

Iniciando em Java

Uma Abordagem Open Source



Vanessa C. Sabino

O que é Java?



- Um conjunto de especificações
 - Linguagem de programação
 - Formato do arquivo .class (bytecode)
 - APIs
 - Máquina virtual e outras ferramentas

Características

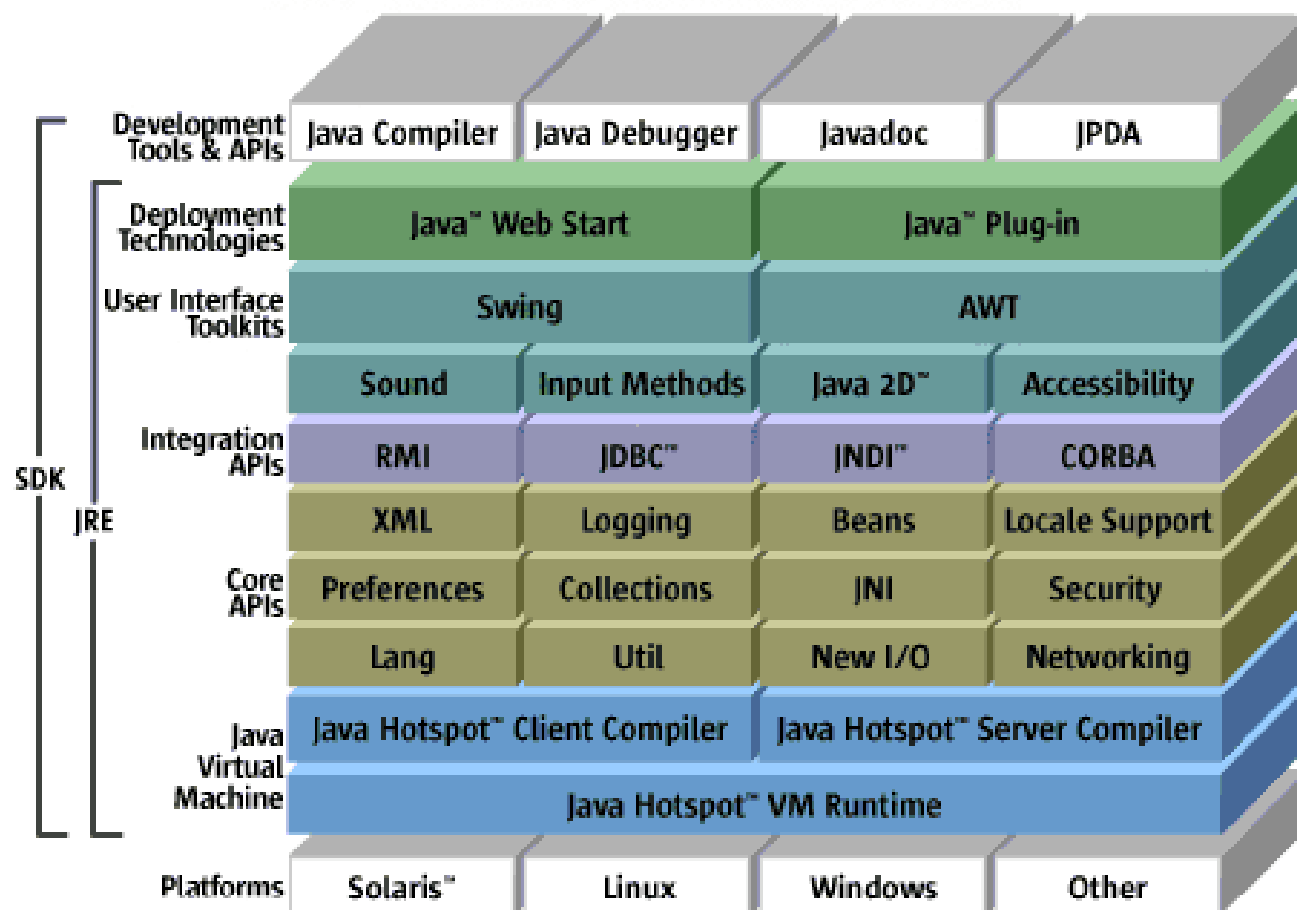


- Independente de plataforma
 - Bytecode: interpretado ou compilado JIT
- Orientação a objetos
- Suporte a multithreading
- Garbage Collector
- Mecanismo de Exceções
- Segurança através do controle de recursos
- Ausência de aritmética de ponteiros

