Lição 2

Classes Servlets

Objetivos

Ao final desta lição, o estudante será capaz de:

- Obter uma visão geral da arquitetura *Servlet*
- Conhecer o ciclo de vida de uma *Servlet*
- Manipular requisições e respostas
- Configurar, empacotar e distribuir uma aplicação WEB
- Conhecer os parâmetros de aplicações WEB

Servlet

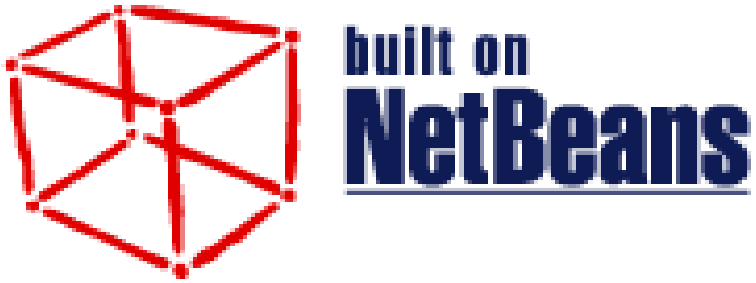
- Para programar:
  - Programação Java
  - Conceitos Cliente/Servidor
  - HTML e HTTP
- Para criar:
  - importar as classes padrão de extensão do pacote javax.servlet e javax.servlet.http.
  - javax.servlet
  - javax.servlet.http

Visão geral da arquitetura Servlet

- Common Gateway Interface (CGI)
- Servlets são projetadas para fornecer aos desenvolvedores uma solução Java para criar aplicações WEB
- Existe somente um processo que gerencia as requisições
- Servlets são carregadas na memória somente uma vez

