bem como o arquivo que permite parar e inicializar o sistema, chamado *catalina.sh*. Para executá-lo basta adicionar 2 parâmetros, um antes do nome do arquivo e outro indicando qual é a ação a ser realizada, se pára o serviço ou o inicia-o, conforme o comando: **`sh catalina.sh start`**

/lib

Todas as bibliotecas estão armazenadas neste local, podendo ser as bibliotecas das aplicações que são desenvolvidas ou bibliotecas do Tomcat, incluindo driver JDBC para conexão com banco de dados.

/logs

O Apache Tomcat registra todas as ações e execuções das aplicações. Neste diretório estão armazenados informações de logs, bem como, execução das aplicações, instalação, erros de acesso, requisição de conexão e entre outros.

/temp

Consiste em um subdiretório para alocar uso de arquivos temporários das aplicações.

/webapps

Considerando o clássico /var/www que o apache utiliza, o subdiretório /webapps possui o mesmo fundamento: Armazenar as aplicações que serão executadas pelo Tomcat. Por padrão, dentro do mesmo, há arquivos do próprio programa e alguns exemplos de jsp e servelets. Caso queira testá-los, acesse http://localhost:8080.

/conf

Diretório onde se encontram os arquivos de configuração, bem como configuração de usuário, do servidor e do componentes que serão executados. Utilizaremos alguns arquivos deste diretórios nas próximas subseções.

Configurando o Tomcat

1.1.1 Acesso de gerenciamento

A instalação do Tomcat traz consigo um ambiente para seu gerenciamento, o qual pode ser acessado pelo endereço http://localhost:8080/manager/html. Este ambiente, porém, tem seu acesso restrito por default. Para acessarmos o manager temos que adicionar usuários ao arquivo **`/opt/apache-tomcat-8.0.27/conf/tomcat-users.xml`**.

Os usuários adicionados devem possuir papéis (roles). Os papéis disponíveis no Apache Tomcat são:

- *manager-gui* – Tem acesso à página de Status do Servidor (Server Status) e ao Gerenciador Web (Manager App) baseado em HTML. O usuário que tem esse papel pode realizar tarefas como deploy e undeply de aplicações, ver status, detectar leaks de memória, expirar seções, entre outras.
- *manager-script* – possui todas as funcionalidades do manager-gui, porém a interface é em texto e não em HTML. Utilizado para quem prefere realizar as tarefas acima citadas em linha de comando.
- *manager-jmx* – acessa o proxy JMX que possui ferramentas e scripts de monitoramento. Também acessa a página de Status.
- *manager-status* – Acessa apenas a página de Status do Servidor (http://localhost:8080/manager/status)
- *admin-gui* – acessa o Host Manager no formato HTML. Não acessa o Manager App nem o Server Status.
- *admin-script* – acessa o Host Manager em formato texto.

Vamos criar um usuário aluno com acesso ao Manager App em formato HTML e ao Server Status. No final do arquivo tomcat-users.xml vamos adicionar as linhas abaixo

```
<role rolename="manager-gui"/>
<user username="aluno" password="ifpe" roles="manager-gui" />
```

No código acima, a primeira linha especifica qual o papel estará disponível para o servidor atual. A segunda linha cria um usuário "aluno" com senha "ifpe" e que tem o papel de manager-gui. Salve o arquivo tomcat-users.xml, pare e reinicialize o serviço do tomcat. Feche e abra o navegador, acesso tomcat e tente acessar o Manager App, o Status Server e o Host Manager.

Exercício 2.1

A. Crie um usuário "useradmin" com senha "souadmin", de forma que tenha permissão para acessar apenas o Manager App e ao Host Manager.