J2SE



Java™ 2 Platform, Standard Edition v 1.4



Histórico do Java



- 1991: início do *Green Team*
 - Convergência entre aparelhos comuns controlados digitalmente e computadores
- 1992: *7 e a linguagem “Oak”
 - Green Team ⇒ FirstPerson
 - Objetivo: revolucionar a indústria de TV e vídeo oferecendo mais interatividade



Histórico (cont.)



- 1994: Criado o WebRunner (HotJava) como um clone do Mosaic (lançado em 1993)
- 1995: Código fonte da versão 1.0a2 liberada na Internet
- 1996: Liberado o JSDK 1.0
- 1997: Java incluído no Netscape e adoção do Personal Java em dispositivos embarcados
- 1998: Primeiro JavaCard (VISA)
- 1998: Formalização do JCP
- 1999: Divisão do Java: J2SE, J2EE e J2ME

Java e Software Livre



- Independência de fornecedor
- Suporte multiplataforma
- Padrões abertos e de origens externas
- Desenvolvimento cooperativo e colaborativo

Java – Instalação



- java.sun.com/j2se/1.4.2/download.html
- Download J2SE v 1.4.2_04 (SDK)
- Linux Platform – self-extracting file
- Variáveis de ambiente:
 - `export JAVA_HOME=/usr/local/j2sdk<versao>`
 - `export PATH=$PATH:$JAVA_HOME/bin`
 - `export CLASSPATH=.:$CLASSPATH` *(se já existir)*

Programa simples



```
public class LinuxChix {  
    public static void main(String[] args) {  
        System.out.println("LinuxChix r0x");  
    }  
}
```

- Salvar em um arquivo chamado **LinuxChix.java**
- Compilar com o comando `javac LinuxChix.java`
- Executar com o comando `java LinuxChix`

IDEs



- Editores de texto simples: vi, emacs, gedit, kwrite

 Jedit ⇒ www.jedit.org

- Apenas 2Mb

 NetBeans ⇒ www.netbeans.org

- IDE oficial

 Eclipse ⇒ www.eclipse.org

- O mais completo e versátil
- Suse, RedHat, Borland, SAP, Intel, Ericsson...

- Modelagem: ArgoUML ⇒ argouml.tigris.org

Sistemas Web



- J2SE: pacotes java.io, java.nio, java.net...
- J2EE: pacote javax.servlet
 - Componente Web
 - O papel do container Web no Servidor Web



- Tomcat ⇒ jakarta.apache.org/tomcat 

Exemplo: Servlet



```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Exemplo extends HttpServlet {

    public void doGet(HttpServletRequest req,
        HttpServletResponse res) throws
        ServletException, IOException {

        PrintWriter pw = res.getWriter();
        pw.println( "<HTML><HEAD>" );
        ...
    }
}
```

Exemplo: JSP



```
<HTML><BODY>
```

```
<%@ page language="java" %>
```

```
<%! int counter = 0; %>
```

```
<% counter++; %>
```

Bem vindo, você é o visitante número

```
<%= counter %>
```

```
</BODY></HTML>
```

Custom Tags



- Facilitam a interação com o designer
- Evitam os problemas de scriptlets:
 - Mistura de linguagens
 - Mistura de dados com a lógica
 - Baixa reusabilidade
 - Difíceis de parametrizar

Exemplo - antes



```
<%@ page import="java.text.SimpleDateFormat" %>
<%@ page import="java.util.Date" %>
<HTML>
<HEAD><TITLE>Exemplo de Scriptlet</TITLE></HEAD>
<BODY>
<H3>
A hora e data no servidor são:
<%
String sformat = "EEEE, d MMMM yyyy 'at' kk:mm:ss z";
SimpleDateFormat format = new SimpleDateFormat(sformat);
Date date = new Date();
String sdate = format.format(date);
out.print(sdate);
%>
</H3>
</BODY>
</HTML>
```