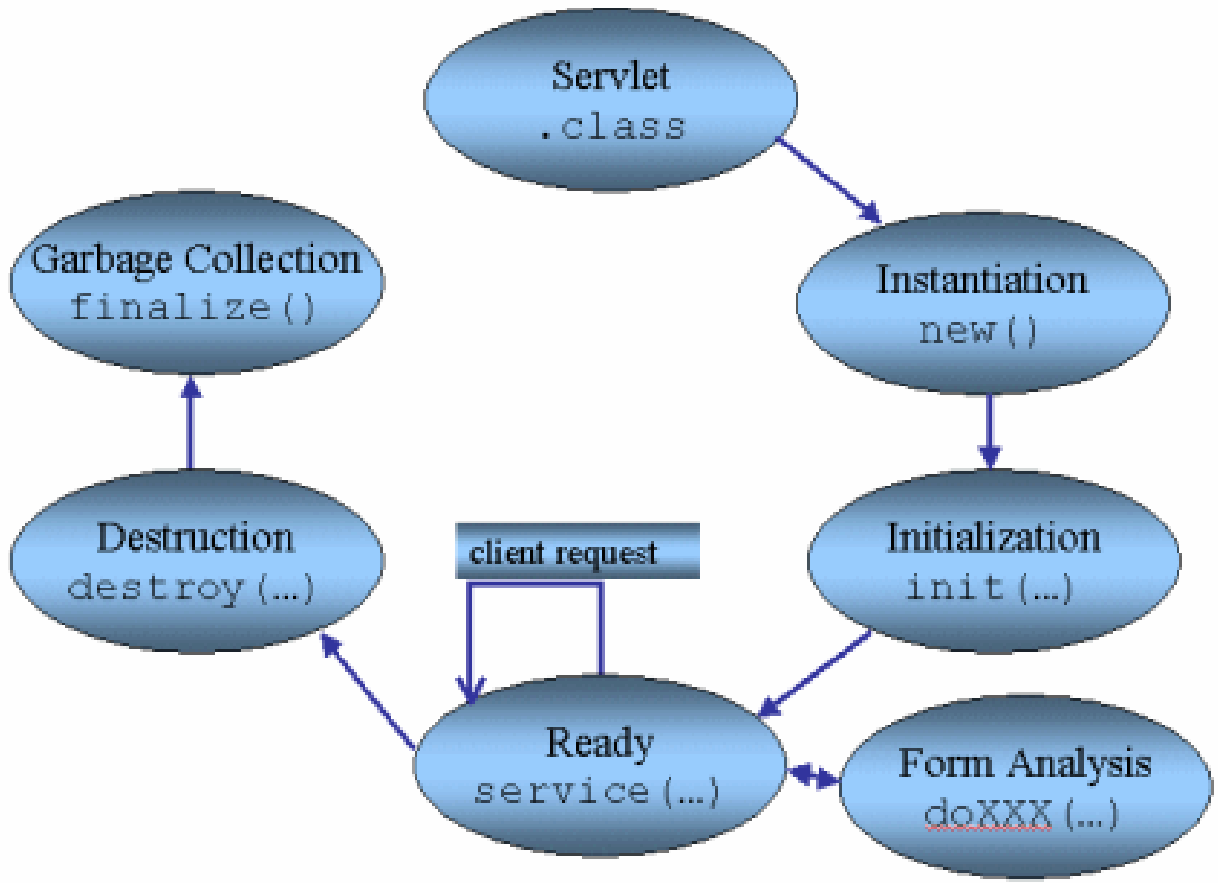
Primeira vista sobre Servlet

- Passaremos agora para o NetBeans



Ciclo de vida da Servlet

- Controlado pelo *container* em que a *Servlet* foi implementado
- Permite que o contêiner resolva os problemas



Ciclo de vida da Servlet: Instanciação

- A classe da servlet é carregada na memória, e uma instancia é criada pelo contêiner
- Pelo padrão, um contêiner realiza o que é chamado *lazy loading*
- *Servlet* atravessa essa fase somente uma vez na vida
- Chamado o método **construtor** sem argumento

Ciclo de vida da Servlet: Inicialização

- *Servlet* é preparada para uso na aplicação
- *Servlet* atravessa também este estágio somente uma vez
- Após esta fase a instância da classe começa a ser chamada de *Servlet*
- Chamado o método **init()**

```
public void init(ServletConfig config)
```

Ciclo de vida da Servlet: Pronta

- A fase em que uma servlet se encontra ao longo de sua vida
- A servlet pode repetidamente ser chamada pelo *container* para fornecer sua funcionalidade
- Chamado o método **service()**

```
public void service(ServletRequest req,
                    ServletResponse res)
```

Ciclo de vida da Servlet: Destruição

- Quando uma servlet deve ser removida de um contêiner
- Desenvolvedores não podem interferir
- Chamado o método **destroy()**

Ciclo de vida da Servlet: Garbage Collection

- Fase no ciclo de vida da *Servlet* equivalente a qualquer outro objeto Java
- Ocorre imediatamente antes que uma instancia do objeto seja removido da memória
- Desenvolvedores não têm nenhum controle direto
- Chamado o método **finalize()**

Gerenciando Requisições e Respostas

- A *servlet* dispõe de um objeto *ServletRequest*
- Servlet possuem uma subclasse chamada HTTPServletRequest
- Fornece métodos adicionais para recuperar informações específicas para HTTP
- Informação de *cookie*, os detalhes de cabeçalho

Gerenciando Requisições e Respostas

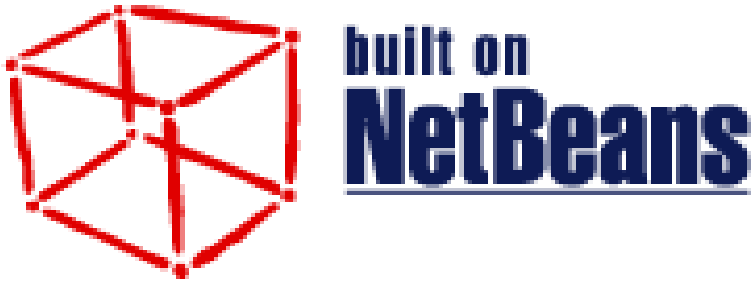
- Dados de Formulário e Argumentos
- Recuperando Informações da URL da Requisição
- Informações de Cabeçalho
- Geração da saída