B. Altere o usuário "useradmin" para que ele tenha acesso ao Host Manager e ao Server Status.

1.1.2 Configurando a porta e o timeout de execução

Por default a porta HTTP onde o Tomcat roda é a 8080. É possível trocar a porta default por outra que esteja disponível, alterando a diretiva Connector do arquivo **/opt/apache-tomcat-8.0.26/conf/server.xml**, que diz respeito ao protocolo HTTP:

```
<Connector port= "8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

Além da porta, esta diretiva permite configurarmos o timeout de conexão (em milisegundos) e a porta de redirecionamento para HTTPS.

Exercício 2.2

A. Altere a porta de execução do Tomcat para 8090 (verifique antes se ela não está sendo usada) e altere o timeout de conexão para 45 segundos.

2. MySQL

1. Instale o mysql server e o mysql client

   ```
   apt-get install mysql-server mysql-client
   ```

   Obtendo suporte ao MySQL no PHP5.

   Instale os seguintes pacotes:

   ```
   apt-get install php5-mysql php5-curl php5-gd php5-intl
   php-pear php5-imagick php5-imap php5-mcrypt php5-memcache
   php5-ming php5-ps php5-pspell php5-recode php5-snmp
   php5-sqlite php5-tidy php5-xmlrpc php5-xsl
   ```

   ```
   service apache2 restart
   ```

   2. Instalando o PhpMyAdmin.

   ```
   apt-get install phpmyadmin
   ```

   Web server to reconfigure automatically: <– apache2

   Configure database for phpmyadmin with dbconfig-common? <– Não

Copie a pasta /usr/share/phpmyadmin para /var/www com o comando

```
cp -R /usr/share/phpmyadmin para /var/www/phpmyadmin
```

Após a instalação, você poderá acessar o PhpMyAdmin inserindo no navegador o ip do seu servidor e adicionar no final /phpmyadmin. No nosso caso http://192.168.100.1/phpmyadmin

3. Implantando projeto PHP no Servidor Debian

Tendo o PHP e o MySQL instalados e configurados no servidor, podemos agora implantar um projeto PHP previamente desenvolvido.

3.1 Transferindo os dados para o servidor

Há várias formas de se transferir arquivos para um servidor debian.

1. De um link da internet, via comando **wget**:
   Ex:

   Acessar /var/www ou /var/www/html com o comando
   `cd /var/www` ou **`cd /var/www/html`**

   Baixar os arquivos desejados
   `wget https://goo.gl/AfcCmX -O site_teste.zip`

   Como neste exemplo o arquivo é zipado proceder com o comando:
   `unzip site_teste.zip`

2. De outro computador via comando scp:

   scp -r endereco_computador_origem endere_cocomputador_destino
   Ex:

   `scp -r /home/aluno/Documentos/site_teste root@192.68.100.1:/var/www`

3.2 Criando as tabelas do projeto

Como o MySQL já foi instalado, podemos criar o banco e as tabelas necessárias ao projeto.
Após isso temos que configurar o projeto para acessar o banco no novo endereço.

4. Servidor FTP

O FTP (File Transfer Protocolo) é o protocolo de transferência de arquivo, que serve basicamente para que usuários possam enviar ou receber documentos para/da Internet. Essa transferência é sempre realizada entre um servidor e um cliente

Um servidor FTP é responsável por prover os mecanismos necessários para upload e download de arquivos e é onde estes arquivos ficam armazenados. O cliente que realiza o upload e/ou downloads de arquivos podem ser o navegador ou programas específicos de transferência de arquivos.

4.1 Instalando e Configurando um Servidor FTP no Linux (Debian)

O primeiro passo é a instalação de um software servidor de FTP. No nosso caso, instalaremos o proFTPd.

```
apt-get install proftpd
```

Escolher entre inetd ou autônomo (standalone) depende da previsão de tráfego de dados que o serviço receberá.

Agora iremos modificar o arquivo de configuração proftpd.conf, que se encontra no caminho **`/etc/proftpd/proftpd.conf`**

Lembre-se de que antes de realizar qualquer modificação em um arquivo de configuração é recomendado realizar uma cópia do arquivo original antes. Assim, para realizar a cópia do arquivo que iremos modificar utilize o comando a seguir:

```
cp /etc/propftpd/propftpd.conf
/etc/propftpd/propftpd.conf-original
```

Utilizar um editor para verficar se as linhas a seguir estão presentes:

- ***ServerName "Nome para o seu servidor"*** → É o nome do seu servidor. Ex.: Servidor FTP
- ***ServerIdent on "Mensagem para os usuarios"*** → É a mensagem que é mostrada quando um usuário vai conectar no servidor.
- ***ServerAdmin*** root@localhost → É o e-mail do administrador do servidor.
- ***ServerType standalone*** → É a forma que o ProFTPD vai trabalhar.
- ***DefaultRoot ~*** → É o diretório onde o usuário do FTP vai ter acesso.
- ***RequireValidShell off*** → Diz se o usuário precisa ter um shell válido. Ex.: bash, sh, csh...

Com o arquivo configurado, vamos criar um usuário chamado **usuario_ftp** que será utilizado especificamente para acessar o FTP:

```
adduser usuario_ftp
```

A senha para o usuário será pedida. Caso isso não acontece, o comando abaixo permitirá a inserção de uma senha para o usuário recém-criado

```
passwd usuario_ftp
```

Insira a senha ***userftp***

Agora reiniciamos o ProFTPd com o comando:

```
/etc/init.d/proftpd restart
```

Se a configuração foi realizada corretamente será possível acessar os arquivos que estão na pasta do usuario_ftp utilizando um navegador, através do endereço **ftp://endereco_do_servidor**, no nosso caso **ftp://192.168.100.1**

4.2 Transferindo um arquivo via linha de comando:

Para transferirmos arquivos para a pasta que criamos no servidor de ftp utilizaremos o programa (que conincidentemente se chama) ***ftp***
Na linha de comando digite

```
ftp
```

Se o programa estiver instalado no computador (geralmente está) será exibido

```
ftp>
```

A partir deste momento devemos criar uma conexão com o servidor ftp, utilizando o comando **`open [host] [porta]`**, onde [host] é o endereço IP da máquina que possui o servidor FTP e [porta] é a porta em que o servidor FTP foi configurado. No nosso exemplo teremos o comando da seguinte forma (acessando pela própria máquina):

```
ftp> open 192.168.100.1 21
```

Para inserir um arquivo na pasta do ftp utilize o comando

```
put arquivo_origem arquivo_destino
```

Ex: **`put ~/Documentos/teste.txt teste.txt`**

Para listar os arquivos que constam no FTP utilizar o comando **`ls .`**

Para fazer download de um arquivo via linha de comando, utilizar o comando get:

Ex: **`get teste.txt`**

Você pode organizar o seu FTP criando pastas, através do comando **mkdir**:

Ex: **`mkdir aula_ftp`**

Exercícios 2.1

1. No servidor FTP recém-criado, crie uma pasta chamada ***exemplos***
2. Crie na sua pasta *Downloads* um arquivo chamado *exemplo1.txt* e escreva algum conteúdo nele.
3. Transfira o arquivo *exemplo1.txt* para a pasta *exemplos*, de forma que o nome do arquivo permaneça o mesmo.
4. Agora crie na sua pasta *Downloads* um arquivo chamado *exemplos2.txt* e transfira este arquivo para a pasta *exemplos*, porém o arquivo deve ser chamado de *exemplos_dois.txt*

5. Descubra qual o endereço IP da sua máquina e informe-o ao colega do lado para que ele acesse a sua pasta do FTP via navegador. Não esqueça de passar o usuário e senha recém-criados também.

6. Crie um novo usuário com o seu primeiro e último nome, e crie uma senha para o seu usuário.

7. Acesse o ftp pelo navegador e utilize as credenciais que você acabou de criar. O que é possível observar ao acessar o FTP com esta nova credencial?

4.3 Configurando o acesso dos usuários ao serviço FTP

Você deseja criar um novo usuário e quer que ele tenha acesso à mesma pasta de usuario_ftp. Para isso devemos criar o novo usuário da seguinte forma:

```
adduser --home /home/usuario_ftp meu_usuario
```

Para restringir o acesso ao serviço de ftp a apenas alguns usuários específicos, teremos de editar o arquivo ***proftpd.conf***. Devemos adicionar as linhas a seguir (caso elas já existam é só modificá-las. *Não esqueça de fazer uma cópia do arquivo antes de modificá-lo*). Acrescentaremos as linhas a seguir para dizer ao proftpd que apenas *usuario_ftp* e *meu_usuario* terão acesso ao serviço FTP:

```
<Limit LOGIN>
AllowUser usuario_ftp
AllowUser meu_usuario
DenyALL
</Limit>
```

Vamos permitir que *usuario_ftp* possa acessar toda a pasta raiz, enquanto que *meu_usuario* possa apenas acessar a pasta *exemplos.* Para isso vamos modificar, novamente, o arquivo *propftpd.conf*:

```
<Directory /home/usuario_ftp/*>
Umask 022 022
AllowOverwrite on
     <Limit ALL>
          Order Allow,Deny
          AllowUser usuario_ftp
          Deny ALL
     </Limit>
</Directory>

<Directory /home/usuario_ftp/exemplos/>
Umas 022 022
AllowOverwrite on
     <Limit ALL>
          Order Allow,Deny
```

```
            AllowUser usuario_ftp
            AllowUser meu_usuario
            Deny ALL
        </Limit>
    </Directory>
```

4.4 FTP Seguro

O protocolo FTP é intrinsecamente inseguro. Em servidores que terão acesso externo, o recomendável é que se impelente um FTP seguro, conhecido como SFTP (SSH File Transfer Protocol).

Para implementarmos o SFTP precisamos que o serviço SSH esteja instalado e rodando no servidor. Caso este não esteja instalado, procederemos com o comando

```
apt-get install ssh
```

Instalando e Configurando um cliente FTP com interface gráfica (Windows/Linux)

Até o momento utilizamos comandos no terminal para manipular as pastas e arquivos do ftp. Estas tarefas podem ser facilitadas com a utilização de um programa cliente ftp com interface gráfica. Temos vários que podem ser baixados e instalados de graça: FileZilla (funciona no Linux e no Windows), getFTP, Kasablanca, WinSCP(funciona apenas no Windows).

5. Implantando um sistema Java Web no Apache Tomcat (Deploy)

Há diversas formas de se implantar um sistema Java Web em um servidor Apache Tomcat. Antes de saber quais são as formas, devemos entender como funcionam alguns parâmetros de configuração do servidor Tomcat, os quais são setados no arquivo server.xml.

No arquivo server.xml há uma diretiva <Host> que trata de opções do host do servidor onde o tomcat se encontra. Nessa diretiva há por padrão os parâmetros:

name → nome do servidor tomcat

appBase → diretório onde os sistemas web serão implantados e a partir de onde eles serão lidos

unpackWars → se estiver setado como **true**, indica que um arquivo .war será descompactado assim que for inserido no appBase

autoDeploy → se estiver setado como **true**, um arquivo .war será lido e implantado de forma automática, enquanto o tomcat estiver rodando.

Teste 1:

1. Copie um arquivo **. war** para a pasta webapps e veja o que acontece.
2. Agora apague a pasta criada. O que aconteceu?
3. E o que acontece se apagarmos o arquivo .war?

Teste 2:

1. Altere o parâmetro autoDeploy para false.
2. Copie um arquivo **. war** para a pasta webapps e veja o que acontece.
3. Reinicie o tomcat. O que acontece?
4. E o que acontece se apagarmos o arquivo .war?

Teste 3:

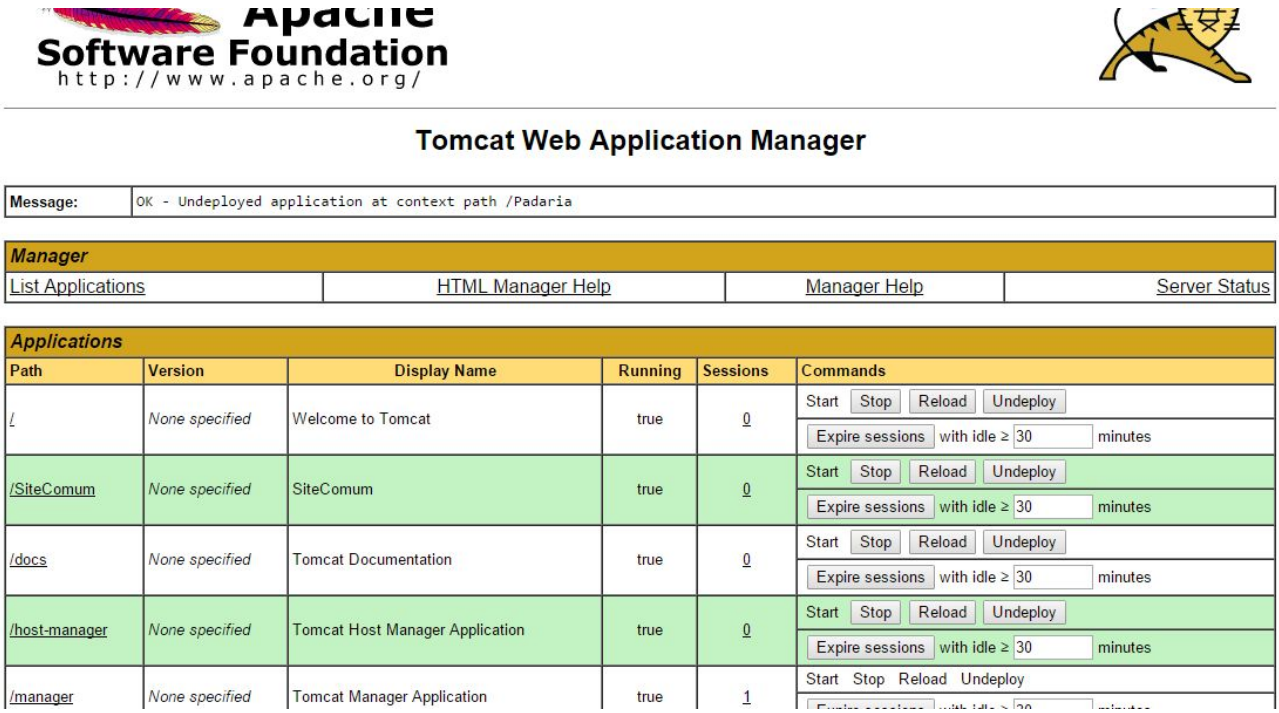
1. Altere o parâmetro unpackWars para false.
2. Copie um arquivo **. war** para a pasta webapps e veja o que acontece.
3. Reinicie o tomcat. O que acontece?
4. E o que acontece se apagarmos o arquivo .war?

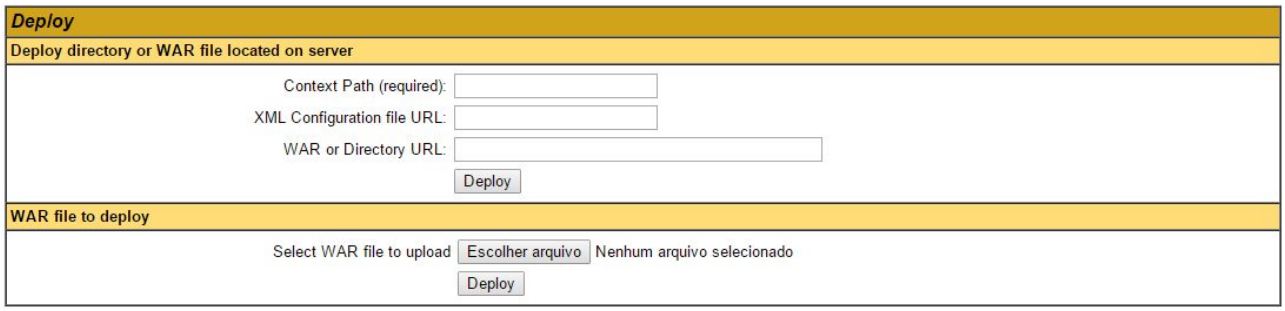
Para os tópicos a seguir vamos considerar o unpackWars sempre como true.

5.1 Realizar deploy copiando os diretórios do sistema ou por arquivo .war

Forma mais simples de realizar deploy de um sistema. Só será possível caso o implantador tenha acesso ao diretório do appBase. Se a opção ***autoDeploy*** for **true**, o tomcat implantará o serviço (descompactará o arquivo .war, caso este tenha sido enviado). Para saber se um serviço foi implantado, devemos acessar o Manager App do Tomcat e verificar se este faz parte da lista **Applications**.

Caso o ***autoDeploy*** esteja desabilitado (**false**), não é viável reiniciar o Tomcar apenas para que o novo sistema seja implantado, pois outros sistemas podem estar configurando um mesmo servidor. Procedemos, então das maneiras a seguir, dependendo do que foi enviado ao servidor. Nos dois casos utilizaremos a área do Manager App chamada ***Deploy*** e o formulário **Deploy directory or WAR file locate on server**. Este formulário serve para realizarmos a implantação de diretório ou arquivo WAR que já esteja localizado no servidor (sem necessidade de upload).