Exemplo – depois



```
<%@ taglib uri="/WEB-INF/taglib.tld" prefix="mytags" %>
```

```
<HTML>
```

```
<HEAD><TITLE>exemplo custom tag</TITLE></HEAD>
```

```
<BODY>
```

```
<H3>
```

```
A hora e data no servidor são: <mytags:date/>
```


```
</H3>
```

```
</BODY>
```

```
</HTML>
```


Taglibs



- Taglibs ⇒ jakarta.apache.org/taglibs 
- JSTL ⇒ java.sun.com/products/jsp/jstl
 - Parte da especificação do JSP 2.0
 - Core
 - Format
 - SQL
 - XML
 - Expression Language

Template Engines



- Melhor separação entre a lógica e a apresentação do sistema

- **Velocity** → jakarta.apache.org/velocity 

- Múltiplos formatos (HTML, WML, etc...)

- Caching

- Tutoriais:

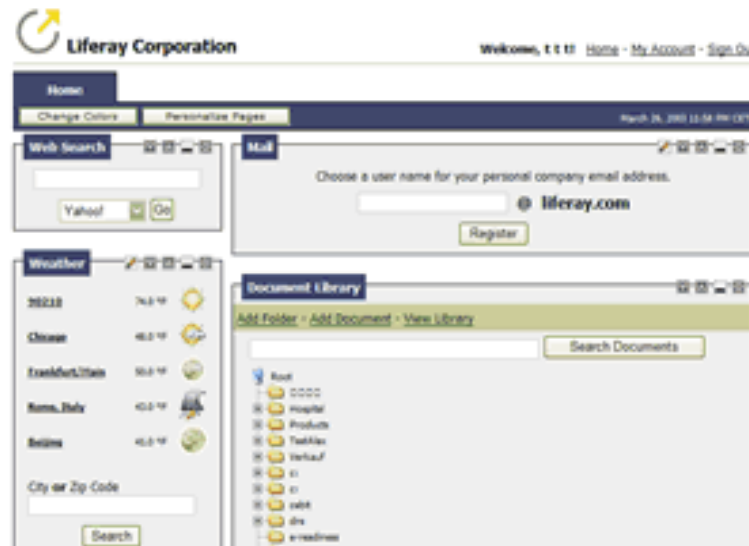
- guj.com.br/user.article.get.chain?page=1&article.id=18
 - guj.com.br/user.article.get.chain?page=1&article.id=26

Ex.: Velocity



```
## toolview.vm
#set ($title = "Tool Listing")
#set ($deck = "A list of content creation tools")
#set ($desc = "Without tools, people are nothing more than
    animals." )
$toolbean.setToolsFile($application.getInitParameter
    ("toolsFile"))
#set ($tools = $toolbean.getTools($request.getParameter
    ("state")))
#foreach ($tool in $tools)
    <HR SIZE=2 ALIGN=LEFT>
    <H3>
    $tool.Name
    </H3>
    <A HREF="$tool.homeURL">$tool.homeURL</A><BR>
    $tool.comments
#end
```