Dados de Formulário e Argumentos

- request.getParameter
- request.getParameterValues
- request.getParameterNames

request.getParameter

- Passaremos agora para o NetBeans



request.getParameter

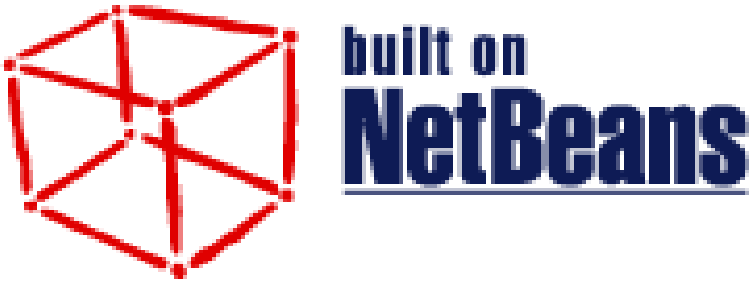
- Para este e outros cenários similares, Java fornece o método getParameter no objeto HttpServletRequest

```
public String getParameter(String parameterName)
```

- Agora, vamos criar o código que recebe o nome do usuário, devolve um cumprimento simples e o HTML usado para visualizar o formulário

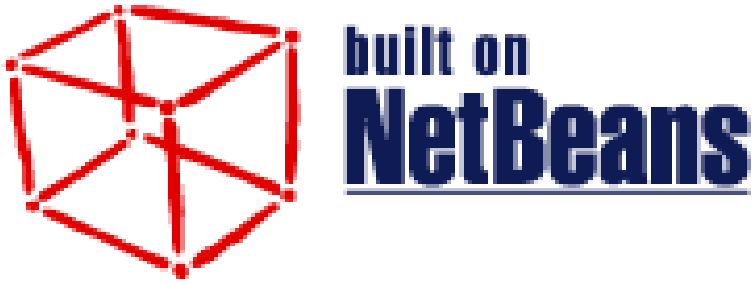
request.getParameterValues

- Passaremos agora para o NetBeans



request.getParameterNames

- Passaremos agora para o NetBeans



Recuperando informações de uma requisição URL

```
http://[host]:[port]/[requestPath]?[queryString]
```

- Host – request.getServerName()
- Port – request.getServerPort()
- Request Path – em Java é dividido em 2 componentes lógicos:
  - Context – O contexto da aplicação web. Pode ser recuperado invocando o método request.getContextPath()
  - Path info – tO restante da requisição, após o nome do contexto Pode ser recuperada invocando o método request.getPathInfo()
- Query String – request.getQueryString()

Recuperando informações de uma requisição URL

- Para a seguinte URL:

  http://www.myjedi.net:8080/HelloApp/greetUser?name=Jedi

- Temos o resultado de cada invocação dos métodos citados:

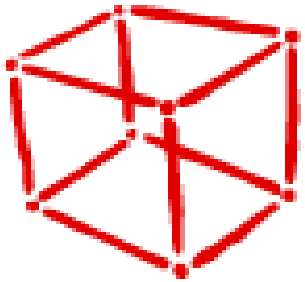
request.getServerName()	***www.myjedi.net***
request.getServerPort()	***8080***
request.getContextPath()	***HelloApp***
request.getPathInfo()	***greetUser***
request.getQueryString()	***name=Jedi***

Informações de Cabeçalho

- Pode ser recuperado a partir do servlet, chamando os seguintes métodos do HttpServletRequest:
  - getHeader(String name)
  - getHeaders(String name)
  - getHeaderNames()
  - getIntHeader(String name)
  - getDateHeader(String name)