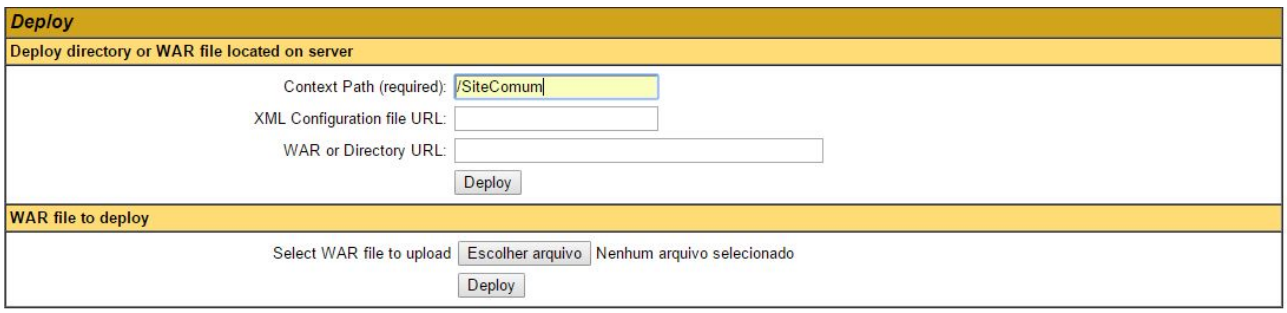




.

5.1.1 Copiando os diretórios e arquivos já descompactados no servidor

É necessário informar o Context Path que nada mais é do que o nome do diretório em que os arquivos do sistema ficarão armazenados. No exemplo, é SiteComum, no qual devemos colocar uma “/” antes.



Caso o diretório ou WAR já esteja na pasta webapps não será necessário preencher os outros dois campos. Lembrando que o diretório ou o arquivo WAR devem ter o mesmo nome do Context Path.

Caso o diretório ou o WAR esteja em outra pasta do servidor deve-se preencher o campo WAR or Directory URL com o caminho onde ele se encontra. Como exemplo, temos que o diretório com todos os arquivos de SiteComum se chama SiteComum e se encontra na Área de Trabalho, logo teremos que passar o caminho **file:/home/usuario/Desktop/SiteComum**. Em outro exemplo temos que o arquivo SiteComum.war está na pasta Documentos, logo temos de passar **file:/home/usuario/Documentos/SiteComum.war**.

5.2 Implantando um sistema/site remotamente, através de arquivo WAR

Para implantar um sistema/site de forma remota, continuaremos na seção Deploy do Manager App, porém utilizaremos o formulários **WAR file to deploy**. Neste caso, apenas faremos uma busca no nosso próprio computador (onde deve estar o arquivo WAR) e logo após pressionar o borão Deploy.

5.2 Atualizando o sistema/site

Bibliografia

1: W3C, W3C Working Group Note 11 February 2004, 2004, http://www.w3.org/TR/ws-arch/#introduction
2: DAMASCENO, J. C., Introdução à Composição de Serviços Web, , http://www.ufpi.br/subsiteFiles/ercemapi/arquivos/files/minicurso/mc8.pdf
3: MuleSoft https://www.mulesoft.com/tcat/tomcat-deploy-procedures
4: Apache Foudation http://apache.org/