Portais



- Nova especificação de Portlets

 Pluto ⇒ jakarta.apache.org/pluto 

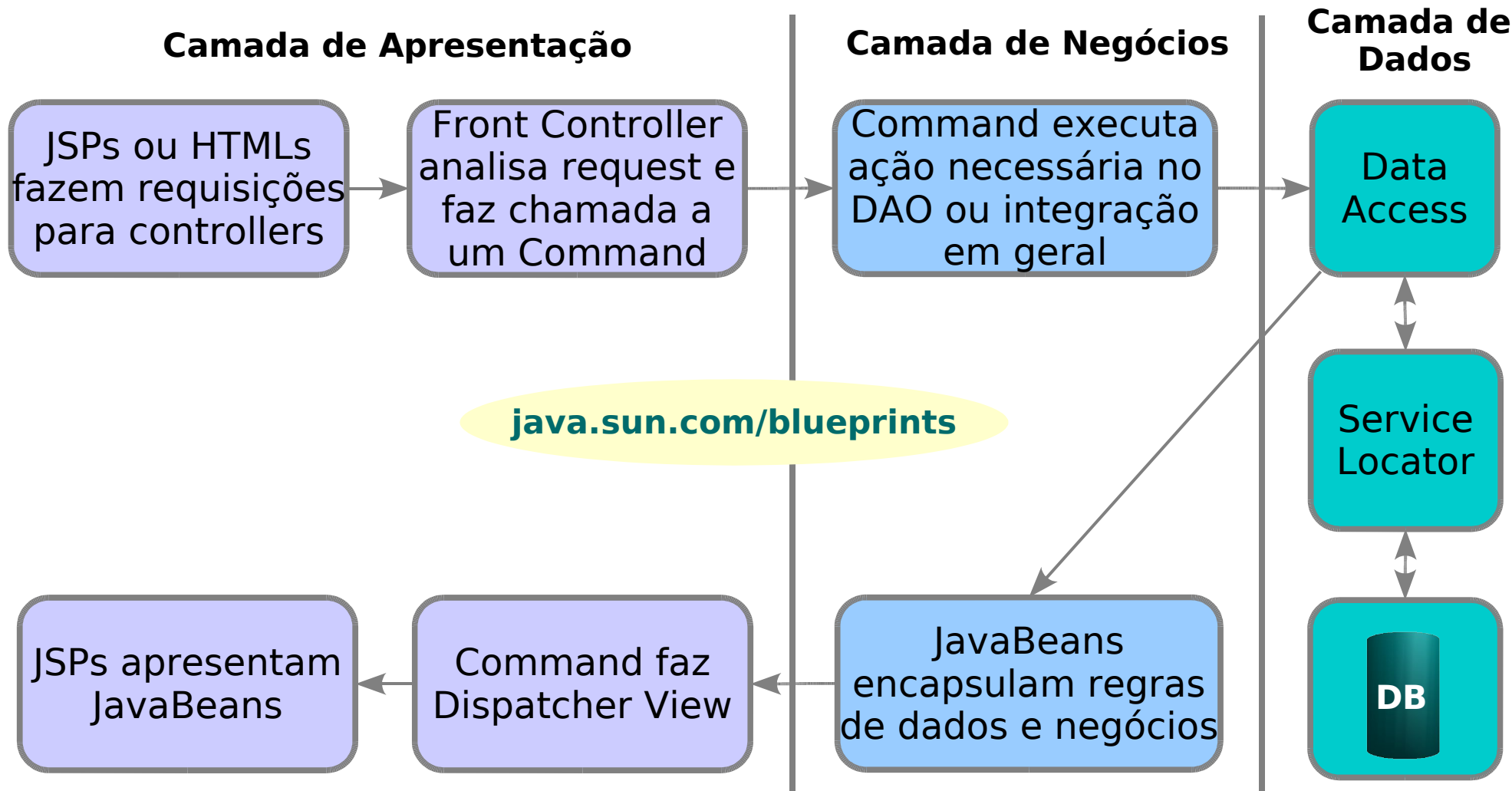
- Jetspeed, jPorta, Liferay, CCM RedHat
- javaboutique.internet.com/reviews/Enterprise_Portals



Relatórios / Gráficos

- JasperReports ⇒ jasperreports.sourceforge.net
 - iReports ⇒ ireport.sourceforge.net 
- Ferramentas para fazer gráficos
 - JFreeChart ⇒ jfree.org/jfreechart
 - Cewolf ⇒ cewolf.sourceforge.net
 - Chart2D ⇒ chart2d.sourceforge.net


Design Patterns



Fonte: GlobalCode




Frameworks MVC



- Conjunto de classes e interfaces que abstraem conceitos e agregam valor ao desenvolvimento
- Automatizam o fluxo de controle da aplicação
- Struts
 - jakarta.apache.org/struts 
 - Apostila: tinyurl.com/2w5m5
- WebWork
 - www.opensymphony.com/webwork
 - WebWork vs. Struts: tinyurl.com/36cha



Persistência de dados

- Drivers JDBC (MySQL, PostgreSQL, etc)
- Mapeamento Objeto-Relacional
 - OJB ⇒ db.apache.org/ojb 
 - Hibernate ⇒ www.hibernate.org 
 - Introdução ao Hibernate: tinyurl.com/2u4cp
- Prevaler ⇒ www.prevaler.org 
 - 15 min. de Prevaler: tinyurl.com/2887g
- HSQLDB ⇒ hsqldb.sourceforge.net