Geração de Saídas

- Passaremos agora para o NetBeans



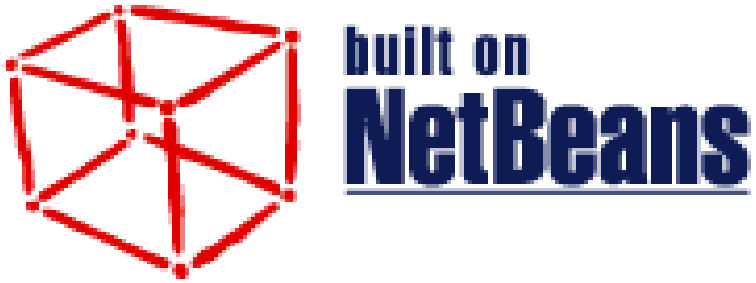
built on NetBeans

Configuração de uma Aplicação Web

- O servlet define um arquivo XML chamado de web.xml que age como um arquivo de configuração para as aplicações web
- Este arquivo é também chamado de descriptor de deploy
- Será utlizado o primeiro exemplo FirstServlet como um ponto de partida para melhor entender a estrutura do web.xml

Configuração de uma Aplicação Web

- Passaremos agora para o NetBeans



web.xml

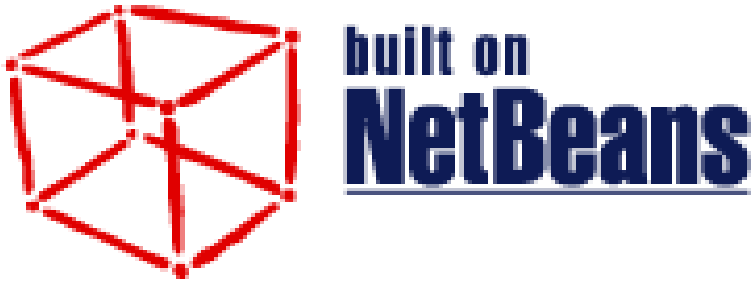
- <web-app>
- <servlet>
- <servlet-mapping>
- <session-config>
- <welcome-file-list>

Empacotando uma Aplicação WEB

- A Aplicação pode ser posta no container web por meio de um único arquivo chamado WAR
- Arquivos WAR são os mesmos de arquivos JARs: eles simplesmente contem códigos java comprimidos utilizando um formato ZIP
- Informalmente, WAR significa Arquivo Web

Gerando arquivos WAR

- Passaremos agora para o NetBeans



Parâmetros do Servlet e Aplicação

- Inicialização dos parâmetros do ServletConfig e do Servlet
- Parâmetros da Aplicação e do ServletContext

ServletConfig e Servlet

ServletConfig

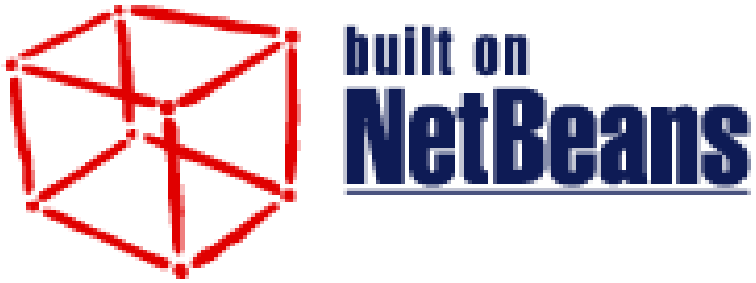
- Passado para um servlet especifico durante sua fase de inicialização.
- Fazendo uso disto, o servlet pode:
  - Recuperar informações especificas para ele mesmo, tais como parâmetros de inicialização
  - Ganhar acesso a uma instancia do objeto de *ServletContext*

- Parâmetros de inicialização são ótimos para:

  - Quando se lida com informações que pode variar com cada *deploy* realizado na aplicação
  - Permite que comportamentos do servlet sejam modificados sem que seja necessário a recompilação do código

ServletConfig e Servlet

- Passaremos agora para o NetBeans



Sumário

- Servlets
- Ciclo de Vida da Servlet
- Gerenciando Requisições e Respostas
- Dados de Formulário e Argumentos
- request.getParameter, request.getParameterValues, request.getParameterNames
- Recuperando informações de uma requisição URL
- Configuração de uma Aplicação WEB
- Empacotando uma aplicação
- Parâmetros do Servlet e Aplicação

Parceiros

- **Os seguintes parceiros tornaram JEDI™ possível em Língua Portuguesa:**