Hibernate – Exemplo

```
<hibernate-mapping>
  <class name="org.LinuxChix" table="linuxchix">
    <id name="id" type="integer" column="id">
      <generator class="identity"/>
    </id>
    <property name="nome" column="NOME" not-
null="true" unique="true" />
  </class>
</hibernate-mapping>
```

EJBs



- Componentes do lado do servidor que encapsulam a lógica de negócios
- Entity, Session e Message-driven beans
- Application server / Contêiner J2EE
 - gerenciamento de transações, segurança, etc
- JBoss ⇒ jboss.org
- XDoclet ⇒ xdoclet.sourceforge.net
 - *Attribute-oriented programming*
- Geronimo ⇒ incubator.apache.org/projects/geronimo



Testes Unitários

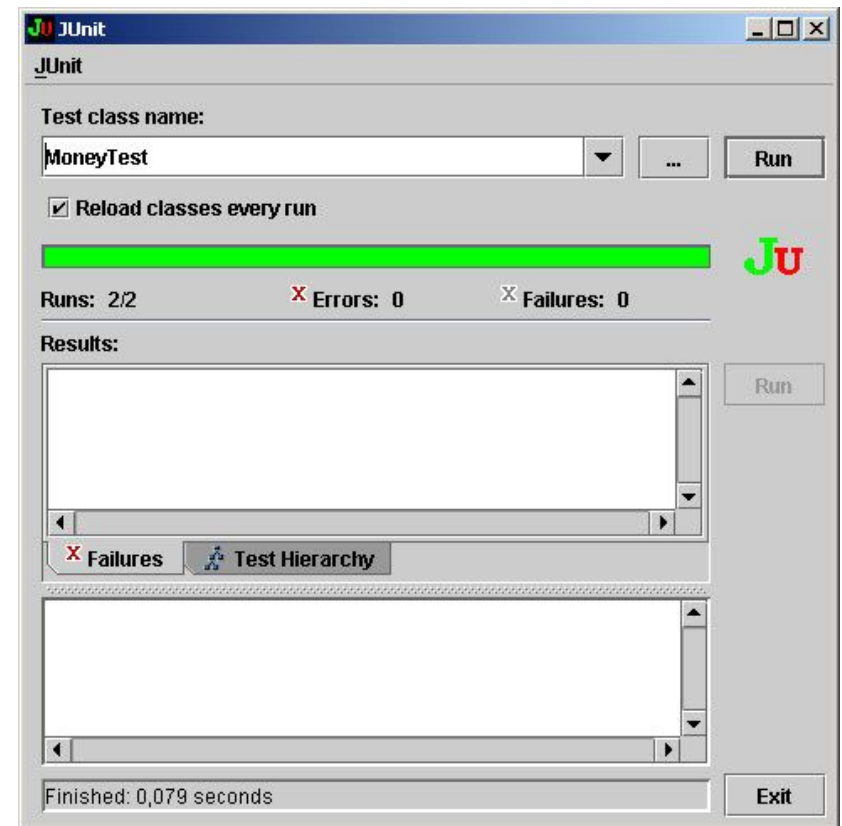


- Test-driven development (XP)

- JUnit → www.junit.org




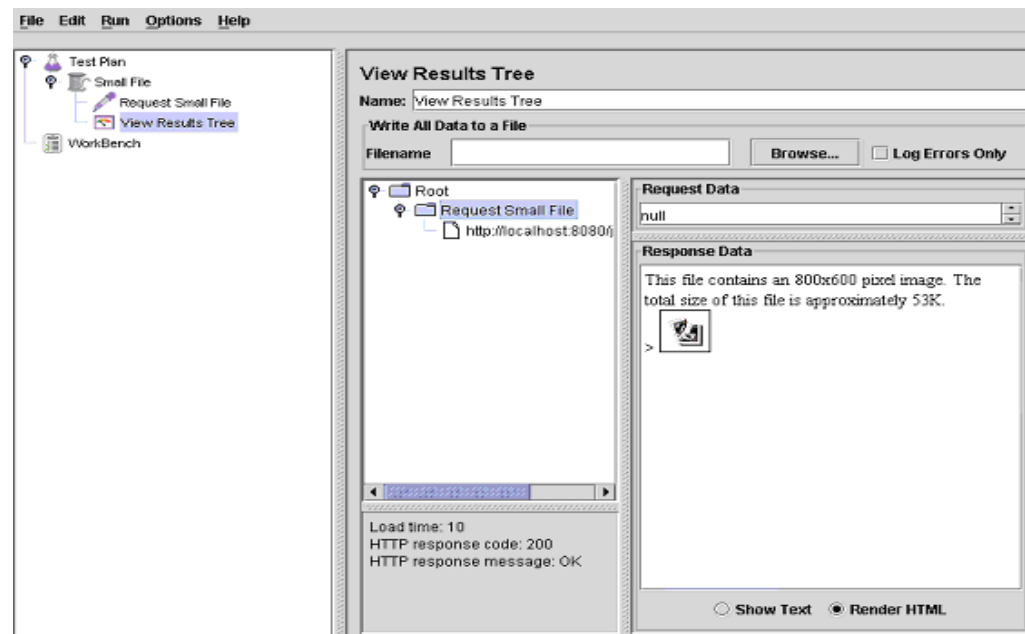
- Test Cases
- Test Suites



Testes de Carga




- Avaliar performance
- JMeter ⇒ jakarta.apache.org/jmeter 
- Plugins de análise de dados e visualização







Ant



- Ferramenta para automatização
 - processos de construção, testes e distribuição
- ant.apache.org 
- Vantagens em relação ao make:
 - Mais multiplataforma
 - Configuração mais clara e organizada
- Tasks (classes) e XML
- Tutoriais: tinyurl.com/2a2de e tinyurl.com/yw4aq


Mais Jakarta



- Jakarta Commons
 - Componentes Java reutilizáveis
 - BeanUtils, Collections, DbUtils, Digester, EL, FileUpload, HttpClient, IO, Lang, Launcher, Logging, Math, Net, Pool Validator e **muito** mais...
 - jakarta.apache.org/commons 
 - onjava.com/pub/a/onjava/2003/06/25/commons.html
- POI ⇒ jakarta.apache.org/poi 
-  ⇒ jakarta.apache.org/lucene 
- FOP ⇒ xml.apache.org/fop

Tendências



- IoC / Dependency Injection
 - Container fornece aos componentes os serviços necessários para serem instanciados
 - Pattern *Separation of Concerns*
 - *Component Oriented Programming*
 - PicoContainer ⇒ picocontainer.codehaus.org
 - NanoContainer ⇒ nanocontainer.codehaus.org
 - Avalon ⇒ avalon.apache.org 



Tendências



- Aspect-Oriented Programming
 - Módulos reusáveis
 - Encapsulam comportamento
 - Pointcuts
 - AspectWerkz ⇒ aspectwerkz.codehaus.org
 - AspectJ ⇒ eclipse.org/aspectj
 - Nanning ⇒ nanning.codehaus.org
 - Tutorial: www-106.ibm.com/developerworks/java/library/j-aspectj/?dwzone=java

Dicas para Criar Projetos Open Source Java



- Maven
 - maven.apache.org 
 - Gerenciamento de projetos Open Source
- Java.net
 - Portal para desenvolvedores Java
 - Incubadora de projetos
 - Infra-estrutura para JUGs
 - etc...
 - linux.java.net 



Java Open Source



- A licença do Java
 - Importância dos padrões
 - JCP ⇒ jcp.org
 - Sun Community Source Licensing
 - Binary Code License Agreement
- Alternativas Open Source
 - Kaffe ⇒ kaffe.org
 - GCJ ⇒ gcc.gnu.org/java

Para lembrar

(ou para quem dormiu e está acordando agora)



- Um dos benefícios da Orientação a Objetos é a reusabilidade do código
- A comunidade Java é uma das mais ativas e organizadas
- Projetos, exemplos de código e livros gratuitos estão disponíveis na internet
- Portanto, antes de reinventar a roda,

pesquise!



The **Apache Jakarta Project**

<http://jakarta.apache.org/>

Google™

Perguntas?

E no caso de mais perguntas futuramente...



<http://www.guj.com.br/forum>

Slides disponíveis em <http://alemdojava.cjb.net/>